SÄHKÖISEN KAUPPAPAIKAN TURVALLISUUS

Mika Valonen

27.11.2005

Joensuun yliopisto
Tietojenkäsittelytiede
Pro gradu –tutkielma

Tiivistelmä

Internetissä toimivaan sähköiseen kauppapaikkaan kohdistuu useita turvallisuusuhkia, jotka voivat aiheuttaa taloudellisia menetyksiä kauppapaikan omistajalle ja asiakkaille. Tässä tutkielmassa käsitellään sähköisen kauppapaikan turvallisuuteen vaikuttavia tekijöitä ja niiden toteutusta. Tärkeimpinä turvallisuutta parantavina tekijöinä esitellään turvallinen tiedonsiirto avoimessa tietoverkossa, käyttäjän luotettava tunnistaminen sekä erilaisia tapoja maksun suorittamiseen sähköisessä kauppapaikassa. Turvallisuustason parantaminen vaatii kuitenkin veronsa, sillä se nostaa kauppapaikan kustannuksia ja heikentää käytettävyyttä. Tästä syystä palvelun pystyttäjän tulee tehdä valintoja, joilla kauppapaikkaan saadaan riittävä turvallisuus kohtuullisilla kustannuksilla.

ACM-luokat (ACM Computing Classification System, 1998 version): C.2.2, E.3

Avainsanat: Sähköinen kauppapaikka, turvallisuus, tiedonsalaus, käyttäjän tunnistus

Sisällysluettelo

1 JOHDANTO

Erilaisten sovellusten määrä Internetissä on kasvanut viime vuosina. Samalla myös käyttäjien määrä ja palveluiden kirjo on kasvanut. Nykyisin Internetissä voi ostaa lukuisia tavaroita ja palveluja sekä käyttää pankkien ja julkishallinnon palveluja. Monet näistä palveluista käsittelevät käyttäjien henkilökohtaisia tietoja, joita käyttäjä ei halua muiden tietoon. Samaan aikaan palvelujen monipuolistumisen ja määrän kasvaessa erilaiset hyökkäykset palveluita ja palvelujen käsittelemiä tietoja kohtaan ovat kasvaneet. Tämän vuoksi palveluntarjoajan täytyy taata palvelun turvallisuus siten, ettei palvelun käsittelemiä tietoja pääse vääriin käsiin ja ettei kukaan ulkopuolinen pääse näitä tietoja huomaamatta muuttamaan.

Tässä tutkielmassa läpikäydään tärkeimmät turvallisuustekijät, jotka sovelluskehittäjän, sähköisen kauppapaikan pystyttäjänä, täytyy ottaa huomioon. Sähköisiä kauppapaikkoja käytetään WWW-selainten avulla, joten ensimmäinen huomioitava asia on tiedon eheys ja salaaminen selaimen ja palvelun välillä. WWW-selaimet kykenevät välittämään tietoa salatun yhteyden yli, mutta jotta salauksen taso olisi riittävä, vaatii se toimenpiteitä palvelun pystyttäjältä. Salatun yhteyden eri osatekijöitä käsitellään luvussa 2.

Yhteyden salaamisen jälkeen, luvussa 3, käsitellään käyttäjän tunnistamista ja valtuuttamista. Sähköisissä kauppapaikoissa käyttäjän tunnistaminen ei ole aina välttämätöntä, mutta se parantaa palvelun turvallisuutta. WWW-palveluissa käytetään yleisesti tunnistusmenetelmänä käyttäjätunnusta ja sitä vastaavaa salasanaa (Pinkas & Sander, 2002). Näin on usein myös sähköisissä kauppapaikoissa, jolloin esimerkiksi pankkitilin numero ja tunnusluku muodostavat käyttäjän tunnistustiedon. Tämän tunnistustavan turvallisuudessa on kuitenkin paljon potentiaalisia turvallisuusuhkia, jotka palvelua rakennettaessa tulee huomioida. Näiden uhkien poistaminen ei ole aina helppoa, sillä tunnistustiedot ovat alttiita hyökkäyksille tunnistustapahtuman aikana ja myös silloin kun tiedot on talletettu kauppapaikan tietovarastoon tai asiakkaan hallussa olevalle paperilapulle.

Käyttäjän tunnistamisen jälkeen järjestelmä valtuuttaa käyttäjän. Käyttäjän valtuuttamisella tarkoitetaan sitä, kun palvelu sallii eri käyttäjille erilaisia toimintoja. Toisella käyttäjällä voi olla oikeus ostaa kaikkia kauppapaikan tuotteita, kun taas toisen tuotevalikoimaa on rajoitettu.

Järjestelmän ylläpitäjällä voi olla oikeus muokata tuotteiden tietoja, mitä ei ole pelkän tuotteiden osto-oikeuden omaavalla käyttäjällä.

Luvussa 4 käsitellään tuotteiden ja palveluiden maksamista sähköisessä kauppapaikassa sekä eri maksutapojen turvallisuutta. Kauppapaikkaan voidaan liittää useita erilaisia maksutapoja, kauppapaikan luonteesta riippuen. Maksutapojen valinnassa tulee ottaa huomioon kaupankäyntitapa, maksujen suuruus ja maksutavan turvallisuus.

Tutkielman lopuksi esittelen luvussa 5, miten edellä mainitut osatekijät voidaan toteuttaa sähköiselle kauppapaikalle J2EE-ympäristössä. Tarkoitus ei ole kuitenkaan luoda toimivaa sähköistä kauppapaikkaa, vaan esimerkinomaisesti käsitellä tärkeimmät seikat ja tekniikat, joilla sovelluskehittäjä voi parantaa kauppapaikan turvallisuutta.

2 TIEDON SALAAMINEN

Erilaisten kauppapaikkojen ja sovelluksien määrä Internetissä on kasvanut viime vuosina. Samalla yhä useammat käyttäjät lähettävät henkilökohtaisia tietoja tietoverkoissa, joissa tieto kulkee hyvin pitkiä matkoja ja useiden eri tietokoneiden kautta. Tällöin henkilökohtaisten ja arkaluontoisten tietojen salaaminen on välttämätöntä, jottei tietoon pääse käsiksi kukaan muu kuin viestin haluttu vastaanottaja. Salaamisen lisäksi tiedonvälityksessä on huolehdittava myös siitä, ettei kukaan kolmas osapuoli voi muuttaa viestiä tai lähettää viestejä muiden osapuolten nimissä.

Tiedon salaamiseen Internetissä on useita eri tapoja ja menetelmiä. Kaikissa menetelmissä on kuitenkin omat heikkoutensa ja vahvuutensa ja useat salausmenetelmät ovat myös menettäneet vahvuutensa tietokoneiden laskentatehon kasvaessa. Tässä tutkielmassa keskitytään tiedon välitykseen HTTPS-protokollalla. *HTTPS-protokolla* on Internetissä yleisesti käytetty tiedonsiirtoprotokolla, joka myös salaa välitettävän tiedon. Se soveltuu hyvin myös sähköisen kauppapaikan tiedonvälitystavaksi, koska WWW-selaimet sisältävät tuen tälle menetelmälle ilman käyttäjältä vaadittavia erillisasennuksia. HTTPS-protokolla muodostuu kahdesta eri osaprotokollasta, joista HTTP-protokolla huolehtii tiedonvälityksestä ja toinen, SSL-protokolla, salaa välitettävän tiedon.

HTTP-protokolla (Fielding & al., 1999) on TCP/IP-yhteyden päällä sovelluskerroksessa toimiva tilaton tiedonsiirtoprotokolla, jota käytetään WWW-sovelluksissa. Yleisimmin HTTP-protokollaa käytetään WWW-selaimien ja sovelluspalvelimien välillä HTML-sivujen lataamiseen. HTTP-yhteys välittää tiedon salaamattomana, joten arkaluontoisen tiedon välittäminen tällaista yhteyttä käyttäen on riskialtista. Tätä tietoturvaongelmaa varten on luotu SSL-protokolla.

SSL-protokolla on alun perin Netscapen 1990-luvulla kehittämä tiedonsalausprotokolla, jota nykyisin käytetään laajalti tiedon salaamiseen. SSL voidaan yhdistää tietoa välittävään protokollaan ja näin luoda salattu tiedonsiirto avoimessa verkossa toimivien osapuolten välille. Uusimmista SSL-protokollaan perustuvista salausprotokollista käytetään nimitystä TLS-protokolla (Rhee, 2003). Tällä hetkellä viimeisin versio on TLS 1.1. Tässä tutkielmassa

HTTPS-protokollaa käsitellään TLS-salauksen mukaisesti, joten myös viittauksissa salausprotokollaan käytetään TLS-muotoa.

Tässä luvussa läpikäydään aluksi tiedon salaamiseen liittyvät vaatimukset. Tämän jälkeen perehdymme salausmenetelmiin, joilla nämä vaatimukset voidaan täyttää. Lopuksi tutkimme miten salauksessa käytetyt menetelmät yhdessä luovat turvallisen yhteyden Internetissä toimivien sovelluksien välille.

2.1 Salatun tiedonvälityksen vaatimukset

Sähköisen kauppapaikan turvallisuuden tärkeimpiä tekijöitä on tiedonvälityksen turvaaminen asiakkaan ja kauppapaikkana toimivan palvelinkoneen välillä. Asiakkaan syöttämät tilaus-, henkilö- ja maksutiedot siirtyvät asiakkaan koneelta palvelimelle useiden eri tietokoneiden kautta, joten ne täytyy salata ulkopuolisilta. Samalla täytyy estää myös tietojen tahallinen tai tahaton muuttuminen ilman että vastaanottaja huomaisi sitä.

Jotta Internetissä tapahtuvaa tiedonvälitystä voidaan pitää turvallisena, täytyy järjestelmän, tässä tapauksessa sähköisen kauppapaikan, täyttää seuraavat vaatimukset (Verisign 2000; Mel & Baker 2001):

- *Todennus* tarkoittaa sitä, että asiakas voi varmistua siitä kenen kanssa asioi. Sähköisen kauppapaikan tapauksessa asiakas voi siis luottaa siihen, että hänen syöttämänsä tiedot menevät juuri oikealle sovellukselle. Samoin kauppapaikalla on oltava keinot varmistua asiakkaan identiteetistä.
- *Luottamuksellisuus* tarkoittaa viestin salaamista siten, että vain viestin tarkoitettu vastaanottaja voi avata viestin.
- *Tiedon eheys* tarkoittaa sitä, että viestiä ei voi muuttaa vastaanottajan huomaamatta.
- *Kiistämättömyys* varmistaa sen, ettei kukaan muu kuin viestin lähettäjä ole voinut luoda kyseistä viestiä.
- *Avaimenvaihto*, joka tarkoittaa sitä, että tiedonvälityksen eri osapuolet voivat turvallisesti vaihtaa tai sopia tiedon salaamisessa käytettävästä salausavaimesta.

Järjestelmä, joka toteuttaa kaikki edellä mainitut turvallisuusvaatimukset, pystyy luomaan turvallisen tiedonvälityksen eri osapuolten välille. Nämä abstraktit turvallisuusvaatimukset voidaan toteuttaa yhdistämällä useita eri salausmenetelmiä, joita tarkastelemme seuraavaksi.

2.2 Salausmenetelmät

Tiedon salaaminen TLS-protokollalla koostuu useasta eri salaustekniikasta, jotka yhdessä luovat turvallisen tavan tiedon välittämiseen salattuna siten, että kaikki osapuolet voivat varmistua tiedon salaamisesta, eheydestä ja muuttumattomuudesta. Tässä luvussa läpikäydään nämä eri osatekniikat ja se, miten ne yhdessä takaavat tiedonvälityksen turvallisuuden.

2.2.1 Salausavain

Salausavain on pitkä numerosarja, jota tiedon salaus- ja purkualgoritmit käyttävät selväkielisen tiedon salaamisessa ja salatun tiedon avaamisessa selväkieliseksi. *Salausalgoritmi* lukee muunnettavaa viestiä, algoritmin tyypistä riippuen, joko bitti tai lohko kerrallaan ja laskee tiedolle salatun arvon salausavainta käyttäen. Näin salattu viesti saadaan avattua selväkieliseksi joko samalla avaimella, jolloin kyseessä on symmetrinen salaus, tai eri avaimella, jolloin kyseessä on asymmetrinen salaus (Coulouris & al., 2001).

Salausavaimen pituus esitetään yleensä bitteinä. Mitä pidempi salausavain, sitä turvallisempi salaus, kun käytetään luotettavaa salausalgoritmia. Salauksen turvallisuutta ei kuitenkaan voida päätellä pelkästä avaimenpituudesta, sillä eri salausalgoritmit luovat yhtä turvallisen salauksen eripituisilla avaimilla. Lisäksi, saman turvallisuustason luomiseen, symmetrisen salauksen avaimet ovat tyypillisesti huomattavasti lyhempiä kuin asymmetrisen salauksen avaimet. Rinteen (2002) mukaan yleispätevä sääntö on, että asymmetrisen avaimen tulee olla noin kymmenen kertaa pidempi kuin symmetrisen avaimen, jotta salaukset ovat yhtä vahvat. Tämä ei päde kuitenkaan asymmetriseen ECC-algoritmiin (Elliptic Curve Cryptosystems). ECC on huomattavasti tehokkaampi ja käyttää lyhyempiä salausavaimia kuin perinteiset asymmetriset algoritmit, kuten RSA ja DSA. ECC-algoritmin 160-bittinen salausavain vastaa turvallisuudeltaan RSA-algoritmin 1024-bittistä salausavainta.

2.2.2 Symmetrinen salaus ja avaimenvaihto-ongelma

Symmetrisessä salauksessa tieto salataan ja puretaan samaa salausavainta käyttäen. Tällöin sekä tiedon salaajalla että tiedon vastaanottajalla, joka purkaa salauksen, täytyy olla hallussaan sama salausavain, joka täytyy pitää salassa kaikilta muilta osapuolilta. Varsinainen salaus ja salauksen purkaminen tapahtuvat käyttämällä käänteisiä laskufunktioita salausavaimen kanssa (Mel & Baker, 2001). Symmetrinen salaus toteuttaa vaatimuksen luottamuksellisuudesta, jos ainoastaan viestin lähettäjällä ja viestin vastaanottajalla on käytetty salausavain.

Symmetrinen salaus on turvallinen ja tehokas tapa salata tietoa, mutta siinä on avaimenvaihtoon liittyvä heikkous. Kun henkilö A haluaa lähettää viestin henkilölle B, täytyy hänen ennen salatun tiedon lähettämistä ilmoittaa mitä salausalgoritmia ja mitä salausavainta viestin salaamiseen on käytetty. Salausalgoritmin henkilö A voi kertoa selväkielisenä, kun käytössä on turvalliseksi todettu algoritmi. Turvallisia algoritmeja ovat algoritmit, joiden vahvuus perustuu käytettyyn salausavaimeen, eikä salauksen laskentatapaan (Mel & Baker, 2001). Laskentatavan tietämällä ei siis voi päätellä viestin sisältöä. Turvallisina pidettyjä algoritmeja ovat muun muassa IDEA ja Blowfish, joissa voidaan käyttää tällä hetkellä riittävän turvallisena pidettyä 128-bittistä salausavainta (Rinne, 2002).

Salausavainta ei kuitenkaan voi lähettää selväkielisenä, koska sen joutuminen kolmannen osapuolen, mahdollisen väärinkäyttäjän, käsiin tekisi koko salauksesta turhan. Tämä ulkopuolinen osapuoli pystyisi avaamaan viestin salausavaimella aivan kuten viestin oikea vastaanottajakin. Salausavaimen toimittamiseen on monta tapaa, mutta lähes kaikissa on heikkoutensa ja toimittamisesta aiheutuu yleensä myös kustannuksia. Yleensä joudutaan etsimään tapa, joka on riittävän turvallinen ja joka kustannuksiltaan vastaa tarvittavaa turvallisuustasoa. Jos kyseessä on kuitenkin salausavain, jolla salataan erittäin luottamuksellisia tietoja, turvallisin tapa on antaa salausavain kädestä käteen, ilman muita henkilöitä tai tietojärjestelmiä (Mel & Baker, 2001).

Yksittäisessä tapauksessa salausavaimen välittäminen kädestä käteen voi toimia, mutta kun salattujen viestien vastaanottajia on useampi kuin yksi, avaimenvaihto käy monimutkaisemmaksi. Esimerkiksi, kun yritys X lähettää salatun viestin asiakkaalleen Y käyttäen salausavainta K1, täytyy asiakkaalle Z lähetetyt viestit salata eri salausavaimella K2.

Jos kaikki asiakkaat käyttäisivät samaa salausavainta, pystyisivät he kaikki avaamaan muille asiakkaille lähetetyt tiedot, mikä ei ole liiketoiminnan kannalta suotavaa. Lisäksi asiakkaat pystyisivät lähettämään yrityksen X nimissä viestejä toisille asiakkaille, mikä aiheuttaisi lisäongelmia. Usean eri salausavaimen käyttäminen käy myös hankalaksi kun vastaanottajien määrä nousee suureksi. Jos yrityksellä on esimerkiksi tuhansia asiakkaita, jokaista varten pitäisi olla oma salausavain. Tällaisen avainmäärän käsittely on hankalaa ja lisää kustannuksia, kun jokainen avain pitää siirtää asiakkaalle turvallisesti (Mel & Baker, 2001). Näitä symmetrisen salauksen heikkouksia vastaan kehitettiin asymmetrinen salaus, jota tarkastellaan seuraavaksi.

2.2.3 Asymmetrinen salaus

Symmetrisen salauksen suurimpana heikkoutena on avaimen turvallinen välittäminen viestin vastaanottajalle. Vastaanottaja ei myöskään voi olla varma, kuka on lähettänyt ja salannut hänen vastaanottamansa viestin, mikäli avain on jaettu useammalle vastaanottajalle (Mel & Baker, 2001). Näihin ongelmiin kehitettiin 1970-luvulla *asymmetrinen salaus*, josta käytetään myös nimitystä epäsymmetrinen salaus. Asymmetrinen salaus perustuu matemaattisiin käänteisfunktioihin, joiden avulla tieto voidaan salata ja purkaa kahta eri salausavainta käyttäen. Jokaisella käyttäjällä on oma yksityinen avain, jolla käyttäjä salaa viestinsä. Yksityistä avainta ei paljasteta kenellekään muulle. Yksityisen avaimen lisäksi käyttäjällä on sanoman purkamiseen käytettävä julkinen avain, jota käyttäjä vapaasti levittää kaikille sanoman vastaanottajille. Yksityisellä avaimella salatun viestin voi avata ainoastaan julkisella avaimella ja vastaavasti julkisella avaimella salatun viestin voi avata ainoastaan yksityisellä avaimella (Viestintävirasto 2004).

Asymmetrisen salauksen avaimet muodostavat yhdessä avainparin, jonka avulla voidaan ratkaista avaimen välitysongelma, kun salaukseen käytettyä avainta ei enää lähetetä vastaanottajalle ja jokaista vastaanottajaa varten ei enää tarvitse luoda omaa salausavainta. Tiedon salaaminen yksityisellä avaimella täyttää myös kiistämättömyyden vaatimuksen, sillä vain yksityisen avaimen haltija voi luoda viestejä, jotka avautuvat julkisella avaimella. Julkisella avaimella salattu viesti ei kuitenkaan toteuta kiistämättömyyden vaatimusta, koska kuka tahansa on voinut käyttää julkista avainta viestin luomiseen (Mel & Baker, 2001).

Asymmetrisen salauksen heikkoutena verrattuna symmetriseen salaukseen on sen tehottomuus. Suurin syy tähän tehottomuuteen on se, että asymmetriset avaimet ovat pääsääntöisesti paljon symmetrisiä avaimia pidempiä, mikä vaikuttaa laskennan tehokkuuteen. Tiedon salaaminen esimerkiksi DES-algoritmilla, joka on symmetrinen salausalgoritmi, on noin sata kertaa nopeampaa kuin asymmetrisellä RSA-algoritmilla (RSA Security, 2004).

2.2.4 Tiivisteet

Tiiviste on mielivaltaisen pitkästä tiedosta laskettu tarkistusarvo, josta ei kuitenkaan voi päätellä mitään alkuperäisestä tiedosta. Tiiviste lasketaan *tiivistefunktiolla*, joka palauttaa vakiopituisen tiivistearvon tiedosta. Hyvän tiivistefunktion ominaisuus on se, että todennäköisyys löytää kaksi sellaista viestiä, joista laskettu tiivistearvo olisi sama, on hyvin pieni (O'Mahony & al., 2001). Tiivisteitä käytetään varmistamaan, että vastaanottajan saama viesti on sama kuin mikä hänelle alun perin lähetettiin. Tiivisteiden avulla voidaan siten toteuttaa vaatimus tiedon eheydestä. Tiivisteitä on hyvin vaikea väärentää, sillä yhdenkin merkin muuttaminen tekstistä muuttaa tiivisteen arvoa huomattavasti.

Tällä hetkellä tiivisteen laskemiseen käytetään yleisesti joko MD5-tiivistealgoritmia, joka luo tekstistä 128-bittisen tiivisteen, tai SHA-algoritmia, joka muodostaa 160-bittisen tiivisteen (Coulouris & al., 2001). Taulukossa 1 näkyvät syötetiedoista ”tiiviste” ja ”Tiiviste” muodostetut tiivistearvot molemmilla algoritmeilla laskettuna. Vaikka syötteet poikkeavat toisistaan vain yhden kirjaimen verran, tiivistearvot poikkeavat toisistaan huomattavasti.

Algoritmi	Syöte	Tiivistearvo
SHA	Tiiviste	ec:b9:2c:c6:2c:6a:41:b7:40:50:2c:24:76:69:e2:0e:0b:44:cc:92
SHA	tiiviste	25:5b:06:87:8f:85:2a:55:8f:45:f1:74:e0:f0:ff:c5:fc:1c:8c:7d
MD5	Tiiviste	35:70:26:fe:33:c4:39:81:f4:58:b5:0d:b7:0f:b2:ed
MD5	tiiviste	96:3c:1a:f8:5e:1a:cc:49:94:72:c0:19:c5:88:e2:64

Taulukko 1: Tiivistearvot SHA- ja MD5-algoritmeille.

Käytännössä kahden saman tiivisteen muodostuminen eri syötetiedoista on kuitenkin mahdollista. Rescorlan (2001) mukaan tällaisten kahden eri syötteen löytyminen riippuu

tiivisteen pituudesta. Jos syötetiedosta lasketun tiivisteen pituus on 128 bittiä, tarvitsee läpikäydä keskimäärin 2^{64} erilaista syötetietoa, jotta löydetään kaksi syötettä, joiden tiivistearvot ovat samat. Tällä hetkellä riittävän turvallisena tiivisteen pituutena pidetään 128 bittiä, mutta tietokoneiden laskentatehon kasvaessa tällaisenkin tiivisteen pituus voi olla riittämätön.

Tiivisteet toimivat tiedonvälityksessä siten, että viestin lähettäjä laskee tekstistä tiivisteen, joka liitetään viestiin ja vastaanottaja laskee vastaavan tiivisteen saamastaan viestistä. Jos tiivisteet ovat samat, on viesti aito, eikä sitä ole kukaan muuttanut viestinvälityksen aikana (Mel & Baker, 2001). Muussa tapauksessa viesti on muuttunut matkalla, jolloin vastaanottaja voi hylätä koko viestin. Koska sellaisen viestin löytäminen, jonka tiiviste olisi sama kuin alkuperäisen viestin tiiviste, kestää liian kauan, tiivisteen väärentäminen ei ole kannattavaa. Ja mikäli viesti on myös luotettavasti salattu, ei kukaan ulkopuolinen osapuoli pysty lukemaan viestiä eikä myöskään väärentämään sitä vastaanottajan huomaamatta.

