Juho Salomäki

Viranomaisten tietoturva-auditoinnin vaatimusten mukaisen mobiiliratkaisun suunnittelu

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikan tutkinto-ohjelma

Insinöörityö

14.2.2024

Tiivistelmä

Tekijä:	Juho Salomäki
Otsikko:	Viranomaisten tietoturva-auditoinnin vaatimusten mukaisen mobiiliratkaisun suunnittelu
Sivumäärä:	58 sivua
Aika:	14.2.2024
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tieto- ja viestintätekniikan tutkinto-ohjelma
Ammatillinen pääaine:	Mobiilisovellukset
Ohjaajat:	Lehtori Ilkka Kylmäniemi (vastuuarvioija) Ratkaisuarkkitehti Tommi Harakkamäki

---

Insinöörityön tavoitteena on selvittää, miten viranomaisen turvallisuusauditoinnin hyväksyntään tähtäävä mobiiliratkaisu voidaan toteuttaa ottaen huomioon viranomaisten ja lainsäädännön asettamat tietoturvallisuuden velvoitteet. Työn tarkoitus on toimia tällaisen mobiiliratkaisun suunnittelun ja toteutuksen ohjeistuksena muun olennaisen dokumentaation ohella. Lisäksi työn tarkoitus on tuoda ohjelmistoratkaisun suunnittelijoille mukaan mobiilisovelluksille ainutlaatuisia näkökulmia.

Insinöörityössä käsitellään aluksi olennaisia esitietoja, jotka liittyvät kansallisiin turvallisuusluokituksiin, viranomaisten auditointikriteeristöihin sekä suositeltuihin teknisen toteutuksen tukena toimiviin standardeihin ja ohjeistuksiin. Esitietojen jälkeen syvennytään tarkemmin tekniseen ratkaisuun ottaen huomioon sovelluksen toteutustapa, ratkaisun ympäristön kovennukset sekä auditoitavuus ja itse sovellukseen suoritettavat tietoturvallisuuden suojaukset. Nykyisen tietotekniikan pilvistymisen myötä tutkitaan myös työn lopussa pilviteknologian hyödyntämiseen liittyviä seikkoja sekä vaatimuksia.

Pääsääntöisesti työssä tutkittuja suojauksia hyödynnetään viranomaisten tietoturvallisuuden auditoinnin hyväksyntään pyrkivän sovelluksen velvoitteiden toteutumisen edistämiseksi. Suojaukset perustuvat kuitenkin laajalti yleisiin ohjelmistoalan tietoturvallisuuden standardeihin, joten työtä voidaan hyödyntää myös yleisenä tietoturvallisuuden ohjeistuksena.

Avainsanat:	tietoturva, ohjelmistokehitys, ohjelmistosuunnittelu, auditointi, mobiilisovellukset, mobiiliratkaisu

---

Abstract

Author:	Juho Salomäki
Title:	Designing a Mobile Solution That Meets the Requirements of Finnish Government's Cybersecurity Audit
Number of Pages:	58 pages
Date:	14 February 2024
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communications Technology
Professional Major:	Mobile Solutions
Supervisors:	Ilkka Kylmäniemi, Senior lecturer (Responsible supervisor) Tommi Harakkamäki, Solutions Architect

---

The goal of this bachelor's thesis is to find out how to create a mobile solution that meets the cybersecurity requirements needed for getting approval of the Finnish government's security audit. The thesis is supposed work as a supplementary guideline alongside other documentation having emphasis on bringing the unique side of mobile applications to the table.

At first the thesis will focus on necessary details needed for understanding the audit process, such as Finnish security classes, Finnish government's audit criteria and recommended guidelines. After that the thesis will focus more on the technical side of creating a secure mobile solution taking into consideration system hardening, auditability, and necessary security features to implement on the actual mobile applications. Due to the growing popularity of cloud computing at the end of the thesis there will be a segment focusing on using cloud solutions and the requirements of doing so.

Mainly the thesis is meant for mobile applications aiming to pass the Finnish cybersecurity audit. Although the application protections are mostly based on industry standards, so the thesis can also be used as a general cybersecurity guideline on designing secure mobile applications.

Keywords: information security, software development, software design, audit, mobile applications, mobile solution

Sisällys

Lyhenteet

ASVS: *Application Security Verification Standard*. OWASP:n ylläpitämä ohjelmistojen tietoturvallisuuteen liittyvistä vaatimuksista koostuva standardi.

BSI: *Bundersamt für Sicherheit in der Informationstechnik.* Saksan liittovaltion tietoturvavirasto.

CAA: *Crypto Approval Authority.* Suomen liikenne- ja viestintäviraston salaustuotteiden hyväksyntäviranomainen.

CORS: *Cross-Origin Resource Sharing.* Mekanismi, jolla voidaan sallia resurssipyyntöjen välittäminen eri lähteiden välillä.

CSA: *Cloud Security Alliance.* Kansainvälinen pilvipalveluiden turvallisuusyhteisö.

CSRF: *Cross-Site Request Forgery.* Kyberhyökkäys, jossa tunnistautunut käyttäjä huijataan tekemään ei-toivottuja asioita.

DISA: *Defence Information Systems Agency.* Yhdysvaltain puolustusministeriön alainen tietojärjestelmävirasto.

DoD: *Department of Defence.* Yhdysvaltain puolustusministeriö.

DSA*:* *Designated Security Authority.* Toimii kansainvälisissä tietoturvallisuusvelvoitteissa tarkoitettuna määrättynä Suomen turvallisuusviranomaisena.

HTTP: *Hypertext Transfer Protocol*. Sovelluskerroksen protokolla verkossa tapahtuvaan tiedonsiirtoon.

HTTPS: *Hypertext Transfer Protocol Secure.* Salausta hyödyntävä versio HTTP-protokollasta.

JSON: *JavaScript Object Notation*. Tiedonvälitykseen ja tallennukseen hyödynnettävä tiedostomuoto.

Julkri: *Julkisen hallinnon tietoturvallisuuden arviointikriteeristö.* Suomen viranomaisille suunnattu julkisen hallinnon tietoturvallisuuden auditointityökalu.

Katakri: *Kansallinen turvallisuusauditointikriteeristö*. Suomen viranomaisille suunnattu tietoturvallisuuden auditointityökalu.

NCSA: *National Communication Security Authority.* Suomen kansallinen tietojärjestelmien ja tietoliikenteen turvallisuusviranomainen.

NIST: *National Institute of Standards and Technology.* Yhdysvaltain kauppaministeriön alainen virasto, jonka tarkoitus on kehittää teknologiaa, standardeja ja mittaustekniikoita.

NSA: *National Security Authority.* Suomen kansallinen turvallisuusviranomainen.

OWASP: *Open Worldwide Application Security Project*. Ohjelmistojen tietoturvallisuuteen keskittyvä kansainvälinen järjestö.

PiTuKri: *Pilvipalveluiden turvallisuuden arviointikriteeristö*. Suomen viranomaisille suunnattu pilvipalveluiden turvallisuuden auditointityökalu.

PWA: *Progressive Web Application*. Verkkosovellus, jolla on natiivia sovellusta muistuttavia piirteitä ja ominaisuuksia.

REST: *Representational State Transfer*. Ohjelmointirajapinnan arkkitehtuurityyli.

SSL: *Secure Sockets Layer*. TLS:ää edeltänyt tietoliikenteen salausprotokolla.

SSRF: *Server-Side Request Forgery.* Kyberhyökkäys, jossa palvelinta huijataan tekemään ei-toivottuja asioita.

STRIDE: *Spoofing Tampering Repudiation Information Disclosure Denial of Service Elevation of Privilege*. Riskianalyysiin hyödynnettävä malli.

TL: *Turvallisuusluokka.* Salassa pidettävän tiedon turvallisuusluokitus.

TLS: *Transport Layer Security.* Tietoliikenteen salausprotokolla.

VAHTI: *Julkisen hallinnon digitaalisen turvallisuuden johtoryhmä.* Suomen valtiovarainministeriön alaisena vuosina 1992–2013 toiminut digitaalisen turvallisuuden johtoryhmä.

XML: *eXtensible Markup Language.* Tiedonvälitykseen ja tallennukseen hyödynnettävä merkintäkieli.

XSS: *Cross-Site Scripting*. Web-sovelluksiin kohdistuva skriptin injektiohaavoittuvaisuus.

1 Johdanto

Kun julkisen sektorin asiakkaalle lähdetään työstämään ohjelmistoratkaisua, on käsiteltävästä tiedosta riippuen varmistettava tietoturvallisuuden lainsäädännöllisten sekä viranomaisten asettamien velvoitteiden toteutuminen. Tässä insinöörityössä tutkitaan, miten nämä viranomaisten asettamat velvoitteet voidaan toteuttaa. Työssä on otettu näkökulmaksi velvoitteiden toteuttaminen mobiiliratkaisun kontekstissa. Toisin sanoen työssä tutkitaan laajasti, miten voidaan toteuttaa tietoturvallinen viranomaisten asettamien velvoitteiden täyttävä mobiilisovellus.

Insinöörityön tarkoitus ei kuitenkaan ole toimia minkäänlaisena kiveen hakattuna kriteeristönä, vaan ohjeistuksena mobiiliratkaisua toteutettaessa muun olennaisen dokumentaation tukena tuoden mobiilisovelluskehitykselle ainutlaatuisia näkökulmia. Tietoturvallisuuden velvoitteiden toteutumisen varmistamiseksi tulee siis käyttää viranomaisten omaa sekä viranomaisten suosittelemaa dokumentaatiota, joita myös tullaan käsittelemään työssä tarkemmin.

Työn toimeksiantajana toimii Cinia Oy, joka on kotimainen valtion enimmäisomistuksessa oleva tietoteknisiin ratkaisuihin erikoistunut yritys. Toimeksiantajalla on asiakkaita sekä julkiselta että yksityiseltä sektorilta. Osa näistä julkisen sektorin projekteista käsittelee yhteiskunnallisesti kriittistä tietoa, joten projektit vaativat viranomaisen hyväksynnän tietoturvallisuuden velvoitteiden toteutumisesta. Työn tarkoitus on tukea yrityksen valmiutta toteuttaa mobiiliratkaisuja, jotka täyttävät nämä viranomaisten asettamat tietoturvallisuuden velvoitteet.

2 Kansalliset ja kansainväliset turvallisuusluokitukset

Turvallisuusluokitellulla tiedolla tarkoitetaan sellaista salassa pidettävää tietoa, johon on tehty turvallisuusluokkaa vastaava merkintä. Turvallisuusluokitettu tieto on kansalliselle tai kansainväliselle turvallisuudelle tärkeää tietoa, joten sen suojaaminen on yhteiskunnalle erittäin tärkeää. Ilman turvallisuusluokituksen merkintää ei salassa pidettävä tieto ole sellaisenaan turvallisuusluokitettua tietoa. Kuitenkin tiedonhallintalautakunnan suosituksen [1] mukaan tietoturvallisuuden velvoitteita toteuttaessa olisi suositeltavaa rinnastaa salassa pidettävä tieto turvallisuusluokitus IV tiedon kanssa, jotta vältetään tilanteita, joissa tarvitaan eri tietojärjestelmät salassa pidettävälle ja turvallisuusluokitus IV tason tiedolle.

Tiedon arkaluonteisuus määrää turvallisuusluokituksen tason. Kansallisessa lainsäädännössä julkisen hallinnon tiedonhallinnasta annetun lain (906/2019) 18 §:n ensimmäisen momentin mukaan

> Turvallisuusluokkaa koskeva merkintä on tehtävä, jos asiakirja tai siihen sisältyvä tieto on salassa pidettävä viranomaisten toiminnan julkisuudesta annetun lain 24 §:n 1 momentin 2, 5 tai 7–11 kohdan perusteella ja asiakirjaan sisältyvän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa vahinkoa maanpuolustukselle, poikkeusoloihin varautumiselle, kansainvälisille suhteille, rikosten torjunnalle, yleiselle turvallisuudelle tai valtion- ja kansantalouden toimivuudelle taikka muulla niihin rinnastettavalla tavalla Suomen turvallisuudelle. [2.]

Salassa pidettävälle tiedolle asetettavat turvallisuusluokat on luokiteltu neljään eri tasoon.

Nämä eri turvallisuusluokituksen tasot ovat esitetty valtioneuvoston asetuksessa asiakirjojen turvallisuusluokittelussa valtionhallinnassa (1101/2019) seuraavalla tavalla:

> 1) turvallisuusluokan I asiakirja, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa erityisen suurta vahinkoa tiedonhallintalain 18 §:n 1 momentissa tarkoitetulle suojattavalle edulle;
>
> 2) turvallisuusluokan II asiakirja, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa merkittävää vahinkoa tiedonhallintalain 18 §:n 1 momentissa tarkoitetulle suojattavalle edulle;
>
> 3) turvallisuusluokan III asiakirja, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa vahinkoa tiedonhallintalain 18 §:n 1 momentissa tarkoitetulle suojattavalle edulle;
>
> 4) turvallisuusluokan IV asiakirja, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa lievää vahinkoa tiedonhallintalain 18 §:n 1 momentissa tarkoitetulle suojattavalle edulle. [3.]

Turvallisuusluokituksista voidaan käyttää lyhenteitä, esimerkiksi turvallisuusluokka I:tä vastaa lyhenne TL I. Turvallisuusluokitettuun asiakirjaan merkitään turvallisuusluokkaa vastaava merkintä, esimerkiksi TL I -luokitettuun asiakirjaan merkitään ERITTÄIN SALAINEN. Taulukosta 1 voidaan havainnoida turvallisuusluokkia vastaavat merkinnät. [3.]

Taulukko 1. Suomen turvallisuusluokitukset, niiden merkinnät ja niihin rinnastettavat kansainväliset merkinnät [3, 3 §; 4].

Suojaustaso	**Kansallinen**	**Euroopan unioni**	**NATO**
Turvallisuusluokka I (TL I)	ERITTÄIN SALAINEN	EU TOP SECRET	NATO (COSMIC) TOP SECRET
Turvallisuusluokka II (TL II)	SALAINEN	EU SECRET	NATO SECRET
Turvallisuusluokka III (TL III)	LUOTTAMUKSELLINEN	EU CONFIDENTIAL	NATO CONFIDENTIAL
Turvallisuusluokka IV (TL IV)	KÄYTTÖ RAJOITETTU	EU RESTRICTED	NATO RESTRICTED

Kansainvälisellä turvallisuusluokitellulla tiedolla tarkoitetaan tietoa, johon tiedonluovuttaja on tehnyt kansainvälisen turvallisuusluokitusmerkinnän. Tällaista tietoa on vain sellainen tieto, jota Suomen valtion kuuluu suojata kansainvälisten sopimuksien tai EU:n (Euroopan unioni) turvallisuussääntöjen perusteella. [4, s. 5.] Valtioneuvoston turvallisuusluokittelusta valtion hallinnossa [3, 4 §] -asetuksen mukaan on mainittu, että kansainväliset turvallisuusluokitukset rinnastetaan kansallisiin turvallisuusluokkiin, kun tietoturvallisuuden velvoitteita toteutetaan, mikäli muita kansainvälisiä tietoturvallisuuden velvoitteita ei ole tiedolle asetettu. Taulukosta 1 voidaan havainnoida, miten EU:n ja Naton turvallisuusluokitukset rinnastuvat kansallisiin turvallisuusluokkiin.

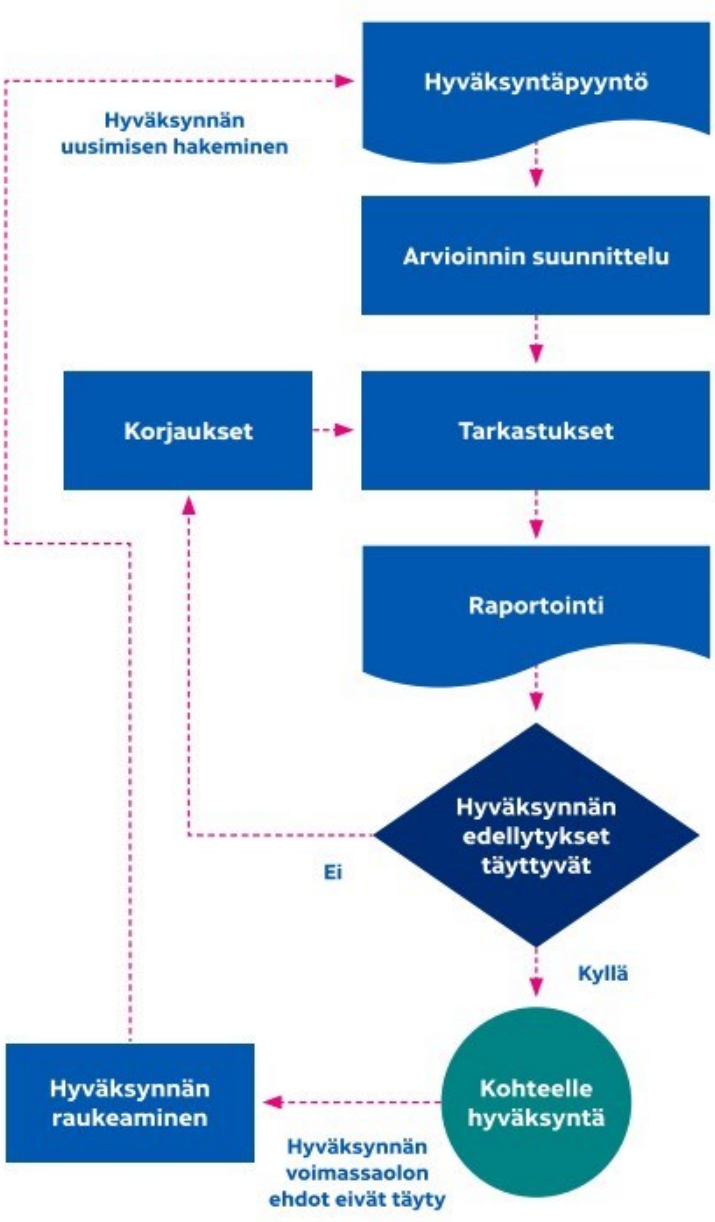
3 Viranomaisten tietoturva-auditointien kriteeristöt

Turvallisuusluokitetun ja salassa pidettävän tiedon käsittelyyn sovelletaan useita kansallisia ja kansainvälisiä lakeja sekä säädöksiä. Lainsäädännön pohjalta sovelluksen tietoturvallisuuden velvoitteiden toteuttaminen ja arvioiminen voi olla käytännön ympäristössä haastavaa. Laillisten vaatimusten selventämiseksi ja tiedon keskittämiseksi kansalliset viranomaiset ovat luoneet auditointikriteeristöjä.

