Tuomas Into

Sosiaalisten verkkosovellusten tietoturva

Tietotekniikan
pro gradu -tutkielma
4. huhtikuuta 2011

Jyväskylän yliopisto

Tietotekniikan laitos

Jyväskylä

Tekijä: Tuomas Into
Yhteystiedot: tuomas.e.into@jyu.fi
Työn nimi: Sosiaalisten verkkosovellusten tietoturva
Title in English: Information Security of Social Software
Työ: Tietotekniikan pro gradu -tutkielma
Sivumäärä: 122
Tiivistelmä: Ihmisten sosiaalinen elämä on siirtynyt kasvavissa määrin sosiaaliseen mediaan. Sosiaalinen media on toimintaympäristö, jossa käyttäjällä on tärkeä rooli tietoturvan toteutuksessa. Tämän ainutlaatuisuuden takia sosiaaliselle medialla on erityisiä vaatimuksia tietoturvallisuudelle. Näitä peilaamalla liiketoiminnallisiin tavoitteisiin, saadaan sosiaalisen median tietoturvariskit. Sosiaalisille medioille tavallisia hyökkäyksiä käydään läpi sekä sitä, kuinka suojautua niiltä. Opinnäytteessä esiteltyjä metodeja ja konventioita käyttämällä suoritetaan hyökkäysskenaariot.
English abstract: The social life of people has somewhat migrated into social media. The social media is an environment, where the user has more impact on the information security. This unique nature poses special requirements on the information security of social media. By mirroring these requirements to the commercial requirements, it is possible to evaluate the information security risks. The most relevant attacks to social media are analyzed and methods of protection introduced. By using the methods and conventions discussed in the thesis, several attack scenarios are planned and executed.
Avainsanat: Pro gradu-tutkielma, sosiaalinen media, sosiaaliset verkkosovellukset, tietotekniikka, tietoturva.
Keywords: Information security, information technology, masters thesis, social media, social software.

1 Johdanto

Sosiaaliset verkkosovellukset ovat informaatiovallankumouksen siivittämä ilmiö, joka on tullut pysyäkseen. Sosiaaliset verkkosovellukset ovat muuttaneet tapojamme ja odotuksiamme siitä, kuinka kommunikoimme toistemme kanssa. Monessa mielessä sosiaaliset verkkosovellukset edustavat nykypäivänä ihmisille ajanviettopaikkoja, jossa voidaan vaihtaa kuulumiset, kerrata päivän kuumimmat uutiset ja yksinkertaisesti kuluttaa aikaa. Monet nuoret ovat siirtyneet kauppojen auloista, paikallisten hampurilaispaikkojen nurkilta ja muista vastaavista ajanviettopaikoista sosiaalisiin verkkosovelluksiin.

Tämän opinnäytteen tutkimuskysymykset ovat:

1. Minkälaisille tietoturvariskeille sosiaaliset verkkosovellukset ovat alttiita ?
2. Kuinka merkittävä rooli käyttäjällä on näiden tietoturvariskien toteutumisessa ?
3. Kuinka sovelluskehittäjä ja käyttäjä voivat välttyä näiltä tietoturvariskeiltä ?

Tutkimuskysymysten toivotaan myös vastaavan siihen, kuinka paljon sosiaalisten verkkosovellusten tietoturvallisuutta voidaan parantaa panostamalla sovelluksen laadullisiin ominaisuuksiin.

Sovelluskehittäjän näkökulma sisältää tietoturvan integroimisen sovelluskehitykseen, olennaisimpien tietoturvariskien ja hyökkäysten läpikäynnin, parannusehdotusten arvioinnin ja hyökkäysskenaarioiden empiirisen analysoinnin. Käyttäjän näkökulma käsittelee inhimillisen tekijän merkitystä tietoturvariskeissä, käyttäjien toiminnan arviointia verkkoyhteisöissä ja sosioteknisiä käyttäjien manipulointitapoja.

Sisältö

1 Taustaa, tavoitteita ja tutkimusasetelma

Seuraavassa kappaleessa tarkastellaan sosiaalista mediaa tällä hetkellä ja sitä minkälaisia ilmiöitä sosiaalisessa mediassa ilmenee. Tämän jälkeen tarkastelemme sosiaalisen median lähihistoriaa ja luodaan lyhyt katsaus joihinkin uusiin toiminnallisuuksiin. Luvun lopussa keskustellaan käytetyistä termeistä ja rajataan tarkemmin työn aihe.

1.1 Sosiaalisten verkkosovellusten asema nykypäivän maailmassa

Ajan myötä sosiaalinen kanssakäyminen on tullut yhä helpommaksi ja monimuotoisemmaksi. Tämä on houkuttanut laajempaa yleisöä, kun käyttö on tullut helpommin lähestyttävämmäksi ja käyttötapoja on tullut lisää. Aivan alun tietokoneasiantuntijoista internetin käyttäjäkunta on muuttunut kaikenkirjavaksi joukoksi. Kehitys on kulkenut blogien, tagien, wikien ja massiivimoninpelien viidakosta kuluneen vuosikymmenen suurimpaan verkkoilmiöön.

Ilmiön ovat tehneet mahdolliseksi muun muassa kotitietokoneiden yleisyys ja nykyisen sukupolven tietokoneiden hyvä tuntemus. Merkittävin muutos on kuitenkin tapahtunut siinä, kuinka käytämme WWWtä. Alkuaikojen WWW mahdollisti vain sivuston kehittäjien luoda sisältöä. Interaktio oli kovin yksipuolista, sivustot staattisia. Nyt käyttäjät voivat luoda itse sisältöä hyvinkin monipuolisesti. Verkosta on tullut vuorovaikutteisempi ja elävämpi. Konsepti on muuttunut radikaalisti, kun miljoonat ihmiset tuottavat sisältöä mitä erilaisemmilla tavoilla, tavoilla joita kehittäjät eivät aina edes osaa ennustaa. Tätä muutosta jotkut kutsuvat Web 2.0:n vallankumoukseksi.

Mikä Web 2.0 oikein on? Web 2.0 on suunnitteluperiaate, jonka painopiste on helppokäyttöisyydessä sekä palveluiden yhteisöllisessä ja dynaamisessa tuottamisessa. Joidenkin mukaan Web 2.0 on juuri se konsepti ja vaikuttaja, joka on mahdollistanut täysin uusia palvelukonsepteja sekä ihmisten verkostoitumisen uusilla ennennäkemättömillä tavoilla [1].

Web 2.0:n vallankumouksen ideaa eivät kaikki allekirjoita. Termiä on pitkään yritetty määrittää ristiriitaisin tuloksin. Monet ovat sitä mieltä, että Web 2.0 ei ole jotain uutta ja mullistavaa. Vastustajat uskovat, että kehitys on joka tapauksessa ollut menossa tähän suuntaan, jota Web 2.0:n kannattajat suitsuttavat. Heidän mukaansa

Web 2.0 on täysin turha termi, jota käytetään markkinointia ajavana kannustimena. Keskustelu on ollut kiivasta suuntaan ja toiseen. Jos ei muuta, niin Web 2.0 on ainakin luonut keskustelua ja tuonut uusia teknologioita ja liiketoimintamalleja esille. Seuraava lainaus, joka on Kari A. Hintikan kirjasta *Web 2.0-johdatus internetin uusiin liiketoimintamahdollisuuksiin* [2], vastaa ehkä eniten totuutta:
"'Web 2.0:n suurin ansio onkin ehkä juuri siinä, että se nosti näkyviin ensimmäisen kerran kokonaisvaltaisesti monia, mutta ei läheskään kaikkia, internetin yksittäisiä evoluutiopolkuja. Ne olisivat edenneet ilman erityistä termiäkin. Mutta niiden tarkastelu kokonaisuutena on luonut uusia palveluita, jotka yhdistelevät polkuja aiempaa tehokkaammin ja näin ovat kiihdyttäneet kokonaiskehitystä."

Teknisestä näkökulmasta tämän mahdollistanut muutos on varsin yksinkertainen vaikkakin sitä merkittävämpi: Web 2.0 on yhdistelmä joustavia ja avoimia verkkosovellusliittymiä ja se sisältö, jonka käyttäjät ovat itse verkkoon luoneet [1].

Kyseenalaista on se, onko tapahtunut niin suuri muutos, että voitaisiin puhua Web 1.0:sta ja Web 2.0:sta. Onko paremminkin kyse Web 1.5:stä? Oli miten oli, tämä muutos jota voidaan kutsua nimellä Web 2.0 on luonut pohjan lukuisille uusille, innovatiivisille verkkosovelluksille.

1.1.1 Sosiaaliset verkkosovellustyypit

Eräs vanha jako sosiaalisille verkkosovelluksille on seuraavanlainen [3]:

1. Kommunikaatio (pikaviestimet)
2. Kokemusten jakaminen (kuva-albumit)
3. Uusien ja vanhojen tuttavuuksien löytäminen (Classmates.com)
4. Suhteiden ylläpitäminen (Friendster)
5. Verkkopelaaminen (massiivimoninpelit)

Artikkeli on vuodelta 2004, joten se on väistämättä vanhentunut WWWn suurten muutosten takia. Artikkeli on ilmestynyt juuri Wikipedian yleistymisen paikkeilla ja siitä puuttuukin kategoria, jota Wikipedia edustaa: Tiedon lisääminen. Nyt tulee huomioida erityisesti ero termien tieto ja informaatio välillä. Tieto on merkityksellisesti koostettua ja jäsennettyä informaatiota. Tässä opinnäytteessä informaatio on mikä tahansa päätelty tai havaittu uskomus, totta tai ei. Toisaalta tiedolla voidaan viitata verkkosivun tyylitiedoston väriasetuksiin, mutta kategorialla (Tiedon lisääminen) tässä kontekstissa viitataan tällöin sisältöön, ts. mitä ja minkälaista tietoa esi-

tetään? Tässä kontekstissa bloggaus ei useimmiten ole tiedon lisäämistä. Mikä sitten on tietoa, riippuu siitä mikä lasketaan informaation merkitykselliseksi muodoksi.

Tämän lisäksi kategoriat 3 ja 4 (uusien ja vanhojen tuttavuuksien löytäminen, suhteiden ylläpitäminen) ovat hieman päällekkäisiä. Jos löydetään vanha tuttavuus, eikö tämän kanssa usein myös aleta pitämään yllä tuttavuussuhdetta uudestaan? Sama pätee uusiin tuttavuussuhteisiin. Korvataan kategoriat 3 ja 4 uudella kategorialla, sosiaalisella verkostoitumisella. Termiin kuuluu siis nyt tuttavuussuhteiden löytäminen, niin uusien kuin vanhojenkin, sekä näiden ylläpitäminen. Suhteiden yhdistävänä tekijänä voi olla muun muassa yhteinen harrastus, vanha kaverisuhde tai ammatillinen suhde. Saadaan seuraavanlainen jako:

1. Kommunikaatio (pikaviestimet)
2. Kokemusten jakaminen (kuva-albumit)
3. Sosiaalinen verkostoituminen (Classmates.com)
4. Tiedon lisääminen (Wikipedia)
5. Verkkopelaaminen (massiivimoninpelit)

Tosiasia on, että ihmiset ovat kanssakäyneet internetin välityksellä alusta lähtien, oli Web 2.0:aa tai ei. Metodit ja alustat ovat vain olleet erilaisia. Esimerkiksi käyttäjistä, jotka keskustelevat toistensa kanssa sähköpostin välityksellä, voidaan muodostaa sosiaalinen verkosto. Sähköpostista on kuitenkin vielä hyvin pitkä matka sosiaalisiin verkkosovelluksiin, kuten Facebookiin, Orkutiin tai Twitteriin. Näistä sovelluksista voidaan käyttää myös termiä *sosiaalinen media*. Termit *yhteisöpalvelu*, *verkkoyhteisö* ja *sosiaalinen verkkosivusto*, tai jotkut näiden variaatiot, esiintyvät myös usein. Tarkennetaan näitä termejä Sanastokeskus TSK:n mukaisten määritysten mukaisesti [4].

Monia aiheeseen liittyviä termejä käytetään sekaisin tai niiden tulkinta vaihtelee [4]. Sekaannusta on lisännyt se, että eri verkkosovellustyypit ovat lähestyneet toisiaan ja omaksuneet laajemman kirjon ominaisuuksia. Monet ennen puhtaasti sosiaaliseen verkostoitumiseen keskittyvät verkkosovellukset ovat omaksuneet monia median jakamisominaisuuksia ja vastaavasti toisinpäin. Päällekkäisyyttä on runsaasti. Parhaimmillaan jotkut sosiaaliset verkkosovellukset voivat kattaa kaikki yo. kategoriat, esimerkiksi Facebook kattaa kaikki viisi.

Sosiaalisella verkkosovelluksella voidaan tarkoittaa mitä tahansa verkkoa hyödyntävää sovellusta tai palvelua, jota käytetään kommunikoimiseen ja yhteistyö-

hön. Englanninkielinen nimi termille on *social software*. Tähän joukkoon kuuluu myös mm. sähköposti ja pikaviestimet.

Sanastokeskus TSK määrittelee **sosiaalisen median** seuraavasti [4]:
"Tietoverkkoja ja tietotekniikkaa hyödyntävä viestinnän muoto, jossa käsitellään vuorovaikutteisesti ja käyttäjälähtöisesti tuotettua sisältöä ja luodaan ja ylläpidetään ihmisten välisiä suhteita. Sosiaaliselle medialle tyypillisiä verkkopalveluita ovat esimerkiksi sisällönjakopalvelut, verkkoyhteisöpalvelut ja keskustelupalstat."

Edelleen termit yhteisöpalvelu ja verkkoyhteisö on määritelty [4]:
"Palvelu, joka tarjoaa mahdollisuuden ihmisten välisten suhteiden luomiseen ja ylläpitämiseen tietoverkon kautta." Näitä termejä voidaan käyttää, kun halutaan korostaa tarkasteltavan kohteen yhteisöä tai yhteisöllisyyttä. Sosiaalisella verkkosivustolla voidaan tarkoittaa yleisesti ottaen sivustoa, joka toimii portaalina sosiaaliselle verkkosovellukselle tai verkkoyhteisölle.

Tämä opinnäyte keskittyy pääasiallisesti *profiilipohjaisiin*, usein identiteetin hallintaan perustuvien, *sosiaalisen median verkkosovelluksiin*. Profiilipohjainen sosiaalinen media tarkoittaa tässä egosentristä verkkosovellusta, jossa käyttäjää edustaa hänen profiilinsa. Kaikki käyttäjän toiminta tapahtuu profiilin kautta. Yllä olevista termeistä voitaisiin käyttää mm. yhteisöpalvelua, joskin osuvin olisi sosiaalinen media. Koska tässä työssä huomio on erityisesti ohjelmistossa ja näiden ohjelmistojen käyttäjissä —siltä osin kuin se koskee tietoturvaa— käytetään termiä sosiaalinen verkkosovellus, jotta voidaan tuoda paremmin esille opinnäytteen tutkimusasetelma.

1.1.2 Sosiaalisten verkkosovellusten vaikutuksia

Sosiaaliset verkkosovellukset ovat edenneet valtavirtaan ja kasvaneet valtaviksi yhteisöiksi. Suuri suosio on aiheuttanut mielenkiintoisia ja välillä yllättäviäkin sivuvaikutuksia. Koska käyttäjät viettävät näissä sovelluksissa paljon aikaa ja vaihtavat keskenään monenlaista informaatiota, monet muut tahot kuin kehittäjät ja käyttäjät ovat kiinnostuneita kaikesta siitä, mitä näissä sovelluksissa tapahtuu. Työnantajat ovat huolissaan paljastavatko heidän työntekijänsä luottamuksellisia tietoja. Moni käyttäjä on saanut potkut työpaikaltaan moittiessaan työnantajaansa sosiaalisissa verkkosovelluksissa. On nykyään tavallista, että työnantajat tutkivat mitä tietoja työpaikan hakijoista löytyy verkosta ja usein he törmäävätkin esimerkiksi työnhakijan Facebook-tilin sivulle. Poliisit hyödyntävät sosiaalisia verkkosovelluksia toimissaan. Hiljattain on ilmennyt, että FBI:n rikostutkijoita koulutetaan käyttämään verkon yhteisöpalveluja, kuten Facebookia, hyväkseen [5]. He luovat tekaistuilla nimil-

lä tilejä ja pyrkivät soluttautumaan rikosepäiltyjen ystäväverkostoihin. Media seuraa innokkaasti suurimpia sosiaalisia verkkoyhteisöjä. Palveluiden kehittäjät, ohjelmistotalot, sponsorit ja mainostajat pohtivat kuumeisesti liiketoimintamalleja hyödyntämään sosiaalisten verkkosovellusten valtavaa suosiota. Sosiologian, tietoverkkojen ja verkkoteorian tutkijat hyödyntävät taas sosiaalisia verkkosovelluksia tutkimuksissaan. Kaikille raha ei ole motivoiva voima, otetaan esimerkiksi rahoituksensa työstä riippumattomista syistä saavat tutkijat tai virkavalta.

Lainsäädäntö on tunnetusti tullut jäljessä internetin ilmiöihin. Tapauksia kuitenkin on, missä käyttäjät ovat joutuneet vastaamaan oikeudessa sanomisistaan tai tekemisistään sosiaalisissa verkkoyhteisöissä [6]. Monia rikoksia on suoritettu sosiaalisten verkkosovellusten välityksellä. Näiden tapausten julkituominen voi kenties muuttaa ihmisten käyttäytymistä verkossa tavanomaisempaan suuntaan ja lisätä ihmisten valveutuneisuutta. Yksityisyyden loukkaaminen ja nettiväkivalta ovat yleisiä tapauksia. Toimijana voi olla ahdistelija, pedofiili tai henkilö joka pyrkii agitoimaan ihmisiä poliittisista tai muista syistä.

Monet haluavat siivunsa kakusta ja pyrkivät tienaamaan rahaa sosiaalisten verkkosovellusten siivellä, yhteistyössä kehittäjien kanssa tai ilman. Verkkorikolliset ovat kiinnittäneet huomionsa sosiaalisiin verkkosovelluksiin, ja kannustimena yhä useammin on raha. On havaittavissa, että tämä trendi tulee voimistumaan (ks. kpl 3.5). Kannustimena voi toimia myös vahingonteko, maineen tavoittelu tai taitojen testaaminen.

On tarpeen huomioida myös kehittäjät. Kehittäjät voivat tahallisesti tai tahattomasti rikkoa käyttäjiensä kanssa tekemiä käyttöehtosopimuksia. Nämä yhtä lailla ovat sosiaalisten verkkosovellusten tietoturvallisuusriskejä.

1.2 Sosiaalisten verkkosovellusten menneisyys ja trendit

Friendster on vuonna 2002 perustettu sosiaalinen verkkosovellus, joka ei perustunut ihmisten yhteisiin kiinnostuksen kohteisiin, kuten monet sitä edeltävät verkkosovellukset, vaan perustui ihmisten tuttavuussuhteisiin. Käyttäjiä sitoi nyt yhteen tuttavuudet ja huomio keskittyi käyttäjiin. Käyttäjät verkostoituivat nopeammin tuttujensa kautta, kuin tuntemattomien, samat intressit jakavien ihmisten kanssa. Friendster kasvoi nopeasti ja vuonna 2003 se alkoi saamaan myös median huomion. Median noteeraus toi Friendsterille paljon kaivattua huomiota ja tämä kasvatti käyttäjämäärää entisestään.

Friendsterissä verkkoyhteisöjen ytimessä eivät olleet ryhmät vaan käyttäjien profiilit. Friendsterin lähestymistapa sosiaalisiin verkkosovelluksiin oli merkittävä ja vuoden 2003 aikana alkoikin ilmestyä vastaavankaltaisia sosiaalisia verkkosovelluksia, jotka toivoivat saavuttavan samanlaisen suosion kuin Friendster. Käyttäjän profiileihin pohjautuvat sosiaaliset verkkosovellukset saivat tuulta alleen ja tähän ideaan pohjautuvia, erilaisiin käyttötarkoituksiin keskittyviä yhteisöpalveluja alkoi ilmestyi nopeasti. Sosiaaliset verkkosovellukset olivat tulleet valtavirran tietoisuuteen. Trendin yhä voimistuttua, alkoivat sosiaalisen median verkkosovellukset omaksua puhtaasti sosialisoivia ominaisuuksia. Youtube, Flickr ja Last.fm ovat esimerkkejä tällaisista sosiaalisen median verkkosovelluksista.

MySpace perustettiin vuonna 2003 kilpailemaan Friendsterin kaltaisten sosiaalisten verkko sovellusten kanssa. Kun huhut levisivät, että Friendster alkaa perimään jäsenmaksuja käyttäjiltään, alkoivat käyttäjät etsimään samankaltaisia, ilmaisia sosiaalisia verkkosovelluksia. MySpace sai paljon uusia käyttäjiä tästä muuttoliikkeestä. MySpace tuli tunnetuksi myös siitä, kuinka se yhdisti käyttäjät ja musiikintekijät tarjoamalla yhteisen median. Kaikki kolme osapuolta hyötyivät tästä yhteydestä. MySpace lisäsi ominaisuuksia, jotka sallivat käyttäjien personalisoida profiilejaan, joka lisäsi MySpacen suosiota entisestään. Vuonna 2005 MySpace ohitti jo silloin dominoivassa asemassa olevan Googlen sivulatauksien määrässä. Vuonna 2006 MySpacesta oli tullut Yhdysvaltojen suosituin sosiaalinen verkkosivusto.

Facebook on vuonna 2004 perustettu sosiaalinen verkkosovellus, joka aluksi oli rajoitettu tiettyjen yliopistojen opiskelijoihin. Myöhemmin Facebook alkoi hyväksymään käyttäjiä myös yliopistojen ulkopuolelta, ja lopulta vuonna 2006 kaikki yli 13-vuotiaat hyväksyttiin Facebookin käyttäjiksi. Tämän liikkeen myötä Facebook kohosi pian MySpacen varteen otettavaksi kilpailijaksi maailman suosituimman verkkoyhteisön tittelistä. Vuonna 2009 Facebook ohitti MySpacen suosiossa [7]. Historia toistaa itseään: 2010 maaliskuussa analyysiyhtiö Experian Hitwisen mukaan [8] Facebook ohitti Yhdysvalloissa Googlen sivulatauksien määrässä. Ei ollut enää epäselvää, mikä on tämän hetken suosituin sosiaalinen verkkosovellus.

Opinnäytteen kirjoittamishetkellä Facebookilla on yli 400 miljoonan käyttäjäkunta ja parhaillaan Facebookin kehittäjät suunnittelevat laajentavan toimintaansa Intiaan. Kuinka paljon 400 miljoonasta käyttäjästä on aktiivikäyttäjiä, on epäselvää. Facebookin oman statistiikan valossa 50 % 400 miljoonasta käyttäjästä kirjautuu tiliinsä päivittäin ja 70 % käyttäjäkunnasta tulee Yhdysvaltojen ulkopuolelta. Tulee myös huomioida, että yhdellä henkilöllä voi olla peitenimellä useampia tilejä. Ei voida siis suoraan päätellä tämän statistiikan pohjalta todellisten henkilöiden määrää, jotka aktiivisesti käyttävät Facebookia. Tosin sama pätee muihinkin sosiaalisiin

verkkosovelluksiin.

Twitter on vuonna 2006 perustettu niin sanottu mikrobloggauspalvelu [9]. Käyttäjät luovat profiilin, johon he lähettävät 140 merkin viestejä, "tweettejä", joissa he kertovat tekemisistään. 140 merkin raja tulee siitä, että alunperin Twitter suunniteltiin pelkästään mobiilikäyttöön. Idea on siis samankaltainen kuin blogeissa, joissa vapaamuotoisesti kerrotaan kiinnostuksen kohteista tai jota pidetään verkkopäiväkirjana. Erona blogeihin on että, tweettejä ei voi kommentoida. Twitterin kehittäjät eivät ole julkaisseet aktiivisten käyttäjien määrää, mutta compete.comin tilastojen mukaan Twitterillä on yli kaksikymmentä miljoonaa uniikkia käyttäjää [10]. Twitter on kasvanut huimasti lähivuosina. Tammikuussa 2010 tweettauksien määrä oli kasvanut n. 1400 % vuodessa, ylittäen 35 miljoonan tweetin määrän per päivä [11]. Koska moni tekee tilin Twitteriin seuratakseen vain jotain julkisuuden henkilöä, ei ole niin selvää kuinka tweettien määrä korreloi käyttäjämäärän kanssa.

Myös muualla maailmassa sosiaaliset verkkosovellukset kasvattivat suosiotaan. Yleisesti ottaen englantia puhuvissa maissa samat sosiaaliset verkkosovellukset ovat saaneet valta-aseman. Alueellisia eroja kuitenkin löytyy. Googlen Orkut on suosituin verkkoyhteisö Brasiliassa yli 100 miljoonalla käyttäjällään. Isossa-Britanniassa, Australiassa, ja Uudessa-Seelannissa Bebo kasvoi suurimmaksi sosiaaliseksi verkkosovellukseksi. Kun kiinalainen pikaviestinpalvelu QQ lisäsi profiilit palveluunsa, tuli se maailman suurimmaksi yhteisöpalveluksi. Suomessa Irc-Galleria on ollut erittäin suosittu. Kaikki sosiaaliset verkkosovellukset eivät ole geneerisiä, vaan jotkut sovellukset keskittyvät tietyn rajatun tematiikan ympärille. Vaikka tarjottujen palveluiden määrä on suuri, monet sosiaaliset verkkosovellukset ovat hyvin lähellä toisiaan ulkoisesti tarkasteltuna kun uudet innovaatiot kopioidaan nopeasti [12]. Esimerkiksi Facebook on pysynyt hyvin ajan hermolla ja on nopeasti omaksunut uusimmat trendit eri palveluiden muodossa.

Kasvamisen trendi on yhä voimakas. Yhdysvalloissa elokuussa 2009 päättynyt tutkimus mittasi ihmisten sosiaalisissa verkkosovelluksissa vietettyä aikaa [13]. Tutkimuksen mukaan ihmisten sosiaalisissa verkkosovelluksissa viettämä aika kolminkertaistui vuodessa. Merkille pantavaa on myös mainonnan kasvu, joka kasvoi 119 % samassa ajassa. Nykyään onkin tavallista näillä sivustoilla nähdä mainoksille varattu tila. Todennäköistä on, että mainontaa kohdennetaan profiileista saatujen tietojen pohjalta.

Sosiaaliset verkkosovellukset ovat osoittautuneet tehokkaiksi työkaluiksi informaation levittämisessä. Informaation levittämisen kustannukset ovat hyvin pienet ja erilaisten ryhmien tai tematiikan ympärille rakentuvissa yhteisöissä myös informaation kohdentaminen käy helposti. Mielipiteiden, kannanottojen tai tiedotusten

ilmaiseminen ei vaadi fyysistä paikallaoloa. Tämä antaa mahdollisuuden, ja alentaa kynnystä, ottaa osaa päivän polttaviin keskusteluihin paikasta riippumattomasti, mahdollistaen voimakkaan vuorovaikutuksen ja entistä suuremman informaation läpäisevyyden. Monet poliittiset vaikuttajat, aktivistit, organisaatiot, musiikkiyhtyeet ja muut ovat panneet tämän merkille ja ovat tehokkaasti hyödyntäneet verkkoyhteisöjä. Obaman vaalikoneisto käytti taitavasti hyväkseen kampanjoinnissa verkkoyhteisöjä, mm. Youtubea [14]. Esimerkki informaation leviämisestä sosiaalisissa verkkosovelluksissa on Haitin maanjäristykset [15]. Kun paikallinen infrastruktuuri oli lähes tuhoutunut, rikkoen mm. puhelinverkon, oli yhteyden saaminen Haitiin hyvin hankalaa. Tekstiviestit mahdollistava infrastruktuuri toimi kuitenkin, mikä nosti Twitterin Haitin maanjäristysten suurimmaksi informaation jakelukanavaksi.

Reaaliaikaisuus ja paikannusjärjestelmät ovat uusia sosiaalisiin verkkosovelluksiin tulossa olevia trendejä [16]. Twitterin luonteeseen olennaisesti kuuluu reaaliaikaisuus. Facebook seurasi perässä ja otti käyttöönsä vuonna 2009 Live Feed-palvelun, joka kerää ja esittää käyttäjän kaverien profiilien päivitykset kootusti. Reaaliaikaisuus tehostaa chat -palveluiden ohella sosiaalisten verkkosovellusten kommunikoinnin nopeutta. Paikannusjärjestelmät voivat jäljittää käyttäjän sijainnin matkapuhelimen kautta. Kun käyttäjän sijainti tiedetään, voidaan tämä tieto yhdistää yleisen karttapalvelun tietoihin, esimerkiksi GoogleMapsin, saaden osoitteen käyttäjän nykyiselle sijainnille. Palvelulla voidaan lisätä entisestään läpinäkyvyyttä käyttäjien toimiin. Paikannusta voidaan käyttää suunnistuksessa, esimerkiksi auttamaan käyttäjää löytämään tietty liike, konserttisali tai toinen käyttäjä. Facebook on julkaissut oman paikannuspalvelunsa, Twitter kehittää parhaillaan omaansa [18] [19].

1.3 Tavoitteet ja rajaus

Tässä opinnäytteessä keskitytään pääasiallisesti järjestelmän ulkopuolisiin, tietokoneilla suoritettaviin, hyökkäyksiin. Organisaation sisäiset, onnettomuuksista johtuvat, tahattomat ja muut tietoturvariskit käsitellään lyhyesti tai ei ollenkaan. Sosiaalisten verkkosovellusten osalta huomio keskittyy egosentrisiin palveluihin, jossa kutakin käyttäjää edustaa ainutlaatuinen profiili. Profiili kuvaa käyttäjän virtuaalista identiteettiä, jonka hän omaksuu verkkoyhteisössä. Kaikki käyttäjän toiminta tapahtuu profiilin kautta.

2 Sosiaalisten verkkosovellusten tietoturvan haasteet

Luodaan katsaus toteutuneisiin uhkatilanteisiin. Tämän jälkeen tutustutaan digitaaliseen jalanjälkeen, joka on tärkeä käsite. Verkkoteoria auttaa ymmärtämään joitain sosiaalisten verkostojen erityispiirteitä. Kappaleessa 3.4 pohditaan sitä, onko olemassa sellaista ihmisen käyttäytymistä, että se lisää tietoturvariskien muodostumisesta. Lopuksi pohditaan luottamuksen ongelmaa, jota monet hyökkäykset käyttävät hyväkseen.

2.1 Katsaus toteutuneisiin uhkatilanteisiin

Sosiaalisten verkkosovellusten tietoturva on murrettu moneen otteeseen. Ensimmäinen mato joka on onnistunut leviämään tehokkaasti sosiaalisissa verkkosovelluksissa on ollut Sammy. Sammy iski vuonna 2005 Myspaceen ja alkoi leviämään hyvin nopeasti. Sammy -mato ei hyödyntänyt käyttäjien profiilitietoja, vaan oli haitaksi verkkoyhteisön yleiselle toiminnalle. Vuoden 2009 huhtikuussa Mikeyy mato alkoi levitä Twitter -mikrobloggauspalvelussa. Mikeyy mato lisäsi saastutettuihin profiileihin tekstiä. Kumpikin näistä madoista on hyvin harmittomia. Ne todistavat näiden järjestelmien haavoittuvuuksista yhtä kaikki [20].

Koobface on ehkä tunnetuin sosiaalisia verkkosovelluksia piinaavista madoista. Koobface ei kuulu harmittomien matojen joukkoon. Koobface havaittiin ensimmäisen kerran joulukuussa 2008. Nimestään huolimatta Koobface ei ole kohdistunut pelkästään Facebookiin. Koobface kykenee ainakin vahingontekoon seuraavissa sosiaalisissa verkkosovelluksissa: Myspace, Facebook, Twitter, Bebo, hi5 ja Friendster. Koobfacen odotetaan kuitenkin pystyvän saastuttamaan muitakin sosiaalisia verkkosovelluksia [21]. Koobface käyttää hyväkseen ihmisten luottamussuhdetta toisiin ihmisiin tai tarkemmin ystäviin, kuten tuttavuuksia usein kutsutaan sosiaalisissa verkkosovelluksissa. Koobface esiintyy kohteen ystävänä, lähettäen viestin tälle, joka sisältää linkin verkkosivuston ulkopuolelle. Tällainen linkittäminen, hyperlinkkien tarjoaminen kiinnostavista ja päivän polttavista aiheista muille tutuille, on hyvin tavallista sosiaalisten verkkosovellusten käyttäjille. Kun viesti näyttää tulevan ystävältä, on ihmisille luontaista luottaa sen sisältöön. Viesti sisältää yleensä jonkin lyhyen tekstin, josta on vaikea päätellä viestin autenttisuutta. Linkki ohjaa käyttäjän taitavasti Youtube palvelun kaltaiseksi tehdylle verkkosivustolle, jossa käyt-

täjää ohjeistetaan asentamaan Adobe Flash Player-liitännäisen uusin versio. Tällöin käyttäjän koneelle asennetaan jokin Koobfacen komponentti. Saastutettuaan koneen Koobface kykenee valjastamaan käyttäjän tilin lähettääkseen samanlaisia harmittomiksi viesteiksi naamioituja hyökkäyksiä edespäin.

Kolmannen osapuolen tekemät pienohjelmat tarjoavat lukuisia uusia käyttötarkoituksia sosiaalisille verkkosovelluksille. Pienohjelmien tuottajana voi toimia periaatteessa kuka tahansa. Monilla verkkoyhteisöillä, kuten Facebookin kehittäjillä [22], ei ole pienohjelmien arviointiprosessia. Taitavasti luodut, aidoilta vaikuttavat verkkorikollisten luomat pienohjelmat soluttautuvat vaivattomasti liitännäisten kirjavaan joukkoon. On myös kyseenalaista, kuinka paljon aikaa ja resursseja sovellusliittymiin kytkettyjen pienohjelmien perinpohjainen varmistaminen veisi. Kaikkein massiivimpien sosiaalisten verkkosovellusten, kuten Facebookin, tapauksessa jokaisen sovellusliittymän varmistaminen olisi valtava tehtävä. Pienohjelmien luonti on usein melko vaivatonta valmiiden, määriteltyjen rajapintojen kautta. Kun ottaa vielä huomioon, että sovellusliittymät mahdollistavat käyttäjien informaation muokkauksen dynaamisilla tavoilla, ei ole ihme, että verkkorikolliset ovat hyvin kiinnostuneita siitä, kuinka sovellusliittymät voisivat palvella heidän rikollisia tarkoitusperiään.

Eräs esimerkki haitallisesta pienohjelmasta on vuonna 2009 ilmestynyt *F a c e b o o k - closing down!!!* -liitännäinen [23]. Käyttäjille pienohjelma näkyi nimellä *Error Check System*. Pian sovelluksen asennuksen jälkeen käyttäjää informoitiin, että hänellä oli jotain vialla profiilissaan. Korjatakseen tämän, käyttäjän tuli klikata pienohjelman tarjoamaa ulkoista linkkiä. Jos käyttäjä klikkaa linkkiä ja etenee tälle saastutetulle sivulle, tekaistu virustorjuntaohjelma käynnistyy varoittaen käyttäjää olemattomista haitakkeista. Jos käyttäjä menee ansaan ja toimii väärennetyn virustorjuntaohjelman ehdottamalla tavalla, tämä asentaa käyttäjän koneelle viruksen.

Madot tai liitännäiset eivät suinkaan ole olleet ainoita uhkia sosiaalisten verkkosovellusten tietoturvalle. Vuonna 2007 Facebookin kehittäjät syyttivät erästä firmaa tiedonlouhinnasta [24]. Yritys oli kirjoittanut skriptejä, jotka keräsivät automatisoidusti Facebookin omia työkaluja käyttäen profiilitietoja. Vuonna 2009 sekä Facebook että Twitter ovat olleet palvelunestohyökkäysten kohteena useaan otteeseen. Tarkoituksena on voinut olla vahingonteko tai kohteena olevien järjestelmien tietoturvan sietokyvyn testaaminen [25]. Aikaisemmin samana vuonna Twitteristä usean julkisuuden henkilön profiilit kaapattiin, mm. Yhdysvaltain presidentin. Tämän hyökkäyksen tarkoituksena oli kiusanteko, kyseessä ei ollut poliittisen merkityksen omaava hyökkäys [26]. Poliittisia tarkoitusperiä omaavia hyökkäyksiä on myös esiintynyt [27]. Yhdysvaltojen kyberrikollisuutta tutkiva organisaatio Cyber

Consequences Unit väittää, että Venäjä-Georgia sodassa 2008 hyökkäyksiä järjestettiin sosiaalisten verkkosovellusten avulla. Väitteen mukaan venäläiset kräkkerit varastivat mm. Facebookista ja Twitterista käyttäjätilejä ja käyttivät näitä hyväkseen kaataakseen Georgian hallinnollisten elinten verkkosivuja. Hyökkäykset verkkosivustoille ajoitettiin tapahtumaan samanaikaisesti varsinaisten sotatoimien kanssa. Tavoitteena lienee ollut informaatioväylien häirintä kriittisellä hetkellä shokkiefektin maksimoimiseksi.

2.2 Digitaalinen jalanjälki

Ihmisistä kertyy valtava määrä digitaalista informaatiota eri laitteisiin ja tietokantoihin. Valvontakamerat, maksulliset tv-kanavat, internetin käyttö, tilisiirrot ja käteisnostot ovat esimerkkejä digitaalisista jalanjäljistä, joita me jätämme jälkeemme. Valvontakameraan tallentuu käynti läheisessä ostoshallissa, tilisiirroista on nähtävissä liikeasioinnit ja muut rahalliset transaktiot, selaimien kekseistä voidaan saada selville kuinka kauan on vietetty aikaa verkkosivulla, verkossa käytetyt hakusanat voidaan poimia, jne... Digitaalisen tiedon määrä ihmistä kohden tulee lisääntymään tulevaisuudessa, kun mm. internetin käyttö mobiililaitteilla yleistyy ja sulautetut järjestelmät lisääntyvät, digitalisoiden yhteiskuntaa entisestään. Digitaalisen informaation kokonaismäärää kasvattaa kehittyvien maiden digitalisoituminen. Odotettavaa on myös sosiaalisten verkkosovellusten käytön lisääntyminen väestön ikääntyessä ja vanhempien sukupolvien väistyessä. Täten, sosiaalisista medioista löytyvän informaation määrä lisääntyy ja digitaalinen jalanjälki näkyy vahvemmin näissä yhteisöissä.

2.2.1 Digitaalisen jalanjäljen yksityisyysongelmat

Tavallista käyttäjää voi askarruttaa kysymys, kuinka tietovaras voi hyötyä hänen digitaalisesta jalanjäljestään? Tietoturvayhtiö Symantec julkaisi huhtikuussa 2010 raporttinsa internetissä piilevistä uhkista [28]. Raportista ilmenee internetin alamaailmassa, mustassa pörssissä, käytettäviä hintoja eri hyödykkeille. Myyntiartikkeleina ovat mm. luottokorttitiedot, sähköpostiosoitteet ja salasanoja verkkosivustojen järjestelmänvalvojan käyttöoikeuksilla. Sähköpostiosoitteita käytetään roskapostitukseen, verkkosivuston järjestelmänvalvojan oikeuksilla voidaan kyseisiä verkkosivustoja muokata hyvinkin monipuolisesti esimerkiksi saastuttaen sivustot haittakoodilla. Hyökkääjillä voi olla poliittinen agenda ja he voivat olla kiinnostuneita käyttäjien poliittisesta suuntautumisesta. Käyttäjällä voi olla arvokasta tietoa työ-

paikkansa puolesta hyökkääjille, joka ei välttämättä ilmene käyttäjälle. Hyökkääjät ovat voineet ottaa kohteekseen yrityksen ja etsiä digitaalisesta jalanjäljestä yhteyksiä kyseiseen yritykseen. Yrityssalaisuudet kiinnostavat hyökkääjiä, sillä ne voivat olla heille, tai heitä työllistäville tahoille, rahanarvoista tietoa. Avainasemassa olevia käyttäjiä voidaan yrittää taivuttaa luovuttamaan arvokasta tietoa.

Varovainen käyttäjä, sellainen joka ei juurikaan itsestään jälkiä jätä, voi kyseenalaistaa yhä, että mitä haittaa hänelle voi olla siitä, että hän ilmoittaa sosiaalisella verkkosivustolla jostain arkisesta toimestaan. Käyttäjän tulee ymmärtää mitkä tiedot hän on merkinnyt julkisiksi. Julkisia tietoja voi seurata työnantaja, vanhemmat ja käytännössä kuka tahansa jolla on pääsy internetiin. Jos käyttäjä ilmoittaa jättävänsä kotinsa pariksi päiväksi, hän ei ehkä tule ajatelleeksi, että kyseisen tiedon voi lukea rikollinen, joka suunnittelee seuraavaa murtovarkauttaan. Käyttäjien profiilitiedoista ilmenevät tiedot kiinnostavat mainostajia. Joitakin käyttäjiä henkilökohtaisten tietojen käyttäminen mainontaan ei häiritse. Liiallinen pidättäväisyys ja skeptisyys ei ole järkevää, mutta on hyvä olla tietoinen siitä, kuinka eri tahot kykenevät hyödyntämään jättämiämme digitaalisia jalanjälkiä.

Oleellinen ominaisuus digitaalisen jalanjäljen tuomissa haasteissa on digitaalisen informaation pysyvyyden ongelma. Jos informaatio poistetaan niiltä sivustoilta, jonne se siirretään, ongelma ei ole vielä ratkaistu. Digitaalisen informaation vahvuus, kopioitavuus, yhdessä nopeiden tiedonsiirtoyhteyksien kanssa mahdollistaa informaation erittäin nopean propagaation. Verkkoon kerran siirretty data kopioituu hyvin nopeasti hakukoneiden tai selaimien välimuisteihin. Mikään ei estä datan lataamista tietokoneiden kiintolevyille. Käyttäjien tulisi olla tietoisia tästä, kun he siirtävät verkkoon arkaluontoista materiaalia.

2.2.2 Mitä tietoja on saatavilla ?

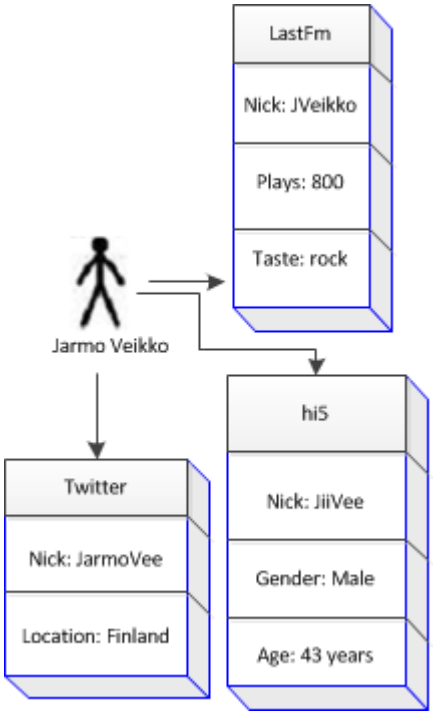
Meidän tulee luottaa suureen joukkoon ammattilaisia —tarkemmin heidän ammattietiikkaansa— jotta tietojamme, joissakin tapauksissa arkaluontoisia, käsiteltäisiin asianmukaisella tavalla. Poliisit, virkamiehet, pankit, puhelinoperaattorit ja mm. verkko-operaattorit säilyttävät asiakkaistaan erinäisiä tietoja. Tarkastellaan seuraavaksi mitä tietoja sosiaalisista verkkosovelluksista yleensä löytyy.

Helpoiten saatavilla on nimi. Monet verkkoyhteisöt kieltävät käyttöehtosopimuksissaan esiintymisen väärällä nimellä, pyrkien edistämään yhteisön avoimuutta. Nimen selville saaminen avaa monia uusia mahdollisuuksia. Hakukoneita käyttämällä voidaan nopeasti löytää onko henkilöllä profiileja muissa sosiaalisissa verkkosovelluksissa, kotisivuja tai muita merkintöjä, esimerkiksi päiväkodissa pidettä-

vien lasten vanhemmat löytyvät internetistä usein. Yhteystiedot ja osoite ovat usein varsin helposti saatavilla, erityisesti sähköpostiosoite. Jotkut tietokannat ovat julkisia, mm. monet julkishallinnon rekisterit ovat vapaasti kansalaisten käytettävissä verkon välityksellä. Näitä tietoja käymällä läpi, varsinkin muista sosiaalisista verkostoista löytyviä tietoja hyödyntämällä, voidaan alkaa rakentamaan pala kerrallaan henkilön digitaalista jalanjälkeä. Tiedon rikastamisprosessiin kuuluu duplikaatti-informaation poistaminen ja informaation jalostaminen tiedoksi. Riippuen siitä mitä sosiaalisia verkkosovelluksia henkilö käyttää, selville voidaan saada nimi, ikä, sukupuoli, kotikaupunki, siviilisääty, lemmikkieläimen nimi, harrastukset, mieltymykset (musiikki, ruoka, kirjallisuus, elokuvat, jne..), tämänhetkinen sijainti ja niin edelleen. Digitaalisesta jalanjäljestä, joka käyttäjästä jää sosiaalisiin verkkosovelluksiin, voidaan käyttää nimeä sosiaalinen verkkojalanjälki.

