hyväksymispäivä arvosana

arvostelija

Keskitetty tunnistautumispalvelu web-sovellusarkkitehtuureissa

Olli Jokinen

Helsinki 9.4.2012

Pro gradu -tutkielma (suunnitelma)

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Olli Jokinen

Työn nimi — Arbetets titel — Title

Keskitetty tunnistautumispalvelu web-sovellusarkkitehtuureissa

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro gradu -tutkielma (suunnitelma)	9.4.2012	27 sivua

Tiivistelmä — Referat — Abstract

Tutkielman tiivistelmä.

ACM Computing Classification System (CCS): A.1 [Introductory and Survey]

I.7.m [Document and Text Processing]: Miscellaneous

Avainsanat — Nyckelord — Keywords
avainsana

Säilytyspaikka — Förvaringsställe — Where deposited
Tietojenkäsittelytieteen laitoksen kirjasto

Muita tietoja — övriga uppgifter — Additional information

Sisältö

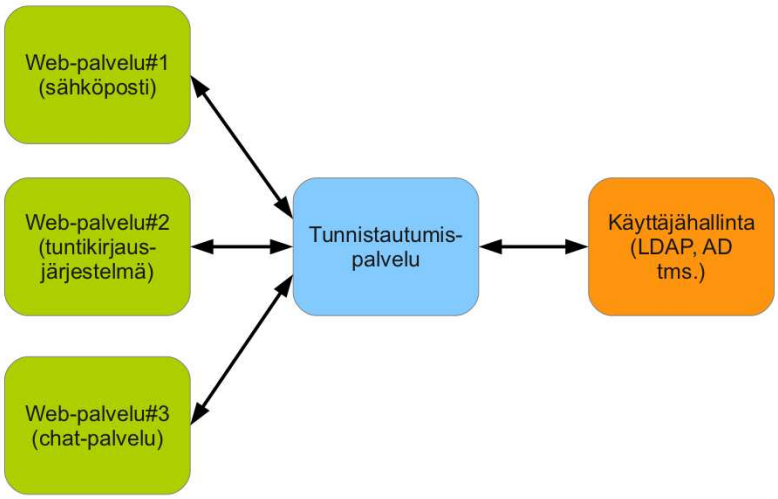
1 Johdanto

Web-sovellusten yleistyessä törmätään yhä useammin käyttäjän tunnistautumisen ongelmaan. Sovelluksen tarjoaja haluaa, että vain tietyillä käyttäjillä on pääsy järjestelmään. Tämä vaatii käyttäjän tunnistamisen, ts. käyttäjä on se, joka väittää olevansa. Kun web-sovelluksia on yhä enemmän, käyttäjän tunnistamiseen käytettäviä valtuutustietoja (credentials) syntyy useita, jolloin käyttäjän täytyy muistaa useita käyttäjätunnus/salasana-pareja.

Samaan ongelmaan törmätään usein myös intranet-palveluissa, kun niiden arkkitehtuuria viedään palveluperustaiseen suuntaan. Tällöin jokaisen palvelun täytyy pystyä tunnistamaan käyttäjä, jotta hänelle voidaan näyttää käyttöoikeuksien mukaan sisältöä. Intranetin palveluilla voi olla oma käyttäjätietokanta, joka synkronoidaan päätietokannan kanssa, jolloin yksittäisellä palvelulla on ajantasainen tieto käyttäjän oikeudesta käyttää palvelua. Tässä tapauksessa ongelmallista on, että käyttäjän tunnukset ja salasanat siirtyvät useaan järjestelmään, joka heikentää järjestelmän tietoturvaa [BDN+11]. Jos taas eri intranetin palveluihin luodaan oma tunnus, syntyy uusi muistettava tunnus/salasana-pari.

Palveluille voidaan avata pääsy organisaation käyttäjähallintaan, jolloin palveluun pääsy edellyttää, että käyttäjän syöttämä tunnus/salasana-pari löytyy organisaation käyttäjähallinnasta. Tämä voi olla järjestelmän ylläpitäjän näkökulmasta ongelmallista, sillä käyttäjäkanta sisältää tietoja, joiden ei haluta päätyvän ulkopuolisten käsiin. Yhtenä ratkaisuna on esitetty keskitettyä tunnistautumispalvelua, jolla on pääsy käyttäjähallintaan ja jota vasten yksittäiset web-sovellukset tunnistavat käyttäjät [BDN+11]. Kuvassa 1 on esitetty arkkitehtuurikuva järjestelmästä, joka käyttää tunnistatumispalvelua käyttäjän tunnistamiseen.

Tässä tutkielmassa paneudutaan mainittuun ongelmakenttään ja tutkitaan ratkaiseeko keskitetty tunnistautumispalvelu esitetyt ongelmat ja millaisia mahdollisia



Kuva 1: Arkkitehtuurikuva järjestelmässä, joka käyttää keskitettyä tunnistautumispalvelua.

uusia ongelma se tuo tullessaan. Pohditaan myös millaisia etuja erillisestä tunnistautumispalvelusta on verrattuna suoraan integraatioon käyttäjähallintaan. Etuja ja haittoja punnitaan järjestelmän ylläpitäjän, käyttäjän ja web-sovellusohjelmoijan näkökulmasta.

Tutkielmassa ei keskitytä kaikille avoimiin tunnistautumispalveluihin (kuten Facebook tai Google), vaan organisaatioihin, joilla on oma käyttäjähallinta. Esimerkki tällaisesta organisaatiosta on Helsingin Yliopisto, jolla on oma Active Directory -käyttäjähallinta [Lat10]. Myös monet yritykset ylläpitävät omaa käyttäjähallintaa esimerkiksi LDAP-järjestelmässä tai keskitetyssä tietokannassa. Esimerkkinä hyötyjä ja haittoja punnittaessa käytetään Helsingin Yliopistoa, mutta myös muunlaiset organisaatiot huomioidaan, mikäli se on perusteltua.

Tutkielma jakautuu kahteen osaan. Luvuissa 2, 3 ja 4 käsitellään ongelmakenttään liittyvää teoriaa. Luvussa 2 esitellään web-sovellukset, erityisesti palveluperustaiset web-sovellukset. Luvussa 3 käydään läpi käyttäjän tunnistautumisen tekniikoita ja ongelmakenttää. Luvussa 4 käsitellään keskitettyä tunnistautumista palveluperus-

taisissa arkkitehtuureissa. Luvut ...

2 Web-sovellukset

Web-sovelluksella tarkoitetaan ohjelmaa, jota suoritetaan palvelimella ja jota käytetään WWW-selaimen avulla [Con99]. Käyttäjän selaimen kautta tekemät pyynnöt vaikuttavat palvelimen tilaan. Esimerkiksi lomakkeella voidaan lähettää palvelimelle tallennettavaa tietoa tai muokata palvelimelle aiemmin tallennettua tietoa. Web-sovelluksilla on oma sisäinen logiikka, joka erottaa ne tavallisista web-sivuista [Con99]. Web-sivu voi myös toimia dynaamisesti, mutta ilman mahdollisuutta vaikuttaa sivuston sisäiseen logiikkaan, ei voida puhuta web-sovelluksesta.

Web-sovelluksissa on kolme peruskomponenttia: selain, verkko ja palvelin. Käyttäjä pyytää selaimella verkon yli palvelimelta selaimen ymmärtämällä kielellä ohjelmoitua tiedostoa, jonka selain visualisoi käyttäjän ymmärtämään muotoon [Con99]. Tiedostojen koodauksessa käytetään yleisemmin HTML-kuvauskielellä, muita käytettyjä kieliä ovat JavaScript ja Flash, joilla laajennetaan HTML-kielen toiminnallisuutta. Pyyntö tehdään HTTP- tai HTTPS-protokollilla, sen mukaan käytetäänkö salaamatonta (HTTP) vai salattua (HTTPS) yhteyttä [Res00].