Kriteeristöt ovat auditointityökaluja viranomaisille, ja mikäli niitä sovelletaan niiden pohjalta tietojärjestelmän tietoturvallisuus ja turvallisuusluokitetun tai salassa pidettävän tiedon turvallisen käsittelyn kyvyn arvioi mahdollisesti toimivaltainen viranomainen. Mikäli arvioitava kohde on tietojärjestelmä, voi viranomaisen hyväksymä tietoturvallisuuden arviointilaitos suorittaa arvioinnin [5, s. 6]. Auditointikriteeristöt perustuvat lainsäädännön vähimmäisvaatimuksiin, joten on olennaista pitää jatkuvasti mielessä turvallisuuskriittinen ajattelutapa koko mobiiliratkaisun kehityksen ja ylläpidon elinkaaren ajan, sillä auditoinnin läpäissyt ratkaisu ei aina ole välttämättä turvallinen ratkaisu. Kriteeristöissä on toteuttajalle jätetty paljon liikkumavaraa ja tietoturvallisuuden vaatimuksia voidaan myös korvata tietoteknisesti vastaavilla suojauksilla, mikäli toteuttaja tämän kokee tarpeelliseksi.

Kun arvioinnin kohdeorganisaatio käsittelee turvallisuusluokiteltua tietoa tietojärjestelmissään, vaatii jokainen turvallisuusluokiteltua tietoa käsittelevä tietojärjestelmä erillisen hyväksynnän viranomaiselta. Tietoturvallisuuden velvoitteiden toteutumisen tietojärjestelmissä arvioi toimivaltaisena viranomaisena liikenne- ja viestintäviraston NCSA-toiminto, joko viranomaisen erillisestä pyynnöstä, velvoitteesta tai osana yritysturvallisuusselvitystä. Arviointiprosessiin hyödynnetään viranomaisten auditointityökaluja. Tietojärjestelmää arvioidaan kansallisesta tai kansainvälisestä turvallisuusluokitellun tiedon näkökulmasta riippuen siitä, mihin käyttötapaukseen arviointia haetaan. Mikäli kohdeorganisaatio pyrkii saamaan tietojärjestelmälle hyväksynnän viranomaiselta, tulee tietojärjestelmän tietoturvallisuusjärjestelyiden olla riittäviä toimivaltaisen viranomaisen ja

kohdeorganisaation oman riskienarvioinnin havaintoihin suhteutettuna. Viranomaisen tietojärjestelmän hyväksyntäprosessi on havainnollistettu tarkemmin kuvassa 1. [5, s. 110.]



Kuva 1. Liikenne- ja viestintäviraston hyväksyntään tähtäävä tietojärjestelmän arviointiprosessi [5, s. 112].

Kansallisesta tietojärjestelmien sekä tietoliikenteen tietoturvallisuudesta vastaa liikenne- ja viestintävirasto (NCSA). Kansainvälisten tietoturvallisuuden velvoitteiden toteutumisesta vastaa kansallisena turvallisuusviranomaisena

ulkoministeriö (NSA) ja määrättyinä kansainvälisiä tietoturvallisuusvelvoitteita toteuttavina turvallisuusviranomaisina toimivat pääesikunta, suojelupoliisi ja puolustusministeriö (DSA). [4, s. 4.]

3.1 Kansallinen turvallisuusauditointikriteeristö (Katakri)

Suomen kansallinen turvallisuusauditointikriteeristö, paremmin tunnettu lyhenteellä Katakri on viranomaisille suunnattu auditointityökalu. Auditointityökalun tarkoitus on arvioida kohdeorganisaation turvallisuusluokitetun tiedon suojaamisen kyvykkyyttä. Katakria käytetään myös yritysturvallisuusselvityksen turvallisuusjärjestelyjen arviointiosuuden auditointityökaluna [6].

Auditointityökalulle asetetut vähimmäisvaatimukset perustuvat kansallisen ja kansainvälisen turvallisuusluokitetun tiedon käsittelyssä kansalliseen lainsäädäntöön sekä EU:n turvallisuussääntöihin. Sovellettavia kansalliseen lainsäädäntöön perustuvia lähteitä ovat laki julkisen hallinnon tiedonhallinnasta (906/2019) sekä valtioneuvoston asetus asiakirjojen turvallisuusluokittelusta valtionhallinnossa (1101/2019). EU:n turvallisuusluokitetun tiedon käsittelyyn vähimmäisvaatimusten lähteenä on käytetty EU:n turvallisuussääntöjä (2013/488/EU). [5, s. 5.]

Katakri on toiminut auditointityökaluna turvaamassa kansallista turvallisuutta jo vuodesta 2009 lähtien, jolloin ensimmäinen versio auditointityökalusta valmistui puolustusministeriön koordinoimana sen aikaisen hallituksen sisäisen turvallisuuden ohjelman osana. Ensimmäisen version jälkeen vastuu auditointityökalusta siirtyi sisäministeriölle, ja sisäministeriön johdolla ensimmäinen päivitysversio Katakriin valmistui vuonna 2011. Vuosien varrella päävastuu Katakrista on siirtynyt puolustusministeriön ja sisäministeriön kautta nykyiselle auditointityökalun hallinnoinnista ja päivittämisestä vastaavalle ulkoministeriön alaiselle kansalliselle turvallisuusviranomaiselle (NSA). Viimeisimmästä päävastuun siirrosta päätöksen tekivät keskeiset ministeriöt vuonna 2014. Ensimmäiset auditointityökalun versiot ovat rakenteellisesti huomattavasti erilaiset kuin nykyinen Katakri. Tämä rakenteellinen muutos suoritettiin vuotena 2015 valmistuneen

kolmannen päivitysversion ohessa, missä Katakri keskitettiin turvallisuusluokitetun tiedon tietoturvallisuuden arviointiin. Viimeisin versio Katakrista julkaistiin vuonna 2020 vastaamaan kansallisessa lainsäädännössä tapahtuneisiin muutoksiin. Vuoden 2020 versiossa on otettu huomioon digitaalisen tietojenkäsittelyn kehitysaskelia ja auditointityökalun ohjeistuksia on täydennetty. [5, s. 2; 7.]

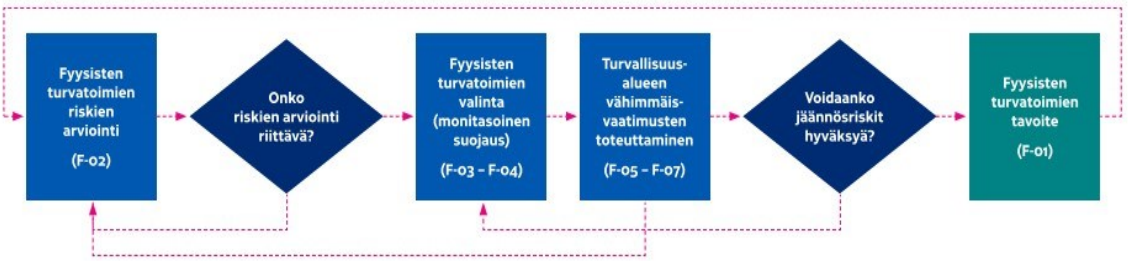
Katakrin rakenne koostuu kolmesta eri osa-alueesta, joihin viitataan auditointityökalussa kirjainlyhenteillä [5, s. 5]. Nämä kolme osa-aluetta ovat:

- turvallisuusjohtamista koskeva osa-alue (T)
- fyysistä turvallisuutta koskeva osa-alue (F)
- teknistä tietoturvallisuutta koskeva osa-alue (I).

Turvallisuusjohtamisen osa-alueessa (T) arvioidaan kohdeorganisaation tietoturvallisuuden hallinnollista kykyä sekä tarkistetaan, että kohdeorganisaatio suorittaa tarvittavat turvallisuustoimenpiteet henkilöille, jotka käsittelevät turvallisuusluokitettua tietoa [5, s. 8]. T-osa-alueeseen kuuluu esimerkiksi kohdeorganisaation poikkeuksien ja tietoturvallisuuden hallinnoinnin kyky, riittävät turvallisuuteen käytettävät resurssit, henkilöstön luotettavuus sekä koulutus ja riittävä organisaatiollinen turvallisuusohjeistus [5, s. 8–21]. Toimivaltaisena viranomaisena arvioi suojelupoliisi tai pääesikunta, ja arviointi on kohdennettu kohdeorganisaatiossa vain siihen osaan, joka käsittelee turvallisuusluokitettua tietoa [5, s. 6; 5, s. 8].

Fyysisen turvallisuuden osa-alueessa (F) arvioidaan kohdeorganisaation asettamia fyysisiä ja teknisiä turvatoimia turvallisuusluokitettua tietoa käsittelevissä fyysisissä ympäristöissä. Tähän osa-alueeseen kuuluvat fyysisten riskien arviointi, fyysiset toimitilat (turva-alueet), fyysisen turvaluokitellun tiedon käsittely ja toimitiloille asetettavat tekniset suojaukset. F-osa-alueen vaatimuksia voidaan käyttää kansallisen tai kansainvälisen turvallisuusluokitetun tiedon fyysisen tiedonkäsittelyn auditoinneissa. Toimivaltainen viranomainen fyysisen turvallisuuden arvioinnissa on suojelupoliisi tai pääesikunta. Mikäli käsitellään kansainvälistä turvallisuusluokitettua tietoa, hyväksyy auditoinnin aina ulkoministeriö

(NSA). Kuvasta 2 nähdään, miten F-osa-alueen auditointiprosessi etenee. [5, s. 22.]



Kuva 2. Fyysisen turvallisuuden osa-alueen (F) arviointiprosessi [5, s. 23].

Kolmas ja sovelluskehityksen kannalta olennaisin osa-alue on teknisen tietoturvallisuuden osa-alue (I). I-osa-alue on jaettu kolmeen osaan, ja nämä osat ovat tietoliikenneturvallisuus, tietojärjestelmäturvallisuus ja käyttöturvallisuus. Osa-alueen tarkoitus on arvioida kohdeorganisaation kyky käsitellä turvallisuusluokitettua tietoa sähköisessä ympäristössä. Teknisen tietoturvallisuuden osa-aluetta (I) hyödynnetään tietojärjestelmän tietoturvallisuuden velvoitteiden toteutumisen arvioinnissa, jonka tekee liikenne- ja viestintäviraston (NSCA) osana turvallisuusluokiteltua tietoa käsittelevän tietojärjestelmän arviointiprosessia. Liikenne ja viestintäviraston (NCSA) tietojärjestelmän arviointiprosessi on tarkemmin havainnollistettu kuvassa 1 (luku 3, kuva 1). [5, s. 63–106.]

3.2 Pilvipalveluiden turvallisuuden arviointikriteeristö (PiTuKri)

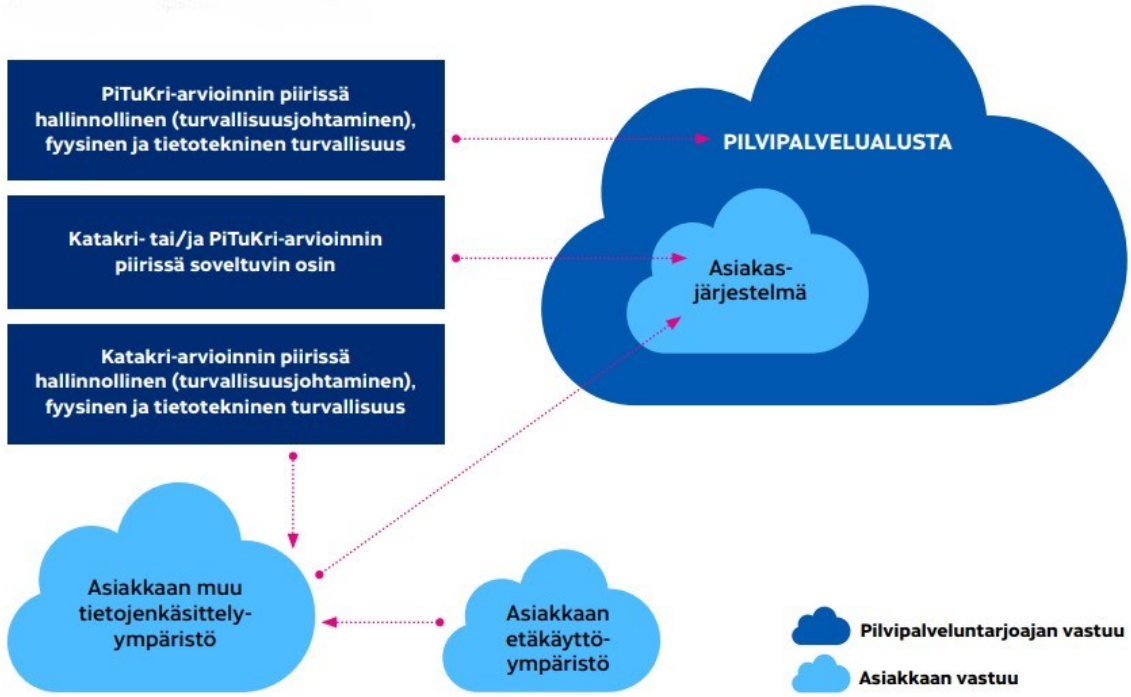
Pilvipalveluiden turvallisuuden arviointikriteeristö, lyhenteeltään PiTuKri on auditointityökalu pilvipalveluiden turvallisuuden arviointiin. PiTuKria voidaan hyödyntää apuna arvioitaessa pilviteknologioita käyttäviä tietojärjestelmiä, jotka sisältävät turvallisuusluokiteltua tietoa, henkilötietoa tai salassa pidettävää tietoa. Tarkennettuna kriteeristössä otetaan kantaa pääasiassa kansalliseen TL IV -tason turvallisuusluokiteltuun ja salassa pidettävään tietoon, mutta kriteeristössä sivutaan myös kansainvälistä RESTRICTED-tason turvallisuusluokiteltua tietoa, mikä turvallisuusluokituksien taulukosta (luku 2, taulukko 1) voidaan nähdä

vastaavan kansallisessa turvallisuusluokittelussa myös tasoa TL IV. Pilvipalveluihin voidaan säilöä paljon tietoa, joten arviointikriteeristössä otetaan kantaa myös turvallisuusluokitellusta tiedosta koostuviin kasaumiin (TL IV - ja TL III -kasauma). [8, s. 3.]

Auditointityökalun vaatimukset perustuvat pääasiassa kansalliseen lainsäädäntöön, ja lainsäädäntöön pohjautuvina lähteinä on käytetty lakia julkisen hallinnon tiedonhallinnasta (906/2019) sekä valtioneuvoston asetusta asiakirjojen turvallisuusluokittelusta valtionhallinnossa (1101/2019). Lainsäädännön ulkopuolisina lähteinä PiTuKrissa on käytetty Katakri 2015 -kriteeristöä, ISO27017 - ja ISO27001 -standardeja, CSA-pilviturvallisuusyhteisön suojausmatriisia sekä BSI:n pilviturvallisuuskriteeristöä. [8, s. 3.]

Liikenne- ja viestintäviraston (NCSA) julkaisu ensimmäisen version PiTuKrista vuonna 2019. Päivitysversion, joka toimii nykyisin uusimpana versiona, PiTuKri sai jo seuraavana vuonna 2020. Tämän päivitysversion nimi on PiTuKri 1.1, ja siihen on tehty uudistuksia käyttäjien palautteeseen perustuen. Uusimmassa versiossa on täsmennetty käsitteitä ja käyttötapauksia. [9; 8, s. 3.]

Auditointityökalua hyödynnetään osana liikenne- ja viestintäviraston tietojärjestelmän tietoturvallisuuden hyväksyntäprosessia (luku 3, kuva 1), mikäli tietojärjestelmä hyödyntää pilviteknologiaa [5, s. 64]. PiTuKria voidaan käyttää tarvittaessa elinkeinoelämän tarpeisiin tai viranomaisten julkisen tiedon turvallisuuden arviointiin [8, s. 3]. Pilvipalveluntarjoajien arviointiin voidaan käyttää PiTuKria, jos tarjoajat päättävät ottaa alustalleen turvallisuusluokiteltua tai salassa pidettävää tietoa käsitteleviä järjestelmiä [8, s. 4]. Tämän kaltaisessa tilanteessa vastuu turvallisuusluokitellun tai salassa pidettävän tiedon käsittely-ympäristöstä jaetaan tietojärjestelmän omistavan asiakkaan ja pilvipalvelutarjoajan välillä. (kuva 3).



Kuva 3. Pilvipalveluntarjoajan ja asiakkaan välinen käsittely-ympäristön vastuunjako [8, s. 4].

PiTuKri koostuu rakenteeltaan yhdestätoista osa-alueesta, jotka on esitetty seuraavalla tavalla:

- esiehdot (osa-alue 1)
- turvallisuusjohtaminen (osa-alue 2)
- henkilöstöturvallisuus (osa-alue 3)
- fyysinen turvallisuus (osa-alue 4)
- tietoliikenneturvallisuus (osa-alue 5)
- identiteetin ja pääsyn hallinta (osa-alue 6)
- tietojärjestelmäturvallisuus (osa-alue 7)
- salaus (osa-alue 8)
- käyttöturvallisuus (osa-alue 9)
- siirrettävyys ja yhteensopivuus (osa-alue 10)
- muutostenhallinta ja järjestelmäkehitys (osa-alue 11).

Jokaiselle osa-alueelle on esitetty osa-alueen vaatimukset, tietotyypit, johon vaatimukset kohdistetaan, vaatimusten soveltuvuuskohteet, suojaustavoitteet ja

toteuttamisen sekä tulkinnan tueksi asetetut lisätiedot. Osa-alueeseen 1 on asetettu esiehtoja tukemaan jatkoarvioinnin mahdollisuuksia nostaen tämän osa-alueen erityiseen asemaan. [8, s. 7.]

3.3 Julkisen hallinnon tietoturvallisuuden arviointikriteeristö (Julkri)

Julkisen hallinnon tietoturvallisuuden arviointikriteeristö, lyhenteeltään Julkri on valtionvarainministeriön julkaisema ja tiedonhallintalautakunnan sekä tietosuojavaltuutetun toimiston yhteistyössä toteuttama suositus tukemaan julkisen hallinnon tietojärjestelmien tietoturvallisuuden kehittämistä ja arviointia [10, s. 7]. Julkrin avulla voidaan arvioida julkisen hallinnon tietojärjestelmien tietoturvallisuuden velvoitteiden toteutumista tietojärjestelmissä, jossa käsitellään salassa pidettävää tietoa, henkilötietoa tai turvallisuusluokiteltua tietoa [10, s. 10]. Arviointikriteeristössä ei oteta kantaa kansainvälisen turvallisuusluokitetun tiedon tietoturvallisuuden arviointiin [10, s. 10].