Riippuen käytetyistä sosiaalisista verkkosovelluksista, sosiaalisen verkkojalanjäljen informaatiosisältö voi vaihdella laajasti. Informaatio syötetään kenttiin. Hyvin harvat kentät ovat pakollisia, joskin käyttäjätilin luomiseksi vaaditaan sovelluksesta riippuen joitain tietoja. Tavallisimpia kenttiä ovat nimi, sukupuoli ja ikä mutta käytännössä informaatio voi koskea mitä vain.

Irani ym. [29] tutkivat mitä tietoja tiedonlouhinnalla on löydettävissä sosiaalisista verkkosovelluksista ja sitä, kuinka helposti tavanomaisilla, julkisilla keinoilla tämä informaatio on saatavilla. Tiedonlouhinta on kielletty useimpien sosiaalisten verkkosovelluksien käyttöehtosopimuksien mukaan, joten tutkijoiden tulee lähes poikkeuksetta pyytää lupa tiedonlouhintaan. Noin 55 000 henkilön otoksesta [29] sosiaalisten verkkosovelluksen käyttäjällä on keskimäärin käyttäjätili 5.7:ssä muussa sosiaalisessa verkkosovelluksessa. Otos on kerätty vuonna 2008, joten luku on todennäköisemmin tänä päivänä suurempi. Mitä useampia sosiaalisia verkkosovelluksia henkilö oli käyttänyt, sitä todennäköisemmin henkilö paljasti enemmän informaatiota itsestään, ts. oli täyttänyt useampia kenttiä. Henkilö jolla oli vain yksi käyttäjätili oli keskimäärin täyttänyt 4.3 kenttää. Henkilö jolla oli käyttäjätilit viidessä eri sosiaalisessa verkkosovelluksessa oli keskimäärin täyttänyt 7.6 kenttää. Merkille pantavaa on se, että nämä kentät olivat julkisesti saatavilla. Kaikki kentät eivät välttämättä ole julkisia. Ne usein ovat rajoitetut näkymään vain profiilissa määritellyille kontakteille. Tästä syystä merkittävää osaa sosiaalisista verkkojalanjäljistä ei voida jälleenrakentaa tavanomaisilla, laillisilla tiedonlouhintamenetelmillä, jota tutkimuksessa käytettiin.



Kuva 2.1: Sosiaalisista verkkosovelluksista löytyvä digitaalinen jalanjälki.

Oletetaan, että rakennetaan kohteen digitaalista jalanjälkeä käyttämällä pelkästään hakukoneita ja muita laillisia työkaluja. Tietämällä käyttäjän pseudonyymi, nimi jolla käyttäjä yksilöi itsensä internetissä, tiedonlouhintaa voidaan tehostaa merkittävästi [29]. Tällöin parhaimmassa tapauksessa, hyökkääjän kannalta, voidaan 40 % kohteen sosiaalisesta verkkojalanjäljestä rekonstruoida. Jos tiedetään käyttäjän nimi, pystytään 10 - 35 % sosiaalisesta verkkojalanjäljestä rekonstruoimaan. Nimi tai pseudonyymi monesti on pääteltävissä sähköpostiosoitteista. Tiedonlouhimisprosessit tulevat kehittymään, jolloin yhä suurempi prosentuaalinen osuus kohteen sosiaalisesta verkkojalanjäljestä pystytään rekonstruoimaan.

Tutkimus havainnollistaa hyvin, kuinka helposti digitaalinen jalanjälki pystytään jälleenrakentamaan. Ilman merkittäviä tietoteknisiä taitoja tavallisilla hakukoneilla voidaan kerätä paljon tietoa kohteesta, olettaen että käytössä on nimi tai pseudonyymi. Määrätietoinen ja osaava verkkorikollinen pystyy kirjoittamaan tiedonlouhintaan erikoistuneita skriptejä, joilla suurien tietomäärien koostaminen voidaan automatisoida ja optimoida.

2.3 Verkkoteoria apuna sosiaalisen verkoston analysoinnissa

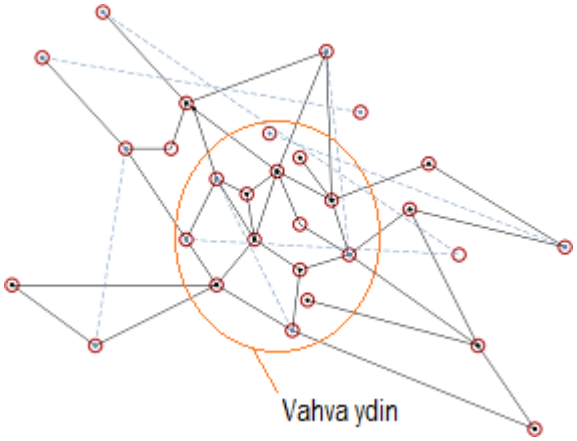
Tietoturvan haasteita tähän mennessä on käsitelty pääasiallisesti käyttäjän näkökulmasta ja erityisesti digitaalista jalanjälkeä koskien. Hyökkääjä ei aina kuitenkaan tavoittele käyttäjien tietoja tai tiettyä käyttäjää. Kohteena voi olla sosiaalinen verkkosovellus yleisesti, ja tällöin tavoitteena voi olla mahdollisimman usean käyttäjän saastuttaminen haitakkeilla tai roskapostin levittäminen. Verkkoteorian tuntemus antaa ymmärrystä näihin hyökkäyksiin sekä myöhemmin esiteltävään luottamuksen ongelmaan.

Verkkoteoriaa [30] on sovellettu monilla eri tieteenaloilla, mm. sosiaalisiin verkostoihin, tietokoneverkkoihin ja hermoverkkoihin. Verkkoteoria soveltuu hyvin myös kuvaamaan sosiaalisia verkkoyhteisöjä. Ymmärrys siitä millaisiksi yhteisöt muodostuvat rakenteeltaan auttaa ymmärtämään näissä sovelluksissa tapahtuvia ilmiöitä. Tietoturvan kannalta tärkeää on tietää mihin tietoturvariski kohdistuu ja kuinka informaatio leviää tällaisissa verkoissa. Koska kysymys on digitaalisesta ympäristöstä, informaatiolla voidaan tarkoittaa haitakkeita, roskapostia ja muuta haitallista materiaalia. Digitaalisen informaation kopioitavuus oleellisesti jouduttaa leviämistä. Oleellista on siis ymmärtää haitallisen materiaalin liikennettä, sen ottamaa muotoa ja käyttäytymistä. Tämä voi tarjota uutta tietoa parempien puolustusmekanismien kehittämiseen, hyökkäysskenaarioiden ennakoimiseen ja ennaltaehkäisevien toimien suunnitteluun.

Kuvitellaan, että käyttäjät muodostavat verkon. Verkon solmuina toimivat käyttäjät. Solmuja yhdistävät viivat ovat relaatioita, solmujen suhteita. Yksinkertaistetaan suhteet yhdenlaisiksi: tuttavuussuhteiksi. Tuttavuussuhteella tarkoitetaan nyt sosiaalisten verkkosovellusten ystävä-, seuraaja- tai tuttavuustoimintoja. Tuttavuussuhteet voivat olla yksi- tai kaksisuuntaisia.

Keskussolmut ovat erityisessä asemassa olevia solmuja. Keskussolmu on solmu, jolla on hyvin paljon suhteita muihin solmuihin. Joissakin verkoissa keskussolmu voi olla merkittävä komponentti verkon rakenteelle, koska se yhdistää monet muut solmut. Keskussolmut ovat merkittäviä komponentteja informaation nopealle leviämiselle verkoissa.

Tutkimus osoittaa [31], että sosiaalisissa verkkosovelluksissa on suuri lukumäärä vahvasti kytkeytyneitä ryhmiä, jotka koostuvat solmuista eli käyttäjistä. Suhteiden lukumäärä on jakaantunut melko tasaisesti solmujen kesken. Solmuilla on keskimäärin paljon suhteita toisiinsa, ts. keskussolmuja on paljon. Verkon ydin koostuu näistä vahvasti kytkeytyneistä rykelmistä. Verkon reunoilla on solmut, joilla ei ole paljon suhteita muihin solmuihin. Näitä solmuja, joilla ei ole paljon suhteita, ei



Kuva 2.2: Yksinkertainen verkosto.

pidetä niin luotettavina [31]. Päästäkseen ytimeen solmun tulee hankkia lisää suhteita. Hankittuaan tarpeeksi suhteita siitä on tullut keskussolmu ja se on sijoittunut lähelle ydintä verkostossa. Keskussolmut, joilla on suuri lukumäärä suhteita, ovat tärkeitä verkon liitettävyydelle. Vahvan ytimen ja keskussolmujen suuren lukumäärän takia polkujen pituudet summittaisesta solmusta toiseen solmuun ovat lyhyitä. Koska polkujen pituudet ovat lyhyitä, informaatio leviää hyvin nopeasti verkostossa. Tällaista verkkoa kutsutaan **pienen maailman verkostoksi** (engl. *Small world network*).

Algoritmit, jotka pyrkivät löytämään vaikutusvaltaisimmat solmut voivat palvella tietoturvan toteutuksessa, niin kuin hyökkäysten toteuttamisessa. Hyökkääjä, joka haluaa maksimoida haitakkeen tai roskapostin leviämisen, haluaa ottaa kohteekseen keskussolmun. Toisin sanoen, hyökkäyksen tehokkuuden maksimoimiseksi hyökkäys kannattaa kohdistaa käyttäjiin, joilla on mahdollisimman paljon suhteita muihin käyttäjiin. Pienen maailman verkostoissa keskussolmuja on suhteellisen paljon. Haitakkeen nopealle leviämiselle riittää pääsy vahvasti kytkeytyneeseen ytimeen. Kyseenalaista on, onko haitakkeiden tarpeen optimoida leviämisstrategiaansa koskemaan kaikkein vaikutusvaltaisimpia keskussolmuja, jos verkoston ydin on jo vahvasti kytkeytynyt. Sosiaalisten verkkosovellusten rakenne on ihanteellinen haittaohjelmien levitykseen pienen maailman verkoston ominaisuuksien takia: lyhyt etäisyys mielivaltaisesta solmusta mihin tahansa toiseen solmuun ja suuri keskussolmujen lukumäärä.

Pienen maailman verkosto on suotuisa muillekin tietoturvariskeille kuin haitakkeille. Pienen maailman verkoston ominaisuuksien myötä informaatio kulkee nopeasti verkossa. Tilapäivitykset, viestit ja syndikoitu informaatio voivat tarjota haitallisen linkin, joka vie yhteisöpalvelun ulkopuolelle. Vaikka haitallinen linkki huo-

mattaisiin nopeasti, ehtii se vaarantamaan usean solmun tietoturvan informaation nopean propagaation vuoksi. [21]

Kun pienen maailman verkosto on tarpeeksi suuri, todennäköisyys joutua yksittäisen hyökkäyksen kohteeksi on erittäin pieni. *"Security by obscurity"* [32] on tietoturvakäsite, joka hyödyntää tätä tietoa. Periaatteena on tehdä kohteista vaikeasti yksilöitäviä tai havaittavia, sosiaalisten verkkosovellusten kontekstissa haetaan suojaa verkoston suuresta koosta. Yhteen solmuun kohdistettuja hyökkäyksiä vastaan verkoston suuri koko tarjoaa hyvän suojan kun taas nopeasti leviäviä haitakkeita vastaan tämä menetelmä ei tuo tarpeeksi vahvaa suojaa. Määrätietoista, kohteen jollain tapaa tuntevaa, hyökkääjää vastaan menetelmä ei auta. Kohdennetut hyökkäykset, jossa hyökkääjällä on kohteesta jotain tietoa, menetelmä on voimaton. Kohteen nimen tietämällä on vaivatonta etsiä kohdetta edustava profiili. Useimmat sosiaaliset verkkosovellukset tarjoavat tehokkaita työkaluja käyttäjien hakemiseksi. Hyökkääjä voi löytää kohteen tietämättä tämän nimeä. Digitaalista jalanjälkeä rekonstruoimalla (ks. kpl 2.2) on mahdollista löytää kohde muita tietoja hyväksikäyttämällä.

2.4 Inhimillisen tekijän merkitys uhkatilanteissa

Toteutuakseen tietoturvauhkat usein tarvitsevat jonkinlaisia toimia käyttäjältä. Kyseessä voi olla sähköpostiviestissä olevan linkin klikkaaminen, käyttäjän vahvistusta vaativan painikkeen painaminen, luottamuksellisten tietojen antaminen vieraalle taholle tai tiedoston lataaminen. Hyökkäyksen onnistumisen kannalta monesti on helpompaa huijata tai harhauttaa ihmistä, kuin yrittää murtaa ohjelmistoa. Hyökkäykset jotka pohjautuvat käyttäjien manipulointiin, sosiotekniset manipulointitavat, ovat yleistyneet viime vuosikymmenen alusta [28]. Tietynlainen käyttäytyminen voi luoda suosiollisen ympäristön tietoturvauhkille. Alentunut valveutuneisuuden taso ja avoimuus altistavat käyttäjän sosioteknisiin manipulointitapoihin. Oleellista on selvittää onko sosiaalisten verkkosovellusten sosiaalisessa aspektissa jotain joka lisää tietoturvauhkien riskiä.

2.4.1 Yksityisyys ja internet käyttäytymisen muutos

Onko ihmisten internet käyttäytyminen on muuttunut vuosien saatossa, ja jos on, niin kuinka? Seuraava lainaus on *The New Yorker*-sanomalehden sarjakuvasta vuodelta 1993:*"On the internet, nobody knows you're a dog"* [33]. Sanonnalla kuvattiin silloisen internetin luonnetta. Silloisessa internetissä toimittiin anonyymina. Yksityisyys ei ollut uhattuna.

Tilanne on muuttunut noin viidessätoista vuodessa päinvastaiseksi. Yhä etenevissä määrin ihmisten sosiaalinen kanssakäyminen on siirtymässä internetiin ja internetissä esiinnytään yhä useammin omana itsenä. Facebookin perustajan, Mark Zuckerbergin, mukaan sosiaalisten medioiden suosion vuoksi ihmisten sosiaaliset normit ovat muuttuneet [34]. Väittämän mukaan ihmisten yksityisyyden käsite on muuttunut avoimempaan suuntaan.

Asia ei välttämättä ole näin yksinkertainen. Luonnollisesti Zuckerbergin väite voi olla puolueellinen. Ihmisten avoimuus henkilökohtaisten asioidensa suhteen lisää kommunikaatiota ja tässä tapauksessa Facebookin käyttöä. Jotta voitaisiin tietää onko yksityisyyden käsite muuttunut, tulee tietää mitä yksityisyys on.

Yksityisyyden käsitettä on vaikea määrittää, sillä kyseessä on subjektiivinen sekä kulttuurisidonnainen termi. Mikä on toiselle yksityistä, ei välttämättä ole sitä toiselle. Yksityisyys on omistajalleen jotakin henkilökohtaista ja arkaa. Hyväksytään tämä avara käsite. Jos yksityisyys olisi muuttunut avoimempaan suuntaan, ihmiset paljastaisivat nyt asioita, joita he eivät olisi ennen paljastaneet. Sellaisia asioita, jotka ennen olivat kuuluneet yksityisyyden piiriin.

Zuckerbergin väitteestä kaikki eivät ole samaa mieltä [35]. Tutkimukset osoittavat, että sosiaalisten verkkosovellusten käyttäjät ovat huolestuneita yksityisyydestään [36] [37]. Tästä huolimatta sosiaaliset verkkosovellukset ovat media, joissa ihmiset kaikkein auliimmin kertovat yksityisistä asioistaan. Tätä ilmiötä kutsutaan yksityisyyden paradoksiksi [37]. Paradoksi ilmiö on sen vuoksi, koska takeita luottamuksesta on vähiten. Toista osapuolta ei ole nähty tai kuultu. On helpompaa esittää olevansa eri henkilö virtuaalimaailmassa kuin reaalimaailmassa.

Käyttäjät eivät ole tietoisia kaikista riskeistä, joille he tulevat alttiiksi, kun heitä koskeva arkaluontoinen sisältö julkistetaan. Tämä luo pohjaa varomattomalle käyttäytymiselle. Schrammelin ym. tutkimuksessa käyttäjän persoonallisuustyypillä ei ollut merkittävää vaikutusta siihen, kuinka paljon hän julkaisee tietoa itsestään sosiaalisissa verkkosovelluksissa. [36]

Yksityisyyden paradoksi antaa ymmärtää, että yksityisyyden käsite olisi muuttunut. Jos kerran ihmiset paljastavat verkkoyhteisöissä tavallista enemmän arkaluontoisia tietoja, voidaanko tästä päätellä että yksityisyyden käsite on muuttunut? Kenties yksityisyyden käsite ei ole muuttunut, vaan esiintyneet ilmiöt kuvastavat ihmisen käyttäytymistä tietyssä, vielä uudessa toimintaympäristössä. Sosiaaliset verkkosovellukset ovat niin uusi ilmiö, että on vaikea sanoa, onko ihmisten yksityisyyden käsite vai internet käyttäytyminen muuttunut. Vain joitain vuosia sitten vastaavia järjestelmiä ei ollut olemassa.

Pohjimmiltaan tietoturvallisuustekniikan näkökulmasta oleellista ei ole onko yksityisyyden käsite muuttunut tai miksi käyttäytyminen on muuttunut. Oleellista on se, minkälaista tämä käyttäytyminen on ja erityisesti mitkä ovat tämän käyttäytymisen seuraamukset tietoturvallisuudelle.

Jokaisella kommunikoimisen muodolla on omia erityispiirteitä. Mihin tarkoitukseen mediaa käytetään, minkälaista roolia teknologia esittää ja mitä ajatuksia mediaan liitetään, määrittelevät tapamme käyttää kyseistä mediaa. Avoimempaan käyttäytymiseen vaikuttaa sosiaalisten verkkosovellusten olennaisin piirre: sosiaalisuus. Yhdessä tekeminen, suhteiden ylläpitäminen ja kommunikaatio vaativat avointa asennetta. Ihmiset ovat huolettomampia ja tavallista valmiimpia vuorovaikutukseen. Yksityisyyden paradoksin toteutumista edistää informaation jakamisen helppous. Kanssakäyminen verkossa ei edellytä toisen ihmisen kohtaamista samalla tavoin kuin esimerkiksi puhelimessa tai kahvilassa. Kommunikaatiosta jää pois eleet, katsekontakti, äänensävy ja muut vihjeet. Tapahtumasta jää pois myös muihin sosiaalisiin tapahtumiin liittyvät normit ja jännitteet. Kynnys kanssakäymiseen alenee. [13]

2.4.2 Vahva liitännäisyys

Sosiaalisissa verkkosovelluksissa käyttäjien väliset suhteet muodostavat vahvan liitännäisyyden. Muodostettavat suhteet ovat merkitykseltään positiivisia tai vähintään neutraaleja, siinä kontekstissa että suhteessa nähdään, edes potentiaalista, hyötyarvoa. Nähty hyötyarvo voi olla hyväksi käyttävää luonnetta: toiselle osapuolelle haitallinen suhde, toiselle osapuolelle hyödyllinen suhde. Suhteet jotka muodostuvat näissä yhteisöissä käyttäjien välille ovat usein nk. heikkoja siteitä [12]. Heikolla siteellä tarkoitetaan etäisempää ihmissuhdetta, jolla ei ole jokapäiväistä tai huomattavaa roolia asianomaisen elämässä. Tämä ei kuitenkaan tarkoita sitä, etteivätkö heikot siteet ole tärkeitä. Heikot siteet toimivat tärkeänä sosiaalisena pääomana omistajalleen. Niiden tehtävä ihmisen sosiaalisessa elämässä on erilainen kuin vahvojen siteiden [38]. Heikkojen siteiden syntymisen taustalla on usein yksi yhteinen tekijä. Yhteisenä tekijänä henkilöiden välillä voi olla sama työpaikka, tiettyyn ystäväpiiriin kuuluminen, yhteinen harrastus, kaukainen sukulaisuussuhde, jne.. Monesti henkilöitä yhdistävä tekijä on yhteinen ystävä tai tuttava. Heikkojen siteiden piiriin eivät kuulu perhe, läheiset ystävät tai usein tavatut kontaktit. Heikoilla siteillä on merkittävä rooli näiden yhteisöjen vahvassa liitännäisyydessä.

Monet muutkin tekijät lisäävät vahvaa liitännäisyyttä. Käyttäjä voi kokea suhteiden suuren lukumäärän osoituksena korkeasta yhteiskunnallisesta asemasta tai

pidettävyydestä. Jotkut verkkoyhteisöt kannustavat suhteiden lisäämiseen palkitsemalla käyttäjiä, jotka kartuttavat suhdelukumääriään [39]. Yhteisöjen vahvaa liitännäisyyttä edistävät sosiaaliset normit, jotka ovat jääneet tähän kommunikaatiomediaan. Lähes poikkeuksetta suhteen muodostuminen vaatii kummaltakin osapuolelta vahvistuksen. Tällaisen pyynnön hylkääminen koetaan monesti loukkauksena, eräänlaisena epäluottamuksen osoituksena [12].

Heikkojen siteiden suuri lukumäärä, uusien suhteiden luominen ja yksityisyyden paradoksi muodostavat luottamuksen ongelman.

2.5 Luottamuksen ongelma

Luottamus on tärkeä tekijä kaikessa sosiaalisessa vuorovaikutuksessa. Kanssakäymisen kannalta tärkeää on, voidaanko luottaa toiseen osapuoleen. Edellisessä kappaleessa esitetyt huomiot luovat tilanteen, jossa ihmisen täytyy luoda luottamus toiseen osapuoleen vähillä vihjeillä ja takeilla.

Luottamuksen ongelman ymmärtämiseksi on olennaista selvittää, kuinka luottamussuhteet muodostuvat käyttäjien välille sosiaalisissa verkkosovelluksissa ja pitävätkö käyttäjät sosiaalisissa verkkosovelluksissa muodostettuja luottamussuhteita yhtä vahvoina kuin reaalimaailmassa olevia luottamussuhteita. Jos käyttäjä luottaa sosiaalisissa verkkosovelluksissa solmittuihin suhteisiin yhtä vahvasti kuin reaalimaailman suhteisiin, käyttäjä on alttiimpi sosioteknisille manipulontitavoille.

Koska sosiaalinen verkkosovellus luo kehyksen sille, kuinka luottamus toteutetaan teknisellä tasolla, ei luottamuksen ongelma ole riippuvainen vain ihmisten välisistä relaatioista vaan myös teknologiasta ja sovituista menettelyistä. Tästä johtuen tarjotut metodit käsitellä luottamusta eivät välttämättä ole tarpeeksi hienovaraisia kaikkien käyttäjien tarpeisiin. Huonosti toteutetut tietosuojamenettelyt tai tietosuojamekanismit tekevät käyttäjän alttiiksi hyökkäyksille kuten liiallinen luottamus toiseen osapuoleen.

Käyttäjät eivät luota yhtä paljon verkkoyhteisöissä tavattuihin henkilöihin, mutta kun he luottavat, joutuvat he vaaralliseen asemaan, sillä yksityisyyden paradoksin mukaisesti käyttäjät julkaisevat enemmän tietoa itsestään kuin muissa medioissa tai tilanteissa [36] [37]. Korrelaatiota esiintyy käyttäjän julkaiseman tiedon määrällä ja hänen suhteidensa lukumäärällä. Lisäksi, käyttäjään, jolla on paljon suhteita, luotetaan tavallisesti [36].

Koska sosiaalisissa verkkosovelluksissa kynnys sosialisoida alenee, käyttäjä luultavasti muodostaa suhteita moniin hänen heikkoihin siteisiin ja tuntemattomiin ihmisiin. Joskus käyttäjät ovat valmiita luomaan suhteen täysin tuntemattomien osa-

puolien kanssa [40]. Tämä vahvistaa käsitystä, että jotkut käyttäjät kilpailevat heikkojen siteiden lukumäärillä ja, että käyttäjät eivät ole tietoisia kaikista tietoturvariskeistä. Käyttäjät luottavat henkilöihin, jotka he ovat tavanneet reaalimaailmassa tai tietävät jotain muuta kautta [39]. Tällaisiin henkilöihin käyttäjät ovat usein valmiita muodostamaan suhteen sosiaalisissa verkkosovelluksissa.

Käyttäjän takeet siitä, että henkilö kenen kanssa hän kommunikoi, on se, kuka hän näyttää olevan, lepäävät sosiaalisen verkkosovelluksen tietoturvallisuudessa. Oletetaan, että tietoturvallisuus vaarantuu. Ilman audiovisuaalisia vihjeitä, sosiaalisten verkkosovellusten luomassa kontekstissa, henkilön imitoimisen onnistuminen riippuu pitkälti siitä, kuinka hyvin hyökkääjä pystyy matkimaan imitoitavan keskustelutyyliä. Jos hyökkääjä pystyy esittämään käyttäjän luottamaa henkilöä vakuuttavasti, käyttäjä luultavasti lankeaa sosiotekniseen manipulointiin.

Luottamuksen ongelmalle olennaista on ainutlaatuisen, tarpeeksi vahvan, yksilöivän tunnisteen puuttuminen sosiaalisista verkkosovelluksista. Sosiaalissa verkkosovelluksissa käyttäjä tunnistetaan profiilin avulla. Käyttäjää pyydetään profiilia luotaessa ilmoittamaan joitain henkilötietoja, sekä usein sähköpostiosoite. Käyttäjän identiteetti sidotaan profiiliin sähköpostiosoitteen, tai muun julkisen tiedon, avulla. Täten kuka tahansa pystyy esiintymään toisen henkilön nimissä. Hyökkääjä ei välttämättä pysty ylläpitämään illuusiota tekaistusta identiteetistään kovin kauan. Onnistuneen hyökkäyksen toteuttamiseksi tämä ei yleensä ole tarpeellista.

Sørensen [41] käsittelee luottamuksen ongelmaa sosiaalisten verkkosovellusten kontekstissa tekemällä seuraavat määritykset ja huomiot:

- **Yksityisyys** (engl. *Privacy*) kuvaa sitä, kuinka hyvin käyttäjä voi suojella ja hallinnoida hänen arkaluontoista sisältöä kaikessa kanssakäymisessä, joka tapahtuu sosiaalisessa verkkosovelluksessa.
- **Turvallisuus** (engl. *Security*) kuvaa sitä, mitä turvallisuusmekanismeja sosiaaliseen verkkosovellukseen on toteutettu käyttäjän arkaluontoisen tiedon suojelemiseksi.
- **Varmuus** (engl. *Reliability*) kuvaa käyttäjän luottamusta paikoillaan olevien tietosuojamenettelyjen ja turvallisuusmekanismien kykyyn suoriutua tehtävästään.
- Kuinka hyvin käyttäjä pystyy suojelemaan yksityisyyttään suhteessa palveluntarjoajiin (sosiaalisen verkkosovelluksen kehittäjät ja kolmannet osapuo-

let) ja suhteessa niihin osajoukkoihin, joihin hän on kytkeytynyt sosiaalisen verkostonsa kautta.

- Tarve selkeälle luottamuksen vahvistavalle toiminnallisuudelle, jonka käyttäjä voi asettaa päälle tai pois.
- Tarve selkeälle menettelylle ja ohjeistukselle kahden osapuolen muodostaessa luottamussuhteen.

Luottamuksen ongelma ei rajoitu vain käyttäjien välille. Sørensen tuo esille [41] palveluntarjoajan ja kolmansien osapuolien roolit käyttäjien arkaluontoisten tietojen käsittelijöinä. Palveluntarjoajan tulee valvoa ja hallita, että kolmannen osapuolen kehittäjät käsittelevät käyttäjien tietoja asianmukaisilla tavoilla. Ilman ulkopuolista, neutraalia tahoa on kyseenalaista, kuinka tämä voidaan taata käyttäjille. Käyttäjien on hyvä olla tietoisia käyttöehtosopimuksen ehdoista ja mikä vastuu palveluntarjoajalla on käyttäjiä kohtaan. Tulkitsemista vaikeuttaa sosiaalisten verkkosovellusten kansainvälinen luonne ja valtioiden eriävät lait tietosuojamenettelyn suhteen.

Koska käyttäjillä on valta päättää keneen luottavat, sosiaalisen verkkosovelluksen määräämässä kehyksessä, tulee käyttäjien ymmärtää kuinka luottamusta käsitellään sosiaalisessa verkkosovelluksessa. Erinomaisesti suunniteltu sosiaalinen verkkosovellus minimoi käyttäjien virheistä ja väärinymmärryksistä johtuvat vahingot. Kaikkia käyttäjien toiminnoista johtuvia tietoturvariskejä ei voi hallita. Käyttäjien on hyvä olla valveutunut, tällä tavoin hän pystyy välttämään monia tietoturvariskejä. Tästä lisää kappaleessa 4.3.

Sørensen [41] esittää kuinka käyttäjät luottavat paljon sosiaaliseen verkkosovelluksen kykyyn suojella heitä. Vähäisten luottamuksen takeiden takia voidaan luottaa allaolevaan tekniikkaan enemmän sillä perusteella, että tällaisia järjestelmiä ei olisi olemassa, jos ne eivät olisi luotettavia. Liiallinen luottamus järjestelmän kykyyn suojella käyttäjää voi tosin olla vahingollista, jos se laskee käyttäjän valveutuneisuuden tasoa.

3 Tietoturva sosiaalisissa verkkosovelluksissa

Luku etenee seuraavasti: ensin tutustutaan siihen, mitä tietoturva on. Sitten käydään läpi niitä käsitteitä, joita tässä työssä käytetään. Kappaleessa 4.3 pohditaan sosiaalisen median ilmiöitä ja piirteitä, siinä määrin kun ne koskevat tietoturvaa. Tämän jälkeen esitellään kuinka sosiaaliset verkkosovellukset mallinnetaan. Kappaleessa 4.5 kuvataan hyökkääjää viiden eri ominaisuuden kautta. Lopulta pohditaan kuinka tietoturva toteutetaan, millä perusteilla, kuinka huomioida tietoturva sovelluskehityksessä sekä käydään läpi joitain hyviä käytäntöjä.

3.1 Tietoturvallisuustekniikan tavoitteet

Tietoturvallisuustekniikka on yksi turvallisuustekniikan osa-alue [42]. Turvallisuustekniikka on hyvin laaja ja poikkitieteellinen tieteenala. Tätä laajuutta selittää turvallisuustekniikan suuri kattavuus. Turvallisuustekniikka suojelee koko organisaatiota, ulkopuolisilta tai sisältäpäin tulevilta uhkilta. Organisaatio käsittää laitteistot, toimitilat, työympäristön, ihmiset, aineettoman omaisuuden ynnä muut organisaation resurssit. Turvallisuustekniikkaan sisältyy osaamista muun muassa valvontatekniikasta, oikeustieteestä, yritystieteestä, psykologiasta ja ohjelmistotekniikasta.

Tietoturvallisuustekniikka keskittyy turvallisuustekniikan tietotekniseen osaalueeseen. Tietoturvallisuustekniikan tarkoitus on suojella tietojärjestelmän tarjoamia palveluita ja informaatiota. Vaikka tietoturvallisuustekniikka on tieteenalana nuori, se on saanut osakseen paljon huomiota niin tiedeyhteisöjen kuin teollisuuden taholta. Tietoturvallisuustekniikan alan ongelmaksi on muodostunut käsiteltävien aiheiden suuri määrä ja monimutkaisuus [43]. Epäselvää on ollut, kuinka turvata tietojärjestelmille ominainen immateriaalinen data. On tärkeää huomioida, että tietoturvallisuustekniikassa ei ole kyse vain teknologiasta, vaan myös ihmisistä ja prosesseista [43]. Tietoturvallisuustekniikka pyrkii tavoitteeseensa takaamalla tietojärjestelmän tietoturva.

Tietoturvan merkitys on korostunut ohjelmistotekniikassa kuluneen vuosikymmenen aikana [44]. Tietojärjestelmien monimutkaistuessa ja niiden arvon kasvaessa on tullut yhä ilmeisemmäksi, että niitä tulee suojella väärinkäytöksiltä. Kasvanut tietoverkostoituminen lisää osaltaan väärinkäytösten riskiä, kun tietojärjestelmät ovat yhteydessä ulkopuoliseen verkkoon yhä useamman päätepisteen kautta.

Hyökkäykset ja vikaantumiset aiheuttavat merkittävien taloudellisten tappioiden lisäksi organisaation maineelle vahinkoa.

Tietoturvan määrittäminen on ollut ongelmallista tiedeyhteisölle [45]. Yasar ym. [44] määrittelevät tietoturvan tietojärjestelmän ominaisuutena, Canal [45] määrittelee tietoturvan negatiivisen logiikan mukaan vikojen ja väärinkäytösten poissaolona. Information Security Management Maturity Model (ISM3) [46] määrittelee tietoturva seuraavankaltaisesti: *"Tietoturva on tulos siitä, että saavutetaan joukko tavoitteita, jotka takaavat, että organisaation liiketoimintapäämäärät saavutetaan"*. Edellinen määritelmä tuo esille organisaation tavoitteet ja tietoturvatekniikan kytköksen suojeltavaan organisaatioon. Saman liiketoiminnan jatkuvuuden ja toimivuuden takaamisen roolin tietoturvalle esittää ISO 17799-standardi [46], joka määrittelee tietoturvan seuraavasti:
"The process of protecting information from a wide range of threats in order to ensure business continuity, minimize business damage and maximize return on investment by preserving **confidentiality**, **integrity** *and* **availability** *of information."*

Tavallisesti tietoturva määritellään CIA-mallin [45] kautta, johon myös ISO-standardi viittaa. CIA-malli antaa kolme tietojärjestelmän käsitettä, jotka tulee säilyttää, jotta tietojärjestelmä olisi tietoturvallinen.

- **Luottamuksellisuus** (engl. *Confidentiality*) takaa sen, että data on saatavilla vain niille tahoille, jotka ovat valtuutettuja kyseiseen dataan. Luottamuksellisuuteen kuuluu olennaisesti tiedon suojeleminen ei-asianomaisilta toimijoilta.

- **Eheys** (engl. *Integrity*) määrittää kuinka tietojärjestelmän dataa käytetään. Tietojärjestelmää tulee käyttää vain tarkoituksenmukaisia kanavia ja metodeja käyttäen. Tarkoitus on estää datan väärinkäyttö varmistamalla, että dataa käyttävät prosessit toimivat turvallisesti ja ennalta määrätyillä tavoilla. Eheys tarkoittaa myös sitä, että data on oikeassa, sille tarkoitetussa esitysmuodossa.

- **Saatavuus** (engl. *Availability*) varmistaa, että data ja palvelut ovat käytettävissä ja saavutettavissa tarkoituksenmukaisina aikoina. Tämä tarkoittaa sitä, että tietojärjestelmää voidaan käyttää tarvittaessa, poislukien ajanjaksot jolloin tietojärjestelmä on pois käytöstä, esim. silloin kun tietojärjestelmälle suoritetaan huolto- ja ylläpitotoimenpiteitä.

Toinen tavallisesti käytetty malli, **tietoturvallisuuden viisi pilaria** (engl. *Five Pillars of Information Assurance*) [47], lisää CIA-malliin kaksi ominaisuutta lisää:

- **Aitous** (engl. *Authenticity*) tarkoittaa sitä, että käyttäjän, laitteen tai prosessin identiteetti varmistetaan. Kyseisellä oliolla tulee olla se identiteetti, minkä se väittää itsellään olevan. Aitous voi myös tarkoittaa sitä, että varmistetaan datan lähde ja eheys, varmistuen näin datan aitoudesta ja oikeellisuudesta. Prosessia, joka varmistaa aitouden, kutsutaan autentikoinniksi.

- **Kiistämättömyys** (engl. *Non-repudiation*) estää jonkin tapahtuman eri osapuolia kieltämästä olleensa osallisia tapahtumassa. Kiistämättömyys tuo tietojärjestelmään jäljitettävyyttä ja luotettavuutta tuomalla tietojärjestelmään kyvyn varmistua siitä, onko jokin toimija tehnyt tietyn toimenpiteen.

Tietoturva on subjektiivinen tietojärjestelmän suhteen. Kuten ISO 17799-standardi esittää, tietoturvan tulee suojella organisaation liiketoiminnan jatkuvuutta takaamalla tietojärjestelmän oikeanlainen toiminta. Täten on tarpeellista selvittää, mitkä toiminnallisuudet ovat oleellisia tietojärjestelmän toiminnalle, jotta tietoturva voidaan toteuttaa tietojärjestelmään [44]. Tietoturvallisuustekniikka eroaa siten turvallisuustekniikasta, että ei voida soveltaa yhtä tehokkaasti hyväksi havaittuja toimenpiteitä, esimerkiksi lukkoja estämään asiattomien pääsy tiettyihin tiloihin. Tietoturva ei ole joukko toimenpiteitä, jotka suoritettua voidaan taata tietojärjestelmän olevan tietoturvallinen, vaan tietoturvan toteuttaminen on jatkuva-aikainen prosessi [44].

Vaikka jokainen tietojärjestelmä eroaa hieman muista tietojärjestelmistä, niihin pätevät tietyt säännönalaisuudet ja samankaltaisuudet. Tämän johdosta on pystytty hyödyntämään **tietoturvamalleja** (engl. *security patterns*) [48], jotka *"ovat hyvin todistettuja valmiita malleja usein toistuviin tietoturvaongelmiin."*

Todellisuudessa tietoturvallinen tietojärjestelmä ei ole immuuni hyökkäyksille, ts. tietoturvallisuus ei voi taata väärinkäytösten absoluuttista poissaoloa. Syy tähän on resurssien rajallisuus. Tietoturvan toteuttaminen järjestelmään ei ole halpaa [44]. Tietoturvan subjektiivisesta kytköksestä tietojärjestelmään johtuen on joskus vaikeaa arvioida tarvittavia toimenpiteitä tietoturvan toteuttamiseksi, joka vaikeuttaa entisestään täysin tietoturvallisen järjestelmän luomista. Tästä syystä useimmat lähestymistavat näkevät tietoturvan toteuttamisen riskinhallintana [43] [44].

3.2 Tietoturvallisuustekniikan peruskäsitteet

Tietoturvallisuustekniikan tieteenalalla on ollut vaikeuksia luoda yhtenäistä ja johdonmukaista kehystä tutkimukselle [42]. Tästä huolimatta on olemassa muutamia peruskäsitteitä, jotka ovat vakiintuneet yleiseen käyttöön. Tässä esitellään termit **etu**

(engl. *asset*), **uhka** (engl. *threat*) ja **haavoittuvuus** (engl. *vulnerability*). Nämä kolme termiä liittyvät läheisesti toisiinsa. Tietoturvallisuustekniikan käsitteitä, jotka liittyvät tietoturvallisuuden ja riskinhallintaan, esitellään kappaleessa 3.6.

3.2.1 Etu

Tietoturvallisuustekniikassa termi etu [48] [49] tarkoittaa resurssia tai informaatiota, jota tulee suojella. Etu on organisaatiolle arvokas, eri edut ovat arvokkaampia kuin toiset edut. Edun käsite on laaja ja sisältää, organisaatiosta riippuen, eri asioita. Edulla tavallisesti tarkoitetaan laitteita (kovalevyt, muistitikut, ...), ihmisiä (henkilöstö, käyttäjät, ...), ohjelmistoja (käyttöjärjestelmät, asiakasohjelmistot, ...), palveluita (sähköpostipalvelu, pankkipalvelut, ...) ja informaatiota (käyttäjien profiilitiedot, sosiaaliturvatunnukset, ...). Edulla voidaan tarkoittaa abstraktimpaa asiaa kuten organisaation mainetta tai osaamista [47]. Suojeltavat edut vaihtelevat organisaation ja tämän tarpeiden mukaan.

Edut vastaavat kysymykseen:"Mitä tulee suojata?"

3.2.2 Uhka

Uhka on tapahtuma, joka on mahdollisesti haitallinen tietojärjestelmän eduille. Jokainen tietojärjestelmän etu on vaarassa yhdelle tai useammalle uhkalle. Riippuen tietojärjestelmästä, jotkut uhkat ovat relevantimpia tietojärjestelmälle kuin toiset. Uhkia ovat esimerkiksi [49]:

- Luvaton pääsy etuihin
- Luottamukselliseksi määritetyn tiedon julkistaminen
- Palvelunesto
- Etuihin kohdistuva varkaus
- Etuihin kohdistuva turmeltuminen
- Haitakkeiden leviäminen tietojärjestelmään
- Fyysinen vaurio

Uhkat vastaavat kysymykseen:"Miltä etuja tulee suojata?"

3.2.3 Haavoittuvuus

Haavoittuvuus mahdollistaa uhkan toteutumisen [48]. Haavoittuvuuden voi muodostaa tietojärjestelmän suunnittelu- tai toteutusvirhe, laitteet, henkilöstö, ohjeet, toimintasuunnitelmat tai muut proseduurit. Haavoittuvuudet kuvaavat konkreettisella ja teknisellä tasolla, kuinka uhka voi ilmentyä. Sama uhka voi toteutua useammalla eri tavalla. Kuten edut ja uhkat, haavoittuvuuksien merkitykset organisaatiolle vaihtelevat. Tämä pitää paikkansa, vaikka kaksi haavoittuvutta koskevat samaa uhkaa. Tällöin toinen haavoittuvuus muodostaa suuremman haittavaikutuksen organisaatiolle. On mahdollista, että tietty uhka tarvitsee enemmän kuin yhden haavoittuvuuden, jotta se voisi toteutua.

Esimerkkejä haavoittuvuuksista ovat lukitsematon toimitila, julkiseen tilaan jätetty luottamuksellisia tietoja sisältävä muistitikku, laite joka ei toimi standardien tai määräysten mukaisesti, huolimattomasti alustettu palomuuri, verkkoprotokolla joka voi käyttää rajattomasti verkkolaitteen resursseja ja ohjelmistovirhe, joka väärinkäytettynä tarjoaa pääsyn sisään tietojärjestelmään.

Nollapäivähaavoittuvuus (engl. *zero day vulnerability*) tarkoittaa kehittäjälle tuntematonta haavoittuvuutta, jota hyökkääjä hyväksi käyttää hyökkäyksessä. Usein puhutaan myös *nollapäivähyökkäyksistä*, jotka hyödyntävät nollapäivähaavoittuvuuksia.

Haavoittuvuudet vastaavat kysymykseen:"Kuinka uhkat voivat toteutua?"

3.3 Tietoturvan rooli sosiaalisten verkkosovellusten kontekstissa

Sosiaaliset verkkosovellukset ovat tavallisesti jatkuvan muutoksen alaisena. Kompleksisuuden ja koon kasvaessa, käyttäjien läpinäkyvyyden lisääntyessä ja henkilökohtaisen informaation määrän kasvaessa, tietoturvan merkitys kasvaa entisestään. Kompleksisuutta lisäävät uudet toiminnot ja ennen kaikkea sovellusliittymät, Web 2.0:n työjuhta. Sosiaalisille verkkosovelluksille, etenkin suosituimmille, ilmestyy sovellusliittymiä käyttäviä pienohjelmia päivittäin. Mikään ei estä hyökkääjää tekemästä omaa pienohjelmaa. Koon kasvaminen tuo omat haasteensa. Laitteisto ja ohjelmistot eivät välttämättä skaalaudu koon kasvamisen mukaisesti. Tämä voi johtaa vikaantumisiin ja voi toimia ensi askeleena tietoturvamurrolle. Luonnollisesti suositut palvelut ja arkaluontoisen tiedon määrän lisääntyminen houkuttelevat enemmän huomiota hyökkääjiltä. Informaation läpinäkyvyys laajentaa käyttäjien digitaalista jalanjälkeä, jota voidaan hyödyntää potentiaalisissa hyökkäyksissä.