Tässä luvussa käydään läpi web-sovellusten historiaa ja niiden perusarkkitehtuurien kehitystä kohti palvelusuuntautuneita järjestelmiä. Luvun loppupuolella avataan web-palveluiden tunnistautumiseen liittyvää ongelmakenttää.

2.1 Historia

WWW:n, ja erityisesti WWW:n edeltäjän Gopherin, sivustot olivat alkujaan lähinnä staattisia dokumentteja, jotka oli linkitetty toisiinsa. Dokumentit muodostivat erillisia arkistoja esimerkiksi tutkijoiden käyttöön. Ajan kuluessa tuli tarpeellisek-

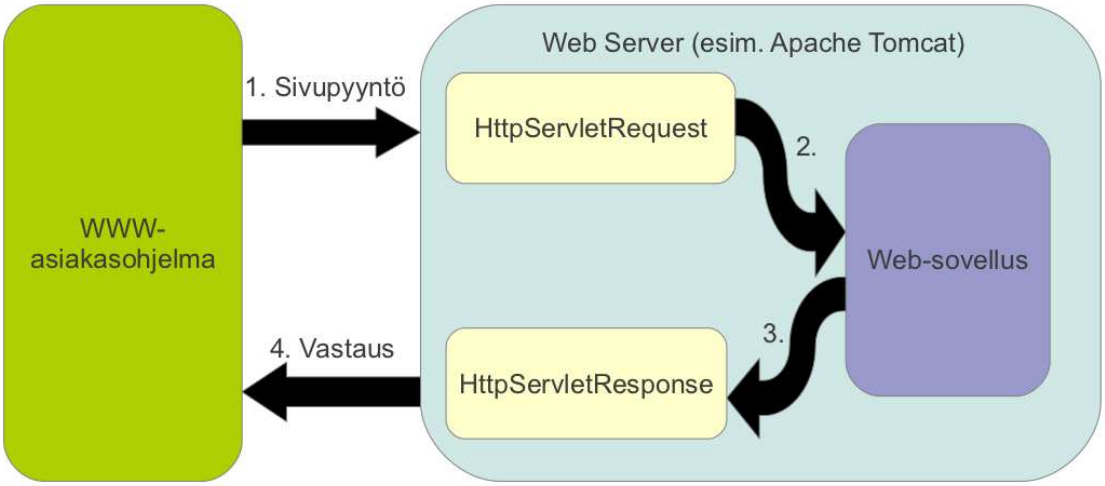
si tehdä sivustoja, jotka reagoivat käyttäjän syötteeseen ja toimivat dynaamisesti käyttäjän syötteen mukaan [CDI+04].

Vuonna 1993 esitelty Common Gateway Interface (CGI) on standardi, jolla voidaan ajaa ohjelmia web-sivujen kautta UNIX-ympäristössä [RC04]. Tyypillisesti CGI:llä ajettavat ohjelmat ovat itsenäisiä ja ne on kirjoitettu jollain skriptikielellä esimerkiksi Perlillä tai PHP:lla. Skripti saa parametrina käyttäjän lähettämät syötteet ja muodostaa sen perusteella käyttäjälle näkyvän HTML-sivun. Ajan kuluessa CGI-ohjelmat alkoivat kasvaa ja niiden arkkitehtuuri monimutkaistua, kun esimerkiksi niissä alettiin käyttää tietokantoja.

CGI-ohjelmien kasvun lisäksi myös niiden suoritukseen vaadittava ajoympäristö alkoi kasvaa ja muodostaa ongelmia CGI:n käytölle. CGI käynnistää suoritettavan prosessin jokaisen sivupyynnön yhteydessä, mikä voi olla hidasta, jos prosessi esimerkiksi lataa muistiin paljon dataa. CGI-ohjelmat korvasi ajan kuluessa erilliset web-palvelimet, joiden sisällä ohjelmakoodi suoritetaan. Erityisesti Sunin 1990-luvulla kehittämä Java-kieli oli merkittävässä roolissa tässä kehityksessä [Con99].

Javan web-käyttöön suunniteltu Enterprise Edition (J2EE) käyttää Servlet-tekniikkaa, joka laajentaa perinteisen web-palvelimen toimintaa mahdollistamalla Java-kielisten sovellusten suorittamisen palvelimen sisällä.. Kuvassa 2 on kuvattu sivupyynnön kulkua Java Servlet-palvelimessa. Käyttäjän pyynnön saatuaan web-palvelin (esimerkiksi Apache Tomcat) ohjaa pyynnön Java Servet-luokalle, joka käsiteltyään sen, palauttaa vastauksen web-palvelimelle, joka näyttää sivun käyttäjälle. Web-palvelin pitää siis Java-prosessia koko ajan käynnissä, ja ympäristöä ei tarvitse käynnistää jokaisen käyttäjän pyynnön yhteydessä uudestaan. Servlet-tekniikan avulla voidaan tehostaa web-palvelimen resurssien (esim tietokantayhteydet) jakamista useamman pyynnön kesken, tehdä transaktiomalleja, muuttaa käyttäjäohjelman tilaa ja hallita web-sovelluksia etänä [Con99]. Muille web-ohjelmointikielille on toteutettu myös omia, Javan Servlettiä muistuttavia, web-palvelimia. Esimerkiksi

Ruby on Rails web-ohjelmointikehys tarjoaa oletuksena Rubyyn sisäänrakennetun WEBrick web-palvelimen, joka käynnistää ympäristön ja ohjaa pyynnöt oikeille Ruby on Rails -luokille [RTH+11].



Kuva 2: Kontrollin kulku Java Servlet-palvelimessa [JEG+10].

Servletien jälkeen paljon suosiota on saanut CGI:stä kehittynyt FastCGI-protokolla, joka korjaa CGI:ssä havaittuja puutteita [Adi97]. Web-palvelin ei käynnistä jokaista pyyntöä kohti uutta palvelinprosessia, vaan pyynnöt lähetetään ja vastaanotetaan pistokkeen (socket) avulla palvelimella pyöriville prosessille, jotka voivat palvelevat montaa pyyntöä. FastCGI-ohjelmat voivat olla myös hajautettu eri palvelimille, jolloin pyyntöjen välitykseen käytetään TCP-yhteyttä. FastCGI:tä voidaan käyttää minkä tahansa kielen kanssa, joka tukee pistokkeiden käyttöä. Näin ollen se on varteenotettava tekniikka, koska ohjelmointikieli ei ole rajattu vain Javaan, vaan käytössä on Javan lisäksi koko kielien kirjo (esimerkiksi PHP, Python ja Ruby) [Adi97].

Tiivistetysti web-sovellusten historiasta voidaan sanoa, että niistä on tullut yksittäisistä palvelinprosesseista itsenäisiä ohjelmia, joita suoritetaan jatkuvasti palvelimella. Ohjelmat saavat kontrollin web-palvelimelta joko laajentamalla web-palvelimen