Julkrin keskeisimpinä lähteinä on käytetty kansallista lainsäädäntöä ja EU:n tietosuoja-asetusta. Suomen lainsäädäntöön pohjautuvat lähteet ovat valtioneuvoston asetus asiakirjojen turvallisuusluokittelusta valtionhallinnossa (1101/2019), laki viranomaisen toiminnan julkisuudesta (621/1999) sekä tietosuojalaki (1050/2018). Tietosuojaa koskevissa asioissa on myös huomioitu EU:n yleinen tietosuoja-asetus ((EU) 2016/679). Myös muita viranomaisten arviointikriteeristöjä on hyödynnetty yhteensopivuuden varmistamiseksi, ja nämä kriteeristöt ovat Katakri ja PiTuKri. [10, s. 8.]

Julkri koostuu rakenteellisesti muiden viranomaisten arviointikriteeristöjen tavalla osa-alueista. Nämä osa-alueet ovat:

- hallinnollinen turvallisuus (HAL)
- fyysinen turvallisuus (FYY)
- tekninen turvallisuus (TEK)
- varautuminen ja jatkuvuudenhallinta (VAR)
- tietosuoja (TSU).

Hallinnollisen turvallisuuden (HAL) osa-alue kattaa tietojärjestelmien käyttöönoton ja hankinnan, henkilöstöturvallisuuden ja hallinnollisen turvallisuuteen liittyviä vaatimuksia. HAL-osa-alueen tarkoitus on varmistaa vastuuhenkilöiden luotettavuus ja varmistaa kohdeorganisaation tietoturvallisuuden menettelyjen sekä hallintajärjestelmien riittävä turvallisuus. [10, s. 12–13.]

Fyysisen turvallisuuden (FYY) osa-alueessa arvioidaan kohdeorganisaation fyysisten toimitilojen teknisiä sekä fyysisiä rajoitteita ja estoja. Osa-alueessa otetaan myös kantaa tiedon fyysiseen säilytykseen sekä käsittelyyn. FYY-osa-alueen kriteerit pohjautuvat Katakrissa esitettyihin vaatimuksiin, etenkin turvallisuusluokitellun tiedon tapauksissa. [10, s. 13–15.]

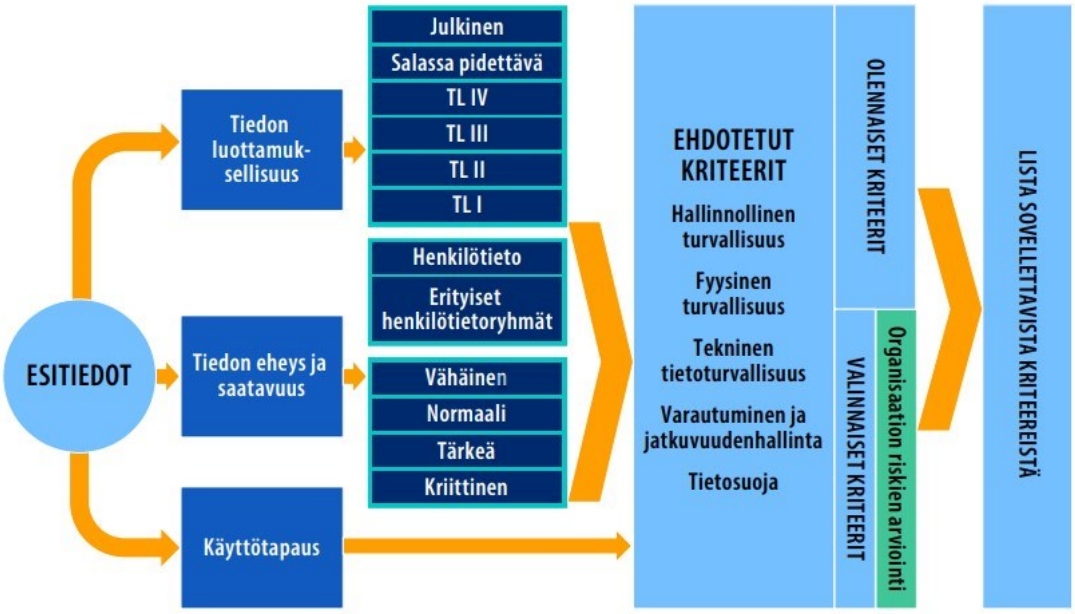
Teknisen turvallisuuden (TEK) osa-alueeseen kuuluu tarkemmin tietojärjestelmien ja tietoliikenteen teknisen tietoturvallisuuden vaatimuksia, mukaan lukien käyttö- ja toimintamallit. Arvioidaan laajalle kokonaisuudelle tai yksittäiselle tietojärjestelmälle tai tietojenkäsittely-ympäristölle. TEK-osa-alueen kriteerit kattavat yleisien teknisten tietoturvallisuuden velvoitteiden toteutumisen sekä mahdollisesti myös teknisten tietosuojavelvoitteiden toteutumisen. [10, s. 15.]

Varautumisen ja jatkuvuudenhallinnan (VAR) osa-alue koostuu normaalioloissa tapahtuvaan varautumiseen sekä jatkuvuudenhallintaan. Osa-alueen ulkopuolelle on jätetty valmiuslaissa määritellyt poikkeusolojen jatkuvuudenhallinnan vaaditut toimenpiteet, mutta kriteeristö kuitenkin tukee osin poikkeusoloihin varautumista. VAR-osa-alueen kriteerit on kohdistettu pääasiassa kohteisiin, jotka on luokiteltu kriittisiksi tai saatavuudeltaan tärkeiksi yhteiskunnan toiminnalle. Osa-alueen kriteerien keskeisimpinä lainsäädännön lähteinä on käytetty lakia julkisen hallinnon tiedonhallinnasta (906/2019) 4 §:n 2. mom:n 2. kohtaa, 13 §:n 1., 2. ja 4. momenttia sekä 15 §. Myös ISO/IEC 27002 -standardia on hyödynnetty tietoturvallisuuden jatkuvuudenhallinnan toimenpiteiden vaatimuksiin. [10, s. 15–16.]

Tietosuojan (TSU) osa-alueeseen on koottu henkilötietojen tietosuojaan ja tietoturvalliseen käsittelyyn liittyviä vaatimuksia. Tällaisia vaatimuksia ovat

esimerkiksi rekisteröidyn oikeudet, tietosuojaperiaatteet ja tiedon käsittelyn lainmukaisuutta koskevat vaatimukset. [10, s. 16.]

Julkri kriteeristöön pohjautuvassa arvioinnissa otetaan huomioon, minkälaista toimintaa kohdeorganisaatio harjoittaa, ja sen perusteella rajataan, suoritetaanko auditointi koko organisaatioon, vai tiettyyn organisaation osaan. Auditoinnissa huomioidaan myös se, minkälaista tietoa kohdeorganisaatio käsittelee järjestelmissään ja käsitteleekö organisaatio esimerkiksi henkilötietoja. Mikäli kohdeorganisaatio pyrkii saamaan hyväksyntätodistuksen Julkri-auditoinnista, tulee organisaation täyttää kaikki kriteerit niin kuin ne on määritelty kriteeristössä tai perustellulla vastaavalla lainsäädäntöön pohjautuvalla hyväksyttävällä tavalla. Kuvassa 4 on havainnollistettu tarkemmin sitä, miten Julkriin pohjautuvassa auditoinnissa sovellettavien kriteerien valinta tapahtuu. [10, s. 23–24.]



Kuva 4. Julkriin perustuvan auditointiprosessin sovellettavien kriteerien valinta [10, s. 24].

4 Ohjeistukset ja standardit turvallisen ohjelmistokehityksen tukena

Tietoturvallisen ohjelmistokehittämisen prosessin sekä sovelluksen turvallisen arkkitehtuurin takaamiseksi on suunnittelua sekä kehitystä tehtävä rinnastaen aihepiirin kuuluvia ohjeistuksia ja standardeja. Viranomaisten tietoturva-auditointien kriteeristöissä on teknisille toteutuksille jätetty liikkumavaraa, mutta ohjelmistokehityksen turvallisuuden toteutumiseksi vaaditaan ohjeistuksiin ja standardeihin pohjautuva kehitys [8, s. 92; 10, s. 105]. Standardien sekä ohjeistuksien käytöstä ohjelmistokehityksen tukena on kriteeristöjen vaatimuksissa esitetty Katakrin [8, s. 92] kohdassa I-13 ja Julkrin kohdassa [10, s. 104–105] TEK-14. Tässä pääluvussa esitellään kriteeristöissä suositeltuja ohjeistuksia sekä standardeja, joita tulee soveltaa auditoinnin läpäisemiseen pyrkivän sovelluksen toteuttamisessa.

4.1 OWASP Application Security Verification Standard (ASVS)

Open Worldwide Application Security Project (OWASP) on avoin kansainvälinen järjestö, jonka tavoitteena on kehittää ohjelmistojen turvallisuutta koulutuksen, työkalujen ja yhteistyön kautta. OWASP:lla on kymmeniä tuhansia jäseniä ja yli 250 osastoa ympäri maailmaa. Järjestö tuottaa avoimia standardeja, työkaluja, menetelmiä, artikkeleita ja muita vapaasti käytettäviä resursseja tietoturvallisen ohjelmistokehityksen tueksi. [11.]

Yksi OWASP-yhteisön tuottama avoin tietoturvallista ohjelmistokehitystä tukeva standardi on OWASP Application Security Verification Standard (ASVS). ASVS:n tarkoitus on tuoda vaatimuksia sisältävä viitekehys tukemaan turvallisen ohjelmiston suunnittelua, kehitystä ja testausta moderneissa sovelluksissa. Viimeisin versio ASVS:tä on versio 4.0.3, jossa on korjattu ja täsmennetty viimeisimmän laajan versiopäivityksen 4.0 vaatimuksia. Viimeisimmässä laajassa versiopäivityksessä 4.0 on suurimpana muutoksena se, että ASVS:ää on otettu mukaan NIST 800-63-3 digitaalisen identiteetin ohjeistus. [12, s. 9.]

Rakenteellisesti ASVS on jaettu neljääntoista osa-alueeseen, jotka kattavat vaatimuksia ohjelmiston tekniseen arkkitehtuurilliseen suojaukseen. Osa-alueisiin merkitään lyhenne ”V”, ja lyhenteeseen lisätään numerojärjestyksessä osa-alueen numero. Nämä osa-alueet ovat:

- arkkitehtuuri, suunnittelu ja uhkamallinnus (V1)
- autentikointi (V2)
- sessionhallinta (V3)
- validointi, sanitointi ja (en)koodaus (V4)
- pääsynhallinta (V5)
- tallennettu salaus (V6)
- virheiden hallinta ja lokitus (V7)
- tiedon suojaus (V8)
- tiedonvälitys (V9)
- haitallinen ohjelmakoodi (V10)
- liiketoimintalogiikka (V11)
- tiedostot ja resurssit (V12)
- ohjelmointirajapinnat ja web-palvelut (V13)
- konfiguraatio (V14). [12.]

Standardin jokaiseen osa-alueen vaatimukseen on eroteltuna vaatimuksia kolmelle eri tasolle (level). Näistä tasoista ensimmäinen kattaa helposti löydettävät haavoittuvuudet ja suojaukset heikolla tasolla suoritettaviin hyökkäyksiin. Toinen taso on taso, jota suositellaan suurimmalle osalle ohjelmistoista, koska tämän tason suojaukset suojaavat suurimmalta osalta ohjelmistoihin kohdistuvista riskeistä. Viimeisin ja standardin turvallisin taso on sen kolmas taso, ja sitä suositellaan käytettäväksi, mikäli ohjelmisto on osana kriittistä infrastruktuuria, maanpuolustusta tai muuta tärkeää kohdetta. [12, s. 12.]

OWASP:lla on myös avoin turvallisen mobiilisovelluskehityksen standardi MASVS (Mobile Application Security Verification Standard). Tämä standardi muistuttaa paljon ASVS:ää, mutta se ei ole yhtä yksityiskohtainen tehtävistä suojauksista kuin ASVS on. [13.]

4.2 VAHTI: Sovelluskehityksen tietoturvaohje

VAHTI-ohjeistus, kokonimeltään ”VAHTI: Sovelluskehityksen tietoturvaohje” on valtiovarainministeriön alaisen vuosina 1992–2013 toimineen valtiohallinnon tietoturvallisuuden johtoryhmän (VAHTI) vuonna 2013 julkaisema tietoturvallisen sovelluskehityksen ohjeistus [14, 15]. Ohjeistuksen tarkoituksena on tukea julkishallinnon organisaatioiden ohjelmistokehitystä, ohjelmistojen ylläpitoa sekä hankkeiden läpivientiä tietoturvallisuuteen liittyvissä asioissa [14, s. 12].

VAHTI-ohjeistus perustuu vanhentuneeseen lainsäädäntöön. Keskeisimpänä kansallisen lainsäädännön lähteenä on käytetty tietoturvallisuusasetusta (681/2010) [14, s. 12]. Vuonna 2019 tulleissa kansallisen lainsäädännön muutoksissa laki julkisen hallinnon tiedonhallinnasta (906/2019) määrittelee julkisen hallinnon organisaatioiden tietoturvallisuuden vaatimukset tietoturvallisuusasetuksen (681/2010) sijaan [15]. Myös teknologiat ja ohjelmistokehityksen tavat ovat muuttuneet ohjeistuksen julkaisuvuodesta 2013 merkittävästi, joten ohjeistusta on lähestyttävä kriittisesti sekä ottaa kehityksen tueksi myös muita uudempia ohjeistuksia sekä standardeja turvallisuuden takaamiseksi. Puitteistaan huolimatta ohjeistus on kuitenkin otettu esille tässä insinöörityössä, sillä ohjeistuksen käyttöä näistä asioista huolimatta suositellaan turvallisen ohjelmistokehityksen tukena uusimmissa Julkrin ja Katakrin versioissa, joissa on huomioitu lainsäädännössä tapahtuneet muutokset [8, s. 92; 10, s. 105].

VAHTI-ohjeistus kattaa laajasti tietoturvallisen ohjelmistokehityksen eri osa-alueita. Ohjeistuksessa otetaan kantaa tietoturvallisuuden näkökulmasta sovelluksen arkkitehtuuriin, projektinhallintaan, laadun varmistamiseen, dokumentointiin, sovelluskehityksen vaiheisiin sekä ympäristön vaatimuksiin. Ohjeissa on myös sivuttu tekijänoikeuksiin ja avoimien tietoverkkojen tietoturvallisuuteen liittyviä erityiskysymyksiä. [14.]

4.3 Kyberturvallisuuskeskuksen ohjeistus

Kyberturvallisuuskeskus on liikenne- ja viestintäviraston alainen viranomainen, jonka tehtävänä on suojella kansallista digitaalista infrastruktuuria uhilta, ylläpitää kansallista tietoturvallisuuden tilannekuvaa ja valvoa tietoturvallisuuden vaatimusten toteutumista [16]. Kuten jo aiemmin mainittua, NSCA-toiminto kuuluu kyberturvallisuuskeskukselle [17, s. 6].

Kyberturvallisuuskeskuksen ohjeistus auttaa ohjelmistojen valmistajia tuottamaan laadukkaita ja tietoturvallisia ohjelmistoja [17, s. 7]. Ohjeistuksessa käsitellään riskianalyysiin, ohjelmiston suunnitteluun, kehitykseen, testaukseen, ylläpitoon sekä käyttöönottoon liittyviä asioita tietoturvallisuuden näkökulmasta [17, s. 3].

Ohjeistus tuo myös ainutlaatuisia näkökulmia siitä, mihin tietoturvallisuuden tarkastajat keskittyvät auditoinneissa. Ohjeistus eroaa muista siten, että siinä on englanninkielinen käännös. Ohjeistuksen hyödyntäminen on suositeltavaa ohjelmistokehityksen tukena.

5 Turvallisen mobiiliratkaisun toteuttaminen

Turvallisen mobiiliratkaisun toteuttaminen on laaja prosessi, ja siinä on otettava huomioon kaikki ratkaisun osa-alueet laitteistosta sovellukseen. Tässä pääluvussa on huomioitu turvalliseen sovelluskehitykseen ja ylläpitoon liittyvät konseptit mobiiliratkaisun kokonaiskuvan saavuttamiseksi.

5.1 Sovelluksen toteutustavan valinta

Mobiiliratkaisun suunnittelu aloitetaan sovelluksen toteutustavan valinnasta, jos mahdollinen sovelluksen toimeksiantaja tai muu osapuoli ei ole pyytänyt tiettyä toteutustapaa sovellukselle. Toteutustavat voivat erota toisistaan merkittävästi, ja sovelluksen toteutustavan valinnassa on otettava huomioon tekniset

vaatimukset, suoritusnopeus, projektin aikataulu, saatavilla olevat resurssit ja sovellustyyppeihin kohdistuvat tietoturvallisuuden eroavaisuudet.

Natiivi sovellus

Natiivilla sovelluksella tarkoitetaan laitteeseen asennettavaa sovellusta, joka on kehitetty tietylle mobiilialustalle [18, s. 6; 19]. Tällaisia mobiilialustoja ovat esimerkiksi Android ja iOS. Natiivikehitystä tehdään käyttäen alustan tukemia työkaluja sekä ohjelmointikieliä. Erilaisten alustojen välillä natiivikehityksessä voi olla käytössä hyvinkin erilaisia kirjastoja, rajapintoja sekä ohjelmointiparadigmoja [18, s. 6].

Natiivisovelluksien erityinen hyöty on natiivisovellusten nopea suoritusnopeus sekä mahdollisuus ottaa kaikki irti käytettävästä mobiililaitteesta. Natiivissa kehityksessä on mahdollista päästä käsiksi mobiililaitteen sekä käyttöjärjestelmän sisäisiin toiminnallisuuksiin. Tällaisia toiminnallisuuksia ovat esimerkiksi laitteen käyttäjän yhteystiedot, tiedostot, sijaintitiedot, mikrofoni, kamera ja muut laitteen sensorit sekä käyttöjärjestelmän toiminnallisuudet [19]. Etenkin tietoturvallisuuden kannalta kiinnostavia toiminnallisuuksia voivat olla laitteen biometriset tunnisteet ja näiden hyödyntäminen sovelluksen autentikoinnissa. Käyttäjäkokemus on yleisesti parempi natiivisovelluksissa kuin web-pohjaisissa sovelluksissa johtuen siitä, että natiivisovellukset eivät ole selaimesta riippuvaisia.