On tarpeellista kyetä erottamaan uhkat, joissa tietoturva ei vaarannu. Ihmisten yksityisyyttä tai ihmisoikeuksia voidaan loukata vaarantamatta tietoturvaa. Perinteinen uhkakuva on verkossa vaaniva pedofiili. Pedofiileja esiintyy verkossa yhä, joskin pelkästään tämä uhkakuva on liian yksipuolinen kattamaan koko totuutta. Ahdistelua voi harjoittaa yhtyeen yli-innokas fani, mustasukkainen aviomies tai radikalisoitunut poliittinen aktivisti. Sovellusten tarjoamat hakutyökalut mahdollistavat henkilöiden jäljittämisen, seuraamisen ja ahdistelun, tietyssä määrin, keinoin, jotka eivät vaaranna tietoturvaa.

Ahdistelija voidaan ilmiantaa kehittäjille sovelluksen työkaluilla tai sähköpostilla. Muita uhkia, joita voidaan harjoittaa tietoturvaa vaarantamatta, ovat kyberkiusaaminen ja nettiväkivalta. Tällainen toiminta tulee ilmoittaa keskustelualueen moderaattorille tai verkkosivuston ylläpitäjälle, jotta siihen voidaan asianmukaisesti puuttua. Suositeltavaa on myös ottaa poliisiin yhteyttä. Poliisin omat yhteydenottokanavat ja tukiryhmät voi löytää myös sosiaalisista verkkoyhteisöistä, mm. Facebookista ja IRC-galleriasta.

Yhteistä yllä oleville tapauksille on se, että ne rikkovat käytettyjen verkkoyhteisöjen käyttöehtosopimuksia ja että ne liittyvät yksityishenkilön yksityisyyden tai ihmisoikeuksien loukkaamiseen. Varsinaisista tietoturvauhkista kaikki eivät kohdistu yksityishenkilöihin tai kosketa yksityisyyttä tai ihmisoikeuksia. Vaikkakin tällainen toiminta voi olla hyökkäyksen motivoivana tekijänä, niin on huomioitava, että tietoturva ei aina voi ehkäistä tämän kaltaisia uhkatilanteita. Kyseessä ei ole tietoturvallisuusriski. Tietoturvan piiristä pois ovat myös poliittiset agitoinnit, valheiden levittäminen ja muu mahdollisesti laiton vaikuttaminen.

3.4 Sosiaalisen verkkosovelluksen mallintaminen

Tietoturvan toteuttamiseksi on hyödyllistä tarkastella sosiaalisia verkkosovelluksia viitekehyksessä, joka auttaa hahmottamaan ongelma-avaruutta. Yleisesti ottaen sosiaalisista verkkosovelluksista ei ole olemassa merkittävästi lähdemateriaalia [14].

Merkille pantavaa on Cutillon ym. [50] esittämät tietoturvan tavoitteet sosiaalisissa verkkosovelluksissa. Cutillo ym. kiinnittävät erityistä huomiota käyttäjien yksityisyyden turvaamiseen, korvaten CIA-mallin luottamuksellisuuden käsitteen **yksityisyyden** (engl. *privacy*) käsitteellä. Mitä yksityisyyden käsite pitää sisällään vaihtelee määrityksestä toiseen. Esimerkiksi tätä käsitettä ei tule sekoittaa Sørensenin yksityisyyden käsitteen kanssa (ks. kpl 2.5). Cutillon ym. yksityisyyden käsite on lähes samanlainen CIA-mallin luottamuksellisuuden käsitteen kanssa. Erona on osapuolten kommunikaation jäljitettävyys. Kiistämättömyyden käsite pitää si-

sällään tämän ominaisuuden. Nämä muutokset eivät tuo tietoturvan toteutukseen lisäarvoa.

Lisäarvoa tarjoavat kuitenkin CIA-mallin käsitteiden laajennukset sosiaalisten verkkosovellusten kontekstissa. Eheyden käsitettä Cutillo ym. laajentavat koskemaan käyttäjän identiteettiä. Eheys on uhattuna, kun käyttäjän identiteettiin kajotaan luvattomasti. Tällainen on esimerkiksi tekaistun identiteetin luominen tekeytymällä toiseksi, aidoksi henkilöksi. Saatavuuteen sisällytetään pääsy omiin profiilitietoihin ja palveluihin, pääsy muiden käyttäjien sisältöön ja vapaus vaihtaa viestejä muiden käyttäjien kanssa tarvittaessa. Yksityisyyden termi on luultavasti sopivampi sosiaalisten verkkosovellusten kontekstiin kuin luottamuksellisuuden käyttäminen, keskittäen huomiota enemmän käyttäjien turvaamiseen. Kuten aiemmin todettiin, tietoturvallisuuden viisi pilaria sopii kontekstiin erinomaisesti. Erityisesti aitouden todentaminen, autentikaatio, on tärkeä ulottuvuus sosiaalisissa verkkosovelluksissa.

Sosiaalisten verkkosovellusten kehittäjien velvollisuus on suojella käyttäjiensä arkaluontoisia tietoja. Jos ajatellaan yksityisyyden käsitettä yleisesti, ei Cutillon ym. [50] määritelmänä, käsitteen subjektiivisuudesta johtuen voi olla epäselvää, mitkä tiedot tulee suojella ulkopuolisilta. Tavallista on käyttää jonkinlaista jakoa ryhmiin sen mukaan, kuinka käyttäjä luottaa muihin. Mahdollisia jakoja ovat ystävät, ystävät ja ystävien ystävät, tietyn harrastuksen jakavat, jne.. Oletetaan, että käyttäjä itse tietää yksityisyytensä parhaiten ja tätä ajatusta tukemaan tavallisesti toteutetaan käyttäjän operoimat työkalut yksityisyyden hallinnoimiseksi.

Sørensenin ja Cutillon mallin yksityisyys käsitteet ovat miltei samanlaisia. Kummankin pääidea on (ks. kpl 2.5): *"Yksityisyys kuvaa sitä, kuinka hyvin käyttäjä voi suojella ja hallinnoida hänen arkaluontoista sisältöä kaikessa kanssakäymisessä, joka tapahtuu sosiaalisessa verkkosovelluksessa."*. Cutillon yksityisyyden käsitteeseen kuuluu piirteitä Sørensenin

Sosiaaliset verkkosovellukset voidaan jakaa kolmeen kerrokseen [50]:
Sosiaalisella kerroksella (engl. *Social network level*) käyttäjät ja heidän suhteensa ovat kuvattuina digitaalisessa esitysmuodossa. Käyttäjille näkyvät, korkean tason funktiot, sijaitsevat tällä kerroksella. Työkalut profiilien etsimiseksi, kommentointimahdollisuudet, chattitoiminnot ym. funktiot sisältyvät tähän kerrokseen. Kaikki puhtaasti sosialisoivat toiminnallisuudet ovat tällä kerroksella.
Sovelluskerros (engl. *Application services level*) sijaitsee sosiaalisen kerroksen alla, tarjoten sille sen tarvitsemat toiminnallisuudet ja toimien välttämättömänä kehyksenä sosiaalisen verkkosovelluksen toiminnalle. Sovelluskerrokseen kuuluvat lii-

tyntäpisteet eri verkkovyöhykkeisiin (internetistä sisäiseen, sisäisestä koostesivustoille, jne..), varastointi- ja kommunikaatiofunktiot. Kaikki käyttäjälle näkyvät tukitoiminnot, jotka eivät kuulu sosiaalisen kerrokseen, kuuluvat sovelluskerrokseen. Esimerkiksi Cutillo ym. [50] sijoittavat sovelluskerrokselle pääsynvalvontaan liittyvien tehtävien hallinnan. Näiden funktioiden tarjoamien palveluiden lisäksi sovelluskerros koordinoi sosiaalisen verkkosovelluksen toimintaa.
Kommunikaatio- ja kuljetuskerros (engl. *Communication and transport level*) kuvaa sosiaalisen verkkosovelluksen tietoverkkojen tasolla. Verkkolaitteet, työasemat, palvelimet, verkkoprotokollat, ym. kuuluvat tähän kerrokseen.

Jokaisella kerroksella tapahtuu muutoksia. Sosiaalinen kerros muuttuu uusien käyttäjien rekisteröityessä palveluun, suhteiden muuttuessa ja kun käyttäjille lisätään uusia työkaluja käytettäviksi. Sovelluskerros muuttuu kun uusia sovellusliittymiä ja palveluita integroidaan tietojärjestelmään. Kommunikaatio- ja kuljetuskerros kokee muutoksia kun topologia (verkkolaitteet, yhteyspisteet toisiin laitteisiin, työasemat, palvelimet) muuttuu, tai kun allaolevaa verkkoprotokollaa tai kryptografista algoritmia vaihdetaan. Kaikkien muutosten tulee tapahtua ennaltamääritellyillä ja hallituilla tavoilla, jotka säilyttävät järjestelmän tietoturvan.

3.4.1 Sosiaalinen kerros

Edetään tarkastelemaan sosiaalisia verkkosovelluksia ISO 17799-standardin tietoturvamääritelmän kautta. Standardin mukaan tietoturvan tulee säilyttää organisaation liiketoiminnan jatkuvuus ja toimivuus takaamalla tietojärjestelmän oikeanlainen toiminta. Täten oleellista on tarkastella, minkälaista on organisaation liiketoiminta ja minkälaisia ovat itse tietojärjestelmät, sosiaaliset verkkosovellukset ohjelmistotasolla.

Sosiaalisten verkkosovellusten tulot koostuvat useimmiten mainonnasta. Muita tulotapoja ovat tilausmaksut tai kertaluontoiset mikromaksut [51]. Tilausmaksut tuovat käyttäjille vaihtoehtoisia palveluita tai sisältöä käytettäviksi. Mikromaksut ovat usein virtuaalisia hyödykkeitä, kuten virtuaalisia lahjoja. Enders ym. [51] esittävät kolme ratkaisevaa tekijää sosiaalisten verkkosovellusten liiketoiminnalle:

- Käyttäjien lukumäärä
- Käyttäjien halukkuus maksaa
- Käyttäjien luottamus

Kiistämättä sosiaalisten verkkosovellusten suurin voimavara ovat käyttäjät, jotka muodostavat sosiaalisten verkkosovellusten sosiaalisen kerroksen yhdessä käyt-

täjille tarjottujen, korkean tason funktioiden, lisäksi. Täten liiketoiminnan jatkuvuuden ja toimivuuden kannalta oleellista on keskittyä käyttäjien tietoturvaan. Tulee kuitenkin huomioida, että sosiaalisissa verkkosovelluksissa toimii muitakin toimijoita kuin käyttäjät. Tietoturvauhkat voivat kohdistua esimerkiksi mainostajiin tai sosiaalisen verkkosovelluksen kehittäjiin. Tavallisesti huomio keskittyy käyttäjiä koskeviin tietoturvauhkiin [37] [50]. Joissakin tapauksissa kolmannen osapuolen kehittäjällä voi olla niin suuri vaikutusvalta, että sosiaalisen verkkosovelluksen kehittäjät tulevat heistä riippuvaisiksi.

Käyttäjistä ja heidän suhteistaan voidaan muodostaa verkosto. Käyttäjien muodostamaa sosiaalista verkostoa käsiteltiin kappaleessa 2.3. Käyttäjäkunnan koostumus riippuu sosiaalisesta verkkosovelluksesta. Jotkut ovat orientoituneet ammattielämään, kuten LinkedIn [52]. Käyttäjäkunnan koostumuksen merkitystä tietoturvalle on hankala arvioida. Aihe vaatii suurten tilastollisten tietomäärien analysointia, eikä tutkielman kirjoituksen aikana tällaisia ollut käytettävissä. Täten aihe sivuutetaan. Kaikkein suurimpien sosiaalisten verkkosovelluksen tapauksessa voidaan olettaa, että käyttäjäkunta koostuu hyvin monimuotoisesta joukosta. Jatkossa tämä olettamus oletetaan todeksi.

Kuten kappaleessa 2.4 todettiin, ovat **sosiotekniset käyttäjien manipulointitavat** (engl. *social engineering*) yleistyneet [28]. Tällaisia tekniikoita käyttämällä hyökkääjien ei tarvitse murtaa tietojärjestelmän ohjelmistotason tietoturvaa saavuttaakseen päämääränsä. Uhka on merkittävä, sillä monimutkaiset ja kalliit ohjelmistotason tietoturvaratkaisut voivat olla voimattomia tällaisia tietoturvariskejä vastaan. Hyökkääjät ovat tietoisia näiden tekniikoiden eduista. Kappaleessa 2.5 esitettiin luottamuksen ongelma, johon nämä tekniikat perustuvat. Kappaleissa 4.3 ja 5.2.1 esitetään joitain ohjeistuksia kyseisten uhkien välttämiseksi.

Matthias ym. [53] tuovat esille moderneissa verkkosovelluksissa käyttäjien vastuulle tuotujen tietoturvamekanismien hallinnan. Usein käyttäjä itse joutuu tutustumaan —usein monimutkaisiin ja lukuisiin— tietoturvamekanismeihin. Tällaisessa ratkaisussa työkalujen ja tietoturvakonseptien ymmärtäminen tulee tehdä mahdollisimman helpoksi käyttäjälle. Erityistä huomiota tulee kiinnittää oletustietoturvaasetuksiin. Tietoturvallisuustekniikan näkökulmasta nämä työkalut usein vastaavat tehtävistä joita pääsynvalvonnalla hallinnoidaan.

Web 2.0:n mahdollistama käyttäjien sisällöntuotanto on erittäin merkittävä erityispiirre tietoturvan suhteen. Jos käyttäjien sisällön lisääminen, koostaminen, päivittäminen ja poistaminen on suunniteltu ja toteutettu huolimattomasti, ne voivat muodostua tietoturvariskeiksi. Näiden toiminnallisuuksien ennakoimattomat käyttömahdollisuudet voivat mahdollistaa tietojärjestelmän väärinkäytökset.

3.4.2 Sovelluskerros

Sosiaaliset verkkosovellukset ovat moderneja, internetissä toimivia verkkosovelluksia. Yleensä pääasiallinen käyttö tapahtuu internetin kautta, näin ei kuitenkaan ole esimerkiksi Twitterin tapauksessa [9]. Tietojärjestelmään on kuitenkin olemassa poikkeuksetta yhteys, web-käyttöliittymä, internetin kautta. Web-käyttöliittymässä hyökkääjä voi pystyä hyödyntämään tuntemattomia haavoittuvuuksia, jotka tarjoavat pääsyn sovelluskerrokseen. Muita yhteyksiä sovelluskerrokseen ovat lukuisat sovellusliittymät ja haavoittuvuudet, jotka ovat hyödynnettävissä ilman järjestelmään sisäänkirjautumista.

Verkkosovelluksilla on useita ominaisuuksia, jotka tulee ottaa huomioon [54]:

1. Verkkosovellusten hajautettu luonne, joka vaikeuttaa sovelluskehitystä ja ylläpitotoimenpiteitä
2. Yhtäaikaisuuden ongelmat (esimerkiksi samanaikaisten pyyntöjen käsittely, kun ne koskevat samaa resurssia)
3. Sovellusliittymät, jotka ovat tärkeässä roolissa verkkosovelluksen koordinoinnissa ja käyttäytymisessä
4. Monimuotoiset synkroniset ja asynkroniset viestintämenetelmät ja prosessien etäkutsut
5. Laaja kirjo käytettyjä teknologioita, skriptaus- ja ohjelmointikieliä ja tiedon esitystapoja
6. Usean eri organisaation väliset tapahtumat, protokollat ja standardit

Käydään lyhyesti läpi näiden ominaisuuksien merkityksestä tietoturvalle.
Kohta 1) tuo lisää käsittelykustannuksia sovelluskehitykseen ja ylläpitoon. Verkkosovellukset voivat olla hyvinkin laajoja ja monimutkaisia kokonaisuuksia, joiden ylläpitäminen vaatii yllättävän paljon aikaa ja resursseja. Hajautettu luonne on rakennettu jakamisen, saatavuuden ja avoimuuden suunnitteluperiaatteita kunnioittaen, ja onkin tärkeää hallita näitä verkkosovellusten ominaisuuksia väärinkäytösten välttämiseksi.
Kohdat 1), 3) ja 5) luovat hyvin monimutkaisen kehyksen verkkosovelluksen toiminnalle. Lukuisten teknologioiden, protokollien ja kielien joukko muodostaa monimutkaisia yhdistelmiä, joiden yhteentoimivuutta on vaikea arvioida. Tuntemattomat loogiset virheet näiden suunnittelussa voivat osoittautua tietoturvariskeiksi.

Huonosti suunnitellut tai integroidut sovellusliittymät voivat rikkoa tietojärjestelmän tietoturvan, vaikka yksittäiset osakomponentit toimisivat tietoturvan kannalta loogisesti oikein. Hajautettu ympäristö tuo omat haasteensa. Monet eri toimijat ja eri verkkolähteistä **koostettu materiaali** (engl. *mashups*) voivat muodostua tietoturvariskeiksi, jos niitä ei valvota.
Kohdan 2) tuomat tietoturvaongelmat ovat ratkaistavissa tietokantojen ja tietorakenteiden huolellisella suunnittelulla, toteutuksella ja ylläpidolla. Lisäksi loogiset kanavat, joita pitkin data saavutetaan, tulee hallinnoida koordinoidulla tavalla, joka minimoi konfliktit. Ylläpidossa tulee ottaa huomioon kasvuvara ja datan jäljennöksien varaamat resurssit.
Kohtaa 4) koskien on myös kiinnitettävä huomiota tietojärjestelmän loogisiin kanaviin, joita pitkin informaatio kulkee. Tuntemattomat prosessit voivat aiheuttaa tietoturvariskejä, jos niiden aitoutta ei varmenneta. Tahallisesti tai tahattomasti muodostuneet syntaksiltaan virheelliset viestit tai viestien suuri lukumäärä voivat lamauttaa tietojärjestelmän. Seurauksena voi olla pääsy tietojärjestelmän tietoturvamekanismien ohi tai tietojärjestelmän menetetyt resurssit.
Kohta 6) pitää sisällään tietoturvariskejä, jotka voivat johtua moninaisista syistä. Väärin ymmärretyt toimintaohjeet eri organisaatioiden välillä, protokollien väärinkäytökset ja epäyhteensopivat standardit ovat esimerkkejä tällaisista riskeistä.

Sosiaalisten verkkosovellusten arkkitehtuuriratkaisut ovat yleisesti tiedossa [14]. Tämä on eduksi hyökkääjälle, joka voi keskittyä etsimään tunnettuja haavoittuvuuksia kyseisistä teknologioista. Useat käytetyistä teknologioista ovat avoimen lähdekoodin teknologioita, joka mahdollistaa hyökkääjän tutustua näiden teknologioiden lähdekoodiin perinpohjaisesti. Sosiaalisten verkkosovellusten kehittäjien tulee olla tietoisia käytettyjen teknologioidensa tunnetuista haavoittuvuuksista.

3.4.3 Kommunikaatio- ja kuljetuskerros

Sosiaalisten verkkosovellusten pohjana on tietoverkko. Tietoverkko vastaa sosiaalisten verkkosovellusten alinta kerrosta, kommunikaatio- ja kuljetuskerrosta. Eduksi on tutustua tietoverkkojen tietoturvateoriaan. Tämän opinnäytteen kannalta tähän kerrokseen sisällytetään tavalliset tietoverkon suojausmekanismit: kryptografiset algoritmit, näitä hyödyntävät salausmenetelmät, avaintenhallinta, autentikointi, palomuurit, virustentorjuntaohjelmistot ja tunkeutumisenestojärjestelmät [49].

3.5 Hyökkääjän analysointi

Tietoturvariski voi muodostua ilman hyökkääjää. Ohjelmistovirhe, jonka takia sovellus syöttää käyttäjän näkyville arkaluontoisia henkilötietoja muista yhteisöpalvelun käyttäjistä, kun sovellusta on käytetty asianmukaisesti, on tietoturvariski. Tietojärjestelmän luottamuksellisuutta ja eheyttä on rikottu sovelluskehittäjien toimesta. Tällainen tietoturvariski ei vaadi hyökkääjää. Tämä opinnäyte keskittyy tietoturvariskeihin, joissa on osallisena hyökkääjä.

Kyberrikollisten kohteena ovat aiemmin olleet pääasiallisesti työpöytäsovellukset, mutta kuluneella vuosikymmenellä huomio on siirtynyt verkkosovelluksiin. Vuoden 2008 loppupuolelle saakka hallinnollisten elinten verkkosivustot ovat olleet verkkohyökkäysten ensisijainen kohde. Sosiaaliset verkkosovellukset ovat nousseet suosituimmaksi kohteeksi hyökkäyksille ja ovat sitä edelleen tutkielman kirjoituksen aikana [19]. Trendi on kasvanut voimakkaasti vuodesta 2008 lähtien. Vuoden 2010 aikana sosiaalisten verkkosovellusten kautta suoritetut hyökkäykset tai näihin kohdistuvat hyökkäykset ovat lisääntyneet 70 %. Useat tietoturvaratkaisuja tarjoavat yritykset ovat nostaneet vuoden 2010 tietoturvateemaksi yhteisöpalveluihin kohdistuvat tietoturvauhkat [28] [55] [56] [57].

Hyökkääjän analysointi voi tarjota arvokasta lisätietoa siihen, minkälaisia hyökkäyksiä tietojärjestelmä tulee kohtaamaan. Hyökkääjän analysoinnissa tavoitteena on ymmärtää hyökkääjän motiiveja, käytettävissä olevia resursseja, hyökkäysmetodeja ja tilaisuuksia toteuttaa hyökkäys. Näitä tietoja hyväksikäyttämällä tavoitteena on vastata kysymykseen: mitkä hyökkäykset ovat todennäköisiä? Tällainen tietämys on arvokasta kehittäjille mietittäessä sovelluksen tietoturvavaatimuksia. Tarkastellaan lähemmin, kuinka tietämys hyökkääjästä voi palvella hyökkäysten ymmärtämisessä. Schneier [58] kuvaa hyökkääjän viiden ominaisuuden kautta:

- **Motivaatio** (engl. *Motivation*)
- **Pääsy** (engl. *Access*)
- **Taito** (engl. *Skill*)
- **Uskallus** (engl. *Risk aversion*)
- **Rahoitus** (engl. *Funding*)

Schneier [58] esittää esimerkin kassakaapin murtamisesta, joten on syytä olettaa hänen työnsä liittyvän turvallisuustekniikkaan. Koska tämä opinnäyte keskittyy tietokoneilla tehtyihin hyökkäyksiin, tietotekniikkarikoksiin, Schneierin malli kaipaa

hieman sovittamista. Schneierin esittämä pääsy tulee ymmärtää yhteyspisteenä, jonka kautta hyökkääjä voi suorittaa hyökkäyksen, jotta sitä ei sekoitettaisi käyttöoikeuksiin, pääsynä resursseihin eikä erityisesti pääsynä fyysisiin tiloihin.

Motivaatio on tärkeä ja usein ensimmäinen askel hyökkääjän ymmärtämiseen. Motivaationa on yhä useammin taloudellisen hyödyn tavoittelu [28]. Tämä on nouseva trendi myös sosiaalisten verkkosovellusten parissa [20]. Motivaationa voi olla pila, toisen naurunalaiseksi tekeminen tai maineen töhriminen. Sosiaalisissa verkkosovelluksissa on paljon "yleisöä", joka toimii kannustimena tällaisille teoille. Esimerkkejä ovat kohde, joka on hyökkääjän vastenmielisenä pitämän ryhmän jäsen, vaikutusvaltainen keskussolmu sosiaalisessa verkostossa tai kohde jonka kanssa hyökkääjällä on ollut erimielisyyksiä. Motivaationa voi olla omien taitojen testaaminen tai maineen tavoittelu. Hyökkääjä saa henkilökohtaisen saavutuksen tai vallan tunteen hyökkäyksen onnistuttua. Hyökkäyksellä voi olla poliittisia tarkoitusperiä [27]. Motivaationa voi olla myös pelkkä vahingonteko. Hyökkääjä voi esimerkiksi hävittää käyttäjien profiileja.

Hyökkääjä on yhä useammin taitava ja päämäärätietoinen kyberrikollinen [57]. Tällöin motivaationa on useimmiten taloudellisen hyödyn tavoittelu. Taloudellinen hyöty ei välttämättä heti realisoidu hyökkääjälle. Hyökkääjä voi saada kohteen tietokoneen käyttöönsä, käyttäjän tätä tietämättä, ja hyödyntää uhrin tietokonetta myöhemmissä hyökkäyksissä. Tavallisesti tämä tarkoittaa tietokoneen käyttämistä roskapostituksessa tai palvelunestohyökkäyksissä [56]. Kuten kappaleessa 2.2 todettiin, hyökkääjä voi käyttää keräämiään tietoja myyntitavarana. Taitava kyberrikollinen voi myydä palveluitaan terroristeille, valtioille tai yrityksille. Tällöin hyökkäyksen motiivi voi olla poliittinen, terrorismi tai kilpailuedun tavoittelu.

Muissa tapauksissa hyökkääjän tunnistaminen on vaikeampaa. Hyökkääjä voi olla tavanomainen käyttäjä, kaupallinen kilpailija, mainetta hakeva kräkkeri, epäsosiaalinen yksilö, ts. potentiaalisten hyökkääjien joukko on suuri ja monimuotoinen. Erityisen vaarallisen uhkakuvan muodostaa hyökkääjä, joka on järjestelmän työntekijä. Mitä enemmän työntekijällä on valtuuksia ja luottamusta organisaation sisällä, sitä vakavamman uhkakuvan hän muodostaa.

Yleisesti ottaen kaikki sovellusliittymät tarjoavat pääsyn sosiaalisiin verkkosovelluksiin. Hyökkääjän ollessa kolmannen osapuolen sovelluskehittäjä, on hyökkääjällä pääsy sosiaalisen verkkosovelluksen sovellusliittymien ohjelmointirajapintaan laajemmilla käyttöoikeuksilla. Kaikki sosiaalisen verkkosovelluksen tarjoamat palvelut kolmansille osapuolille muodostavat pääsyn tietojärjestelmään, esimerkiksi syndikointi- ja koostepalvelut. Pääsy järjestelmään voi käydä myös suojaamattomien verkkoresurssien kautta, esimerkiksi sovelluksen testaukseen tarkoitetun

verkkosivun kautta. Hyökkäyksen ei ole pakko koskea sosiaalisen verkkosovelluksen kehittäjää, se voi kohdistua myös muihin toimijoihin, esimerkiksi kolmannen osapuolen sovelluspalvelimiin, joihin on pääsy sosiaalisesta verkkosovelluksesta. Kaikki nämä liittyvät sovelluskerrokseen.

Sosiaalisen kerroksen kannalta hyökkääjä voi toteuttaa hyökkäyksen sovelluksen omalla käyttöliittymällä. Käyttäjätilin luominen useimpiin sosiaalisiin verkkosovelluksiin on ilmaista. Hyökkääjällä on käytettävissä tavanomaiset työkalut sisällöntuotantoon sekä kaikki palvelut, jotka ovat käyttäjän käytettävissä. Hyökkääjällä on pääsy profiileihin, jotka on määritelty julkisiksi. Muihin profiileihin hyökkääjällä on pääsy kyseisen sosiaalisen verkkosovelluksen pääsynvalvonnan määräämien sääntöjen mukaisesti. Hyökkääjä voi kuulua kohteen ystävien joukkoon kyseisessä verkkoyhteisössä, joka lisää käyttöoikeuksia tarkastella kohteen tietoja.

Viimeksi, hyökkääjän on mahdollista toteuttaa hyökkäys järjestelmään kommunikaatio- ja kuljetuskerroksen kautta. Pääsyn mahdollistaa mikä tahansa tietoverkon ulkoreunalla oleva verkkolaite. Ajallisesti pääsy sosiaalisiin verkkosovelluksiin on aina, poikkeuksen muodostaa verkkoyhteisön huolto- ja ylläpitotoimenpiteet.

Taito on merkittävin tekijä sen kannalta, onnistuuko hyökkäys. Jos hyökkääjä on kyberrikollinen, esimerkiksi teollisuusvakoilua suorittava kräkkeri, tulee odottaa tällaisella hyökkääjällä olevan tietotaitoa suorittaa hyökkäys. Potentiaalisten hyökkääjien joukko on suuri ja monimuotoinen, joten on syytä olettaa, että hyökkääjän taito suorittaa hyökkäyksiä sisältää suurta vaihtelua. Odotettavaa on, että suositut sosiaaliset verkkosovellukset kohtaavat monia, huonosti toteutettuja hyökkäyksiä.

On huomioitava hyökkääjän käytössä olevat apukeinot. Lukuisten hyökkäysten vaatima tieto on saatavilla internetistä tai kirjoista [59]. Tietoverkon analysointiin ja tietoverkon tietoturvan arvioimiseen tehtyjä työkaluja [59] voi käyttää hyökkäyksessä apuna. Vaarallisimpia ovat kräkkereiden luomat rikollisiin toimiin tarkoitetut työkalut (engl. *crimeware kit*) [28]. Täten myös taitamaton kykenee, oikeilla työkaluilla ja tarvittavilla tiedoilla, suorittamaan vakavasti otettavia, kehittyneitä hyökkäyksiä.

Uskallus määrää kuinka riskialttiita toimenpiteitä hyökkääjät ovat valmiita tekemään. Tietoverkkojen mahdollistama anonyymius pienentää kiinnijäämisen pelkoa [49, s95]. Kyberrikollisten tavallinen infrastruktuuri hyökkäyksille [56], orjakonejoukot, tekevät kyberrikollisten jäljittämisen miltei mahdottomaksi. Orjakonejoukon voi valjastaa moniin eri tehtäviin: haitakkeiden levittämiseen, roskapostitukseen, palvelunestoon, teollisuusvakoiluun, ym. Kun hyökkäys voidaan suorittaa anonyymisti usean välityspalvelimen tai orjakonejoukon turvin, se lisää huomattavasti uskallusta suorittaa hyökkäyksiä.

Jos hyökkääjällä on rahoittaja, on luultavaa, että hyökkääjä on ammattimainen kyberrikollinen. Tärkeä rahoituksella saatava hyödyke tietokoneella suoritettuihin, tietokoneverkkoihin kohdistuviin hyökkäyksiin, on orjakonejoukko [56]. Usein vaarallisemman vaihtoehdon tarjoaa kohdeorganisaation sisältä palkattu luottamuksen ja käyttöoikeuksia omaava työntekijä. Tilanteesta riippuen, rahoituksella voi saada muunlaista kriittistä tietoa tai materiaalia: salasanoja, yhteystietoja, kulkulupakortteja, ym.

3.6 Tietoturvan toteuttaminen sosiaalisiin verkkosovelluksiin

Tämä kappale on jäsennelty seuraavasti: Aluksi luodaan lyhyt katsaus tietoturvallisuuden hallintaan. Kappaleessa 3.6.1 luodaan yleiskatsaus riskinhallintaan, kappaleessa 3.6.2 tarkastellaan tietoturvan sisällyttämistä ohjelmistotuotantoon ja kappaleessa 3.6.3 tarkastellaan sosiaalisten verkkosovellusten olennaisimpia tietoturvakonsepteja.

Tietoturvapolitiikka (engl. *Information security policy*) luodaan tavallisesti tietoturvallisuuden hallinnoimiseksi [60]. Tietoturvapolitiikka on usein ensimmäinen asiakirja, josta tietoturvallisuutta aletaan toteuttamaan organisaation tietojärjestelmiin. Tietoturvapolitiikka luo säännöt ja käytännöt sille, kuinka organisaatio hallinnoi, suojelee ja jakaa arkaluontoista informaatiota. Tietoturvapolitiikka määrittää myös kunkin (tietokonepohjaisissa järjestelmissä henkilöiden ja prosessien) oikeudet ja pääsyn organisaation etuihin (mukaanlukien informaatioon) [60]. Tietoturvapolitiikan linjausten mukaisesti edetään yksityiskohtaisimpiin asiakirjoihin, jotka keskittyvät kapeampiin tietoturvallisuuden osa-alueisiin, esimerkiksi käyttöoikeuksien valtuutukseen. Nämä asiakirjat ohjeistavat, määräävät ja linjaavat tietojärjestelmien ja sovellusten suunnittelua, toteutusta ja ylläpitoa, siten että tarvittava tietoturvan taso saavutetaan ja ylläpidetään. On tärkeää huomioida, että tietoturvaa tulee ylläpitää, sillä mikään tietojärjestelmä ei toimi staattisessa ympäristössä.

Apuna —tai ratkaisuna— tietoturvallisuuden hallintaan voidaan käyttää tähän tarkoitettuja standardeja, tietoturvallisuuden hallintajärjestelmiä [61] tai tarkistuslistatyyppisiä menetelmiä. Tarkistuslistatyyppiset [44] menetelmät eivät ole yhtä kattavia ja yhdenmukaisia, vaan tarjoavat lähinnä yleisesti tunnettuja käytäntöjä ja hyviä neuvoja.

Useiden organisaatioiden tietoturvaa koskeva tutkimus [62] osoittaa, että tietoturvapolitiikalla on tärkeä vaikutus organisaation tietoturvallisuuden laatuun. Hyvin määritelty tietoturvapolitiikka auttaa selvittämään organisaatiolle mikä on

tietoturvan merkitys ja parantaa tietoturvaa luovien prosessien suunnittelua, toteuttamista ja hallinnointia.

Tietoturvallisuuden hallintamenetelmät (standardit, hallintajärjestelmät ja tarkistuslistat) tarjoavat laajemmalla, usein organisaation kattavalla tasolla tietoturvallisuutta. Suositeltavaa on hyödyntää myös vapaasti saatavilla olevaa, asiantuntijoiden koostamaa, tietämystä verkkosovellusten tietoturvasta. OWASP- ja WASC-organisaatiot tarjoavat runsaasti materiaalia ja hyviä ohjeita [63] [64]. Erityisesti sosiaalisten verkkosovellusten tietoturvariskejä käsittelee **ENISAn** (European Network and Information Security Agency) vuonna 2007 valmistunut raportti [65].

3.6.1 Riskinhallinta

Riskinhallinta kuvaa niitä koordinoituja toimenpiteitä, jotka ohjaavat ja hallinnoivat organisaatioon kohdistuvia riskejä. Riski voi olla mitä tahansa, joka uhkaa organisaation liiketoimintaa. Riskinhallinta on tärkeä prosessi, sillä se luo liiketoiminnan kautta vertailuperusteet organisaation turvallisuutta parantaville toimenpiteille. Näin turvallisuustekniikka saa oikeutuksen toiminnalleen organisaatiossa.

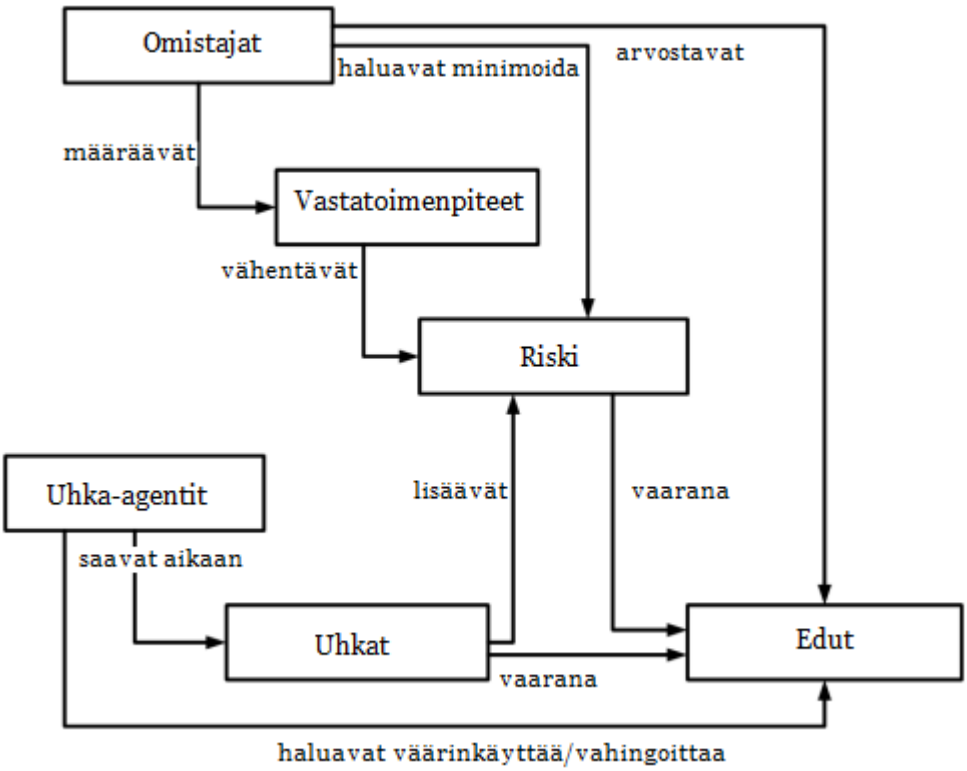
Riskinhallintaan kuuluu riskien tunnistaminen, riskien käsittely ja riskien seuraamukset [61]. Kukin riski arvioidaan sen asiaankuuluvuuden ja todennäköisyyden kannalta, kuinka kalliiksi riskin välttäminen tulee sekä mitä seuraamuksia riskin ilmentymisellä on organisaatiolle. Näiden tietojen pohjalta organisaatio päättää kuinka toimia kyseisen riskin kanssa. Riskien käsittely jaetaan yhteen tai useampaan kategoriaan [44] [66]. Tavallisesti riskien käsittely sisältää seuraavat kategoriat:

- **Välttäminen** (engl. *Avoidance*)
- **Minimointi** (engl. *Reduction*)
- **Jakaminen** (engl. *Sharing*)
- **Hyväksyminen** (engl. *Retention*)

Välttäminen koskee toimintaa, jossa riski pyritään väistämään tai poistamaan. Minimointi koskee riskejä, joita ei voi täysin välttää mutta joiden vaikutusta tai esiintyvyyttä tulee vähentää. Jakaminen koskee toimenpiteitä, jossa riskin vaikutusta pyritään vähentämään jakamalla se muiden tahojen kesken. Vakuutuksen ottaminen organisaatiolle on riskien jakamisen muoto. Kaikkia riskejä ei voi välttää, vähentää tai jakaa. Riski tulee tällöin hyväksyä.

Tietoturvariskit ovat erityisesti riskejä, jotka liittyvät organisaation tietoturvallisuuden vaarantumiseen. Common criteria -standardi [67] on kansainvälinen stan-

dardi tietokoneiden tietoturvan arvioimiseksi. Standardin esittämästä mallista (kuva 4.1) nähdään, kuinka riski sijoittuu suhteessa muihin tietoturvan käsitteisiin.



Kuva 3.1: Tietoturvan konseptit ja niiden väliset suhteet.

Uhka-agentti (engl. *threat agent*) kuvaa mitä tahansa uhkaavan tapahtuman toimeenpanijaa. Tässä kontekstissa hyökkääjä on uhka-agentin alijoukko. **Vastatoimenpide** (engl. *Countermeasure*) kuvaa mitä tahansa toimenpidettä, jolla pyritään hallitsemaan riskiä (riskinkäsittely). Uhkan ja riskin olennaisin ero on siinä, että uhkat ovat eriarvoisia eri organisaatioille. Uhkista voi muodostua riskejä tai ne voivat lisätä jotain riskiä. Esimerkiksi uhkat *luvaton pääsy etuihin* ja *luottamukselliseksi määritetyn tiedon julkistaminen* voivat lisätä riskiä *kilpailuedun menettäminen vieraille tahoille*. Riskinhallinnan tehtävä on tunnistaa mitkä uhkat ovat organisaatiolle varsinaisia riskejä.

Riskinhallinta on prosessi, joka alkaa ennen sovelluskehitystä ja jatkuu sovelluksen koko elinkaaren ajan. **Alustava riskinhallinta** (engl. *Preliminary risk assessment*) [48] suoritetaan ennen sovelluskehitystä. Tarkoituksena on päättää, ylittääkö järjestelmästä saatavat hyödyt järjestelmään liittyvät riskit. Jos ylittää, järjestelmää voidaan alkaa kehittämään luomalla tälle tietoturvavaatimukset. **Elinkaarel-**

lisessa riskinhallinnassa (engl. *Life cycle risk assessment*) [48] järjestelmän arkkitehtuuri, komponentit, tietorakenteet ja muut teknologiset ratkaisut ovat tiedossa. Arviointiprosessi jatkuu koko elinkaaren ajan ja tuottaa palautetta sovelluskehittäjille. Palautteen tarkoitus on pitää sovelluksen tietoturvan taso riittävänä, vaikuttamalla suunnittelu- ja toteutusratkaisuihin.

Muun muuassa voittoa tavoittelemattomat organisaatiot [19] [63] ja virustorjuntaohjelmistoja valmistavat yritykset [55] [56] [57] ovat useiden vuosien ajan tilastoineet hyökkäyksiä ja raportoineet uusista hyökkäyksistä. Tilastoista voidaan päätellä nousevia trendejä. Nämä lähteet ovat arvokkaita lähteitä analysoitaessa mahdollisia tietoturvariskejä. Sosiaalisten verkkosovellusten kannalta todennäköisimmät tietoturvariskit ovat sosioteknisiä manipulointitapoja hyödyntävät hyökkäykset, roskapostitus, haitakkeet, XSS-haavoittuvuudet *(Cross site scripting)*, selainten haavoittuvuuksia hyödyntävät hyökkäykset ja sovellusliittymien haavoittuvuuksia hyödyntävät hyökkäykset [20] [57]. Verkkosovelluksina potentiaalisten tietoturvariskien luokkaan tulee lukea myös OWASP-organisaation luetteloimat verkkosovellusten kymmenen vaarallisinta tietoturvariskiä [63].

3.6.2 Tietoturvan integroiminen sovelluskehitykseen

Tietoturva tulee ottaa huomioon sovelluksen koko elinkaaressa [44] [48, s7], aina määrittelystä ylläpitoon. Onnistuneen sosiaalisen verkkosovelluksen ylläpito- ja jatkokehitysvaiheet kattavat merkittävän osan elinkaaresta. Tietoturva tulee integroida myös näihin käyttöönoton jälkeisiin vaiheisiin.

Tietoturvavaatimukset antavat sovellukselle lähtökohdat, millaisia tietoturvallisuutta luovia toimenpiteitä sovellus tarvitsee. Tietoturvavaatimukset tulee kuvata sellaisella tarkkuudella, että niistä ilmenee selvästi, millaisilla toimenpiteillä kukin vaatimus voidaan täyttää. Sovelluskehityksessä aikaisessa vaiheessa tehdyt väärät suunnitteluratkaisut tulevat sitä kalliimmaksi korjata, mitä pitemmällä ohjelmistotuotannossa ollaan. Tämä pätee myös tietoturvallisuutta koskeviin ratkaisuihin [48, s7]. Tästä syystä erityistä huomiota tulee kiinnittää siihen, että valitaan sovellukselle sopivimmat tietoturvallisuutta luovat ratkaisut. Sopivaa ratkaisua valittaessa tulee kiinnittää huomiota sen toteutukseen vaadittuja taitoja, resursseja, arkkitehtuuriratkaisuja, sekä ylläpidollisia toimenpiteitä, kustannustehokkuutta ja sen vaikutusta sovelluksen ei-toiminnallisiin vaatimuksiin.

Usein luotetaan määrään enemmän kuin laatuun ja toteutetaan epäsopivia ratkaisuja. Palkataan lisää henkilöstöä, mutta ei huomioida kuinka paljon uusi väki hyödyttää organisaatiota [62]. Tällaiset ratkaisut eivät tarjoa riittävää suojaa tai voi-

vat joissain tapauksissa vähentää tietojärjestelmän tietoturvallisuutta. Epäsopivat ratkaisut tuhlaavat organisaation resursseja. Optimaalisen ratkaisun löytäminen on usein mahdotonta, joten on hyvä hyödyntää tietämystä samankaltaisista, toimivista ratkaisuista.

Hyökkäyksiä ja haavoittuvuuksia tutkimalla voidaan oppia minkä takia vastaavat tietojärjestelmät ovat vaarantuneet. Tämä auttaa kehittäjiä välttämään samojen virheiden toistamista sovelluskehityksessä. Sovellusten tietoturvan arvioiminen pohjautuu usein tunnettujen haavoittuvuuksien etsimiseen. Uusien haavoittuvuuksien löytäminen on yhä vaikea ongelma tietoturvallisuustutkimuksessa. [68]

Väärinkäyttötapausten (engl. *Abuse cases*) [48, s9] luominen auttaa hyökkäysten mallintamisessa. Ajattelutapa on sama kuin käyttötapauksissa, sillä erolla että, tavoitteena on hyökätä tietojärjestelmään ja ajatella kuten hyökkääjä. Väärinkäyttötapaukset voivat tuoda ennaltanäkemättömiä, yksinkertaisiakin, haavoittuvuuksia esiin tietojärjestelmästä.