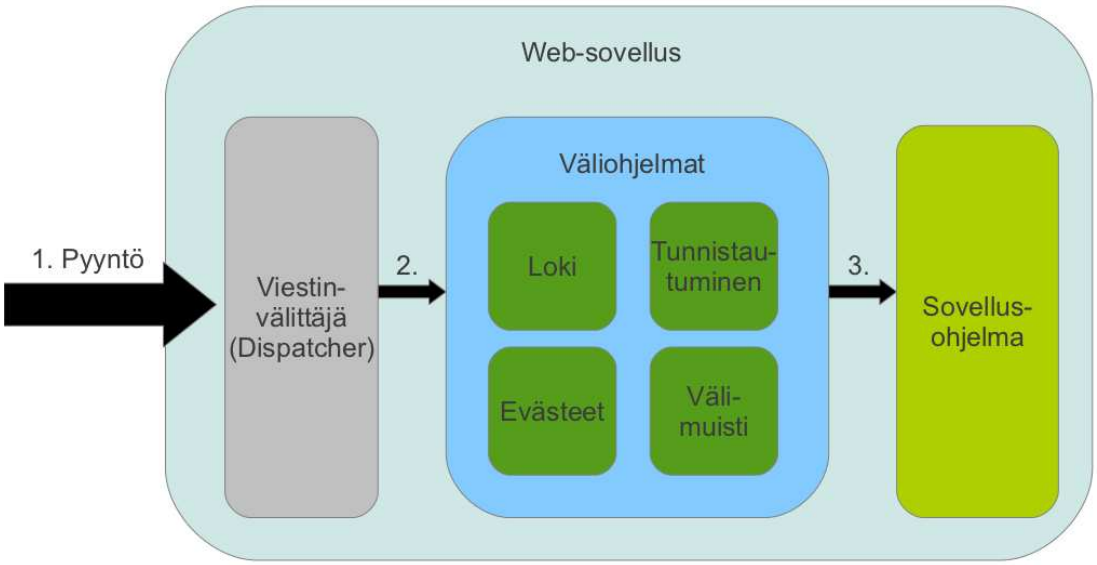
toimintaa (Servlet, WEBrick yms) tai pistokkeiden avulla (FastCGI). Web-sovellukset tuottavat käyttäjän syötteen ja web-sovelluksen sen hetkisen tilan mukaan käyttäjälle dynaamisen HTML-muotoisen sivun.

2.2 Web-sovellusten arkkitehtuuri

Servlet- tai FastCGI-tekniikka ei sido web-sovelluksen arkkitehtuuria, vaan molempia tekniikoita käytettäessä sovellusten perusarkkitehtuuri on sama. Sovellukset ovat jatkuvasti pyöriviä prosesseja, jotka saavat syötteenään käyttäjän syöttämän datan lisäksi joukon erilaisia suoritusympäristöön liittyviä resursseja ja muuttujia (mm. tietokantayhteys) [Con99]. Syötteen perusteella sovellus tuottaa tulosteen, joka palautetaan käyttäjälle.

Tyypillinen tapa toteuttaa web-sovelluksia nykypäivänä on käyttää sovelluskehystä, mikä tuo oman lisänsä sovellusten arkkitehtuuriin. Käytettyjä kehyksiä on esimerkiksi Javalla toteutettu Spring, Pythonin Django ja Ruby-ohjelmointikieleen luottava Ruby on Rails [JHA+05, FBC08, RTH+11]. Näissä sovelluskehyksissä erillinen viestinvälittäjä (dispatcher) ottaa vastaan pyynnön ja välittää sen eteenpäin sovellusohjelmalle. Pyyntö ei siirry suoraan sovellusohjelmalle, vaan se välitetään väliohjelma-kerroksen läpi kuvassa 3 esitetyllä tavalla. Nämä väliohjelmat esimerkiksi tekevät merkintöjä lokitiedostoihin, asettavat ja lukevat käyttäjän selaimen evästeitä tai asettavat ympäristömuuttujia sovellusohjelmaa varten.

Sovellusohjelmasta on pyritty karsimaan paljon yleisiä tehtäviä sovelluskehyksen tai käyttäjän toteuttaman väliohjelman tehtäväksi. Web-sovellus käyttää hyväkseen sovelluskehyksen tarjoamia parametreja, jotka ovat esimerkiksi viittauksia muuttujiin, joita säilytetään käyttäjän istunnon ajan muistissa. Myös käyttäjän tunnistautuminen on yksi sovelluskehyksen tehtävistä [RTH+11]. Tällöin sovellusohjelma saa ympäristömuuttujana kirjautuneen käyttäjän tiedot (esimerkiksi tunnistenumeron ja



Kuva 3: Web-sovelluskehyksellä toteutetun web-sovelluksen kontrollin kulku [RTH+11].

käyttöoikeudet), joita se käyttää hyväksi ohjelmalogiikassaan.

CGI-ohjelmien tietovuopohjaisesta arkkitehtuurityylistä on sovelluskehysten myötä menty modulaarisiin arkkitehtuureihin. Saman sovelluskehyksen sisällä voi olla useampia ohjelmamoduuleita, joita viestinvälittäjä kutsuu esimerkiksi URL-osoitteen perusteella. Kun ohjelmamoduuleiden ei tarvitse toteuttaa kaikkea perustoiminallisuutta itse, yksinkertaistuu niiden rakenne. Web-sivustot saattavat käyttää hyväksi myös useampaa yksittäistä web-sovellusta, jolloin puhutaan palveluperustaisista arkkitehtuureista.

2.3 Palveluperustaiset arkkitehtuurit

Palveluperustainen arkkitehtuuri (service oriented architecture, SOA) on arkkitehtuurityyli, jossa autonomiset sovellukset tarjoavat rajapinnan muille sovelluksille palveluidensa käyttämiseksi [Jos07]. Sovellusten tehtävä on rajattu ja muista so-

velluksista riippumaton. Esimerkiksi matkatoimiston web-sivulla voidaan käyttää lento- ja hotelliyhtiöiden tarjoamien web-sovellusten rajapintoja matkan varaamiseen ja pankin valuuttakurssi-rajapintaa hintojen muuttamiseen valuutasta toiseen. Sovellukset käyttävät toistensa palveluita Internetin yli web services -rajapintojen avulla [Jos07].

Palveluperustaisia sovelluksia käytettäessä ohjelmoija ei ole kiinnostunut yksittäisen sovelluksen teknisestä toteutuksesta, vaan sovellusten tarjoamista rajapinnoista eli palveluista [Jos07]. Rajapintojen muoto on tavallisesti rakenteellinen JSON- tai XML-muotoinen dokumentti, joka on helposti koneen luettavissa. Eri tyyppiset web-sovellukset tai -palvelut käyttävät näitä rajapintoja eri tavalla. Koko maailman sään tarjoava palvelu saattaa kysyä säätietoja paikallisilta sääpalveluilta ja koostaa niistä JSON-dokumentin, jonka selaimessa pyörivä JavaScript-ohjelma rikastaa käyttäjän ymmärtämäksi sääkartaksi. Yhden sivulatauksen yhteydessä käyttäjä saattaa huomaamattaan käyttää useita eri web-palveluita.

Yksittäiset palvelut, kuten säätilapalvelu, voivat olla kaikille avoimia, mutta jotkut palvelut taas haluavat tunnistaa käyttäjän, jotta hänelle voidaan näyttää häntä koskeva data ja muiden data pysyy piilossa. Esimerkiksi palvelu, joka pitää kirjaa opiskelijoiden suorittamista kursseista, näyttää vain tunnistetun käyttäjän suoritukset. Käyttäjän tunnistautumiseen voidaan käyttää, muiden palveluiden tapaan, erillistä identiteetintarjoajaa, josta kerrotaan enemmän luvussa 4. Tämä edellyttää sekä käyttäjällä olevaa tunnusta identiteetintarjoajan palveluun (esimerkiksi Facebook, Google tai organisaation oma palvelu) että palvelun ylläpitäjä luottaa identitetiintarjoajaan.

3 Tunnistautuminen ja pääsynhallinta

Yksittäiset web-palvelut vaativat joissakin tapauksissa käyttäjän tunnistautumisen, jotta ne voivat palvella kyseistä käyttäjää. Koska web-palveluiden on tarkoitus olla riippumattomia ympäröivästä maailmasta, on yleiskäyttöisten ratkaisujen löytäminen tunnistautumisongelmaan tärkeää.

Web-sovelluksia ajetaan tyypillisesti avoimessa Internet-verkossa, ja ne sisältää dataa, johon halutaan asettaa pääsyrajoituksia [Lyn11]. Rajoitukset voivat koskea koko järjestelmää ja sen dataa, jolloin vain tietyt henkilöt pääsevät järjestelmään. Tällaisia rajoituksia voidaan toteuttaa esimerkiksi rajaamalla pääsy dataan vain organisaation sisäverkosta. Usein kuitenkaan käytön rajaaminen organisaation sisälle ei pelkästään riitä, vaan tarvitaan tarkempia pääsyrajoituksia. Esimerkiksi sähköpostijärjestelmässä sähköpostit saavat näkyä vain lähettäjälle ja vastaanottajalle. Pääsyrajoitusten vuoksi käyttäjä täytyy tunnistaa, jotta hän näkee vain hänelle tarkoitetun datan.