Koska natiivikehitykseen kuuluu alustarajoituksia ja kyseisellä alustalla käytetään ainoastaan alustan työkaluja sovelluskehitykseen, voi sopivien kehittäjien löytäminen olla haastavaa, etenkin pienemmillä organisaatioilla, joiden ydinosaaminen ei ole mobiilisovelluskehityksessä. Esimerkiksi jos lähdetään rakentamaan natiivisovellusta Androidille, kehitykseen tarvitaan tiimi Java- tai Kotlin-ohjelmistokehittäjiä, joilla on ymmärrystä Androidin ekosysteemistä, työkaluista, turvallisuudesta sekä kehitystavoista. Tällaisen tiimin kasaaminen voi mahdollisesti vaatia organisaatiolle, jolla ei ole käytettävissä natiivikehittäjiä, uuden henkilöstön palkkaamista tai laajamittaista kouluttamista. Jos myöhemmin tulee

tarve saada sovellus toiselle alustalle, tarvitaan taas uusi tiimi alustakohtaisella osaamisella.

Turvallisuuden näkökulmasta natiivi sovellus on turvallisin vaihtoehto, sillä natiivit sovellukset on integroitu läheisesti laitteen käyttöjärjestelmän kanssa antaen käyttöön alustan tarjoamia turvallisuusominaisuuksia. Arkkitehtuurillisesti mobiilissa käyttöjärjestelmässä käytetään sovelluksille hiekkalaatikoita. Käytännössä tämä tarkoittaa sitä, että jokaiselle käyttöjärjestelmän sovellukselle annetaan oma ainutlaatuinen käyttäjätunnus (UID) ja jokainen prosessi ajaa omassa virtuaalikoneen avulla eriytetyssä hiekkalaatikossaan. Hiekkalaatikot ovat myös omia käyttäjiään käyttöjärjestelmän sisällä oikeuksien rajoittamiseksi. Tämä tarkoittaa sitä, että hiekkalaatikoilla on oletetusti rajoitettu pääsy käyttöjärjestelmän ominaisuuksiin ja ominaisuuksilla ei ole pääsyä toisiin sovelluksiin. Tämä eriyttäminen mahdollistaa sen, että mikäli jokin sovellus on haitallinen, ei tällä sovelluksella ole pääsyä muihin sovelluksiin. Hiekkalaatikko on kuitenkin mahdollista rikkoa, mutta tämä vaatisi käyttöjärjestelmän ytimeen pääsyä. [20; 21.]

Alustariippumaton sovellus

Alustariippumaton sovellus, eli hybridisovellus on sovellus mitä voidaan kehittää käytettäväksi useammalle alustalle. Teknisesti alustariippumattomat sovellukset ovat web-sovelluksia, joita ajetaan natiivisovelluksen sisälle upotetussa selaimessa [22]. Koska alustariippumattomat sovellukset ovat natiivisovelluksen sisäisiä web-sovelluksia, tarvitaan natiivisovelluksen käyttöliittymän tuntuman saamiseksi erilaisia käyttöliittymän kehitystyökaluja [22]. Tällaisia alustariippumattoman sovelluksen käyttöliittymän kehityksen tukena olevia työkaluja ovat esimerkiksi React Native, Ionic ja Flutter. Alustariippumattomat sovellukset asennetaan natiivisovelluksen tavalla mobiililaitteelle ja alustariippumattomilla sovelluksilla on vain yksi koodikanta, joka käännetään ja ajetaan kaikilla alustoilla [19]. Alustariippumattomalla sovelluksella on myös pääsy liitänteitä hyödyntäen natiivisovelluksen tavalla mobiililaitteen sisäisiin toiminnallisuuksiin [22].

Taloudellisesta näkökulmasta katsoen alustariippumattomalla toteutustavalla voi säästää rahaa, sillä tarvitaan vain yksi tiimi kehittämään sovellusta. Tällaiseen projektiin sopivien kehittäjien löytäminen organisaatiossa voi mahdollisesti olla helpompaa kuin natiivikehityksessä, sillä kehitystä tehdään web-teknologioita ja ohjelmointikieliä käyttäen. Alustariippumattoman sovelluskehityksen työkalut voidaan tarvittaessa normaalille web-kehittäjälle kouluttaa todella nopeasti, tehden projektiin sopivien ihmisten etsimisestä huomattavasti helpompaa.

Vaikka alustariippumattomat sovellukset tarjoavat liitänteiden kautta pääsyn laitteen sisäisiin toiminnallisuuksiin, on kehittäjä liittänteiden rajoitteiden varassa. Tästä voi esiintyä ongelmia komplekseja sovelluksia tehtäessä. Alustariippumattoman sovelluksen suoritusnopeus on heikompi natiivisovellukseen verrattaessa.

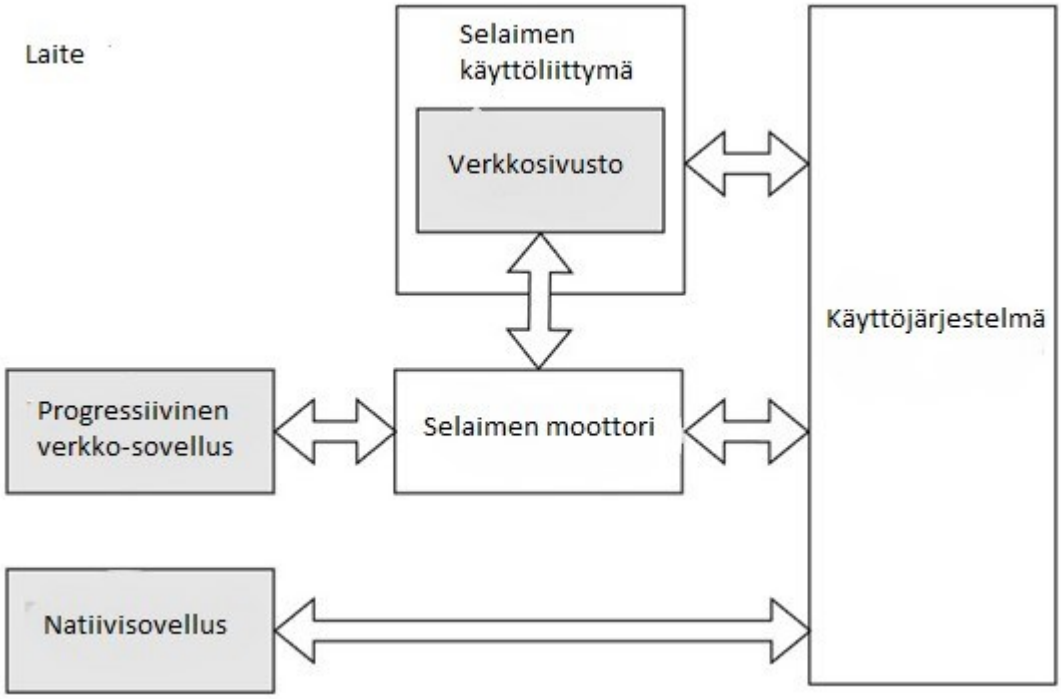
Turvallisuuden näkökulmasta alustariippumattomilla sovelluksilla voi olla enemmän haavoittuvuuksien riskejä, kuin puhtaasti natiivissa sovelluksessa. Tällaisia haavoittuvuuksia voivat tuoda esimerkiksi kehityksessä mahdollisesti käytettävät JavaScript-pohjaiset kehykset ja kirjastot, joten on suositeltavaa seurata kehyksille tai kirjastoille kohdistettuja parhaita turvallisuuskäytäntöjä. [13, s. 13.]

Web-sovellus

Web-sovelluksella tarkoitetaan sovellusta, johon pääsee käsiksi käyttäen mobiililaitteen selainta. Tällainen sovellus kehitetään käyttäen web-kehittämisen työkaluja, kuten JavaScript-ohjelmointikieltä, HTML-kuvauskieltä, CSS-tyylitiedostoja ja käyttöliittymäkehyksiä sekä kirjastoja.

Web-sovelluksen tekeminen on ehdottomasti halvin ja nopein tapa tuottaa mobiililaitteelle sopeutuva sovellus, mutta tämä rajoittaa sovellukseen saatavia ominaisuuksia. Vaikka nykypäivänä selaimilla on pääsy joihinkin laitteen toiminnallisuuksiin, kuten sijaintitietoihin ja kameraan, ei web-sovelluksella voida kuitenkaan hyödyntää kaikkia toiminnallisuuksia samalla tavalla kuin natiivi tai alustariippumattomalla sovelluksella voidaan [22]. Web-sovellukset ovat myös huomattavasti natiivisovelluksia hitaampia.

Käyttäjäkokemus voi web-sovelluksen käyttäjälle olla huonompi, kuin natiivin tai alustariippumattoman sovelluksen käyttäjälle. Olemassa on tapoja tehdä web-sovelluksesta käyttäjäystävällisempi mobiilikäyttäjälle. Yksi tapa on tehdä progressiivinen web-sovellus (progressive web application tai PWA). Kuten kuvasta 5 voidaan nähdä, PWA esitetään ilman selaimen käyttöliittymää, mikä tekee käyttäjäkokemuksesta natiivimaisemman. PWA:t tarjoavat myös muita ominaisuuksia kuten sovellusikonin laitteeseen, offline- ja taustatoiminnallisuuden kytkykyyksiä sekä muita kiinnostavia normaalista web-sovelluksesta poikkeavia toiminnallisuuksia [22; 23].



Kuva 5. Erilaisten sovellusten toteutusmuotojen ja käyttöjärjestelmän välinen vuorovaikutus mobiililaitteessa [23].

Yksi huomattava hyötypuoli web-toteutustavassa on päivityksien tuominen käyttäjille vaatimatta käyttäjien interaktiota. Web-sovelluksessa voi kehittäjät viedä uuden version tuotantoon, kun taas natiivi ja alustariippumattomissa sovelluksissa pitää laitteelle päivittää sovelluksen uusin versio.

5.2 Ympäristön turvaaminen ja auditoitavuus

Viranomaisen auditoinnin läpäisyyn pyrkivässä mobiiliratkaisussa tulee sovelluksen lisäksi myös koko tietojenkäsittely-ympäristö olla suojattuna sekä auditoitavissa. Tässä luvussa käsitellään tarkemmin ympäristön suojaukseen, auditoitavuuteen sekä päätelaitteen valintaan liittyviä seikkoja.

Ympäristön kovennus

Oletusarvoisesti järjestelmät eivät ole turvallisia, vaan niissä on paljon ominaisuuksia, jotka eivät ole välttämättömiä käyttövaatimuksille. Tällaisten turhien toiminnallisuuksien pitäminen järjestelmässä tuo suuremman haavoittuvuuspinta-alan ja hankaloittaa riskien tunnistusta [5, s. 80].

Kansalliseen ja kansainväliseen lainsäädäntöön on asetettu tietojärjestelmiä ja tietoliikennejärjestelyjä koskevia vaatimuksia [3, 11 §; 24, liite IV]. Näiden velvoitteiden toteutumiseksi on tunnistettava ympäristöön kuuluvat järjestelmät ja toteutettava tarvittavat kovennuksen toimenpiteet.

Kovennus on järjestelmän toiminnallisuuksien vähentämistä siten, että järjestelmässä on vain käyttövaatimusten kannalta olennaiset ominaisuudet, oikeudet ja ohjelmistot. Kovennuksessa lähdetään liikkeelle ensin tunnistamalla kovennettavat kohteet. Tällaisia kohteita voi olla mobiiliratkaisussa esimerkiksi palvelimet, mobiililaitteet, työasemat ja muut oleelliset järjestelmät. Kun kohteet on tunnistettu, tulee määritellä kovennuksen toteutus ja suorittaa kovennus määritelmien mukaisesti. Kovennuksen jälkeen järjestelmästä tulee olla poistettuna kaikki ylimääräiset palvelut, ohjelmistot ja yhteydet. On jatkuvasti huolehdittava siitä, että tehdyt kovennukset pysyvät järjestelmässä päällä ja että tarpeelliset ohjelmistot ovat ajan tasalla. [5, s. 81; 10, s. 95–96.]

Kovennuksen suorittamisen tueksi on olemassa paljon ohjeistuksia. Näiden kovennusohjeistuksien käyttö on myös vaadittua turvallisuusluokiteltua tietoa käsittelevien ympäristöjen kovennuksissa. TL IV -tason ympäristön kovennuksessa riittää järjestelmäkohtaisesti yhden ohjeistuksen hyödyntäminen, mutta

TL III - ja TL II -turvallisuusluokitetun ympäristön kovennuksessa tulee hyödyntää useampaa kovennusohjetta tai tiukentaa ohjeistuksen toteutusta [5, s. 81].

Yksi ohjeistuksia tuottava taho on Yhdysvaltain puolustusministeriön (DoD) alainen tietojärjestelmävirasto DISA. DISA tuottaa myös julkisesti avoinna olevia teknisiä ohjeistuksia (STIG) perustuen Yhdysvaltain puolustusministeriön ja NIST:n vaatimuksiin [25]. Näitä ohjeistuksia voi hyödyntää, kun kovennuksia toteutetaan. Ohjeistuksia voi löytää esimerkiksi NIST:n tarjoamasta ”National Checklist Program (NCP)” -tietokannasta [26].

Päätelaitteen valinta

Päätelaite käsitteenä on laaja, koska sillä voidaan tarkoittaa mobiiliratkaisun kontekstissa esimerkiksi työasemia tai mobiililaitteita [27, s. 14]. Päätelaitteella tarkoitetaan kuitenkin tässä kontekstissa mobiililaitetta, jossa suunniteltavaa sovellusta tullaan kehittämään.

Suurin osa markkinoilla olevista päätelaitteista on suunnattu tavallisille kuluttajille. Tällaisten päätelaitteiden valitseminen turvallisuusluokitettua tietoa käsitteleviin mobiiliratkaisuihin on täysin sallittua, jos valmistaja on luotettava ja tarvittavat järjestelmäkovennukset tehdään laitteelle kovennusohjeen mukaisesti.

Kovennuksien toteuttaminen voi olla työlästä, joten markkinoilla on myös valmiiksi kovennettuja päätelaitteita, jotka on suunnattu viranomaiskäyttöön. Viranomaiskäyttöön tarkoitettuja päätelaitteita valmistaa esimerkiksi suomalainen Bittium, joka tarjoaa jopa TL III -tasolle sertifioituja päätelaitteita [28]. Myös osa tunnetuista valmistajista tuottaa kuluttajille suunnattuihin päätelaitteisiin laajempaa turvallisuutta kaipaaville asiakkaille erilaisia lisäominaisuuksia. Esimerkiksi Samsung tarjoaa Knox-nimistä turvallisuusalustaa mobiililaitteilleen, ja Knox-alusta on läpäissyt Katakri 2015 -kriteeristön sekä monen muun valtion viranomaisen auditoinnin [29].

Kaikkia laitevalmistajia ei kuitenkaan voida pitää luotettavina. Kyseenalaisessa asemassa ovat etenkin kiinalaiset valmistajat, kuten Huawei ja ZTE. EU on

luokitellut Huawein ja ZTE:n turvallisuusriskeiksi ja kieltänyt näiden valmistajien tuotteiden käytön EU:n komission sisäisissä verkoissa sekä kehottanut jäsenmaita kieltämään kyseisten valmistajien käyttämisen mobiiliverkkojen rakentamisessa [30]. Varsinaista kieltoa ei ole kansallisesti asetettu kiinalaisille valmistajille, mutta esimerkiksi suojelupoliisi [31] on todennut Kiinan kohdistavan Suomeen kybervakoilua muiden valtioiden ulkopoliittisten näkemyksien ja teknologisen tuotekehitystiedon keräämiseksi. Tällaiset asiat heikentävät kiinalaisten valmistajien luotettavuutta ja voivat mahdollisesti olla este kansainvälisten tietoturvallisuuden velvoitteiden toteutumiseksi, joten turvallisuusluokiteltua tietoa käsittelevän mobiiliratkaisun päätelaitteen valitsemisessa on suotavaa jättää kiinalaiset valmistajat pois harkinnasta.

Jäljitettävyys

Jäljitettävyydellä tarkoitetaan lokitiedon keräämistä ympäristön tapahtumista, jotta saadaan selkeä ymmärrys järjestelmässä tapahtuneista tapahtumista myöhempää analyysia varten. Tapahtumien keräämisellä voidaan ehkäistä luottamuksellisen tiedon luvatonta käsittelyä ja edistää järjestelmän auditoitavuutta [5, s. 85].

Tyypillisesti lokitietoa kerätään ympäristöstä (mobiililaitteet, työasemat, palvelimet ja verkkolaitteet) sekä itse kehitettävästä sovelluksesta. Yleensä tietoa kerätään käyttäjäaktiviteetista, tietoturvallisuuden poikkeuksista sekä järjestelmän toiminnallisuudesta. [5, s. 85.]

Kun sovelluksen riskianalyysia toteutetaan, määritellään mistä osista sovellusta lokitietoa on kerättävä. Tyypillisesti sovelluksessa lokimerkintä tehdään rajapintakutsun yhteydessä.

Lokitietoa voidaan kerätä esimerkiksi seuraavissa tilanteissa:

- huomattava määrä epäonnistuneita tunnistautumisen yrityksiä
- luottamuksellisen tiedon tai henkilötiedon poistaminen, lukeminen tai muuttaminen (lokimerkintään on merkittävä kenen tietoja ja mitä tietoja käsiteltiin sekä kuka näitä tietoja käsitteli)
- käyttäjien valtuuksien muutokset
- henkilötietojen tai käyttöehtojen suostumuksen muutokset
- epäonnistuneet valtuutusyritykset
- epäonnistuneet kutsut
- istuntotunnisteiden tarkastuksen mahdolliset virheet sekä muutokset istunnon tilassa.

Lokimerkinnät tulee tallentaa reaaliajassa ja niissä pitää olla aikaleima (UTC) yhtenäisessä muodossa. Luettavuuden kannalta on suositeltavaa tallentaa lokimerkinnät JSON-objekteina. [32, s. 71–72.]

Lokitietoa kerätään keskitetylle turvallisesti suojatulle palvelimelle, jota vain tietyt henkilöt pääsevät käyttämään. Tämän palvelimen lokitieto olisi suositeltavaa varmuuskopioida erilliseen järjestelmään kerran päivässä, jotta voidaan välttää tilanteita, jossa palvelimelle käy jotakin, ja lokitieto häviää samalla. Tiedon säilyttämisen aika riippuu säilytettävästä tiedosta. Mikäli tietoa koskee viranomaistoiminnan rikosoikeudelliset vanhentumisajat, tulee tietoa säilyttää ainakin 5 vuotta palvelimella. Yleisesti muuta, kuin viranomaistoiminnan rikosoikeudellista vanhentumisaikaa koskevaa tietoa voidaan TL IV -tason ympäristössä säilyttää ainakin 6 kuukautta. Korkeamman turvallisuusluokan tietoa käsittelevissä ympäristöissä tulee lokitietoa säilyttää palvelimella tiedosta riippuen 2–5 vuotta. [5, s. 86.]