Jotkut tietojärjestelmän vaatimukset voivat olla ehdottomia. Niistä voi muodostua rajoitteita. Tietojärjestelmän rajoitteet olennaisesti vaikuttavat siihen, mitä suojauksia järjestelmälle toteutetaan. Organisaation tai liikekumppanien vaatimukset voivat asettaa rajoitteita suunnittelulle. Infrastruktuuri, jossa tietojärjestelmä tulee toimimaan, voi asettaa rajoitteita sovellukselle. Riippuvuudet, kuten riippuvuus kolmannen osapuolen tuottamaan palveluun, voivat luoda rajoitteita sovellukselle. Laitteisto voi vaikuttaa sovelluksen teknologiaratkaisuihin. Käytettävissä olevat toimitilat ja laitteiston sijoittelu voivat vaatia erityisiä toimenpiteitä.

Kappaleessa 3.4.2 esitettyjen huomioiden lisäksi haasteellista verkkosovellusten sovelluskehitykselle on niiden jatkuva evoluutio ja sovelluskehityksen nopeus. Tavanomaiset sovellusten suunnittelumetodologiat eivät sovellu hyvin siihen erikoislaatuiseen ja vaativaan kontekstiin, jossa verkkosovellukset toimivat. Verkkosovelluksille tarkoitettuja suunnittelumetodologioita, jotka ovat antaneet hyviä tuloksia, on olemassa. [54]

Sovelluskehityksessä voidaan hyödyntää ohjelmistokehyksiä, jotka tarjoavat usein käytettyjen toiminnallisuuksien osatoteutuksia (autentikaatio, valtuuttaminen, istunnon hallinta, jne..). Ohjelmistokehystä valittaessa tulee huomioida muiden kriteerien lisäksi ohjelmistokehyksen implikaatiot tietoturvalle.

Koska verkkosovellukset joutuvat toimimaan erittäin heterogeenisessä toimintaympäristössä, on mahdotonta varmistaa kaikkien osakomponenttien (kielet, protokollat, ohjelmistokehykset ja ohjelmakirjastot, valmiit ohjelmistokomponentit, ym.) tietoturvallisuus. Muiden osakomponenttien vikoja voi joutua korjaamaan jälkeenpäin usein standardista poikkeavilla, paikkauksilla, jotka harvoin ovat yhtä

varmoja kuin alusta alkaen tietoturvalliseksi tehty osakomponentti [48, s8]. Verkkosovellusten kehityksessä käytetään monia valmiskomponentteja. Kehittäjät voivat korvata valmiskomponentit omilla komponenteillaan. Eräs vaihtoehto on sovittaa, tai korvata, joitain osakomponentteja, siltä osin kun lainsäädäntö, sopimukset ja osakomponenttien keskinäiset yhteensopivuudet sallivat. Esimerkiksi Facebookin kehittäjät ovat muokanneet HTML-merkkauskieltä omiin tarkoituksiinsa sopivammaksi, nimeten sen FBML:ksi (FaceBook Markup Language) [69]. Verkkosovelluksille, jotka toimivat vaativassa toimintaympäristössä ja ovat valmistettu lukuisista osakomponenteista, syvyyssuuntainen turvallisuusajattelu on tärkeä periaate tietoturvallisuuden toteuttamisessa.

Syvyyssuuntainen turvallisuusajattelu (engl. *Defense in depth*) [48, s13] on tietoturvallisuustekniikassa käytetty periaate, jossa muodostetaan tietojärjestelmän jokaiselle kerrokselle suojaus. Tarkastellaan suojausten toteuttamista kappaleessa 3.4 esitetyn mallin mukaisesti. Esimerkkejä suojauksista kommunikaatio- ja kuljetuskerroksella on tietokantapalvelimen datan turvaaminen, sovellustasolla käyttäjän autentikointi ja sosiaalisella kerroksella käyttäjien välisen kommunikaation salaus. Varsinainen suojausmekanismi, joka toteuttaa suojauksen, voi sijoittua eri kerrokselle. Käyttäjien välisen kommunikaation salaus voidaan naiivisti toteuttaa salaamalla kaikki verkkoliikenne, jolloin suojausmekanismi sijoittuu kommunikaatio- ja kuljetuskerrokselle. Toisaalta sama voidaan saavuttaa TLS-protokollalla [49, s30], joka toimii OSI:n ISO-kerroksen mukaisessa jaottelussa kuljetuskerroksen ja sovelluskerroksen välissä. Cutillon ym. mallin käyttäminen suojausten ja suojausmekanismien yhteydessä aiheuttaa epäselvyyksiä. Olennaisesti ongelmana on suojauksen funktion (käyttäjien välisen kommunikaation salaus) epäyhtenevä sijoittuminen kerroksiin sen toteutuksen kanssa (saavutetaan toteuttamalla järjestelmään TLS-protokollan mukainen salaus) ja se, kuinka luokitella suojausmekanismit. Mallia ei ole tarkoitettu tähän tarkoitukseen, mutta sitä voidaan hyödyntää suunnittelussa, jos hyväksytään ennalta mainittujen ongelmien relaksoinnit.

Mitkä kerrokset kykenevät suojaamaan hyökkäystä vastaan, riippuu olennaisesti hyökkäyksestä. Kappaleessa 3.5 esitettiin erilaisia yhteyspisteitä, pääsyjä, joita pitkin hyökkääjä voi suorittaa hyökkäyksen. Yhteyspisteet olennaisesti määrittelevät sen, minkälaisen polun hyökkäys voi ottaa tietojärjestelmään. Hyökkäyksen polku määrittelee sen, mitkä kerrokset suojaavat hyökkäystä vastaan. Hyökkäys, joka hyödyntää sosiaalisen verkkosovelluksen web-käyttöliittymää, voi näennäisesti ilmentyä sovellukselle tavanomaisina käyttöpyyntöinä.

Tällöin hyökkäyksen käyttöpyynnöt ovat tietoliikennettä, jota ei estetä tietojärjestelmän kommunikaatio- ja kuljetuskerroksella. Hyökkäys ohittaa kommunikaatio- ja kuljetuskerroksella olevat suojaukset, mutta hyökkäys voidaan silti estää sovelluskerroksella.

Tietoturvan arviointi on prosessi, jossa pyritään saamaan varmistus sille, että paikoillaan olevat suojausmekanismit toimivat tietoturvavaatimusten mukaisesti [61, s35]. Tietoturvatestausta on suositeltavaa suorittaa jokaisessa sovelluskehityksen vaiheessa, mikäli vain mahdollista [70]. On useita tapoja arvioida sovelluksen tietoturvaa. Ulkopuolisten tietoturva-asiantuntijoiden palkkaaminen voi tuoda varmuutta, mutta voi olla kallis vaihtoehto. Koodikatselmoinnit, organisaation sisäiset tai ulkopuolisten suorittamat, voivat löytää vikoja, joita tietoturvan arviointityökalut eivät löydä. Koodikatselmoinnin varjopuolena on arvionnin hitaus. **Tunkeutumistestauksen** (engl. *Penetration testing*) [48, s9] tarkoituksena on simuloida hyökkäyksiä tietojärjestelmään ja oppia löytämään tällä tavoin tietojärjestelmän haavoittuvuuksia. Tunkeutumistestaus antaa sitä parempia tuloksia, mitä todenmukaisemmassa ja valmiimmassa ympäristössä se voidaan suorittaa. Lukuisia verkkosovellusten tietoturvan arviointityökaluja on käytettävissä sovelluskehityksen eri vaiheisiin [70]. Koska laaja-alaiset testaukset eivät ikinä voi löytää kaikkia haavoittuvuuksia, on suositeltavaa laatia tietojärjestelmälle toipumissuunnitelma [61, s20], joka *"määrittelee keinot toiminnan keskeytyksen minimoimiseksi ja palauttamiseksi normaaliksi"*.

3.6.3 Tietoturvan säilyttävä toteutus

Syvyyssuuntaisen turvallisuusajattelun toteuttamiseksi on suositeltavaa jakaa tietojärjestelmä vyöhykkeisiin [71, s11]. Jakaminen voidaan tehdä kullakin kerroksella erikseen. Sosiaalisen kerroksen vyöhykkeet edustavat esim. maineeseen perustuvia luottamusjärjestelmiä. Sosiaalisen kerroksen vyöhykkeet voivat olla erittäin hienojakoisia ja lukuisia. Tavanomainen ratkaisu on kuitenkin käsitellä kaikkia käyttäjiä tasapuolisesti, jolloin sosiaalisen kerroksen ulottuvuus häviää vyöhykejaottelusta. Sovelluskerroksen vyöhykkeen muodostaa tietojärjestelmän ohjelmistoarkkitehtuuri. Kommunikaatio- ja kuljetuskerroksella tavallisia vyöhykkeitä ovat **omaverkko** (engl. *intranet*), asiakkaille tarkoitettu vierasverkko ja **harmaa vyöhyke** (engl. *demilitarized zone*), joka on yhteydessä ulkoiseen verkkoon. Käytettäessä Cutillon ym. [50] mallia jaottelussa, tulee huomata kuinka, vyöhykkeissä esiintyy tiettyä päällekkäisyyttä. Esimerkiksi taustajärjestelmässä olevan palvelimen data, joka esittää käyttäjien profiileja, kuuluu sosiaaliseen kerrokseen, mutta koska palvelin on osa

tietoverkkoa, kuuluu se myös kommunikaatio- ja kuljetuskerrokseen. Ideaalisessa tilanteessa vyöhykejako voidaan ottaa suoraan ohjelmiston arkkitehtuurista tai tietoverkon topologiasta.

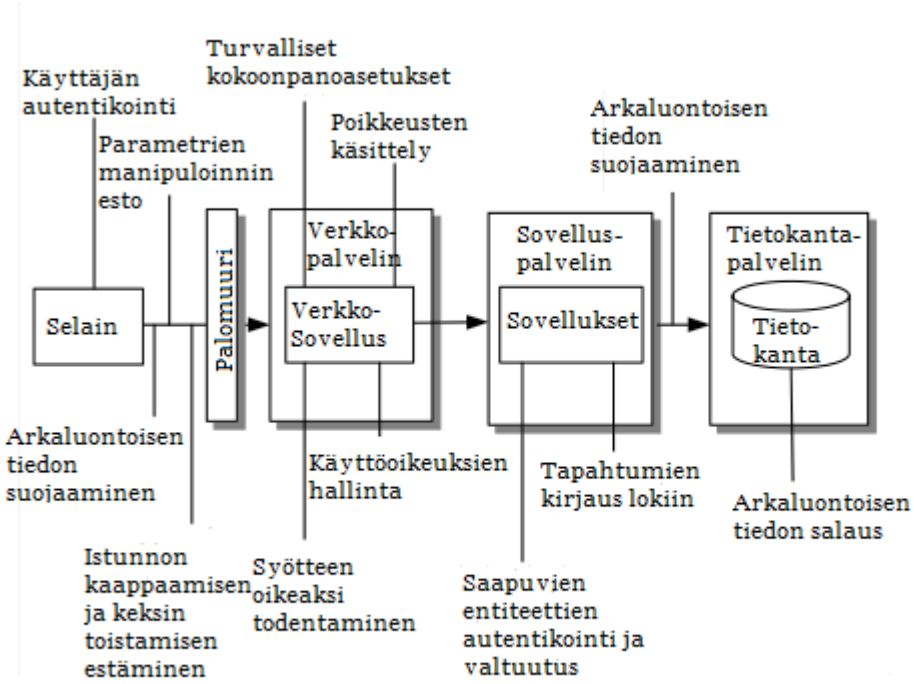
Tietojärjestelmän vaatimukset toimivat ohjeistuksena, kuinka hienojakoisesti vyöhykkeet määritetään ja mukautetaan. Vyöhykkeen vaihtuessa voivat tietoturvallisuusvaatimukset muuttua. Tärkeää on eristää vyöhykkeet toisistaan ja määrätä kullekin vyöhykkeelle täsmälliset pääsynvalvonnalliset rajoitteet. Rajaamalla tietovuot yhteyspisteisiin jotka yhdistävät vyöhykkeitä, voidaan vyöhykkeiden välille rakentaa kerrostetusti suojauksia.

Sosiaalisen kerroksen vyöhykkeissä käyttäjäsolmun sijainti ei implikoi sen potentiaalista vaarallisuutta. Sovelluskerroksen ja kommunikaatio- ja kuljetuskerroksen ulkoreunalla olevat vyöhykkeet tarvitsevat erilaisia suojauksia kuin taustajärjestelmät. Suurin osa tiedonsiirrosta kohdistuu näihin vyöhykkeisiin. Lisäksi niiden suora yhteys ulkopuoliseen verkkoon asettaa erityisiä vaatimuksia näille vyöhykkeille. Tällaisia vyöhykkeitä ovat edustajärjestelmät kuten palvelimet ja tietoverkon reunalla olevat verkkolaitteet (kuvassa 4.2 ”Verkkopalvelin” ja ”Palomuuri”). Sovelluskerroksella ulkoreunalla olevalla vyöhykkeellä tavanomaisia suojauksia ovat pääsynvalvonta, syötteen kelvolliseksi todentaminen ja poikkeusten käsittely [71, s71]. Kommunikaatio- ja kuljetuskerroksen suojauksia ovat arkaluontoisen tiedon salaus, tunkeutumisenestojärjestelmät, palomuurit, sisempien vyöhykkeiden suojaus NAT-protokollalla ja lokitietojen kirjaaminen [49, s82-83].

Kuvassa 4.2 järjestelmän ulkopuolinen toimija on selain, mikä ei ole ainoa yhteyspiste tietojärjestelmään. Sosiaalinen verkkosovellus voi tarjota, tai käyttää järjestelmän ulkopuolisia, palveluita sovellusliittymien kautta. Sovellusliittymien hallinnoimiseen kannattaa muodostaa yhtenäinen, formaali tapa, jolla sovellusliittymiä voidaan hallitusti integroida olemassaolevaan arkkitehtuuriin. Sovellusliittymät voivat vaatia useamman vyöhykkeen. Vyöhykejakoa tehtäessä on tarpeellista selvittää, mitä funktioita ja mitä tietoja sovellusliittymä välttämättä tarvitsee. Selvittämällä pienohjelmien käyttämät resurssit ja tietojärjestelmän riippuvaisuus sovellusliittymään, voidaan varautua mahdollisiin liittymän väärinkäyttötapauksiin.

Käyttäjille näkymättömiä yhteyspisteitä ovat ylläpidolliset yhteyspisteet [71, s86]. Tällaisilla yhteyspisteillä on tavallisesti laajemmat käyttöoikeudet ja voivat vaarantuessaan aiheuttaa paljon tuhoa tietojärjestelmälle. Erityistä huolta tulee kiinnittää kehittäjien ”testausrajapintoihin”, joilla kehittäjät testaavat uusia, vielä kehitteillä olevia toiminnallisuuksia.

Epäsuorat yhteyspisteet tietojärjestelmään kuten organisaation sähköpostipalvelut ja etäyhteydet voivat vaarantua, ja muodostua uhkaksi tietojärjestelmälle. Roska-



Kuva 3.2: Tietojärjestelmän jakaminen vyöhykkeisiin ja suojausten toteutus kerrostetusti.

postinsuodatus, virustentorjuntaohjelmistot, työntekijöiden valistaminen ja etäyhteyksien salaus vähentävät näiden tietoturvariskien mahdollisuutta.

Selain toimii rajapintana käyttäjän ja sosiaalisen verkkosovelluksen edustajärjestelmän välillä. On useita syitä, minkä takia selainta ei voida pitää luotettavan syötteen lähteenä. Selain välittää HTTP-protokollan otsaketiedoissa pyyntöjä ja vastauksia käyttäjän ja järjestelmän välillä. Hyökkääjä voi manipuloida HTTP-protokollan otsaketietoja [72, s3]. HTML-merkkauskieltä käsitellään eri tavoin eri valmistajien selaimissa [54, s474]. Tämä voi aiheuttaa ristiriitatilanteita, jotka voivat aiheuttaa tietoturvan vaarantumisen. Selaimien liitännäiset voivat ohittaa tietoturvamekanismit tai tarjota haavoittuvaisuuksia, joita on helppo hyödyntää [20]. Lisäksi selaimet voivat sisältää omia haavoittuvuuksia, joita hyökkääjä voi hyödyntää.

Kriittisen datan eheys ja luottamuksellisuus tulee varmistaa käyttämällä salausmenetelmiä. Hyvä käytäntö on suojata käyttäjien ja asiakkaiden luottamukselliset tiedot sekä kaikki arkaluontoista tietoa sisältävä kommunikaatio, kuten avaimia tai tunnisteita sisältävät viestit. Salausmenetelmän valinnassa tulee kiinnittää huomiota datan kriittisyyteen. Liian tehokkaat menetelmät haittaavat järjestelmän suoritustehoa. Jotkut menetelmät voivat olla vanhentuneita tai epäluotettavia. Suositeltavaa

on käyttää ei-omisteisia standardimenetelmiä [73]. Salausmenetelmiä voidaan käyttää myös tapahtumien kiistämättömyyden todentamiseen [73]. Joissakin tapauksissa lokit voivat riittää, joskin merkittäviin toimintoihin kannattaa rakentaa vahva mekanismi kiistämättömyyden osoittamiseksi. Esimerkiksi tällainen toiminto voisi olla rahalla ostettava virtuaalinen hyödyke.

Salausmenetelmät eivät voi taata syötteiden vaarattomuutta. Tärkeää on kyetä erottamaan varsinainen syöte järjestelmän sisäisistä komentokäskyistä. Jos syöte tulee epäluotettavasta lähteestä, kuten käyttäjältä, tulee se osoittaa kelvolliseksi. Syötteen kelvolliseksi osoittamista ei tule pelkästään jättää sovelluskerroksen ulkoreunalla oleville komponenteille [71, s74]. Monimutkaisessa arkkitehtuurissa voi olla vaikea erottaa luotettava lähde epäluotettavasta lähteestä. Epäselvissä tilanteissa on parasta pitää lähdettä epäluotettavana. Luotettavana lähteenä voi pitää taustajärjestelmässä olevaa tietokantaa, johon tietojärjestelmän ulkopuolelta kenellekään ei ole kirjoitusoikeutta.

Kaikkien tuntemattomien entiteettien identiteetit tulee varmentaa. Entiteetti voi olla muu kuin käyttäjä, esimerkiksi itsenäinen prosessi joka pyytää valtuutusta resursseihin. Ei ole välttämätöntä suojata tietojärjestelmän kaikkia palveluita autentikoinnilla. Ei-kriittiset palvelut, kuten uutissyötteiden tarjoaminen, voidaan suojata eristämällä tämä yhteyspiste järjestelmän muista osista luomalla tälle oma vyöhyke.

Kaikki luottamuksellisiin resursseihin kohdistuvat pyynnöt tulee valtuuttaa. Tavallisesti sosiaalisissa verkkosovelluksissa käyttäjän profiili määrittelee käyttöoikeudet muiden entiteettien sisältöön [53, s578]. Sosiaaliset verkkosovellukset voivat sisältää lukuisia eri entiteettejä, jossa jokaisella on omanlaiset tarpeensa. Käyttäjäroolien hyvä suunnittelu ehkäisee potentiaalisten tietoturvariskien syntymistä.

Joissakin tapauksissa käyttöoikeudet ovat selviä. Esimerkiksi käyttäjällä ei tulisi missään tapauksessa olla käyttöoikeuksia järjestelmätason resursseihin. Hyvä käytäntö on toimia **pienimmän etuoikeuden periaatteen** (engl. *principle of least privilege*) mukaisesti [47, s43]. Periaate määrää, että valtuutusta vaativa entiteetti valtuutetaan vain niillä käyttöoikeuksilla, jotka ovat täysin välttämättömiä ko. entiteetin funktion suorittamiseksi. Suunnittelun aikana käyttäjäroolin, jota kyseinen entiteetti edustaa, käyttöoikeuksia voidaan nostaa hiljalleen, kunnes olio voi suorittaa funktionsa. Näin rooli ei sisällä ylimääräisiä käyttöoikeuksia, jota voitaisiin hyödyntää hyökkäyksessä.

Esimerkiksi kolmannen osapuolen tarjoamalla työkalulla, jolla muokataan omia valokuvia, ei tulisi olla oikeutta kuin niihin käyttäjän tietoihin, mitkä ovat välttämättömiä työkalun toiminnalle. Jos työkalussa muokataan kuvia yksi kerrallaan,

pienimmän etuoikeuden periaatteen mukaisesti pienohjelmalle annetaan käyttöoikeudet kuviin yksi kerrallaan.

Helppokäyttöisyys on tärkeä ei-toiminnallinen ominaisuus sosiaalisille verkkosovelluksille. Kappaleessa 3.4.1 huomioitiin käyttäjien vastuulla olevat tietoturvamekanismit, joiden tulee oleellisesti olla helppokäyttöisiä. Joidenkin toiminnallisuuksien ollessa liian helppokäyttöisiä, vaarantuu sovelluksen tietoturva. Käyttäjien salasanojen vahvuus on tällainen toiminnallisuus. Huomiota kannattaa kiinnittää myös mekanismeihin, jolla salasanan voi saada käsiinsä sitä tietämättä (salainen kysymys tai automaattinen salasanan toimitus sähköpostiin). [74] Tärkeää on pitää käyttäjät informoituna. Käyttäjien valistus, selkeät ohjeet ja poikkeustilanteissa oikeanlainen informaatio auttavat ehkäisemään monia orastavia tietoturvariskejä.

Tietojärjestelmän suojelemiseksi tulee tarkastaa ulospäin näkyvä informaatio — kaikilla kerroksilla ja vyöhykkeillä—. Mitä tietämättömämpi hyökkääjä on tietojärjestelmän sisäisestä tilasta, konfiguraatiosta ja arkkitehtuurista, sitä paremmassa suojassa järjestelmä on [59]. Erityisesti tulee huomioida auki olevat palvelut ja portit, käytössä olevat protokollat, laitteisto ja niihin asennetut mikro-ohjelmistot, käyttöjärjestelmät ja virhetilanteet.

Virhetilanteet voivat olla vaaraksi tietojärjestelmän saatavuudelle. On suositeltavaa jäsentää selkeitä virhetiloja ja määrittää niiden seuraamukset sovelluksen toiminnalle. Virhetilanteet on tarpeellista testata, jotta ne toimivat odotetulla tavalla. Vikaantumiset voivat vuotaa tietoa tietojärjestelmän sisäisestä toimintalogiikasta tai asiakkaiden (käyttäjät, kolmannen osapuolen kehittäjät) dataa. Jos järjestelmälle on luotu toipumissuunnitelma (ks. kpl 3.6.2), selviää siitä kuinka toimia virhetilanteissa.

4 Sosiaalisten verkkosovellusten tietoturvariskit ja niiltä suojautuminen

Tämä luku on järjestelty seuraavasti: kappaleessa 5.1 tutustutaan mistä tekijöistä merkittävimmät tietoturvariskit koostuvat. Kappaleessa 5.2 esitellään sosiaalisille medioille tyypillisiä tietoturvariskejä. Luvun loppu on varattu toimenpiteisiin, joilla käyttäjä voi suojata itseään.

4.1 Sosiaalisten verkkosovellusten olennaisimmat tietoturvariskit

Sosiaalisten verkkosovellusten olennaisimmat tietoturvariskit muodostuvat seuraavista lähtökohdista:

1. Arkaluontoiset tiedot
2. Luottamuksen ongelma
3. Sisällöntuotanto
4. Pienen maailman verkosto
5. Verkkosovellusten heikko tietoturva

4.1.1 Arkaluontoiset tiedot

Sosiaaliset verkkosovellukset sisältävät valtavia määriä arkaluontoista tietoa. Koska tieto tallennetaan asiakas-palvelin mallin mukaisesti [14], tieto on keskitetysti yhdessä paikassa. Tämä lisää hyökkääjien kiinnostusta merkittävästi. Keskitetty rakenne, yhdessä sosiaalisen kerroksen hakutyökalujen kanssa, helpottaa sosiaalisen verkkosovelluksen tiedonlouhintaa.

Yksityisyyden paradoksi (ks. kpl 2.4.1) esittää, että käyttäjät ovat huolestuneita yksityisyydestään, mutta silti jakavat tietojaan auliimmin ympäristössä, jossa ei voi olla varma vastakumppanin identiteetistä. Tilannetta pahentaa se, että useimmat käyttäjät eivät ole tietoisia kaikista vaaroista, joille he tulevat alttiiksi, kun heitä koskeva arkaluontoinen tieto tai sisältö julkistetaan (ks. kpl 2.4.1).

Yksityisyyden käsitteen subjektiivisuuden (ks. kpl 2.4.1) vuoksi kehittäjät antavat käyttäjien itsensä päättää mikä heidän julkaisemansa tieto heille on yksityistä. Käyttäjät hallitsevat kehittäjien luomissa puitteissa ja kehittäjien työkaluilla yksityisyysasetuksiaan. Monet uudet käyttäjät eivät vaihda yksityisyysasetuksien oletusarvoja [39]. Mikään ei estä jotain muuta käyttäjää julkaisemasta arkaluontoista sisältöä toisesta käyttäjästä [75], esimerkiksi sellaisen käyttäjän, jolla on käyttöoikeudet olemassa ja täten pääsy arkaluontoiseen sisältöön.

Mitä enemmän käyttäjät tallentavat dataansa muualle kuin omaan koneeseensa, sitä vähemmän heillä on kontrollia siihen, kuinka turvata se. Toisin sanoen, heidän datansa suojauksen järjestäminen on kehittäjien käsissä. Tärkeää on selvittää kenellä on vastuu arkaluontoisen tiedon vuodettua. Mitä enemmän vastuullisuutta siirretään kehittäjille, sitä halukkaampia he ovat toteuttamaan tietoturvamekanismeja, jotka suojelevat käyttäjien yksityisyyttä, jotta käyttäjien luottamus säilyisi ja kehittäjät välttyisivät lakikiistoilta. Toisaalta, kehittäjät eivät voi olla täysin vastuussa kaikesta siitä, mitä käyttäjät tekevät. Niin kauan kun vastuuta ei selvästi määrätä kenellekään, kukaan tuskin ottaa sitä. [76]

Arkaluontoisten tietojen suojelemisen tärkeys antaa enemmän painoarvoa hyvin toteutetulle pääsynvalvonnalle. Autentikaatio, niin prosessien kuin käyttäjien, ja käyttöoikeuksien valtuuttaminen tulee toteuttaa huolellisesti ja tiukoin, mahdollisimman täsmällisin säännöin. Käyttäjien ohjeistaminen yksityisyyteen liittyvistä ongelmakohdista toimii ennaltaehkäisevänä toimenpiteenä.

4.1.2 Luottamuksen ongelma

Sosiotekniset käyttäjien manipulointitavat hyödyntävät erityisesti luottamuksen ongelmaa ja näitä hyödyntävien hyökkäysten torjunta tarvitsee teknisten suojamekanismien lisäksi ei-teknisiä toimenpiteitä. Sen lisäksi, mitä linkkejä käyttäjät klikkailevat, on heillä paljon valtaa sisällöntuotantoon ja omien yksityisyysasetuksien hallinnoimiseen sekä käyttöoikeuksia muihin resursseihin. Tästä syystä on usein helpompaa valjastaa käyttäjä palvelemaan hyökkääjää tämän tarkoitusperissä, kuin yrittää murtaa jokaista vahvaa suojauskerrosta. Sosiotekninen manipulointitapa ohittaa useita, vältettävässä tapauksessa kaikki, sovelluksen suojausmekanismit.

Sosiaalisissa verkkosovelluksissa pitäisi puhua ennemmin profiilin autentikoinnista kuin identiteetin autentikoinnista (ks. kpl 2.5). Profiili pystytään luomaan vaivattomasti toisen henkilön identiteettiä käyttäen [40]. Esittämällä jotain toista, voidaan hyödyntää entisestään olemassaolevaa luottamuksen ongelmaa. Tätä kutsutaan identiteettivarkaudeksi.

Luottamuksen ongelmaa voidaan hallita, tai ainakin sen vaikutusta riskeihin pienentää, ottamalla käyttöön maineeseen perustuvia luottamusjärjestelmiä. Näiden tarkempi käsittely sivuutetaan tässä. Informaatiolla voidaan ehkäistä tietoturvariskejä. Tällaisia ovat hyvät ohjeet, valistaminen ja poikkeustilanteista tiedottaminen.

4.1.3 Sisällöntuotanto

Sisältö voi tulla lukuisista eri lähteistä ja se voi olla hyvin monimuotoista. Jokainen eri tiedon esitysmuoto voi vaatia omanlaisiaan suojauksia. Datan erottaminen järjestelmän sisäisistä komentokäskyistä on erittäin tärkeää. Kaiken sisällön oikeanlaiseksi todentaminen on mittava tehtävä. Sisällöntuotannollisia toimintoja tulee kontrolloida, jotta niitä ei voida väärinkäyttää, esimerkiksi yrittämällä kaataa järjestelmä resurssien loppumiseen.

Sisällöntuotannossa tulee huolehtia resurssien riittävyydestä, jotta tietojärjestelmän eheys ja saatavuus ei vaarannu. Tietoturvalliseen resurssienhallintaan kuuluu myös virhetilanteiden huomioon ottaminen. Esimerkiksi tietokantojen peilaamisella voidaan ehkäistä laitteiston vikaantumisesta johtuvia tietoturvariskejä.

Mikä tahansa entiteetti, jolla on käytössään profiili, voi luoda haitallista sisältöä, mukaan lukien automatisoidut skriptit tai haitakkeet. Ilman vahvaa autentikointimenetelmää sosiaalisten verkkosovellusten on mahdotonta tietää, onko profiili tekaistu tai onko skripti luonut profiilin (ks. kpl 2.5). Sovelluksen omilla sosiaalisen kerroksen toiminnoilla haitallinen sisältö voidaan luoda ja lähettää edespäin.

Sovellusliittymien lukuisat eri käyttötavat voivat mahdollistaa ennalta näkemättömiä haavoittuvuuksia. Huonosti integroidut sovellusliittymät voivat vuotaa käyttäjien tietoa tai näihin tietoihin käsiksi pääsevä kolmannen osapuolen sovelluskehittäjä voi väärinkäyttää asemaansa, esimerkiksi myydä käyttäjän tietoja eteenpäin.

Sisällöntuotanto tuo mukanaan sisällön rajoittamiseen ja hallitsemiseen liittyvät tarpeet. Edelleen, autentikaatio ja käyttöoikeuksien valtuuttaminen ovat tärkeitä ominaisuuksia sisällön suojelemisessa.

4.1.4 Pienen maailman verkosto

Käyttäjien muodostama verkosto on pienen maailman verkosto (ks. kpl 2.3). Tämän verkoston ominaisuudet tekevät sosiaalisista verkkosovelluksista ihanteellisia kohteita roskapostille ja haitakkeille. Pienen maailman verkoston ominaisuudet helpottavat tiedonlouhimista tietojärjestelmästä. Samat ominaisuudet yhdessä luottamuksen ongelman kanssa mahdollistavat räjähdysmaisen nopean propagaa-

tion haitakkeille ja roskapostille. Roskaposti voidaan lähettää tavanomaisia kanavia myöten, käyttäen vain sosiaalista verkkosovellusta sähköpostiosoitteiden kartuttamiseen. Sosiaalisesta verkostosta ilmenevät suhteet voivat edelleen epäsuorasti paljastaa arkaluontoista tietoa.

Identiteetin hallintaan pohjautuvissa sosiaalisissa medioissa käyttäjän verkossa näkyvä digitaalinen sosiaalinen verkosto kuvaa —jossain määrin— käyttäjän yhteydetöntä, reaalista, sosiaalista verkostoa. Täten käyttäjän digitaalinen sosiaalinen verkosto —edelleen jossain määrin— on pienen maailman verkosto, niin kauan kun yhteydetön, reaalinen sosiaalinen verkosto on pienen maailman verkosto. Pienen maailman verkostojen ominaisuuksien tuomia haasteita voidaan lieventää valvomalla hyvin vahvasti kytkeytyneitä keskussolmuja tai hallitsemalla verkostojen muodostumista kehittäjien toiveiden mukaisesti. Jälkimmäinen vaihtoehto luultavasti rajoittaa liikaa sovelluksen käyttöä ja tekee siitä mahdottoman toteuttaa.

4.1.5 Verkkosovellusten heikko tietoturva

Kappaleissa 3.4.2 ja 3.6.2 käsiteltiin verkkosovelluksille ominaisia piirteitä, jotka vaikeuttavat tietoturvan toteuttamista näihin järjestelmiin. Erityisesti verkkosovellusten nopea sovelluskehitys ja verkkosovelluksille tarkoitettujen suunnittelumenetelmien puuttuminen heikentävät ohjelmistojen laaduntasoa, jättäen ohjelmistoihin ohjelmistovirheitä, jotka voivat toimia haavoittuvuuksina [77] [78]. Lisäksi laaduntasoa laskee keskiverto verkkosovelluskehittäjän heikko osaamisen taso [77]. Usein ohjelmistovirheitä pidetään perimmäisenä syynä, mikä mahdollistaa uhkan toteutumisen [78].

Verkkosovellusten ohjelmistovirheisiin tulee kiinnittää erityistä huomiota, varsinkin kun sosiaaliset verkkosovellukset omaksuvat yhä useampia toimintoja ja palveluita [20]. Ohjelmistojen testaaminen varmentaa vain siellä olevat ohjelmistovirheet, oikea lähestymistapa onkin omaksua tietojärjestelmää tukeva suunnittelumetodologia [77]. Pitäytymällä oikeassa suunnittelumetodologiassa, joka integroi tietoturvan sovelluskehitykseen, vähennetään huomattavasti potentiaalisten haavoittuvuuksien lukumäärää.

Tehostetun laadunvarmistuksen lisäksi kannattaa omaksua nopea päivitystahti, mielellään automatisoitu. Sovelluksen kaikkien komponenttien lisäksi kannattaa huolellisesti päivittää kommunikaatio- ja kuljetuskerroksen komponentit ja tarkistaa näiden konfiguraatio.

4.2 Sosiaalisille verkkosovelluksille ainutlaatuiset tietoturvariskit

Tässä luvussa käydään lyhyesti läpi joitain sosiaalisille verkkosovelluksille ominaisia tietoturvariskejä. On sovelluskohtaista mitkä seuraavista tietoturvariskeistä ovat vakavia, huomioitavia tai mitättömiä.

4.2.1 Heikko identiteetin suoja

Cutillon ym. [50] mukaan esiintyminen toisena henkilönä rikkoo uhrin identiteetin eheyttä (ks. kpl 3.4). Koska sosiaalisissa verkkosovelluksissa käyttäjän identiteetti sidotaan tämän profiiliin, kaikki toisen henkilön profiilin väärinkäytökset ovat tietoturvauhkia. Käyttäjän identiteetin sitominen profiiliin rajoittaa kunkin käyttäjän profiilien lukumäärän yhteen.

Käyttäjän sisäänkirjautuessa käyttäjä autentikoidaan ja käyttäjälle luodaan istunto, joka on voimassa niin kauan kun käyttäjä on sisäänkirjautuneena. HTTP-protokolla on tilaton protokolla, jolloin istunnon hallinnoiminen jää sovelluskerrokselle toteutettavaksi. Kun käyttäjä on autentikoitu käyttäjätunnuksen ja salasanan avulla, saa hän istuntotunnisteen. Istuntotunnistetta käytetään aina, kun välitetään pyyntöjä tai vastauksia käyttäjän ja järjestelmän välillä [74].

Hyökkääjä voi kaapata istuntotunnisteen tai voi pystyä arvaamaan sen. Tarpeellista on suojata istuntotunnisteet ja tehdä niistä ennalta arvaamattomia [72, s2]. Sisäänkirjautumissivu ja kaikki tätä seuraavat sivut tulee turvata TLS-salausprotokollalla, jotta hyökkääjä ei pääse käsiksi istuntotunnisteeseen [74]. Jos hyökkääjä kykenee arvaamaan istuntotunnisteen, ei salausprotokollista ole mitään hyötyä. Tämän takia istuntotunnisteen tulee olla tarpeeksi ennalta arvaamaton. Tämä voidaan saavuttaa käyttämällä tarpeeksi vahvoja (vähintään 128-bittinen), pseudosatunnaisia ja suuren merkistön omaavia istuntotunnisteita [74]. Profiilin tuhoaminen, sisällöntuotanto ja muut tärkeät toiminnot voidaan suojata edelleen syvyyssuuntaisen turvallisuusajattelun mukaisesti pyytämällä käyttäjätunnus ja salasana uudelleen.

Edellä mainitut suojausmekanismit eivät kuitenkaan estä hyökkääjää luomasta profiilia toisen henkilön nimellä. Kappaleessa 2.5 huomioitiin, kuinka sosiaalisissa verkkosovelluksissa ei ole tarpeeksi vahvaa käyttäjät yksilöivää tunnistautumismekanismia. Tämä heikko identiteetin suoja mahdollistaa monet tietoturvauhkat [50]. Heikon identiteetin suojaa hyödyntäviä hyökkäyksiä ovat mm. istunnon kaappaus, **istunnon kiinnittäminen** (engl. *session fixation*)[82, s28] ja identiteettivarkaus.

Vahvan tunnistautumismekanismin käyttöönotto vaatii käyttäjien hyväksynnän. Koska sosiaalisissa verkkosovelluksissa suurin osa käyttäjien toimista tapahtuu sisäänkirjautuneena, voi tarpeeksi vahvan tunnistautumismekanismin käyttöönotto

pelotella joitain käyttäjiä pois. Sisäänkirjautuminen tulisi yhä olla melko vaivatonta. Kompromissiratkaisuna voisi olla tiettyjen profiilien vapaaehtoinen vahva tunnistautuminen. Käyttäjät voisivat todeta, että juuri ko. profiili kuuluu varmasti kyseiselle henkilölle. Tällainen ratkaisu vähentää suosittujen henkilöiden identiteetin väärinkäyttämistä tehokkaasti.

Kehittäjät voivat suorittaa ajoittaisia tarkastuksia suosittuihin keskussolmuihin. Tavallisesti tämä tarkoittaa identiteettivarkauksien kitkemistä, eli aidon profiilin tunnistamista. Heikon identiteetin suojan takia henkilön aitoudesta voidaan varmistua ottamalla yhteyttä asianomaiseen tai tämän läheiseen tuttavapiiriin. Edelleen, kehittäjät voivat hyödyntää käyttäjiä vähentämään väärinkäytösten lukumäärää tarjoamalla käyttäjille kanavia tai mekanismeja väärinkäytösten ilmoittamiseksi.

Väärinkäytösilmoitusten ja kehittäjien omien tarkastusten lisäksi salasana ovat paikoillaan olevia identiteettiä suojaavia tietoturvamekanismeja. Kehittäjien tulee vaatia käyttäjältä tarpeeksi vahvaa salasanaa ja ohjeistaa hyvistä käytännöistä salasanojen suhteen. Esimerkiksi saman salasanan käyttöä useammassa paikassa tulisi välttää.

4.2.2 Sovellusliittymät

Sovellusliittymät voivat tarjota hyvin monimutkaisia, vaihtelevia ja hyödyllisiä toimintoja käyttäjille. Kun pienohjelmia integroidaan järjestelmään on tarpeen selvittää kunkin pienohjelman funktio selvästi. Jos sovellusliittymiä tarjotaan järjestelmään liian löyhin perustein, tämä avaa oven tietojärjestelmän väärinkäytöksille.

Optimaalisessa tilanteessa kukin sovellusliittymä tarjoaa vain ja ainoastaan ne resurssit ja palvelut, joita tarvitaan suorituksessa olevan tehtävän suorittamiseksi. Jokainen pienohjelma tulisi suorittaa suljetussa, hallitussa ympäristössä. Ympäristö rajoittaa pienohjelmaa suorittamasta sille kuulumattomia toimintoja, kuten yhteydenottoa ulkopuolisen verkon laitteisiin. [80]

Pienohjelmien vaikutusta verkkosovelluksen normaaliin toimintaan on hyvä testata. Lukuisat ajossa olevat pienohjelmat voivat kuluttaa tavallista enemmän resursseja tai aiheuttaa arvaamattomia vikaantumistilanteita. Kehittäjien on syytä huomioida puutteellisesti ohjelmoidut pienohjelmat. On hyvä selvittää mitä tapahtuu vikaantumistilanteessa. Sovellusliittymiä, joiden käyttämiä resursseja ei valvota, voidaan valjastaa palvelunestohyökkäyksiin [59, s650]. Lisäksi sovellusliittymät voivat epäsuorasti tuoda tietojärjestelmään haavoittuvuuksia.

Käyttäjien arkaluontoiset tiedot ovat arvokas resurssi monelle taholle. On tarpeellista luoda vahvat käytännöt tietojen suhteen sekä toteuttaa tarpeeksi vahvat

tietoturvamekanismit näiden suojelemiseksi. Pienimmän etuoikeuden periaatetta kannattaa soveltaa sovellusliittymien valtuutukseen. Selvittämällä sovellusliittymän tarvitsemat tietotyypit ja kuinka kommunikointi rajapinnan välillä käy, voidaan pienimmän etuoikeuden periaattetta soveltaa. Tietojärjestelmän ei tule sokeasti myöntää pienohjelmalle tietoja käyttäjästä, jota tämä ei tarvitse oman funktionsa suorittamiseksi. Prosessien turvallinen autentikointi ja valtuutus voidaan vahvistaa mm. kaksipisteyhteyksillä, tunneloimisella [49, s77], IPSecilla [49, s118] tai esimerkiksi erityisesti sovellusliittymien suojattuun hallintaan liittyvällä avoimen lähdekoodin OAuth-protokollalla [91].

4.2.3 Kohdennetut hyökkäykset

Kohdennetut hyökkäykset ovat hyökkäyksiä jotka sovitetaan käyttäjä- tai ryhmäkohtaisesti hyväksikäyttämällä kohteen digitaalista jalanjälkeä. Sosiaalinen verkkosovellus itse voi muodostaa merkittävän osan käyttäjän digitaalisesta jalanjäljestä. Tiedonlouhinta on tehokkain tapa hyökkääjien kerätä tietoa käyttäjistä. On havaittavissa, että kohdennetut hyökkäykset tulevat jatkossa lisääntymään [83].

Uhrin sosiaalisen verkoston läpikäymisen estämisellä voidaan rajoittaa tiedonlouhintaa uhrista. Käyttäjän ystävistä voitaisiin suodattaa yksilöiviä tietoja pois, erityisesti sukunimi. Tämä ei kykenisi estämään älykkäämpiä hyökkäyksiä, jotka hyödyntävät heuristiikoita, joilla voidaan löytää muita suhteita käyttäjien välillä. Heuristiikka voi hyödyntää yhteisiä ystäviä ja kun tarpeeksi yhteneväisyyttä esiintyy, merkitä nämä profiilit ja louhia näistä tietoja. [86]

Tiedonlouhiminen ei aina ole kiellettyä. Kun nykyinen lainsäädäntö ei ota kantaa tarpeeksi selkeästi [76], sovelluksen käyttöehtosopimus antaa puitteet tiedonlouhimisen laillisuudelle. On kuitenkin epäselvää missä menee raja. Käyttöehtosopimuksessa voidaan kieltää automatisoidut tiedonkeruumenetelmät. Ongelmallista yhä on, kuinka tunnistaa suorittaako profiilien hakutoimintoja skripti vai ihminen [82, s74]. Erityisesti mainostajat ja hyökkääjät ovat kiinnostuneita sosiaalisissa verkkosovelluksissa olevasta datasta.

Yksityisyysasetuksia säätämällä — olettaen, että sellaisia on— käyttäjät voivat tehokkaasti estää kohdennettuja hyökkäyksiä. Käyttäjää koskevaa tietoa voi kuitenkin löytyä epäsuorasti muiden käyttäjien profiilisivuilta tai käyttäjän itsensä viesteistä. Eräs ratkaisu on älykäs syötteiden tarkistus, joka tunnistaa arkaluontoiset tiedot, esimerkiksi syntymäpäivät, ja merkitsee ne yksityisiksi. Ilman käyttäjien apua tiedon merkitsemisessä, tällaisen automaattisen syötteen tarkistuksen toteuttaminen vaikuttaa hankalalta. Kun käyttäjiä käytetään apuna tiedon merkitsemisessä, li-

sää se heidän vastuutaan ja yleisrasitetta. Huonosti hallittu toiminto voi olla uusi kanava väärinkäytöksille.