2.2.5 Digitaalinen allekirjoitus

Digitaalinen allekirjoitus on menetelmä, jolla viestin lähettäjä vakuuttaa vastaanottajalle että viesti on juuri tältä lähettäjältä eikä sitä ole kukaan muuttanut matkalla. Digitaalinen allekirjoitus tapahtuu siten, että lähettäjä salaa lähettämänsä viestin omalla yksityisellä avaimellaan, jota ei ole kellään muulla. Sen jälkeen vastaanottaja avaa viestin lähettäjän julkisella avaimella, joka avaa viestin selväkieliseen muotoon. Koska yksityinen ja julkinen avain muodostavat asymmetrisen salauksen avainparin, vain yksityisellä avaimella salattu viesti avautuu oikein julkisella avaimella. Koska kellään muulla kuin vastaanottajalla ei ole yksityistä avainta, viestin täytyy olla alkuperäisen lähettäjän salaama. Tämä tapa toteuttaa myös salaukseen liittyvän kiistämättömyysvaatimuksen eli viestin lähettäjä ei voi kieltää lähettäneensä viestiä, koska kukaan muu ei voi luoda sellaista viestiä (Mel & Baker, 2001 ss. 119).

Yleensä digitaalista allekirjoitusta käytetään yhdessä tiivisteen kanssa, jolloin digitaalinen allekirjoitus tehdään tiivisteelle (Coulouris & al., 2001 ss. 264-265). Tämä tapa on yhtä turvallinen ja samalla tehokkaampi, koska allekirjoitus tarvitsee yleensä tehdä huomattavasti lyhyemmälle tiedolle.

2.2.6 Julkisen avaimen välitys

Asymmetriseen salaamiseen kuuluu suurena osana avaimenvälitys, joka tarkoittaa sitä, että viestin lähettäjä toimittaa viestin vastaanottajalle oman julkisen avaimensa viestin avaamista varten. Julkiset avaimet ovat vastaanottajalle vapaasti saatavissa, mutta avainten välitykseenkin liittyy tietoturvauhkia. Mikäli avain lähetetään verkon yli suojaamattomassa muodossa, voi kolmas osapuoli vaihtaa avaimen haluamakseen, esimerkiksi omaksi julkiseksi avaimekseen. Tämän jälkeen kolmannen osapuolen on helppoa lähettää väärennettyjä viestejä, jotka avautuvat väärennetyllä avaimella (Mel & Baker, 2001). Tätä uhkaa varten on luotu tapoja varmistaa avaimen oikeellisuus ja alkuperä.

Julkisen avaimen infrastruktuuri, *PKI* (Public Key Infrastructure), on yleisesti käytetty toimintamalli julkisten avainten hallinnointiin ja välitykseen, jonka avulla voidaan välittää julkinen avain turvallisesti viestin vastaanottajalle (Viestintävirasto, 2004). PKI:ssa on kaksi tapaa toteuttaa julkisen avaimen välitys: *X.509* ja *PGP*. Selaimet käyttävät näistä X.509-standardia (Housley & al., 2002), joten se soveltuu hyvin turvallisen yhteyden muodostamiseen sähköiseen kauppapaikkaan.

X.509-standardiin kuuluu oleellisesti *varmentaja*, joka on symmetriseen salaukseen liittyvä luotettu kolmas osapuoli. Varmentajan tehtävänä on varmistaa, että salauksessa käytetty julkinen avain todella on salaukseen osallistuvan osapuolen julkinen avain, eikä kenenkään muun. Varmennus perustuu digitaaliseen allekirjoitukseen, jonka varmentaja on luonut kyseisen osapuolen julkisesta avaimesta sekä varmentajaan ja osapuoleen liittyvistä tiedoista. Tällaista digitaalista allekirjoitusta kutsutaan *digitaaliseksi sertifikaatiksi* (Mel & Baker, 2001, ss. 163-191).

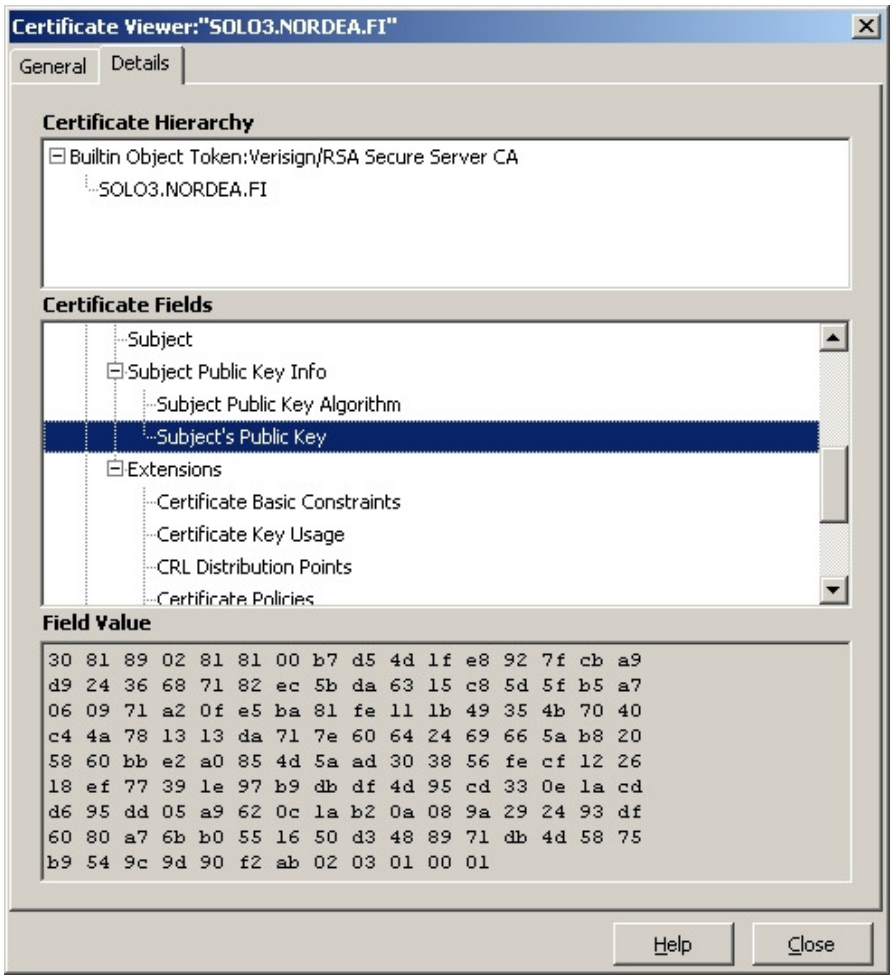
Kuvassa 1 on Nordean Solo-palvelun käyttämä digitaalinen sertifikaatti, jossa yläosassa nähdään yleisiä sertifikaatin omistajan tietoja. Sertifikaatti sisältää myös tietoja varmentajasta, sertifikaatin voimassaolosta sekä sertifikaatista lasketut tiivistearvot.

Kuva 1: Nordean Solo-palvelun käyttämä digitaalinen sertifikaatti.

Avaimen välitys X.509-standardia käyttäen tapahtuu seuraavasti (Mel & Baker, 2001):

- Asiakkaana toimiva sovellus, esimerkiksi WWW-selain ottaa yhteyden palvelimeen. Palvelin vastaa lähettämällä varmentajan allekirjoittaman digitaalisen sertifikaattinsa asiakkaalle. Digitaalisen sertifikaatin palvelun ylläpitäjä on hankkinut luotettavalta varmentajalta.
- Asiakas varmentaa digitaalisen sertifikaatin asiakaskoneella olevalla varmentajan julkisella avaimella. Yleisimpien varmentajien julkiset avaimet tulevat selainten asennuspakettien mukana, joten niiden väärentäminen jonkun ulkopuolisen toimesta on hyvin hankalaa. Käyttäjä voi myös laskea asennuspaketista tiivisteen ja verrata sitä selaimen valmistajan tai jonkun muun luotetun tahon asennuspaketista laskemaan tiivisteeseen. Selaimen asennuksen jälkeen on selaimen käyttäjän vastuulla huolehtia siitä, ettei kukaan pääse muuttamaan tietokoneelle tallennettuja julkisia avaimia.

- Mikäli digitaalisen sertifikaatin tiedot ja tarkistussummat täsmäävät, asiakas salaa seuraavat palvelimelle lähettämänsä viestit sertifikaatin julkista avainta käyttäen.



Kuva 2: Nordean Solo-palvelun käyttämä julkinen avain.

2.2.7 Istuntokohtaisen salausavaimen luominen

Kuten aiemmin käsiteltiin, asymmetrinen salaus on salaustehokkuudeltaan symmetristä salausta heikompi, mutta ratkaisee symmetrisen salauksen avaimenvälitysongelman. Näitä salaustapoja voidaan käyttää kuitenkin yhdessä, jolloin saavutetaan turvallinen avaimenvaihto ja tehokas tiedon salaaminen (Rinne, 2002). Tällöin varsinainen tieto salataan symmetrisellä salauksella, mutta salauksessa käytetty salausavain tai sen laskentaparametrit salataan asymmetrisesti. Jokaisen istunnon alussa lasketaan uusi salausavain, jolloin yksittäisen

salausavaimen vääriin käsiin joutuminen ei paljasta kaikkia osapuolten välillä kulkeneita viestejä. Seuraavaksi läpikäymme yleisiä symmetrisen salausavaimen muodostustapoja.

Ensimmäinen avaimenvaihtoalgoritmi julkaistiin jo vuonna 1976 ja on nimeltään Diffie-Hellman-algoritmi (Mel & Baker, 2001). Algoritmin mukaan avaimenvaihto tapahtuu osapuolten A ja B välillä siten, että aluksi A ja B sopivat laskennassa käytetyistä julkisista lukuarvoista. Tämän jälkeen molemmat muodostavat omat salaiset lukuarvonsa ja syöttävät ne yhdessä julkisten lukuarvojen kanssa ennalta määrättyyn eksponenttifunktioon. Tämän jälkeen A ja B lähettävät funktion palauttamat arvot toisilleen ja laskevat arvoja käyttäen salaisen avaimen. Tätä avainta voidaan sitten käyttää tiedon salaamisessa symmetrisesti. Laskentatapa perustuu matemaattisiin funktioihin, joiden avulla A ja B muodostavat saman symmetrisen avaimen, mutta kukaan ulkopuolinen ei voi muodostaa samaa avainta, vaikka saisikin käsiinsä laskennassa välitetyt lukuarvot.

Diffie-Hellman-algoritmi voidaan kuitenkin murtaa *välimieshyökkäyksellä* (man in the middle –attack). Välimieshyökkäyksessä ulkopuolinen taho kaappaa lähetetyt viestin, muokkaa sitä haluamakseen ja lähettää väärennetyn viestin vastaanottajalle, ilman että vastaanottaja huomaa viestin muokkaamista. Tällä tavoin hyökkääjä C voi lähettää haluamansa arvot osapuolille A ja B, jotka tämän jälkeen muodostavat hyökkääjän haluaman salausavaimen tiedonvälitystä varten.

Diffie-Hellman-algoritmin lisäksi TLS-protokolla voi käyttää myös RSA-algoritmia avaimen välitykseen. Tämä vaihtoehto vaatii, että toinen osapuoli lähettää julkisen avaimensa toiselle avaimenvaihtoon osallistuvalle osapuolelle. Julkinen avain lähetetään tyypillisesti digitaalisen sertifikaatin mukana heti kun osapuolet aloittavat salatun yhteyden muodostamisen. Avaimenvaihto etenee tällöin siten, että A ja B lähettävät ensin satunnaislukuja salaamattoman yhteyden yli ja samalla sopivat käytettävästä symmetrisestä algoritmistä. Tämän jälkeen A salaa B:n julkisella avaimella satunnaislukuja sisältävän viestin ja lähettää sen B:lle, joka avaa viestin yksityisellä avaimellaan. Lähetettyihin satunnaislukuihin perustuen molemmat osapuolet voivat muodostaa symmetrisen salauksen käyttämän salaisen avaimen. Kukaan ulkopuolinen ei voi päätellä tätä salaista avainta, koska osa satunnaisluvuista lähetettiin salattuna (Rescorla, 2001). Samalla tämä avaimenvälitystapa estää myös välimieshyökkäyksen onnistumisen.

2.3 Tiedon salaaminen TLS-protokollalla

Tiedon salaamiseen TLS-protokollalla käytetään useita salaustekniikoita, jotka yhdessä muodostavat turvallisen salaustavan. Tässä osiossa käydään läpi turvallisen yhteyden luominen ja tiedonvälitys näitä salaustekniikoita käyttäen. Viestien sisältö ja välittäminen läpikäydään korkeammalla tasolla selkeyden vuoksi, sillä jokaisen HTTPS-viestin määrittely ei ole kokonaisuuden kannalta oleellista.

2.3.1 Käyttäjän tunnistus TLS-protokollalla

TLS-yhteyden muodostaminen vaatii aina palvelinsovelluksen tunnistamisen, mutta myös asiakkaan tunnistaminen digitaalista sertifikaattia käyttäen on mahdollista. Rescorlan (2001) mukaan tällainen tunnistaminen soveltuu tapaukseen, jossa palveluja halutaan tarjota vain tietyille asiakkaille. Tällöin käyttäjä voidaan tunnistaa jo yhteyden luomisvaiheessa ja lopettaa yhteys välittömästi jos käyttäjän tunnistus epäonnistuu.

Käyttäjän tunnistus salatun yhteyden luomisvaiheessa tapahtuu aina palvelimen pyynnöstä. Käyttäjän tunnistus vaatii myös, että asiakas on hankkinut digitaalisen sertifikaatin, samaan tapaan kuin palvelinsovelluskin. Tunnistus tapahtuu siten, että asiakas lähettää palvelimelle digitaalisen sertifikaattinsa ja yksityisellä avaimellaan salatun viestin, joka sisältää tiivistearvon edellisistä viesteistä. Palvelin avaa viestin asiakassertifikaatin julkisella avaimella ja vertaa asiakkaan laskemaa tiivistettä omaan tiivistearvoonsa (Rescorla, 2001). Jos arvot täsmäävät, kuuluu sertifikaatti asiakkaalle, koska vain yksityisen avaimen haltija on voinut luoda kyseisen viestin. Jos arvot eivät täsmää, tunnistus epäonnistuu ja palvelin voi päättää yhteyden.

TLS-määrityksen (Dierks & Allen, 1999) mukaan palvelimen ei kuitenkaan ole pakko estää asiakkaan pääsyä palveluun, vaikka asiakkaalla ei olisikaan tarvittavaa sertifikaattia, vaan tämä on jätetty palvelinsovelluksen toteuttajan harkintaan. Ja koska käyttäjän tunnistus digitaalista sertifikaattia käyttäen tapahtuu aina palvelinsovelluksen aloitteesta, on palvelimen vastuulla päättää tietoturvan taso.

WWW-palvelun luonteesta riippuu halutaanko käyttäjä tunnistaa jo yhteyden luomisvaiheessa. Tässä tutkielmassa perehdytään sähköisen kauppapaikan turvallisuuteen ja jos palvelun on tarkoitus olla avoin kaikille halukkaille, on käytännöllistä jättää pois asiakkaan tunnistaminen salatun yhteyden luomisessa. Sähköisen kauppapaikan käytettävyys kärsii, jos jokaisen asiakkaan täytyy hankkia digitaalinen sertifikaatti tunnistautumista varten. Varsinaisen sovelluksen käyttö vaatii kuitenkin asiakkaan tunnistamisen, joka voidaan toteuttaa muilla tavoin. Näihin tapoihin perehdytään tutkielman seuraavassa luvussa.

2.3.2 Tiedon salaaminen HTTPS-protokollalla

Tiedonvälitys HTTPS-yhteyttä käyttäen tapahtuu Rescorlan (2001) mukaan seuraavasti:

1. Asiakassovellus, sähköisen kauppapaikan tapauksessa WWW-selain, ottaa yhteyden palvelinsovellukseen eli sähköiseen kauppapaikkaan ja lähettää viestissä parametrilistan tiedon salausta varten. Parametrilista sisältää tiedon niistä salausalgoritmeista ja tiivistefunktioista, joita asiakassovellus pystyy käsittelemään. Lisäksi parametrina välittyy satunnaislukuja, joita käytetään salausavaimen luomisessa sekä viestien muuttumattomuuden tarkistamisessa.

2. Palvelinsovellus valitsee parametrilistasta salauksessa käytettävät menetelmät sen mukaan mitä salausmenetelmiä palvelinsovellus tukee ja mitkä tarjoavat parhaan turvallisuuden salaukselle. Nämä salausmenetelmät palvelinsovellus palauttaa asiakkaalle omana parametrilistanaan. Parametrilista sisältää myös palvelinsovelluksen luomia satunnaislukuja, joita käytetään yhdessä asiakassovelluksen satunnaislukujen kanssa salausavaimen luomiseen ja viestien tarkistamiseen.

3. Palvelinsovellus lähettää asiakkaalle digitaalisen sertifikaattinsa, jonka aitouden asiakas tarkistaa luotetun kolmannen osapuolen, varmentajan, digitaalisella sertifikaatilla. Tämä varmentajan sertifikaatin asiakas on saanut WWW-selaimen asennuspaketin mukana.

4. Jos palveluun on määritelty asiakkaan tunnistaminen digitaalisella sertifikaatilla, palvelin pyytää asiakkaalta sertifikaattia. Jos asiakkaalla on sertifikaatti, se lähettää

sertifikaatin sekä yksityisellä avaimellaan salatun tiivistearvon edellisistä viesteistä. Palvelin varmistaa sertifikaatin aitouden, avaa viestin sertifikaatin julkisella avaimella ja vertaa tiivistearvoa itse laskemaansa arvoon. Käyttäjän tunnistus onnistuu, jos arvot ovat samat. Jos asiakkaalla ei ole sertifikaattia, se ilmoittaa siitä tyhjällä sertifikaattiviestillä, minkä jälkeen palvelin voi halutessaan päättää yhteyden. Tämän kohdan yli hypätään jos palvelu ei sisällä asiakkaan tunnistamista digitaalisella sertifikaatilla.

5. Tämän jälkeen asiakas lähettää palvelimelle lisää salausparametreja, joita käytetään salaisen avaimen laskemiseen. Viesti on salattu palvelimen sertifikaatin mukana tulleella julkisella avaimella, joten ainoastaan palvelinsovellus voi onnistuneesti avata viestin. Näin myös asiakas voi varmistua vastapuolen identiteetistä, koska vain sertifikaatin omistaja voi luoda oikean salaisen avaimen ja jatkaa tiedonvälitystä kyseistä avainta käyttäen.

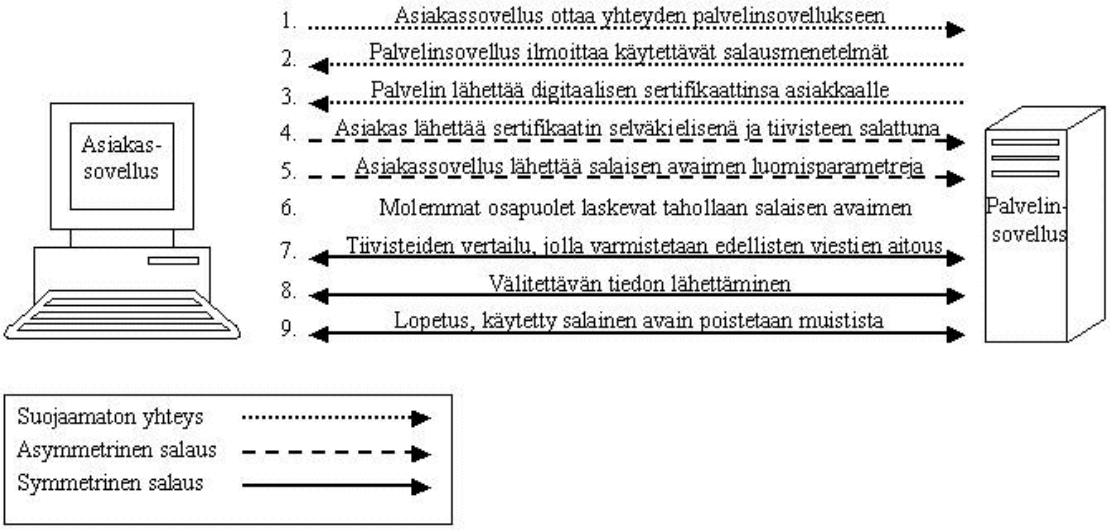
6. Molemmat osapuolet luovat salausparametreja käyttäen saman salaisen avaimen. Tällä menetelmällä tiedonsalauksessa käytettävä salausavainta ei tarvitse missään vaiheessa lähettää toiselle osapuolelle, mikä lisää salauksen turvallisuutta.

7. Seuraavaksi asiakas lähettää palvelimelle tiivistearvon äskeisistä, yhteyden luomisessa käyttämistään viesteistä. Palvelinsovellus laskee vastaavan arvon asiakkaan lähettämistä viestistä ja varmistaa että tiivisteet ovat samat. Samalla tavalla osapuolet laskevat tiivisteet myös palvelimen lähettämistä viesteistä. Tällä tavoin estetään se, ettei kukaan pysty muuttamaan viestejä osapuolten huomaamatta ja vaikuttaa käytettyihin salausalgoritmeihin ja parametreihin. Nämä viestit ovat ensimmäiset, jotka salataan edellisessä kohdassa muodostetulla salaisella avaimella.

8. Mikäli tiivisteet täsmäävät, aloitetaan varsinaisen tiedon siirtäminen. Tieto salataan ja avataan salaisella avaimella ja jokaisesta viestistä lasketaan tiivistearvot, joiden pitää täsmätä. Viestejä, joiden tiivistearvot eivät täsmää, ei kummankaan osapuolen pidä huomioida.

9. Tiedon siirron jälkeen asiakas lähettää lopetusviestin, johon palvelin vastaa. Molemmat osapuolet poistavat muistista käytetyn salaisen avaimen, joten lähetettyjä

viestejä ei enää jälkikäteen voi avata ja seuraavalla kerralla sovellusten on luotava uusi salausavain.



Kuva 3: HTTPS-yhteyden muodostaminen ja tiedonsiirto salatun yhteyden yli.

3 KÄYTTÄJÄN TUNNISTUS JA VALTUUTUS

Tiedon salaamisen lisäksi sähköisen kauppapaikan turvallisuutta voidaan parantaa käyttäjän tunnistuksella ja käyttöoikeuksien myöntämisellä. Väärinkäytön estämiseksi kauppapaikat voivat vaatia käyttäjää rekisteröitymään ja samalla myöntävät käyttäjälle käyttöoikeuden palveluun. Tässä luvussa käsitellään sähköisten kauppapaikkojen yleisimmät tunnistustavat ja niihin liittyvät heikkoudet.

Käyttäjän tunnistamisen lisäksi eri käyttäjille voidaan myös jakaa erilaisia oikeuksia kauppapaikan toimintoihin. Suurin osa käyttäjistä toimii vain ostajina, mutta kanta-asiakkailla voi olla omia lisätoimintojaan ja kauppapaikan ylläpitäjillä voi vastaavasti olla oikeuksia esimerkiksi hinnan muuttamiseen, jota ei kauppapaikan asiakkailla voi olla. Tässä luvussa perehdytään myös tapoihin erilaisten käyttöoikeuksien myöntämiseen sähköisen kauppapaikan yhteydessä.