5.3 Mobiilisovelluksen suojaus

Varsinaisen toteutettavan mobiilisovelluksen suojaaminen on mobiiliratkaisun tietoturvallisuuden keskeisin osa. Se, mitä suojauksia kuuluu toteuttaa ja miten nämä suojaukset toteutetaan, riippuu kehitettävästä sovelluksesta sekä sovelluksen toteutustavasta. Tässä luvussa on otettu esille tyypillisiä sovellukselle

tehtäviä suojauksia, ja luvussa on pyritty huomioimaan tarvittavat suojaukset erilaisten toteutustapojen näkökulmista. Sovelluksen tietoturvallisuuden turvaamiseksi on sovelluksen suunnittelussa ja toteutuksessa perehdyttävä tarkemmin jo edellä mainittuihin standardeihin, ohjeistuksiin sekä arviointikriteereihin.

Toteutettavasta sovelluksesta riippumatta on tärkeänä turvallisuutta lisäävänä tekijänä huomioitava, että sovelluksen kuuluu olla yksinkertainen, sillä ylimääräinen ohjelmakoodi lisää sovelluksen haavoittuvuuspinta-alaa ja tekee riskientunnistamisesta monimutkaisempaa. Yksinkertaisella, mahdollisimman vähän turhia toiminnallisuuksia sisältävällä sovelluksella voidaan turvallisuuden lisäämiseksi tehdä merkittäviä taloudellisia säästöjä ohjelmistokehityksessä sekä viranomaisen tarkastuksen kulujen yhteydessä [17, s. 23]. Tämä johtuu siitä, että on vähemmän kehitettävää sekä tarkastettavaa, johon taloudellisia resursseja kuluisi. Joten suunnitteluvaiheessa tulee pohtia, minkälaisia komponentteja sovellus oikeasti tarvitsee ja lähtökohtaisesti yrittää tehdä sovelluksesta mahdollisimman yksinkertainen, kuitenkaan tinkimättä laadusta, tietoturvallisuudesta tai käytettävyydestä.

Riskianalyysi

Mobiilisovellukseen kohdistuvien riskien tunnistaminen on yksi tärkeimpiä asioita sovelluksen turvallisuuden takaamiseksi. Riskianalyysissa, eli uhkamallinnuksessa otetaan selvää siitä, minkälaisia uhkia kohdistuu sovellukseen, mitkä ovat sovelluksen kriittisimmät osat, mitä tietoa pitää erityisesti suojata, mitä voi mennä pieleen ja missä hyökkäys voi tapahtua [33].

Tyypillinen riskianalyysi toteutetaan tiimin kesken, joka koostuu sovelluksen arkkitehtuurin tuntevista toimihenkilöistä. Suositeltavaa riskianalyysiin olisi kutsua paikalle mahdollisen asiakasorganisaation sekä pilvipalvelun tarjoajan edustajia, jotta voidaan saada kokonaiskuva sovellukseen kohdistuvista uhkista. Riskianalyysin toteutus tapahtuu työpajamaisesti, ja aikaa on hyvä varata reilusti, sillä prosessi on erittäin tärkeä sovelluksen turvallisuudelle. [32, s. 48.]

Riskianalyysissa on hyvä lähteä liikkeelle mallintamalla yleisempiä mobiilisovelluksiin kohdistuvia uhkia, joita löytää esimerkiksi OWASP:n Mobile Top 10 -listauksesta. Suositeltua olisi käyttää olemassa olevia riskianalyysin malleja. Yksi tällainen riskianalyysiin hyödynnettävä malli on Microsoftin STRIDE-menetelmä.

STRIDE-menetelmä on yleisessä käytössä oleva riskianalyysin toteutuksen tukena oleva työkalu. Työkalun tarkoitus on helpottaa riskianalyysin toteuttajia tunnistamaan uhkia sekä helpottamaan turvallisuuteen liittyvää keskustelua kategorisoimalla erilaiset uhkatilanteet. Menetelmän nimi STRIDE muodostuu uhkien kategorioista, nämä kategoriat ovat:

- **S**poofing (huijaaminen/väärentäminen). Todennuksen väärentäminen.
- **T**ampering (peukalointi). Tiedon muokkaaminen.
- **R**epudiation (kiistettävyys). Mahdollisuus kiistää sovelluksessa tehdyt asiat.
- **I**nformation Disclosure (tiedon paljastuminen). Tiedon vuotaminen ulkopuolisille toimijoille, joiden ei pitäisi päästä tietoon käsiksi.
- **D**enial of Service (palvelun esto). Sovelluksen saatavuus ja palvelunestohyökkäykset.
- **E**levation of Priviledge (oikeuksien ylittäminen). Käyttöoikeuksien ylittäminen yli sallittujen oikeuksien.

STRIDE:n hyödyntäminen selkeyttää uhkien ymmärtämisen kaikille osapuolille, myös osapuolille, joilla ei ole teknistä taustaa. [32, s. 49; 34.]

Riskien kokonaiskuvan saavuttamiseksi, on tärkeää ymmärtää käsiteltävään tietoon ja mahdolliseen asiakasorganisaatioon kohdistuvat uhkat. Auditointityökaluissa ja laissa tietoturvallisuuden vaatimukset on esitetty riippuen tiedon turvallisuusluokasta, mutta tietoon voi kohdistua laajempaa uhkaa myös turvallisuusluokasta riippumattomista, kuten poliittisista syistä. Tämän kaltaisten laajempien ja turvallisuusluokista riippumattomien uhkatilanteiden havainnointi on tärkeää, ja mikäli tämän kaltainen tilanne havaitaan, on syytä ottaa esille tietoturvallisuuden koventaminen yli laissa vaadittujen vähimmäisvaatimuksien.

Ohjelmointirajapintojen turvaaminen

Ohjelmointirajapinnat koostuvat protokollista ja määritelmistä sekä ohjelmointirajapintoja hyödyntäen sovellukset voivat keskustella sujuvasti keskenään. Kuten muussakin ohjelmistokehityksessä, törmää myös mobiilisovelluskehityksessä usein ohjelmointirajapintoihin. Ohjelmointirajapintojen toteuttamiselle on useita erilaisia tyylejä, esimerkiksi SOAP-protokolla ja REST-arkkitehtuuri. Tässä luvussa rajaus on kuitenkin tehty REST-arkkitehtuurille johtuen sen suosiosta mobiilisovelluskehityksessä.

REST on arkkitehtuurin muoto ohjelmointirajapinnan toteuttamiselle, ja se muodostuu englanninkielisistä sanoista Representational State Transfer. REST, toisin kuin SOAP, ei ole protokolla, mikä tarkoittaa sitä, että se voidaan toteuttaa erilaisilla tavoilla ottaen kuitenkin huomioon arkkitehtuurilliset vaatimukset, jotta rajapintaa voidaan kutsua REST-ohjelmointirajapinnaksi. Tällaiset arkkitehtuurilliset vaatimukset ovat:

- välimuistiin tallennettava tieto, jolla parannetaan asiakkaan ja palvelimen vuorovaikutusta
- tiedon vakiomuodon siirron saavuttamiseksi yhtenäinen rajapinta komponenttien välillä
- palvelintyyppien kerroksellinen hierarkiassa oleva järjestelmä, joka on näkymätön asiakkaalle
- tilaton yhteys asiakkaan ja palvelimen välillä
- HTTP-protokollan kutsuja hyödyntävä palvelimista, asiakkaista sekä resursseista koostuva asiakkaan ja palvelimen välinen arkkitehtuuri.

REST-arkkitehtuurin suosiota on lisännyt sen helppo toteutus, nopeus, keveys ja helppo skaalautuvuus. Varsinkin keveys ja nopeus ovat vaikuttavia asioita, kun mobiilisovellusta toteutetaan, joten soveltuu REST-arkkitehtuuri tällaiseen ratkaisuun. [35.]

Koska REST ei ole protokolla tai standardi, antaa se kehittäjälle valtuudet päättää tiedon esitysmuodon. Turvallisuuden kannalta kuitenkin esitysmuodon tulee olla yhtenäinen koko sovelluksessa. Tyypillisiä tiedon esitysmuotoja ovat

JavaScript Object Notation (JSON) ja eXtensible Markup Language (XML). Digi- ja väestöviraston turvallisen sovelluskehityksen käsikirjassa [32, s. 74] mainitaan, että JSON-esitysmuotoa tulisi käyttää uusien ohjelmointirajapintojen toteuttamisessa johtuen sen hyvästä ohjelmistotuesta sekä yksinkertaisesta esitystavasta. Käsikirjassa [32, s. 75] mainitaan siitä, että XML-jäsentimissä on virheellisyyksiä tiedon tulkitsemisessa. Näiden perusteitten takia olisi suositeltavaa käyttää JSON-esitysmuotoa muiden sijaan.

Tiedon esitysmuodon eheyden validointi on ohjelmointirajapintojen turvallisuuden kannalta tärkeää. JSON-skeema tulee huolellisesti validoida ennen sen syöttämistä. Skeeman validoinnin lisäksi on suositeltavaa validoida JSON-oliot sekä niiden arvot. [12, s. 56–57.]

Myös varsinaiselle REST-ohjelmointirajapinnalle on asetettava tiedon esitysmuodon validoinnin lisäksi turvallisuuden takaamiseksi suojauksia. Tällaisia suojauksia voivat olla seuraavat asiat:

- Ohjelmointirajapinnan verkko-osoitteessa ei saa vahingossa paljastaa mitään kriittistä tietoa.
- Ohjelmointirajapinnassa tarkistetaan, että ylätunnisteessa tuleva mediatyyppi (Content-Type) on muotoa application/json.
- Ohjelmointirajapinnassa varmistetaan, että käytössä olevat HTTP-metodit vastaa toiminnallisuutta tai käyttäjän roolia.
- Ohjelmointirajapinnan tiedon ja kutsun eheys tarkistetaan esimerkiksi sitomalla eheystarkistus salaisuuteen.
- Ohjelmointirajapinnassa varmistetaan, että vain ylläpitäjällä on pääsy suorittamaan ylläpitoon liittyviä kutsuja.
- Ohjelmointirajapinnassa varmistetaan, että ylätunnisteesta puuttuvat ja odottamattomat mediatyypit hylätään ja hylätylle kutsulle palautetaan sopivat HTTP-metodit (puuttuva: 415, odottamaton: 406).
- Web-sovelluksissa varmistetaan, että HSTS-tunnisteen (HTTP Strict Transport Security) käyttö on pakollista TLS-yhteyden pakottamiseksi.
- Web-sovelluksissa varmistetaan, että CSRF-tunnistetta hyödynnetään.

Näiden suojauksien lisäksi myös yhteyksissä tulisi hyödyntää TLS-protokollaa tai muuta turvallista salausmenetelmää ylätunnisteiden ja tiedon suojaamiseksi. [12, s. 56–57.]

CORS-politiikan (Cross-Origin Resource Sharing) konfigurointi on myös tärkeä osa turvallisuutta. CORS on tapa, jolla voidaan sallia JavaScript-ohjelmien resurssipyynnöt erilaisten lähteiden välillä ohittaen niin sanotun same-origin-rajoituksen [32, s. 76]. Sovelluksen CORS-politiikka riippuu toteutettavasta sovelluksesta, mutta tyypillisesti turvallisuussyistä Allow-Control-Access-Origin ei saa olla arvoa null ja myös wildcard-arvoa pitää käyttää varovasti, sillä se sallii kaikki lähteet [12, s. 62; 32, s. 76].

Tiedostojen turvallinen käsittely

Useassa sovelluksessa käsitellään tiedostoja, etenkin Android-pohjaisissa sovelluksissa paikallinen säilytys tapahtuu pääsääntöisesti tiedostoilla. Tiedostojen turvallisella käsittelyllä voidaan vähentää esimerkiksi sovellukseen kohdistuvia palvelunesto- ja SSRF-hyökkäyksiä muiden hyökkäyksien ohella [12, s. 54].

Tiedostojen latauksessa voidaan sovellukseen kohdistaa palvelunestohyökkäyksiä. Palvelunestohyökkäys voidaan toteuttaa suojaamattomaan sovellukseen lataamalla suuria tiedostoja, jotka täyttävät sovelluksen tiedostojen säilytyksen. Tällaisten hyökkäyksien estämiseksi on sovellukseen asetettava tiedoston maksimi koko sekä rajoitettava yksittäisen käyttäjän tiedostojen latauksen määrää tietyllä aikavälillä. [12, s. 54.]

Turvallisuusluokiteltua tietoa käsittelevässä sovelluksessa ei todennäköisesti tiedostoja ladata epäluotettavasta lähteestä, mutta kuitenkin tällaisessa tapauksessa pitää varmistaa, että tiedoston tyyppi vastaa odotettua tiedoston tyyppiä. Myös epäluotettavista lähteistä ladattavat tiedostot pitää skannata mahdollisilta viruksilta. [12, s. 54–55.]

SSRF-hyökkäyksessä hyökkääjä yrittää huijata palvelinpuolen sovellusta lähettämään palvelimelle kutsuja, joita hyökkääjän ei kuuluisi lähettää näin

pääsemällä käsiksi tietoon, johon hyökkääjän ei kuuluisi päästä. SSRF-hyökkäys on erittäin vaarallinen sovelluksen turvallisuudelle, sillä hyökkäyksen avulla hyökkääjä voi mahdollisesti syventää hyökkäystä tunnistamalla sisäisen verkon IP-osoitteita, hyökkääjä voi skannata avoimia portteja tai lukea tietoa, jonka lukemiseen hyökkääjällä ei pitäisi olla oikeuksia. SSRF-hyökkäykseen hyökkäysvektoreina voidaan käyttää verkko-osoitteita tai validoimattomia syötteitä, mutta mahdollisesti myös tiedostoja. Kun käyttäjä lähettää tiedoston palvelimelle, voi tiedoston nimen metadataan olla upotettuna jonkin näköinen kutsu. Tällaisen tilanteen välttämiseksi kuuluu myös tiedostojen nimet joko validoida tai jättää kokonaan huomioimatta. Tyypillisesti myös SSRF-hyökkäykseltä puolustautumiseksi tulee sovelluksen palvelimelle määritellä vain tietyt järjestelmät taikka resurssit, joihin palvelin voi lähettää kutsuja tai ladata tietoa. [36.]

Syötteiden validointi

Sovelluksen syötteiden validoinnissa tarkistetaan syötetyn tiedon muodon oikeellisuus, minkä avulla voidaan estää sovellukseen kohdistuvia hyökkäyksiä. OWASP:n [12, s. 35] mukaan yleisin tietoturvallisuuden virhe sovellusta toteuttaessa on syötteiden huono validointi. Syötteiden validoinnilla voidaan vähentää jopa 90 prosenttia web-pohjaisiin sovelluksiin kohdistuvista hyökkäyksistä. Web-pohjaisella sovelluksella tarkoitetaan web-sovellusta sekä alustariippumatonta sovellusta johtuen alustariippumattoman sovelluksen arkkitehtuurista, jossa natiiviin sovellukseen upotetaan web-selain. Tällaisia hyökkäyksiä ovat esimerkiksi injektio -, XSS - ja puskurin ylivuotovirhe -hyökkäykset. Tyypillisesti natiiviin sovellukseen kohdistuu huomattavasti vähemmän XSS-hyökkäyksiä johtuen siitä, etteivät natiivit sovellukset vaadi selainta, jollei sitä erikseen sovellukseen upoteta. Kuitenkin natiivit sovellukset ovat yhtä haavoittuvaisia muille hyökkäyksille, joten syötteet pitää validoida huolellisesti turvallisuuden edistämiseksi.

Tarvittava validointi riippuu syötteestä ja siinä käsiteltävästä tiedosta, mutta olennaisia asioita ovat kuitenkin:

- vahvan kirjoitusmuodon ylläpitäminen
- loogisuuden tarkastus
- pituuden tarkastus
- yleisen kaavan seuraaminen.

Myös erilaiset syötteisiin kohdistuvat hyökkäykset on otettava validoinnissa huomioon, ja syötteet ovat validoitava näiden varalta. Varsinaisen validoinnin lisäksi myös tiedon ulostulon koodaus tulee olla mahdollisimman lähellä ohjelmatulkkia. [12, s. 35–38.]

Käyttäjien tunnistautuminen

Tunnistamalla käyttäjä voidaan varmistaa käyttäjän identiteetti. Identiteetin varmistus on tärkeää, jotta voidaan estää ulkopuolisten pääsy sovelluksen käyttäjätileille. Mobiilisovellukset tarjoavat useita normaalista poikkeavia tunnistuksen menetelmiä ja tässä osiossa esitetään tapoja, miten niitä voidaan hyödyntää tyypillisen salasanapohjaisen tunnistuksen tukena.

Tyypillisin tunnistuksen tapa on tunnistautua käyttäen käyttäjänimeä ja salasanaa. Turvallisen salasanan valitsemisesta on tullut vaikeampaa teknologian kehittyessä, sillä salasanoja on helpompi rikkoa. Käytännössä kuitenkin salasanan kuuluu olla tarpeeksi pitkä, sillä tämä lisää salasanan mahdollisia permutaatioita, mikä vaikeuttaa salasanan rikkomista. Toinen tärkeä asia on salasanan kompleksisuus. Salasanan ei kuulu olla looginen, sillä logiikka helpottaa salasanan arvaamisen mahdollisuutta. Kun sovellusta tehdään, ei voida kuitenkaan aina luottaa siihen, että käyttäjä valitsee turvallisen salasanan, vaan sovelluksessa on vaadittava salasanan asettamiseen ehtoja.

Tällaisia ehtoja voi esimerkiksi olla:

- Salasana on ainakin 12 kirjainta pitkä.
- Yli 64 kirjainta tai pidemmät salasanat ovat sallittuja.
- Salasanalle ei aseteta vaatimuksia koskien erikoismerkkejä, pieniä tai isoja kirjaimia.
- Syötetyt salasanat testataan yleisesti rikottujen salasanojen tietokantaan.
- Käyttäjä voi vaihtaa salasanan, ja salasanan vaihtamiseen tarvitaan vanha salasana ja vahva tunnistautuminen.
- Ohjeistus käyttäjälle turvallisen salasanan luomiseksi, esimerkiksi salasanan vahvuus palkin merkein.

Monissa sovelluksissa on salasanan asettamiselle vaadittu erikoismerkkien, pienien tai isojen alkukirjaimien käyttö. Nojautuen ASVS:n [12, s. 22] ohjeistukseen ei minkäänlaisia merkkien vaatimuksia saa salasanan asetukselle laittaa, sillä joukko-opin mukaan kaikenlaisten vaatimusten asettaminen salasanalle pienentää mahdollisten salasanojen määrää. Suositeltavaa olisi ohjeistaa käyttäjää luomaan turvallinen salasana, esimerkiksi salasanan vahvuus palkin merkein. Salasanojen pituuteen kuuluu asettaa kuitenkin vaatimus, sillä lyhyiden salasanojen rikkominen on huomattavasti helpompaa kuin pitkien. [12, s. 21–22.]