Mitä vaivattomampaa käyttäjien sosiaalisen verkoston läpikäyminen on, sitä helpompaa hyökkääjien on suorittaa kohdennettuja hyökkäyksiä ja tehdä tiedonlouhintaa. Yleisesti ottaen käyttäjien sosiaalisen verkoston läpikäymisen rajoittaminen voi olla joidenkin, hyvin avoimien, sosiaalisten verkkosovellusten funktiolle liian rajoittava ratkaisu.

4.2.4 Haitalliset linkit

Linkit ovat erittäin tärkeä piirre Webin toiminnalle. Sosiaalisissa verkkosovelluksissa on tavallista, että käyttäjät voivat viitata muualle verkkoon tarjoamalla linkin. Sovelluksen tulee kyetä erottamaan käyttäjän syötteestä mahdolliset hyperlinkit. Linkkiä lisättäessä on todettava linkin aitous, ts. se, että käyttäjän syöttämä merkkijono sisältää kelvollisen URL-osoitteen. URL-osoite, joka tarkastetaan puutteellisesti, voi sisältää koodi-injektiohyökkäyksen [82] (ks. kpl 5.2.4).

Haitallisille verkkosivuille johtavien linkkien erottaminen turvallisista linkeistä on hankala tehtävä. Sovelluksessa voidaan ylläpitää kiellettyjen verkkosivujen listaa, joka käydään läpi kutakin linkkiä lisättäessä. Jos merkkijonosta saatu URL-osoite viittaa kiellettyyn verkkosivuun, kyseisestä merkkijonosta ei muodosteta linkkiä. Tällaisten ratkaisujen ongelmana on usein skaalautuvuus. Edes hyvin usein päivitetty kiellettyjen verkkosivujen lista ei tule ikinä kattamaan kaikkia haitallisia verkkosivuja. Edelleen, jos kiellettyjen verkkosivujen listan päivityksessä hyödynnetään käyttäjien palautetta, voi lista sisältää virheellisin perustein lisättyjä verkkosivuja. Sovelluksessa voidaan ylläpitää myös mahdollisesti haitallisten verkkosivujen listaa, jolloin käyttäjää varoitetaan, kun tämä klikkaa epäilyttävää linkkiä.

Valveutuneet käyttäjät voivat havaita haitallisen linkin tarkkailemalla tämän osoitetta. Tästä syystä hyökkääjä yrittää saada haitalliset linkit näyttämään niin vaarattomilta kuin mahdollista. Jos mahdollista, hyökkääjä yrittää saada käyttöönsä domain-osoitteita, jotka muistuttavat luotettavia tahoja, saadakseen käyttäjän luottamuksen linkin vaarattomuudesta.

URL-lyhentimet ovat uudelleenohjausverkkopalveluita jotka korvaavat URL-osoitteita uusilla, lyhyillä verkkopalvelun hallinnoimilla uudelleenohjauslinkeillä. Syöttämällä URL-osoitteen verkkopalveluun saadaan uusi URL-lyhentimen tarjoama URL-osoite. Tämä uusi URL-osoite uudelleenohjaa alkuperäiseen annettuun verkko-osoitteeseen. Jotkin sovellukset, erityisesti sähköpostisovellukset, eivät osaa tulkita liian pitkiä URL-osoitteita kelvollisiksi linkeiksi. Hyökkääjä voi käyttää URL-

lyhennintä kätkemään haitallisen linkin verkko-osoitteen. URL-lyhentimillä muodostetut URL-osoitteet tekevät päättelyn URL-osoitteen haitallisuudesta mahdottomaksi niin käyttäjille kuin valvottujen verkkosivujen listaukseen perustuville ratkaisuille. [87] [79] Käyttäjien kannattaa suhtautua varovaisuudella epäluotettavasta lähteestä tuleviin hyperlinkkeihin, jotka osoittavat uudelleenohjauspalveluun.

4.2.5 Epäsosiaaliset verkostot

Athanasopoulos ym. [80] käyttävät nimitystä **epäsosiaaliset verkostot** (engl. *anti-social networks*) sosiaalisista verkostoista, joita hyväksikäytetään kyberrikollisuudessa tai muussa laittomassa toiminnassa. Epäsosiaalinen verkosto valjastetaan palvelemaan hyökkääjän tarkoitusperiä käyttäjien tietämättä, jolloin verkoston kautta ja usein tämän resursseilla suoritetaan edelleen hyökkäyksiä, kuten roskapostitusta, haitakkeiden levittämistä tai palvelunestohyökkäyksiä. Hyökkääjä väärinkäyttää käyttäjiä yhdistävää allaolevaa sovelluskerrosta suorittamaan hänen haluamiaan toimenpiteitä. Käytännössä epäsosiaalinen verkosto on hyökkääjän hallitsema orjakonejoukko.

Epäsosiaalinen verkosto voi olla vaarallinen hyökkääjän käsissä. Suuret epäsosiaaliset verkostot sisältävät paljon kapasiteettia monenlaisten hyökkäysten suorittamiseksi. Epäsosiaaliset verkostot eivät ole pelkästään vaarallisia sosiaalisen verkkosovelluksen ylläpitäjille. Roskapostia tai haitakkeita voi lähettää ja palvelunestohyökkäyksiä suorittaa sosiaalisen verkkosovelluksen ulkopuolelle. Hyökkääjän anonyymius on turvassa, sillä orjakoneiden käyttäminen kätkee tehokkaasti hyökkääjän todellisen identiteetin.

Athanasopoulos ym. tutkivat epäsosiaalisen verkoston muodostamista luomalla mielenkiintoiselta vaikuttavan pienohjelman Facebookiin [80]. Tavallisen toimintansa lisäksi pienohjelma lataa käyttäjälle näkymättömiä elementtejä kohdesivustolta, jolloin vaarattomalta vaikuttavan pienohjelman haittavaikutukset eivät ilmene käyttäjälle. Esimerkin ideana on tyrehdyttää kohdesivuston resurssit luomalla sille tarpeetonta tietoliikennettä. Toisin sanoen, epäsosiaalinen verkosto valjastettiin palvelunestohyökkäykseen (ks. kpl 5.2.8). Tutkimus osoitti, että on mahdollista luoda epäsosiaalinen verkosto verkkoyhteisöjen sisälle ja valjastaa se palvelemaan hyökkääjän tarkoitusperiä.

Suojellakseen epäsosiaalista verkostoa hyökkääjä pyrkii piilottamaan epäsosiaalisen verkoston olemassaolon. Tämä voi onnistua luomalla epäsosiaalisen verkoston toiminnasta vaikeasti havaittavaa ja rajoittamalla epäsosiaalisen verkoston käyttöä. Jos hyökkääjä käyttää epäsosiaalista verkostoa täysimittaiseen hyökkäykseen, ku-

ten hajautettuun palvelunestohyökkäykseen, epäsosiaalinen verkosto todennäköisesti paljastuu hyvin pian. Tämä johtaa sosiaalisen verkkosovelluksen ylläpitäjien osalta epäsosiaaliseen verkkoon kuuluvien käyttäjätilien jäädyttämiseen.

4.3 Käyttäjien ennaltaehkäisevät toimenpiteet

Tieto mahdollisista vaaroista motivoi ja auttaa käyttäjiä ottamaan ennaltaehkäiseviä toimenpiteitä. Koska sosiaaliset verkkosovellukset ovat usein jatkuvan kehityksen alla, käyttäjien arkaluontoiset tiedot ovat aina enemmän tai vähemmän alttiita hyökkäyksille [79]. Seuraavat käytännöt ovat yleisluontoisia ja pätevät kaikkiin sosiaalisiin verkkosovelluksiin. Käytännöt perustuvat osittain Euroopan verkko- ja tietoturvaviraston *(ENISA)* ja *Trend Micron*, tietoturvaratkaisuja tarjoavan yhdysvaltalaisen yrityksen, suosituksiin [65] [79].

1. Tutustu sosiaalisen verkkosovelluksen toimintoihin, jotta ymmärrät kuinka ne toimivat.
2. Säädä yksityisyysasetukset sinulle sopiviksi.
3. Julkaise vain sellaista tietoa, jonka paljastuminen ei vaivaa sinua.
4. Älä luota täysin henkilöihin, joita et tunne hyvin.
5. Lisää ystäviksesi vain henkilöitä, joihin luotat.
6. Vältä epäilyttävien linkkien klikkaamista, erityisesti, jos linkki tulee tuntemattomalta.
7. Käytä sosiaalisen verkkosovelluksen omia kanavia ja työkaluja roskapostittajien ja häiritsevien henkilöiden ilmoittamiseen.
8. On kohteliasta olla julkaisematta mahdollisesti arkaluontoista sisältöä muista käyttäjistä.
9. Luo vahva salasana: Helppo muistaa mutta vaikea arvata. Älä käytä samaa salasanaa useammassa tilissä.
10. Jos epäilet, että salasanasi on vuotanut, se kannattaa vaihtaa välittömästi.

Useimmat sosiaaliset verkkosovellukset mahdollistavat käyttäjien julkaista kuvia. Kuvat voivat paljastaa arkaluontoista tietoa, joita käyttäjä ei tule ajatelleeksi, esimerkiksi auton rekisterinumeron [65]. Sovellus voi tarjota mahdollisuuden jakaa

kokonaisia tiedostokansioita, jolloin on tarpeellista selvittää ettei tiedostokansio pidä sisällään arkaluontoista materiaalia.

Sosiaaliset verkkosovellukset sisältävät lähes poikkeuksetta toimintoja, jossa kanssakäyminen toisen osapuolen kanssa voidaan halutessa välttää täydellisesti. Häiritsevien ihmisten lisäksi näillä voidaan hillitä roskapostittajien viestitulvaa.

5 Hyökkäykset

Aluksi käydään läpi kuinka hyökkäyksiä käsitellään, minkälainen taksonomia hyökkäyksiä varten on valittu ja mitä muutoksia taksonomiaan tehdään. Tämän mukaisesti käsitellään yhdeksän yleistä hyökkäystä.

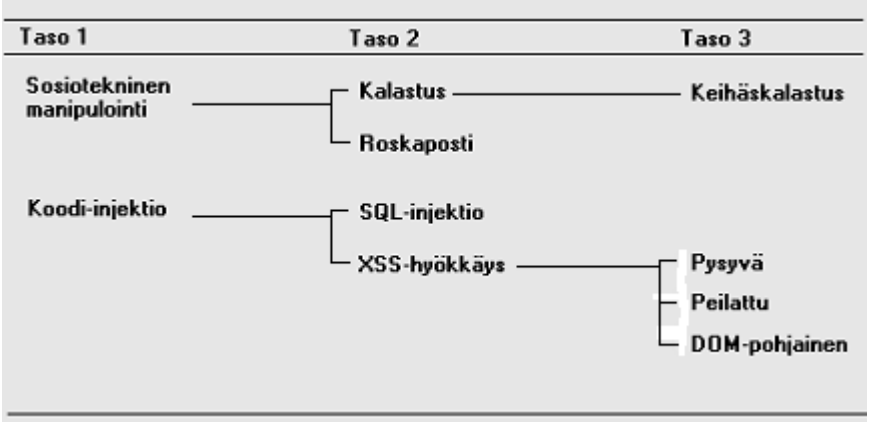
5.1 Hyökkäysten lajittelu

Tietoturvallisuustekniikkaa on vaivannut yhteisen kielen, *taksonomian*, puuttuminen hyökkäyksiä ja haavoittuvuuksia koskien. Hyvä taksonomia järjestää haavoittuvuudet ja hyökkäykset ominaisuuksien perusteella hyvin määriteltyihin ja selkeisiin kategorioihin [68]. Tähän työhön on valittu Hansmanin ja Huntin taksonomia [81] ohjenuoraksi hyökkäysten käsittelyssä.

Tässä taksonomiassa hyökkäys määritellään neljän ulottuvuuden kautta, joista vain ensimmäinen ulottuvuus on pakollinen. Taksonomia antaa melko vapaat kädet hyökkäysten lajitteluun, tosin Hansman ja Hunt painottavat mahdollisimman spesifisten määritysten tärkeyttä. Ulottuvuudet ovat: [81]

- **Hyökkäysvektori** (engl. *attack vector*) on metodi, jolla hyökkäys saavuttaa kohteensa.
- **Kohde** (engl. *attack target*) kertoo mitä vastaan hyökkäys suoritetaan.
- **Haavoittuvuudet** (engl. *vulnerabilities and exploits*) joita hyökkäys hyödyntää.
- *Neljäs ulottuvuus* on varattu hyökkäyksen sivuvaikutuksille, jos hyökkäysvektori ei riitä kuvaamaan hyökkäyksen vaikutusta. Tässä työssä tätä ulottuvuutta käsitellään eri tavoin. Neljänteen ulottuvuuteen viitataan **vaikutuksina.**

Hyökkäysvektorin tarkoitus ilmenee paremmin kuvasta 6.1.



Kuva 5.1: Hyökkäysvektorin kategorioita.

Hyökkäysten käsittelyn ei ole tarkoitus olla kattava, vaan läpi käydään sosiaalisille verkkosovelluksille olennaisimmat hyökkäykset. Olennaisimpien hyökkäysten löytämiseksi —ja niiltä suojautumiseksi— on turvauduttu OWASP-, WASC-, APWG- ja SANS-organisaatioiden ohjeistuksiin ja verkkoyhteisöjen omiin listauksiin. Edelleen näistä on valikoitu kaikkein relevantimmat hyökkäykset.

5.1.1 Taksonomian soveltuvuus sosiaalisiin verkkosovelluksiin

Taksonomiaa käsitellään sosiaalisten verkkosovellusten kontekstissa, joten on tarpeen tarkastella taksonomian soveltuvuutta ja tehdä joitain huomioita. Hansman ja Hunt epäilevät, että hyökkäyksistä ei pysty lajittelemaan hierarkista ja diskreettiä puumuotoista kategoriaa. Syynä on hyökkäysten lajittelemisen vaikeus niiden ominaisuuksien suhteen. Monet hyökkäykset sisältävät ominaisuuksia useasta kategoriasta. Igure ym. [68] väittävät, että tällaisen kategorisoinnin luominen voi olla mahdollista, jos taksonomia on tarpeeksi hyvin rajoitettu.

Kategorisoinnin vaikeuden ja monitulkintaisuuden vuoksi taksonomia on ennen kaikkea otettu työhön palvelemaan hyökkäysten käsittelyä, ei kategorisoimaan hyökkäyksiä mahdollisimman osuvasti. Täten hyökkäyksiä käsitellään tässä työssä korkealla abstraktiotasolla. Taksonomialla on mahdollista käsitellä hyökkäyksiä paljon yksityiskohtaisemmalla tasolla. Korkean abstraktiotason takia tässä työssä hyökkäysvektori on yhtä kuin hyökkäyksen nimi. Nimien määrittelyssä käytetään tämän hetken (2010) yleisimpiä nimiä vastaaville hyökkäyksille.

Hyökkäysvektori on tärkein ulottuvuus ja hyökkäysvektorin tulisi määritellä hyökkäys mahdollisimman tarkkaan, mikäli vain mahdollista. Koska ei aina ole selvää, mitä vaikutuksia esimerkiksi XSS-hyökkäyksellä voi olla, on tässä työssä *nel-*

jännen ulottuvuuden roolia laajennettu. Tässä työssä neljännellä ulottuvuudella ei tarkoiteta vain hyökkäyksen sivuvaikutuksia, vaan kaikkia mahdollisia vaikutuksia. Kohteen ei ole tarkoitus selvittää, miksi hyökkäys tehdään (esim. käyttäjää tai kehittäjää *vastaan*), vaan mikä on hyökkäyksen kohde ohjelmistoteknisestä näkökulmasta (selain, käyttöjärjestelmä, sovelluskerros, jne..).

Hansman ja Hunt esittävät [81] vaihtoehtoisten ulottuvuuksien esille tuontia tarpeen mukaan. Jos hyökkäyksiä käsitellään yksityiskohtaisemmin, sosiaalisten verkkosovellusten kontekstissa suotavaa olisi käsitellä Hansmanin ja Huntin esittämää **propagaatio** (engl. *propagation*) ulottuvuutta. Erityisesti haitakkeet ja roskapostitus ovat hyökkäyksiä, joille on tavallista nopea leviäminen. Ulottuvuuksien **vaurio** (engl. *damage*) ja **hinta** (engl. *cost*) ovat riskinhallinnan näkökulmasta tärkeitä. Vaurion on tarkoitus mitata vahinkoja tietojärjestelmälle, ei niinkään kuvata hyökkäyksen seuraamuksia. Hansman ja Hunt esittävät viimeiseksi **suojaus** (engl. *defence*)-ulottuvuuden, johon tässä työssä kiinnitetään erityistä huomiota. Tässä työssä otetaan kantaa myös propagaatioon, mutta ei muihin vaihtoehtoisiin ulottuvuuksiin.

5.2 Hyökkäykset ja niiltä suojautuminen

Ensimmäiseksi hyökkäyksistä käydään läpi sosiotekniseen manipulointiin perustuvat hyökkäykset. Koodi-injektioiden jälkeen tarkastellaan keskenään läheisiä hyökkäyksiä: XSS:ää ja CSRF:ää. Lopuksi esitellään palvelunestohyökkäys ja haitakkeet.

5.2.1 Sosiotekninen manipulointi

Sosiotekninen manipulointi on ehkä hyökkääjän yleisin hyökkäystekniikka. Se on helppo toteuttaa ja tehokas, sillä usein ihmistä pidetään tietoturvallisuuden heikoimpana lenkkinä. On helpompaa saada käyttäjä tekemään hyökkääjän haluama toimenpide, kuin murtaa paikoillaan olevaa suojausmekanismia. [83]

Tarkoitus on herättää käyttäjien uteliaisuus erilaisilla tempuilla, juonilla tai harhautuksilla. Tavallisesti hyödynnetään ajankohtaisia aiheita, shokeeraavia tai empatiaan vetoavia uutisia ja uskomattomia, tekaistuja tarjouksia. Tämän kaltaiset tekniikat koittavat saada käyttäjän toimimaan äkillisesti ja hetken mielijohteesta. Käyttämällä hienovaraisempia sosiaalisia manipulointitapoja hyökkääjä yrittää päästä käyttäjän luottamuksen piiriin. Kun hyökkääjä on kerran päässyt käyttäjän luottamuksen piiriin, voi hyökkääjä väärinkäyttää tätä luottamusta edistääkseen omia tarkoitusperiään.

Useimmiten sosiotekninen manipulointi toimii ensimmäisenä *hyökkäysvektorina* muille hyökkäyksille. Erityisesti sosioteknistä manipulointia käytetään kalastuksessa, identiteettivarkauksissa, roskapostituksessa ja haitakkeiden levittämisessä. Pelkkä sosiotekninen manipulointi yksistään voi olla riittävä *hyökkäysvektori*, esimerkiksi sähköpostin kautta salasanoja pyydettäessä. Tarkoituksena on yksinkertaisesti saada käyttäjä suorittamaan jokin toimenpide, joka edesauttaa hyökkääjän pyrkimyksiä. Tämä voi olla linkki haitalliselle verkkosivulle, arkaluontoisen tiedon kalasteluyritys tai sosiaalisen verkkosovelluksen *haavoittuvuutta* koodi-injektiolla hyödyntävä hyökkäys.

Kohteena on aina *ihminen*. Hyökkääjän kannalta keskussolmut ovat monessa mielessä kannattavia *kohteita* (ks. kpl 2.3). Keskussolmut omaavat vahvan liitännäisyyden ja heihin luotetaan. Tällaisen keskussolmun tietoturvallisuuden vaarantaminen tai tällaiseksi keskussolmuksi tekeytyminen on hyvä strategia hyökkääjän kannalta.

Digitaalisen jalanjäljen takia tulee olettaa, että hyökkääjällä on käytettävissä joitakin tietoja kustakin käyttäjästä. Hyökkääjä voi hyödyntää näitä tietoja suunnitellessaan kohdennettua sosioteknistä hyökkäystä. Täten käyttäjä, joka ei ole tietoinen digitaalisesta jalanjäljestään, on alttiimpi sosiotekniselle manipuloinnille.

Bilge ym. [40] tutkivat kuinka hyökkäyksissä pystyy hyödyntämään tekaistuja profiileja. Luottamussuhteiden muodostaminen käyttäjiin oli merkittävästi helpompaa kun käytössä oli vaikutusvaltaisen henkilön tekaistu profiili. Oletettavasti, käyttäjät luottivat tekaistuun profiiliin joka esitti henkilöä, jonka he tunsivat entuudestaan. Tämä pitää erityisesti paikkansa, jos tekaistu profiili esittää henkilöä, joka kuuluu käyttäjän vahvoihin siteisiin.

Keskussolmuksi pääseminen ei vaadi älykästä sosiaalista kanssakäymistä muiden kanssa, automatisoitu skripti pystyy saamaan vahvan liitännäisyyden [40] [65]. Huomiota tulisi kiinnittää profiilin omistajan viestihistoriaan, mieltymyksiin ja muihin ominaisuuksiin. Puuttuvat viestit, liian geneeriset mieltymykset tai muut asiat voivat toimia vihjeenä profiilin aitoudesta.

Sosiaaliset verkkosovellukset on suunniteltu sosiaalisen kanssakäymiseen, jolloin sosioteknisiltä hyökkäyksiltä on hankala *suojautua*. Monet sosiotekniset hyökkäystekniikat sisältävät tekstiä, jolloin niiden tehokkuus on osittain riippuvainen käytetystä kielestä. Käyttäjien valistaminen ei ole yleensä tuottanut hyviä tuloksia sosioteknistä manipulointia vastaan [83]. Tutustumalla ko. sosiaalisen verkkosovelluksen sääntöihin ja suosituksiin on mahdollista välttyä joiltain huijauksilta. Paras *suojaus* sosioteknistä manipulointia vastaan on käyttäjien oma harkinta ja varovainen asenne.

5.2.2 Kalastus

Kalastuksessa tarkoitus on ohjata käyttäjä hyökkääjän tekemälle verkkosivulle ja kerätä käyttäjältä jotain arkaluontoista tietoa, usein salasanoja. Käyttäjä harhautetaan sosioteknisellä manipuloinnilla luovuttamaan halutut tiedot Web-lomakkeeseen. Käyttäjän lähettäessä lomakkeen, arkaluontoiset tiedot päätyvät hyökkääjälle. Verkkosivu on luotu muistuttamaan —usein hyvin yksityiskohtaisesti— luotetun tahon tai organisaation sivuja, jotta käyttäjän epäilykset eivät herää.

Kalastus on pääasiallisesti sähköposteja vaivaava hyökkäys, joskin sosiaalisiin verkkosovelluksiin suunnatut kalastukset ovat kasvussa [84] [57]. Sosiaaliset verkkosovellukset eivät ole yhtä luotettava ja henkilökohtainen kanava kuin sähköposti, mikä selittää miksi useimmat kalastusyritykset tehdään sähköpostiin. Sosiaaliset verkkosovellukset tarjoavat hyökkääjille spesifistä tietoa joka auttaa tiettyyn henkilöön kohdistuvassa kalastushyökkäyksessä, nk. **keihäskalastuksessa** (engl. *spear phishing*) [65].

Sosiaalisissa verkkosovelluksissa kalastussivuille vieviä linkkejä voidaan syöttää viesteihin. Viestien *propagaatiota* voidaan oleellisesti jouduttaa hyödyntämällä *koodi-injektiohaavoittuvuutta* [65]. Vahva liitännäisyys ja käyttäjien suuri määrä sosiaalisista verkkosovelluksista kiinnostaa hyökkääjiä.

Hyökkääjä voi käyttää hyväkseen käyttäjien luottamusta ystäviinsä, jos hyökkääjä kykenee saastuttamaan näiden profiileja. Tavallisesti käyttäjät eivät osaa epäillä ystäviensä profiileissa esiintyviä linkkejä. Saastutettuun profiiliin voi olla mahdollista syöttää linkki, joka vie kalastussivulle. *MySpacessa* levinnyt *JS/Quickspace* mato hyödynsi tällaista sosioteknistä huijausta [65].

APWG-organisaatio on erikoistunut kalastuksen torjuntaan ja esittää ohjeita, kuinka välttää kalastushyökkäyksiä. Listan pituuden takia vain oleellisimmat ohjeet sosiaalisten verkkosovellusten kannalta huomioidaan tässä.

Käyttäjien tavallinen tae turvatusta ja luotettavasta sivustosta on ollut sertifikaatti tai selaimen osoiterivin URL-osoitteessa oleva `https`-määrite. Kumpikaan ei ole tae sivuston aitoudesta [48, s61] [85]. Huomiota kannattaa kiinnittää URL-osoitteeseen ja sertifikaatin tarjoajaan. Paras ratkaisu on kirjoittaa varmasti turvalliseksi tiedetyn verkkosivuston osoite suoraan osoiteriville, välttää sähköposteissa olevien linkkien klikkaamista ja noudattaa varovaisuutta sosiaalisissa verkkosovelluksissa olevien viestien suhteen.

Selaimet tarjoavat liitännäisiä, jotka varoittavat kalastusyrityksistä [85]. On suositeltavaa tutustua siihen, kuinka nämä toimivat, sillä ne eivät takaa täydellistä *suojausta* kalastusyrityksiä vastaan.

Suojausten luomiseksi kehittäjien tulee muodostaa kaikki käyttäjille suunnatut kyselyt tarpeeksi selkeällä, ainutlaatuisella tavalla, välttäen niitä samoja sosiaalisen kerroksen funktioita, joilla käyttäjät kommunikoivat tavallisesti keskenään. Näin käyttäjät voivat havaita eron kalasteluyritysten ja aidon kyselyn välillä. Esimerkiksi kun käyttäjä kirjautuu, hänet voidaan suoraan uudelleenohjata sivulle, jossa kehittäjien Web-lomake on. Käyttäjille kannattaa myös ilmoittaa, mitä tietoja voidaan kysellä, esimerkiksi tavallinen käytäntö on, että salasanoja ei ikinä kysytä.

5.2.3 Roskaposti

Roskaposti on vaivannut sähköpostia jo pitkään. Roskapostin tarkoitus on houkutella uhria klikkaamaan haitallista linkkiä tai liitetiedostoa, käyttäen sosioteknistä manipulointia. Tavanomaista sähköpostiin lähetettyä roskapostia koskevat *suojaukset* ovat pitkälle kehittyneitä. Usein viestit ovat tökerösti toteutettuja ja vain murto-osa pääsee läpi roskapostisuotimista. [86]

Sosiaalisissa verkkosovelluksissa roskaposti on sosiaalisen kerroksen funktioiden väärinkäyttämistä. Roskapostin lähettäminen ei vaadi minkään *haavoittuvuuden* hyödyntämistä, ellei sovelluksessa ole roskapostinsuodatusta. On mahdollista, että sosiaaliset verkkosovellukset korvaavat osittain sähköpostin [65], mikä houkuttelee roskapostittajia siirtymään etenevissä määrin näiden piiriin. Roskapostin lukumäärä on kasvanut sosiaalisissa verkkosovelluksissa [57].

Sähköpostin suhteen käyttäjät ovat tulleet tietoisemmaksi roskapostin suhteen, jotta he osaavat välttää niissä olevia huijauksia. Oletettavaa on, että ajan myötä samankaltainen valveutuneisuus kehittyy sosiaalisen verkkosovellusten käyttäjille. Valitettavaa on, että varsinaisten huijausten lisäksi roskaposti vie sovellusten resursseja ja häiritsee sovelluksen asianmukaista käyttöä.

Hyökkääjän kannalta tehokkainta on hyödyntää luottamuksen ongelmaa ja lähettää roskapostia muiden profiilien kautta. Tällöin käyttäjät eivät kykene yhdistämään mitään profiilia roskapostiin. Tämä suojelee hyökkääjän anonymiteettiä aivan kuten orjakonejoukot hälvennyttävät hyökkääjän anonymiteetin tavanomaista roskapostia lähetettäessä. Tämä vaatii oikeanlaisen *haavoittuvuuden* hyödyntämistä. *Kohteena* on käyttäjä, kuten kaikissa sosioteknisissä hyökkäyksissä.

Hyökkääjälle on tätäkin helpompaa luoda profiili, josta lähettää roskapostia. Hyökkääjä voi yrittää saastuttaa muita profiileja levittääkseen roskapostia edelleen tai lähettää ystäväpyyntöjä muille käyttäjille luodakseen luotettavuuden vaikutelmaa. Hyökkääjä voi odottaa kunnes hän pääsee keskussolmuksi, kunnes hän saa tarpeeksi ystäviä ja häneen luotetaan, ja hyökätä tämän jälkeen.

Sosiaaliset verkkosovellukset mahdollistavat hyökkääjien luoda kohdennettuja roskaposteja, keihäskalastuksen tapaan. Kohdennetut roskapostit voivat edelleen olla tavanomaista sähköpostia tai sosiaalisen verkkosovelluksen sisäistä viestiliikennettä. Kohdennetut roskapostit ovat paljon tehokkaampia kuin hakuammunnalla lähetetyt roskapostit. Roskapostinsuodatus ei kykene erottamaan kohdennettua roskapostia tavanomaisesta. Julkisesti saatavilla olevasta informaatiosta voidaan erottaa tietoja, joita hyödyntää roskapostissa. Esimerkiksi syntymäpäivätoivotuksen sisältävä roskaposti voidaan lähettää uhrille juuri silloin, kun hänen syntymäpäivänsä on. Hyökkääjä voi hyödyntää uhrin ystävien sähköpostiosoitteita roskapostissa, väärentäen sähköpostin uhrin ystävän nimissä. [86]

Ennen tuntemattomalta tulevan ystäväpyynnön hyväksymistä on suositeltavaa tutustua tämän profiiliin ja jos tarpeellista on, keskustella toisen osapuolen kanssa. On huomioitava, että vaikka tällä olisi paljon ystäviä, se ei ole tae luotettavuudesta [87]. Kehittäjien näkökulmasta hyödyllistä on tarkkailla nopeasti kasvavia solmuja ja solmuja, jotka lähettävät paljon ystäväpyyntöjä, jotka hylätään. Nämä voivat olla merkkejä roskapostittajista tai identiteettivarkauksista.

Roskapostin *vaikutuksia* voivat olla linkki haitalliselle verkkosivulle tai muun haitallisen tiedoston linkittäminen. Roskaposti voi toimia näin *hyökkäysvektorina* muille hyökkäyksille kuten koodi-injektiohyökkäykselle, haitakkeille tai kalastukselle. Näin ei kuitenkaan usein ole, roskapostia on myös keskustelukanavien luvaton käyttö mainostukseen.

Usein paikoillaan on toiminto, jonka avulla käyttäjät voivat usein ilmoittaa jonkun muun roskapostittajiksi. Tämän ratkaisun ongelmana on se, kuka valvoo näiden toimintojen oikeanmukaista käyttöä. Jos käytössä on mainejärjestelmä, voidaan eri käyttäjien roskaposti-ilmoituksia käsitellä eri-arvoisina.

Monet sosiaaliset verkkosovellukset ovat ottaneet käyttöön suodattimia, joilla hallita viestiliikennettä [65]. Roskapostin tunnistaminen koneellisesti on ollut hankalaa, sillä roskapostittajat oppivat uusia tapoja välttää suodatusmekanismeja [87]. Lee ym. [87] tutkivat mitä yhteisiä, roskapostittajille ominaisia piirteitä näiden profiileista ja viesteistä on löydettävissä. Näiden piirteiden avulla tutkimuksessa opetettiin luokittelijaa (tietokonetta) lajittelemaan viestit roskapostiksi tai aidoiksi viesteiksi. Tulokset ovat lupaavia, joskin menetelmän soveltaminen vaatii profiileista ja viesteistä erityisesti roskapostittajille ominaisten piirteiden tunnistamista, jolloin menetelmä ei välttämättä sovellu yhtä hyvin jokaiseen sovellukseen.

5.2.4 Koodi-injektio

Koodi-injektio hyödyntää useiden web-teknologioiden luontaista heikkoutta: Ei ole olemassa selkeää erottelua käyttäjien antamalle syötteelle ja ohjelmiston komentokäskyille [82, s4]. Hyökkääjä antaa syötteen, joka sisältää haitallisen koodin, tietojärjestelmään liityntäpisteen kautta. Liityntäpisteen tulee olla sellainen, jossa syöte voidaan tulkata koodiksi ja koodi suorittaa komentokäskyinä. Tavallisesti haitallinen koodi piilotetaan muun syötteen sekaan tai esitetään eriävässä muodossa *suojausten* ohittamisen toivossa.

Koodi-injektiohaavoittuvuudet ovat erittäin yleisiä verkkosovelluksissa [78], erityisesti SQL-injektiot ja XSS-hyökkäykset. OWASP on arvioinut koodi-injektion verkkosovellusten kaikkein vaarallisimmaksi hyökkäykseksi [63]. Eräs syy koodi-injektioiden suosioon on niiden suorittamisen helppous [78].

Alttiita teknologioita on useita, mm. SQL, LDAP, XPATH, HTML, JavaScript, ActionScript sekä ohjelmiston omien funktioiden väärinkäyttö [82] [63]. Hyökkäyksen *kohde* on mikä tahansa tietojärjestelmän liityntäpiste, joka ottaa vastaan syötteitä. Sosiaalisissa verkkosovelluksissa tämä tarkoittaa web-lomakkeita, keksejä, HTTP-otsakkeita ja kaikkia palvelimen kanssa keskustelevia protokollia, palveluita ja etäkutsuja. Koska koodi-injektiot ovat erittäin riippuvaisia käytetyistä teknologioista, *kohteena* on ainakin teknologia, mutta *kohteen* voi muodostaa monikko, josta ilmenee täsmällisemmät tiedot. Esimerkiksi koska jotkin koodi-injektiot ovat selainkohtaisia [82, s39-40,s99], tällöin *kohteen* muodostaa monikko $\{teknologia, sivu, selain\}$, josta ilmenee käytetyn teknologian ja selaimen lisäksi verkkoresurssi, johon hyökkäys kohdistuu. *Haavoittuvuus* on puutteellinen syötteen varmentaminen. Hyökkäyksen onnistuttua *vaikutukset* vaihtelevat datan menetyksestä, datan turmeltumiseen, verkkosivujen muokkaukseen, palvelunestoon, isäntäkoneen haltuunottoon ja moniin muihin vahinkoihin [63] [78].

Seuraava esimerkki havainnollistaa nk. *directory traversal* hyökkäystä. Esimerkissä hyökkääjä pääsee käsiksi verkkopalvelimen hänelle kuulumattomiin hakemistoihin syöttämällä sovellukseen sopivan koodi-injektion. Esimerkki on lainattu *Hacking exposed web 2.0*-teoksesta [82, s12].

Verkkosivusto ottaa käyttäjältä saaduista HTTP-pyynnöistä parametrina kielen, jolla käyttäjä haluaa sivuston esitettävän. Seuraava PHP-koodinpätkä esittelee verkkosivuston toimintalogiikan.

```
1 <?php
2 $language = "main-en";
3 if (is_set($_GET['language']))
4    $language = $_GET['language'];
5 include("/usr/local/webapp/static_files/" . $language . ".html");
6 ?>
```

Oletetaan lisäksi, että ko. sivu on saavutettavissa verkko-osoitteesta `http://foo.com/webapp/static.php?language=main-en;`. Koodissa oleva `include`-funktio esittää nyt käyttäjälle parametrina saadun tiedoston.

Jos hyökkääjä syöttää seuraavan GET-pyynnön palvelimelle `http://foo.com/webapp/static.php?language=../../../../etc/passwd%00`, funktio palauttaa tiedoston `/etc/passwd` sisällön hyökkääjälle, jota ei alunperin ole tarkoitettu loppupään käyttäjien saataville. `/etc/passwd` sisältää UNIX-pohjaisissa käyttöjärjestelmissä käyttäjätunnuksien salasanoja. Tiedoston salasanat ovat kryptattu. Pääsy tiedostoon on kuitenkin tietoturvariski.

Oleellista on tunnistaa kaikki liityntäpisteet, joissa käytetään jotain tulkkia ja varmistua siitä, että komentokäskyt erotetaan selkeästi syötteestä [65]. Koodi-injektio pyritään usein estämään syötteen suodattamisella [82, s99]: Syöte prosessoidaan suodattimen läpi, jolloin siitä poistetaan tietyn logiikan mukaisesti merkit tai lohkot, jotka tulkki kääntäisi koodiksi. *Suojaus* koodi-injektiota vastaan on todentaa syöte kelvolliseksi.

Suodatus voi toimia negatiivisella tai positiviisella logiikalla. Negatiivinen logiikka poistaa erikoismerkit tai merkkijonot, jotka on kukin määritelty. Esimerkiksi JavaScriptin estämiseksi suodatin voi estää `<script>` ja `<\script>` tunnisteiden käytön. Positiivinen logiikka määrittelee päinvastoin merkit ja merkkijonot, joita *voi* käyttää. Kannattaa käyttää suodatinta, joka on suunniteltu juuri kyseiselle tulkille [63]. Harvinaisemman esitystyypin ilmentyessä ja suodattimen puuttuessa voi olla tarpeellista kääntää syöte varmasti hallittuun merkistöön kuten ASCII-merkistöön.

Hyökkääjällä on monia keinoja, joilla syöttää koodia järjestelmään syötteen suodatuksesta huolimatta. Käytetyistä teknologioista ja järjestelmästä riippuu mille syötteille järjestelmä on *haavoittuvainen*. Syöte voidaan koodata eri esitysmuotoon tai syöte voidaan rikkoa erikoismerkeillä tai tunnisteilla siten, että se läpäisee suodatuksen [82, s99-102]. Jotta hyökkäys onnistuisi, ei suodatuksen läpi pääseminen riitä, tulkin täytyy vielä kääntää syötetty merkkijono ajettavaksi koodiksi.

Koska hyökkääjät keksivät usein uusia keinoja kuinka esittää haitallinen koodi tulkille, siten että haitallisen koodin suodatus vältetään, on turvallisempaa määrit-

tää se mikä on sallittua, kuin se, mikä ei ole sallittua [82, 103]. Suodatuksen määrittäminen positiivisen logiikan mukaan vie enemmän aikaa, mutta selvittämällä kunkin liityntäpisteen tapauksessa tarvittavat datan esitysmuodot ja kielet, voidaan testaamalla varmistaa sovelluksen turvallinen toiminta. Onkin paljon varmempaa laajentaa positiivisen logiikan ajattelua myös sovelluksen toimintoihin. Selvittämällä mitä toimintoja ja datan esitysmuotoja missäkin liityntäpisteessä todella tarvitaan, voidaan turhat toiminnot karsia pois ja kieltää tiettyjen esitysmuotojen käyttö.

Syötteen rajoittaminen kontekstin suhteen on suositeltavaa. Tämä tarkoittaa merkkien tai lukujen suuruuden rajoittamista ja oikeiden tietotyyppien käyttöä. Tyhjien merkkijonojen ja nollien huomioiminen on hyödyllistä, niillä voi olla erityinen rooli joissakin ohjelmointikielissä [88, s83].

Vastuualueiden rajan vetäminen luo epäselvän tilanteen. Jos verkkosivulla on haitallista koodia sisältävä merkkijono, ja selain kääntää merkkijonon automaattisesti ASCII-merkeiksi —ilman sovelluksen väliintuloa—, onko vastuu tietoturvallisuudesta selainten kehittäjillä vai verkkosovelluksen kehittäjillä? [82, s99]

Jos on mahdollista luoda rajapinta tai käyttää sellaista, sen käyttäminen estää tehokkaasti koodi-injektiohyökkäyksiltä. Parametrisoitujen rajapintojen käyttäminen on paljon turvallisempaa kuin tulkkien käyttäminen, silti nämä voivat yhä toimia liityntäpisteinä koodi-injektiolle. [63]

Kehittäjien kannattaa erityisesti olla varovainen skriptauskielien kanssa, jotka ovat hyvin vahvoja ja dynaamisia verrattuna rajatummin määriteltyihin kieliin kuten tietokannan määrittely- ja kyselykieliin (SQL, XPath). Tämä tarkoittaa useimmiten JavaScriptia ja Ajaxia [82, s88].

5.2.5 SQL-injektio

Verkkosovelluksissa dynaamisten toimintojen luomiseksi hyödynnetään tietokantoja. Interaktiivisuuden luomiseksi käyttäjien syötettä käytetään tietokantahauissa. Tietokantojen käyttämiseksi ja hallintaan on olemassa erillisiä kieliä, kuten Structured Query Language.

SQL-injektio on koodi-injektiohyökkäys, jossa käytetään tietokannan määrittely- ja kyselykieltä SQL:ää. SQL on hyvin suosittu ja sitä käytetään usein verkkosovelluksissa [82, s5]. Tietoturvan kannalta vajavaisesti ohjelmoidut tietokantarajapinnat voivat olla *haavoittuvaisia* SQL-injektioille. Tietokantapalvelu usein luottaa tietokantarajapintaan, jolloin SQL-injektio omaa tarvittavat käyttöoikeudet ja ohittaa kommunikaatio- ja kuljetuskerroksen *suojaukset*, kuten palomuurit [59, s573]. Tämä tie-

tokantarajapinta on SQL-injektion *kohde*. *Vaikutukset* voivat vaihdella luottamuksellisten tietojen julkaisemisesta aina tietojen hävittämiseen.

Havainnollistetaan SQL-injektiota *haavoittuvaisen* profiilien hakukentän esimerkillä. Hakukenttä ottaa vastaan käyttäjän syötteen, käyttää sitä *sellaisenaan* SQL-kyselyssä, ja palauttaa ohjelmiston sisäiselle toimintalogiikalle kaikkien niiden käyttäjien tunnisteet, jotka vastaavat kyselyn tulosta. Seuraava Java-koodinpätkä on *haavoittuvainen* koodi-injektiolle.

```
1 String kayttajan_syote=req.getParameter("profiili_haku");
2
3 String sql_haku="SELECT id FROM Profiles" +
4        "WHERE profilename LIKE '"+kayttajan_syote+"%'";
5
6 ResultSet kyselyn_tulokset = stmt.executeQuery(sql_haku);
```

Oletetaan, että hyökkääjä on valistuneella arvauksella tai virhe-ilmoituksia hyödyntämällä saanut selville relaatiotaulun `Profiles` olemassaolon. Kun hyökkääjä hakee profiileja seuraavalla merkkijonolla `'; DROP TABLE Profiles; --`, muodostuu SQL-kyselystä seuraanlainen:

```
SELECT id FROM Profiles WHERE profilename LIKE '';
DROP TABLE Profiles;
```

Nyt kun yo. SQL-kysely suoritetaan tietokannassa, se palauttaa niiden profiilien tunnisteet, joiden profiilinimi on tyhjä (`''`) ja hyökkääjän haluaman toiminnon: relaatiotaulun `Profiles` tuhoamisen (`DROP TABLE Profiles`). Hyökkääjä yksinkertaisesti hylkää kommenttimerkeillä (`--`) syötteidensä oikealla puolella olevan SQL-syntaksin (`%'`).

SQL-kielen tuntemuksella hyökkääjä voi pystyä aavistamaan kuinka SQL-kysely muodostuu sisäisessä toimintalogiikassa. Monesti on mahdollista oppia tarvittava syntaksi sovelluksen virhe-ilmoituksista tai pelkästään kokeilemalla [82, s7]. Eräs vaihtoehto on muodostaa SQL-kyselyt dynaamisesti, jolloin on tarpeellista todentaa käyttäjien syöte kelvolliseksi. Tällöin on tarpeellista käyttää oikeanlaista suodatinta [89]. Edellä olevassa esimerkissä tietokantakyselyt muodostettiin dynaamisesti. Esimerkissä oleva Java-koodinpätkä on erityisen huolimattomasti suunniteltu, sillä syöte otetaan vastaan sellaisenaan.

Hyökkääjä toivoo tarkkaan valikoitujen merkkijonojen ohittavan suodatuksen. Seuraavia merkkijonoja käytetään tavallisesti: Heittopilkku (`'`), kaksi viivaa (`--` ja

puolipiste (`;`) [59, s576]. Heittopilkulla merkitään muuttujien paikkoja. Kahdella viivalla merkitään kommenttien paikkoja, ts. kaikki näiden merkkien jälkeen tuleva syntaksi hylätään. Puolipisteellä päätetään kysely. Esimerkissä heittopilkulla ja puolipisteellä päätettiin edellinen kysely ja aloitettiin uusi kysely tietokantaan. Lopulta SQL-injektion kannalta turha osio hylättiin kommentoinnilla.