Käyttäjä voidaan tunnistaa jo salatun yhteyden luomisen yhteydessä, käyttäjän digitaalisen sertifikaatin avulla, mikä kuvattiin edellisessä luvussa. Digitaalinen sertifikaatti on turvallinen käyttäjän tunnistustapa, mutta se ei ole kovin käytännöllinen henkilöasiakkaille suunnatulle kauppapaikalle. Tällöin jokaisen asiakkaan tulisi hankkia digitaalinen sertifikaatti, joka on maksullinen ja lisää siten kauppapaikan käytön kustannuksia sekä myös käytön hankaluutta. Tästä syystä käyttäjän tunnistus tehdään muilla tavoin, joihin perehdytään tässä luvussa.

3.1 Käyttäjän tunnistus ja tunnistusmenetelmien kategoriat

Käyttäjän tunnistus tarkoittaa Tullochin (2003) määritelmän mukaan henkilön, tietokoneen tai prosessin tunnistuksen varmistamista. Tämä tarkoittaa sitä, että järjestelmä ei varsinaisesti pyri tunnistamaan käyttäjää, vaan pyrkii todentamaan käyttäjän esittämän väitteen identiteetistään. Tähän todentamiseen järjestelmä käyttää jotain käyttäjältä saamaansa lisätietoa kuten salasanaa, sormenjälkeä tai digitaalista sertifikaattia.

Käyttäjän tunnistukseen on olemassa useita eri menetelmiä, jotka voidaan Kumarin (2004) mukaan jakaa loogisesti kolmeen eri kategoriaan:

1. Käyttäjä tietää jotain, mitä kukaan muu ei tiedä ja käyttää tietoa tunnistukseen. Salasanat ja PIN-koodit ovat yleisimpiä tällaisia tietoja.
2. Käyttäjällä on hallussaan jotain yksilöivää, kuten pankki- tai älykortti.
3. Käyttäjällä on jokin sellainen yksilöivä ominaisuus tai piirre, jota ei muilla käyttäjillä ole. Biometriset tunnisteet, kuten sormenjäljet, toteuttavat tämän ehdon.

Näiden lisäksi Basu & Muylle (2003) määrittelevät neljännen kategorian:

4. Käyttäjä kuuluu johonkin rajattuun ryhmään. Käyttäjä voi esimerkiksi kirjautua järjestelmään edustamansa yrityksen työntekijänä, jolloin käyttäjän tunnistaminen suoritetaan yrityksen perusteella eikä fyysisen käyttäjän perusteella.

Eri kategorioihin kuuluvia tunnistustapoja voidaan myös yhdistellä. Käyttäjällä voi esimerkiksi olla älykortti, jonka käyttäminen vaatii biometrisen tunnistuksen ja PIN-koodin syöttämisen. Tällöin älykortin hukkaaminen ei mahdollista kortin luvatonta käyttöä, koska vain kortin oikealla haltijalla on mahdollisuus täyttää tunnistuksen kriteerit. Tällainen tunnistustapojen yhdistely parantaa tunnistuksen turvallisuutta (Corcoran & al., 1999; Kumar, 2004).

3.2 Sähköisen kauppapaikan tunnistusmenetelmät

Käyttäjän tunnistus sähköisessä kauppapaikassa ei ole aina välttämätöntä, mutta se nostaa kauppapaikan turvallisuustasoa. Tunnistusmenetelmissä on kuitenkin omat heikkoutensa ja ominaisuutensa, jotka täytyy ottaa huomioon järjestelmää suunnitellessa ja toteutettaessa.

Kaikki tunnistusmenetelmät eivät myöskään mielestäni sovellu käyttäjän tunnistamiseen sähköisessä kauppapaikassa ennen kuin vaadittava tekniikka arkipäiväistyy ja tietämys tunnistusmenetelmistä kasvaa. Siten esimerkiksi biometrinen tunnistus ei mielestäni ole vielä käytännöllinen, koska se vaatii biometrisen tunnisteen keräämistä jokaiselta käyttäjältä, mihin käyttäjät saattavat suhtautua epäilevästi (Basu & Muylle, 2003). Myös kustannukset ovat biometrisessa tunnistuksessa huomattavasti suuremmat kuin esimerkiksi käyttäjätunnuksen ja

salasanan perusteella tapahtuvassa tunnistuksessa. Näin ollen perehdymme salasanan ja älykortin avulla tehtävään tunnistukseen sekä niiden heikkouksiin ja turvaamistapoihin.

3.2.1 Tunnistus käyttäjätunnusta ja salasanaa käyttäen

Yleisin tapa tunnistaa käyttäjä sähköisissä kauppapaikoissa on käyttäjätunnuksen ja salasanan käyttäminen. Tunnistuksessa käyttäjä lähettää käyttäjätunnuksen sekä salasanan WWW-selainta käyttäen palvelinsovellukselle, joka tarkistaa tietojen oikeellisuuden vertaamalla niitä oman tietovarastonsa tietoihin. Tämän tunnistustavan etuina ovat käytännöllisyys ja sopivuus sekä palveluntarjoajalle että loppukäyttäjälle (Pinkas & Sander, 2002). Tämä tunnistustapa ei myöskään vaadi palvelun käyttäjää asentamaan tietokoneelleen lisälaitteita, joten tunnistamisen aiheuttamat kustannukset ovat käyttäjälle pienemmät kuin monessa muussa tunnistusmenetelmässä.

3.2.1.1 Salasanaan perustuvan tunnistuksen heikkoudet

Käyttäjätunnuksella ja salasanalla suoritettavaan tunnistukseen liittyy huomattavia turvallisuusriskejä. Kun käyttäjät rekisteröityvät useisiin Internetissä toimiviin palveluihin, joihin sisäänkirjaudutaan käyttäjätunnuksella ja salasanalla, käy useiden monimutkaisten salasanojen muistaminen hankalaksi. Tällöin käyttäjä, muistamista helpottaakseen, valitsee joko useita yksinkertaisia salasanoja, käyttää samoja salasanoja useissa eri palveluissa tai jopa kirjoittaa salasanoja muistilapuille (Halderman & al., 2005). Kaikki nämä tavat heikentävät tunnistuksen turvallisuutta.

Yksinkertaisten salasanojen käyttö altistaa tunnistuksen sanakirjahyökkäykselle (Pinkas & Sander, 2002). *Sanakirjahyökkäys* (dictionary attack) on hyökkäys, joka käy läpi listaa yleisistä sanoista ja tunnetuista heikoista salasanoista, tekee niistä yksinkertaisia muunnoksia ja koettaa kirjautua näillä muunnoksilla järjestelmään. Sanakirjahyökkäys on osoittautunut hyvin tehokkaaksi tavaksi löytää heikkoja salasanoja (Yan, 2001). Mikäli hyökkääjällä on tiedossa useita käyttäjätunnuksia, eikä ole väliä kenen käyttäjän nimissä hän sisäänkirjautuu järjestelmään, niin hyökkäys on mielestäni vielä tehokkaampi, koska tällöin kasvaa todennäköisyys sille, että joku näistä käyttäjistä on valinnut heikon salasanan.

Samojen salasanojen käyttö taas mahdollistaa sen, että jos hyökkääjä selvittää yhdessä palvelussa käytettävän salasanan, niin samalla salasanalla voi kirjautua myös muihin järjestelmiin. Tämä on ongelmallista varsinkin silloin, jos samaa salasanaa käytetään eri turvallisuusasteisissa palveluissa (Ives & al., 2004). Tällöin hyökkäämällä heikomman turvallisuuden palvelua vastaan, hyökkääjä voi saada haltuunsa myös korkeamman turvallisuusasteen omaavan palvelun salasanat.

Salasanojen muistiinkirjoittaminen laskee tunnistuksen turvallisuustason hyvin matalalle, koska kuka tahansa, joka saa salasanan käsiinsä, pystyy kirjautumaan järjestelmään. Salasanojen muistiinkirjoittaminen on kuitenkin hyvin yleistä. Summers ja Bosworth (2004) mainitsevat vuonna 2003 tehdyn tutkimuksen, jonka mukaan tietotekniikka-alalla työskentelevistä 55 % oli kirjoittanut salasanoja muistiin ainakin kerran ja 9 % tutkimukseen osallistuvista kirjoitti muistiin kaikki salasanat.

Edellä mainittujen turvallisuusuhkien lisäksi salasanoja voidaan murtaa *raa'an voiman hyökkäyksellä* (brute force attack), jolloin hyökkääjä läpikäy kaikki mahdolliset salasanat käyttäjätunnukselle kunnes löytää oikean salasanan. Mahdollisten salasanojen määrä kasvaa kuitenkin nopeasti salasanan pituuden kasvaessa, joten pitkille salasanoille tämä hyökkäystapa on hyvin hidas (Tulloch, 2003). Tämän vuoksi raa'an voiman hyökkäystä ei usein käytetä sellaisenaan murtamaan salasanoja, vaan yhdistettynä sanakirjahyökkäykseen. Tällöin salasanan murtamista yrittävä ohjelma muodostaa lyhyitä merkkijonoja ja yhdistelee niitä sanoihin, jotka ovat yleisten sanojen listassa. Esimerkiksi merkkijono ”123” ja yleisten sanojen listasta sana ”salasana”, muodostavat salasanan ”salasana123”, jolla hyökkäävä ohjelma yrittää kirjautua järjestelmään.

Myös sähköisen kauppapaikan tapa hallinnoida salasanoja voi muodostaa turvallisuusriskin. Tällainen tilanne voi syntyä, jos järjestelmään murtaudutaan ja murtautuja saa kopioitua käyttäjien salasanatiedot. Tämä tilanteen toteutuminen vaatii hyökkääjältä huomattavaa tietämystä järjestelmästä, joten sen toteuttaminen on hankalampaa, mutta tällaisesta murrosta syntyvät vahingot voivat myös olla hyvin mittavat.

3.2.1.2 Turvaamismenetelmät

Salasanojen turvallisuuden parantamiseksi on esitetty monia eri tapoja. Tässä osiossa läpikäydään sellaisia yleisiä tapoja, joista ei tule loppukäyttäjälle kustannuksia, joten ne soveltuvat hyvin Internetissä toimivalle sähköisen kauppapaikalle. Yleisiä keinoja ovat sisäänkirjautumisen hankaloittaminen hyökkäävälle ohjelmalle, salasanan vaihtaminen sekä salasanojen suojaus järjestelmän tietovarastoissa.

Yleisinä suojaustapoina sanakirjahyökkäystä vastaan Pinkas & Sander (2002) mainitsevat viiveen lisäämisen ja lukituksen, jotka molemmat pyrkivät hankaloittamaan hyökkääjän sisäänkirjautumista. *Viiveen lisääminen* tarkoittaa sitä, että järjestelmä ei vastaa välittömästi sisäänkirjautumispyyntöön, vaan viivyttää sitä lyhyen ajanjakson. Ihmiselle sekunnin odottaminen sisäänkirjautumisvaiheessa ei yleensä ole kriittinen, mutta hyökkäävän järjestelmän toimintaa se hidastaa huomattavasti. Viiveen lisääminen ei kuitenkaan auta, jos hyökkäävälle ohjelmalle ei ole merkitystä millä käyttäjätunnuksella se kirjautuu järjestelmään. Tällöin hyökkäävä ohjelma voi yrittää sisäänkirjautumista eri käyttäjätunnuksella välittömästi sen jälkeen kun edellinen kirjautumisyritys on lähetetty järjestelmälle, eikä ohjelman tarvitse odottaa vastausta edelliseen kirjautumisyritykseen.

Lukituksessa järjestelmä sulkee käyttäjän tilin määräajaksi kun käyttäjän tunnuksilla yritetään kirjautua useita kertoja lyhyen ajan sisällä, väärää salasanaa käyttäen. Tällöin hyökkäävä ohjelma ei pysty kirjautumaan järjestelmään, vaikka käyttäjän salasana olisikin sanakirjahyökkäyksellä helposti murrettavissa. Lukitus tuo kuitenkin mukanaan uusia ongelmia (Pinkas & Sander, 2002). Hyökkäävä ohjelma voi tarkoituksella yrittää sisäänkirjautumista väärällä salasanalla niin kauan kunnes käyttäjän tili suljetaan. Tällöin myöskään tilin oikea käyttäjä ei pääse kirjautumaan järjestelmään, mikä tässä tapauksessa oli hyökkääjän tarkoituskin. Internetissä toimivissa huutokaupoissa tällaisilla hyökkäyksillä on estetty kilpailevan tarjouksen tekeminen. Lukitus aiheuttaa myös järjestelmän ylläpidolle lisäkustannuksia, kun käyttäjät, joiden tili on lukittuna, tarvitsevat asiakastukea.

Hyökkäävän ohjelman sisäänkirjautumista voidaan hankaloittaa myös käänteisellä Turingin testillä (Pinkas & Sander, 2002). *Käänteinen Turingin testi* on testi, joka yrittää selvittää onko kysymykseen vastaamassa ihminen vai tietokone. Käänteinen Turingin testi esittää kysymyksen käyttäjälle, johon ihmisen on helppo vastata oikein, mutta tietokoneelle hyvin vaikeaa. Vastausmahdollisuuksia täytyy myös olla niin suuri määrä, ettei vastauksen löytäminen arvaamalla ole todennäköistä. Esimerkkinä tällaisesta testistä Pinkas & Sander

(2002) mainitsevat tunnistussovelluksen, joka esittää käyttäjälle kuvan, jossa näkyy satunnainen kirjainjono. Käyttäjän on syötettävä tämä merkkijono ennen varsinaista järjestelmään sisäänkirjautumista. Kuvaa on käsitelty siten, etteivät kuvanlukuohjelmat pysty selvittämään mitä merkkejä kuvassa on. Tällöin ihmisen on helppo selvittää testi, mutta järjestelmään hyökkäävälle ohjelmalle testin läpäisy on hyvin epätodennäköistä. Kun tällainen tunnistusmenetelmä edeltää varsinaista järjestelmään kirjautumista, ei hyökkäävä ohjelma pääse suorittamaan sanakirjahyökkäystä tai raa'an voiman hyökkäystä kohdejärjestelmään. Näiden hyökkäysten poistumisen myötä ei järjestelmän myöskään tarvitse huolehtia käyttäjien tilien lukitsemisesta, mikä nostaa järjestelmän käyttömukavuutta.

Vaikka ohjelmallisten hyökkäysten onnistumista ja nopeutta voidaan rajoittaa, jää murrettavan salasanan etsimiseen riittävästi aikaa, jos salasana pysyy jatkuvasti samana. Tämän vuoksi Tulloch (2003) ehdottaa salasanojen säännöllistä ja usein tapahtuvaa vaihtamista. Käytetty järjestelmä, tässä tapauksessa sähköinen kauppapaikka, ohjaa käyttäjää siten, että se vaatii käyttäjän vaihtamaan salasanan aina tietyn ajanjakson kuluttua edellisestä salasanan vaihdosta. Järjestelmä valvoo tämän lisäksi, ettei käyttäjä valitse liian yksinkertaisia salasanoja. Tällöin järjestelmä varmistaa, että käytetyt salasanat ovat riittävän pitkiä ja sisältävät pelkkien kirjainten lisäksi myös numeroita ja erikoismerkkejä. Vaihtuvan salasanan etuna on se, että salasanaa vastaan hyökkäävällä ohjelmalla on vähemmän aikaa löytää oikea salasana ja vaikka hyökkääjä saisikin jonkin vanhentuneen salasanan selville, siitä ei olisi mitään hyötyä. Tämä tietysti vaatii myös sen, että jo kerran vanhentunutta salasanaa ei käytetä uudelleen salasanana.

Vaihtuvien salasanojen erikoistapaus on kertakäyttöinen salasana. *Kertakäyttöiset salasanat* toimivat siten, että käyttäjä syöttää eri salasanan joka kerran kun kirjautuu järjestelmään. Yksi tapa toteuttaa tällainen tunnistus on muodostaa etukäteen lista satunnaisista salasanoista ja toimittaa lista sitten käyttäjälle. Tästä listasta käyttäjä valitsee aina seuraavan käyttämättömän salasanan sisäänkirjautuessaan järjestelmään (Halevi & Krawczyk, 1999). Kun listan salasanat eivät ole käyttäjän itsensä muodostamia, niin salasanat eivät myöskään ole helposti murrettavissa sanakirjahyökkäyksellä. Toinen tapa on käyttäjän hallinnoima laite, joka muodostaa itsenäisesti salasanan järjestelmään kirjautumista varten (Summers & Bosworth, 2004). Laite on synkronoitu järjestelmän kanssa siten, että sekä tämä laite että järjestelmä muodostavat tahoillaan saman salasanan. Molempia tapoja rajoittava tekijä on kuitenkin se,

että käyttäjällä on oltava kirjautumishetkellä joko salasanalista tai salasanan muodostava laite käytettävissä.

Tämän lisäksi kertakäyttöisten salasanojen käyttö vaatii, että myös käytettävä järjestelmä todentaa identiteettinsä (Halevi & Krawczyk, 1999). Jos järjestelmän käyttö ei vaadi myös palvelun tunnistamista, voi hyökkääjä esiintyä käyttäjälle kohdejärjestelmänä ja saada siten haltuunsa käytetyn salasanan. Kertakäyttöiset salasanat eivät myöskään ole täysin turvallinen tunnistustapa. Koska käyttäjän tunnistus perustuu salasanalistaan tai salasanan muodostavaan laitteeseen, kuka tahansa jolla on hallussaan kyseinen lista tai laite, voi kirjautua järjestelmään.

Tunnistustapahtuman lisäksi tunnistustiedot voivat joutua ulkopuolisten käsiin kauppapaikkaan tehdyn tietomurron seurauksena. Tällöin hyökkääjä voi yhdellä kertaa päästä käsiksi kaikkien käyttäjien tunnistustietoihin ja siten aiheuttaa laajaa asiakaskuntaa koskevia vahinkoja. Tietovarastoon tallennettavien salasanojen turvallisuutta voidaan parantaa tallentamalla salasanasta tiivistearvo varsinaisen salasanan sijasta. Tällöin tiivistearvon joutuminen vääriin käsiin ei välittömästi paljasta varsinaista salasanaa. Tehokkain tapa löytää tiivistearvoa vastaava salasana on raa'an voiman hyökkäys, jonka tehokkuutta voidaan vähentää kahdella hyvin yleisesti käytetyllä tavalla (Halderman & al., 2005):

- Laskemalla ensin salasanasta tiivistearvo ja käyttämällä tätä tiivistearvoa uuden tiivistearvon syötteenä. Tiivistearvojen laskemista jatketaan ennalta määrätty määrä, jonka jälkeen viimeinen tiivistearvo tallennetaan tietovarastoon.
- Laskemalla talletettava tiivistearvo syötteestä, joka sisältää salasanan lisäksi jotain ylimääräistä dataa. Tämä ylimääräinen data täytyy olla siten talletettu, ettei hyökkääjä pääse dataan käsiksi edes tekemänsä tietomurron aikana. Data voi olla vaikka erillisessä järjestelmässä, joka huolehtii tiivistearvon laskemisesta.

Molemmat tavat hidastavat, lisääntyneen operaatiomäärän myötä, oikean salasanan etsimistä tiivistearvon avulla. Samalla kuitenkin hidastuu ja monimutkaistuu myös asiakkaiden sisäänkirjautuminen, kun käyttäjän tunnistus vaatii lisälaskentaa.

3.2.2 Tunnistus älykorttia käyttäen

Käyttäjätunnusta ja salasanaa turvallisempi tunnistusmenetelmä on älykortilla tunnistaminen, jonka toiminta perustuu julkisen avaimen menetelmään (Rinne, 2002). Älykortit voidaan jakaa niiden ominaisuuksien mukaan kahteen eri kategoriaan. Ensimmäiseen kategoriaan kuuluvat muistikortit, joista kerran niihin tallennettu tieto voidaan lukea, esimerkiksi magneettinauhaa käyttäen. Tällaisia kortteja ovat yleisesti käytetyt pankki- ja luottokortit. Toiseen kategoriaan kuuluvat prosessorin sisältävät kortit, jotka kykenevät tiedon tallennuksen lisäksi suorittamaan laskutoimituksia (Shelfer & Procaccino, 2002). Älykorteista juuri nämä prosessorin sisältävät kortit mahdollistavat käyttäjän luotettavan tunnistuksen.

Tunnistus tapahtuu käyttäjän henkilökohtaisella älykortilla ja ainoastaan käyttäjän tietämällä PIN-koodilla, joka on tallennettu myös älykortille. PIN-koodin lisäksi älykortille on tallennettu käyttäjän digitaalisia sertifikaatteja, joista käytetään myös nimitystä varmenne. Varmenne sisältää käyttäjän julkisen avaimen ja yksilöiviä tietoja sekä käyttäjästä että varmenteen myöntäjästä. Varmenteiden ja PIN-koodin lisäksi kortille on tallennettu käyttäjän yksityisiä avaimia (Rinne, 2002). Yleensä näitä yksityisiä avaimia on ainakin kaksi kappaletta, joista toista käytetään digitaalisiin allekirjoituksiin ja toista salaukseen sekä käyttäjän tunnistamiseen.

Rinteen (2002) mukaan useamman yksityisen avaimen tallentamiseen älykortille on kaksi hyvää syytä. Ensinnäkin, jos yksityisiä avaimia olisi vain yksi ja sitä käytettäisiin sekä allekirjoittamiseen että salaamiseen, niin käyttäjää saattaisi erehdyksessä allekirjoittaa dataa, jota ei ole tarkoitettu allekirjoitettavaksi. Toinen syy on se, että salaukseen käytettävistä avainpareista halutaan joissakin tapauksissa ottaa varmuuskopioita. Näitä varmuuskopiota käytetään yleistyvässä määrin yrityksissä, joissa sähköpostit salataan salausavaimilla, mutta halutaan samalla varmistaa, että sähköpostit voidaan lukea kun työntekijä on jostain syystä estynyt. Digitaaliseen allekirjoitukseen käytettävästä salausavaimesta ei pidä kuitenkaan missään tilanteessa ottaa varmuuskopiota, ettei tule epäselvyyksiä siitä, millä avaimella viestit on allekirjoitettu. Näistä syistä johtuen digitaaliseen allekirjoittamiseen käytetään eri avainta kuin viestien salaamiseen.

Älykortilla tunnistaminen tapahtuu siten, että käyttäjä asettaa älykortin tietokoneeseen asennettuun lukulaitteeseen ja syöttää PIN-koodin, joko tietokoneen tai lukulaitteen

näppäimistöltä. Mikäli syötetty arvo täsmää kortille tallennetun PIN-koodin kanssa, älykortti aktivoituu siten, että lukulaite pystyy lukemaan kortin julkisia tietoja (Väestörekisterikeskus, 2005). Tämän jälkeen lukulaite hakee kortilta käyttäjän varmenteen ja lähettää sen tunnistusta pyytävälle palvelulle. Palvelu varmistaa käyttäjän varmenteen oikeellisuuden ja myöntää käyttöoikeuden. Varmenteen tarkistus perustuu varmentajan digitaaliseen allekirjoitukseen, kuten julkisen avaimen menetelmässä on tapana. Suomessa älykorttien varmentajana toimii Väestörekisterikeskus.