Pelkästään käyttäjänimen ja salasanan perusteella tunnistautuminen ei kuitenkaan ole turvallista, sillä se mahdollistaa käyttäjätilin kaappauksen sosiaalisella manipuloinnilla, esimerkiksi tietojenkalastelulla. Käyttäjätilin kaappauksen mahdollisuuden vähentäminen on sovellukseen toteutettava monivaiheinen tunnistautuminen. Monivaiheisella tunnistautumisella tarkoitetaan sitä, että yhden tunnistautumisen menetelmän lisäksi käytetään myös toista tai useampaa tunnistautumisen menetelmää. Tunnistautuminen monivaiheisesti alkaa olemaan jo ohjelmistokehityksessä erittäin suosittu menetelmä tietoturvallisuuden lisäämiseksi, ja tällaisen menetelmän toteuttaminen kirjautumisen yhteydessä on vaadittua ottaen huomioon turvallisuusluokiteltua tietoa käsittelevän sovelluksen kriittisyyden. Huomioitavaa kuitenkin on, että kaikki tunnistautumisen

menetelmät eivät ole yhtä turvallisia esimerkiksi tekstiviestin tai sähköpostin käyttämistä tunnistautumista ei pidetä yhtä luotettavina kuin muita menetelmiä [12, s. 22].

Mobiilisovelluksiin voidaan toteuttaa useita normaalista poikkeavia tunnistautumisen menetelmiä hyödyntäen mobiililaitteen sensoreita, ja tämän avulla voidaan tehdä monivaiheisesta tunnistautumisesta käyttäjäystävällisempää. Tällaista tunnistautumisen tapaa kutsutaan biometriseksi tunnistautumiseksi. Biometrinen tunnistautuminen on erittäin käyttäjäystävällinen tunnistautumistapa ja se soveltuu hyvin toiseksi tunnistautumistavaksi monivaiheiseen tunnistautumiseen. Kriittisissä sovelluksissa biometrinen tunnistautuminen ei ole sellaisenaan riittävä. NIST:n erikoisjulkaisun [37] mukaan biometrisen tunnisteen heikko tietoturvallisuus johtuu siitä, että biometriset piirteet eivät ole varsinaisia salaisuuksia, ja niissä nojaudutaan todennäköisyyteen eivätkä ne ole deterministisiä. Biometrisen tunnistautumisen hyödyntäminen osana monivaiheista tunnistautumista on kuitenkin sallittu mobiilisovelluksissa, niin kauan kuin biometrisen tunnistautumisen lisäksi vaaditaan jokin muu tunnistautumisen tapa.

Sovelluksen toteutustavalla on merkitystä sille, voidaanko biometrisiä tunnistautumisen menetelmiä hyödyntää. Natiivissa ja natiiviliitäntöjen kautta alustariippumattomissa sovelluksissa saa biometriset tunnistukset helposti käyttöön, mutta web-pohjaisissa sovelluksissa tarvitaan kolmannen osapuolen ohjelmointirajapintoja ja näiden rajapintojen turvallisuutta on suositeltavaa arvioida erikseen.

Suunniteltaessa sovellusta olisi suositeltavaa toteuttaa myös uusi tunnistautuminen kriittisten toiminnallisuuksien tai tietoon pääsyn yhteydessä. Myös ylläpitopaneelissa tehtäviin kriittisiin muutoksiin voidaan vaatia useamman käyttäjän varmistusta ennen kuin toiminnallisuus suoritetaan [5, s. 76].

Lisäksi muita tapoja tunnistautumisen turvaamiseksi tulee toteuttaa. Esimerkiksi voidaan sallia vain sidosryhmien VPN:ien IP-osoitteista tulevia kirjautumisyrityksiä tai sallia vain tietyistä maista tai alueilta tulevia kirjautumisyrityksiä.

Kuitenkaan enempää kuin 100 kirjautumisyritystä ei saisi tulla tunnin sisällä, ja mikäli yrityksiä tulee paljon, olisi tästä hyvä tallentaa lokimerkintä.

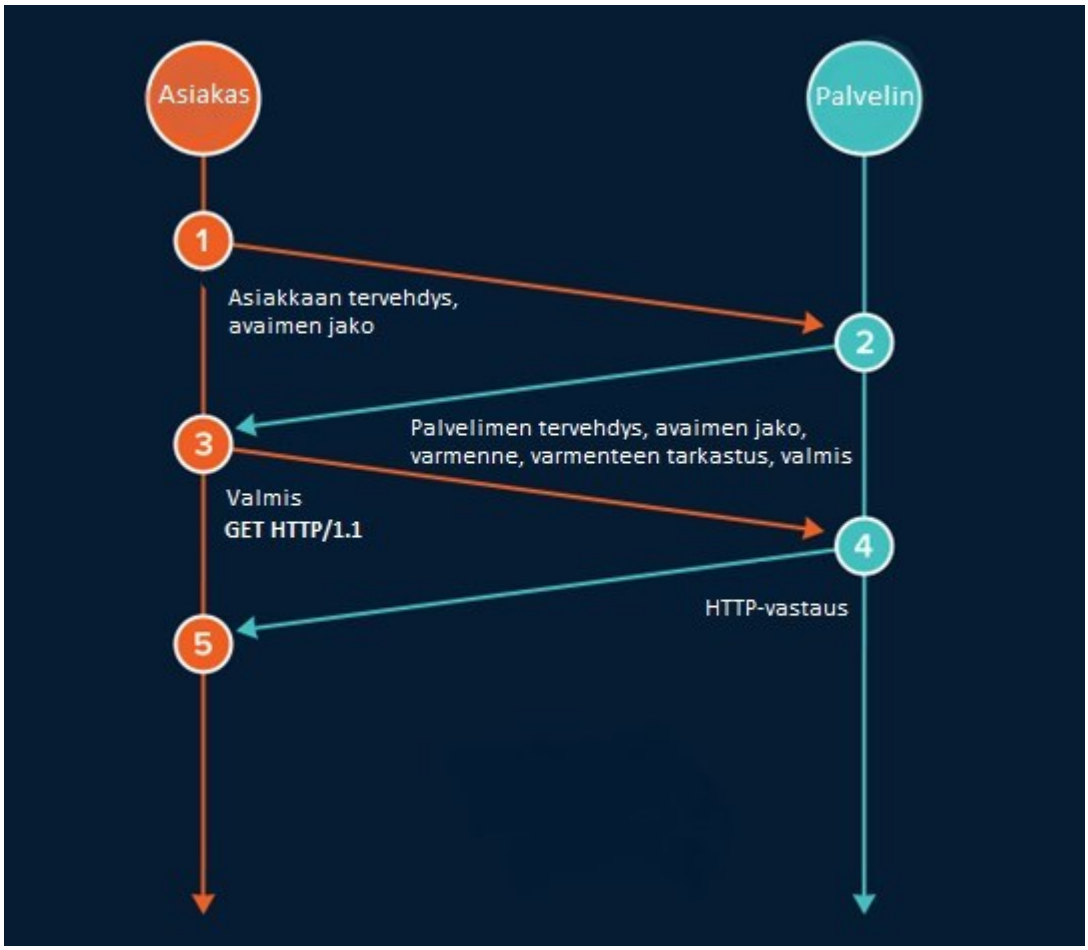
Yhteydet

Tyypillisesti mobiiliratkaisuissa sovellus keskustelee jonkin palvelimen kanssa, jolloin sovelluksen ja palvelimen välille syntyy yhteys. Näitä yhteyksiä voi olla useita varsinkin, jos kyseessä on laajempi sovellus. Tämä yhteys asettaa mahdollisuuden ulkopuoliselle taholle kaapata sovelluksen ja palvelimen välillä kulkevaa tietoa, joten ratkaisussa on varmistettava, että tarpeelliset suojaukset on asetettu yhteyksille.

Yhteyden suojaamiseen on käytettävä salattua yhteyttä. Yhteyden salaamiseksi on olemassa salausprotokollia. Salausprotokollista yleisimmässä käytössä ja vahvemman suojauksen takia suositeltava on edeltäjänsä SSL-protokollan (Secure Sockets Layer) korvaajaksi tullut TLS-protokolla (Transport Level Security). TLS-protokolla on käytössä HTTPS-yhteyden salauksessa, ja sitä voidaan myös hyödyntää esimerkiksi sähköposti- tai VoIP-yhteyksien salaamisessa [38].

Yhteyden salaamisen kokonaiskuvan saamiseksi on ensin ymmärrettävä, mikä on SSL/TLS-varmenne. Varmenne toimii käytännössä digitaalisena passina, jolla voidaan varmentaa yhteyden toisen osapuolen identiteetti. Tämä varmenne on datatiedosto, joka sisältää salaukseen käytettävän julkisen avaimen, myöntäjän tiedot sekä digitaalisen allekirjoituksen, TLS-version ja olennaiset päivämäärät. [38; 39.]

TLS-yhteys avataan suorittamalla TLS-kättely. TLS-kättely tapahtuu esimerkiksi tyypillisessä sovelluksen ja palvelimen välisessä yhteydessä TCP-yhteyden avaamisen tai siinä tapahtuvan ”kolmen käden kättelyn” jälkeen. TLS-kättely on havainnollistettu tarkemmin kuvassa 6. Kättelyssä luodaan istuntoavaimet, joita käytetään salaamiseen niin kauan kuin yhteys on päällä [38; 39].



Kuva 6. Asiakkaan ja palvelimen välinen TLS 1.3 -kättely [40].

Ei salattu yhteys kuitenkaan aina tarkoita turvallista yhteyttä. TLS-varmenteet eivät varsinaisesti vaadi minkään tietyn tahon allekirjoitusta, vaan niitä voidaan allekirjoittaa myös itse. Tämä käytännössä tarkoittaa sitä, että ulkopuoliset tahot voivat allekirjoittaa itse huonosti konfiguroituja TLS-varmenteita [41]. Tällaisten ja muiden varmenteiden väärinkäyttötilanteiden ehkäisemiseksi on sovellukseen toteutettava varmenteen pinnaus (certificate pinning). Varmenteen pinnauksessa määritellään vain tietyt varmenteet, joihin voidaan luottaa yhteyttä muodostaessa [41]. Varmenteen pinnauksen konfigurointi riippuu sovelluksen toteutustavasta, mutta esimerkiksi natiivia sovellusta kehittäessä Androidille pinnatut varmenteet määritellään tietoliikenteen turvallisuuskonfiguraationa Manifest-tiedostoon.

Istunnonhallinta

Käyttäjän istunto tyypillisesti syntyy sen jälkeen, kun käyttäjä on tunnistautunut sovelluksessa. Kuten mainittu, tämän tunnistautumisen kuuluu olla monivaiheinen, ja sama koskee uudelleen tunnistautumista ottaen huomioon käsiteltävän tiedon kriittisyyden [12, s. 30]. Istunnot voidaan kategorisoida kahteen eri kategoriaan, jotka ovat tilalliset ja tilattomat istunnot.

Tilallisessa istunnossa käyttäjän tunnistautumisen jälkeen generoidaan ainutlaatuinen istuntotunniste. Tämä tunniste toimii viittauksena käyttäjän tietoihin palvelimella. Tilallisessa istunnossa ainutlaatuinen tunniste tallennetaan paikallisesti esimerkiksi web-sovellusta kehittäessä keksiin, ja tämä keksi lähetetään palvelimelle jokaisen kutsun yhteydessä, keksin sisällä olevan tunnisteen perusteella palvelin voi tunnistaa käyttäjän. Kuitenkin mobiilisovellusta tehtäessä suositeltavaa olisi hyödyntää tilatonta istuntoa sen tarjoaman skaalatutuvuuden ja nopeuden takia. Natiiveissa sovelluksissa keksien implementointi on myös haastavaa. [42.]

Toisin kuin tilallisessa istunnossa, jossa istunnon tilaa on seurattava palvelimella, tilattomassa istunnossa ei tällaiselle ole tarvetta. Tilattomassa istunnossa käyttäjän tunnistamiseen tarvittavat tiedot tallennetaan niin sanottuun tokeniin. Yleisin käyttäjän istuntoon käytettävä tokeni on JWT (JavaScript Web Token), mutta muita käytettäviä tokeneita ovat esimerkiksi SAML ja OAuth. Rakenteellisesti JWT-tokenissa on ylätunniste, mikä sisältää salausalgoritmin, jolla digitaalinen allekirjoitus on luotu, esimerkiksi HSHA256 (HMAC SHA256). Salauksen lisäksi ylätunniste sisältää tokenin tyypin. Ylätunnisteen lisäksi tokeni koostuu myös varsinaisesta käyttäjän tunnistamiseen tarvittavasta tiedosta. Kuten keksit tilallisessa istunnossa, myös tilattomassa istunnossa tokenit lähetetään jokaisen palvelinkutsun yhteydessä käyttäjän tunnistamiseksi. [42.]

Tilattomassa istunnossa on tietoturvan takaamiseksi varmistettava, että tokenit ovat digitaalisesti allekirjoitettuja HMAC-salausalgoritmilla, esimerkiksi käyttäen edellä mainittua HSHA256:ta (HMAC SHA256) [42]. Tokenien generoinnissa

pitää huomioida, että tokenilla on ainakin 64-bittinen entropia [12, s. 29]. Tokenissa ei saa myöskään säilyttää kriittistä tietoa käyttäjästä. Kun natiivia sovellusta tehdään, kuuluu tokenit säilyttää turvallisessa paikassa paikallisesti, esimerkiksi Androidin KeyStoressa [42]. Vastaavasti web-sovelluksissa tokenit on tallennettava turvallisesti selaimeen, esimerkiksi suojattuihin kekseihin [12, s. 29].

Istunnot ovat vain hetkellisiä yhteyksiä ja turvallisuuden takaamiseksi istunnot kuuluu katkaista. Tyypillisesti katkaiseminen tapahtuu käyttäjän ulos kirjautumisen, salasanan vaihdon tai käyttäjän toimimattomuudesta johtuvan aikakatkaisun yhteydessä. Istuntojen jatkuminen aiheuttaa tietoturvallisuuden kannalta ongelmia, ja tämän takia tulee varmistaa, että istunto on katkaistu uloskirjautumisen tai aikakatkaisun yhteydessä. [12, s. 30.]

Käyttäjän identiteetin varmistamiseksi tulee käyttäjän uudelleen tunnistautua tietyn aikavälein. Turvallisuusluokiteltua tietoa käsittelevässä sovelluksessa tästä pitää olla erittäin tiukka. ASVS:n [12, s. 30] mukaan korkeaa tietoturvan kaipaavissa sovelluksissa voi uudelleen tunnistautua, mikäli istunto on ollut päällä 12 tuntia tai jos käyttäjä on ollut toimimattomassa tilassa 15 minuuttia. Uudelleen tunnistautumisen täytyy aina olla monivaiheinen.

Salausmenetelmät

Kaikki luottamuksellinen tieto pitää suojata viranomaisen hyväksymällä salausmenetelmällä. Tyypillisesti salattavan tiedon piiriin kuuluu luottamuksellinen tieto (salassa pidettävä tai turvallisuusluokiteltu), käyttäjälle tai palvelulle kriittinen tieto sekä henkilötiedot.

Turvallisuusluokiteltua tietoa käsittelevän sovelluksen salausmenetelmien soveltuvuuden tarkastaa auditointiprosessissa liikenne- ja viestintäviraston alainen CAA (Crypto Approval Authority). Salausmenetelmien kryptografiset vaatimukset riippuvat siitä, minkä tason turvallisuusluokiteltua tietoa sovellus käsittelee. CAA ylläpitää ohjeistusta salauksien vähimmäisvaatimuksista

turvallisuusluokittain ja tästä kyseisestä ohjeistuksesta löytää tarkat tiedot erilaisten salausalgoritmien kryptografisista vahvuuksista. [43, s. 1.]

CAA:n ohjeistukseen [43] perustuen, yhteinen tekijä salausmenetelmissä on salauksen kryptografinen vahvuus. TL IV -tason tietoa käsitellessä tulee salauksen kryptografisen vahvuuden olla vähintäänkin 128-bittiä. Käytettävien bittien määrä voi heitellä 1–3 bittiä salausmenetelmässä, mutta kryptografinen vahvuus ei saa olla huomattavasti vähemmän kuin asetettu vähimmäisvaatimus on. Suositeltavaa olisi hyödyntää vahvempaa salausta, mikäli mahdollista, sillä ohjeistuksessa esitettävät vaatimukset ovat vain viranomaisen vähimmäisvaatimuksia. [43.]

Tietoliikenteen salauksessa tulee salaukseen hyödyntää jo työssä aikaisemmin käsiteltyä TLS-protokollaa tai tarvittaessa IPSec-protokollaa (Internet Protocol Security). Protokollasta riippumatta tulee protokollan viimeisintä versiota käyttää. [43, s. 4.]

Paikallinen säilytys

Paikallisella säilytyksellä tarkoitetaan sovelluksessa tai mobiililaitteessa sisäisesti säilytettävää tietoa. Tässä osiossa keskitytään pääsääntöisesti natiiveihin ja alustariippumattomiin sovelluksiin, sillä web-sovellukset eivät voi hyödyntää alustan tarjoamaa paikallista säilytystä samalla tavalla selaimen asettamien rajoitteiden takia.

Android hyödyntää tyypillistä tiedostojärjestelmää tiedon säilytykseen. Kun sovellusta toteutetaan, voi tietoa säilyttää joko julkisissa kohteissa, mihin useampi sovellus pääsee käsiksi, tai sovellukselle yksityisissä kohteissa.

Androidin kohteet tiedon säilytykselle ovat seuraavat:

- Sovelluskohtaiset tiedostot. Näihin tiedostoihin pääsee käsiksi vain sovellus. Tiedostot ovat sidonnaisia sovellukseen tarkoittaen sitä, että kun sovellus poistetaan, myös tiedostot poistuvat.
- Sovelluksen preferenssit. Preferenssit ovat avain-arvo-parin muodossa tallennettavaa tietoa. Ne ovat myös sidonnaisia sovellukseen ja yksityisiä sovellukselle. Preferensseihin ei kuitenkaan kuulu tallentaa mitään kriittistä tietoa.
- Dokumentit ja muut tiedostot. Dokumentit ja muut tiedostot tallennetaan julkiseen kohteeseen niin, että muut sovellukset pääsevät niihin. Nämä tiedostot eivät ole sovellukselle yksityisiä eivätkä tiedostot poistu sovelluksen poiston yhteydessä.
- Mediatiedostot. Mediatiedostoihin muut sovellukset voivat päästä ainoastaan, jos niille on annettu lupa lukea ulkoisia tiedostoja (READ_EXTERNAL_STORAGE). Mediatiedostot eivät poistu sovelluksen poiston yhteydessä.
- Paikalliset tietokannat. Kevyt paikallinen tietokanta kuten SQLite. Paikallisien tietokantojen tieto on sovellukselle yksityistä ja paikalliset tietokannat poistuvat sovelluksen poiston yhteydessä.