Ongelmia aiheuttaa se, kuinka jotkin sovellukset vaativat tiettyjen erikoismerkkien sallimisen syötteeksi. Esimerkiksi kuvitellaan hakukenttä, joka hakee tietokannasta sukunimiä. Kehittäjän mielestä SQL-syntaksin koskemattomuuden suojelemiseksi on hyvä kieltää yläheittomerkin kieltäminen (’). Kehittäjä ei kuitenkaan tällöin tule ajatelleeksi sukunimiä, jossa esiintyy yläheittomerkki, kuten *O’Brian*. [89] Erikoismerkkien kokonaan kieltämisen sijasta parempi ratkaisu voi olla syötteen erikoismerkkien koodaaminen eri esitysmuotoon eli **merkistökoodaus** (engl. *escaping*). Syötettä koodattaessa eri esitysmuotoon tulee huolehtia, että tietojärjestelmän eri osissa on sovittu yhtenäinen käytäntö käytössä olevista merkistöistä.

Dynaamisten tietokantahakujen sijaan kehittäjä voi määrittää ennalta kaiken SQL-koodin, jota ainoastaan voidaan käyttää [89]. Määräämällä ennalta joukon kyselyitä, jotka saavat parametreinä tarvittavat tiedot, voidaan välttää vaarallisten kyselyjen väärinkäyttöä kuten `DROP TABLE` -kyselyn käyttöä. Tämä ohjelmointityyli erottaa toisistaan koodin ja datan, minkä puute lähtökohtaisesti on perimmäinen syy *koodi-injektiohaavoittuvuuksille*. Tämä lähestymistapa onkin paljon dynaamisia kyselyjä turvallisempi ja toimii tehokkaana *suojauksena* SQL-injektiota vastaan.

Ennalta määrätyt SQL-kyselyt voidaan sijoittaa joko sovellukseen tai tietokantaan [89]. Vapaasti suomennettuna sovellukseen tallennettuja SQL-kyselyjä kutsutaan **parametrisoiduiksi kyselyiksi** (engl. *parameterized queries*) ja tietokantaan tallennettuja SQL-kyselyjä kutsutaan **tallennetuiksi proseduureiksi** (engl. *stored procedures*). Joskus parametrisoitujen kyselyjen tapauksessa tietokantayhteyksien suorituskyky saattaa kärsiä [89]. Tämä ei välttämättä sovi kehittäjille, koska tietokantayhteyksien suorituskyky on hyvin oleellinen ominaisuus sosiaalisten verkkosovellusten toiminnalle [14].

Tallennettujen proseduurien käyttäminen voi joissain tapauksissa tuoda ilmi uusia tietoturvariskejä. Tällainen on esimerkiksi MS SQL-palvelin, jossa on oletuksena kolme käyttäjäroolia: `db_datareader`, `db_datawriter` ja `db_owner`. Joissain konfiguraatioissa käyttäjienhallinta on rajoitettu näihin käyttäjärooleihin ja jotta tietokantaan sijoitetut kyselyt toimisivat, verkkosovellusten täytyy aina toimia `db_owner` käyttäjäroolissa. Tällä käyttäjäroolilla on täydet valtuudet tietokantaan. Tällöin jos hyökkääjä pystyy löytämään sopivan *haavoittuvuuden* sovelluskerroksesta, pääsee

hän käsiksi tietokantaan verkkosovelluksen käyttöoikeuksilla, jolloin hänellä on täysi valta tietokantaan. [89]

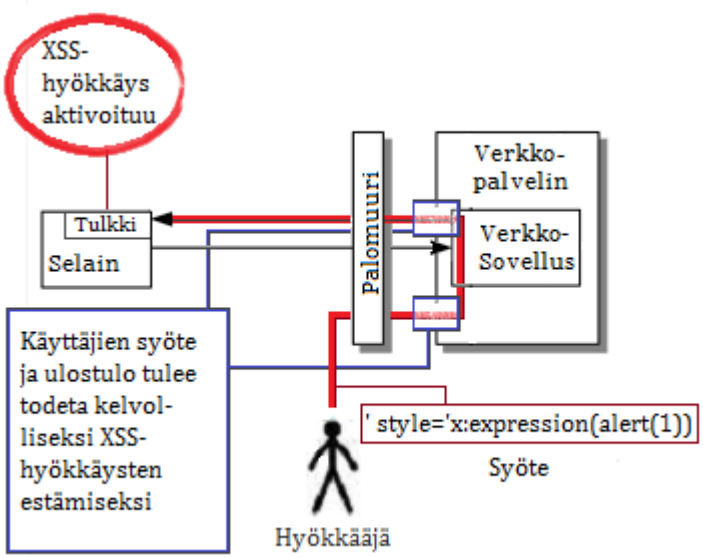
Tietokannan *suojaukseksi* —myös muilta uhkilta— on hyödyllistä konfiguroida palvelimet. Käyttäjäroolit kannattaa säätää pienimmän etuoikeuden periaatteen mukaisesti (ks. kpl 3.6.3). Huomiota kannattaa kiinnittää myös siihen, millä käyttöjärjestelmän käyttäjäoikeuksilla tietokannanhallintaohjelmisto ajetaan [89]. Tutustumalla laitteistoihin ja palvelinohjelmistoihin voidaan välttyä monilta potentiaalisilta tietoturvariskeiltä, kuten yllä olevalta tallennettujen proseduurien tietoturvariskiltä. Oikea konfiguraatio voi mahdollisten tietoturvaparannusten lisäksi tuoda lisää suorituskykyä tietojärjestelmään. Kehittäjien kannattaa testata sovellustaan *SQL-haavoittuvuuksien* löytämiseen tehdyillä työkaluilla [59, s575].

5.2.6 Cross Site Scripting (XSS)

Selaimien ehkä tärkein tietoturvakäytäntö on **saman lähteen tietoturvakäytäntö** (engl. *same origin policy*). Tämä käytäntö rajoittaa dynaamisen sisällön lukuoikeuksia HTTP-vastauksiin ja kekseihin, jotka tulevat *samasta lähteestä*. Sama lähde määritellään isäntänimen, protokollan ja portin monikkona. Käytäntö ei ota kantaa kirjoitusoikeuksiin. Täten verkkosivut voivat lähettää HTTP-pyyntöjä mille tahansa muulle verkkosivustolle. Verkkosivut voivat kuitenkin olla reagoimatta HTTP-pyyntöihin. [82, s22].

Cross Site Scripting(XSS)-hyökkäyksen tavoite on kiertää *saman lähteen tietoturvakäytäntö* injektoimalla dynaamista sisältöä verkkosivulle [82, s32]. Dynaamisen sisällön mahdollistavat selainten tukemat skriptauskielet, kuten JavaScript ja VBScript. Haitallinen koodi sijoitetaan usein HTML-tunnisteiden sisään [59, s571] [88, s32]. Käyttäjän ladatessa verkkosivun selain tulkkaa dynaamisen sisällön suoritettavaksi koodiksi ja ajaa sen. *XSS-haavoittuvaisen* verkkosivuston kautta hyökkäys etenee *kohteeseensa*, käyttäjän selaimeen.

XSS-hyökkäykset ovat riippuvaisia käytetystä skriptauskielestä. Jos verkkopalvelin tai selain ei tue hyökkäyksen skriptauskieltä, on tietojärjestelmä (tai käyttäjä) immuuni kyseiselle hyökkäykselle. Koska sosiaalisissa verkkosivustoissa on usein dynaamista sisältöä, hyökkäysalustoja XSS-hyökkäyksille on runsaasti. Käyttäjä voi kytkeä selaimesta pois JavaScriptin, jota usein käytetään XSS-hyökkäyksissä [88, s32]. Tällöin käyttäjä oleellisesti menettää monia hyödyllisiä toimintoja, jotka ovat voineet olla lähtökohtaisesti syynä siihen, miksi käyttää kyseistä sosiaalista verkkosovellusta. Selaimet käsittelevät eri tavoin skriptauskieliä, kuten JavaScriptiä. Täten jotkut XSS-hyökkäykset ovat riippuvaisia käytetystä selaimesta [82, s39]



Kuva 5.2: Saman lähteen tietoturvakäytännön kiertäminen XSS-hyökkäyksellä.

Haitallinen koodi voi löytää selaimen tulkkiin yllättäviä teitä pitkin. internet Explorer 7 ja sitä aiemmat versiot käsittelevät virheellisesti kuvia. GIF-formaatissa osa biteistä on varattu kommenttilohkolle. Jos hyökkääjä sijoittaa haitallisen koodin GIF-kuvan kommenttilohkoon, internet Explorer 7 ja sitä aiemmat versiot suorittavat koodin, kunhan vain selain lataa kuvan. Toisin sanoen, käyttäjän vierailu verkkosivulla, jonne kuva on upotettu, riittää. [82, s42-43]

Tavallisimpia *vaikutuksia* XSS-hyökkäyksellä on käyttäjän istuntokeksien varastaminen, istuntojen ja käyttäjätilien kaappaaminen, uudelleenohjaukset ja maineen menetys organisaatiolle, jonka verkkosivustoa käytetään hyökkäyksessä [59, s571]. *XSS-haavoittuvuutta* voidaan myös käyttää hyväksi haitakkeen levittämisessä, kuten MySpacessa levinnyt Sammy XSS-mato osoitti [82, s55].

On kolmentyyppisiä XSS-hyökkäyksiä [88, s32-36]:

1. **Pysyvä XSS** (engl. *Persistent attack*)
2. **Peilattu XSS** (engl. *Non-persistent attack*)
3. **DOM-pohjainen XSS** (engl. *DOM-based attack*)

Pysyvän XSS-hyökkäyksen tapauksessa haitallinen koodi jää verkkosivulle. Verkkosivulle täytyy olla mahdollista lisätä syötteitä, kuten tekstiä. Esimerkiksi keskustelufoorumi voi olla altis pysyvälle XSS-hyökkäykselle. Pysyvässä XSS-hyökkäyksessä haitallinen koodi yleensä tallentuu verkkopalvelimen tietokantaan. Käyttäjä

altistuu välittömästi hyökkäykselle, jos hän vierailee pysyvän XSS-hyökkäyksen sisältävällä verkkosivulla. [88, s32-33]

Peilatut ja DOM-pohjaiset XSS-hyökkäykset vaativat käyttäjää klikkaamaan haitallista linkkiä tai vierailemaan hyökkääjän verkkosivulla, jossa on haitallinen weblomake. Linkin tapauksessa linkki itse sisältää haitallista koodia. Web-lomake hyödyntää käyttäjän syöttämiä tietoja ja lähettää lomakkeen tiedot verkkosivulle, jossa on *XSS-haavoittuvuus*. [88, s32]

Pysyvissä ja peilatuissa XSS-hyökkäyksissä haitallinen koodi tulee HTTP-vastauksen mukana, joka palautetaan verkkopalvelimen toimesta käyttäjälle. DOM-pohjaisessa XSS-hyökkäyksessä verkkosivua käytetään toimittamaan ajonaikaista koodia käyttäjälle, jonka käyttäjän selain suorittaa, ilman että haitallinen koodi kulkee verkkopalvelimelle. [88, s32-36]

Esimerkiksi verkkosivu joka saa URL-osoitteeseen parametrinä käyttäjän nimen voisi olla seuraavanlainen:
`http://www.esimerkki.com/tervetuloa.html?name=Keijo`
DOM-pohjainen XSS-hyökkäys voidaan muodostaa seuraavalla linkillä:
`http://www.esimerkki.com/tervetuloa.html?name=<script>alert(document.cookie)</script>`
Kun käyttäjä klikkaa linkkiä, selain ohjaa käyttäjän `www.esimerkki.com` verkko-osoitteeseen ja suorittaa haitallisen koodin lokaalisti käyttäjän koneessa (`<script>alert(document.cookie)</script>`).

Kaikissa tapauksissa saman lähteen tietoturvakäytäntö kierretään ja hyväksikäytetään käyttäjän selaimen luottamusta turvalliseen verkkosivustoon.

XSS-hyökkäyksiltä välttyäkseen kehittäjien tulee huolehtia käyttäjien syötteen sekä ulostulon kelvolliseksi todentamisesta. Pelkästään kulmasulkeiden suodattamisella torjutaan useimmat XSS-hyökkäykset. Kun käyttäjälle välitetään HTTP-vastaus (verkkosivu), tulee erikoismerkit koodata, kuten kulmasulkeet, eri esitysmuotoon. Tällöin käyttäjän selain ei tulkitse niitä suoritettavaksi koodiksi. Esimerkiksi ASCII-merkistön kulmasulkeita vastaa ISO-8859-1 merkistössä seuraavat merkkijonot: < on `%3C` ja > on `%3E`. Syötteestä kannattaa suodattaa joitain erikoismerkkejä, mikäli se vain on sosiaalisen verkkosovelluksen funktion kannalta mahdollista, kuten `< > ( ) # & "`. Jos sovelluksessa käytetään keksejä, voidaan niiden `http-only`-lippu säätää päälle, jolloin skriptauskielet eivät voi lukea keksejä. Tämä parantaa sovelluksen luotettavuutta, mutta tämä suojauskeino vaatii käyttäjältä selaimen, joka tukee tätä toimintoa. [59, s571-573]

5.2.7 Cross Site Request Forgery (CSRF tai XSRF)

Cross Site Request Forgery (CSRF)-hyökkäyksessä käyttäjän selain pakotetaan, käyttäjän tietämättä tai haluamatta, lähettämään HTTP-pyyntö verkkosivustolle. Olennaista hyökkäykselle on se, että hyökkääjä pystyy tekeytymään käyttäjäksi ja toimimaan näin käyttäjän käyttöoikeuksilla.

Hyökkääjä suorittaa sosiaalisella kerroksella toiminnon käyttäjän valtuuksilla. *Vaikutukset* vaihtelevat arkaluontoisen tiedon julkistamisesta käyttäjän profiilin tuhoamiseen, ystävien poistamiseen tai lisäämiseen ja kaikkiin mahdollisiin toimintoihin, joita käyttäjä voisi tavallisesti tehdä sovelluksella. CSRF-hyökkäys joka julkistaa arkaluontoista informaatiota voisi pyytää sosiaalista verkkosovellusta paljastamaan käyttäjän keskusteluhistorian. Koska hyökkääjä suorittaa toiminnon käyttäjän selaimella tämän puolesta, sosiaalisissa verkkosovelluksissa lähes aina CSRF-hyökkäyksen *kohde* on jokin sosiaalisen kerroksen funktio.

CSRF muistuttaa XSS-hyökkäystä siinä, että XSS-hyökkäys käyttää hyväkseen käyttäjän selaimen luottamusta verkkosivustoon, kun taas CSRF-hyökkäys käyttää hyväkseen verkkosivuston luottamusta selaimeen [88, s32].

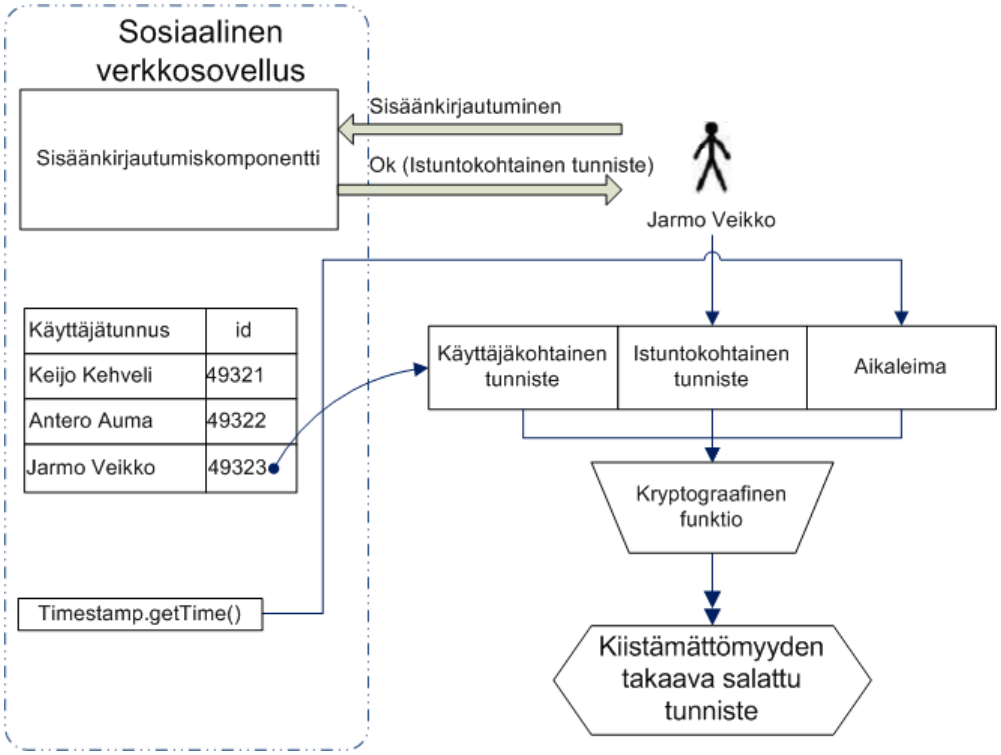
Hyökkääjä yrittää arvata minkälaisen muodon käyttäjän ja verkkosivuston viestintä ottaa [88, s38]. Tämä voi esimerkiksi onnistua kuuntelemalla verkkoliikennettä käyttäjän selaimen tai verkkosivuston välillä tai tutkimalla verkkosivuston HTML-lähdekoodia, mikä onnistuu esimerkiksi selaimella. Jos hyökkääjä onnistuu selvittämään kuinka viestintä tapahtuu, hän pystyy jäljittelemään pyyntöjä käyttäjän puolesta. Tällöin hyökkäykselle usein riittää että, käyttäjä on sisäänkirjautunut sosiaaliseen verkkosovellukseen ja että hyökkääjä saa käyttäjän selaimen toimittamaan hyökkääjän haluaman pyynnön (*hyökkäyksen vaikutuksen*).

XSS-hyökkäysten tapaan voi olla **peilattuja tai pysyviä CSRF-hyökkäyksiä** [82, s78-80]. Pysyvä CSRF-hyökkäys sisällyttää haitallisen koodin samalle verkkosivulle, jonne pyyntö lopulta kohdistuu. Jos sosiaalinen verkkosovellus sisältää tilan, jonne voi upottaa koodia, voi verkkosovellus olla altis pysyvälle CSRF-hyökkäykselle. Esimerkiksi kolmannen osapuolen kehittäjä voi luoda pysyviä CSRF-hyökkäyksiä, jotka voivat pakottaa käyttäjän suorittamaan toimintoja, esimerkiksi ostamaan virtuaalihyödyke tietämättään. Peilattu CSRF-hyökkäys vaatii käyttäjää vierailemaan hyökkääjän verkkosivulla, josta CSRF-hyökkäys toimeenpannaan.

XmlHttpRequest (XHR) on kirjasto, jolla voidaan asynkronisesti keskustella verkkopalvelimen kanssa [82, s103]. Käyttäjän selaimen ei tarvitse ladata koko verkkosivun sisältöä uudelleen, vaan osa verkkosivun sisällöstä voidaan päivittää. XHR -kirjastolla hyökkääjä voi saada käyttäjän selaimen toimittamaan pyyntöjä sosiaa-

liseen verkkosovellukseen ilman käyttäjän tietämystä. Ajax, jota usein käytetään XHR:än yhteydessä, on selaintekniikkakokonaisuus [2, s39]. Tästä syystä kaikki verkkosovellukset ovat alttiita CSRF-hyökkäyksille, jotka käyttävät XHRää. XHR on tehokas kirjasto web-sovelluksien kehittämisessä, mutta myös vaarallinen sen vuoksi. Skriptauskielien avulla hyökkääjä pystyy imitoimaan monimutkaisiakin käyttäjän toimintoja. Esimerkiksi monivaiheinen —käyttäjän palautetta useasti vaativa— tapahtuma voidaan toistaa skriptauskielillä. [63]

Olennaista on pyrkiä selvittämään pyynnön alkuperä. Toisin sanoen CSRF-hyökkäyksen estää *suojaus*, joka tuo kiistämättömyyden tietojärjestelmän toimintoihin. *Saman lähteen tietoturvakäytäntö* sallii mistä tahansa internetistä pyyntöjen lähettämisen verkkosovellukselle. Sosiaalisen kerroksen funktion kutsu tulee vahvistaa tulevan sallitulta verkkosivulta. Tämä onnistuu vaatimalla jokaisesta sallitusta lähteestä käyttäjä- ja istuntokohtainen ennalta arvaamaton salattu tunniste [82, s86]. Tunniste voidaan koostaa esimerkiksi kryptografisella funktiolla, jossa parametreina on palvelimelta saatu salaisuus, käyttäjätunnus ja aikaleima. Tässä ratkaisussa palvelimelta saatu salaisuus kiinnittää käyttäjän tiettyyn istuntoon, käyttäjätunnus kiinnittää käyttäjän tiettyyn profiiliin ja aikaleima kiinnittää käyttäjän tiettyyn hetkeen.



Kuva 5.3: Kiistämättömyyden takaavan tunnisteen muodostaminen kryptografisella funktiolla.

Kehittäjien kannattaa tarkastaa erityisesti ne sosiaalisen kerroksen funktiot, joita hyökkääjät haluaisivat käyttää väärin. Esimerkiksi ystäväpyyntöjen lähettämiset, sisällön luomiset tai poistamiset ja käyttäjän liittäminen johonkin ryhmään ovat tällaisia funktioita. Funktiot, joilla on vakavia seuraamuksia, kuten profiilin poistaminen, kannattaa varmentaa pyytämällä käyttäjätunnus ja salasana uudelleen.

Edelleen, CSRF-hyökkäystä voidaan hyödyntää muiden hyökkäystekniikoiden yhteydessä, esimerkiksi roskapostituksessa. Hyökkääjä voi käyttäjän profiililla mainostaa jotain tuotetta tai sivustoa. Näin hyökkääjä pyrkii hyödyntämään luottamuksen ongelmaa ja toimimaan anonyymisti.

Käyttäjä voi välttää CSRF-hyökkäyksen riskiä kirjautumalla ulos järjestelmästä, silloin kun hän ei sitä käytä. Tällöin CSRF-hyökkäykset eivät voi hyödyntää käyttäjän voimassaolevaa istuntoa.

5.2.8 Palvelunestohyökkäys

Palvelunestohyökkäys (*Denial of Service, (DoS)*) on uhka tietojärjestelmän saatavuudelle. Tarkoituksena palvelunestohyökkäyksessä on estää tietojärjestelmän normaali toiminta tyrehdyttämällä sen resurssit, vikaannuttamalla se tai väärinkäyttämällä sen toimintoja [88, s41]. Usein palvelunestohyökkäykset ovat kursailemattomia ja puhtaaseen voimaan perustuvia.

Palvelunestohyökkäys voidaan suorittaa tietojärjestelmän ulkopuolelta tai sisäpuolelta. Edelleen hyökkäyksen *kohde* voi olla sisäpuolella tai ulkopuolella. *Kohde* voi olla mikä tahansa sosiaalisesta, sovellus- tai kommunikaatio- ja kuljetuskerroksesta. Kommunikaatio- ja kuljetuskerroksella palvelunestohyökkäys voi yrittää estää järjestelmän normaalin toiminnan luomalla suuren määrän yhteyspyyntöjä järjestelmään [88, s41] tai mainostamalla OSPF-reititysprotokollalla harhaanjohtavia reittejä [49, s112], eväten käyttäjiltä palvelun. Sovelluskerrokseen, johon tavallisella käyttäjällä ei ole suoraa liityntäpistettä, hyökkääjä voi yrittää päästä käsiksi puskuriylivuodolla [88, s41]. Sosiaalisen kerroksen väärinkäyttäminen tarkoittaa sovelluksen käyttöliittymän käyttämistä palvelunestohyökkäykseen.

Esimerkki sosiaalisen kerroksen palvelunestohyökkäyksestä on sisäänkirjautumistoiminnon väärinkäyttäminen. Tekniikka, jossa hyökkääjä kokeilee satunnaisesti salasanaksi kaikkia mahdollisia merkkijonoja, estetään tavallisesti jäädyttämällä käyttäjätili joksikin aikaa. Näin kuitenkin hyökkääjä voi suorittaa palvelunestohyökkäyksen, jossa estetään tiettyä käyttäjää pääsemästä sosiaaliseen verkkosovellukseen [74]. Tämän kyseisen palvelunestohyökkäyksen estämisessä voidaan käyt-

tää CAPTCHA-testejä varmistamaan, että käyttäjä on ihminen. Tällä voidaan ehkäistä kuitenkin vain automatisoidut palvelunestohyökkäykset.

Hajautetut palvelunestohyökkäykset *(Distributed Denial of Service, (DDoS))* ovat palvelunestohyökkäyksen erikoismuoto, jossa palvelunestohyökkäys suoritetaan usealla tietokoneella [49, s332]. Hyökkääjällä on hallussaan useita tietokoneita tai hänellä on käytössään orjakonejoukko. Kun hyökkääjä antaa komennon, tietokoneet suorittavat koordinoidusti hajautetun palvelunestohyökkäyksen. Hajautetut palvelunestohyökkäykset ovat hankalia torjua ja niiden suorittamiseksi on olemassa valmiita työkaluja. Hajautetut palvelunestohyökkäykset muodostavat vaarallisen uhkan sosiaalisille verkkosovelluksille.

Ei ole olemassa tyydyttävää menetelmää hajautettujen palvelunestohyökkäysten torjumiseksi. Joitakin menetelmiä voidaan kuitenkin käyttää. Hajautettu palvelunestohyökkäys voidaan estää tunnistamalla verkkoliikenteestä, mikä on palvelunestohyökkäykseen kuuluvaa liikennettä ja mikä on tavallista liikennettä, ja suodattamalla kuulumaton liikenne pois. Tämän erottaminen ei ole helppoa. Lisäksi kehittyneemmät palvelunestohyökkäykset lisäävät joukkoon tavallista liikennettä hämätäkseen palvelunestohyökkäyksen havaitsemista. Palvelunestohyökkäyksen voi torjua jäljittämällä hyökkäyksen alkuperän ja estämällä sen. Tehokas ratkaisu hajautettujen palvelunestohyökkäysten torjumiseen on hajautettu *suojausjärjestelmä*. Tämän aiheen tarkempi käsittely sivuutetaan. [49, s332-s333]

Tarkastellaan sosiaalisten verkkosovellusten kannalta ainutlaatuista palvelunestohyökkäystä, jossa hyökkäysalustana toimii käyttäjien muodostama sosiaalinen verkosto.

Erittäin suositut keskussolmut voivat viedä paljon resursseja sekä sisäisestä että tietojärjestelmän ulkopuolisesta verkosta. Esimerkiksi oletetaan, että on olemassa keskussolmu, jolla on 1 000 000 seuraajaa tai ystävää. Tämä keskussolmu julkaisee linkin tiedostoon, jota ylläpitää ulkopuolinen tiedostonjakopalvelu. Lukuisat innokkaat seuraajat haluavat katsoa mistä on kyse ja lataavat tiedoston. Tämä koituu suureksi rasitteeksi tiedostonjakopalvelulle, kun tiedoston suosio kasvaa räjähdysmäisesti. Yhtälailla supersuositut keskussolmut vaativat ylimääräisiä resursseja laitteistolta ja ohjelmistolta, kun ne yksinään muodostavat merkittävän osan verkkoyhteisön verkkoliikenteestä.

Tällaisia keskussolmuja hyökkääjät haluavat väärinkäyttää palvelunestohyökkäyksissä, jos mahdollista. Ilman identiteettivarkautta (profiilin haltuunottoa) hyökkääjä luultavasti yrittää masinoida hyökkäystään melko tuntemattoman profiilin kautta (suositun keskussolmun näkökulmasta).

Näiden suosittujen keskussolmujen hyödyntämistä voi estää hienojakoisemmilla ystäväjoukoilla ja toimeenpanemalla tiukempia pääsynvalvonnallisia rajoitteita ulommille, valtaville ystäväjoukoille. Tällä voidaan ehkäistä tuntemattomia *nollapäivähaavoittuvuuksia*. Menetelmä voi kuitenkin olla liian rajoittava monelle sosiaaliselle verkkosovellukselle.

Luottamusjärjestelmiä, jotka rajoittavat sosiaalisen kerroksen funktioiden käyttöä solmun maineen mukaan, voidaan käyttää rajoittamaan uusien, tuntemattomien solmujen väärinkäyttöä. Hyökkääjä voi kuitenkin odottaa tarpeeksi kauan, että hänellä on hyvä maine. Tämä ratkaisu heikentää uusien käyttäjien näkökulmasta sovelluksen houkuttelevuutta. Lisäksi tämä ei estä hyökkäyksiä, jotka väärinkäyttävät solmuja, joilla on jo hyvä maine.

On hyvä pitää paikallaan säätelyjärjestelmiä, jotka aktivoituvat kun jotkut solmut alkavat käyttää liikaa resursseja. Viestiliikenteen määrää, ryhmien luomista ja ystävien lisäämistä kannattaa valvoa, ja rajoittaa tarpeen mukaan. Tottelemattomat solmut voidaan jäädyttää väliaikaisesti. Nämä ratkaisut eivät toimi täydellisenä *suojauksena* palvelunestohyökkäyksiä vastaan, mutta pystyvät pienentämään hyökkäyksen vaikutusta tietojärjestelmään.

5.2.9 Haitakkeet

Haitakkeet muodostavat hyvin laajan ja vaihtelevan joukon. Haitake on yleisnimi viruksille, madoille, troijalaisille, vakoiluohjelmille jne.. [83]. Tässä työssä haitaketta käytetään samassa, laajassa kontekstissa. Oleellista haitakkeelle on sen kyky levitä ja toistaa hyökkäystä, usein itsenäisesti.

Kuten sosiaalisissa verkkosovelluksissa yleensä, sosiotekninen manipulointi on etenevissä määrin ensimmäinen *hyökkäysvektori* haitakkeiden levittämisessä [83]. Sosiotekninen manipulointi haitakkeiden levittämiseen noudattaa samankaltaisia keinoja kuin roskapostin, kalastelun ja muiden hyökkäysten ohella. Haitakkeiden levittämisessä voidaan yhdistää myös muita hyökkäystekniikoita, kuten Sammy-mato, joka hyödynsi MySpacen *XSS-haavoittuvuutta*.

Haitakkeiden *kohde* on yleensä sähköposti [83]. Kasvavissa määrin sosiaaliset verkkosovellukset ovat *kohde* haitakkeille. Lisäksi, jotkut uskovat sosiaalisten verkkosovellusten korvaavan osittain sähköpostin roolin (ks. kpl 5.2.3). Jos näin käy, tämä varmasti ohjaa haitakkeiden kehittäjiä keskittämään huomionsa sosiaalisiin verkkosovelluksiin.

Abraham ym. [83] havaitsivat, että lähes aina haitakkeiden toiminnot ovat seuraavat:

1. Käyttäjän sosiotekninen manipulointi
2. *Suojausten* ohittaminen
3. *Vaikutuksen* toimeenpano
4. *Propagaatio*

Toiminnot eivät aina etene yllä olevassa järjestyksessä. Haitake voi keskittyä leviämiseen ja aktivoitua myöhemmin (3. vaihe) ulkopuolisesta komennosta tai tiettyjen ehtojen täyttyessä. Joidenkin haitakkeiden *vaikutus* voi olla *suojausten* ohittaminen (jolloin 2. vaihe on yhtä kuin 3. vaihe). Tällainen on esimerkiksi **takaovi** (engl. *backdoor*), joka varmistaa hyökkääjälle pääsyn koneen *suojausten* ohi tämän haluamana, myöhempänä ajankohtana. Haitakkeet voidaan jaotella erityispiirteidensä avulla tai aktivoitumis- ja leviämistapojen kautta.

Haitakkeen *propagaatio* on tärkeä ulottuvuus haitakkeille. internetiin verrattuna sosiaalinen verkkosovellus pitää sisällään paljon kiinteämmän ja tiivimmän käyttäjäkunnan (yhteisön). Käyttäjien keskinäinen kommunikointi on paljon tehokkaampaa ja noudattaa tiettyjä säännönalaisuuksia. Haitakkeet hyödyntävät leviämisstrategioissaan alustan samankaltaisuutta, tässä tapauksessa käyttäjiä yhdistävää sosiaalista verkkosovellusta. Saastuttamalla yhden käyttäjän haitake saa lukuisia, tulevia potentiaalisia *kohteita* lisää vahvan liitännäisyyden takia. Tämä edesauttaa merkittävästi leviämistä.

Haitakkeiden kehittäminen kestää jonkin aikaa, verrattuna moniin muihin esiteltyihin hyökkäyksiin, kuten roskapostiin, koodi-injektioon, palvelunestohyökkäyksiin. Näitä muita hyökkäyksiä voi nopeasti ajaa, sillä useimmissa niissä voidaan hyödyntää sovelluksen omaa toimintalogiikkaa hyökkäysten suorittamiseksi. Työkaluja haitakehyökkäysten suorittamiseksi on olemassa [83]. On huomattavissa, että sosiaalisille verkkosovelluksille sovitetut haitakkeet tulevat lisääntymään [57].

Vaikutukset voivat olla hyvin vaihtelevia. Joitain tavallisia *vaikutuksia* ovat **näppäinlokin** (engl. *key logger*), *takaoven* tai *rootkitin* asentaminen uhrin koneelle [83]. Näppäinloki kirjaa ylös lokiin käyttäjän näppäinlyönnit. Kaikki salasanat, mukaan lukien sosiaalisen verkkosovelluksen, vaarantuvat. Rootkitit ovat haitakkeita, jotka ovat erikoistuneet itsensä piilottamiseen. Usein rootkit odottaa tiettyjen ehtojen täyttymistä tai komentoja, jolloin niiden varsinainen *vaikutus* paljastuu. Monet rootkitit aktivoituvat, kun käyttäjä vierailee tietyllä verkkosivulla [83]. Rootkit voisi

odottaa käyttäjän kirjautumista sosiaaliseen verkkosovellukseen, ja toimittaa CSRF-hyökkäyksiä käyttäjän valtuuksilla.

Esimerkiksi eräs **troijalainen** (engl. *trojan horse*), haitake joka tulee jonkun hyödylliseltä vaikuttavan tiedoston yhteydessä, odottaa käyttäjän avaavan pankin verkkosivun. Tämän jälkeen troijalainen suorittaa kalastushyökkäyksen esittämällä käyttäjälle tekaistun verkkosivun. Troijalaisella voi olla konfiguraatiotiedosto sadoille eri pankeille, joka määrittelee millainen verkkosivu käyttäjälle tulee esittää. [56]

Esimerkki näppäinlokin asentavasta haitakkeesta on *W32.HLLW.Fizzer@mm*. Haitake käyttää sähköpostia leviämiseen ja saastuttamiseen. Saastutettuaan koneen *W32.HLLW.Fizzer@mm* kirjaa ylös käyttäjän näppäinlyönnit lokiin, jonka haitake salaa ja sijoittaa käyttäjän koneelle. Myöhemmin haitake ottaa yhteyden hyökkääjän hallitsemille IRC-palvelimille ja odottaa komentoja hyökkääjältä. Edelleen, saatuaan komennon hyökkääjältä, haitake lähettää lokitiedoston hyökkääjälle. [83]

Haitake ei aina toimi näkymättömänä. Hyökkääjä voi esittää käyttäjälle ponnahdusikkunoita, jotka varoittavat käyttäjää olemattomasta uhkasta, usein haitakkeesta, ja ohjeistaa käyttäjää lataamaan ohjelman, jolla uhka voidaan poistaa. Ladattu ohjelma on haitake. Tällaisia pelotteluun kuuluvia hyökkäystekniikoita kutsutaan *scarewareksi*. Hyökkäystekniikka, jossa haitake salaa käyttäjän tiedostot, estäen niiden käytön, ja vaatii käyttäjältä rahaa niiden avaamiseksi, kutsutaan *ransomwareksi*. [83]

Haitaketta voidaan käyttää muuttamaan käyttäjän kone orjakoneeksi. Tällöin konetta voidaan hyödyntää hyökkääjän haluamana aikana monella tapaa. Orjakone voidaan asettaa vastaanottamaan päivityksiä haitakkeeseen hyökkääjältä [56]. Orjakoneen tavallisia käyttötapoja ovat haitakkeiden levittäminen, roskapostitus ja palvelunestohyökkäykset.

Sosiaalisissa verkkosovelluksissa mediatiedostot ja linkit haitallisille verkkosivuille ovat haitakkeiden pääasiallinen levitystapa. Käyttäjän selain voidaan skriptauskielillä pakottaa lataamaan haitake tietojärjestelmän ulkopuolisesta sijainnista. Tämä vaatii (XSS tai CSRF) *haavoittuvuuden* löytämistä tietojärjestelmästä. Helpoin tapa on tarjota linkki, joka vie haitalliselle verkkosivulle, ja sosioteknistä manipulointia käyttäen saada käyttäjä lataamaan tiedosto, joka sisältää haitakkeen.

Käyttäjät voivat *suojautua* haitakkeilta asentamalla virustorjuntaohjelmiston. Koska virustorjuntaohjelmistot perustuvat haitakkeiden **tuntomerkkien** (engl. *signature*) löytämiseen, ne eivät löydä uusia haitakkeita, joiden tuntomerkkejä niiden tietokannassa ei ole [83]. Riskien minimoimiseksi on tärkeää päivittää virustorjuntaohjelmisto mahdollisimman usein. Käytön helpottamiseksi on suositeltavaa automatisoida päivitykset. On huomioitava, että jotkut haitakkeet kykenevät alasaja-

maan koneen *suojaukset*. Tämä usein tarkoittaa virustorjuntaohjelmiston sulkemista ja käyttöjärjestelmäpäivitysten estämistä [83].

Kehittäjät voivat skannata kaikki tiedostot, jotka käyttäjät lisäävät sosiaaliseen verkkosovellukseen. Tämä lisää käsittelykustannuksia ja hidastaa järjestelmän toimintaa. Tämä ratkaisu ei auta uusiin haitakkeisiin, järjestelmän ulkopuolelle vieviin linkkeihin tai tapauksiin, jossa saastunut käyttäjä kirjautuu järjestelmään.

Riski, että sosiaalinen verkkosovellus joutuu alttiiksi haitakkeille on suuri. Tästä syystä on tärkeää tutustua kuinka rajoittaa ja estää haitakkeen leviäminen. Kehittäjät voivat tutkia haitakkeiden *propagaatiota* ja kehittää tämän pohjalta aikaisia varoitusjärjestelmiä [83]. Käyttäjien yhteistoiminta voi toimia aikaisena hälytysjärjestelmänä. Muodostamalla kanavia, joita pitkin käyttäjät voivat varoittaa haitakkeista, voidaan haitakkeiden leviäminen estää tai hidastaa. Niin kauan kun käytössä olevat haitakkeiden levittämisstrategiat ovat tarpeeksi tehokkaita, hyökkääjät käyttävät niitä. *Suojausten* kehittämiseksi on tärkeää pysyä ajan tasalla uusimmista sosioteknisistä manipulointitavoista ja leviämisstrategioista, ja tarjota käyttäjille tietoa ja kanavia haitakkeiden tunnistamiseksi.

Ratkaisuna Abraham ym. [83] kehottavat organisaatioita lisäämään tietoisuutta sosioteknistä manipulointia vastaan. Tilannetta hankaloittavat menneet negatiiviset kokemukset valistuksen tehokkuudesta. Valistettavien suuri lukumäärä vaikeuttaa viestin saamista perille.

Usein haitake yrittää peitellä jälkiään välttyäkseen paljastumiselta. Näiden puolustautumisstrategioiden tunteminen auttaa haitakkeiden löytämisessä aikaisessa vaiheessa ja heuristiikkojen kehittämisessä, joiden tarkoitus on havaita haitakkeet tunnistamalla haitakkeille ominainen käyttäytyminen. [83]

6 Hyökkäysskenaariot

Tässä kappaleessa esitellään yksinkertainen sosiaalinen verkkosovellus, jonka tietoturvaa allekirjoittanut yrittää murtaa edellisessä kappaleessa käsitellyillä hyökkäyksillä.

6.1 Lähtöasetelma

Tavoitteena on luoda kuviteltu tilanne, jossa verkkosovellukseen tieten tahtoen jätetään haavoittuvuuksia ja pyrkiä hyödyntämään näitä hyökkäyksissä. Tämän jälkeen pohditaan ja mahdollisuuksien mukaan testataan kyseisille hyökkäyksille tarkoitettuja suojauksia. Kokeilua varten on luotu pelkistetty sosiaalinen verkkosovellus, joka sisältää joitain tavallisia sosiaalisen verkkosovelluksen piirteitä.

Kokeilun tulokset on jaettu 5 skenaarioon. Skenaariot ovat irrallisia ja toisistaan riippumattomia. Kaikki paitsi ensimmäinen skenaario havainnollistavat kappaleessa 5 käsiteltyjä hyökkäyksiä. Ensimmäisessä skenaariossa käsitellään tavanomaisia hyökkääjän toimia ennen varsinaisen hyökkäyksen aloittamista: tiedonkeruuta tietojärjestelmästä ja valmistautumista.

Aktiivisina toimijoina ovat hyökkääjä ja käyttäjä. Kun allekirjoittanut on sekä luonut sovelluksen että suorittaa hyökkäysskenaariot, on hyökkääjän roolia varovaisesti pohdittu. Hyökkääjän valistuneisuuden takia skenaariot ovat parhaimmillaan hyökkäysten ja vastaavien suojausten havainnollistamisessa. Skenaarioissa kuitenkin pohditaan, kuinka hyökkääjä voisi päätyä skenaarion alkuasetelmaan ja niihin valintoihin, joita skenaarioissa suoritetaan. Käyttäjällä ei useimmissa skenaarioissa ole suurempaa merkitystä, kuin sovelluksen toiminnan takaaminen.

Skenaariot muistuttavat tunkeutumistestauksia (ks. kpl 3.6.2), joissa [47, s53] *"arvioijat, usein tiettyjen rajoituksien alaisena, pyrkivät kiertämään tai murtamaan tietojärjestelmän suojaukset."* Rajoituksina toimii kunkin skenaarion tavoite, ts. hyökkääjän tavoite, sekä skenaarioiden pitäminen mustalaatikkotestauksena: Aiempaa tietämystä tietojärjestelmän sisäisestä rakenteesta ei saa hyödyntää ilman hyviä perusteluita. Testausympäristö on identtinen kussakin skenaariossa.

6.1.1 Testausympäristö

Skenaariot toteutettiin Jyväskylän Yliopiston tietokonelaboratoriossa. Kaikissa paitsi viimeisessä skenaariossa käytettiin kolmea konetta, jotka olivat palvelinkone, käyttäjäkone ja hyökkääjän kone. Kullekin koneelle asennettiin Windows XP SP 2-käyttöjärjestelmä. Palvelinkoneelle asennettiin verkkosovellus ja hyökkääjän koneelle skenaarioissa tarvittavat työkalut. Käyttäjäkoneen tarpeisiin riitti mainiosti parin vaihtoehtoisen selaimen asentaminen.

Käytössä oli yksi tehokkaampi kone, jossa ajettiin seitsemää virtuaalikonetta **VirtualBox** -ohjelmistolla. Kukin virtuaalikone oli identtinen lukuunottamatta MAC-osoitetta ja IP-osoitetta. Virtuaalikoneiden lisäksi käyttöön otettiin yksi tietokone. Kaikissa koneissa paitsi virtuaalikoneita ajavassa koneessa käyttöjärjestelmänä oli Windows XP SP 2. Virtuaalikoneita pyörittävässä tietokoneessa käyttöjärjestelmänä oli Windows 7.

Tietokoneet yhdistettiin kytkimellä ja asetettiin lähiverkkoon saman aliverkon alle. Testausympäristö oli suljettu; internetyhteyttä ei ollut käytettävissä.

6.2 Sovelluksen esittely

Luvun alussa tutustutaan sovelluksen teknologiaratkaisuihin, tämän jälkeen tutustutaan ohjelmistokehykseen ja sen merkitystä tietoturvalle. Luvun lopussa kuvataan sovelluksen toimintaa.

6.2.1 Sovelluksen teknologiaratkaisut

Sovellukseen on otettu samoja web-teknologioita, jotka usein ovat käytössä Web 2.0:n yhteydessä. Ajaxin käyttö mahdollistaa tavallista interaktiivisemman kanssakäymisen käyttäjän kanssa. Lukuisat tietokantakyselyt, erityisesti SQL:n käyttö, ovat tavanomaisia. Tavallista on myös monien valmiiden teknologiaratkaisujen yhdessäkäyttö: Verkkosovelluksen pohjaksi on otettu WAMP-ohjelmistokokonaisuus.

WAMP-ohjelmistokokonaisuus tarkoittaa verkkosovelluksen ajamista Windows-käyttöjärjestelmässä, Apache-verkkopalvelimen päällä, MySQL:n toimiessa tietokantarajapintana ja PHPn ollessa skriptauskieli verkkosovelluksen dynaamisille tarpeille. WAMP-ohjelmistokokonaisuuteen voi sisältyä eri skriptauskieliä (Perl, Python).