Älykortilla tehtävää tunnistamista pidetään tunnistustapana, johon liittyy vain vähän turvallisuusuhkia. Älykortin tiedonsalausmenetelmät ovat vahvoiksi todettuja ja nykyisten, hyvin suojattujen älykorttien tietoja on fyysisestikin hyvin vaikea lukea luvattomasti. Koska älykortin tekniikkaa vastaan hyökkääminen menestyksekkäästi on hyvin vaikeaa, älykortin tunnistuksen heikoimpana lenkkinä on sen haltija (Rinne, 2002). Mikäli älykortin haltija kadottaa kortin yhdessä PIN-koodin kanssa, voi kortin löytäjä aiheuttaa monenlaista vahinkoa alkuperäiselle käyttäjälle. Tällöin kortin löytäjä voi esiintyä kortin alkuperäisenä haltijana ja aiheuttaa myös suoraa taloudellista vahinkoa, jos korttia voi käyttää tunnistuksen lisäksi maksutapahtumissa. Siksi PIN-koodi tulee aina säilyttää erillään älykortista. Älykorttia ei voi käyttää ilman PIN-koodia, joten pelkän kortin hukkaaminen ei siten luo kovin suurta ja välitöntä turvallisuusriskiä. Samoin PIN-koodin joutuminen vääriin käsiin ei aiheuta ongelmia niin kauan kuin kortti on sen oikealla haltijalla. Hukkaamistilanteessa käyttäjän tulee kuitenkin tehdä katoamisilmoitus mahdollisimman nopeasti, jotta kortin liikkeellelaskija voi kuolettaa kortin ja siten estää kortin väärinkäyttö.

Älykortti ei hyvästä turvallisuustasosta huolimatta ole kuitenkaan kovin suosittu tunnistusväline sähköisissä kauppapaikoissa. Syynä siihen ovat lähinnä kortin hankkimisen ja lukulaitteen aiheuttamat lisäkustannukset sekä lukulaitteen pakollisuus. Tällä hetkellä lukulaite on vain harvoissa tietokoneissa, joten palvelun saatavuus tällä tunnistustavalla rajoittuu vain näihin tietokoneisiin. Älykortteja on käytetty viime vuosiin asti pääasiallisesti vain julkisen hallinnon järjestelmissä, mutta tilanne on parantumassa vähitellen. Älykortteja voi käyttää tällä hetkellä esimerkiksi Internetin pankkipalveluissa, sähköpostien salaamisessa ja erilaisissa maksutapahtumissa (Rinne, 2002). Kaikissa näissä tapauksissa voidaan käyttäjän tunnistus tehdä älykortilla turvallisemmin kuin käyttäjätunnusta ja salasanaa käyttäen.

3.2.3 Yhteenveto tunnistusmenetelmistä

Sähköisessä kauppapaikassa on omat erikoispiirteensä, jotka rajoittavat tapoja, joilla käyttäjän tunnistaminen voidaan ja kannattaa toteuttaa. Sähköisen kauppapaikan käytölle ei haluta asettaa suurta kynnystä, mistä syystä monimutkaisimmat, vaikka usein turvallisemmat, tekniikat eivät ole laajalti käytössä.

Yleisin käyttäjän tunnistustapa sähköisissä kauppapaikoissa on tällä hetkellä salasanaan perustuva tunnistus. Salasanat ovat kuitenkin alttiita järjestelmällisille hyökkäyksille sekä käyttäjän huolimattomuudelle. Näistä syistä salasanojen turvallisuutta on pyritty vahvistamaan usein eri keinoin, vaihtelevin tuloksin. Kehittyneet tekniikat ja käyttäjien koulutus parantavat salasanojen turvallisuutta, mutta turvallisuuden taso on silti matalampi kuin kehittyneemmissä tunnistusmenetelmissä.

Tästä syystä korkean turvallisuuden asteen vaativat sovellukset käyttävät käyttäjän tunnistamiseen muita menetelmiä, kuten biometrisia tunnisteita ja älykortteja. Sähköiseen kaupankäyntiin ei biometristen tunnisteiden käyttö ole vielä levinnyt, mutta älykortin käyttö on hitaasti yleistymässä. Älykortin käyttö lisää kuitenkin kustannuksia ja vaatii että käyttäjän koneeseen on asennettu älykortin lisälaite.

Eri tunnistusmenetelmät ovat turvallisuudeltaan, kustannuksiltaan ja käytettävyydeltään hyvin erilaisia. Internetissä toimivalle kauppapaikalle täytyy löytää jonkinlainen kompromissi näiden ominaisuuksien välitä, jottei järjestelmään sisäänkirjautuminen nouse liian suureksi esteeksi käyttäjille, mutta myös turvallisuuden taso on riittävä.

3.3 Käyttäjän valtuuttaminen

Käyttäjän valtuuttamisella (authorization) tarkoitetaan käyttöoikeuksien myöntämistä järjestelmään sisäänkirjautuneelle käyttäjälle. Käyttäjän valtuuttamisen tarkoitus on estää luvaton pääsy järjestelmän toimintoihin sekä sellaisten tietojen käsittely, joihin käyttäjälle ei ole myönnetty oikeutta (Hartman & al., 2003). Käyttöoikedet voivat kohdistua järjestelmän osiin, kuten WWW-sivuihin tai tietokannan tietoihin, tai järjestelmän toimintoihin. Käyttäjältä

voidaan siten estää pääsy joihin järjestelmän osiin ja toisaalta käyttäjälle voidaan antaa oikeus suorittaa joitain järjestelmän toimintoja, mutta samalla estää toisia toimintoja.

Tulloch (2003) jakaa käyttäjän valtuuttamisen kahteen eri kategoriaan: Resurssikohtaiseen ja roolipohjaiseen valtuuttamiseen. *Resurssikohtainen valtuutus* määrittää ketkä käyttäjä kyseistä resurssia saavat käyttää ja mitä toimintoja heillä on oikeus siihen kohdistaa. Esimerkiksi järjestelmään tallennettu tiedosto voi olla resurssi, johon toisilla käyttäjillä on pelkästään lukuoikeus, kun taas toiset käyttäjät omaavat myös kirjoitusoikeuden. Resurssien valtuutustiedot kirjoitetaan käyttöoikeuslistoihin, jotka kasvavat käyttäjämäärän kasvaessa. Käyttöoikeuslistaa täytyy myös jatkuvasti päivittää, kun yksittäisen käyttäjän oikeuksissa tapahtuu muutoksia (Joshi & al., 2001). Tämän vuoksi resurssikohtainen valtuutus on työläs ylläpidettävä kun järjestelmällä on laaja käyttäjäkunta.

Roolipohjainen valtuutus jakaa käyttäjät loogisiin ryhmiin, rooleihin. Nämä roolit liitetään resursseihin halutuilla toiminto-oikeuksilla siten, että jokaiseen roolin oikeudet ovat erikseen määritelty. Tällöin käyttäjä ei liity suoraan resurssiin vaan rooliin ja kaikilla samaan rooliin kuuluvilla käyttäjillä on samat käyttöoikeudet järjestelmän tietoihin ja toimintoihin. Käyttäjä voi myös kuulua useisiin eri rooleihin, jotka yhdessä muodostavat käyttäjän oikeudet. Roolipohjainen valtuutus vaatii resurssipohjaista valtuutusta vähemmän ylläpitotyötä (Park & al., 2001), joten se soveltuu resurssikohtaista valtuutusta paremmin järjestelmiin, joissa on suuria käyttäjämääriä.

Sähköinen kauppapaikka, mikäli siihen on toteutettu käyttäjän tunnistus, vaatii samalla myös käyttäjän valtuuttamisen. Valtuuttamisella voidaan antaa järjestelmää ylläpitäville henkilöille erilaisia oikeuksia kuin kauppapaikan asiakkaille. Myös erilaisille asiakasryhmille voidaan antaa oikeus eri toimintoihin valtuutusta käyttäen. Tällä hetkellä suosituin valtuutustapa Internetin kauppapaikoissa on roolipohjainen valtuutus, joka soveltuu tehtäväänsä hyvin.

4 SÄHKÖISEN KAUPANKÄYNNIN MAKSUTAVAT

Sähköiseen kaupankäyntiin oleellisena osana kuuluu maksun suorittaminen asiakkaalta palveluntarjoajalle. Maksutietojen välittäminen avoimessa tietoverkossa vaatii luonnollisesti vahvaa tietojen salausta ja tietojen muuttumattomuuden varmistusta. Tässä luvussa perehdytään yleisimpiin maksutapoihin sähköisessä kaupankäynnissä.

4.1 Luottokortilla maksaminen

Luottokortilla maksaminen on yksi yleisimmistä maksutavoista sähköisessä kaupankäynnissä (Claessens & al., 2003). Tällöin kauppapaikan asiakas lähettää WWW-selaimensa välityksellä luottokortin tiedot sähköiselle kauppapaikalle, joka lähettää laskun tiedot luottokortin myöntäjälle, joka puolestaan lisää laskun asiakkaan tilille maksettavaksi. Asiakkaan ja kauppapaikan välinen yhteys on suojattu TLS-protokollalla, joten luottokortin tietoihin ei tässä vaiheessa kukaan ulkopuolinen pääse käsiksi.

TLS-protokolla kuitenkin vain suojaa käytetyn tietoliikenneyhteyden, mutta ei ratkaise maksamiseen liittyviä kaikkia muita turvallisuusongelmia (Khu-smith & Mitchell, 2002). TLS-protokolla mahdollistaa käyttäjän tunnistuksen, muttei vaadi sitä, jolloin luottokorttia käyttävä asiakas ei välttämättä ole luottokortin oikea haltija. Lisäksi sähköisen kauppapaikan tietovarastoon salaamattomana tallentuvat luottokortin tiedot ovat alttiita tietomurroille.

4.1.1 SET-protokolla

Internetissä tapahtuvaan luottokortilla maksamiseen kehitettiin 1990-luvulla useita menetelmiä maksutapahtuman turvaamiseksi. Vuonna 1996 suuret luottokorttiyhtiöt Visa ja MasterCard sekä tietotekniikkayritykset Netscape, IBM ja Microsoft yhdistivät menetelmänsä SET-protokollaksi (O'Mahony & al., 2001). *SET* (Secure Electronic Transactions) on protokolla luottokortilla maksamiseen, joka on kehitetty juuri sähköistä kaupankäyntiä varten. Pääpiirteissään maksutapahtuma SET-protokollaa käyttäen etenee seuraavasti (O'Mahony & al., 2001):

- Maksutapahtumassa toimii kolme eri osapuolta: Luottokorttia käyttävä asiakas, sähköisen kauppapaikan kauppias ja luottokortin myöntäjä. Kaikilla osapuolilla on oma digitaalinen sertifikaattinsa, joita käytetään osapuolten tunnistamisessa ja tiedon salaamisessa. Kaikki osapuolten välillä liikkuva tieto salataan julkisen avaimen menetelmää käyttäen. Asiakkaan sertifikaatti on tallennettu työasemalle asennettuun sähköiseen lompakkoon. Lisäksi kauppiaan ja luottokortin myöntäjän välillä toimii yhdyskäytävä, johon kauppias lähettää SET-protokollan mukaiset viestit.
- Asiakas hyväksyy tilauksensa ja käynnistää maksutapahtuman lähettämällä kauppiaalle maksutapahtuman aloitusviestin. Tähän kauppias vastaa lähettämällä asiakkaalle oman ja luottokortin myöntäjän digitaalisen sertifikaatin sekä digitaalisesti allekirjoittamansa määrämuotoisen viestin, jonka avulla asiakas voi varmistua kauppiaan identiteetistä.
- Tämän jälkeen asiakas lähettää kauppiaalle kaksiosaisen viestin, joka koostuu tilaustiedoista ja maksutiedoista. Ensimmäisen osan tilaustiedot on salattu kauppiaan julkisella avaimella, joten vain kauppias voi lukea nämä tiedot. Toisen osan sisältämät maksutiedot puolestaan on salattu luottokortin myöntäjän julkisella avaimella, joten kauppias ei pysty lukemaan maksutietoja. Molemmat osat sisältävät myös asiakkaan muodostaman kaksoisallekirjoituksen. *Kaksoisallekirjoitus* on tilaustietojen tiivisteestä ja maksutietojen tiivisteestä laskettu tarkistusarvo, jonka asiakas on digitaalisesti allekirjoittanut yksityisellä avaimellaan. Kaksoisallekirjoituksen mukana kulkee myös asiakkaan digitaalinen sertifikaatti, jonka avulla allekirjoitus voidaan todentaa.
- Kauppias avaa viestin tilaustiedot ja varmistaa kaksoisallekirjoituksen aitouden, minkä jälkeen kauppias lähettää luottokortin myöntäjälle viestin, jolla varmistetaan että asiakkaalla on oikeus käyttää luottokorttia. Viestissä kulkee mukana asiakkaan lähettämät maksutiedot, joiden perusteella varmistaminen tehdään. Samalla kortinmyöntäjä varmistaa kaksoisallekirjoituksen avulla, että asiakkaan lähettämät tiedot täsmäävät kauppiaan tietojen kanssa, joten kauppias ei pysty laskuttamaan eri summaa kuin mikä asiakkaalle on ilmoitettu.
- Varmistuksen jälkeen kauppias hyväksyy maksutapahtuman ja lähettää siitä tiedon asiakkaalle. Asiakas maksaa ostoksensa seuraavassa luottokorttilaskussa.
- Maksutapahtuman hyväksymisen jälkeen kauppias lähettää luottokortin myöntäjälle pyynnön maksun siirtämisestä omalle tililleen. Pyyntö voi myös sisältää usean

ostotapahtuman maksut. Luottokortin myöntäjä varmistaa pyynnön tiedot siirtää maksettavan summan kauppiaan tilille.

SET-protokollaa pidetään hyvin turvallisena maksumenetelmänä, mutta sillä on paljon muita heikkouksia. Yksi näistä heikkouksista on se, että SET-protokolla vaatii toimiakseen useiden eri komponenttien asentamista (Khu-smith & Mitchell, 2002;O'Mahony & al., 2001). Käyttäjän työasemalle tulee asentaa sähköinen lompakko, joka toimii yhdessä selaimen kanssa, sähköiseen kauppapaikkaan täytyy asentaa SET-protokollaa tukevat ohjelmistot ja myös yhteyskäytävän luominen kauppapaikan ja luottokorttiyhtiön välillä lisää järjestelmän rakentamistyötä.

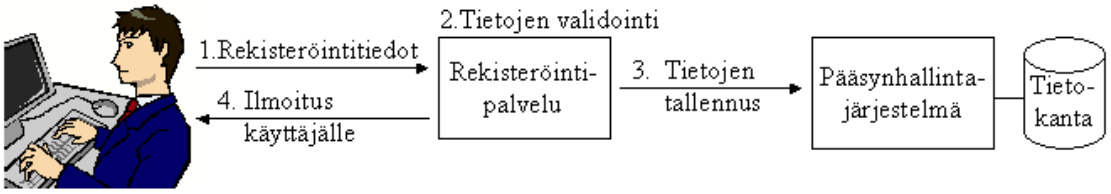
Komponenttien asentamisen lisäksi SET-protokollaa pidetään hyvin monimutkaisena ja huonosti Internetin maksupalveluihin soveltuvana (O'Mahony & al., 2001). Kun SET-protokolla julkaisu tapahtui 1990-luvun lopulla, sen käytön odotettiin parissa vuodessa leviävän laajalti. Näin ei kuitenkaan käynyt, edellä mainituista ongelmista johtuen. Tästä syystä luottokortilla maksamiseen on viime vuosina haettu keinoja yhdistää riittävä turvallisuus ja käytännöllisyys.

4.1.2 Nykyiset järjestelmät luottokortilla maksamiseen

Tällä hetkellä suurimmat luottokorttiyhtiöt Visa ja MasterCard käyttävät sähköisessä kaupankäynnissä omia maailmanlaajuisia maksumenetelmiään, jotka tosin ovat hyvin samankaltaisia toiminnallisuudeltaan. Molemmat menetelmät salaavat välitettävän tiedon TLS-protokollalla ja molempiin on myös lisätty lisätoiminto, jonka tarkoitus on varmistaa, että sähköisen kauppapaikan asiakas on luottokortin oikea haltija. Visa käyttää järjestelmästään nimeä Verified by Visa (Visa, 2005a) ja MasterCard nimeä SecureCode (MasterCard, 2005). Tässä tutkielmassa perehdytään Visan tapaan toteuttaa käyttäjän tunnistus maksutapahtuman yhteydessä.

Visan Verified by Visa –maksujärjestelmän käyttäjän tunnistus toteutetaan Visan 3-D Secure –protokollalla (Visa, 2005b). *3-D Secure* koostuu kahdesta eri osasta: Rekisteröitymisestä ja käyttäjän tunnistamisesta. Jokaisen Verified by Visa -järjestelmän asiakkaan täytyy rekisteröityä ennen kuin sähköinen kaupankäynti luottokorttia käyttäen on mahdollista.

Rekisteröinti tapahtuu luottokortin myöntäjän WWW-sivuilla. Rekisteröinti ei vaadi asiakasta asentamaan työasemalleen mitään lisälaitteita tai –ohjelmia, mutta Visan älykortin käyttö lukulaitteen välityksellä on myös mahdollista. Rekisteröintiprosessissa käyttäjältä kysytään luottokortin numero sekä henkilökohtaisia tietoja, joiden perusteella rekisteröintijärjestelmä varmistuu käyttäjän identiteettistä. Rekisteröinnin yhteydessä käyttäjä valitsee myös tavan, jolla hänet tunnistetaan ostotapahtuman yhteydessä (Visa, 2005b). Tunnistaminen voi tapahtua älykortilla tai salasanalla sekä Visan niin määritellessä, myös muilla tavoin. Tunnistustavat vaihtelevat maantieteellisestä alueesta ja sovelluksen käyttötarkoituksesta riippuen. Jos tunnistusmenetelmäksi valitaan salasanaan perustuva tunnistus, niin käyttäjä syöttää rekisteröinnin yhteydessä salasanan, jota käytetään ostotapahtuman yhteydessä varmistamaan käyttäjän identiteetti. Tietojen syöttämisen jälkeen rekisteröintitiedot tallennetaan luottokortin myöntäjän hallinnoimaan pääsynhallintajärjestelmään.



Kuva 4: Käyttäjän rekisteröityminen Visan maksujärjestelmään.

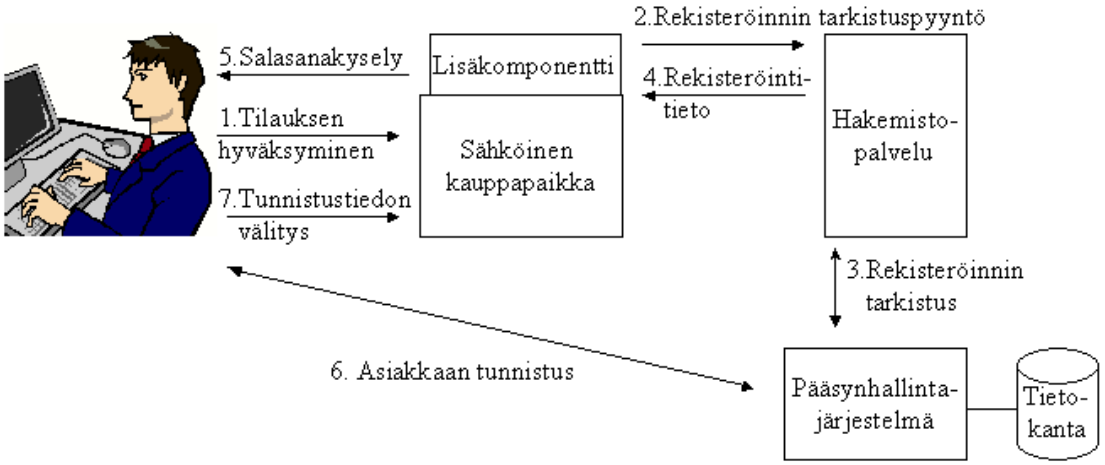
Käyttäjän tunnistaminen muodostaa 3-D Securen toisen osan ja se vaatii lisäasennuksia sähköiseen kauppapaikkaan. Kauppapaikkaan täytyy asentaa kauppiaan lisäkomponentti, jota järjestelmä käyttää käyttäjän tunnistamisen yhteydessä. Lisäksi kauppiaan tulee rekisteröityä Visan palvelun käyttäjäksi ja hankkia digitaaliset sertifikaatit kauppapaikan ja Visan järjestelmien välistä tiedonsiirron salaamista varten.

Varsinainen tunnistusprosessi alkaa siitä kun käyttäjä on valinnut ostettavat tuotteet ja hyväksyy tilauksen (Visa, 2005b). Tilauksen hyväksymisen yhteydessä käyttäjä myös syöttää luottokortin numeron. Kauppapaikka vastaanottaa asiakkaan hyväksymisilmoituksen ja välittää kauppapaikkaan asennetulle lisäkomponenttille asiakkaan tunnistuspyynnön, jonka mukana kulkee luottokortin numero. Lisäkomponentti lähettää Visan järjestelmässä olevalle hakemistopalvelulle pyynnön tarkistaa, onko kyseinen luottokortti rekisteröity Verified by Visa-palveluun. Hakemistopalvelu tarkistaa rekisteröintitiedon luottokortin myöntäjän

pääsynhallintajärjestelmästä, johon asiakkaan rekisteröintitiedot on tallennettu. Tämän jälkeen hakemistopalvelu palauttaa rekisteröintitiedon kauppapaikan lisäkomponentille, joka jatkaa tunnistusprosessia. Jos asiakas ei ollut rekisteröitynyt, kauppapaikka voi yrittää jotain muuta tunnistusmenetelmää.

Mikäli rekisteröinti on tehty, avaa lisäkomponentti asiakkaalle uuden ikkunan, jossa asiakkaan tunnistaminen tapahtuu. Tunnistus suoritetaan rekisteröinnin yhteydessä valittua menetelmää käyttäen. Tunnistustiedot välittyvät avautuneelta ikkunalta suoraan luottokortin myöntäjän pääsynhallintajärjestelmään, joten kauppias ei pääse sitä näkemään. Pääsynhallintajärjestelmä tarkistaa tiedot, muodostaa vastausviestin ja allekirjoittaa sen digitaalisesti.

Vastausviesti palautuu asiakkaan WWW-selaimen kautta kauppapaikan lisäkomponentille, joka varmistaa viestin aitouden. Jos viesti sisältää tiedon tunnistuksen onnistumisesta, lisäkomponentti palauttaa siitä tiedon kauppapaikka-järjestelmälle, joka voi jatkaa maksutapahtumaa.



Kuva 5: Käyttäjän tunnistus luottokorttimaksun yhteydessä.

Visan kehittämä 3-D Secure -protokolla vahvistaa käyttäjän tunnistusta luottokorttimaksun yhteydessä, koska se vaatii kortinhaltijalta tunnistustietoa ennen maksun hyväksymistä. Luottokortin hukkaaminen yhdessä PIN-koodin tai rekisteröinnissä syötetyn salasanan kanssa vaarantaa tämänkin tunnistuksen turvallisuuden, mutta riski on pienempi kuin pelkän

luottokortin numeron perusteella tapahtuvassa tunnistuksessa. Joten heikkoutensa tässäkin tunnistustavassa on, mutta 3-D Secure on kuitenkin kohtuullisen hyvä kompromissi käytettävyyden ja turvallisuuden välillä. Muutama vuosi sitten suurin ongelma Internetissä tapahtuvassa kaupankäynnissä oli juuri käyttäjän luotettava tunnistaminen, mutta 3-D Securen avulla Visa odottaa tunnistamiseen liittyvien väärinkäytöksien huomattavasti vähentyvän.