Turvallisuuden takaamiseksi tulee varmistaa, että tiedot tallennetaan oikeisiin kohteisiin, ja etenkin kriittisen tiedon kannalta tulee olla erittäin tarkka, ettei tieto päädy kohteeseen, jossa muut sovellukset pääsevät siihen käsiksi. [44.]

Kun tieto tallennetaan sovellukselle yksityiseen kohteeseen, tätä tietoa suojelee muilta sovelluksilta sovelluksen hiekkalaatikko. Kuitenkaan tieto ei sellaisenaan ole turvassa ulkopuolisilta tahoilta. Esimerkiksi hiekkalaatikon voi teknisesti rikkoa, mikäli hyökkääjä pääsee käyttöjärjestelmän ytimeen käsiksi [20]. Toinen tilanne voi esimerkiksi olla sellainen, jossa ulkopuolinen toimija pääsee mobiililaitteeseen fyysisesti käsiksi. Kriittisen tiedon paljastumisen ulkopuolisille välttämiseksi on tieto yksityiseen kohteeseen tallennuksen lisäksi suojattava sille sopivalla salausmenetelmällä.

5.4 Sovelluskehitys ja sovelluksen ylläpitäminen

Selkeiden ja harkittujen sovelluskehityksen sekä ylläpidon periaatteiden määrittäminen on tärkeää turvallisuuden ylläpitämiselle koko sovelluksen elinkaaren ajan.

Käyttäjien ja oikeuksien hallinta

Jatkuvalla käyttäjien ja oikeuksien hallinnalla voidaan vähentää sovelluksen hyökkäyspinta-alaa ja estää mahdolliset oikeuksien väärinkäyttötilanteet. Käyttäjien ja oikeuksien hallinnan prosessin tulee olla selkeä sekä dokumentoitu turvallisuuden takaamiseksi. Käyttäjien ja oikeuksien hallinnassa tulee seurata vähimpien oikeuksien periaatetta, mikä käytännössä tarkoittaa sitä, että vain tarpeelliset käyttäjät sekä oikeudet ovat voimassa ja kaikki muut tulee poistaa.

Projektin toimihenkilöillä tulee olla selkeät roolit siihen, kuka on vastuussa mistäkin tehtävästä. Tietyille vastuuhenkilöille tehtävien rajaaminen selkeyttää sen, kenelle vastuu hallinnasta menee ja kenelle ilmoitetaan mahdollisista muutoksista. Rajaamisella voi mahdollisesti vähentää myös sosiaalisen manipuloinnin hyökkäyksien pinta-alaa. Sovelluksen vastuuhenkilöiden tehtävät sekä sovelluksen ylläpitoroolit riippuvat paljon sovelluksesta, mutta suositeltavaa olisi erotella tehtävät toisistaan ja jakaa tehtävät eri henkilöille. Tällä pyritään välttämään tilannetta, jossa jollakin toimihenkilöllä on liikaa valtuuksia.

Käyttöoikeuksien sujuva hallinnointi on tärkeää projektin turvallisuuden kannalta. Oikeuksien hallinnan menetelmät tulee olla selkeästi dokumentoituna ja annettuna käyttöoikeuksien hallinnasta vastaavalle toimihenkilölle.

Käyttöoikeuksien hallinnoinnin kannalta tärkeitä menetelmiä ovat:

- Käyttöoikeuksien ajantasaisuuden tarkastaminen tietyn aikavälein (esim. 6 kk).
- Oikeuksien muutostilanteiden hallinnointi: Mitä kuuluu tehdä erilaisissa tilanteissa, mitkä vaativat oikeuksien korottamista tai laskemista.
- Käyttäjien muutostilanteiden hallinnointi: Miten toimitaan, kun esimerkiksi projektista lähtee toimihenkilö ja käyttäjä pitää poistaa, tai kun uusi henkilö tulee projektiin.

Nämä menetelmät ovat projektikohtaisia ja voivat vaihdella, mutta lähtökohtaisesti dokumentoitu prosessi käyttäjien ja oikeuksien hallinnalle kuuluu olla.

Riippuvaisuuksien hallinta

Sovelluksen riippuvaisuudet ovat olennainen osa sovelluskehitystä, ja kolmannen osapuolen komponenttien käyttö nopeuttaa sovelluskehitystä huomattavasti. Kun lähdetään liittämään kolmannen osapuolen riippuvaisuuksia ohjelmakoodiin, se tuo kuitenkin omat riskinsä sovelluksen turvallisuuteen esimerkiksi haavoittuvuuksien osalta. Näiden riskien pienentämiseksi on kehitettävä tiimin sisäisesti jonkinlainen yhtenäinen menetelmä riippuvuuksien valinnalle sekä hallinnalle. Riippuvaisuuksiin kohdistuvia haavoittuvuuksia on tärkeä seurata ja tarpeen tullen päivittää.

Erilaiset muuttujat tulee riippuvaisuuksien valinnassa huomioida. Etenkin avoimen lähdekoodin ja luotettavien tahojen riippuvaisuuksia tulee käyttää. Avoimen lähdekoodin riippuvaisuuksissa hyvä puoli on se, että ohjelmakoodi on avointa, joten ohjelmakoodissa ei ole mitään ylimääräistä tai mikäli on, niin se voidaan löytää koodia tutkimalla. Avoimen lähdekoodin riippuvaisuuksissa tulee myös arvioida kehittäjien luotettavuus, sillä jatkuva kolmannen osapuolen koodin katselmointi voi olla työlästä. Myös mahdolliset lisenssit tulee ottaa huomioon valinnassa. [17, s. 28.]

Ohjelmakoodin katselmoinnit

Ohjelmakoodin katselmointi on tyypillinen osa ohjelmistokehityksen prosessia. Katselmointien avulla voidaan edistää ohjelmakoodin laadukkuutta. Käytännössä ohjelmakoodin katselmointi tarkoittaa prosessia, jossa ennen kuin ohjelmakoodi siirtyy kehittäjän haarasta pääkehityshaaraan, katselmoi jokin toinen projektissa toimiva kehittäjä tämän ohjelmakoodin. Kyseinen katselmoija voi hyväksyä, tai mikäli kokee koodin puutteelliseksi, hylätä ohjelmakoodin siirtymisen kehittäjän omasta haarasta pääkehityshaaraan. Toisen kehittäjän mielipiteen saaminen ohjelmakoodin laadukkuudesta vähentää ohjelmakoodin virheellisyyttä.

Ohjelmakoodin laadukkuus vähentää haavoittuvuuksien muodostumista, mutta laadukkuudella on myös tärkeä tarkoitus sovelluksen auditoitavuudelle. Kyberturvallisuuskeskuksen [17, s. 28] mukaan perusteelliseen turvatarkastukseen kuuluu olennaisena osana ohjelmakoodin katselmus. Mikäli ohjelmakoodin tarkastajat eivät kykene navigoimaan ohjelmakoodissa kriittisiin osiin huonon laadun takia, voi auditoinnin läpäiseminen tämän takia hankaloitua [17, s. 28]. Ohjelmakoodin katselmointi auttaa tarvittavan laadukkuuden saavuttamista, mutta muita olennaisia asioita ovat hyvä dokumentointi ja selkeä versionhallinta.

Sovelluksen testaaminen

Sovelluksen testaus on tärkeä osa sovelluskehityksen prosessia. Testaamalla sovellus voidaan varmistaa, että sovellus toimii oikein ja vaatimusten mukaisesti. Testaus on osa laadunvarmistamista, ja kriittistä tietoa sisältävä sovellus tulee testata sovelluksen korkean laadun varmistamiseksi.

Sovelluksen testaukseen hyödynnetään erilaisia testausmenetelmiä. Tyypillisimpiä testausmenetelmiä ovat:

- Yksikkötestaus. Yksikkötestit ovat pieniä kehittäjien itse kirjoittamia automatisoituja testejä, joiden kuuluu kattaa suurin osa ohjelmakoodista. Yksikkötestauksessa on tarkoitus testata jokin tietty osa toiminnallisuuden ohjelmakoodista, ja testeissä kuuluu testata, toimiiko ohjelmakoodi oikein sekä normaaleissa että ääritapauksissa. Tämä on tehokas tapa tuoda kehittäjälle nopeasti tietoa siitä, mikäli ohjelmakoodissa on virheitä ja kehittäjien kuuluu nämä virheet korjata siten, että testit läpäistään.
- Komponenttitestaus. Komponenttitesteissä testataan yksilöllisesti jokin tietty ohjelmiston komponentti. Suositeltavaa olisi, että komponentit testattaisiin automatisoidusti ainakin kerran vuorokaudessa.
- Järjestelmätestaus. Järjestelmätestauksessa testataan koko järjestelmän koontiversio. Tyypillisesti koko järjestelmän testaus suoritetaan harvemmin kuin esimerkiksi komponenttitestaus johtuen sen laajuudesta. Järjestelmän testaaminen voi olla automatisoitua tai manuaalista riippuen testaukseen käytetyistä resursseista.
- Penetraatiotestaus. Penetraatiotestauksessa testataan sovelluksen tietoturvallisuutta. Tässä testauksen muodossa testaajat yrittävät manuaalisesti tunkeutua järjestelmään hyökkääjinä. Tyypillisesti penetraatiotestauksen tekevät organisaation sisäiset tai kolmannen osapuolen tietoturvallisuuden asiantuntijat.
- Stressitestaus. Stressitestauksella pyritään selvittämään sovelluksen kyky suoriutua suuresta kuormituksesta tai muista poikkeustilanteista.
- Fuzz-testaus. Fuzz-testauksessa testataan sovelluksen tietoturvallisuutta antamalla sovellukselle poikkeuksellisia ja tarkoituksella virheellisiä syötteitä. Tämä on kyberturvallisuuskeskuksen [16, s. 32] mukaan tyypillinen sovelluksen turvallisuuden testaamisen muoto turvallisuustarkastuksissa.
- Hyväksymistestaus. Kolmannen osapuolen suorittama testaus, jossa yhdistetään edellä mainittuja testausmenetelmiä ja koetetaan etsiä sovelluksesta virheitä.

Turvallisuuden tarkastuksessa asiantuntijat hyödyntävät todennäköisesti useampaa testausmenetelmää varmistaakseen, että sovellus on korkealaatuinen sekä tietoturvallinen. Tämän takia on tärkeää ylläpitää hyviä ja dokumentoituja testauskäytäntöjä. Myös kolmannen osapuolen suorittamilla hyväksymistestauksilla voidaan tarkistaa sovelluksen laatua etukäteen ennen varsinaista

viranomaisen suorittamaa testausta, ja testauksesta saaduilla tuloksilla voidaan nopeuttaa viranomaisen prosessia. [17, s. 30–34.]

5.5 Sovelluksen jakelu

Sovelluksen jakelulla tarkoitetaan varsinaisen sovelluksen jakamista käyttäjille. Jakelun muoto vaihtelee riippuen sovelluksen toteutustavasta. Esimerkiksi natiivissa sovelluksessa välitetään pakettitiedosto, joka asennetaan laitteiseen, kun taas verkkosovelluksessa sovellukseen päästään käsiksi verkkoselaimen välityksellä. Tässä luvussa keskitytään pääsääntöisesti natiivien sekä alustariippumattomien sovelluksien jakeluun.

Tyypillisesti jos kuluttajalle tehdään sovellusta, sovelluksen jakeluun hyödynnetään yleisimpiä jakelukanavia, kuten Google Play Storea, sillä sieltä voidaan saavuttaa suurin osa käyttäjäkunnasta. Kuitenkaan tällaisia jakelukanavia ei voida turvallisuusluokiteltua tietoa käsittelevän sovelluksen jakeluun hyödyntää, sillä muuten näihin sovelluksiin pääsevät myös ulkopuoliset tahot käsiksi. Tämän takia organisaatiolla kuuluu olla yhtenäinen ja turvallinen strategia yksityisten sovellusten jakamiselle.

Sovellusten pakettitiedostoja voidaan jakaa melkeinpä mitä tahansa kanavaa hyödyntäen, esimerkiksi vaikka TLS-sähköpostin tai verkkosivuston välityksellä. Kuitenkaan tällaiset jakelumuodot eivät välttämättä ole tehokkaita ja päivitysten tuominen voi olla haastavaa. Tämän johdosta tehokkaan ja turvallisen jakelun suorittamiseen organisaatiolla voi esimerkiksi olla oma tai luotettavan kolmannen osapuolen yksityinen sovelluskauppa, tai mahdollisesti sovellukset voidaan ladata ja päivittää päätelaitteelle laitteen etähallinnan kautta.

6 Pilvipalvelut ja turvallinen tiedonkäsittely pilvessä

Mobiiliratkaisua suunnitellessa voi pilvipalveluiden hyödyntäminen olla houkutteleva vaihtoehto pilvipalveluiden tuomien taloudellisten säästöjen ja helpon skaalautumisen ansiosta. Pilvipalveluiden hyödyntämisessä on kuitenkin ensin

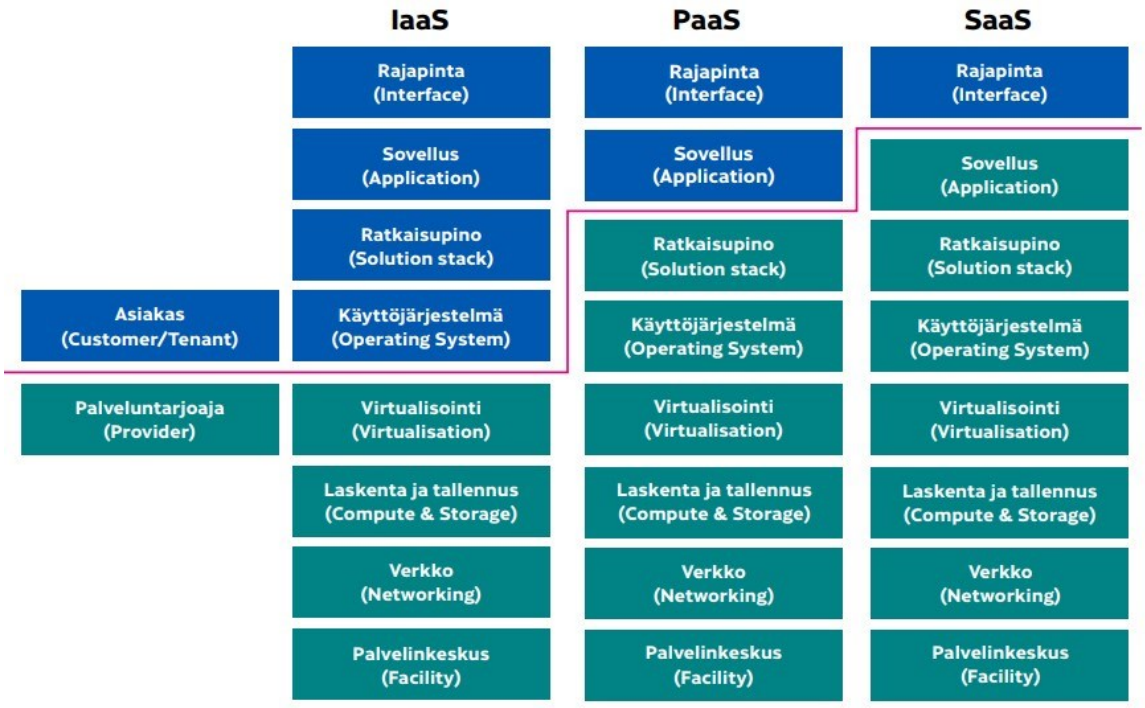
huomioitava pilvipalveluiden palveluntarjoajiin, tiedon säilytykseen ja turvallisuuteen kohdistuvat vaatimukset.

6.1 Pilvipalvelut ja vastuunjako

Pilvellä tarkoitetaan palvelimia ja niiden ohjelmistoja, joihin pääsee käsiksi verkon kautta. Pilvipalvelulla taas tarkoitetaan pilvessä toimivia resursseja, joita palveluntarjoajat tarjoavat [45]. Tällaisia pilvipalveluita voivat olla esimerkiksi infrastruktuuri, tallennustila, analytiikka, ohjelmistot ja kehitystyökalut [45]. Pilvipalvelut voidaan jakaa omiin palvelumalleihinsa ja tällaiset palvelumallit ovat:

- **Infrastructure-as-a-Service** (IaaS). IaaS tai suomeksi infrastruktuuri palveluna -palvelumalli tarkoittaa palvelua, jossa palveluntarjoaja tarjoaa asiakkaalle käyttöön palvelimen, jossa asiakas voi vapaasti hallita käyttöjärjestelmiä, saatua tallennustilaa ja ohjelmistoja. Asiakas ei kykene kuitenkaan hallinnoimaan taustalla toimivaa pilvi-infrastruktuuria. [46, s. 3.]
- **Platform-as-a-Service** (PaaS). PaaS tai suomeksi ohjelmistoalusta palveluna -palvelumalli tarkoittaa palvelua, jossa palveluntarjoaja tarjoaa asiakkaalle mahdollisuuden tuoda ohjelmistoja pilvi-infrastruktuuriin. [46, s. 2.]
- **Software-as-a-Service** (SaaS). SaaS tai suomeksi ohjelmisto palveluna -palvelumalli tarkoitettaa palveluntarjoajan pilvi-infrastruktuurissa olevaa ohjelmistoa, jota asiakas voi hyödyntää. Palveluntarjoaja tarjoaa tällaiseen palveluun käyttöliittymän tai rajapinnan. [46, s. 2.]

Palvelumallin mukaan vastuu turvallisuudesta jakautuu asiakkaan ja palveluntarjoajan välille. Vastuunjako voidaan havainnoida kuvasta 7. Tulee kuitenkin ottaa huomioon, että eri palveluntarjoajilla voi olla erilaiset näkemykset siitä, mitä he haluavat tarjota tietyissä palvelumalleissa, ja tämä otetaan huomioon pilvipalvelun turvallisuuden vastuunjaossa [8, s. 10].



Kuva 7. Pilvipalveluiden palveluntarjoajan ja asiakkaan välinen turvallisuuden vastuunjako palvelumalleittain [8, s. 10].