LAMP-ohjelmistokokonaisuus on yleinen ratkaisu monissa verkkosovelluksissa. LAMP on kokoelma avoimen lähdekoodin ohjelmia, joka on suosittu juuri ilmai-

suutensa, helppokäyttöisyytensä ja joustavuudensa takia. LAMP eroaa WAMPista käyttöjärjestelmän suhteen. LAMPissa käytössä on Linux. Windows on valittu Linuxin sijasta allekirjoittaneen vähäisen Linux kokemuksen seurauksena.

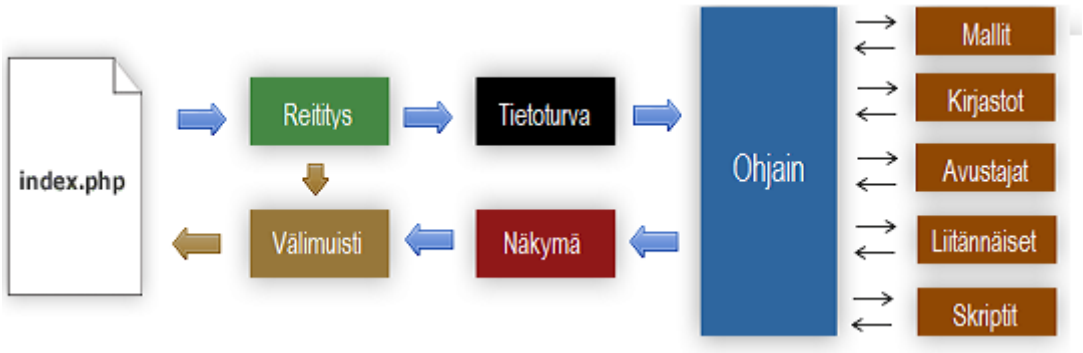
Varsinaisen sovelluksen kehityksessä on hyödynnetty ohjelmistokehystä, joka esitellään seuraavassa kappaleessa. Ohjelmistokehys on valittu niiden käytön yleisyyden vuoksi sekä nopeuttamaan kehitystyötä. Näin skenaarioissa tulee lisäksi arvioitua käytetyn ohjelmistokehyksen merkitystä sovelluksen tietoturvalle. Kattavan ohjelmistokehyksen tulisi huomioida myös sovelluksen tietoturva.

6.2.2 CodeIgniter -ohjelmistokehyksen esittely

CodeIgniter on PHP:lla toteutettu avoimen lähdekoodin ohjelmistokehys. CodeIgniter on luotu nopeuttamaan sovelluskehitystä tarjoamalla kattavan joukon valmiita kirjastoja, funktioita ja muita apukeinoja kehittäjälle. CodeIgniter pakottaa kehittäjän luomaan sovelluksen **MVC** (engl. *Model-View-Control*) -ajattelun pohjalta. MVC on eräs tapa toteuttaa ohjelmiston sisäinen rakenne, arkkitehtuuri. MVC jakaa sovelluksen kolmeen osa-alueeseen, pyrkien eristämään kunkin toisistaan, vähentäen niiden välistä riippuvuutta. Riippuvuuden vähentäminen mahdollistaa kunkin osakomponentin mahdollisimman itsenäisen kehittämisen ja testaamisen. Etuna tästä on myös se, että yhden osakomponentin muokkaaminen tai vaihtaminen vaatii vain vähän muutoksia sovelluksen muihin osakomponentteihin. [MVC-lähde]

MVC:n sovellusaluejako selviää termin muodostavista kirjaimista: **Model** (engl. *Malli*), **View** (engl. *Näkymä*) ja **Control** (engl. *Ohjain*). Malli pitää sisällään sovelluksen datan ja logiikan siitä, kuinka dataa käsitellään. Näkymälle kuuluu sovelluksen käyttöliittymään liittyvät tehtävät. Ohjain luo, poistaa ja hallinnoi mallia ja näkymää (tai malleja ja näkymiä, arkkitehtuurista riippuen). [MVC-lähde]

Käydään läpi lyhyesti, kuinka CodeIgniter tulkitsee MVC:tä. CodeIgniterissä mallit toimivat tietokantarajapintoina. Näkymät ovat HTML-merkkauskielellä ja minimaalisella määrällä PHP:tä kirjoitettuja käyttöliittymän (verkkosivun) osia, kuten otsakkeita, sivupalkkeja ja leipätekstiosioita. Ohjaimet hallitsevat sovelluksen tietovuota ja suoritusjärjestystä. Vain ohjaimet voivat esittää näkymiä tai kutsua malleja. Yksinkertaistettuna ohjaimen voi ajatella toimivan näkymän ja mallin välissä, pyytäen, saaden ja välittäen dataa eteenpäin. Vaikka kokonaisuus on monimutkaisempi, kehittäjän kannalta oleellisimmat osakomponentit ovat mallit, näkymät ja ohjaimet. Ohjelmistokehys huolehtii muista tukitoimista. CodeIgniterin tarkempi toiminta selviää kuvasta 7.1.



Kuva 6.1: CodeIgniter-sovelluskehyksellä toteutetun sovelluksen arkkitehtuurikaavio.

Ohjelmistokehys suorittaa joitain pakollisia tehtäviä, ennen kuin kontrolli annetaan ohjaimille. Näihin tehtäviin kuuluu vaihtoehtoisen reitityksen käsittely, joka sivuutetaan tässä, ja tietoturvakerroksen toimintojen käsittely. *Tulee huomioida, että kuvan 7.1 "tietoturva"-laatikko esittää vain joitain ohjelmistokehyksen tietoturvaa parantavia toimintoja*, ei sovelluksen tai tietojärjestelmän tietoturvaa sen laajemmassa merkityksessä. Seuraavassa kappaleessa tarkastellaan näitä toimintoja tarkemmin, sekä URL:n muodostusta ohjelmistokehyksessä. Kuvan 7.1 oikeassa laidassa näkyy mallin lisäksi ohjelmistokehyksen tarjoamat kirjastot ym. apukeinot, joista kukin täytyy erikseen ottaa käyttöön ohjaimissa ohjelmistokehyksen rajapintoja käyttäen. Ratkaisu on hyvä, koska tällöin turhat toiminnallisuudet eivät kuormita sovellusta. Kun ohjain on käsitellyt sivun ja mahdollisen syötteen, valmistelee se näkymän ja esittää sen käyttäjälle. Sovellus voi tallentaa välimuistiin käsiteltyjä sivuja, joka vähentää usein ladattujen sivujen tuottamaa kuormaa sovellukselle.

6.2.3 CodeIgniter-ohjelmistokehyksen tietoturvaa parantavat ominaisuudet

CodeIgniterissä on joitain sisäänrakennettuja tietoturvallisuutta parantavia ominaisuuksia. Käydään läpi URL:ien ositus, oletusasetuksia ja sitä, kuinka CodeIgniter tukee syötteen kelvolliseksi todentamista.

Eniten käyttöä ohjaava ominaisuus on **URL:ien ositettu käsittely**. Ohjelmistokehys pyrkii siistimään tällä epäselvät ja pitkät URL:it. Pitkät ja sekavat URL:it ovat tietoturvariski käyttäjälle, kun selaimen osoiterivillä on turhan pitkä ja sekava merkkijono. Tietoturvan kannalta tärkein on syötteen annon rajoittaminen URL-osoitteen kautta. Tätä varten HTML-protokollan GET-metodi on poistettu käytöstä CodeIgniter-ohjelmistokehyksessä. URL-osoitteisiin sallitaan vain alfanumeeristen merkkien lisäksi tilde (~), piste, pilkku, alaviiva ja viiva. Tämä rajoittaa ei-toivotun syötteen

pääsemistä ohjaimiin. Parametrit välitetään yhä URL:issa, mutta hieman eri tavoin. URL:in ositettu käsittely puretaan CodeIgniterissä seuraavasti:

```
http://www.esimerkki.com/index.php/ohjain/funktio/param1/param2/
```

Toimialueen jälkeen URL-osoitteessa on tiedoston nimi (`index.php`), ohjaimen nimi, funktion nimi ja lopuksi parametrit. Yo. esimerkki kulkee `index.php` -tiedoston läpi, joka sisältää ohjelmistokehyksen alustavia tukitoimintoja, jonka jälkeen suoritusjärjestys hyppää *"ohjain"*-ohjaimeen ja täältä funktioon *"funktio"*. Ohjaimessa käytettävissä ovat parametrit *"param1"* ja *"param2"*. Parametrejä voi olla useampi kuin esimerkissä olevat kaksi. CodeIgniter sallii tiettyä joustavuutta. Funktiota tai parametreja ei tarvitse antaa URL:issa. Jos funktiota ei määritellä, suoritetaan ohjaimesta `index()`-funktio. Lisäksi kehittäjällä on käytössään joitain vaihtoehtoisia reititysmahdollisuuksia, joiden tarkempi käsittely sivuutetaan tässä.

Verkkopalvelimesta riippuen aina ajettava `index.php` voidaan häivyttää, joka selkeyttää URL:eja entisestään. CodeIgniter mahdollistaa tiedostotarkentimien (`html, php, ..`) lisäyksen tai HTML:n GET-metodien sallimisen. Tässä työssä `index.php` on häivytetty URL-osoitteesta, mutta muuten URL:ien käsittely tapahtuu CodeIgniterin oletusasetuksien mukaisesti. Tällöin URL muodostuu seuraavasti:

```
http://www.esimerkki.com/ohjain/funktio/param1/param2/
```

CodeIgniterin **tietoturvaa parantavia oletusasetuksia** ovat seuraavien yleisesti vaikeasti hallittavien PHP-direktiivien muokkaaminen: `register_globals` ja `magic_quotes_runtime`.

- `register_globals` on PHP-direktiivi, joka luo koodiin automaattisesti joitakin muuttujia. Koska PHP-kieli ei vaadi muuttujien alustusta, kehittäjä joka ei ole tietoinen `register_globals` -direktiivin toiminnasta, ei voi olla täysin varma muuttujien oikeista arvoista koodia suoritettaessa. `register_globals` asetetaan CodeIgniterissa pois päältä. `register_globals` on asetettu pois päältä uusimmissa PHP-versioissa (4.2.0 lähtien). [phpnet http://www.php.net/]

- `magic_quotes_runtime` on PHP-direktiivi, joka korvaa ulkopuolisesta syötteestä lainausmerkit (`"`) kenoviivoilla (`\`). PHP-direktiivi aiheuttaa sekaannusta, koska joissakin verkkopalvelimissa ko. PHP-direktiivi on oletusarvoisesti päällä ja joissakin pois päältä. Kehittäjä, joka ei huomioi tätä, ei voi olla täysin varma syötteen oikeasta muodosta. `magic_quotes_runtime` on asetettu pois päältä CodeIgniterissa.

Eräitä CodeIgniterin **syötteen kelvolliseksi todentavia ominaisuuksia** ovat sisäänrakennettu XSS-suodatin, CSRF-suojaus ja parameterisoitujen SQL-kyselyiden tuki. Koska turhat toiminnallisuudet aiheuttavat yleisrasitetta sovellukselle eikä kehittäjää haluteta pakottaa tiettyyn suunnittelumetodologiaan, oletusarvoisesti näistä mikään ei ole käytössä. CodeIgniter-ohjelmistokehyksen suunnittelijat suosittelevat vahvasti syötteen kelvolliseksi todentamista jollain tapaa, kuten toivottavaa on missä tahansa verkkosovelluksessa. Näitä ominaisuuksia käsitellään tarkemmin skenaarioissa, joissa ne toimivat asianmukaisina suojauksina.

CodeIgniter sisältää muita tietoturvaa parantavia ominaisuuksia kuten kryptografisia funktioita. Näiden ominaisuuksien käsittely sivuutetaan tässä työssä.

6.2.4 Sovelluksen toiminta

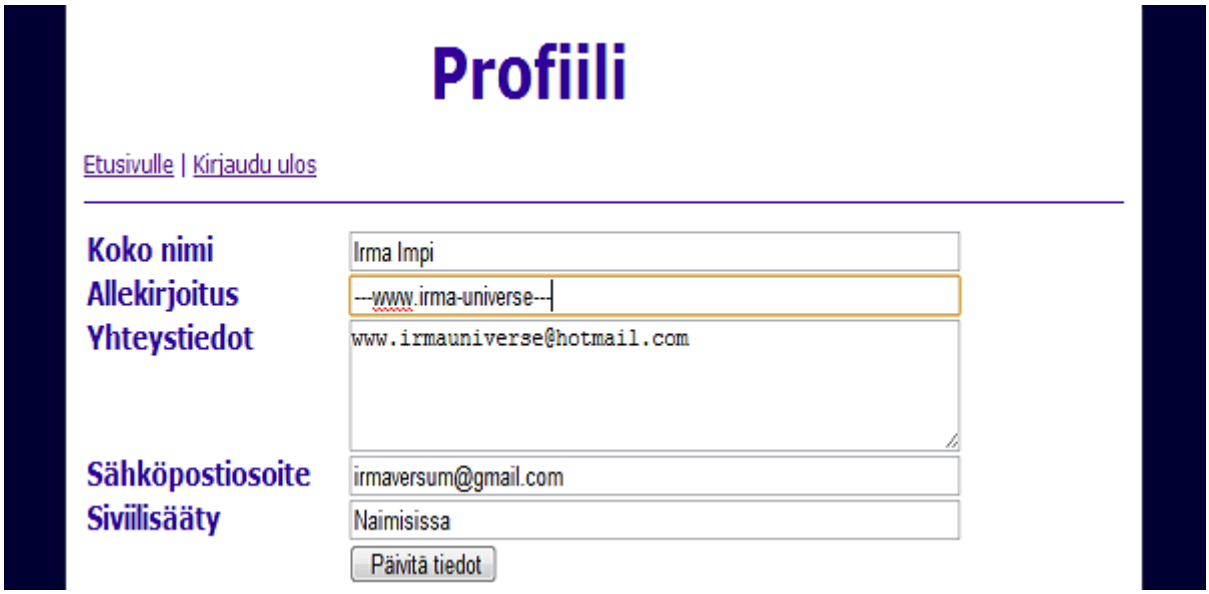
Käydään läpi testisovelluksen toimintaa tutustumalla sovelluksen käyttöliittymään ja siihen, kuinka se liittyy taustalla oleviin malleihin, näkymiin ja ohjaimiin. Testisovellus sisältää kolme sivua:

- Viestisivu (ks. kuva 7.2) on sovelluksen sivu, jossa käyttäjät viettävät suurimman osan ajastaan. Viestisivulla on kaksi instanssia, joista asianmukainen esitetään riippuen siitä, onko käyttäjä kirjautunut järjestelmään. Sivu sisältää hieman Twitteria muistuttavan julkisen ja jaetun keskustelualueen. Vain järjestelmään kirjautuneet käyttäjät voivat lisätä viestejä keskustelualueelle. Käyttäjien julkaisemat viestit lisätään sivulle siten, että uusimmat viestit ovat ylimpinä. Viestit lisätään Ajaxia käyttäen, jos vain käyttäjän selain tukee sitä. Käyttäjille sallitaan hyperlinkkien tarjoaminen. Tästä syystä kaikki muut paitsi `<a>`-tunnisteet poistetaan käyttäjien syötteestä. Kukin viesti on rajoitettu 140 merkkiin. Viestisivu sisältää linkit *"Profiilin muokkaussivulle"* ja uloskirjautumiselle. Käyttäjän kirjautuessa ulos, tämän istunto poistetaan ja käyttäjä ohjataan samaisen sivun toiseen instanssiin. Kuvasta 7.2 selviää sivun ulkoasu, kun käyttäjä on kirjautuneena järjestelmään. Uloskirjautuneena käyttäjä näkee vain viestit ja *"kirjaudu sisään"*-linkin.

- Profiilin muokkaussivu (ks. kuva 7.3) sisältää lomakkeen, jossa käyttäjä voi muokata profiiliaan. Profiilin tietoja ei sovelluksessa hyödynnetä millään tavoin, ne ovat olemassa vain skenaarioita varten. Tiedot tallennetaan kuitenkin tietokantaan. Profiilin muokkaussivu ei sisällä yksityisyysasetuksia.

- Sisäänkirjautumissivu (ks. kuva 7.4) sisältää yksinkertaisen sisäänkirjautumislomakkeen.

Kuva 6.2: Testisovelluksen viestisivu.



Kuva 6.3: Testisovelluksen profiilin muokkaussivu.

Kuva 6.4: Testisovelluksen sisäänkirjautumissivu.

Malleja sovelluksessa on kaksi: `posts_model` ja `users_model`. `posts_model` malli toimii tietokantarajapintana viesteihin liittyvissä toimissa. `users_model` -malli toimii vastaavasti tietokantarajapintana käyttäjiin liittyvissä toimissa. Tietokanta sisältää relaatiotaulut istunnoille, käyttäjille ja viesteille. Tietokannassa salasanat ovat salakirjoitettu MD5-salausalgoritmilla. Tietokantaohjaimena toimii MySQLi.

Testisovellus sisältää neljä ohjainta. Koska sovellus sijaitsee verkkopalvelimella kansion `tupre` alla ja verkkopalvelimen IP-osoite on 192.168.1.4, näitä vastaavat ohjaimet, URL-osoitteet ja sivut ovat:

- `Posts`-ohjain, joka vastaa viestien lisäämisestä.
  `http://192.168.1.4/tupre/posts` ja
  `http://192.168.1.4/tupre/` (uudelleenohjaus)
  Viestisivu.

- `Profile`-ohjain, joka vastaa profiilin hallinnasta.
  `http://192.168.1.4/tupre/profile`
  Profiilin muokkaussivu.

- `Login`-ohjain, joka vastaa autentikoinnista.
  `http://192.168.1.4/tupre/login`
  Sisäänkirjautumissivu.

- `Logout`-ohjain, joka vastaa istunnon tuhoamisesta ja uudelleenohjauksesta.
  `http://192.168.1.4/tupre/logout`
  Viestisivu (uudelleenohjaus).

Koska uloskirjautumissivua ei sovelluksessa ole, tällä ei ole näkymää. Kullekin aiemmin esitetylle sivulle, mukaanlukien sivun ylä- ja alapalkit, on oma näkymänsä. Sovelluksessa lisätään käyttäjien viestit Ajaxia käyttäen, ts. verkkopalvelimelta ladataan vain vaihtuneet osiot (viestit). Tätä varten sovelluksessa on lisäksi näkymä tämän mahdollistamiseksi.

6.3 Skenaario: Kohteen tiedustelu

Tässä työssä tiedustelu jaetaan kahteen eri joukkoon, sen mukaan voiko tiedustelua havaita vai ei. Skenaariossa suoritetaan kohteen verkkotiedustelu ja tarkastellaan saatuja tuloksia.

6.3.1 Hyökkäyksen tavoite

Foot printing [59, s8] ja **finger printing** -termit kuvaavat hyökkääjän ensimmäisiä toimia, sen jälkeen kun hyökkääjä on valinnut kohteensa. Hyökkääjä kerää tässä vaiheessa kohteesta informaatiota, jota hän voisi hyödyntää tulevissa hyökkäyksissä. Termillä **foot printing** [59, s8] viitataan informaationkeruuseen, joka on mahdollista suorittaa erinäisin keinoin, kohteen epäilyksiä herättämättä. Tässä vaiheessa kohteen tietojärjestelmään ei tunkeuduta. Samaisessa lähteessä [59, s44] termillä **scanning** (engl. *verkkotiedustelu*) viitataan aktiivisiin, tietojärjestelmässä mahdollisesti havaittaviin, tiedustelutoimiin. Termejä selittämään lainataan analogiaa, joka lähteessä esitetään: *"Jos foot printing on yhtä kuin informaation paikallistaminen, niin verkkotiedustelu on yhtä kuin seinien koputtaminen ikkunoiden ja ovien löytämiseksi."* Tässä skenaariossa kiinnitämme enemmän huomiota jälkimmäiseen vaiheeseen.

Julkisesti saatavilla olevan informaation läpikäyminen on eräs foot printingin vaihe. Organisaation verkkosivut, lehdistä tai muualta saadut yhteystiedot, työntekijöiden tiedot, puhelinnumerot ja sähköposti-osoitteet mm. kiinnostavat hyökkääjää. Organisaation toimialueen läpikäyminen voi paljastaa huonosti ylläpidettyjä etäyhteys-, asiakastuki- tai muita verkkopalveluita. Hakukoneiden välimuisteihin on voinut jäädä informaatiota, joita hyökkääjä voi hyödyntää. Webin keskustelualueista voi löytää apua tarvitsevia organisaation työntekijöitä, jotka viesteissään paljastavat jotain organisaation tietojärjestelmistä, esimerkiksi jotain tietoturvakäytännöistä tai muunlaista arkaluontoista informaatiota. WHOIS-hakupalveluilla voidaan selvittää kohteen toimialueet ja näistä edelleen voidaan DNSllä (internetin nimipalvelujärjestelmä) selvittää toimialueiden taustalla olevan tietojärjestelmän topologiaa, sekä toimialueita vastaavat IP-osoitteet. [59]

Tässä skenaariossa oletamme, että hyökkääjä on jo suorittanut foot printing -vaiheen ja saanut selville verkkopalvelimen IP-osoitteen. Testiympäristön asetelman ollessa kovin pieni ja verkon suljettu, ei ole kovin mielekästä suorittaa tätä vaihetta.

Verkkotiedustelun tavoitteena on saada selville mahdollisimman paljon informaatiota tietojärjestelmästä. Tavallisesti tämä tarkoittaa verkkolaitteiden ja topologian, auki olevien porttien ja käyttöjärjestelmien tunnistamista. Näiden tietojen avulla hyökkääjä voi selvittää tietokannasta kuten *OpenSourceVulnerabilityDatabasesta*, sisältääkö tietojärjestelmä joitain tunnettuja haavoittuvuuksia. Hyökkääjän näkökulmasta verkkotiedustelu voidaan mieltää onnistuneeksi, jos tietojärjestelmästä löytyy haavoittuvuuksia ja hyökkääjän anonymiteetti ei vaarannu. Riippuen tiedustelun tunkeilevuuden tasosta, hyökkääjästä jää joitain jälkiä, tavallisesti lokitiedostoihin. Verkkotiedustelu voi jäädä huomioimatta. Paikoillaan olevat suojausmekanismit, henkilöstö ja organisaation toimintatavat määräävät pannaanko verkkotiedustelua merkille.

6.3.2 Toteutus

Palvelinkoneella on Windows XP -käyttöjärjestelmän Service Pack 2-laajennuksessa tullut palomuuri. Koska topologia on hyvin yksinkertainen (1 hyppy palvelinkoneelle), topologiaa ei tiedusteltu skenaariossa. Verkkopalvelimelle jätettiin oletusasetukset mutta `.htaccess` -tiedostolla rajoitettiin verkkosovelluksen hakemistorakenteen selaamista. Skenaariossa ei tarkkailtu työkalujen jättämiä jälkiä tietojärjestelmään.

Skenaariossa käytettiin neljää eri työkalua: **Zenmappia**, **wiktoa**, **superscannia** ja **webscarabia** [1]. Kukin työkalu ajettiin ensin mahdollisimman kevyillä asetuksilla ja tämän jälkeen kaikkein raskaimmilla asetuksilla. Kukin työkalu on verkkokehittäjien ja tietoturva-asiantuntijoiden käyttämiä työkaluja tietoverkkojen tietoturvakatselmointiin. Ideaaliset murtotestaustyökalut pyrkivät käyttämään samoja metodeja kuten hyökkääjät, jotta murtotestaukset olisivat mahdollisimman autenttisia.

Työkalut väärinkäyttävät verkkostandardien ja -protokollien kommunikaation avoimuutta ja automaatiota. Joitain tavallisia keinoja ovat HTMLn GET, POST, TRACE tai muiden metodien käyttö ja palvelupyyntöjen lähettäminen verkkopalvelimen portteihin. Työkaluista kaikki paitsi webscarab ovat aktiivisia skannaustyökaluja.

[1] Työkalut löytyvät vastaavasti: Zenmap (http://nmap.org/zenmap/), wikto (http://www.sensepost.com/labs/tools/pentest/wikto), superscan (http://www.mcafee.com/us/downloads/free-tools/superscan.aspx) ja webscarab (http://dawes.za.net/rogan/webscarab/)

Webscarab toimii **man-in-the-middle** periaatteella. Kaikki verkkoliikenne kaapataan webscarabin välityspalvelimelle, jossa se analysoidaan ja tavallisesti ohjataan muuttumattomana eteenpäin.

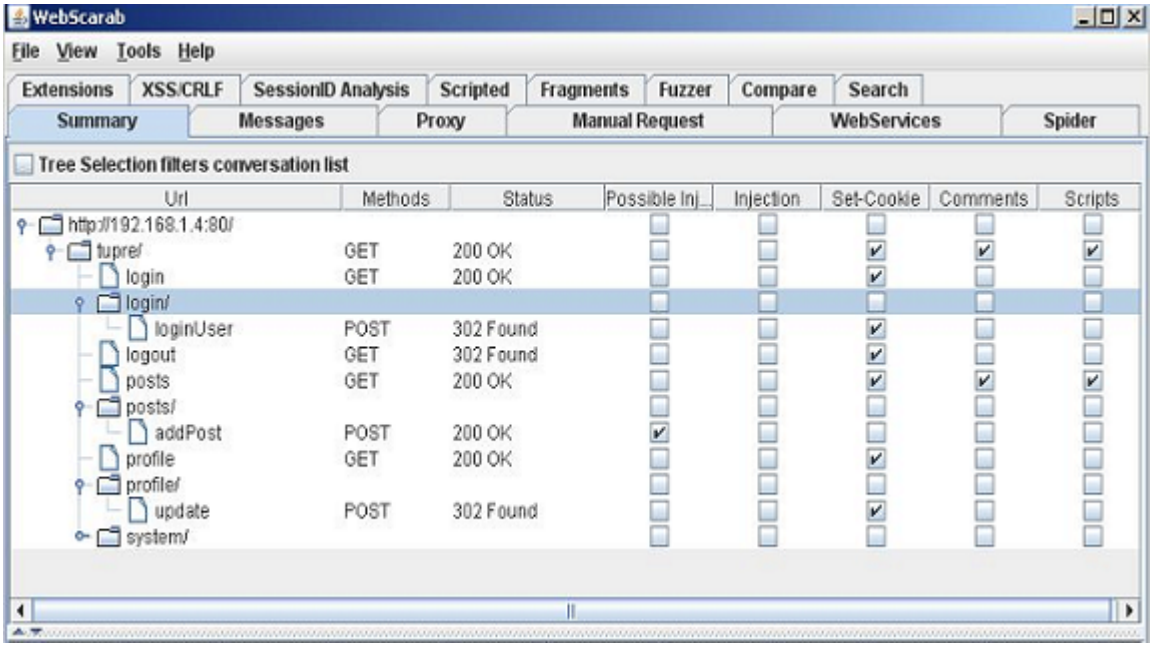
6.3.3 Tulokset

Kaikki työkalut mahdollistivat palvelimen tyypin sekä skriptauskielen käytön selville saannin. Kukin työkalu sai selville lisäksi oikeat versiot (Apache/2.2.11 (Win32) ja PHP/5.3.0). Kaikki auki olevat portit ja verkkopalvelimella olevat palvelut saatiin selville. Webscarab sai selville myös käytetyn JavaScript-kirjaston, *jQueryn*. Zenmap ja superscan saivat selville lisäksi käytetyn käyttöjärjestelmän. Jos verkkosovellukselle lähettää HTTP-pyynnön `/phpmyadmin/`, sovellus vastaa, että käyttäjällä ei ole oikeutta kyseiseen kansioon. Koska *phpMyAdmin* on MySQL-tietokannan hallintatyökalu, voidaan tästä päätellä, että sovellus käyttää SQL-kieltä. Näillä tiedoilla hyökkääjä on saanut selville käytetyn ohjelmistokokonaisuuden: WAMP.

Vaikka CodeIgniter ja `.htaccess` -tiedosto rajoittavat hakemistorakenteen selausta, pystyivät wikto ja webscarab selvittämään jossain määrin verkkosovelluksen hakemistorakenteen. Webscarab sai selville, että verkkopalvelimella on hakemiston system alla kansiot `system`, `css`, `js` ja `pics`. Skenaarion toteutushetkellä Google-haun *"Directory structure system css js pics"* kolmesta hakutuloksesta ensimmäiset kaksi sisälsivät vahvoja viitteitä CodeIgniter- ohjelmistokehykseen.

Webscarab ja monet selaimien laajennokset sallivat keksien selaamisen. Kun keksien nimeksi oli jätetty CodeIgniterin oletusnimi, `ci_session`, voidaan tästä päätellä, että sovelluksessa hyödynnetään CodeIgniter-ohjelmistokehystä. Tämä tieto yhdessä hakemistorakenteen selville saamisen kanssa helpottaa hyökkääjää suorittamaan tulevia hyökkäyksiä, koska hyökkääjällä on nyt tietoa sovelluksen sisäisestä rakenteesta, tietoturvamekanismeistä ja mahdollisista paikallaan olevista oletusasetuksista.

Mahdollisina haavoittuvuuksina wikto löysi HTTP:n TRACE metodin väärinkäyttöä hyödyntävän haavoittuvuuden, XSS-haavoittuvuuden, HTML-injektiohaavoittuvuuden ja SQL-injektiohaavoittuvuuden. Lisäksi foot printing -vaiheeseen mainittakoon, että superscan tarjosi automatisoituja WHOIS -kyselyitä, joilla tuntemattoman toimialueen IP-osoite voidaan selvittää.



Kuva 6.5: Webscarab saa selville hakemistorakennetta ja uudelleenohjauksia (HTTP 302 -tilakoodit).

6.3.4 Suojautuminen

Verkkopalvelimen ja skriptauskielen (PHP) selville saamista voi olla liki mahdotonta häivyttää, koska se ilmoitetaan kussakin HTTP-vastauksessa. Auki olevien porttien rajaaminen yhteen vaikeutti zenmapia saamasta käyttöjärjestelmää selville. Tämä on miltei aina kuitenkin mahdoton ratkaisu ja tuskin ikinä kannattavaa, ottaen huomioon ratkaisun kustannukset. Jos *Netbios*-palvelu oli Windowsissa päällä, superscan selvitti nopeasti käyttöjärjestelmän.

Hyvä ratkaisu on oletusasetusten muuttaminen, sillä monet automatisoidut työkalut pyrkivät juuri hyödyntämään niiden olemassaoloa. CodeIgniterin oletushakemistorakennetta muuttamalla voidaan suojata tietojärjestelmää ainakin automatisoiduilta hyökkäyksiltä. Suositeltavaa on myös keksien nimen vaihtaminen. Tällöin ohjelmistokehyksen selville saaminen vaikeutuu huomattavasti.

Foot printing -vaiheen suojautumiskeinoja ovat julkisesti saatavilla olevan informaation läpikäyminen ja arviointi, unohtamatta sisar- ja lapsiorganisaatioiden paljastamaa informaatiota. Työntekijöiden tietoturvakoulutuksella voidaan tarkentaa mitä tietoa esimerkiksi puhelimitse tai sähköpostitse voidaan luovuttaa. Jotkut toimialueen nimen antajat tarjoavat anonymiteetin tarjoavia palveluita [59, s33].

6.4 Skenaario: Cross Site Scripting (XSS)

Skenaariossa hyödynnetään yleisiä XSS- hyökkäysten tekniikoita. Tämän jälkeen esitellään työkalu, jota skenaariossa käytettiin mahdollisten XSS-haavoittuvuuksien etsimiseen. Lopulta analysoidaan skenaarion tulokset.

6.4.1 Hyökkäyksen tavoite

Skenaarion tavoitteena on saada syötettyä skriptauskieltä sovellukselle siten, että sovellus suorittaa koodin. Varsinaista haitallista koodia tietojärjestelmään ei syötetä, vaan tarkoituksena on päästä sovelluksen paikoillaan olevien suojausten ohi. Jos hyökkäyksen tavoitteeseen ei päästä, suojausten tasoa alennetaan, kunnes ajettavaa koodia saadaan syötettyä sovellukseen. Skenaarion XSS-hyökkäys lasketaan onnistuneeksi, jos syötetty koodi avaa viesti-ikkunan tekstillä `XSS`. Ehkä tavanomaisin tapa tulostaa yo. viesti-ikkuna JavaScriptilla on syöttää komento:
`<script>alert("XSS")</script>`

Hyökkääjä voisi sokeasti kokeilla eri XSS-hyökkäyksiä esimerkiksi automatisoidun työkalun avulla. Olettaen, että hyökkääjällä ei ole käytössään orjakoneita, eräs anonymiteettiä suojaava keino olisi selvittää paikoillaan olevat suojaukset, haravoida kohdesivusto täysin ja luoda mahdollisimman samankaltainen kopio omalle paikalliselle palvelimelle. Hyökkääjä voisi tällöin rauhassa testata erilaisia hyökkäyksiä ja selvittää, mikä XSS-hyökkäys toimii. Hyökkäys voi epäonnistua silti, jos sovelluksen syötteen käsittely ei välttämättä ole odotetun kaltainen, johtuen muutoksista tai hyökkääjän vääristä johtopäätelmistä.

6.4.2 Toteutus

Apuna skenaariossa käytettiin OWASP-organisaation kotisivuilta löytyvää **Cal9000-sovellusta** [2] joka hyödyntää nk. *"XSS-cheatsheettia"*. Cal9000 on tietoturvakatselmointiin luotu sovellus, joka sisältää mm. työkalun injektiohyökkäysten suorittamiseen. Hyökkäys voi epäonnistua selainten eri tapoihin käsitellä syötettä tai niiden omiin suojauksiin. Yhtälailla selaimien tapa käsitellä syötettä eri tavoin voi olla hyökkäykselle eduksi. Cal9000:en erilaiset merkistökoodaustyökalut ovat tärkeitä, kun yritetään muotoilla syötteestä sen muotoista, että se läpäisisi kaikki suojaukset.

Koska hyökkäykset voivat olla selainkohtaisia, XSS-hyökkäykset testattiin internet Explorerilla, Mozilla Firefoxilla, Google Chromella ja Operalla. Kustakin selaimesta käytössä oli silloisin uusin versio. Suojauksena sovelluksessa oli CodeIgnite-

[2] Cal9000 löytyy täältä http://www.owasp.org/index.php/Category:OWASP_CAL9000_Project

rin oma XSS-suodatin sekä kaksi funktiota, `strip_tags()` ja `trim()`. Funktiota `strip_tags()` kutsuttiin parametrilla <a>, jolloin funktio riisuu syötteestä kaikki hakasulkeet, lukuunottamatta `<a>` -tunnisteita. `<a>` -tunnisteet sallivat käyttäjien lisätä keskustelualueelle hyperlinkkejä. `trim()` -funktio poistaa syötteestä lopulta tyhjät merkit. Koska suojaukset olivat identtiset viestialue-sivulla (`posts`) ja profiilin muokkaussivulla (`profile`), XSS-hyökkäyksiä kokeiltiin vain viestialue-sivulla.

Pelkästään kokeilemalla eri syötteitä hyökkääjä voi kyetä arvaamaan, että sovelluksessa käytetään `strip_tags()` -funktiota (tai vastaavaa metodia). Jos sovelluksella on käyttöohjeet viestialueen sivulle, voi tästä edelleen selvitä, että käyttäjien sallitaan lisäävän vain hyperlinkkejä (ei sallita esimerkiksi upotettuja elementtejä). Tällöin hyökkääjä voi ottaa strategiakseen syöttää haitallinen koodi sallittujen `<a>`-tunnisteiden sisään. Erityyppisistä XSS-hyökkäyksistä testattiin pysyviä ja peilattuja XSS-hyökkäyksiä.

6.4.3 Tulokset

CodeIgniterin XSS-suojaus esti kaikki Cal9000:en XSS-hyökkäykset ja XSS-hyökkäykset, jotka muotoiltiin Cal9000:en merkistökoodaustyökaluilla. On huomattava, että Cal9000:n XSS-hyökkäyksiä ei ole päivitetty vuosiin. Tämän jälkeen CodeIgniterin XSS-suojaus poistettiin päältä.

Koska kaikki syöte tallennetaan tietokantoihin, ainoa mahdollisuus sisällyttää peilattuja XSS-hyökkäyksiä oli lisätä keskustelualueelle haitallisia hyperlinkkejä. Jos itse URL-osoite sisälsi haitallisen koodin, ohjelmistokehykseen sisäänrakennettu XSS-suojaus ei tehnyt merkkijonosta hyperlinkkiä. Koska testisovellus ei tarkista ulkopuolisten sivustojen vaarallisuutta, haitallista koodia sisältävän verkkosivun linkittämistä ei testisovelluksessa voida estää.

Odotettavasti `strip_tags` funktio esti tehokkaasti yksinkertaiset XSS-hyökkäykset. Ainoa toimiva strategia oli syöttää haitallinen koodi `<a>` -tunnisteiden sisään. Funktio oli kuitenkin helposti hämättävissä antamalla sille merkkijono, joka alkaa `<a>` -tunnisteella ja URL-koodaamalla haitallisen koodin. Esimerkiksi seuraava merkkijono toimii peilattuna XSS-hyökkäyksenä, liittäen viestialueelle haitallisen linkin:

```
<a href="javascript://www.anywhere.com/%0dalert("XSS")">linkki</a>
```

Kun sovellukseen saatiin syötettyä läpi oikeanmuotoinen XSS-hyökkäys, eivät jotkut selaimet kuitenkaan antaneet ajaa sitä. Erityisesti internet Explorerin XSS-suoja esti monet hyökkäykset. Cal9000:en internet Explorer-kohtaiset merkistökoo-

daukset olivat liian pitkiä tai eivät toimineet enää käytössä olevalla versiolla. Kun XSS-suoja kytkettiin selaimesta pois päältä, pysyvät XSS-hyökkäykset aktivoituivat.

6.4.4 Suojautuminen

Koska merkistökoodaus kasvattaa usein merkkijonon pituutta, syötteen pituuden rajaaminen estää tehokkaasti joitain merkistökoodaukseen luottavia XSS-hyökkäyksiä. Tästä syystä viestialueen 140 merkkiin rajoitettu tekstikenttä esti joidenkin merkistökoodattujen XSS-hyökkäysten suorittamisen. Vaikka profiilin muokkaussivua ei testattu skenaariossa, olisi se ollut paremmin turvassa XSS-hyökkäyksiltä, juuri tekstikenttien pienemmän koon suhteen. Edelleen, skenaariossa oleva merkistökoodattu koodinpätkä esittää vaarattoman viesti-ikkunan. Todellista vahinkoa tekevä haitallinen koodi on todennäköisesti pitempi.

Nykyään selaimissa voi olla omat XSS-suojauksensa, mikä vaikeuttaa XSS-hyökkäysten suorittamista. Kehittäjä ei kuitenkaan voi luottaa selainten XSS-suojauksiin sokeasti, vaan paikoillaan on oltava jonkinlainen mekanismi syötteen oikeaksi todentamiseksi.

Vaikka DOM-pohjaisia XSS-hyökkäyksiä ei suoritettu, tulisi CodeIgniter-ohjelmistokehyksen ositettu URL:n käsittely ja GET-metodien hylkääminen olla tehokkaita suojia DOM-pohjaisia XSS-hyökkäyksiä vastaan.

6.5 Skenaario: SQL-injektio

Tässä skenaariossa testataan erittäin yleistä koodi-injektiota: SQL-injektiota.

6.5.1 Hyökkäyksen tavoite

Skenaarion tarkoitus on saada syötettyä vahingollinen SQL-kysely tietojärjestelmään. Edelleen, jos vahingollista SQL-kyselyä ei saada syötettyä järjestelmään, suojauksia alennetaan, kunnes hyökkäyksen tavoite toteutuu.

Oleellista onnistuneille SQL-injektioille on tietokantojen sisäisen rakenteen selville saaminen valistuneilla arvauksilla tai verkkotiedustelulla. Ilman onnistunutta verkkotiedustelua tavallista on kokeilla usein käytettyjä tietokannan kenttien tai taulujen nimiä. Oikean SQL-syntaksin selville saaminen on myös tärkeää. Oikea syntaksi selviää käytetystä tietokannan hallintajärjestelmästä.

6.5.2 Toteutus

Apuna SQL-injektion syöttämisessä käytettiin sovelluksen omia syötekenttiä ja erilaisia automatisoituja SQL-injektiotyökaluja. Kokeiltuja työkaluja olivat **absinthe**, **sqlmap**, **pangolin** ja **bobcat** [3]. Sekä profiilin muokkaussivua että viestisivua käytettiin SQL-injektioiden testaamiseen.

Tietokannan hallintajärjestelmänä verkkosovelluksessa oli MySQLn versio 5.1.36. Suojauksina sovelluksessa olivat syötteen oikeaksi todentavat toiminnallisuudet (aiemmin esitellyt funktiot `strip_tags()` ja `trim()`) ja parameterisoidut SQL-kyselyt. Edellä mainitut funktiot olivat suojauksena vain viestisivulla.

6.5.3 Tulokset

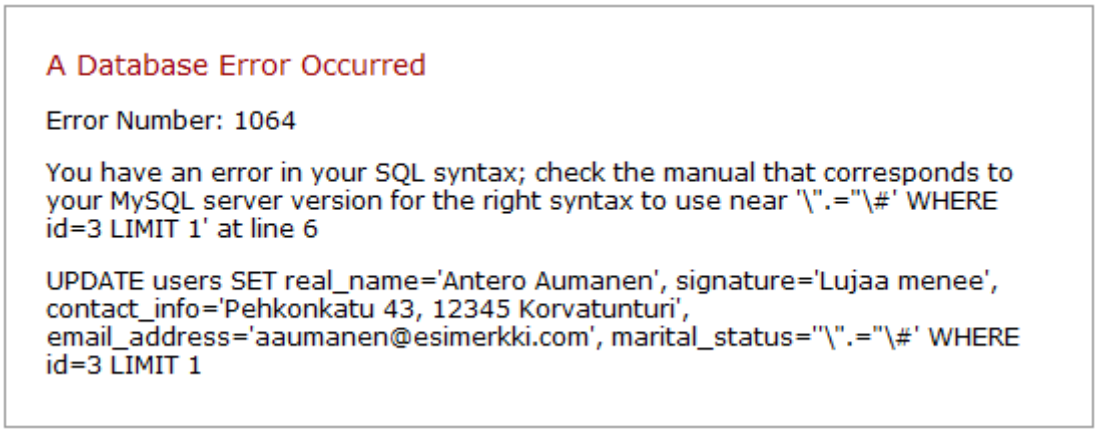
Parameterisoidut SQL-kyselyt osoittautuivat tehokkaaksi suojaukseksi. Nämä otettiin pian pois käytöstä. Seurauksena SQL-kyselyistä tuli dynaamisia, mahdollistaen koodin syöttämisen järjestelmään.

Käytettävät työkalut pärjäsivät huonosti skenaariossa. Ongelmalliseksi osoittautui sovelluksen uudelleenohjaukset, GET-metodin kieltäminen ja CodeIgniter-ohjelmistokehyksen URL:in ositettu käsittely. Jos sokeita SQL-kyselyitä saatiin ajettua, ne eivät tuottaneet onnistuneita SQL-injektioita.

CodeIgniterin oletusasetukset eivät salli useiden SQL-kyselyiden suorittamista yhdellä kerralla. Tästä syystä vain sovelluksessa esitettyjä SQL-kyselyitä voitiin hyödyntää (tässä tapauksessa INSERT ja UPDATE). Viestisivulla oli vain yksi yhteyspiste SQL-injektion syöttämiseen, kun taas profiilin muokkaussivulla yhteyspisteitä oli viisi. Viestisivun parempien suojausten takia ainoa allekirjoittaneen onnistunut SQL-injektio oli viestin aikaleiman muuttaminen halutulla tavalla. Tarkastellaan seuraavaksi profiilin muokkaussivun SQL-injektion yhteyspisteitä. Alla olevassa koodissa näkyy `user_model`-mallin dynaaminen SQL-kysely.

```
1 ····· $q = "UPDATE users SET¶
2 ·········· real_name='$realName',¶
3 ·········· signature='$signature',¶
4 ·········· contact_info='$contactInfo',¶
5 ·········· email_address='$emailAddress',¶
6 ·········· marital_status='$maritalStatus'¶
7 ·········· WHERE id=$userId LIMIT 1";¶
```

[3] Työkalut löytyvät: Absinthe (http://0x90.org/releases.php), Sqlmap (http://sqlmap.sourceforge.net/), Pangolin (http://www.nosec-inc.com/en/products/pangolin/), ja Bobcat (http://web.mac.com/nmonkee/pub/bobcat.html)



Kuva 6.6: Virhetulosteiden näkyminen käyttäjälle on vakava tietoturvariski.