Verified by Visa –järjestelmää voi nykyisin käyttää myös verkkopankkien tunnuksilla, kunhan tunnukset on liitetty järjestelmään (Visa, 2005c). Tämä helpottaa palvelun käyttöönottoa, kun erillistä salasanaa ei enää tarvita. Verified by Visa-järjestelmä onkin levinnyt laajalti sähköisten kauppapaikkojen tunnistusmenetelmäksi.

4.2 Verkkopankin käyttäminen

Maksutapahtuma sähköisessä kauppapaikassa voidaan toteuttaa myös Internetissä toimivien verkkopankkien kautta. Pankkien välisiin tilisiirtoihin on kehitetty useita erilaisia menetelmiä, jotka soveltuvat erilaisiin käyttötapauksiin. Menetelmän turvallisuus, käytettävyys ja asiakkaiden hyväksyntä ratkaisevat, mitkä näistä menetelmistä menestyvät parhaiten (O'Mahony & al., 2001). Tässä tutkielmassa keskitytään siihen miten Suomessa toimivia verkkopankkeja voidaan käyttää sähköisen kauppapaikan maksutapahtumassa.

Useat Suomessa toimivat liikepankit ovat toteuttaneet verkkopankkeihinsa toiminnon, jolla sähköisen kauppapaikan maksutapahtuma voidaan suorittaa. Tällöin kauppapaikan WWW-sivuilta siirrytään väliaikaisesti verkkopankin sivuille maksamaan kauppapaikassa valitut ostokset ja palataan takaisin kauppapaikan sivuille. Verkkopankin sivuilla käydessään asiakas syöttää verkkopankin tunnuksensa, joiden avulla asiakas tunnistetaan, jotta laskutus kohdistuu oikeaan pankkitiliin.

Pankeista esimerkiksi Sampo ja Osuuspankki ovat dokumentoineet WWW-sivuillaan sähköisen kauppapaikan liittämisen verkkopankkiin. Toteutustavat ja välitettävät tiedot ovat hyvin samankaltaisia, joitakin eroja tietosisällöissä kuitenkin on. Tässä tutkielmassa läpikäydään Sampo-pankin tapa toteuttaa verkkomaksaminen sähköiseen kauppapaikkaan.

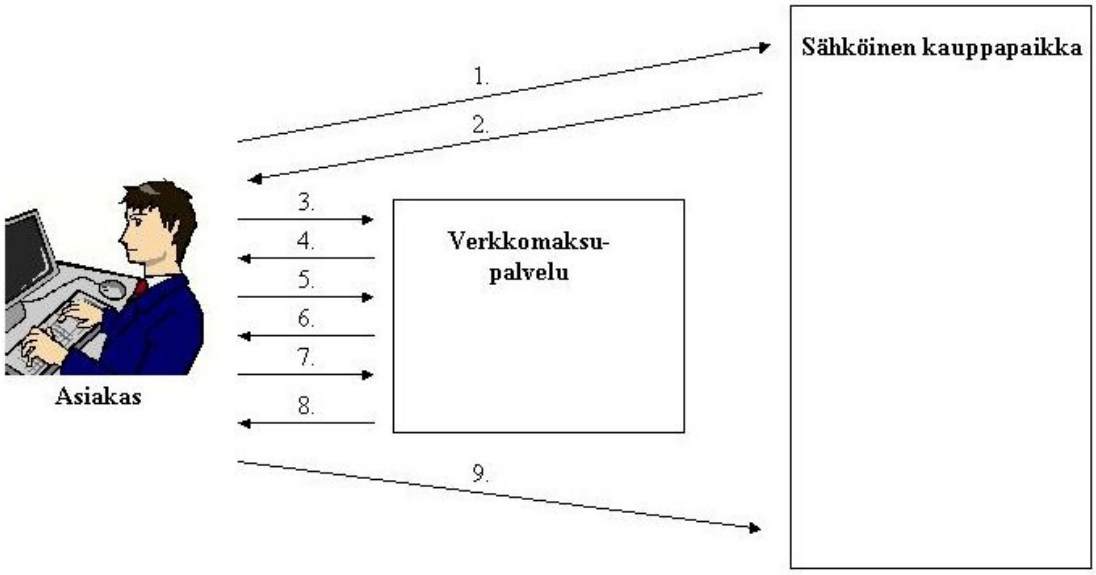
Jotta kauppapaikka voisi käyttää verkkopankin palvelua, tulee kauppapaikan omistajalla, kauppiaalla, olla tili siinä pankissa, jota maksamisessa aiotaan käyttää. Jos kauppiaalla on tili useammassa pankissa ja niitä kaikkia halutaan käyttää maksukanavana, niin kauppapaikkaan joudutaan rakentamaan oma liittymä jokaisen pankin verkkopalveluun. Näin eri pankkien asiakkaat voivat käyttää samaa kauppapaikkaa ja ainoastaan maksutapahtuma ohjautuu eri pankkiin.

Verkkopankin käyttäminen kauppapaikan yhteydessä vaatii myös, että kauppias on tehnyt tästä palvelusta sopimuksen pankin kanssa. Sopimuksen myöntämisen yhteydessä pankki myöntää kauppapaikalle kauppatunnuksen, eräänlaisen asiakasnumeron, joka kulkee jokaisessa maksutapahtumassa mukana. Pankit myös veloittavat sopimuksen tekemistä palvelumaksun. Maksutapahtuma verkkopankissa etenee kuvan 6 mukaisesti (Sampo Pankki, 2005):

1. Asiakas hyväksyy tilauksen ja siirtyy maksamaan ostoksensa.
2. Sähköinen kauppapaikka palauttaa asiakkaalle maksupyynnön, joka sisältää ostotapahtuman tiedot. Maksupyyntö sisältää kauppapaikan luomat maksutiedot verkkopankeille, joihin kauppapaikka on liittynyt asiakkaaksi.
3. Asiakas valitsee maksutavaksi verkkomaksun, jolloin maksutiedot välittyvät verkkomaksupalveluun, valitun pankin palvelimelle. Samalla välittyy tieto paluuosoitteesta, johon pankin tulee suoritus ohjata maksutapahtuman jälkeen. Pankki tarkistaa maksutietojen eheyden ja oikeellisuuden sekä kauppapaikan verkkopalvelusopimuksen.
4. Jos tiedot ovat virheettömät, verkkomaksupalvelu lähettää asiakkaalle tunnistuspyynnön.
5. Asiakas lähettää verkkomaksupalveluun asiakasnumeronsa ja salasanan, joiden perusteella asiakas voidaan tunnistaa. Nämä tiedot ovat samoja, joita asiakas käyttää verkkopankissa asioidessaan.
6. Verkkomaksupalvelu varmistaa asiakkaan henkilöllisyyden, muodostaa laskun tiedot ja lähettää ne asiakkaalle hyväksyttäväksi.
7. Asiakas hyväksyy laskun maksettavaksi.
8. Verkkomaksupalvelu vastaanottaa hyväksymistiedon ja suorittaa tilisiirron asiakkaan tililtä sähköisen kauppapaikan tilille. Tämän jälkeen palvelu lähettää asiakkaalle

tiedon maksutapahtuman onnistumisesta, siihen liittyvän tarkistustiedon sekä tiedon siirryttävästä paluuosoitteesta.

9. Asiakas vastaanottaa tiedon maksun hyväksymisestä, jonka jälkeen WWW-selain palaa sähköiseen kauppapaikkaan, paluuosoitteen määräämälle sivulle. Sähköinen kauppapaikka varmistaa maksutapahtuman onnistumisen tarkistustiedon avulla ja toimittaa asiakkaalle tilauksen tuotteet.



Kuva 6: Verkkopankissa maksaminen.

Verkkopankin käyttäminen maksutapahtumassa perustuu julkisen avaimen järjestelmään, joten maksutapahtuman tiedonvälitystä voidaan pitää riittävän turvallisena. Välitettävä tieto kulkee osapuolten välillä salattua yhteyttä käyttäen ja verkkomaksupalvelu vaatii sekä asiakasta että kauppapaikkaa todentamaan identiteettinsä maksutapahtuman aikana (Sampo Pankki, 2005). Tietojen muuttumattomuus varmistetaan maksutiedoista laskettavien tiivistearvojen avulla, joten asiakas ja kauppapaikka eivät voi muiden osapuolten huomaamatta muuttaa maksutietoja. Myöskään asiakkaan tunnistustiedot eivät koskaan kulje sähköisen kauppapaikan kautta, joten tiedot eivät tallennu kauppapaikan tietovarastoihin, jossa ne olisivat alttiina tietomurroille. Pankkien tietovarastoissa tunnistustiedot tietenkin ovat, mutta mielestäni pankkien tietojärjestelmiltä voidaan odottaa suurempaa turvallisuustasoa kuin Internetissä toimivalta kauppapaikalta.

Koska verkkomaksupalvelun ja sähköisen kauppapaikan turvallisuus on hyvä maksutapahtuman aikana, suurin riskitekijä yksittäisen maksutapahtuman aikana on käyttäjän tunnistus, joka perustuu asiakkaan asiakasnumeroon ja salasanaan. Näiden tietojen paljastuminen mahdollistaa sen, että joku ulkopuolinen voi maksaa ostoksensa toisen asiakkaan pankkitililtä. Siten salasanan tai salasanalistan turvallinen säilytys on tämän maksutavan oleellinen turvallisuutta parantava tekijä.

Verkkomaksaminen sähköisen pankkipalvelun kautta on hyvin suosittu maksutapa, mutta vaatii pankin asiakkuutta sekä sähköiseltä kauppapaikalta että kauppapaikan asiakkaalta. Tämä maksutapa ei vaadi asiakkaalta lisäasennuksia työasemalleen ja sen turvallisuusaste on hyvä. Maksutapa on käytössä mm. Joensuun yliopiston ilmoittautumisjärjestelmässä, jossa opiskelija voi ilmoittautumisen yhteydessä maksaa ylioppilaskunnan maksun verkkopankkitunnuksiaan käyttäen.

4.3 Sähköinen raha

Sähköinen raha on yleinen maksutapa pienten maksujen suorittamiseen asiakkaan ja kauppiaan välillä. Sähköisen rahan käyttö alkaa siten, että asiakas lataa ensin rahaa erilliselle maksutilille tai fyysiselle laitteelle, kuten älykortille (O'Mahony & al., 2001). Rahan lataaminen tapahtuu asiakkaan omalta pankkitililtä tai erillisenä maksusuorituksena, jolloin raha voidaan siirtää esimerkiksi älykortille erillistä lukulaitetta käyttäen. Rahojen tallennuspaikasta käytetään nimityksiä sähköinen kukkaro tai sähköinen lompakko. Lataamisen jälkeen asiakas maksaa ostoksensa sähköisellä rahalla kauppapaikoissa, joissa on valmius siihen. Maksutapahtumassa asiakkaan sähköisestä lompakosta siirtyy suoritus kauppiaan lompakkoon, josta kauppias voi edelleen siirtää rahat pankkitililleen.

Sähköisen rahan käyttöön on kehitetty useita eri järjestelmiä, joiden ominaisuudet vaihtelevat laajalti. Eroja löytyy eritoten soveltuvuudessa pienten maksujen maksamiseen, osapuolille koituvista kustannuksista ja tietoturvan tasosta. Yksi monipuolisimmista maksujärjestelmistä on 1990-luvulla kehitetty CAFE-järjestelmä (Conditional Access for Europe), joka toteuttaa useimmat sähköiselle rahalle asetetut vaatimukset (O'Mahony & al., 2001):

- Kaikkien osapuolien turvallisuus tunnetuilla salausmenetelmillä toteutettuna.

- Asiakas voi maksaa ostoksensa ilman että kauppiaan tarvitsee ottaa maksutapahtuman aikana yhteyttä sähköisen rahan myöntäjään, joka useimmiten on pankki. Rahojen siirto kauppiaan tilille tapahtuu vasta maksutapahtuman päätyttyä. Tämä toimintatapa parantaa maksutapahtuman suorituskykyä.
- Samaa rahan käyttö useammin kuin kerran paljastaa väärinkäytön.
- Asiakas voi suorittaa maksun anonyyminä, jolloin asiakkaan henkilöllisyys ei paljastu kauppiaalle tai pankille. Henkilöllisyys paljastuu ainoastaan väärinkäyttötilanteessa, asiakkaan maksaessa samalla sähköisellä rahalla useampaan kertaan.
- Maksujärjestelmä mahdollistaa usean eri valuutan tallentamisen sähköiseen lompakkoon.
- Sähköisen lompakon hukkaaminen ei hävitä asiakkaan siihen tallentamia rahoja, vaan pankki pystyy palauttamaan ne asiakkaan avustuksella.

CAFE-järjestelmä sisältää paljon turvallisuutta ja käytettävyyttä lisääviä ominaisuuksia, mutta on kuitenkin vain yksi lukuisista sähköisen rahan maksumenetelmistä. Edellä mainittuja vaatimuksia voidaan mielestäni pitää kuitenkin hyvänä lähtökohtana käytännölliselle ja turvalliselle sähköisen rahan järjestelmälle.

Useat tämän hetken sähköisen rahan maksujärjestelmät perustuvat älykortin käyttämiseen maksuvälineenä ja maksaminen tapahtuu kauppiaan päätelaitteella, johon on erikseen asennettu turvallisuutta parantavia komponentteja. Tässä tutkielmassa läpikäydään kuitenkin maksutapoja, jotka soveltuvat sähköiseen kaupankäyntiin siten, että asiakas suorittaa ostamisen WWW-selaimen välityksellä, vaikkapa omalta kotikoneeltaan. Älykorttia voidaan hyödyntää myös tällaisissa maksutapahtumissa, mutta tällöin asiakas lähettää kauppiaalle useimmiten luottokortin numeron eikä sähköistä rahaa (Rinne, 2002). Seuraavaksi perehdymme Ecash-maksujärjestelmään, joka oli yksi ensimmäisistä sähköisen rahan järjestelmistä sekä suomalaiseen Digiraha-järjestelmään.

4.3.1 Ecash

Ecash on sähköisen rahan järjestelmä, joka kehitettiin vuonna 1995 DigiCash-yhtiön toimesta. Ecash-järjestelmän tavoitteena oli mahdollistaa sähköisen rahan käyttö siten, että asiakkaan henkilöllisyys ei paljastu missään vaiheessa maksutapahtumaa (O'Mahony & al., 2001).

Tällöin ei kauppias eikä myöskään pankki ole tietoinen maksutapahtuman suorittavasta tahosta.

Ecash-järjestelmässä rahan lataaminen tapahtuu asiakkaan tietokoneella olevalla lompakko-ohjelmalla, joka siirtää kolikkoja sähköisessä muodossa pankin ja asiakkaan välillä. Aluksi ohjelma muodostaa asiakkaan haluamat kolikot ja jokaiselle kolikolle satunnaisen sarjanumeron, jolla jokainen kolikko voidaan tunnistaa (O'Mahony & al., 2001). Sarjanumero valitaan suuresta numeroavaruudesta, jotta todennäköisyys sille, että joku toinen asiakas käyttäisi saman sarjanumeron omaavaa kolikkoa, olisi hyvin pieni.

Sarjanumeron muodostamisen jälkeen lompakko-ohjelma piilottaa sarjanumeron pankilta käyttämällä sokean allekirjoituksen menetelmää. *Sokealla allekirjoituksella* tarkoitetaan tiedon digitaalista allekirjoitusta, ilman että allekirjoittaja näkee todellista viestiä, jota on allekirjoittamassa (O'Mahony & al., 2001). Ecash-järjestelmän tapauksessa asiakkaan lompakko-ohjelma muodostaa allekirjoitettavan tiivisteen alkuperäisestä sarjanumerosta ja satunnaisluvusta ennen kolikoiden lähettämistä pankille. Tämän jälkeen pankki allekirjoittaa asiakkaan lähettämien kolikoiden tiedot pankin yksityisellä avaimella, veloittaa asiakkaan tiliä kolikoiden arvon verran ja palauttaa kolikot asiakkaan lompakko-ohjelmalle. Samalla pankki liittää kolikoihin viimeisen voimassaolopäivän. Lompakko-ohjelma tallentaa kolikot käyttäjän tietokoneelle ja poistaa samalla kolikoista sarjanumeron piilotuksen, jolloin kolikoiden tiedot ovat samat kuin jos pankki olisi allekirjoittanut kolikot ilman sokeaa allekirjoitusta. Jotta tämä onnistuisi, täytyy sarjanumeron sokeuttamisfunktion olla käänteinen pankin allekirjoitusfunktioon nähden.

Maksutapahtumassa kauppiaan pyytämään maksusuoritukseen asiakas vastaa lähettämällä maksusumman edestä sähköisessä muodossa olevia kolikoita sekä maksutietoja kuten pankkitunnuksen, maksun suuruuden ja tiivisteen, joka on muodostettu asiakkaan itse luomasta ja salassapidettävästä asiakastunnuksesta (O'Mahony & al., 2001). Kolikoiden summan tulee olla täsmälleen maksusumman suuruinen ja tarvittaessa asiakas voi ladata lompakkoonsa lisää kolikoita ennen maksutapahtuman etenemistä. Kauppiaalle lähetetyt kolikot asiakas on salannut pankin julkisella avaimella ennen niiden lähettämistä, joten kauppias tai kukaan ulkopuolinen ei voi muuttaa kolikoiden tietoja. Samalla kolikoiden mukana kulkee asiakkaan laskema tiiviste maksutiedoista.

Kun kauppiaan järjestelmä vastaanottaa asiakkaan maksuviestin, se varmistaa maksutietojen oikeellisuuden ja lähettää pankille maksupyynnön, joka sisältää kolikot ja maksutiedot (O'Mahony & al., 2001). Pankin järjestelmä avaa kolikoiden salauksen ja varmistaa että kauppiaan ja asiakkaan hyväksymät maksutiedot ovat samat. Varmistus tapahtuu laskemalla maksutiedoista tiiviste ja vertaamalla sitä asiakkaan lähettämään, kolikoiden mukana kulkevaan tiivisteeseen. Samalla pankki tarkistaa, että kolikoiden voimassaoloaika ei ole umpeutunut ja ettei kyseisiä kolikoiden sarjanumeroita jo löydy tietokannasta, johon maksutapahtumassa käytettyjen kolikoiden sarjanumerot aina tallennetaan. Sarjanumeroita vertaamalla voidaan estää kolikoiden käyttö useampaan kertaan. Koska tietokantaan kertyy nopeasti suuri määrä kolikoiden sarjanumeroita, järjestelmän täytyy toimintakykynsä säilyttääksen poistaa niitä säännöllisesti tietokannasta. Tämä toteutetaan siten, että tietokannasta poistetaan sellaisten kolikoiden tiedot, joiden voimassaoloaika on päättynyt, sillä tällaiset kolikot eivät enää kelpaa maksuvälineeksi.

Jos tarkistuksissa ei löydy väärinkäyttöä, pankki tallettaa tietokantaan kolikoiden sarjanumerot ja asiakastunnuksesta lasketun tiivisteen, joka tuli osana maksutietoja. Tiivisteen avulla asiakas voi todistaa suorittaneensa maksun, mikäli kauppias ei jostain syystä toimita ostoksia asiakkaalle. Tietojen tallennuksen jälkeen pankki hyvittää kauppiaan tiliä kolikoiden summan verran ja ilmoittaa kauppiaalle onnistuneesta maksutapahtumasta, minkä jälkeen kauppias toimittaa asiakkaalle ostokset.

Ecash-järjestelmä täyttää useimmat sähköiselle rahalle asetetut vaatimukset. Ensinnäkin, järjestelmän maksutapahtumaa voidaan pitää turvallisena, koska kolikot ja maksutiedot salataan julkisen avaimen menetelmällä. Samalla kolikoiden useampaan kertaan käyttäminen on estetty ja asiakkaan henkilöllisyys ei paljastu maksutapahtuman aikana.

Asiakkaan tiedot paljastuvat kuitenkin silloin jos pankki palauttaa asiakkaan hukkaamia kolikoita. Ecash mahdollistaa palauttamisen, jos asiakkaalla on talletettuna kolikoiden sarjanumeron piilottamiseen käytetty satunnaisluku (O'Mahony & al., 2001). Tällöin pankki, joka on tallettanut tiedot asiakkaan suorittamista sähköisen rahan latauksista, voi palauttaa digitaalisesti allekirjoittamansa kolikot asiakkaalle. Tämän jälkeen asiakkaan lompakko-ohjelma poistaa kolikoista sarjanumeron piilotuksen ja palauttaa kolikot pankille. Pankki tarkistaa mitkä kolikot on käytetty ja hyvittää loput kolikot asiakkaan tilille.

Ecash-järjestelmällä voidaan suorittaa hyvin pieniä maksuja ja järjestelmä tukee myös eri valuutoiden käyttöä (O'Mahony & al., 2001). Pientenkin maksujen vaatimat lukuisat viestinvälitykset ja salaustoiminnot heikentävät kuitenkin järjestelmän suorituskykyä. Samoin tietokantaan tallennettavat kolikoiden sarjanumerot ja niihin liittyvät tarkistukset nostavat järjestelmän ylläpitotöitä ja kustannuksia. Myös asiakkaalta järjestelmä vaatii toimenpiteitä lompakko-ohjelman asentamisen muodossa.

4.3.2 Digiraha

Digiraha on Suomessa toimiva sähköisen rahan järjestelmä, joka on suunniteltu pienten ostoksien maksamiseen Internetissä. Järjestelmän tuottaa Osuuspankki, mutta sen käyttäjiksi ovat liittyneet useimmat Suomessa toimivat pankit (Osuuspankki, 2005a). Digirahasta on pyritty tekemään mahdollisimman helppokäyttöinen, mutta samalla turvallinen järjestelmä.

Digirahan käyttö perustuu sähköiseen kukkaroon, jonka asiakas voi avata Digirahan WWW-sivuilla (Osuuspankki, 2005a). Avaamisen yhteydessä käyttäjä syöttää henkilötietonsa sekä kukkaron vastatilinä toimivan pankkitilin numeron. Järjestelmä tarkistaa henkilötiedot, avaa kukkaron ja luo käyttäjän tarvitsemat tunnukset sekä salasanat, jotka toimitetaan käyttäjälle postitse.

Avaamisen jälkeen käyttäjä voi siirtää rahaa pankkitililtä kukkaroonsa siten, että kukkarossa on maksimissaan 250 euroa, joka on palveluun määritelty maksimimäärä. Tämän jälkeen käyttäjä voi maksaa ostoksiaan sellaisissa sähköisissä kauppapaikoissa, jotka tukevat Digirahan käyttöä (Osuuspankki, 2005a). Sähköistä rahaa voi myös siirtää kukkarosta toiseen ja halutessaan käyttäjä voi siirtää rahaa myös takaisin omalle pankkitililleen. Täysin maksuton palvelu ei ole käyttäjälle, sillä Osuuspankki veloittaa kukkaronhaltijalta pientä kuukausimaksua, jos kukkaroa on käytetty kuukauden aikana. Maksun suuruus on tällä hetkellä 0,33 euroa kuukaudessa.

Kauppiaan näkökulmasta Digirahan liittäminen sähköisen kauppapaikan maksutavaksi on samankaltainen kuin verkkopankin liittäminen, mikä käsiteltiin aiemmin tässä luvussa. Kauppiaan täytyy tehdä verkkosopimus jossain osuuspankissa ja tämän lisäksi kauppiaan on tilattava Digirahan käyttöoikeus. Tämän jälkeen kauppiaan kukkaro on valmis käytettäväksi ja kauppias voi liittää Digirahan verkkokauppaansa.