Luvun ensimmäisessä aliluvussa käydään läpi tunnistautumisen osatekijät, mitä tarkoitetaan puhuttaessa tunnistautumisesta tai laajemmin pääsynvalvonnasta. Luvussa 3.2 käydään läpi toimintaympäristöä, jossa tunnistautumista tarvitaan. Luku 3.3 käsittelee web-palveluiden tunnistautumisen nykytilannetta, ja luvussa 3.4 esitellään keskitetty tunnistautumispalvelu, jolla nykytilanteessa syntyviä ongelmia voidaan ratkaista.

3.1 Turvallisuuden osatekijät

Palveluiden turvallisuus koostuu kolmesta osatekijästä: tunnistautumisesta (authentication), pääsynvalvonnasta (access control) ja auditoinnista (audit) [SS96]. Tunnistautumisessa käyttäjän identiteetti varmistetaan, esimerkiksi käyttäjätunnuksen

ja salasanan avulla. Tämän jälkeen pääsynvalvonta tarkistaa, onko kyseisellä käyttäjällä oikeutta tehdä pyytämäänsä toiminto. Auditoinnissa analysoidaan järjestelmän tuottamaa dataa, esimerkiksi lokitiedostoja, aktiivisesti, ja käyttäjän pääsy järjestelmään voidaan estää, jos luvatonta käyttöä esiintyy [SS96]. Tämän tutkielman painopiste on käyttäjän tunnistautumisessa ja osittain myös pääsynvalvonnassa, mutta auditointi ei kuulu tutkielman aihepiiriin.

Tunnistautumista ja pääsynvalvontaa ei voida täysin erottaa toisistaan. Käyttäjään liittyy attribuutteja, joiden perusteella pääsynvalvonta voidaan tehdä järjestelmän sisällä. Pääsynvalvonta voidaan tehdä esimerkiksi roolipohjaisena pääsynvalvontana, jolloin käyttäjän rooli organisaatiossa (esimerkiksi esimies/alainen) vaikuttaa siihen, mitä palveluita tai palveluiden osia hänellä on oikeus käyttää.

3.2 Ympäristön kuvaus

Yritysten tai yhteisöjen web-sovellukset ovat avoimia kaikkia Internetin käyttäjiä pienemmälle osajoukolle. Esimerkiksi yrityksen intranet-järjestelmään on pääsy vain yrityksen työntekijöillä, jotka on kirjattu tietokantaan. Erilaisin palomuuri-asetuksin pääsy intranet-järjestelmään voidaan rajata vain yrityksen sisäverkkoon, mutta sekään ei poista tunnistautumisen tarvetta. Aivan kuin Internet-sovelluksissa yleensä, myös intranet-järjestelmässä halutaan tietää, kuka yrityksen työntekijöistä sitä kulloinkin käyttää, jotta työntekijälle osataan näyttää vain häntä koskevaa dataa.

Joissakin tapauksissa intranet-järjestelmä pitää sisällään myös käyttäjähallinnan, jolloin yrityksessä ei ole erillistä tietokantaa käyttäjille, vaan jokaiselle työntekijälle luodaan erillinen tunnus intranetiin. Toisissa tapauksissa taas halutaan hyödyntää erillistä käyttäjähallintaa, jolloin käyttäjän tiedot ovat esimerkiksi erillisellä AD-palvelimella (Active Directory), jota vasten intranet-järjestelmä tunnistaa käyttäjät.

Palvelusuuntautuneissa arkkitehtuureissa samaa käyttäjätietokantaa käyttäviä so-

velluksia tai palveluita voi olla useita. Tällöin ns. pääkäyttäjäkannasta voidaan luoda oma paikallinen kopio jokaista palvelua varten. Tästä seuraa monenlaisia synkronointiongelmia [ZLZ11]. Esimerkiksi työntekijän irtisanoutuessa joudutaan tunnus poistamaan kaikista tietokannoista erikseen. Myös osoitteen yms. tietojen muutokset täytyy päivittää kaikkiin tietokantoihin. Lisäksi käyttäjälle syntyy saman järjestelmän sisällä monia tunnuksia, joihin saattaa liittyä erilliset salasanat. Käyttäjän kannalta on myös ikävää kirjautua jokaiseen osapalveluun erikseen.

Koko käyttäjäkantaa ei kuitenkaan tarvitse kopioida jokaiselle sovellukselle, vaan yksittäiset sovellukset voivat tunnistautua erillistä tunnistautumispalvelua vasten [SM12]. Tällöin käyttääkseen yrityksen intranet-palvelua, täytyy käydä tunnistautumassa tunnistautumispalvelussa. Nykypäivänä monet web-sovellukset toimivat juuri ulkoisen tunnistaumispalvelun kautta. Viihdesivustoille voi luoda tunnuksen kirjautumalla Facebookin tai LinkedInin kaltaisten sivustojen kautta [SM12]. Myös esimerkiksi Kelan sivuston käyttöä varten tunnistaudutaan pankkitunnuksilla Tupas-järjestelmän avulla [Fin11].

3.3 Tunnistautuminen web-palveluissa

Web-palveluihin tunnistautuminen tehdään sovelluksen väliohjelmakerroksessa (kuva 3). Tällöin varsinaisen sovelluslogiikan ei tarvitse tietää tunnistautumisen tekniikasta mitään, vaan sovellusohjelmalle välitetään vain kirjautuneen käyttäjän tiedot. Lähestymistapa mahdollistaa tunnistautumismekanismin vaihtamisen ilman muutosta sovellusohjelman logiikkaan. Väli- ja sovellusohjelmalla on tällöin sovittu rajapinta, jonka mukaan käyttäjän tiedot päätyvät sovellusohjelmalle.

Tunnistautumisen yhteydessä käyttäjä ohjataan sivulle, jossa olevaan lomakkeeseen hän syöttää käyttäjätunnuksen ja salasanan, joita verrataan järjestelmään tallennettuihin tietoihin. Sivuohjauksen yhteydessä web-palvelu lisää HTTP-pyynnön pa-

rametreihin tiedon siitä, minne käyttäjä ohjataan onnistuneen kirjautumisen jälkeen [RHHL11]. Kirjautumisen jälkeen web-palvelu saa käyttäjän perustiedot (esim id-numero ja nimi) ja käyttäjään liittyvän metadatan, jota käytetään hyväksi pääsynvalvonnassa.

Tunnus/salasana-parille on olemassa myös vaihtoehtoisia tunnistautumismenetelmiä. Tunnistautumismenetelmät voidaan jakaa kolmeen ryhmään [BDN+11]. Tunnistautumiseen voidaan käyttää jotain käyttäjän tuntemaa asiaa, yleisesti salasanaa. Toinen vaihtoehto on käyttää jotain, mitä käyttäjä omistaa. Tällainen on esimerkiksi matkapuhelin, johon lähetettävää tunnuslukau käytetään tunnistautumisessa [vTJJvT09]. Kolmas vaihtoehto on käyttää jotain käyttäjän fyysistä ominaisuutta, käytännössä biometristä dataa, kuten sormenjälkeä tai silmän iiristä, jotka ovat jokaisella ihmisellä yksilölliset. Näillä menetelmillä voidaan parantaa tunnistamisen luotettavuutta [BDN+11].