Koodissa kukin viidestä muuttujasta toimii yhteyspisteenä SQL-injektiolle. Epähuomiossa profiilin muokkaussivulle oli jäänyt päälle asetus, joka näyttää käyttäjälle tietokannan virhetulosteet. Virhetulosteista ilmenee relaatiotaulun `users` sisäinen rakenne ja käytössä oleva tietokannan hallintajärjestelmä. Näillä tiedoilla hyökkääjälle selviää, kuinka hän voisi rikkoa dynaamisen SQL-kyselyn rakenteen, ja päästä syöttämään omaa koodia järjestelmään. Koska SQL-kyselyä ei voi vaihtaa, yritettiin WHERE -lause hävittää SQL-kyselystä. Tällöin MySQL-syntaksin mukaisesti UPDATE -komento vaikuttaa relaatiotaulun `users` kaikkiin riveihin.

WHERE -lausetta yritettiin hävittää SQL-kyselystä kommenttimerkeillä sekä karkaamalla merkkijonon kontekstista tilaan, jossa voidaan suorittaa PHPn **echo** komento. Skriptauskielen onnistunut ajaminen on itse asiassa XSS-hyökkäys. Yllättäen tehokkaana suojauksena toimivat rivinvaihtomerkit, jotka näkyvät yo. koodissa kunkin rivin päässä olevina merkkeinä. Uusien rivien luominen onnistui, mutta epäselväksi jäi, kuinka vaihtaa riviä.

Tämä skenaario osoittaa hyvin, miksi hyökkäyksiä voi olla vaikea kategorisoida: profiilin muokkaussivun hyökkäys hyödynsi XSS:ää ja SQL-injektiota. Huolimatta vakavista tietoturvariskeistä haitallisen SQL-injektion syöttäminen epäonnistui.

6.5.4 Suojautuminen

Parameterisoidut SQL-kyselyt, tallennetut proseduurit ja syötteen oikeaksi todentaminen ovat suojauksia SQL-injektiota vastaan. SQL-kyselyiden rajoittaminen vain yhteen kyselyyn kerrallaan toimii myös suojauksena. Syötteen oikeaksi todentamisessa tulee käyttää hyväksi havaittuja funktioita ja toimintatapoja. Syötteen oikeaksi

todentamisessa on tärkeää ymmärtää, kuinka sovellus käsittelee syötettä. Sovelluksen syötteen käsittelyä on hyvä testata, usealla eri selaimella.

Tietojärjestelmä kokonaisuutena lopulta määrittelee, kuinka syötettä käsitellään. Esimerkiksi UNIX- ja Windows-käyttöjärjestelmissä uuden rivin käsittelyyn liittyviä merkkejä **telan palautus** (engl. *carriage return*) ja **rivinvaihto** (engl. *linefeed*) käsitellään eri tavalla. Ilman hyvää kokonaiskuvaa tietojärjestelmästä turvallisinta on käyttää parameterisoituja SQL-kyselyitä tai tallennettuja proseduureja, jolloin virhetila syötteen käsittelyssä jää mahdollisimman pieneksi.

6.6 Skenaario: CSRF -hyökkäys

Sovelluksen elinkaaren alkuaikoina CSRF -hyökkäyksiin ei otettu kantaa. Ohjelmistokehyksestä ilmestyi 28.1.2011 uusi versio, jossa eräänä uutena toimintona esiteltiin CSRF -suojaus, jolloin CSRF-hyökkäysskenaario tuli ajankohtaiseksi.

6.6.1 Hyökkäyksen tavoite

CSRF-hyökkäyksessä tietojärjestelmän ulkopuoliselta toimialueelta lähetetään pyyntö käyttäjän puolesta. Oleellista on, että pystytään sovelluksen näkökulmasta esiintymään käyttäjänä, väärinkäyttäen käyttäjän voimassa olevaa istuntoa. Erityisesti tarkastelun alaisena on ohjelmistokehyksen CSRF-suojaus.

6.6.2 Toteutus

Koska CSRF-suojaus tuli ohjelmistokehykseen vasta versiossa 2.0, päivitettiin ensin sovelluksen ohjelmistokehys. Kun sovellus oli onnistuneesti siirretty uuteen versioon, varmistettiin CSRF-hyökkäyksen toimivuus. Lopulta tehtiin tarvittavat konfiguraatiot CSRF-suojauksen aktivoimiseksi, ja toistettiin hyökkäys.

Skenaariota varten käyttäjä kirjautui sisään sovellukseen. Virtuaalikoneella erillisellä palvelimella isännöitiin `csrf-testi.html` -tiedostoa, jossa käyttäjä vieraili, aktivoiden CSRF-hyökkäyksen. `csrf-testi.html` -sivu sisälsi näkymättömän lomakkeen, joka suoritti hyökkääjän haluamilla parametreillä käyttäjän profiilin muokkauksen. Täten skenaariossa toteutettiin peilattu CSRF-hyökkäys.

Alla näkyy osa `csrf-testi.html` -sivun koodista. Profiilin muokkaussivun HTML-lähdekoodin tutkiminen sisältää tarvittavat kenttien nimet (`name`-kentät). Hyökkääjä tarvitsee yhä tiedon, mikä URL-osoite suorittaa profiilin päivityksen. Tässä tapauksessa hyökkääjä pystyi saamaan tarvittavan funktion (`update`) nimen selville selaimella tai verkkotiedustelun avulla.

```
1  <form name="auto" action="http://192.168.1.4/tupre/profile/update"
2  method="POST">
3  <input type="hidden" name="real_name" value="cracker" />
4  <input type="hidden" name="signature" value="sig" />
5  <textarea name='contact_info'>Tervetuloa</textarea>
6  <input type="hidden" name="email_address" value="e@taalla.com" />
7  <input type="hidden" name="marital_status" value="?" /></form>
8  <script type="text/javascript" language="JavaScript">
9  document.auto.submit();</script>
```

6.6.3 Tulokset

Sovelluksessa ei vaadittu profiilin muokkaussivun kentiltä mitään arvoja. Tästä syystä sovelluksessa oli hyvin yksinkertainen CSRF-haavoittuvuus, jota voitiin hyödyntää `csrf-testi.html` sivustolla uudelleenohjaamalla käyttäjä osoitteeseen `http://192.168.1.4/tupre/profile/update`. Kun `update`-funktio ei saanut POST-arvoja, tämän seurauksena käyttäjän profiilin kaikki tiedot tyhjentyivät.

Alkuperäisen CSRF-hyökkäyksen ohjelmistokehyksen CSRF-suojaus esti onnistuneesti. CSRF-hyökkäys oli kelvollinen, sillä kun CSRF-suojaus kytkettiin pois päältä, hyökkäys saatiin ajettua onnistuneesti.

6.6.4 Suojautuminen

Yksinkertainen CSRF-haavoittuvuus paikattiin vaatimalla profiilin muokkaussivun lomakkeelta kenttien tiedot. Tämä onnistui hyödyntämällä CodeIgniter-ohjelmistokehyksen `form validation`-kirjaston apufunktioita.

Oleellista on vahvistaa palvelupyynnön tulevan sallitulta verkkosivulta (ks. kpl 5.2.7). CodeIgniterin CSRF -suojaus lisää kuhunkin lomakkeeseen kiistämättömyyden takaavan tunnisteen. On huomattava, että Ajax -pyyntöjen tapauksessa kehittäjän on liitettävä näihin CSRF-tunnisteen nimi, jotta CSRF-suojaus toimisi [4]. Skenaarion kirjoitushetkellä ohjelmistokehyksen ohjeista tämä ei tullut esille. Täten sovelluksen viestisivu sisältää CSRF-haavoittuvuuden, jos yo. paikkausta ei suoriteta.

CSRF-hyökkäyksistä vastuussa olevia on hankala jäljittää, kun tavallisten käyttäjien selaimet suorittavat hyökkäykset. Mahdollista on kirjata ylös lokiin käyttäjiä, jotka eivät läpäise CSRF-suojausta ja ottaa heihin yhteyttä. Erittäin kyseenalaista on kuitenkin, suostuvatko käyttäjät paljastamaan selaushistoriaansa. Menetelmä voi olla kelvollinen, jos CSRF-hyökkäykset aiheuttavat tarpeeksi vahinkoa sekä käyttä-

[4] http://aymsystems.com/ajax-csrf-protection-codeigniter-20.

jille että kehittäjille. Hyökkääjää ei voi saada näinkään selville, jos haitallinen koodi on syötetty CSRF-hyökkäyksen ajavalle sivustolle omistajien tietämättä.

6.7 Skenaario: Hajautettu palvelunestohyökkäys (DDoS)

Skenaario esittelee lyhyesti, kuinka hajautettu palvelunestohyökkäys voidaan toteuttaa ja kuinka sen vaikutukset ilmenivät sovelluksen toiminnassa.

6.7.1 Hyökkäyksen tavoite

Palvelunestohyökkäyksen tavoite on saada kuormitettua palvelinkonetta siten, että sovelluksen käyttäminen vaikeutuu huomattavasti. Tavoite mielletään onnistuneeksi, jos palvelimen tarjoamia sivuja joudutaan lataamaan yli viisi sekuntia tai jos sovellus alkaa hylkäämään merkittävästi HTTP-pyyntöjä.

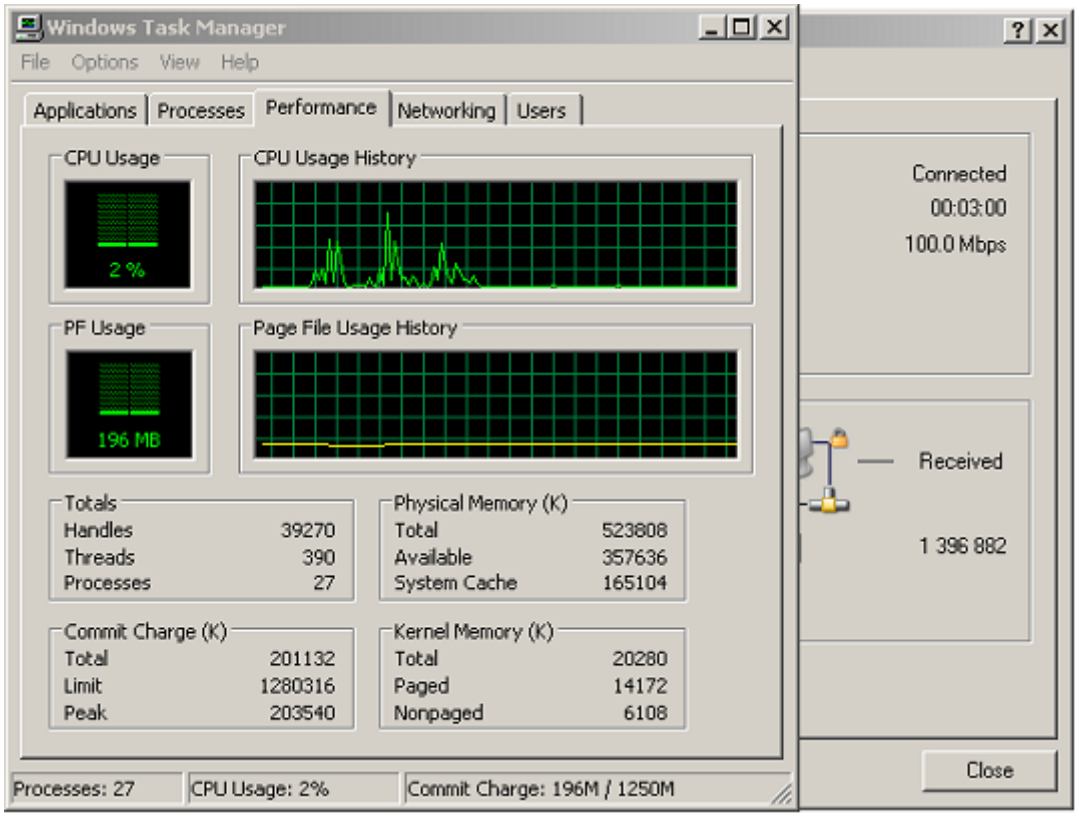
6.7.2 Toteutus

Hajautettuun palvelunestohyökkäykseen käytettiin avoimen lähdekoodin **HyenaeFE**-työkalua. Sovelluksella käynnistettiin **taustaprosessi** (engl. *daemon*), johon hyökkäykseen osallistuvat koneet ottivat yhteyden. Taustaprosessin avulla hajautettu palvelunestohyökkäys voitiin aloittaa yhtäaikaisesti. Kun palvelunestohyökkäys aktivoitiin, taustaprosessiin liittyneet koneet lähettivät UDP-yhteysprotokollalla verkkoliikennettä palvelinkoneelle. Verkkoliikenteen pakettikuorma generoitiin summittaisesti.

Ensin hajautettuun palvelunestohyökkäykseen liitettiin viisi konetta. Tämän jälkeen hyökkäys toistettiin kymmenellä koneella. Samanaikaisesti palvelinkoneen toimintaa tarkkailtiin **Windowsin Task Manager**-sovelluksella ja kokeiltiin sovelluksen toimintaa yleisesti.

6.7.3 Tulokset

Alla olevasta kuvasta näkyy liikenteen määrä minuutin kuluttua siitä, kun hyökkäys aloitettiin viidellä koneella. Kuvasta ilmenee, kuinka palvelinkoneen suorittimen kuorma nousi hyökkäyksen alussa. Kuorma tasaantui kuitenkin pian. Sovelluksen käytössä ei ilmentynyt ongelmia.



Kuva 6.7: Viiden koneen hajautetun palvelunestohyökkäyksen kuorma minuutin kuluttua.

Kun hyökkäys toistettiin kymmenellä koneella, havaittiin samankaltainen nousu suorittimen kuormassa. Tässäkin tapauksessa kuorma tasaantui pian. Kun sovellusta käytettiin samanaikaisesti, osassa sivujen latauksista oli havaittavissa hienoista viivettä. Tämä viive ei kuitenkaan ollut merkittävää siinä mielessä, että se olisi haitannut sovelluksen toimintaa. Tulokset viittaavat siihen, että onnistuneeseen palvelunestohyökkäykseen tarvitaan kohteesta riippuen kymmeniä, tai satoja, orjakoneita. Skenaario olisi ollut lähempänä aitoa tilannetta, jos verkkopalvelimelle olisi simuloitu käyttäjien normaali käyttökuorma.

6.7.4 Suojautuminen

Kuten aiemmin huomioitiin (ks. kpl 5.2.8), palvelunestohyökkäyksiä vastaan ei ole olemassa tyydyttävää suojausta. Testisovelluksessa ei ollut paikallaan mitään suojausta. Tässä tapauksessa orjakonejoukko oli kuitenkin liian pieni tavoitteen saavuttamiseksi.

6.8 Johtopäätöksiä skenaarioista

Tarkastellaan kappaleessa 4 esitettyjä sosiaalisten verkkosovellusten tietoturvariskejä. Lähtökohdiksi, joista sosiaalisten verkkosovellusten tietoturvariskit johtuvat, esitettiin *arkaluontoiset tiedot*, *luottamuksen ongelma*, *sisällöntuotanto*, *pienen maailman verkosto* ja *verkkosovellusten heikko tietoturva*. Empiirisen tutkimuksen rajallisuuden takia

Skenaarioissa arkaluontoisia tietoja edustivat käyttäjien viestit sekä profiilien tiedot. Luottamuksen ongelmaa ei voitu mitata skenaarioissa, koska varsinaista sosiaalista kanssakäymistä käyttäjän ja hyökkääjän roolien välillä ei ollut. Sisällöntuotantoa edusti viestien lisääminen ja erityisesti käyttäjien lisäämät hyperlinkit. Haitalliset linkit toimivat kelpona yhteyspisteenä hyökkäyksille skenaarioissa. Skenaariossa haitallisia linkkejä ei kätketty uudelleenohjauspalveluiden taakse (ks. kpl 5.2.4), joilla kokeneempaa käyttäjää olisi voitu harhauttaa. Pienen maailman verkosto ei sovelluksessa esiintynyt, sillä käyttäjien välillä ei ollut mitään relaatioita. Jotta pienen maailman verkoston tutkiminen olisi mielekästä, vaatisi se suuremman käyttäjäkunnan. Lisäksi käyttäjäkunnan olisi mielellään oltava autenttinen. Verkkosovellusten heikko tietoturva on viisikon teknisin ulottuvuus, joka sai suurimman huomion skenaarioissa. Skenaarioissa esiin tullut virhetulosteiden näkyvyys ja oletusasetusten vaihtamatta jättäminen olivat hyviä esimerkkejä tästä ulottuvuudesta. Käydään läpi tähän ulottuvuuteen liittyviä huomioita.

Skenaarioihin valmistautuminen osoitti, kuinka paljon valmiita, matalan oppimiskynnyksen työkaluja on saatavilla tietoturvakatselmointiin sekä hyökkäysten suorittamiseen. Työkalut eivät voi korvata ammattitaitoa ja työpanosta, mutta ovat arvokas etu tietojärjestelmän tietoturvakatselmoinnissa. Oikea työkalu oikeaan tehtävään: työkaluja tuntuisi olevan runsaasti ja laadulliset erot niiden välillä suuria.

CodeIgniter -ohjelmistokehyksen tietoturva oli allekirjoittaneen mielestä odotettua parempi. CodeIgniter tarjosi miltei kaikkiin skenaarioihin jonkin suojauksen. Aiempi tutkimustyö hyökkäyksistä oli avuksi oikean suojauksen valinnassa ja implementoinnissa. Ainoa moittimisen aihe oli uuden CSRF-suojauksen puutteellinen dokumentointi, joka tosin tullaan luultavasti nopeasti korjaamaan.

Injektiohyökkäyksiä torjuttaessa on koodin *tietoturvallinen laatu* olennaisesti tärkeä tekijä. Alla oleva ohjelmistokehys voi tarjota monia hyviä apufunktioita tai metodeja syötteen kelvolliseksi todentamiseen. Ohjelmistokehyksen yleiskäyttöisen luonteen takia apufunktiot eivät aina sellaisenaan sovi sovellukseen. Kehittäjän on oltava tietoinen mahdollisista vaaroista ja osata käyttää näitä oikein. Ymmärtämällä kontekstin ja käyttämällä positiivista logiikkaa (ks. kpl 6.2.4) voidaan syöte rajata

sovelluksen vaatimalla tavalla. Syötteen rajoittaminen kontekstin mukaisesti auttoi testisovelluksessa korjaamaan CSRF-haavoittuvuuden (ks. kpl 6.6.3).

Tietojärjestelmän tietoturvaan kuuluvat myös palvelimet, laitteet ja näiden ohjelmistot. Näiden asetusten oikea konfiguraatio voi olla yhtä tärkeää kuin sovelluksen koodin *tietoturvallinen laatu*. Ensimmäisessä skenaariossa ei ollut paikallaan kehittyneempiä ratkaisuja verkkotiedustelun estämiseksi, kuten **tunkeutumisenhavaitsemisjärjestelmää** (engl. *Intrusion Detection System*). Verkkotiedustelun tunkeilevuuden taso useimmilla työkaluilla ei ollut suuri. Allekirjoittanut uskoo, että tietojärjestelmän ohjelmistokokonaisuuden (kuten WAMP, LAMP) selville saaminen useimmissa tapauksissa ei ole vaikeaa. Tärkeää on pitää sovelluksen toimintaa ylläpitävät ja tukevat ohjelmistot päivitettyinä, jotta tietojärjestelmän haavoittuvuuksien lukumäärä olisi mahdollisimman pieni.

7 Yhteenveto

Tietoturva kokonaisuutena on prosessi, joka sisältää teknisen ulottuvuutensa lisäksi ihmiset ja ympäristön. Tietoturvallisuuden laajaan piiriin kuuluu teknisen puolen lisäksi yrityksen tietoturvamotivaatio, tietoisuus riskeistä, riskinhallinta, sekä henkilöstön riittävät resurssit. Kartoittamalla mahdollisia tietoturvariskejä voidaan arvioida pahimmat uhkakuvat ja priorisoida kehityskohteita. Riskinhallinta auttaa tekemään valintoja niin suunnittelussa kuin julkaisun jälkeisissä päätöksissä. Suojausmekanismit voivat rajoittaa sovelluksen käyttämistä tai käytettävyyttä siinä määrin, että sovelluksen toiminnalliset vaatimukset eivät täyty.

Verkkosovelluksen menestymiseen vaaditaan jonkinasteinen käyttäjien luottamus tuotteeseen. Käyttäjät arvostavat kehittäjien aktiivista luonnetta ja esillä oloa tietoturvaa koskevissa asioissa. Luottamusta lisäävät ohjeistus, prosessien läpinäkyvyys, aktiivinen rooli viestinnässä, paremman käyttökokemuksen tavoittelu ja itse suojausmekanismien kehitys. Menestymisen yksi ehto on hyvä käytettävyys. Sosiaalisessa mediassa hyvään käytettävyyteen kuuluvat ainakin helppokäyttöisyys, tehokkuus ja käytön mielekkyys. Kun teknologinen käyttökonteksti on tarpeeksi vaihteleva, jotkut yhdistelmät voivat olla keskenään epäsopivia. Tämä voi tuoda järjestelmään vaikeasti havaittavia riskejä tai heikentää käyttäjien luottamusta sovellukseen. Usein sosiaalisilla verkkosovelluksilla on pitkä elinkaari, jolloin ylläpito nousee tärkeään rooliin. Ylläpidon huolellinen, tietoturvallisuuden huomioon ottava, suunnittelu ja toteutus ennaltaehkäisevät tietoturvariskejä.

Käyttäjät ovat tärkeässä roolissa sosiaalisten verkkosovellusten tietoturvassa. Käyttäjien interaktiivisen luonteen takia, käyttäjäkunta säätelee yhteisön tietoturvavaatimuksia. Hyökkäysten näkökulmasta verkkoyhteisöissä käyttäjillä on moninainen rooli. He voivat samanaikaisesti toimia hyökkäysten kannustimina, mahdollistajina ja edelleen viejinä. Mitä enemmän käyttäjät tallentavat datansa ulkoiseen lähteeseen (kehittäjän pilveen), sitä suurempi vastuu kehittäjillä on huolehtia datan oikeanlaisesta käsittelystä. Ilman selvää lainsäädäntöä ja sosiaalisten verkkosovellusten valvontaa on mahdotonta sanoa, käsitelläänkö sitä oikein. Tilastollisesti riskiä joutua hyökkäyksen kohteeksi lisäävät suuri näkyvyys, verkostoituneisuus ja kuuluisuus. Odotettavaa on, että palveluiden suuri suosio kiinnittää ammattimaisten hyökkääjien mielenkiinnon. Tietoa ja työkaluja hyökkäyksien suorittamiseen on saatavilla varsin helposti Webistä. Esitellyistä hyökkäyksistä palvelunesto-

hyökkäykselle ei ole olemassa tyydyttäviä suojauksia.

Kaikille sosiaalisen kerroksen funktioille hyvä käytettävyys on pakollista. Yksityisyysasetukset eivät saa kirjaimellisesti olla yksityisiä, jolloin yksityisyysasetukset eivät ota huomioon käyttäjien välisiä relaatioita. Mitä enemmän käyttäjille annetaan työkaluja organisoida sisältöä, lisätään sisällön oikeellisuuden tarkkuutta mutta lisätään käyttäjien työkuormaa ja avataan ovi toiminnon väärinkäyttämiselle. Itsenäisinä entiteetteinä verkkoyhteisöt voivat lisätä tai pienentää riskejä. Esimerkki tietoturvariskejä pienentävästä suojausmekanismista on metatietojen käyttäminen. Käyttäjien syöttämillä metatiedoilla luokitellaan käyttäjien luomaa sisältöä, vähentäen näin väärinkäytösten riskiä.

Sosiaalisilla verkkosovelluksilla on piirteitä, joka tekee tietoturvallisuuden toteuttamisen haasteelliseksi. Luottamuksen ongelmaa hyödyntävä hyökkäysvektori, sosiotekninen manipulointi, on merkittävä hyökkäys. Tulokset antavat ymmärtää, että sosiaalinen media on erityinen ympäristö, joka on tavallista alttiimpi sosiotekniselle manipuloinnille. Tietoturvallisuuden teknisestä tietoturva-osaamisesta on vähiten hyötyä sosioteknisiä manipulointitapoja vastaan. Tärkeää on ymmärtää, kuinka minimoida luottamuksen ongelman haasteet. Eräs yleistynyt ja helposti toteutettava sosioteknistä manipulointia hyödyntävä hyökkäys on identiteettivarkaus.

Tietoturvallisuus on nopeasti muuttuva osa-alue. Kehittäjien kannattaa olla valveutuneita ja valmiita muutoksille. Suositeltavaa on omaksua sovellusta tukevat suunnittelumetodologiat, hyväksi havaitut tietoturvakäytännöt tai suunnitella alusta asti tietoturvapolitiikka. Minkä tahansa näiden omaksuminen auttaa integroimaan tietoturvan sovelluskehitykseen. Tärkeää on pystyä luomaan mittaus- ja laadunvarmistusjärjestelmä, joka määrittelee kullekin vaatimukselle ehdot, jolloin tämä on täytetty. Apuna voidaan käyttää esimerkiksi mittauksia, kuten roskapostin suhteellinen osuus viesteistä, ja käyttäjien palautetta. Tarkoitus on hallita prosessia.

Informaation määrän lisääntyessä ja monimuotoistuessa, tulevaisuuden haasteeksi muodostuu tietojen käsittelyyn liittyvät tietoturvakysymykset. Tietojen käsittely voidaan suorittaa tarpeeksi tehokkaan, vastuualueet yksioikoisesti määrittelevän, mallin mukaisesti. Selkeä määrittely ja jako auttaa osallisia ymmärtämään oman roolinsa tietoturvaa koskien paremmin ja toimimaan tehokkaammin. Näin vähennetään monitulkintaisuutta ja ehkäistään rajatapauksia.

Mitä enemmän on käytettävissä tutkimustuloksia sosiaalisen median kontekstista ja sen merkityksistä ihmisen käyttäytymiselle, voidaan tietoturvariskien kartoittamista ja arvioimista tehostaa. Kehittäjien kannattaa muistaa, että käyttäjät voivat toimillaan parantaa verkkoyhteisön tietoturvallisuutta. Jos käyttäjiä ei huomioida sosiaalisen median tietoturvakysymyksissä, moni verkkosovellusta kohtaava tieto-

turvariski suurenee. Sosiaaliset verkkosovellukset eivät ole vielä kokeneet (kevät 2011) katastrofaalisia hyökkäyksiä, ehkä MySpacen Sammy -matoa lukuunottamatta. Opinnäyte korostaa, kuinka on odotettavaa, että tällaiset hyökkäykset tulevat lisääntymään. Sosiaaliset verkkosovellukset joutuvat jatkossa kiinnittämään erityistä huomiota tietoturvallisuuteen. Käyttäjien huomioiminen tietoturvan tärkeänä toisena osapuolena tulee tiedostaa. Selkeä lainsäädäntö tai sopimus rajatapauksia koskien, käyttäjien potentiaalin hyödyntäminen ja informaation läpinäkyvyys auttavat kehittäjiä luomaan tietoturvallisemman verkkoyhteisön, joka nauttii käyttäjiensä luottamusta.

Lähteet

[1] Uche Ogbuji, *Real Web 2.0: Bookmarks? Tagging? Delicious!*, http://www.ibm.com/developerworks/library/wa-realweb1/, viitattu 8.7.2010, julkaistu 26.10.2006.

[2] Kari A Hintikka, *Web 2.0 - johdatus internetin uusiin liiketoimintamahdollisuuksiin*, TIEKE Tietoyhteiskunnan kehittämiskeskus ry.

[3] Dare Obasanjo, *Social Software is the Platform of the Future*, *http://www.25hoursaday.com/weblog/PermaLink.aspx?guid=06ff2206-27a3-4d55-81d8-bbee37073d6d*, viitattu 8.9.2010, julkaistu 5.10.2004.

[4] *Sosiaalisen median sanasto*, Sanastokeskus TSK ry, 2010. ISBN 978-952-9794-26-3. Saatavilla http://www.tsk.fi/tsk/sosiaalisen_median_sanasto_tsk_40-513.html.

[5] Iltalehti, *Facebook-kaveri voi olla FBI:n agentti*, http://www.iltalehti.fi/digi/2010031711310229_du.shtml, viitattu 8.7.2010, julkaistu 17.3.2010.

[6] *Facebook solvaus toi sakkoja*, http://www.mikropc.net/kaikki_uutiset/article384699.ece, viitattu 8.7.2010.

[7] *Social Networks: Facebook Takes Over Top Spot, Twitter Climbs*, http://blog.compete.com/2009/02/09/facebook-%09myspace-twitter-social-network/, viitattu 8.7.2010.

[8] *Facebook Reaches Top Ranking in US*, http://weblogs.hitwise.com/heather-dougherty/2010/03/facebook_reaches_top_ranking_i.html, viitattu 8.7.2010.

[9] Danah Boyd, Scott Golder, Gilad Lotan; *Tweet, Tweet, Retweet: Conversational Aspects of Retweeting on Twitter*, Conversational Aspects of Retweeting on Twitter. HICSS-43. IEEE: Kauai, HI, January 6.

[10] *Industry Category: Social Networking -> Personal Networking*, http://siteanalytics.compete.com/twitter.com/, viitattu 8.7.2010.

[11] *Measuring Tweets*, http://blog.twitter.com/2010/02/measuring-tweets.html, viitattu 8.7.2010.

[12] Nicole B. Ellison, Cliff Lampe, Charles Steinfield; *Social Network Sites and Society: Current Trends and Future Possibilities*, interactions archive Volume 16 , Issue 1 (January + February 2009), SECTION: The potential for technology-enabled connections, Pages: 6-9.

[13] Esma Aïmeur, Sébastien Gambs, Ai Ho; *Towards a Privacy-Enhanced Social Networking Site*, pp.172-179, 2010 International Conference on Availability, Reliability and Security, 2010.

[14] Won Kim, Ok-Ran Jeong, Sang-Won Lee; *On social Web sites*, Information Systems 35 (2010) 215-236.

[15] *Haitin katastrofi nosti Twitter-palvelun valokeilaan*, http://www.hs.fi/ulkomaat/artikkeli/Haitin+katastrofi+nosti+Twitter-palvelun+valokeilaan/1135252145135, viitattu 8.7.2010.

[16] *Wikipedia: Social network service*, http://en.wikipedia.org/w/index.php?title=Social_network_service&oldid=418594233, viitattu 8.7.2010.

[17] *Twitter Blog: Location, Location, Location*, http://blog.twitter.com/2009/08/location-location-location.html, viitattu 8.7.2010.

[18] Ben Parr, *Facebook Launches Its Location Features* , http://mashable.com/2010/08/18/facebook-launches-its-location-features-live/, viitattu 21.9.2010, julkaistu 18.8.2010.

[19] http://projects.webappsec.org/Web-Hacking-Incident-Database-2009-Annual-Report *Web Hacking Incident Database 2009 Annual Report*, http://projects.webappsec.org/Web-Hacking-Incident-Database-2009-Annual-Report, viitattu 8.7.2010.

[20] Weimin Luo, Jingbo Liu, Jing Liu, Chengyu Fan; *An Analysis of Security in Social Networks*, pp.648-651, 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, 2009.

[21] Jonell Baltazar, Joey Costoya, Ryan Flores; *The Real Face of KOOBFACE: The Largest Web 2.0 Botnet Explained*, Trend Micro Threat Research.

[22] *Facebook De-friends Its App Verification Program*, http://www.pcworld.com/businesscenter/article/181093/facebook_defriends_its_app_verification_program.html, viitattu 8.7.2010.

[23] *Reported to Facebook for violating their terms of service?*, http://www.sophos.com/blogs/gc/g/2009/02/27/reported-facebook-violating-terms-service/, viitattu 8.7.2010.

[24] *Facebook "hacked by porn site"*, http://www.pcpro.co.uk/news/148908/facebook-hacked-by-porn-site, viitattu 8.7.2010.

[25] *Hackers hit Twitter and Facebook*, http://news.bbc.co.uk/2/hi/technology/8188201.stm, viitattu 8.7.2010.

[26] *Top 8 Social Media Security Threats*, http://information-security-resources.com/2009/08/17/top-8-social-media-security-threats/, viitattu 8.7.2010.

[27] *Report: Russian hackers used Twitter, Facebook in '08 Georgian war*, http://www.bizjournals.com/sanfrancisco/stories/2009/08/17/daily9.html, viitattu 8.7.2010.

[28] Marc Fossi et. al, *Symantec Global internet Security Threat Report*, Trends for 2009, Volume XV, Published April 2010.

[29] Danesh Irani, Steve Webb, Kang Li, Calton Pu;
Large Online Social Footprints–An Emerging Threat, Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 03, Pages: 271-276, 2009.

[30] *Wikipedia: Graph theory*, http://en.wikipedia.org/wiki/Graph_theory, viitattu 8.7.2010.

[31] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, Bobby Bhattacharjee; *Measurement and Analysis of Online Social Networks*, In Proceedings of the 5th ACM/USENIX internet Measurement Conference (IMC'07), 2007.

[32] OWASP Foundation, *http://www.owasp.org/index.php/Avoid_security_by_obscurity*, viitattu 4.8.2010.

[33] *Wikipedia: On the internet, nobody knows you're a dog*, http://en.wikipedia.org/wiki/On_the_internet,_nobody_knows_you're_a_dog, viitattu 8.7.2010.

[34] *Privacy no longer a social norm, says Facebook founder*, http://www.guardian.co.uk/technology/2010/jan/11/facebook-privacy, viitattu 8.7.2010.

[35] Guardian News and Media, *Danah Boyd: "People looked at me like I was an alien"*, http://www.guardian.co.uk/technology/2009/dec/09/interview-microsoft-researcher-danah-boyd, viitattu 8.7.2010.

[36] Johann Schrammel, Christina Köffel, Manfred Tscheligi; *Personality Traits, Usage Patterns and Information Disclosure in Online Communities*, Proceedings of the 2009 British Computer Society Conference on Human-Computer Interaction, Cambridge, United Kingdom, Pages: 169-174, 2009.

[37] Jan Nagy, Peter Pecho; *Social networks security*, 2009 Third International Conference on Emerging Security Information, Systems and Technologies.

[38] Stratis Ioannidis, Augustin Chaintreau; *On the strength of weak ties in mobile social networks*, SNS '09: Proceedings of the Second ACM EuroSys Workshop on Social Network Systems, March 2009.

[39] Johann Schrammel, Christina Köffel, Manfred Tscheligi; *How much do you tell?: information disclosure behaviour indifferent types of online communities*, C&T '09: Proceedings of the fourth international conference on Communities and technologies, June 2009.

[40] Leyla Bilge and Thorsten Strufe and Davide Balzarotti, Engin Kirda; *All Your Contacts Are Belong to Us: Automated Identity Theft Attacks on Social Networks*, International World Wide Web Conference, Proceedings of the 18th international conference on World wide web, SESSION: Security and privacy/session: web security, ISBN:978-1-60558-487-4.

[41] L. Sørensen, *User managed trust in social networking - Comparing Facebook, MySpace and Linkedin*, Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, 2009. Wireless VITAE 2009. ISBN: 978-1-4244-4066-5, p 427 - 431.

[42] Anderson Ross, *Security Engineering, A Guide to Building Dependable Distributed*; Systems, John Wiley & Sons Inc, New York, 2001.

[43] Myers,Jr., J. Paul and Riela, Sandra; *Taming the Diversity of Information Assurance & Security*, J. Comput. Small Coll., vol 23, no 4, year 2008.

[44] Ansar-Ul-Haque Yasar, Davey Preuveneers, Yolande Berbers, Ghasan Bhatti; *Best Practices for Software Security: An Overview* Multitopic Conference, 2008. INMIC 2008. IEEE International, Issue Date: 23-24 Dec. 2008.

[45] Vicente Aceituno Canal, *On Information Security Paradigms*, The ISSA Journal, 2005.

[46] I.Tashi and S.Ghernaouti - Hélie, *Efficient Security Measurements and Metrics for Risk Assessment*, Proceedings of the 2008 The Third International Conference on internet Monitoring and Protection, p131-138, 2008.

[47] *National Information Assurance (IA) Glossary*, Committee on National Security Systems, CNSS Instruction No. 4009, 26 April 2010.

[48] Tatu Lamminmäki, *Utilizing security patterns in application development*, Master's thesis, Department of Information Technology, Jyväskylä University, April 2008.

[49] James Joshi, Saurabh Bagchi, Bruce S. Davie, Adrian Farrel, Bingrui Foo, Vijay K. Garg, Matthew W. Glause, Gaspar Modelo-Howard, Prashant Krishnamurthy, Pete Loshin, James D. McCabe, Lionel M. Ni, Larry L. Peterson, Rajiv Ramaswami, Kumar N. Sivarajan, Eugene H. Spafford, George Varghese, Yu-Sung Wu and Pei Zheng; *Network Security: Know It All*, Elsevier Inc., May, 2008. ISBN: 978-0-12-374463-0.

[50] Leucio Antonio Cutillo, Thorsten Strufe and Refik Molva; *Safebook: A privacy-preserving online social network leveraging on real-life trust*, Communications Magazine, IEEE Volume: 47 , Issue: 12, Publication Year: 2009 , Page(s): 94-101.

[51] Albrecht Enders, Harald Hungenberg, Hans-Peter Denker and Sebastian Mauch; *The long tail of social networking. Revenue models of social networking sites*, European Management Journal, Volume 26, Issue 3, June 2008, Pages 199-211.

[52] LinkedIn -sosiaalinen media, *http://www.linkedin.com/*, viitattu 4.8.2010.

[53] Matthias Quasthoff, Harald Sack and Christoph Meinel; *Who Reads and Writes the Social Web? A Security Architecture for Web 2.0 Applications*, IEEE Computer Society, internet and Web Applications and Services, International Conference on.

[54] Alessandro Marchetto, *Special section on testing and security of Web systems*, International Journal on Software Tools for Technology Transfer (STTT), Volume 10, Number 6 / December, 2008.

[55] *Q4 2009 Security Threat Summary*, http://www.f-secure.com/en_EMEA/security/security-threats/threat-summaries/2009-4.html, viitattu 8.7.2010.

[56] McAfee Labs, *McAfee Threats Report: First Quarter 2010*, McAfee Labs Technical White Papers.

[57] *Sophos Security Threat Report: 2010*, http://www.sophos.com/sophos/docs/eng/papers/sophos-security-threat-report-jan-2010-wpna.pdf, viitattu 8.7.2010.

[58] Bruce Schneier, *Attack Trees*, SANS Network Security 99, 8 October 1999.

[59] Stuart Mcclure, Joel Scambray, George Kurtz; *Hacking Exposed 6: Network security secrets & solutions*, The McGraw-Hill Companies, May, 2009. ISBN-13: 9780071613743.

[60] Oladimeji, E.A.; Supakkul, S.; Chung, L.;, *Representing Security Goals, Policies, and Objects*, Computer and Information Science, 2006 and 2006 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. ICIS-COMSAR 2006. 5th IEEE/ACIS International Conference on , vol., no., pp.160-167, 10-12 July 2006.

[61] Juha Kalander, *Security management: feedback system requirements and realization*, Diploma thesis, Espoo, Faculty of Electronics, Communications and Automation, December 5, 2007.

[62] Wade H. Baker, Linda Wallace, *Is Information Security Under Control?: Investigating Quality in Information Security Management*, IEEE Security and Privacy, pp. 36-44, January/February, 2007

[63] *OWASP Top Ten Project*, http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project, viitattu 14.8.2010.

[64] *Web Application Security Consortium, WASC*, http://www.webappsec.org, viitattu 25.8.2010.

[65] Giles Hogben, *ENISA Position Paper : Security Issues and Recommendations for Online Social Networks*, saatavilla http://www.enisa.europa.eu/act/res/other-areas/social-networks/security-issues-and-recommendations-for-online-social-networks, Nov 14 2007.

[66] Dorfman, Mark S., *Introduction to Risk Management and Insurance (9 ed.)*, Englewood Cliffs, N.J: Prentice Hall. ISBN 0-13-224227-3.

[67] *Common criteria for information technology security evaluation. Part 1: Introduction and general model - version 3.1*, http://www.commoncriteriaportal.org/, viitattu 26.8.2010. July 2009, CCMB-2009-07-001.

[68] Vinay M. Igure and Ronald D. Williams, *Taxonomies of Attacks and Vulnerabilities in Computer Systems*, IEEE Communications Surveys, 1st Quarter 2008, Volume 10, No 1, 20085.

[69] FaceBook's Markup Language, *http://wiki.developers.facebook.com/index.php/FBMLspec* viitattu 26.8.2010.

[70] Mark Curphey; Rudolph Araujo, *Web Application Security Assessment Tools*, IEEE Educational Activities Department, IEEE Security and Privacy, Volume 4 , Issue 4 (July 2006).

[71] Microsoft Corporation, *Improving Web Application Security: Threats and Countermeasures*, ISBN-13 : 978-0-735618-42-8, Microsoft Press (24 Sep 2003).

[72] Dieter Gollmann, *Securing Web applications*, Elsevier Advanced Technology Publications, issn: 1363-4127, Vol. 13, No. 1, 2008.

[73] *OWASP Guide to Cryphography*, http://www.owasp.org/index.php/Guide_to_Cryptography, viitattu 8.9.2010.

[74] *OWASP Authentication Cheat Sheet*, http://www.owasp.org/index.php/Authentication_Cheat_Sheet, viitattu 8.9.2010.

[75] Ai Ho, Maiga, A., Aimeur, E.; *Privacy protection issues in social networking sites*, Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on, ISBN: 978-1-4244-3807-5, p 271-278.

[76] Xi Chen, Shuo Shi; *A Literature Review of Privacy Research on Social Network Sites*, mines, vol. 1, pp.93-97, 2009 International Conference on Multimedia Information Networking and Security, 2009.

[77] Trifonov, G., *Reducing the number of security vulnerabilities in web applications by improving software quality*, Applied Computational Intelligence and Informatics, 2009. SACI '09. 5th International Symposium on, ISBN: 978-1-4244-4477-9, p 51-54.

[78] Fonseca, J., Vieira, M.; *Mapping software faults with web security vulnerabilities*, Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008. IEEE International Conference on, ISBN: 978-1-4244-2397-2, p 257-266.

[79] David Sancho, *Security Guide to Social Networks*, A Trend Micro White Paper, August 2009.

[80] E. Athanasopoulos, A. Makridakis, S. Antonatos, D. Antoniades, S. Ioannidis, K.G. Anagnostakis, E.P. Markatos; *Antisocial Networks: Turning a Social Network into a Botnet*, ISC '08 Proceedings of the 11th international conference on Information Security, ISBN: 978-3-540-85884-3.

[81] S. Hansman, R. Hunt; *A Taxonomy of Network and Computer Attacks*, Comp. & Sec., vol. 24, no. 1, Feb. 2005, pp. 31-43.

[82] Rich Cannings, Himanshu Dwivedi, Zane Lackey; *Hacking Exposed Web 2.0: Web 2.0 Security Secrets and Solutions*, McGraw-Hill Osborne Media; 1 edition (December 17, 2007). ISBN-10: 0071494618.

[83] Abraham S., Chengalur-Smith I.; *An overview of social engineering malware: Trends, tactics, and implications*, Technology in Society (2010), doi:10.1016/j.techsoc.2010.07.001.

[84] *Phishing Activity Trends Report - 1st Quarter 2010*, Anti-Phishing Working Group - Released Sept 23, 2010. Www.apwg.org

[85] *Consumer Advice: How to Avoid Phishing Scams* http://www.antiphishing.org/consumer_recs.html, viitattu 8.10.2010.

[86] Garrett Brown, Travis Howe, Micheal Ihbe, Atul Prakash, Kevin Borders; *Social Networks and Context-Aware Spam*, In CSCW, 2008.

[87] Lee Kyumin, James Caverlee, Steve Webb; *Uncovering social spammers: social honeypots + machine learning*, SIGIR '10: Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval, ISBN: 978-1-4503-0153-4, p. 435-442.

[88] Web Application Security Consortium, *WASC Threat Classification 2.0*, saatavilla http://projects.webappsec.org/f/WASC-TC-v2_0.pdf. Viitattu 13.10.2010.

[89] *OWASP SQL Injection Cheat Sheet*, http://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet, viitattu 14.10.2010.

[90] Jason Lam, Johannes B. Ullrich; *AppSec - Cross Site Request Forgery: What Attackers Don't Want You to Know*, May 22, 2009, SANS Institute InfoSec Reading Room - Application/Database Sec.

[91] *OAuth*, http://oauth.net/, viitattu 29.10.2010.