Maksutapahtumassa asiakkaalla on useita vaihtoehtoja suorittaa maksu sähköisestä kukkarostaan, sillä Digiraha tukee kolmenlaisia maksutapauksia (Osuuspankki, 2005b):

- Ensimmäisessä tapauksessa asiakas painaa kauppapaikan WWW-sivulla näkyvää Digirahan kuvaketta, johon on liitetty ostosten maksutiedot. Kuvakkeen valinta välittää tiedot Digiraha-palvelimelle, joka pyytää asiakasta tunnistautumaan. Asiakas sisäänkirjautuu Digiraha-palveluun ja hyväksyy maksutiedot, minkä jälkeen maksutapahtuman suoritus palaa kauppiaan järjestelmään ja kauppias saa tiedon maksun hyväksymisestä. Kauppias tarkistaa maksun tiedot ja ilmoittaa asiakkaalle maksutapahtuman hyväksymisestä.
- Toinen vaihtoehto maksamiseen on mobiililaitteen käyttö. Maksutapahtumassa asiakas lähettää Digirahan palvelunumeroon SMS-viestin, joka sisältää käyttäjän tunnistustiedot, kauppiaan tunnuksen ja tuotetunnuksen. Digiraha tarkistaa tietojen oikeellisuuden ja lähettää kauppiaalle tuotteen ostopyynnön. Kauppiaan järjestelmä vastaa tuotteen hintatiedolla, jonka mukaisesti Digiraha veloittaa asiakasta. Veloituksen jälkeen Digiraha lähettää kauppiaalle ilmoituksen maksun onnistumisesta, minkä jälkeen kauppias voi luovuttaa ostoksen asiakkaalle.
- Kolmantena maksuvaihtoehtona on maksuvaltuutuksen luonti ja käyttö. Tässä tapauksessa asiakas antaa kauppiaalle maksuvaltuutuksen, jolloin kauppias voi lähettää veloituspyyntöjä suoraan Digirahalle, ilman että asiakas tunnistautuu maksutapahtumassa. Maksuvaltuutustieto kulkee tällöin veloituspyyntöjen mukana, jonka perusteella Digiraha voi suorittaa veloituksen asiakkaan kukkarosta. Digirahan määrityksen mukaan valtuutuksen käyttö edellyttää kuitenkin, että asiakas on luotettavasti tunnistautunut kauppiaan järjestelmässä ja että kauppias on sopinut valtuutuksen käytöstä Osuuspankin kanssa. Maksuvaltuutuksella suoritettavien maksujen arvoa on myös rajoitettu siten, että ostokset saavat maksaa enintään 10 euroa.

Sähköiselle rahalle asetettuja vaatimuksia Digiraha ei toteuta kattavasti. Tiedonvälitys kauppiaan järjestelmän ja Digirahan välillä kulkee suojatun TLS-yhteyden yli, joten siltä osin järjestelmää voidaan pitää turvallisena. Turvallisuudesta joudutaan mielestäni kuitenkin tinkimään asiakkaan kohdalla silloin kun asiakas on tehnyt maksuvaltuutuksen. Tämä on

mielestäni myös järjestelmän suurin turvallisuusongelma. Maksamista on tällöin kyllä rajoitettu tiettyyn arvoon ja kauppiaan odotetaan suorittavan asiakkaan tunnistus, mutta vaadittavaa tunnistusmenetelmää ei ole määritelty (Osuuspankki, 2005b). Jos asiakas on myös sopinut Digirahan käyttämisestä matkapuhelimellaan, niin puhelimen hukkaaminen saattaa vaarantaa sähköisen kukkaron rahat. Varsinkin jos asiakas on tallentanut kauppapaikan tai Digirahan tunnistustiedot puhelimen muistiin.

Myös vaatimus maksutapahtuman suorittamisesta asiakkaan ja kauppiaanvälillä siten, että tapahtuman aikana ei tarvitse ottaa yhteyttä sähköisen rahan järjestelmään, ei toteudu. Kaikissa Digirahan maksutapahtumissa kauppiaan järjestelmä kommunikoi Digiraha-järjestelmän kanssa, mikä hidastaa maksun hyväksymistä. Samoin vaatimukset useasta eri valuutasta ja asiakkaan anonyymiteetistä maksutapahtuman aikana jäävät Digirahassa toteutumatta.

Vaatimukset sähköisen rahan moneen kertaan käyttämisen huomaamisesta ja hukattujen rahojen palauttamisesta Digiraha kuitenkin toteuttaa. Asiakkaan rahat eivät ole asiakkaan hallinnoimalla fyysisellä laitteella, vaan erillisellä maksutilillä Digiraha-järjestelmässä. Maksutapahtumassa rahamäärä vähenee tililtä, joten samaa rahaa ei voi käyttää uudelleen. Rahojen hukkumista ei myöskään pääse tapahtumaan maksutiliä käytettäessä.

Digiraha toteuttaa sähköiselle kauppapaikalle asetettuja vaatimuksia vajavaisesti ja maksutapahtuman turvallisuus on kyseenalainen kun veloitus tapahtuu asiakkaan maksuvaltuutusta käyttäen. Turvallisuuden lisäksi maksutavoissa esiin nousee usein myös käytettävyys. Digiraha ei vaadi asiakkaalta erillisiä asennuksia ja se on helppo tapa hoitaa maksutapahtumia, joiden rahasummat ovat melko pieniä. Se, tuleeko Digiraha vakiinnuttamaan asemansa sähköisen rahan maksutapana, riippuu mielestäni siitä, omaksuvatko kauppapaikkojen asiakkaat sen käytön. Tällä hetkellä Digiraha ei ole kovin yleinen maksuväline, sillä Suomessa Digiraha-järjestelmää käyttää alle kaksikymmentä sähköistä kauppapaikkaa.

4.4 Mobiili maksaminen

Mobiilissa maksamisessa asiakas suorittaa maksun langattomalla laitteella, kuten matkapuhelimella tai PDA-laitteella, jolloin maksutiedot kulkevat langattoman tietoverkon yli vastaanottajalle. Mobiiliin maksamiseen on luotu lukuisia maksutapoja, jotka soveltuvat erisuuruisten maksujen suorittamiseen. Seuraavaksi läpikäymme yleisimmät tavat mobiiliin maksamiseen, jonka jälkeen käsittelemme mobiiliin kaupankäyntiin liittyviä turvallisuusuhkia.

4.4.1 Tapoja mobiiliin maksamiseen

Mobiiliin maksamiseen on kehitetty useita eri menetelmiä. Aiemmin tässä luvussa esitellyt maksumenetelmät sivuavat näitä menetelmiä, mutta toteutustavoissa täytyy ottaa huomioon mobiililaitteiden erikoispiirteet tiedonkäsittelyssä ja tiedonvälityksessä. Turvallisuusnäkökohtien lisäksi toimivan sovelluksen täytyy myös olla helppokäyttöisiä ja suorituskyvyltään sellaisia, jotta ne saavuttavat asiakkaiden hyväksynnän toimivana maksutapana.

Yleisin laskutustapa mobiilipalveluiden käytössä on maksun lisääminen matkapuhelimen laskuun (Mallat & al., 2004). Maksutavan etuna on se, ettei asiakkaan tarvitse erikseen rekisteröityä palveluun ja tämä tapa on asiakkaalle käytännöllinen myös pieniä maksuja maksaessa. Palveluntarjoajalta tämä maksutapa vaatii sopimuksen tekemistä teleoperaattoreiden kanssa, jotta laskutukset saadaan hoidettua automaattisesti operaattoreiden järjestelmissä. Tämän maksutavan turvallisuuden keskeisenä tekijänä on se, että asiakas ei hukkaa puhelintaan ja sen mukana teleoperaattorin hänelle myöntämää SIM-korttia, johon asiakkaan tunnistus perustuu. Näiden hukkaamisen myötä kuka tahansa ulkopuolinen voi ostaa palveluita puhelinliittymän oikean haltijan maksatettavaksi.

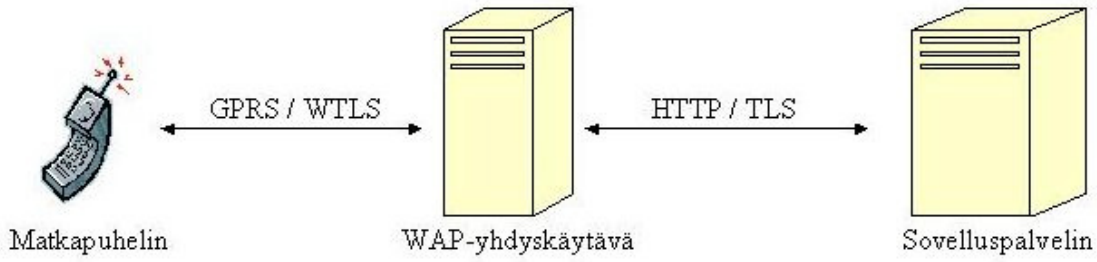
Esimerkki suomalaisesta palvelusta, joka käyttää maksutapana puhelinlaskulla veloittamista, on Helsingin kaupungin liikennelaitoksen matkalipun ostojärjestelmä (Mallat & al., 2004). Järjestelmä toimii siten, että asiakas lähettää matkapuhelimellaan SMS-viestin palvelun käyttämään puhelinnumeroon, mihin palvelu vastaa lähettämällä sähköisen matkalipun asiakkaan matkapuhelimeen. Matkalippu on välittömästi voimassa ja matkalipun veloitus tapahtuu seuraavan puhelinlaskun yhteydessä.

Toinen tapa maksaa mobiilimaksuja on luottokorttitietojen lähettäminen palveluntarjoajalle. Tällöin asiakas syöttää maksutiedot, kuten luottokortin numeron, mobiililaitteeseensa ja lähettää ne sähköiseen kauppapaikkaan langattoman verkon yli. Kaikki mobiililaiteet eivät kuitenkaan pysty muodostamaan HTTP-yhteyttä kauppapaikkaan, vaan yhteys muodostetaan langattoman tiedonvälityksen keinoin (O'Mahony & al., 2001):

- *WAP-protokolla* (Wireless Application Protocol) on langattoman tiedonvälityksen määritys, joka kerää joukon tiedonsiirtoprotokollia yhteen. WAP-muotoinen viestinvälitys ei siten ole tiukasti määritelty, vaan siinä voidaan käyttää erilaisia tiedonvälitystapoja. Tällaisia tapoja ovat esimerkiksi SMS-viestit tai GPRS-protokolla, joka on yleisesti käytetty langattomien laitteiden tiedonvälitysprotokolla.
- *WAP-yhdyskäytävä* on käyttäjän mobiililaitteen ja Internetin välillä toimiva välityspalvelin. WAP-yhdyskäytävä muuntaa mobiililaitteelta vastaanottamansa WAP-muotoiset viestit HTTP-protokollan mukaiseksi ja lähettää ne sovelluspalvelimelle. Kun sovelluspalvelin vastaa HTTP-protokollaa käyttäen, yhdyskäytävä muuntaa viestit WAP-muotoon ja välittää ne mobiililaitteelle.

Yhdyskäytävä voi olla teleoperaattorin hallinnoima, mutta sähköisellä kauppapaikalla voi olla myös oma yhdyskäytävä. Tällöin asiakkaan täytyy erikseen asettaa mobiililaitteensa käyttämään kyseistä yhdyskäytävää, silloin kun asiakas suorittaa ostoksia kauppapaikassa.

WAP-yhteys mobiililaitteen ja yhdyskäytävän välillä voidaan myös salata käyttäen WTLS-protokollaa, joka vastaa TLS-protokollaa langattomassa tiedonvälityksessä (O'Mahony & al., 2001). Tällöin WAP-yhdyskäytävä purkaa WTLS-salauksen ja salaa tiedon uudelleen TLS-protokollalla ennen sovelluspalvelimelle lähettämistä. Ja vastaavasti sovelluspalvelimelta tulevat viestit yhdyskäytävä muuntaa WTLS-muotoon (Kuva 7).



Kuva 7: WAP-yhteys.

Myös sähköisen rahan käyttäminen onnistuu mobiilisti. Esimerkkinä tästä on suomalainen SmartRestaurant-järjestelmä, jossa asiakas voi tilata ja maksaa ruoka-annoksensa mobiililaitetta käyttäen (Lukkari & al., 2004). Hyväksytyt tilaukset välittyvät ravintolaan, joka saa tiedon tilatusta ruoka-annoksesta ja sekä kellonajasta, jolloin asiakas saapuu noutamaan ostoksensa. Maksaminen noudattaa yleistä sähköisen rahan maksutapaa: Käyttäjä siirtää etukäteen rahaa pankkitililtään erilliselle maksutilille, jolta palvelu maksetaan tilauksen yhteydessä. Tilauksen hyväksymisen yhteydessä maksu siirtyy palveluntarjoajan maksutilille, josta ne myöhemmin siirretään palveluntarjoajan pankkitilille.

4.4.2 Mobiiliin maksamiseen liittyviä ongelmia

Mobiilissa maksamisessa on useita rajoituksia, jotka sovelluksen ja sen maksutapahtuman toteutuksessa täytyy ottaa huomioon. Tällaisina rajoittavina tekijöinä Tarasewich (2003) mainitsee langattomien laitteiden näytön, joka on huomattavasti työaseman näyttöä pienempi, langattoman verkon hitauden ja mobiilin laitteen työasemaa heikomman suorituskyvyn. Kaikki nämä rajoitukset vaikuttavat sovelluksen käytettävyyteen sekä sovellusten toteuttamismenetelmiin.

Toinen mobiiliin kaupankäyntiin oleellisesti liittyvä haaste, käytettävyyden lisäksi, on jälleen turvallisuus (Herzberg 2003; Tarasewich 2003). Käyttäjän tunnistamiseen liittyvät riskit ovat mobiilissa kaupankäynnissä suuremmat kuin työasemalta tehtävässä kaupankäyntitapahtumassa. Tämä siksi, että mobiilissa kaupankäynnissä asiakkaan tunnistaminen tehdään usein, ainakin osittain, mobiililaitteen tunnistamisen kautta. Jos palvelu ei vaadi maksutapahtuman yhteydessä käyttäjältä minkäänlaista tunnistustietoa, kuten PIN-koodia, kuka tahansa, joka saa laitteen haltuunsa, voi käyttää mobiilipalvelua.

Käyttäjän tunnistaminen mobiililaitteella voi kuitenkin olla raskas operaatio, joka heikentää palvelun käytettävyyttä. Jonkinlainen kompromissi on jälleen saavutettava käytettävyyden ja turvallisuuden välillä, johon Salvi & Sahai (2002) esittävät portaittaisen toimintamallin. Mallissa asiakkaat rekisteröityvät maksupalveluun ja asettavat käytettävät tunnistusmenetelmät erisuuruisille maksumäärille. Malli koostuu neljästä eri tasosta:

- Tasolla 0 järjestelmä ei vaadi asiakkaan tunnistamista. Tämä taso soveltuu erittäin pienien maksujen suorittamiseen, jolloin luvaton käyttö ei yksittäisenä tapahtumana aiheuta suurta taloudellista vahinkoa.
- Tasolla 1 järjestelmä vaatii asiakkaalta tunnistustietona oikeellisen PIN-koodin syöttämisen, ennen kuin maksu suoritetaan asiakkaan tililtä kauppiaan tilille.
- Tasolla 2 asiakas velvoittaa erillisen välittäjäosapuolen digitaalisesti allekirjoittamaan maksutapahtuman, asiakkaan puolesta. Maksutapahtuman aikana asiakkaan tunnistus perustuu asiakkaan välittäjälle lähettämään PIN-koodiin, jonka oikeellisuustarkistuksen jälkeen välittäjäosapuoli varmistaa maksutapahtuman digitaalisella allekirjoituksella.
- Taso3 on mallin korkein turvallisuustaso ja siinä digitaalinen sertifikaatti on tallennettu mobiililaitteeseen. Maksutapahtumassa asiakkaan on ensin syötettävä oikea PIN-koodi, minkä jälkeen sertifikaattia voidaan käyttää maksutapahtuman allekirjoittamiseen.

Käyttäjän tunnistamisen lisäksi langattoman yhteyden toteutus on turvallisuusriski, joka saattaa vaarantaa maksutapahtuman. Rinne (2002) mainitsee, että GSM-puheluiden salaus on ollut murrettavissa reaaliaikaisesti jo vuosien ajan. Samoin WAP-yhdyskäytävien läpi kulkevien viestien turvallisuus on aiemmin ollut kyseenalainen, sillä kun yhdyskäytävä muuntaa viestejä WAP-muodosta HTTP-protokollan mukaiseksi, ja toisinpäin, niin viesti on hetken aikaa salaamattomassa muodossa palvelimella ja mahdollisen murtautujan luettavissa. Tämä ongelma on kuitenkin korjattu WAP 2.0-määrityksessä, joka mahdollistaa yhteyden ilman välityspalvelinta.

Toisaalta keskimääräinen sähköisen kauppapaikan asiakas tai kauppias ei suuresti pysty vaikuttamaan langattomien yhteyksien turvallisuuteen. Tällöin mielestäni kauppiaan tulee

löytää maksutavaksi sellainen, joka tarjoaa riittävän turvallisuuden, mutta myös asiakkaille soveltuvan käyttötavan.

4.4.3 Mobiilin kaupankäynnin tulevaisuus

Mobiililaitteiden määrä on viime vuosina ollut suuressa kasvussa ja sitä mukaa myös sellaisten Internetissä toimivien kauppapaikkojen, joissa ostotapahtuma voidaan suorittaa mobiililaitteella (O'Mahony & al., 2001). Mobiililaitteiden käytettävyys, pienistä näytöistä ja hitaista yhteyksistä johtuen, on ollut vaatimatonta, mikä ei ole saanut asiakkaita kiinnostumaan mobiilista kaupankäynnistä. Samoin käyttäjiä on epäilyttänyt langattoman tiedonsiirron turvallisuus.

Mobiilin kaupankäynnin tulevaisuuden uskotaan kuitenkin olevan valoisampi. Nykyistä kehittyneempien, langattomien verkkojen, kuten 3G:n käyttöönoton ja mobiililaitteiden kasvavan suorituskyvyn uskotaan lisäävän erilaisten mobiilipalveluiden mahdollisuuksia (O'Mahony & al., 2001; Mallat & al., 2004). Tällöin erilaisten mobiilin kaupankäynnin sovellusten sekä niihin liittyvien maksutapojen määrän uskotaan nousevan.

5. SÄHKÖISEN KAUPPAPAIKAN TOTEUTTAMINEN J2EE-YMPÄRISTÖSSÄ

Aiemmissa luvuissa käsiteltiin tiedon salaamista, käyttäjän tunnistusta ja erilaisia sähköisen kauppapaikan maksutapoja. Kaikki nämä ovat osa turvallisen kauppapaikan luomista. Tässä luvussa perehdytään siihen, miten nämä osat toteutetaan käytännössä. Ensin luodaan TLS-salauksen vaatimat komponentit sähköiseen kauppapaikkaan ja läpikäydään niiden asennus sovelluspalvelimelle. Toisena osana toteutetaan käyttäjän tunnistus ja valtuutus ja lopuksi toteutetaan verkkopankin liittäminen sähköiseen kauppapaikkaan. Toteutukset tehdään Java-ympäristössä käyttäen J2SE 5.0- ja J2EE 1.4-versioita ja sovelluspalvelimena toimii Apache Tomcat 5.5. Kaikki nämä työkalut ovat maksuttomia ja saatavilla useille eri käyttöjärjestelmille.

5.1 Yhteyden muodostus HTTPS-protokollalla

WWW-palvelun yhteyden salaaminen TLS-protokollalla perustuu julkisen avaimen menetelmään. Tällöin kauppiaalla on käytössään yksityisestä ja julkisesta avaimesta muodostuva avainpari, jolla salaustoimenpiteet suoritetaan. Jotta julkisen avaimen välityksen aikana kukaan ulkopuolinen ei voi muuttaa avainta, asetetaan avain osaksi digitaalista sertifikaattia, jonka luotettu taho, varmentaja, digitaalisesti allekirjoittaa. Tämän jälkeen sertifikaatti asennetaan sovelluspalvelimelle, joka lähettää sen asiakkaalle aina HTTPS-yhteyttä muodostettaessa.

TLS-yhteyden asennus koostuu seuraavista osista (Kumar, 2004):

1. Yksityisen ja julkisen avaimen muodostus. Samalla luodaan digitaalinen sertifikaatti WWW-palvelun tiedoilla ja tallennetaan avaimet ja sertifikaatti avainvarastoon.
2. Sertifikaatista luodaan sertifikaattipyyntö CSR (Certificate Signing Request), joka lähetetään varmentajalle. Varmentaja muodostaa sertifikaattipyynnöstä digitaalisen sertifikaatin ja palauttaa sen allekirjoitettuna.
3. Digitaalisesti allekirjoitetun sertifikaatin tallentaminen käytettävän palvelimen avainvarastoon ja sovelluspalvelimen konfigurointi.

Digitaalisen sertifikaatin muodostukseen käytämme *keytool*-työkalua, joka on Javan komentorivipohjainen ohjelma digitaalisten sertifikaattien käsittelyyn (Kumar, 2004). Avainparin ja sertifikaatin muodostuksessa keytool-työkalulle annetaan parametreja, joilla voidaan vaikuttaa muodostuviin salausavaimiin ja avainvarastoon. Pakolliset parametrit ovat avainvaraston ja avainparin nimi. Lisäparametreina käyttäjä voi syöttää salausavainten käyttämän salausalgoritmin ja avainten pituuden sekä avainvaraston tyypin. Jos lisäparametreja ei ole annettu parametrilistassa, ohjelma käyttää oletusarvoja. Myös avainvaraston salasana voidaan antaa komentorivin parametrina, mutta turvallisempi tapa on syöttää se ohjelman suorituksen aikana.

Työkaluohjelman käynnistyessä käyttäjältä kysytään ensiksi avainvaraston salasana, jos sitä ei ole annettu parametrina. Tämän jälkeen käyttäjä antaa ohjelmalle syötteenä sertifikaatille tallentuvia tietoja kuten sertifikaatin omistajan nimi ja osoitetiedot. Tietojen vahvistamisen jälkeen sovellus luo käyttäjän kotihakemistoon avainvarastotiedoston, joka sisältää digitaalisen sertifikaatin. Tämän jälkeen muodostamastamme digitaalisesta sertifikaatista luodaan sertifikaattipyyntö. Sertifikaattipyyntö sisältää sertifikaatin sellaisessa muodossa, että varmentaja voi lukea ja digitaalisesti allekirjoittaa sertifikaatin tiedot. Sertifikaattipyyntö luodaan edelleen keytool-työkalua käyttäen, joka tallentaa sertifikaattipyynnön omaksi tiedostoksi.

Sertifikaattipyyntö lähetetään varmentajalle, joka palauttaa sertifikaattipyynnöstä luodun sertifikaatin, digitaalisesti allekirjoitettuna. Varmentajalta saatu sertifikaatti tallennetaan aiemmin luomaamme avainvarastoon, jossa se korvaa alkuperäisen sertifikaatin.