Tunnistautumispalveluun on määritelty web-palvelut, jotka käyttävät sitä tunnistamiseen. Tunnistautumispalvelu, nimensä mukaisesti, hoitaa pelkästään käyttäjän identiteetin varmistamisen, mutta sillä ei ole mitään tietoa yksittäisen web-palvelun toiminnasta. Esimerkiksi Facebook-tunnistautumista käyttävä valokuvien jakopalvelu ei suinkaan tallenna valokuvia Facebookiin, vaan omaan tietokantaansa [MMCvM10]. Tällöin web-palvelussa täytyy olla käyttäjätietokanta, joka pitää sisällään käyttäjän id-numeron ulkopuolisessa tunnistautumispalvelussa ja käyttäjään liittyviä sisäisiä resursseja, kuten valokuvia. Kun uusi käyttäjä tunnistautuu hyväksytysti tunnistautumispalvelun kautta, lisätään sovelluksen käyttäjätietokantaan uusi rivi. Mitään tunnistautumisinformaatiota (tunnus/salasana) ei tallenneta tietokantaan, vaan jokaiselle käyttäjän tietokantariville lisätään id-numero tunnistautumispalvelussa.

3.4 Keskitetty tunnistautuminen

Monet web-palvelut ovat ulkoistaneet käyttäjien tunnistautumisen Facebookin ja LinkedInin tapaisille toimijoille, keskittyen vain oman sovelluksensa toteuttamiseen [SM12]. Järjestely helpottaa toisaalta sovelluskehittäjän arkea, koska kaikkea ei tarvitse tehdä itse, mutta myös loppukäyttäjän käyttökokemus paranee, sillä samalla verkkoidentiteetillä pääsee moneen eri palveluun.

Organisaatioiden sisäisissä palveluissa Facebookin tai LinkedInin kaltaisten palveluiden käyttö ei välttämättä tule kysymykseen. Intranet-järjestelmien ylläpitäjät eivät mahdollisesti halua siirtää käyttäjähallintaansa ulkopuolisen yrityksen haltuun. Tällöin vaihtoehtona on tarjota Facebookia tai LinkedIniä vastaava keskitetty tunnistautumispalvelu, joka on integroitu organisaation olemassa olevaan käyttäjähallintajärjestelmään.

Keskitetyn tunnistautumisen tarkoituksena on tarjota palvelu, jota vasten käyttäjä voidaan tunnistaa erillisestä web-sovelluksesta ilman uuden identiteetin luontia [SM12]. Tunnistautumispalvelun ja sitä käyttävien web-sovellusten välillä on luottamussuhde, jolloin web-sovellukseen ei tarvitse luoda omaa käyttäjille tunnistautumistietoja (käyttäjätunnus/salasana), vaan se voi luottaa tunnistautumispalvelun tunnistamiin käyttäjiin.

Etuna tunnistautumispalvelusta on järjestelmän tietoturvan ja tunnistautumisen luotettavuuden paraneminen. Keskitetyssä palvelussa käyttäjän tunnistautumisen luotettavuutta voidaan parantaa esimerkiksi vaatimalla tavallista web-sovellusta vahvempia salasanoja. Toisaalta on havaittu, että käyttäjät valitsevat vahvempia salasanoja palveluihin, jotka he kokevat tärkeiksi verrattuna vähemmän tärkeisiin web-palveluihin [FH07].

Käyttäjien tunnistautumista voidaan vahvistaa myös lisävarmistuksilla, kuten puhelimen kautta tehtävällä todennuksella tai biometriikalla [BDN+11]. Yksittäisen

palvelun kohdalla vaikkapa matkapuhelinvarmennuksen käyttöönotto on hankalaa, mutta kun tunnusta käytetään useampaan palveluun, sen käyttö on helpommin perusteltavissa. Tunnistautumiseen tehtävät parannukset eivät vaadi muutoksia yksittäiseen web-sovellukseen, koska web-sovelluksen ja tunnistautumispalvelun välinen rajapinta ei muutu.

Keskitetyn ratkaisun myötä erillisten web-sovellusten ei tarvitse integroitua suoraan organisaation käyttäjähallintajärjestelmiin, vaan pelkästään tunnistautumispalvelulla on pääsy sinne. Pelko käyttäjiä koskevan datan joutumisesta vääriin käsiin vähenee, koska vain tunnistautumispalvelu integroituu käyttäjähallintajärjetelmään. Tällöin kriittisen käyttäjähallinnan integraatiopisteiden määrä vähenee, kun web-sovellukset eivät ota suoraan yhteyttä käyttäjähallintaan.

Jos käyttäjähallinta on esimerkiksi SQL-tietokanta, joka ei tue tunnistautumista, keskitetyllä tunnistautumispalvelulla ratkaistaan tilanne, jossa käyttäjädata ja käyttäjiin liittyviä tunnistautumistietoja käyttäjätunnuksineen ja salasanoineen on kopioitu jokaiseen erilliseen web-sovellukseen. Tällöin ei synny aiemmin mainittuja synkronointiongelmia esimerkiksi henkilön jättäessä organisaation.

Tunnistautumispalvelu toimii siis eräänlaisena ”palomuurina” web-sovelluksen ja käyttäjähallinnan välillä. Sillä on oma rajapinta web-sovellukseen päin, joten organisaatiossa on mahdollista siirtyä esimerkiksi SQL-tietokannasta LDAP-tietokantaan ilman muutoksia web-sovelluksiin. Myös useiden käyttäjähallintajärjestelmien käyttö on mahdollista. Esimerkiksi tietojenkäsittelytieteen laitoksen intranetissä suoritettava tunnistautumispalvelu voi ensin hakea käyttäjää tietojenkäsittelytieteen laitoksen järjestelmästä ja jos käyttäjää ei löydy, haetaan käyttäjää yliopiston järjestelmästä.

Kääntöpuolena keskitetyssä ratkaisussa on käytetyn tunnuksen ja salasanan kalastelun käyminen houkuttelevaksi, koska niiden avulla hyökkääjä pääsee käyttäjän

nimissä useaan palveluun tai jopa luomaan käyttäjän identiteetillä tunnuksia uusin palveluihin. Tästä syystä keskitettyyn tunnistautumispalveluun kohdistuu normaalia web-palvelua suuremmat odotukset tietoturvalle, joten siinä käytettyjen teknologioiden täytyy olla luotettavia.

Web-sovellusten ja tunnistautumispalvelun väliseen rajapintaan on ehitetty useita tunnistautumisprotokollia, joilla tunnistautuminen voidaan tehdä turvalliseksi ja käyttäjälle helpoksi [BYXM09]. Tunnistautumisprotokollat ovat käytössä avoimen Internetin puolella, ja esimerkiksi Facebookilla ja Googlella on ollut merkittävä rooli näiden protokollien syntyhistoriassa [BYXM09]. Näitä protokollia käsitellään tarkemmin seuraavassa luvussa.

4 Keskitetyn tunnistautumisen teknologiat

Verkkotunnistautumiseen kehitettyjä protokollia hyödyntämällä käyttäjän tunnistetietoja (esim. käyttäjänimi ja salasana) ei tarvitse siirtää verkon yli ja web-palvelut voivat ulkoistaa tunnistautumisen erillisen palvelun tehtäväksi [BDN+11]. Tällöin käyttäjä voi samalla identiteetillä tunnistautua useaan eri palveluun, ilman että hänen tarvitsee muistaa palvelukohtaisia tunnus- ja salasanayhdistelmiä [BYXM09].

Tässä luvussa keskitytään keskitetyssä tunnistautumispalvelussa käytettyihin teknologioihin. Teknologiat ovat samoja, jotka ovat tälläkin hetkellä käytössä Internetin web-palveluissa [SM12]. Ensimmäisessä alaluvussa käydään läpi yleisiä periaatteita koskien keskitettyä tunnistautumista.

Toisessa alaluvussa tutustutaan tunnistautumisprotokolliin. Ensin käydään läpi toimintaperiaatteita, jonka jälkeen esitellään tunnistautumisprotokollaksi soveltuvia standardeja. Tämän jälkeen tutkitaan SAML- ja OAuth-protokollien soveltuvuutta tunnistautumispalvelun toteutukseen.

4.1 Keskitetyn tunnistautumisen periaatteet

Keskitetyn tunnistautumisen lähtökohta on käyttäjän tunnistetietojen poistaminen web-palvelun hallinnasta. Käyttäjä ei syötä tunnistetietojaan missään vaiheessa web-palveluun, vaan tunnistautuminen tehdään erillisessä tunnistautumispalvelussa. Nykyisin mm. Facebook ja Google tarjoavat julkiset API-rajapinnat, joiden avulla web-palvelut voivat käyttää niitä tunnistautumiseen [SM12].

Facebook-kirjautuminen on käytössä monissa web-palveluissa ja sen käyttö on suoraviivaista. Kuvassa 4 on kuvattu kirjautuminen Porkkanamafia-ryhmän WWW-sivulle käyttäen Facebookia. Käyttäjä klikkaa WWW-sivulla olevaa ”Login with Facebook” -nappia, jonka jälkeen käyttäjän selaimeen avautuu Facebookin varmennusikkuna, jossa käyttäjää pyydetään varmentamaan kirjautuminen. Selaimen osoitekenttä osoittaa käyttäjälle, että kirjautuminen tapahtuu nimenomaan Facebook-sivulla (joka on varmennettu SSL-sertifikaatilla), joten käyttäjän Facebook-tunnistetiedot eivät päädy Porkkanamafian haltuun, vaan kirjautuminen hoidetaan suoraan Facebookiin [SM12].

Kirjautumisen jälkeen käyttäjälle luodaan rivi käyttäjätietokantaan, jossa on viittaus hänen Facebook-tunnukseensa. Kun käyttäjä tämän jälkeen palaa sivulle ja kirjautuu sisään jälleen Facebook-tunnuksilla, voidaan käyttäjä yhdistää kannasta löytyvään vanhaan käyttäjään. Näin palvelu on ulkoistanut tunnistautumisen ulkopuoliselle taholle, eikä käyttäjän tarvitse muistaa uusia tunnistetietoja, vaan hän voi käyttää Facebook-kirjautumista jatkossa tullessaan Porkkanamafian sivulle.

Samaa periaatetta voidaan käyttää myös organisaatioiden sisäisessä tunnistautumispalvelussa. Organisaation sisäiseen palvelusuuntautuneeseen arkkitehtuuriin toteutetaan erillinen web-palvelu, joka on yhteydessä olemassa olevaan käyttäjätietokantaan (esim. LDAP) ja tarjoaa Facebookia vastaavan tunnistautumisen.

Kuva 4: Käyttäjän kirjautuminen Facebook-tunnuksilla Porkkanamafian web-palveluun.

4.2 Keskitetyn tunnistautumisen rajapintaprotokollat

Tunnistautumiseen liittyvien käsitteiden läpikäynti ennen protokollien yksityiskohtaista esittelyä auttaa tunnistautumiseen liittyvien periaatteiden hahmottamista. Käsitteet ovat yleisluontoisia ja eivätkä kosketa vain tiettyjä protokollaa. Protokollien yhteydessä käytetään käsitteitä asiakasohjelma, tunnistautumispalvelu, suojattu resurssi, valtuutustieto (credentials), valtuutusavain (authorization code) ja pääsyvaltuutus (access token) [BDN+11].

Asiakasohjelmalla tarkoitetaan web-palvelun käyttäjän pääteohjelmaa, jolla hän kirjautuu web-palveluun käyttäen keskitettyä tunnistautumispalvelua. Käytännössä asiakasohjelma on web-palvelun tapauksessa käyttäjän WWW-selain, joka pystyy tekemään uudelleenohjauksia sivustolta toiselle. Uudelleenohjaus on HTTP-protokollan perustoiminnallisuutta, joten mikä tahansa HTTP/1.1-standardin WWW-selain käy asiakasohjelmaksi [FGM+99].

Tunnistautumispalvelu on web-palvelu, johon käyttäjä ohjataan tekemään tunnis-

tautuminen. Onnistuneen tunnistautumisen jälkeen tunnistautumispalvelu ohjaa asiakasohjelman takaisin tunnistautumista pyytäneen palvelun määrittelemään osoitteeseen [BDN+11]. Avoimen Internetin puolella tunnistautumispalvelu voi olla esimerkiksi Facebook tai LinkedIn.

Tunnistautumisprotokollien yhteydessä suojatulla resurssilla tarkoitetaan resurssia, jonka käyttö vaatii tunnistautumisen ja käyttöoikeuden. Yleisessä tapauksessa suojatulla resurssilla tarkoitetaan yksittäistä resurssia (käyttäjän valokuvaa), johon halutaan asettaa pääsyrajoituksia [BDN+11]. Tämän tutkielman puitteissa suojatulla resurssilla tarkoitetaan tunnistautumista vaativaa web-palvelua.

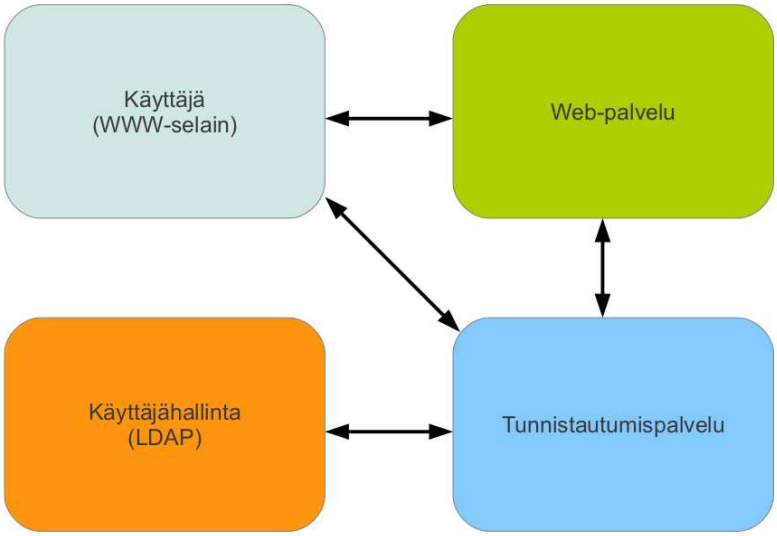
Valtuutustieto koostuu yksilöivästä tunnisteesta ja siihen liittyvästä salaisesta avaimesta. Tämän tutkielman puitteissa valtuutustiedolla tarkoitetaan käyttäjän tunnusta ja salasanaa.

Kirjauduttuaan sisään tunnistautumispalvelimelle, käyttäjä saa valtuutusavaimen, jonka hän lähettää eteenpäin suojatun resurssin omistajalle. Valtuutusavain ei pidä sisällään käyttäjän valtuutustietoja, vaan ainoastaan tunnistautumispalvelin osaa lukea sen [BDN+11]. Saatuaan valtuutusavaimen käyttäjältä voi suojatun resurssin omistaja hakea pääsyvaltuuden käyttäjän tietoihin tunnistautumispalvelusta.

Pääsyvaltuutus on tunnistautumispalvelimelta saatava yksilöivä tunniste, jonka avulla suojatun resurssin omistaja voi pyytää käyttäjän tiedot tunnistautumispalvelulta. Pääsyvaltuutus on voimassa tietyn ajan, jonka jälkeen se täytyy uusia tunnistautumispalvelimella [BDN+11]. Pääsyvaltuutusta voidaan käyttää myös tunnistautumispalvelusta erillään olevien resurssien valtuuttamiseen. Esimerkiksi web-sovellus voi hakea tunnistautumispalvelulta pääsyvaltuuden, jolla hän hakee valokuvia valokuvien jakopalvelusta [SM12].

4.2.1 Rajapintaprotokollien toimintaperiaate

Tunnistautumisessa on kolme osapuolta: asiakasohjelma, web-palvelu ja tunnistautumispalvelu. Osapuolet on esitetty kuvassa 5, jossa on mukana myös tunnistautumispalvelun käyttämä käyttäjänhallinta. Käyttäjähallinta voi olla myös osa tunnistautumispalvelua tai oma komponenttinsa, tunnistautumisen kannalta sillä ei ole väliä.