Tässä vaiheessa olemme luoneet sertifikaatin ja hankkineet siihen varmentajan digitaalisen allekirjoituksen. Sertifikaatti on avainvarastotiedostossa testi.ks, sertifikaattipyyntö tiedostossa testi.csr ja varmentajan palauttama digitaalinen sertifikaatti tiedostossa testi.p7b. Käytetyt keytool-komennot näkyvät kuvassa 8.

```
keytool –genkey –keystore testi.ks –alias testiavain –keyalg RSA

keytool –certreq –keystore testi.ks –alias testiavain –file testi.csr

keytool –import –keystore testi.ks –alias testiavain –file testi.p7b
```

Kuva 8: Digitaalisen sertifikaatin muodostus ja avainvarastoon tallennus.

Kun allekirjoitettu digitaalinen sertifikaatti on tallennettu avainvarastoon, konfiguroidaan sovelluspalvelin käyttämään tätä sertifikaattia. Käyttämämme sovelluspalvelin on Apache Tomcat ja sen TLS-määritys on XML-muotoisessa tiedostossa server.xml (Apache Software Foundation, 2005). Salatun yhteyden määrittelevä osuus on oletusarvoisesti kommentoituna, joten salauksen käyttöönotossa poistetaan ensin kommenttirivit. Samalla lisätään avainvaraston salasana keystorePass-attribuutilla ja avainvarastotiedoston sijainti keystoreFile-attribuutilla. XML-muotoinen määritys, jonka mukaan Tomcat luo TLS-kuuntelijan on esitetty kuvassa 9.

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443 -->
  <Connector port="8443" maxHttpHeaderSize="8192"
         maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
         enableLookups="false" disableUploadTimeout="true"
         acceptCount="100" scheme="https" secure="true"
         clientAuth="false" sslProtocol="TLS"
         keystorePass="salasana" keystoreFile="C:\Keystore\testi.ks"/>
```

Kuva 9: Salatun yhteyden konfigurointi Tomcat-palvelimelle.

Konfiguroinnin jälkeen käynnistetään sovelluspalvelin, joka luo TLS-kuuntelijan määrittelemäämme porttiin. Kaikki sähköiseen kauppapaikkaamme tulevat HTTP-yhteydet, jotka käyttävät kyseistä porttia, salataan TLS-protokollalla.

5.2 Käyttäjän tunnistus

Käyttäjätunnuksen ja salasanaan perustuva tunnistustapa on hyvin suosittu WWW-sovelluksissa ja se soveltuu käytettävyydeltään hyvin myös sähköisen kauppapaikan

tunnistusmenetelmäksi. Tämä tunnistusmenetelmä voidaan toteuttaa HTTP-yhteyden yli kolmella eri tavalla (Kumar, 2004):

- Perustunnistuksessa (Basic Authentication) palvelin lähettää yhteyden ottaneelle asiakassovellukselle, sähköisen kauppapaikan tapauksessa WWW-selaimelle, tunnistuspyynnön. Selaimeen avautuu erillinen ikkuna, johon käyttäjä syöttää käyttäjätunnuksen ja salasanan, jotka selain muuntaa base64-muotoon ja lähettää palvelimelle. Palvelin tarkastaa tunnistustiedot ja niiden täsmätessä, palauttaa selaimen pyytämän HTML-sivun tai muun WWW-resurssin. Base64-menetelmä ei salaa tietoja, vaan ainoastaan muuttaa binääridatan ASCII-muotoon tiedonsiirron ajaksi. Tästä syystä tunnistustiedot ovat helposti ulkopuolisten selvitettävissä, mikäli niitä ei erikseen salata.
- Tiivisteeseen perustuva tunnistus (Digest Authentication) toimii aluksi vastaavalla tavalla kuin perustunnistus, mutta eroaa selaimen lähettämien tunnistustietojen osalta. Tunnistustiedoissa kulkee käyttäjätunnus, mutta ei salasanaa, vaan tiivistearvo, jonka selain laskee käyttäjätunnuksesta, salasanasta ja palvelimen lähettämästä satunnaisdatasta. Palvelin laskee vastaavan tiivisteen järjestelmään tallennetuista tiedoista ja vertaa sitä käyttäjän lähettämiin tietoihin. Tällä tavalla käyttäjän tunnistus voidaan suorittaa ilman että salasanaa kuljetetaan verkon yli, mikä parantaa tunnistusmenetelmän turvallisuutta.
- HTML-lomakeen avulla tehtävässä tunnistuksessa HTML-sivulle on määritelty lomake, joka sisältää joukon parametreja, tässä tapauksessa käyttäjätunnuksen ja salasanan. Lomakkeen parametrit lähetetään HTTP-pyyntönä sovelluspalvelimelle, joka vertaa tietoja järjestelmään tallennettuihin tunnistustietoihin. HTML-lomakkeeseen perustuvassa tunnistusmenetelmässä tiedot kulkevat selväkielisenä, joten tietojen turvallinen välitys vaatii erillisen salauksen, kuten TLS-protokollan, käyttämistä. Menetelmä on kuitenkin hyvin suosittu sillä se mahdollistaa sovelluskohtaisten HTML-sivujen muodostamisen sisäänkirjautumistapahtumaan.

Edellä mainituista toteutamme HTML-lomakkeeseen perustuvan tunnistusmenetelmän, joka käyttää kahta erillistä WWW-sivua. Toteutamme nämä sivut JSP-sivuina (JavaServer Pages). *JSP-sivu* on dynaaminen WWW-sivu, joka muodostuu HTML-koodista ja Javalla kirjoitettavasta ohjelmakoodista. Ensimmäinen tunnistusmenetelmän sivuista on sisäänkirjautumissivu, jossa käyttäjä syöttää tunnistustietonsa ja lähettää ne

sovelluspalvelimelle. Tunnistuksen onnistuessa sovelluspalvelin ohjaa käyttäjän WWW-palvelun sivuille. Jos tunnistus epäonnistuu, käyttäjä ohjataan tunnistusmenetelmän toiselle sivulle, jolla ilmoitetaan käyttäjälle epäonnistumisesta. HTML-sivujen luomisen jälkeen tunnistusmenetelmä otetaan käyttöön konfiguroimalla sovelluspalvelimen asetuksia.

Ensiksi luomme sisäänkirjautumissivun, jonka HTML-lomakkeeseen on määritelty tunnistuksessa tarvittavat tiedot. Palvelun pystyttäjä voi muodostaa haluamansa näköisen sisäänkirjautumissivun, kunhan sivulla on tunnistuksen tarvitsemat osat. Näitä osia ovat J2EE-määrityksen mukaan lomakkeen toiminta-, käyttäjätunnus- ja salasanaparametrit, joita käyttäen sovelluspalvelin pystyy toteuttamaan käyttäjän tunnistuksen. Yksinkertainen sisäänkirjautumissivu login.jsp esitetään kuvassa 10.

```
<html>
<head><title>Sisäänkirjautuminen</title></head>
<body bgcolor="#ffffff">
<h1>OY Firma AB - Sähköinen kauppapaikka</h1>
<h2>Sisäänkirjautuminen</h2>
<form action="j_security_check" method="POST" name="loginForm">
<table cellspacing="3" cellpadding="1">
<tr>
  <td width="100"><b>Käyttäjätunnus:</b> </td>
  <td><input type="text" size="15" name="j_username"></td>
</tr>
<tr>
  <td width="100"><b>Salasana:</b></td>
  <td><input type="password" size="15" name="j_password"></td>
</tr>
<tr>
<td> </td>
<td><input type="submit" name="laheta" value="Sisään"></td>
</tr>
</table>
</form>
</body>
</html>
```

Kuva 10: Sisäänkirjautumissivu.

Jos käyttäjän tunnistus epäonnistuu, selain ohjataan sisäänkirjautumisen virhesivulle. Tämän sivun ulkoasun ja sisällön saa palvelun pystyttäjä vapaasti määritellä. Kuvassa 11 esitetään yksinkertainen virhesivu loginerror.jsp.

```
<html>
<head>
<title>Sisäänkirjautuminen epäonnistui</title>
</head>
<body bgcolor="#ffffff">
<h1>OY Firma AB - Sähköinen kauppapaikka</h1>
<h2>Sisäänkirjautuminen epäonnistui</h2>
<a href="login.jsp">Takaisin sisäänkirjautumissivulle</a>
</body>
</html>
```

Kuva 11: Sisäänkirjautumisen virhesivu.

HTML-sivujen luomisen lisäksi sovelluskehittäjän täytyy määritellä sovelluspalvelin käyttämään HTML-lomakkeeseen perustuvaa käyttäjän tunnistusta. J2EE-määrityksen mukaan tämä tehdään WWW-sovelluksen web.xml-tiedostossa, joka on sovelluksen yleinen konfigurointitiedosto. Tiedostoon määritellään käytetty tunnistusmenetelmä, sovelluksen osat joissa tunnistus vaaditaan, käyttäjiltä vaaditut roolit sekä tunnistuksessa käytetty sisäänkirjautumis- ja virhesivu. Kuvassa 12 esitetään konfigurointitiedoston osa, jossa asetetaan tunnistusmenetelmäksi HTML-lomake, jota käytetään kaikilla JSP-sivuilla käyttäjille, jotka ovat rooleissa asiakasrooli tai yllapitorooli. Käytettävä sisäänkirjautumissivu on login.jsp ja virhesivu on loginerror.jsp.

```
<security-constraint>
  <display-name>RMB2 Security Constraint</display-name>
  <web-resource-collection>
    <web-resource-name>RMB2 protected files</web-resource-name>
    <url-pattern>*.jsp</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>asiakasrooli</role-name>
    <role-name>yllapitorooli</role-name>
  </auth-constraint>
  <user-data-constraint>
    <transport-guarantee>INTEGERAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>

<login-config>
  <auth-method>FORM</auth-method>
  <realm-name>RMB2 Form Based Authentication</realm-name>
  <form-login-config>
    <form-login-page>/login.jsp</form-login-page>
    <form-error-page>/loginerror.jsp</form-error-page>
  </form-login-config>
</login-config>

<security-role>
  <role-name>asiakasrooli</role-name>
</security-role>
<security-role>
  <role-name>yllapitorooli</role-name>
</security-role>
```

Kuva 12: Tunnistusmenetelmän määrittely.

Tunnistusmenetelmä asetettiin toimimaan tietyillä käyttäjien rooleilla, joten myös sovelluksen käyttäjät salasanoineen sekä roolitukset on määriteltävä. Oletusarvoisesti Tomcat-sovelluspalvelimessa käyttäjien tunnistustiedot ja roolit tallennetaan konfigurointitiedostoon tomcat-users.xml. Tiedot tallennetaan XML-muodossa kuvan 13 mukaisesti.

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
 <role rolename="asiakasrooli"/>
 <role rolename="yllapitorooli"/>
 <role rolename="manager"/>
 <role rolename="admin"/>

 <user username="yllapitaja" password="yllapitaja"roles="yllapitorooli,asiakasrooli"/>
 <user username="asiakas" password="asiakas" roles="asiakasrooli"/>
 <user username="admin" password="" roles="admin,manager"/>
</tomcat-users>
```

Kuva 13: Käyttäjätunnukset ja roolit Tomcatin XML-tiedostossa.

Käyttäjätunnusten ja roolien tallentaminen tiedostoon ei ole kuitenkaan hyvä ratkaisu (Kumar, 2004). Ensinnäkin, kuka tahansa joka pääsee näkemään tämän tiedoston, saa haltuunsa kaikkien asiakkaiden selväkieliset salasanat. Tämän lisäksi järjestelmän ylläpitäjän täytyy päivittää tekstitiedostoa aina kun järjestelmälle tulee uusi käyttäjä.

Edellä mainituista syistä johtuen, parempi tapa on tallentaa käyttäjätunnukset ja salasanat tietokantaan. Tällöin tiedot voidaan tallentaa automaattisesti käyttäjän rekisteröityessä palveluun eikä järjestelmän ylläpitäjän tarvitse tehdä tallennusta aina uuden asiakkaan liittyessä palvelun käyttäjäksi. Samalla voidaan tehdä salasanojen selvittäminen hankalammaksi mahdollisten tietomurtojen tekijöille. Tämä toteutetaan siten, ettei tallenneta tietokantaan käyttäjän salasanaa vaan tiiviste salasanasta. Tällöin sisäänkirjautumisen yhteydessä lasketaan käyttäjän antamasta salasanasta tiiviste ja verrataan sitä tietokantaan tallennettuun tiivisteeseen. Tiivisteen laskeminen Javalla on toteutettu kuvassa 14.

```
package kauppapaikka;

import java.security.MessageDigest;
import java.io.ByteArrayOutputStream;
import java.io.PrintStream;

public class Tiivistelaskuri {

  public String laskeTiiviste(String syote) {
    byte[] tiiviste = luoTiiviste(syote.getBytes());
    String tiivisteTekstina = muunnaTiivisteTekstiksi(tiiviste);
    return tiivisteTekstina;
  }

  private byte[] luoTiiviste(byte[] syote) {
    try {
      MessageDigest md = MessageDigest.getInstance("MD5");
       md.update(syote);
       return md.digest();
    } catch (Exception e) {
       e.printStackTrace();
    }
    return null;
   }

   private void hexDigit(PrintStream ps, byte x) {
     char c;
     c = (char) ((x >> 4) & 0xf);
     if (c > 9) {
         c = (char) ((c - 10) + 'a');
      } else {
         c = (char) (c + '0');
      }
      ps.write(c);

      c = (char) (x & 0xf);
      if (c > 9) {
         c = (char) ((c - 10) + 'a');
      } else {
         c = (char) (c + '0');
      }
      ps.write(c);
   }

   private String muunnaTiivisteTekstiksi(byte[] tiiviste) {
      // Muunnetaan tiiviste heksamuotoon ja palautetaan se tekstimuodossa
      ByteArrayOutputStream bas = new ByteArrayOutputStream();
      PrintStream ps = new PrintStream(bas);
      for(int i = 0; i < tiiviste.length; i++) {
         hexDigit(ps, tiiviste[i]);
      }
      String tiivisteTekstina = bas.toString();
      return tiivisteTekstina;
   }

}
```

Kuva 14: Tiivisteen laskeminen.

Käyttäjän onnistuneen tunnistamisen jälkeen asiakkaan selain ohjautuu kauppapaikan sivuille. Kauppapaikan sivuilla asiakas läpikäy tuotteita ja lisää niitä loogiseen ostoskoriinsa. Ostoskoria ja muita käyttäjän suorittamia toimintoja varten sovelluspalvelin luo jokaista käyttäjää kohden oman istuntotiedon, joka tallennetaan sovelluspalvelimen muistiin.

Jotta palvelu pystyy yhdistämään käyttäjän oikeaan istuntotietoon tilatonta tiedonsiirtoprotokollaa käytettäessä, sovelluspalvelin muodostaa jokaiselle istuntotiedolle yksilöivän tunnisteen. Tämän jälkeen tunnistetieto toimitetaan asiakkaalle käytettäväksi siten, että jokaisessa asiakkaan lähettämässä HTTPS-kutsussa kulkee mukana tämä tunnistetieto (Hartman & al., 2003). Yleinen tapa välittää tämä tunnistetieto on käyttää evästettä. *Eväste* on asiakkaan koneelle tai selaimen muistiin tallentuva pieni tekstitiedosto, joka sisältää tunnistetiedon ja mahdollisesti tietoja myös käyttäjän muista toiminnoista. Evästeen sisältö

riippuu siitä, mitä tietoja palvelun pystyttäjä haluaa asiakkaan koneelle tallentaa. Tästä syystä evästeiden käyttö ei ole aina täysin turvallista, joten käyttäjä voi estää evästeiden käytön selaimensa asetuksia muutamalla tai sallia vain istuntokohtaisten evästeiden käytön, jolloin evästeet poistuvat asiakkaan koneelta kun asiakas sulkee selaimen.

Mikäli evästeitä ei haluta käyttää palvelussa, istunnon tunnistetieto voidaan välittää myös kutsuttavan WWW-osoitteen eli URLin uudelleenkirjoituksella tai HTML-lomakkeen piilotietoina. Molemmissa tapauksissa kaikkiin kauppapaikan sivujen toimintoihin liitetään tunnistetieto ennen sivun lähettämistä asiakkaan selaimelle. Kun asiakas valitsee sivulta toiminnon, tunnistetieto välittyy takaisin sovelluspalvelimelle, joka yhdistää tunnisteen istuntotietoon. Tunnistetiedon lisääminen sivulle vaatii siten enemmän toteutustyötä kuin evästeen käyttö. Tämän luvun esimerkkikauppapaikassa käytetään istuntokohtaista evästettä, jonka avulla välitetään ainoastaan istunnon tunnistetieto.

5.3 Verkkopankin liittäminen sähköiseen kauppapaikkaan

Maksutavoista toteutettavaksi valitsin verkkopankin sen turvallisuuden ja käytännöllisyyden takia. Koko maksutapahtuma suoritetaan suojatun yhteyden yli ja maksutiedoista lasketaan kauppiaan ja pankin järjestelmien toimesta tiivisteet, joilla voidaan varmistaa että maksun tiedot eivät muutu asiakkaan tai jonkun ulkopuolisen toimesta. Käytännöllisyyttä lisää se, ettei asiakas tarvitse maksuun mitään lisälaitteita ja maksut siirtyvät välittömästi asiakkaan tililtä kauppiaan tilille. Kotimaisilla asiakkailla on useimmiten myös tili jossain suomalaisessa pankissa, joten asiakkaan ei tarvitse tehdä erillistä sopimusta tämän maksutavan käyttämisestä. Maksutapahtumassa asiakas käyttää pankin hänelle myöntämiä verkkopankkitunnuksia.

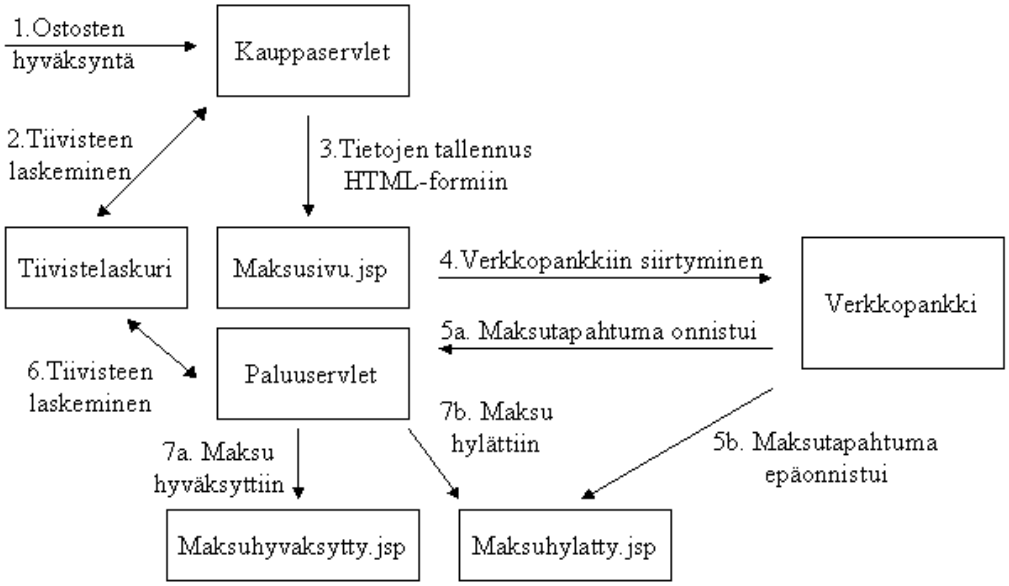
Kauppapaikan liittämisen verkkopankkiin toteutin J2EE-määrityksen mukaisia JSP-sivuja ja HTTP-servlettejä käyttäen. JSP-sivut ovat tutkielman liitteenä 1 ja muu Java-toteutus liitteessä 2. Toteutuksessa ei oteta kantaa verkkokaupan muuhun toiminnallisuuteen, vaan keskitytään pelkästään verkkopankin käyttämiseen maksutapana.

Sähköisen kauppapaikan maksutapahtuma alkaa siitä kun asiakas hyväksyy ostokset ja siirtyy maksamaan ostoksia. Tällöin kauppapaikkaan toteuttamani KauppapaikkaServlet muodostaa

asiakkaan hyväksymistä ostotiedoista ja pankin kauppiaalle myöntämästä kauppiasnumerosta ja salaisesta tunnusluvusta MD5-tiivisteen Tiivistelaskuria käyttäen. Tiivisteen laskemisen jälkeen maksuun liittyvät tiedot tulostetaan HTML-lomakkeen piilotietoina Maksusivu.jsp-sivulle, jolle asiakas siirtyy hyväksyessään ostoksensa. Sivulta tiedot välittyvät verkkopankkiin kun käyttäjä painaa sivulla näkyvää verkkopankkitunnusta.

Kun maksun tiedot välittyvät pankkiin, pankki lähettää asiakkaalle tunnistuspyynnön. Asiakas sisäänkirjautuu verkkopankkiin pankin myöntämillä tunnuksilla ja maksaa ostoksensa. Pankin järjestelmä tarkistaa maksutiedot tiivistettä käyttäen ja suorittaa tilisiirrot, jos tiedot täsmäävät. Jos maksutapahtuma onnistuu, suoritus siirtyy verkkopankista poistuttaessa kauppiaan Maksusivu.jsp-sivun HTML-lomakkeessa määrittelemään paluuosoitteeseen. Tässä esimerkkitapauksessamme suoritus ohjautuu PaluuServletille. Jos maksutiedot eivät vastaa tiivistettä tai maksutapahtumassa tapahtuu virhe, suoritus siirtyy vastaavasti kauppiaan määrittelemään virheosoitteeseen. Virheosoitteena käytämme Maksuhylatty.jsp-sivua, joka ilmoittaa asiakkaalle maksutapahtuman epäonnistumisesta.

Maksutapahtuman onnistuessa PaluuServlet muodostaa Tiivistelaskuria käyttäen uuden tiivisteen. Tiiviste lasketaan pankin lähettämän paluuviestin maksutiedoista sekä kauppiaan salaisesta tunnusluvusta, jonka jälkeen PaluuServlet vertaa että tiiviste on sama kuin pankin laskema tiiviste. PaluuServlet varmistaa myös, että asiakkaan maksusuoritus vastaa ostosten alkuperäistä hintaa. Näillä menetelmillä varmistetaan, että asiakas tai kukaan muu ulkopuolinen ei ole muuttanut maksutietoja. Jos maksutiedot ja tiivisteet täsmäävät, maksutapahtuma on onnistunut ja pankki on siirtänyt maksun asiakkaan tililtä kauppiaan tilille. Tällöin kauppapaikka ilmoittaa asiakkaalle maksun onnistumisesta Maksuhyvaksytty.jsp-sivua käyttäen ja toimittaa tilatut ostokset asiakkaalle. Jos maksutiedot eivät täsmää, asiakas ohjataan virhesivulle Maksuhylatty.jsp. Maksutapahtuman eteneminen komponenttien välillä esitetään kuvassa 15.



Kuva 15: Verkkomaksun eteneminen.

6. YHTEENVETO

Tässä tutkielmassa perehdyttiin sähköisen kauppapaikan turvallisuuteen. Sovelluskehittäjän näkökulmasta tärkeimpiä turvallisuuteen liittyviä tekijöitä ovat tiedon salaaminen asiakkaan ja kauppapaikan välillä, käyttäjän luotettava tunnistaminen sekä maksutapahtuman toteutus.

Internetissä toimivan sähköisen kauppapaikan ja asiakkaan käyttämän WWW-selaimen välillä kulkeva tieto salataan useimmiten TLS-protokollalla. TLS-protokolla on turvalliseksi osoittautunut salausmenetelmä, joka salaa tiedon julkisen avaimen menetelmää käyttäen. Julkisen avaimen menetelmä tarjoaa työkalut luotettavaan tiedonsalaukseen, kun käytetään turvallisiksi todettuja salausalgoritmeja ja riittävän vahvoja salausavaimia.