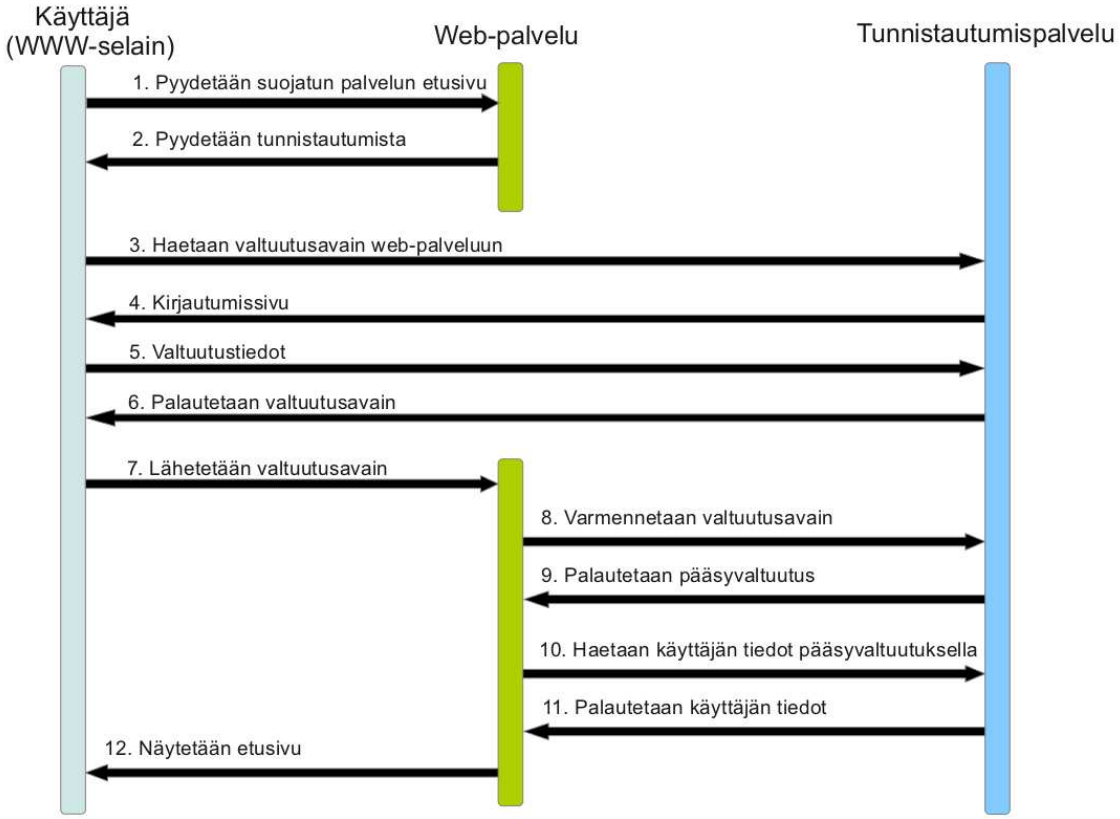


Kuva 5: Keskitetyn tunnistautumisen osapuolet.

Kuvassa 6 on tunnistautumisprotokollien sekvenssikaavio, jossa on kuvattu vaiheet käyttäjän kirjautuessa web-palveluun. Ensimmäiseksi käyttäjä menee asiakasohjelmalla web-palveluun, joka pyytää tunnistautumista erillisessä tunnistautumispalvelussa. Käyttäjän asiakasohjelma ohjataan tunnistautumispalvelun sivulle, joka on yhteydessä organisaation käyttäjähallintaan. Jos käyttäjähallinnasta löytyy käyttäjän syöttämät tunnistetiedot, palautetaan käyttäjälle valtuutusavain, jonka käyttäjä lähettää takaisin web-palvelulle [BDN+11]. Tämän jälkeen web-palvelu varmistaa tunnistautumispalvelulta valtuutusavaimen oikeellisuuden ja tunnistautumispalvelu palauttaa käyttäjän tiedot web-palvelulle [BDN+11].

Useimmat vaiheista tapahtu käyttäjältä näkymättömissä selaimen uudelleenohjauk-

sella. Käyttäjälle näkyvät vaiheet ovat 4 ja 5, joissa käyttäjältä pyydetään käyttäjätunnus ja salasana sekä varmistetaan tietojen lähetys web-palveluun. Käyttäjän syötettä vaativat vaiheet on esitetty aiemmassa kuvassa 4.



Kuva 6: Tunnistautuminen sekvenssikaaviona.

4.2.2 Rajapintaprotokollien standardeja

Tunnistautumisprotokollia tutkitaan ja kehitetään monen eri tahon toimesta. Web Services -teknologioita standardoinut OASIS (Organization for the Advancement of Structured Information Standards) on kehittänyt XML-pohjaista SAML-kieltä tunnistautumisprotokollaksi [CKPM09]. SAML tarjoaa tunnistautumisprotokollan lisäksi joukon muita web-sovellukseen liittyviä turvallisuusstandardeja [SS07].

Myös Microsoft on kehittänyt oman Windows Live ID -standardin, joka tarjoaa myös tunnistautumisprotokollan [BYXM09]. Windows Live ID on osittain suljet-

tu standardi, joka tarjoaa tunnistautumisprotokollien lisäksi täydellisen keskitetyn käyttäjän identiteetin hallinnan [BYXM09]. Suljetun lähdekoodin vuoksi Windows Live ID ei ole tämän tutkielman kannalta kiinnostava protokolla.

Avoimen lähdekoodin maailmassa on syntynyt OpenID, jota mm. Google käyttää tunnistautumisprotokollanaan [BYXM09]. OpenID:stä on haarautunut OAuth-protokolla, jonka ominaisuusjoukko on OpenID:tä suppeampi ja joka on tarkoitettu nimenomaan tunnistautumiseen ja pääsynhallintaan [PSK+11]. OpenID ja OAuth ovat tämän tutkielman kannalta kiinnostavia teknologioita, sillä ne ovat avoimia ja niiden kehitystyö on aktiivista [SM12]. OpenID:n, OAuthin ja SAML:n ominaisuuksia ja soveltuvuutta keskitetyn tunnistautumispalvelun rajapintaprotokollaksi tarkastellaan seuraavissa alaluvuissa.

4.2.3 SAML

Security Assertion Markup Language (SAML) on OASIS-komitean kehittämä XML-pohjainen avoin standardi tunnistautumiseen ja pääsynhallintaan [CKPM09]. Standardin versio 1.0 julkaistiin marraskuussa 2002, versio 2.0 maaliskuussa 2005 ja viimeksi päivitetty versio lokakuussa 2009.

SAML määrittelee XML-pohjaiset työkalut tunnistautumisen ja pääsynhallinnan toteuttamiseen. Varsinainen toteutus, esimerkiksi mitä tietoja siirretään ja millä tavalla, jätetään SAML:ssä toteuttajan päätettäväksi [ZY10]. Varsinaiset SAML-viestit voivat kulkea esimerkiksi synkronisesti SOAP- ja HTTP-protokollalla. SAML soveltuu avoimena ja XML-pohjaisena protokollana käytettäväksi Web Services -standardilla toteutetuissa web-sovelluksissa.

Noin sivu lisää, jotta selviää mikä SAML loppujen lopuksi on.

4.2.4 OpenID ja OAuth

OAuth on avoin tunnistautumisrajapinta hajautetuille web-sovelluksille. Se mahdollistaa käyttäjien resurssien jakamisen palveluiden välillä ilman käyttäjätunnuksen tai salasanan luovuttamista kolmannelle osapuolelle. Se perustuu erilaisten valtuutusavainten (token) välittämiseen palveluiden välillä [RHHL11]. OAuth on yleisesti käytössä web-sovelluksissa, joissa halutaan näyttää käyttäjälle kuuluvia resursseja (esimerkiksi valokuvia), jotka sijaitsevat toisessa sovelluksessa [MMCvM10].