Sähköisen kauppapaikan tunnistusmenetelmistä ylivoimaisesti käytetyin on tunnistus käyttäjätunnuksella ja salasanalla. Tämä tunnistusmenetelmä on kuitenkin hyvin haavoittuva ja sitä vastaan tunnetaan useita tuloksellisia hyökkäysmuotoja. Näitä hyökkäyksiä vastaan on kehitetty vastatoimia, jotka kuitenkaan eivät usein estä hyökkäyksen onnistumista, vaan ainoastaan hidastavat sitä. Hyökkäysten torjuminen vaikuttaa myös negatiivisesti järjestelmän suorituskykyyn ja käytettävyyteen.

Käyttäjätunnusta ja salasanaa turvallisempi tunnistusmenetelmä on älykortin käyttö, jolloin asiakas tunnistautuu älykortille tallennettua digitaalista sertifikaattia ja PIN-koodia käyttäen. Tunnistusmenetelmä hyödyntää julkisen avaimen menetelmää ja siten sen murtaminen kohtuullisessa ajassa on lähes mahdotonta. Älykortilla tapahtuva tunnistaminen vaatii kuitenkin asiakkaalta kortin ja lukulaitteen hankintaa, mikä nostaa käyttökustannuksia. Asiakas ei voi myöskään käyttää palvelua tietokoneelta, jossa ei ole älykortin lukulaitetta, mikä rajoittaa palvelun käyttömahdollisuuksia. Älykortin käyttö on aiemmin ollut vähäistä, mutta on nyt kuitenkin hitaasti lisääntymässä.

Sähköiseen kauppapaikkaan on mahdollista liittää useita erilaisia maksumenetelmiä, joiden ominaisuudet vaihtelevat turvallisuuden, kustannustason ja käytettävyyden suhteen. Kansainväliselle kauppapaikalle luottokortin ja sähköisen rahan käyttö soveltuvat mielestäni hyvin, kun taas mobiililaitteiden käyttö soveltuu paremmin pienten maksujen maksamiseen ja

maakohtaisiin kauppapaikkoihin, jolloin välitettävän tiedon ei tarvitse kulkea useiden eri teleoperaattoreiden verkoissa. Verkkopankin käyttö puolestaan mahdollistaa korkean turvallisuustason ja tarjoaa asiakkaille helppokäyttöisen maksutavan.

Mielestäni sähköisen kauppapaikan turvallisuus koostuu edellä mainituiden osien muodostamasta kokonaisuudesta, jossa heikoin osa määrittelee pitkälti koko palvelun turvallisuuden. Salatun yhteyden käyttö ei luo turvallista kauppapaikkaa, jos samaan aikaan asiakkaan maksutiedot ovat helposti ulkopuolisten saatavilla. Ennen kauppapaikan pystyttämistä palvelulle tulisi suunnitella yhtenäiset turvallisuusvaatimukset, joilla kauppapaikalle saadaan haluttu turvallisuustaso.

Kaikkien turvallisuutta parantavien menetelmien mukana tulee myös muita seikkoja, jotka vaikuttavat palvelun pystyttämiseen. Vahvat tiedonsalausmenetelmät ja asiakkaiden tunnistaminen älykortilla nostavat kauppapaikan turvallisuutta, mutta samalla aiheuttavat kustannuksia sekä kauppiaalle että asiakkaille. Kustannusten lisäksi myös palvelun käytettävyys voi laskea kasvaneiden viiveiden ja vaivalloisten tunnistusmenetelmien myötä. Tämän vuoksi palvelun pystyttäjä joutuu tekemään valintoja turvallisuuden, kustannusten ja käytettävyyden välillä. Turvallisuus on tärkeä osa kauppapaikkaa, mutta se ei saa luoda palvelun käytölle niin suurta kynnystä, että asiakkaat jättävät kauppapaikan.

VIITELUETTELO

Apache Software Foundation (2005): *The Apache Tomcat 5.5 Servlet/JSP Container*, WWW-sivusto, http://tomcat.apache.org/tomcat-5.5-doc/index.html (30.10.2005).

Basu, A., Muylle, S. (2003): Authentication in E-Commerce. *Communications of the ACM* **46**(12), 159-166.

Claessens, J., Preneel, B., Vandewalle, J. (2003): (How) can mobile agents do secure electronic transactions on untrusted hosts? A survey of the security issues and the current solutions. *ACM Transactions on Internet Technology* **3**(1), 28-48.

Corcoran, D., Sims, D., Hillhouse, B. (1999): Smart Cards and Biometrics: The cool way to make secure transactions. *Linux Journal* **6**(3), artikkeli nro 7.

Coulouris, G., Dollimore, J., Kindberg, T. (2001): *Distributed Systems, Concepts and Design*. Addison-Wesley, USA.

Dierks, T., Allen C. (1999): *The TLS Protocol Version 1.0*. IETF RFC 2246.

Fielding, R., Gettys, J., Mogul, J., Frystyk H., Masinter, L., Leach, P., Berners-Lee, T. (1999): *Hypertext Transfer Protocol – HTTP/1.1*. IETF RFC 2616.

Halderman, J.A., Waters, B., Felten, E.W. (2005): A convenient method for securely managing passwords. *Proceedings of the 14th intenational conference on World Wide Web*, ACM Press, New York, NY, USA, 471-479.

Halevi, S., Krawczyk, H. (1999): Public-key cryptography and password protocols. *ACM transactions on information and system security* **2**(3), 230-268.

Hartman, B., Flinn, D.J., Beznosov, K., Kawamoto, S. (2003): *Mastering Web Services Security*. Wiley Publishing, USA.

Herzberg, A. (2003): Payments and Banking with Mobile Personal Devices. *Communications of the ACM* **46**(5), 53-58.

Housley, R., Polk, W., Ford, W., Solo, D. (2002): *Internet X.509 Public Key Infrastructure, Certificate and Certificate Revocation List (CRL) profile*. IETF RFC 3280.

Ives, B., Walsh, K.R., Schneider, H. (2004): The domino effect of password reuse. *Communications of ACM* **47**(4), 75-78.

Joshi, J.B.D., Aref, W.G., Ghafoor, A., Spafford, E.H. (2001): Security models for web-based applications. *Communications of the ACM* **44**(2), 38-44.

Khu-smith, V., Mitchell, C.J. (2002): Using GSM to enhance e-commerce security. *Proceedings of the 2nd international workshop on mobile commerce*, ACM Press, New York, NY, USA, 75-81.

Kumar, P. (2004): *J2EE Security For Servlets, EJBs and Web Services*, HP Books, USA.

Lukkari, J., Korhonen, J., Ojala, T. (2004): SmartRestaurant - Mobile Payments in Context-Aware Enviroment. *Proceedings of the 6th internation conference on Electronic commerce*, ACM Press, New York, NY, USA, 575-582.

Mallat, N., Rossi, M., Tuunainen, V.K. (2004): Mobile Banking Services. *Communications of the ACM* **47**(5), 42-46.

MasterCard (2005): *Introducing MasterCard SecureCode*. http://www.mastercardmerchant.com/securecode/index.html (29.9.2005).

Mel, H.X., Baker, D. (2001): *Cryptography Decrypted*, Addison-Wesley, New Jersey.

Park, J.S., Sandhu, R., Ahn, G-J. (2001): Role-based access control on the web. *ACM Transactions on Information and System Security* **4**(1), 37-71.

O'Mahony, D., Peirce, M., Tewari, H. (2001): *Electronic payment systems for e-commerce*, 2.ed. Artech House, USA.

Osuuspankki (2005a): *Digiraha*, WWW-sivusto, http://www.digiraha.net/ (23.10.2005).

Osuuspankki (2005b): *Digirahan kauppiasdokumentit*, WWW-sivusto, https://www.op.fi/ui/bottom.asp?path=14427;14735;77351;77415&data=/templates/julkaisu.asp (24.10.2005).

Pinkas, B., Sander, T. (2002): Securing Passwords Against Dictionary Attacks. *Proceedings of the 9th ACM conference on Computer and communications security* (toim. Atluri, V.), ACM Press, New York, NY, USA, 161-170.

Rhee, M.Y. (2003): *Internet Security, Cryptographic Principles, Algorithms and Protocols*, John Wiley & Sons Ltd, Chippenham, Wiltshire.

Rescorla, E. (2001): *SSL and TLS: Designing and Building Secure Systems*, Addison-Wesley, USA.

Rinne, T. (2002): *Älykortit – tekniikka, sovellusalueet ja käyttöönotto*. Talentum Media Oy, Jyväskylä.

RSA Security (2004): RSA Laboratories, http://www.rsasecurity.com/rsalabs, (23.4.2005).

Salvi, A.B., Sahai, S. (2002): Dial M for Money. *Proceedings of the 2nd international workshop on Mobile commerce*, ACM Press, New York, NY, USA, 95-99.

Sampo Pankki (2005): *Sammon verkkomaksupalvelu, palvelun kuvaus ja palveluntarjoajan ohjekirja*, http://domino.sampo.fi/external/sbd/tuotteet.nsf/liitteet/Verkkomaksu_kauppiasohjeet.pdf/$file/Verkkomaksu_kauppiasohjeet.pdf (2.10.2005).

Shelfer, K.M., Procaccino, J.D. (2002): Smart card evolution. *Communications of the ACM* **45**(7), 83-88.

Summers, W.C., Bosworth, E. (2004): Password policy: the good, the bad, and the ugly. *Proceedings of the winter international synposium on Information and communication technologies*, Trinity College Dublin, 1-6.

Tarasewich, P. (2003): Designing Mobile Commerce Applications. *Communications of the ACM* **46**(12), 57-60.

Tulloch, M. (2003): *Microsoft Encyclopedia of Security*, Microsoft Press, USA.

Verisign, Inc (2000): *Building an E-Commerce Trust Infrastructure, SSL Server Sertificates and Online Payment Services*, http://www.verisign.com/static/003191.pdf (7.5.2005).

Viestintävirasto (2004): *Tietoturvallisuuden perusteet*, http://www.ficora.fi/suomi/tietoturva/ttperusteet.htm, (23.4.2005).

Visa (2005a): *Visa Europe*. http://www.visaeurope.com/iusevisa/seehowitworks.html (29.9.2005).

Visa (2005b): *Visa Authenticated Payment Program, 3-D Secure*. http://partnernetwork.visa.com/pf/3dsec/main.jsp (15.11.2005).

Visa (2005c): *Visa – Verified by Visa*. http://www.visa.fi/Visa sähköisesti/Verified by Visa.html (30.9.2005).

Väestörekisterikeskus (2005): *Väestörekisterikeskus*. WWW-sivusto, http://www.sahkoinenhenkilokortti.fi/ (11.9.2005).

Yan, J.J. (2001): A Note on Proactive Password Checking. *Proceedings of the 2001 workshop on New security paradigms*, ACM Press, New York, NY, USA, 127-135.

LIITE 1: Verkkopankkimaksun JSP-sivut

Maksusivu.jsp

```
<%@page import="kauppapaikka.MaksutiedotBean" %>
<% MaksutiedotBean maksutiedot = (MaksutiedotBean)session.getAttribute("maksutiedot");
%>
<html>
<head>
<title>Sähköinen kauppapaikka</title>
</head>
<body bgcolor="#ffffff">
<h1>OY Firma AB - Sähköinen kauppapaikka</h1>
<p>Maksu verkkopankkia käyttäen</p>
<form method="post" action="https://www514.sampo.fi/A15/vemaha/VemahaApp">
<input name="KNRO" type="hidden" value="<%= maksutiedot.getKnro() %>">
<input name="SUMMA" type="hidden" value="<%= maksutiedot.getSumma() %>">
<input name="VIITE" type="hidden" value="<%= maksutiedot.getViite() %>">
<input name="VALUUTTA" type="hidden" value="<%= maksutiedot.getValuutta() %>">
<input name="VERSIO" type="hidden" value="<%= maksutiedot.getVersio() %>">
<input name="OKURL" type="hidden" value="<%= maksutiedot.getOkUrl() %>">
<input name="VIRHEURL" type="hidden" value="<%= maksutiedot.getVirheUrl() %>">
<input name="TARKISTE" type="hidden" value="<%= maksutiedot.getTarkiste() %>">
<input type="IMAGE"
src="https://www.sampo.fi/verkkopalvelu/verkkomaksu/logoverkkomaksu.gif" BORDER=0>
</form>
<a href="kauppapaikka.jsp">Paluu kauppapaikkaan</a>
</body>
</html>
```

Maksuhyvaksytty.jsp

```
<html>
<head>
<title>Sähköinen kauppapaikka</title>
</head>
<body bgcolor="#ffffff">
<h1>OY Firma AB - Sähköinen kauppapaikka</h1>
<P>Maksutapahtuma hyväksytty. </P>
<a href="kauppapaikka.jsp">Paluu kauppapaikkaan</a>
</body>
</html>
```

Maksuhylatty.jsp

```
<html>
<head>
<title>Sähköinen kauppapaikka</title>
</head>
<body bgcolor="#ffffff">
<h1>OY Firma AB - Sähköinen kauppapaikka</h1>
<P>Maksutapahtumassa tapahtui virhe! </P>
<a href="kauppapaikka.jsp">Paluu kauppapaikkaan</a>
</body>
</html>
```

LIITE 2: Verkkopankkimaksun Java-luokat

KauppapaikkaServlet.java

```
package kauppapaikka;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;

public class KauppapaikkaServlet extends HttpServlet {
  private static final String TESTI_KAUPPIASNUMERO = "000000000000";
  public static final String TESTI_SALAINEN_AVAIN = "testi";
  private static final String TESTI_VALUUTTA = "EUR";
  private static final String TESTI_VIITENUMERO = "9861156";
  private static final String TESTI_VERSIO = "2";
  private static final String OK_URL = "https://213.243.189.95:8443/paluuservlet";
  private static final String VIRHE_URL = "https://213.243.189.95:8443/maksuhylatty.jsp";
  private static final String CONTENT_TYPE = "text/html";
  //Process the HTTP Get request
  public void doGet(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
    try {
      if(request.getParameter("siirryMaksamaan") != null) {
        muodostaMaksutiedot(request);
        String seuraavaSivu = "maksusivu.jsp";
        request.getRequestDispatcher(seuraavaSivu).forward(request,
          response);
      } else {
       // Muut kauppapaikan toiminnot toteutetaan tähän
      }
    } catch (Exception e) {
```

```
      e.printStackTrace();
    }
  }
  //Process the HTTP Post request
  public void doPost(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
    doGet(request, response);
  }
  private void muodostaMaksutiedot(HttpServletRequest request)
          throws ServletException, IOException {
    MaksutiedotBean tiedot = new MaksutiedotBean();
    tiedot.setKnro(TESTI_KAUPPIASNUMERO);
    tiedot.setSumma(laskeSumma(request));
    tiedot.setValuutta(TESTI_VALUUTTA);
    tiedot.setViite(TESTI_VIITENUMERO);
    tiedot.setVersio(TESTI_VERSIO);
    tiedot.setOkUrl(OK_URL);
    tiedot.setVirheUrl(VIRHE_URL);
    laskeTiivistearvoMaksutiedoille(tiedot);
    HttpSession session = request.getSession(true);
    session.setAttribute("maksutiedot", tiedot);
  }
  private void laskeTiivistearvoMaksutiedoille(MaksutiedotBean tiedot) {
    Tiivistelaskuri laskuri = new Tiivistelaskuri();
    String tiivistearvo = laskuri.laskeTiivisteArvo(tiedot, TESTI_SALAINEN_AVAIN);
    tiedot.setTarkiste(tiivistearvo);
  }
  private String laskeSumma(HttpServletRequest request)
          throws ServletException, IOException {
    int yhteissumma = 0;
    // Maksettavan summan laskeminen tulee tähän
    return String.valueOf(yhteissumma);
  }
}
```

MaksutiedotBean.java

```
package kauppapaikka;
public class MaksutiedotBean implements java.io.Serializable {
  private String knro = null;
  private String summa = null;
  private String viite = null;
  private String valuutta = null;
  private String versio = null;
  private String okUrl = null;
  private String virheUrl = null;
  private String tarkiste = null;
  private String status = null;
  private String maksutapa = null;
  public void setKnro(String value) { this.knro = value; }
  public void setSumma(String value) { this.summa = value; }
  public void setViite(String value) { this.viite = value; }
  public void setValuutta(String value) { this.valuutta = value; }
  public void setOkUrl(String value) { this.okUrl = value; }
  public void setVirheUrl(String value) { this.virheUrl = value; }
  public void setTarkiste(String value) { this.tarkiste = value; }
  public void setVersio(String value) { this.versio = value; }
  public void setStatus(String value) { this.status = value; }
  public void setMaksutapa(String value) { this.maksutapa = value; }
  public String getKnro() { return this.knro; }
  public String getSumma() { return this.summa; }
  public String getViite() { return this.viite; }
  public String getValuutta() { return this.valuutta; }
  public String getOkUrl() { return this.okUrl; }
  public String getVirheUrl() { return this.virheUrl; }
  public String getTarkiste() { return this.tarkiste; }
  public String getVersio() { return this.versio; }
  public String getStatus() { return this.status;}
  public String getMaksutapa() { return this.maksutapa; } }
```

Tiivistelaskuri.java

```
package kauppapaikka;
import java.security.MessageDigest;
import java.io.ByteArrayOutputStream;
import java.io.PrintStream;

public class Tiivistelaskuri {
  public String laskeTiivisteArvo(MaksutiedotBean maksutiedot, String salainenAvain) {
    String syoteTeksti = salainenAvain + maksutiedot.getSumma() + maksutiedot.getViite()
        + maksutiedot.getKnro() + maksutiedot.getVersio() + maksutiedot.getValuutta()
        + maksutiedot.getOkUrl() + maksutiedot.getVirheUrl();
    // Lasketaan syötetekstistä tiivistearvo
    byte[] tiiviste = luoTiiviste(syoteTeksti.getBytes());
    // Muunnetaan tiiviste heksamuotoon ja palautetaan se tekstimuodossa
    ByteArrayOutputStream bas = new ByteArrayOutputStream();
    PrintStream ps = new PrintStream(bas);
     for (int i=0; i < tiiviste.length; i++) {
       hexDigit(ps, tiiviste[i]);
     }
     String tiivisteTekstina = bas.toString().toLowerCase();
     return tiivisteTekstina;
  }

  public String laskeTiivisteArvoPaluuviestille(MaksutiedotBean maksutiedot,
          String salainenAvain) {
    String syoteTeksti = salainenAvain + maksutiedot.getViite() + maksutiedot.getSumma()
        + maksutiedot.getStatus() + maksutiedot.getKnro() + maksutiedot.getVersio()
        + maksutiedot.getValuutta();
     // Lasketaan syötetekstistä tiivistearvo
     byte[] tiiviste = luoTiiviste(syoteTeksti.getBytes());
     // Muunnetaan tiiviste heksamuotoon ja palautetaan se tekstimuodossa
     ByteArrayOutputStream bas = new ByteArrayOutputStream();
     PrintStream ps = new PrintStream(bas);
```

```
    for (int i=0; i < tiiviste.length; i++) {
      hexDigit(ps, tiiviste[i]);
    }
    String tiivisteTekstina = bas.toString().toUpperCase();
    return tiivisteTekstina;
  }

public String laskeTiiviste(String syote) {
  byte[] tiiviste = luoTiiviste(syote.getBytes());
  String tiivisteTekstina = muunnaTiivisteTekstiksi(tiiviste);
  return tiivisteTekstina;
}

private byte[] luoTiiviste(byte[] syote) {
  try {
    MessageDigest md = MessageDigest.getInstance("MD5");
     md.update(syote);
     return md.digest();
  } catch (Exception e) { e.printStackTrace(); }
  return null;
 }

 private void hexDigit(PrintStream ps, byte x) {
   char c;
   c = (char) ((x >> 4) & 0xf);
   if (c > 9) {
      c = (char) ((c - 10) + 'a');
    } else {
       c = (char) (c + '0');
    }
    ps.write(c);

    c = (char) (x & 0xf);
    if (c > 9) {
```

```
            c = (char) ((c - 10) + 'a');
        } else {
            c = (char) (c + '0');
        }
        ps.write(c);
    }

    private String muunnaTiivisteTekstiksi(byte[] tiiviste) {
        // Muunnetaan tiiviste heksamuotoon ja palautetaan se tekstimuodossa
        ByteArrayOutputStream bas = new ByteArrayOutputStream();
        PrintStream ps = new PrintStream(bas);
        for(int i = 0; i < tiiviste.length; i++) {
            hexDigit(ps, tiiviste[i]);
        }
        String tiivisteTekstina = bas.toString();
        return tiivisteTekstina;
    }
}
```

PaluuServlet.java

```
package kauppapaikka;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;

public class PaluuServlet extends HttpServlet {
  private static final String CONTENT_TYPE = "text/html";
  public void doGet(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
    boolean maksuOk = false;
    try {
      maksuOk = tarkistaMaksunPaluutiedot(request);
    } catch (Exception e) { e.printStackTrace(); }
    String seuraavaSivu = "maksuvirheellinen.jsp";
    if (maksuOk) {
      lahetaOstoksetAsiakkaalle(request);
      seuraavaSivu = "maksuhyvaksytty.jsp";
    }
    request.getRequestDispatcher(seuraavaSivu).forward(request, response);
  }

  //Process the HTTP Post request
  public void doPost(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
    doGet(request, response);
  }

  private boolean tarkistaMaksunPaluutiedot(HttpServletRequest request)
          throws ServletException, IOException {
    MaksutiedotBean paluutiedot = muodostaPaluutiedot(request);
    Tiivistelaskuri laskuri = new Tiivistelaskuri();
```

```
    String paluuviestistaLaskettuTiiviste =
  laskuri.laskeTiivisteArvoPaluuviestille(paluutiedot,
KauppapaikkaServlet.TESTI_SALAINEN_AVAIN);
    if (paluutiedot.getTarkiste() != null &&
    paluutiedot.getTarkiste().equals(paluuviestistaLaskettuTiiviste)) {
    HttpSession session = request.getSession();
    MaksutiedotBean tiedot = (MaksutiedotBean) session.getAttribute("maksutiedot");
    if (tiedot != null && tiedot.getSumma().equals(paluutiedot.getSumma())) {
      return true;
    }
    return false;
  }

  private MaksutiedotBean muodostaPaluutiedot(HttpServletRequest request)
          throws ServletException, IOException {
    MaksutiedotBean paluutiedot = new MaksutiedotBean();
    paluutiedot.setKnro(request.getParameter("KNRO"));
    paluutiedot.setValuutta(request.getParameter("VALUUTTA"));
    paluutiedot.setViite(request.getParameter("VIITE"));
    paluutiedot.setSumma(request.getParameter("SUMMA"));
    paluutiedot.setVersio(request.getParameter("VERSIO"));
    paluutiedot.setStatus(request.getParameter("STATUS"));
    paluutiedot.setTarkiste(request.getParameter("TARKISTE"));
    paluutiedot.setMaksutapa(request.getParameter("MTAPA"));
    return paluutiedot;
  }

  private void lahetaOstoksetAsiakkaalle(HttpServletRequest request)
          throws ServletException, IOException {
    // yhteys toimitusjärjestelmään, joka lähettää asiakkaalle ostokset
  }
}
```