Alunperin OAuthin kehitystyö alkoi marraskuussa 2006, kun Blaine Cook kehitty Twitter-palveluun OpenID-tukea. OAuth on määritelty RFC-dokumentissa numero 5849. Sen ensimmäinen versio (1.0) julkaistiin lokakuussa 2007 ja päivitetty versio (1.0a) kesäkuussa 2009 [RHHL11]. OAuthin versio 2.0 on myös kehitteillä ja se on tarkoitus julkaista marraskuussa 2012 [RHHL11].

Tähän vielä n. sivu-kaksi tekstiä, niin että selviää mitä OpenID ja OAuth on ja mikä niiden suhde toisiinsa on.

5 Arkkitehtuuri

Tämä on vielä auki, myös otsikon osalta. Tarkoitus poimia teorialuvuista oleellinen ja esitellä esimerkiksi arkkitehtuuri web-sovellukselle, joka käyttää keskitettyä tunnistautumispalvelua.

Pituus edellisten päälukujen tasoa, 6-7 sivua.

6 Tulosten evaluaatio

Tutkielman (erityisesti luvun 5) kriittinen arviointi, osattiinko esitellyllä arkkitehtuurilla vastata johdannossa esitettyihin kysymyksiin.

Vielä hieman auki, mutta pituus noin 5 sivua.

7 Rajoitukset ja laajennettavuus

Mitä rajoituksia toteutuksella on? Kuinka voisi laajentaa? Kertakirjautuminen ja autorisointi hyviä suuntia. Kertakirjautuminen on aika peruskauraa OAuthin kanssa, autorisoinnista sen sijaan voi saada ihan mielenkiintoista pohdintaa aikaan.

Pituus 3-5 sivua.

8 Johtopäätökset

Johtopäätökset kaikesta edellämainitusta. Kerrataan tärkeimmät havainnot. Oliko hommassa ylipäätään järkeä? Tutkimustulosten integroiminen isoon kuvaan ja tärkeimmät jatkojalostusmahdollisuudet. Pituus n. 1-2 sivua.

Lähteet

Adi97 Adida, B., It all starts at the server. *Internet Computing, IEEE*, 1,1(1997), sivut 75 –77.

BDN+11 Burr, W. E., Dodson, D. F., Newton, E. M., Perlner, R. A., Polk, W. T., Gupta, S. ja Nabbus, E. A., Electronic authentication guideline. Tekninen raportti SP 800-63-1, National Institute of Standards and Technology Special Publication, December 2011.

BYXM09 Bin, W., Yuan, H. H., Xi, L. X. ja Min, X. J., Open identity management framework for SaaS ecosystem. *ICEBE '09. IEEE International Conference on e-Business Engineering*, oct. 2009, sivut 512 –517.

CDI+04 Challenger, J. R., Dantzig, P., Iyengar, A., Squillante, M. S. ja Zhang, L., Efficiently serving dynamic data at highly accessed web sites. *IEEE/ACM Trans. Netw.*, 12,2(2004), sivut 233–246.

CKPM09 Cantor, S., Kemp, J., Philpott, R. ja Maler, E., Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. Tekninen raportti, Organization for the Advancement of Structured Information Standards OASIS., lokakuu 2009. URL `http://saml.xml.org/saml-specifications`.

Con99 Conallen, J., Modeling web application architectures with uml. *Commun. ACM*, 42,10(1999), sivut 63–70.

FBC08 Forcier, J., Bissex, P. ja Chun, W., *Python Web Development with Django*. Addison-Wesley Professional, ensimmäinen painos, 2008.

FGM+99 Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P.

ja Berners-Lee, T., Hypertext transfer protocol – http/1.1. RFC 2616, IETF, United States, 1999.

FH07 Florencio, D. ja Herley, C., A large-scale study of web password habits. *Proceedings of the 16th international conference on World Wide Web*, WWW '07, New York, NY, USA, 2007, ACM, sivut 657–666.

Fin11 Finanssialan Keskusliitto, Pankkien Tupas–tunnistuspalvelun tunnistusperiaatteet v2.0b. *Verkkojulkaisu*. URL `http://www.fkl.fi/teemasivut/sahkoinen_asiointi/Dokumentit/Tupas-tunnnistusperiaatteet_v20b.pdf`.

JEG+10 Jendrock, E., Evans, I., Gollapudi, D., Haase, K. ja Srivathsa, C., *The Java EE 6 Tutorial: Basic Concepts*. Prentice Hall Press, Upper Saddle River, NJ, USA, neljäs painos, 2010.

JHA+05 Johnson, R., Hoeller, J., Arendsen, A., Risberg, T. ja Kopylenko, D., *Professional Java Development with the Spring Framework*. Wrox Press Ltd., Birmingham, UK, UK, 2005.

Jos07 Josuttis, N., *SOA in Practice: The Art of Distributed System Design*. O'Reilly Media, Inc., 2007.

Lat10 Lattu, M., Keskitetysti hallitun työaseman anatomia. *Tietotekniikkaa yliopistolle, Helsingin yliopiston tietotekniikkapalveluiden tiedotuslehti 1/2010*.

Lyn11 Lynch, L., Inside the identity management game. *Internet Computing, IEEE*, 15,5(2011), sivut 78 –82.

MMCvM10 Machulak, M. P., Maler, E. L., Catalano, D. ja van Moorsel, A., User-managed access to web resources. *Proceedings of the 6th ACM workshop*

on Digital identity management, DIM '10, New York, NY, USA, 2010, ACM, sivut 35–44.

PSK+11 Pai, S., Sharma, Y., Kumar, S., Pai, R. ja Singh, S., Formal verification of oauth 2.0 using alloy framework. *Communication Systems and Network Technologies (CSNT), 2011 International Conference on*, june 2011, sivut 655 –659.

RC04 Robinson, D. ja Coar, K., The Common Gateway Interface (CGI) Version 1.1. RFC 3875, IETF, October 2004.

Res00 Rescorla, E., HTTP Over TLS. RFC 2818, Internet Engineering Task Force, United States, toukokuu 2000.

RHHL11 Recordon, D., Hardt, D. ja Hammer-Lahav, E., The OAuth 2.0 authorization protocol. *Network Working Group*, 5849, sivut 1–47. URL `http://tools.ietf.org/html/draft-ietf-oauth-v2-16`.

RTH+11 Ruby, S., Thomas, D., Hansson, D., Breedt, L. ja Clark, M., *Agile Web Development with Rails*. Pragmatic Bookshelf, 2011.

SM12 Shehab, M. ja Marouf, S., Recommendation models for open authorization. *Dependable and Secure Computing, IEEE Transactions on*, PP,99(2012), sivu 1.

SS96 Sandhu, R. ja Samarati, P., Authentication, access control, and audit. *ACM Comput. Surv.*, 28, sivut 241–243.

SS07 Saklikar, S. ja Saha, S., Next steps for security assertion markup language (saml). *Proceedings of the 2007 ACM workshop on Secure web services*, SWS '07, New York, NY, USA, 2007, ACM, sivut 52–65.

vTJJvT09 van Thanh, D., Jorstad, I., Jonvik, T. ja van Thuan, D., Strong authentication with mobile phone as security token. *Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on*, oct. 2009, sivut 777 –782.

ZLZ11 Zhang, Z., Liu, L. ja Zhang, R., Research in synchronization optimization of the distributed heterogeneous database data. Teoksessa *Advances in Computer Science, Environment, Ecoinformatics, and Education*, Lin, S. ja Huang, X., toimittajat, osa 218 sarjasta *Communications in Computer and Information Science*, Springer Berlin Heidelberg, 2011, sivut 65–69.

ZY10 Zhang, Y.-S. ja Yang, J., Research of dynamic authentication mechanism crossing domains for web services based on SAML. *2nd International Conference on Future Computer and Communication (ICFCC)*, May 2010, sivut 395 – 398.