Palvelumallien lisäksi pilvipalveluiden toteutus määritellään eri toteutusmalleilla. Toteutusmalleja on useita, mutta PiTuKrin [8, s. 11] mukaan nämä toteutusmallit ovat arvioitavissa yleisimpien toteutusmallien pohjustamana. Yleisimmät pilvipalvelun toteutusmallit ovat:

- **Public cloud** tai julkinen pilvi. Julkisessa pilvessä pilvi on avoinna verkon kautta julkisesti kaikille, ja pilveä voi käyttää useampi taho. Infrastruktuuri sijaitsee julkisessa pilvessä palveluntarjoajan toimitiloissa. Yleisesti tunnettuja julkisia pilviä ovat esimerkiksi Amazon Web Services, Microsoft Azure ja Google Cloud Platform.
- **Private cloud** tai yksityinen pilvi. Yksityisessä pilvessä pilvi on tarkoitettu vain yhden organisaation sisäiseen käyttöön. Pilvi-infrastruktuuri voi sijaita organisaation toimitiloissa tai jossakin muualla, ja pilven voi omistaa joko organisaatio tai kolmas osapuoli.
- **Hybrid cloud** tai yhdistelmä pilvi. Yhdistelmä pilvessä organisaatiolla on käytössä julkinen ja yksityinen pilvi, jotka ovat yhdistetty yhdeksi infrastruktuuriksi. [46, s. 3.]

Turvallisuuden kannalta toteutusmallin valinnassa on huomattavia eroja. Yksityinen pilvi on aina turvallisin tapa toteuttaa pilvipalvelu, sillä yksityisessä pilvessä

koko pilvi on infrastruktuurista lähtien eroteltuna ulkopuolisista toimijoista vain yksityiseen käyttöön. Julkisessa pilvessä taas toimii myös ulkopuolisia toimijoita samassa pilvessä, tehden hyökkäyspinta-alasta laajemman. [8, s. 11.]

Jotta pilvipalvelun soveltuvuutta voidaan arvioida, tulee pilvipalvelulla olla selkeä järjestelmäkuvaus. Järjestelmäkuvaus on selkeä ja yksityiskohtainen kuvaus pilven infrastruktuurista, verkosta ja järjestelmäkomponenteista. Kuvauksessa määritellään myös pilvipalvelun vastuunjako (kuva 7), palvelu- ja toteutusmallit, muutostenhallinnan käytännöt ja periaatteet, kriittisten poikkeustilanteiden käsittelyprosessit sekä tarvittavat palvelutasosopimukset. Mikäli ulkoisille tai alihankkijoille on siirretty toimintoja, pitää tästä ilmoittaa järjestelmäkuvauksessa, jotta voidaan arvioida sitä, kuinka paljon toimintoja ulkoisille toimijoille on sijoitettu palveluntarjoajalta. Koska kuvauksessa kuuluu huomioida koko järjestelmä, pitää kuvaukseen lisätä turvatoimet, periaatteet, menettelyt ja valvontatoimet mitä käytetään siltä aikaväliltä, kun pilvipalvelua hyödynnetään. [8, s. 14.]

Selkeän järjestelmäkuvauksen lisäksi esiehtona jatkoarvioinnin suorittamiselle tulee pilvipalvelusta olla kuvattuna myös lainsäädäntöjohdannaiset riskit sekä velvoitteet. Lainsäädäntöjohdannaisten riskien kuvaukseen lisätään pilvessä käsiteltävän tiedon, komponenttien ja toimintojen fyysiset sijainnit koko tiedon elinkaaren ajalta. Mikäli ulkoisia toimijoita, sovellettavan lainsäädännön takia palveluun pääseviä toimijoita tai alihankkijoita on osana pilvipalvelua, pitää näistä kuvaukseen ilmoittaa, jotta ulkoisten toimijoiden luotettavuus voidaan arvioida. Pilvipalvelussa käsiteltäviin tiedon sekä käytön oikeuspaikka ja lainsäädäntö pitää myös kuvaukseen merkitä. Kuvauksen perusteella on varmistettava, että lainsäädäntöjohdannaiset riskit tai palveluntarjoajan sopimusehdot eivät ole este pilvipalvelun käyttöön käyttötapauksessa. On varmistettava, että tiedot sijaitsevat elinkaaren ajan vain sopimuksessa olevissa fyysisissä sijainneissa. [8, s. 15.]

6.2 Turvallisuusluokitetun tiedon käsittely

Kun turvallisuusluokiteltua tietoa käsitellään, kuuluu käsittelyssä olla erityisen tarkka, ja käsittelyssä on huomioitava lainsäädännölliset turvallisuuden velvoitteet kansallisen turvallisuuden takaamiseksi. Korkeamman turvallisuusluokitellun (TL III ja TL II) tiedon käsittelyssä tulee huomioida myös lisäsuojavaatimukset, mutta tässä luvussa ei oteta tarkemmin kantaa asiaan.

Turvallisuusluokitellun tiedon käsittelyyn pilvipalveluissa kohdistuu erilaisia rajoituksia riippuen tiedon turvallisuusluokasta. Kuitenkin jokaisen turvallisuusluokan tiedon käsittelyssä yhteisenä tekijänä on tiedon sekä pilvipalvelun fyysinen sijainti. Turvallisuusluokiteltua tietoa käsiteltäessä tulee pilvipalvelun sijaita fyysisesti Suomessa turva-alueella, tämä koskee myös kansainvälistä RESTRICTED-tason tietoa, kuitenkin ottaen huomioon tiedon kansainvälisen omistajan erityisvaatimukset. Pilvipalvelun tarjoajan tulee olla luotettava kansallinen toimija. Toimijan luotettavuus voidaan tarkistaa esimerkiksi yritysturvallisuusselvityksellä. Myöskään muiden valtioiden viranomaisilla ei saa olla pääsyä turvallisuusluokiteltuun tietoon tasosta riippumatta. [8, s. 16.]

TL IV -tason tiedon käsittelyssä on huomattavia eroja toteutusmallissa verraten korkeammin turvallisuusluokitellun tiedon käsittelyyn. TL IV -tason tietoa käsittelevässä pilvipalvelussa ei ole toteutusmallille asetettu varsinaista vaatimusta. Viimeisimmän valtiohallinnon pilvipalvelulinjauksen mukaan [47, s. 21–22], TL IV -tason tietoa käsitellessä voi pilvipalvelun toteuttaa myös julkisessa pilvessä, mikäli muut lainsäädännölliset velvoitteet täytetään. Korkean turvallisuusluokan omaavan tiedon käsittelyssä tulee pilvipalvelun toteutusmallin olla yksityinen tai yhteisöpilvi [8, s. 16].

Pilvipalveluissa voidaan käsitellä toteutettavan sovelluksen mukaan myös erittäin suurta määrää salassa pidettävää tai turvallisuusluokiteltua tietoa. Tällaisessa tilanteessa syntyy tiedosta kasauma. Kasaumalla käytännössä tarkoitetaan vain suurta määrää salassa pidettävää tai turvallisuusluokiteltua tietoa, mille annetaan korkeampi turvallisuusluokka suuren tietomäärän takia [8, s. 8].

Kasaumatilanteet tulee huomioida lainsäädännöllisiä velvoitteita toteuttaessa, sillä esimerkiksi jos käsitellään suurta määrää TL IV -tason tietoa, koostuu siitä TL III -tason kasauma ja tässä tilanteessa tulee toteuttaa TL III -tason kasauman velvoitteet, jotka rajaavat toteutusmallia [8, s. 16].

6.3 Henkilötietojen käsittely

Henkilötietojen käsittelyyn kohdistuu vähemmän lainsäädännöllisiä rajoituksia kuin esimerkiksi turvallisuusluokiteltuun tietoon, kuitenkin henkilötietojen käsittelyssä on huomioitava EU:n tietosuoja-asetus ja kansallinen tietosuojalaki. Nämä lainsäädännöt on asetettu yleisesti henkilötiedoille, joten niiden vaatimukset tulee toteuttaa myös pilvipalvelussa, mikäli pilvi tulee käsittelemään henkilötietoja.

Henkilötietoja käsittelevään pilvipalvelun toteutusmalliin ei kohdistu rajoituksia, niin kauan, kun tietosuoja, tietoturva ja jatkuvuudenhallinnan velvoitteet toteutuvat. Esimerkiksi julkisen pilvipalvelun hyödyntäminen on sallittua henkilötietoja käsittelevässä pilvipalvelussa. Myös palveluntarjoajiin ei kohdistu rajoitteita, mikäli riskejä arvioitaessa säilytettävien tietojen luonteen perusteella ei ilmene rajoittavia tekijöitä. [8, s.16; 47.]

Tietosuojavelvoitteiden toteutumiseksi pilven sekä tiedon fyysisellä sijainnilla on merkitystä. Sijainnin pitää olla tietosuojasäädösten sallimilla maantieteellisillä alueilla, pääsääntöisesti henkilötietojen säilytys on rajoitettu EU/ETA-alueelle [8, s. 16]. Henkilötietoja ei saa siirtää tämän alueen ulkopuolelle tai kansainväliselle organisaatiolle, jollei EU:n komissio ole todennut alueella tai organisaatiossa suoritettavaa suojausta riittäväksi esimerkiksi vastaavuuspäätöksellä [47, s. 20]. Alueen ulkopuolelle tapahtuvaan siirtoon tarvitaan myös riittävä siirtoperuste [47, s. 20]. Siirrolla tarkoitetaan myös alueen ulkopuolelta etäyhteydellä tapahtuvaa henkilötietojen käsittelyä [47, s. 21].

Poikkeuksellisesti EU:n komissio on päättänyt vastaavuuspäätöksessään Yhdysvaltojen tietosuojan olevan riittävä henkilötietojen käsittelyyn. Tämä käytännössä tarkoittaa sitä, että henkilötietoja voidaan siirtää vastaavuuspäätöksen

siirtoperusteena Yhdysvaltalaisille yrityksille, jotka ovat sitoutuneet Yhdysvaltain ja EU:n väliseen tietosuojakehykseen. Tämän ja muidenkin vastaavuuspäätöksien aktiivinen seuranta on suotavaa varmistaakseen tietosuoja-asetuksen velvoitteiden toteutumisen. [47, s. 21.]

Kuten muissakin tietotyypeissä, tulee myös henkilötiedoissa ottaa huomioon mahdolliset kasaumatilanteet. Pilvipalveluissa käsiteltävän henkilötiedon määrä voi olla suuri, ja tällöin suuresta määrästä henkilötietoja muodostuu TL IV -kasauma. Tällainen tilanne asettaa pilvipalvelulle lisärajoitteita. Esimerkiksi TL IV -kasauma tiedon käsittelyssä pilven fyysinen sijainti rajoittuu ainoastaan Suomeen, ja palveluntarjoajana saa toimia tällöin ainoastaan luotettava kansallinen toimija. Myöskään minkään muun valtion viranomaisella ei TL IV -kasaumatietoon saa olla suoraa tai epäsuoraa pääsyä. [8, s. 16.]

7 Yhteenveto

Insinöörityössä perehdyttiin siihen, miten lainsäädännölliset sekä viranomaisten asettamat tietoturvallisuuden velvoitteet voidaan toteuttaa viranomaisen hyväksyntään pyrkivälle mobiiliratkaisulle. Tarkemmin asiaa tutkittiin pohjautuen kansallisten viranomaisten asettamiin turvallisuuskriteeristöihin, ottaen huomioon kuitenkin lainsäädännölliset lähteet kansallisesta sekä kansainvälisestä lainsäädännöstä.

Työssä selvisi, että tietoturvallisuuden vaatimuksia on paljon, ja että ne vaihtelevat huomattavasti riippuen sovelluksen toteutustavasta. Näitä vaatimuksia tutkittiin laajalla ja yleisellä tasolla. Työtä voisi jatkaa toteuttamalla jonkin tietyn sovelluksen esimerkiksi alustariippumattomalla toteutustavalla, ja täten pureutua tarkemmin tietyn osa-alueen teknisiin mobiilisovelluksen suojauksiin tuoden enemmän askeleittain kulkeutuvan ohjeistuksen. Työn tarkoitus oli tutkia asiaa koko mobiiliratkaisun näkökulmasta jättäen toteutuksen tavan työn lukijalle, ja siinä työssä onnistuttiin.

Työssä tutkitut tietoturvallisuuden vaatimukset ovat kuitenkin vain tämänhetkisiä. Lainsäädäntö ja lainsäädäntöön pohjautuvat viranomaisten luomat käytännön kriteeristöt muovautuvat ajan kuluessa, joten on tärkeää ajantasaisuuden kannalta seurata niissä tapahtuvia muutoksia sekä uudistuksia. Myös insinöörityötä voisi jatkaa päivittämällä työhön näitä lainsäädännöllisiä ja kriteeristöihin tehtäviä muutoksia.

Lähteet

1 Suositus turvallisuusluokiteltavien asiakirjojen käsittelystä. 2021. Valtiovarainministeriön julkaisuja 2021:5. Helsinki: Valtiovarainministeriö.

2 Laki julkisen hallinnon tiedonhallinnasta. 2019. 906/9.8.2019.

3 Valtioneuvoston asetus asiakirjojen turvallisuusluokittelusta valtionhallinnossa. 2019. 1101/28.11.2019.

4 Kansainvälisen turvallisuusluokitellun tietoaineiston käsittelyohje. 2020. Verkkoaineisto. Ulkoministeriö. <https://um.fi/documents/35732/0/KV-TIEDON+K%C3%84SITTELY-OHJE+NSA+%28SUOMI%29_CLEAN.pdf/888aeb84-3e23-df3b-1774-a936f29a3fd9?t=1604931832075/>. Luettu 24.11.2023.

5 Katakri 2020: Tietoturvallisuuden auditointityökalu viranomaisille. 2020. Traficomin julkaisusarja 232/2020. Helsinki: Liikenne- ja viestintävirasto Traficom.

6 Yritysturvallisuusselvityksen laatiminen. Verkkoaineisto. Suojelupoliisi. <https://supo.fi/yritysturvallisuusselvityksen-laatiminen/>. Luettu 27.11.2023.

7 Kansallinen turvallisuusauditointikriteeristö on julkaistu. 2020. Verkkoaineisto. Valtioneuvosto. <https://valtioneuvosto.fi/-/kansallinen-turvallisuus-auditointikriteeristo-katakri-2020-julkaistu/>. 18.12.2020. Luettu 30.11.2023.

8 Pilvipalveluiden turvallisuuden arviointikriteeristö (PiTuKri). 2020. Traficomin julkaisuja 13/2020. Helsinki: Liikenne- ja viestintävirasto Traficom.

9 Peltonen, Leeni. 2020. PiTuKri auttaa pilvipalvelun valinnassa. Verkkoaineisto. Erillisverkot. <https://www.erillisverkot.fi/pitukri-auttaa-pilvipalvelun-valinnassa/>. 21.2.2020. Luettu 24.11.2023.

10 Julkisen hallinnon tietoturvallisuuden arviointikriteeristö (Julkri): Suositus ja kriteeristö. 2023. Valtiovarainministeriön julkaisuja 2023:46. Helsinki: Valtiovarainministeriö.

11 About the OWASP foundation. Verkkoaineisto. OWASP. <https://owasp.org/about/>. Luettu 12.12.2023.

12 ASVS 4.0.3. 2021. OWASP Application Security Verification Standard (ASVS). OWASP.

13 MASVS 2.0.0. 2023. OWASP Mobile Application Security Verification Standard (MASVS). OWASP MAS.

14 VAHTI: Sovelluskehityksen tietoturva ohje. 2013. VAHTI 1/2013. Helsinki: Valtiovarainministeriö.

15 VAHTI-ohjeet. Verkkoaineisto. Suomidigi. <https://www.suomidigi.fi/ohjeet-ja-tuki/vahti-ohjeet/>. Päivitetty 8.8.2023. Luettu 12.12.2023.

16 Jurvanen, Lauri. Mitä kyberturvallisuuskeskus tekee? Verkkoaineisto. Savelan. <https://www.savelan.fi/mita-kyberturvallisuuskeskus-tekee/>. Luettu 2.2.2024.

17 Turvallinen tuotekehitys: kohti hyväksyntää. 2018. Traficomin julkaisuja 003/2018. Helsinki: Liikenne- ja viestintävirasto Traficom.

18 Vuorela, Antti. 2016. Mobiilin uutispalvelun teknologiastrategia: Web- natiivi- hybriditeknologioiden vertailu mobiilin uutispalvelun vaatimusten näkökulmasta. Diplomityö. Aalto-yliopisto. Perustieteiden korkeakoulu. Aalto-doc-tietokanta.

19 What’s the Difference Between Web Apps, Native Apps, and Hybrid Apps? Verkkoaineisto. Amazon Web Services. <https://aws.amazon.com/compare/the-difference-between-web-apps-native-apps-and-hybrid-apps/>. Luettu 14.12.2023.

20 Application Sandbox. 2023. Verkkoaineisto. Android Source. <https://source.android.com/docs/security/app-sandbox/>. Päivitetty 23.8.2023. Luettu 15.12.2023.

21 Application fundamentals. 2023. Android Developers. <https://developer.android.com/guide/components/fundamentals/>. Päivitetty 10.10.2023. Luettu 15.12.2023.

22 Griffith, Chris. What is Hybrid Mobile App Development? Verkkoaineisto. Ionic. <https://ionic.io/resources/articles/what-is-hybrid-app-development/>. Luettu 14.12.2023.

23 What is progressive web app? 2023. Verkkoaineisto. MDN Web Docs. <https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Guides/What_is_a_progressive_web_app/>. Päivitetty 4.7.2023. Luettu 14.12.2023.

24 Neuvoston päätös EU:n turvallisuusluokiteltujen tietojen suojaamista koskevista turvallisuussäännöistä. 2013. Päätös 2013/488/EU. Verkkoaineisto. Euroopan unionin virallinen lehti 23.9.2013. <https://eur-

lex.europa.eu/legal-content/FI/TXT/?uri=CELEX%3A32013D0488>. Luettu 22.12.2023.

25 Foster, Stuart. 2022. What is DISA STIG Overview + STIG Security. Verkkoaineisto. Perforce. <https://www.perforce.com/blog/kw/what-is-DISA-STIG/>. 19.9.2022. Luettu 22.12.2023.

26 Checklist repository. Verkkoaineisto. National Institute of Standards and Technology (NIST). <https://ncp.nist.gov/repository?sortBy=modifiedDate%7Cdesc>. Luettu 22.12.2023.

27 Päätelaitteiden tietoturvaohje. 2013. VAHTI 5/2013. Helsinki: Valtiovarainministeriö.

28 Bittium Tough Mobile 2 C. Verkkoaineisto. Bittium. <https://www.bittium.com/defense-security/bittium-tough-mobile-2-c/>. Luettu 25.12.2023.

29 Knox certifications and guidance. Verkkoaineisto. Samsung Knox. <https://www.samsungknox.com/en/knox-platform/knox-certifications/>. Luettu 25.12.2023.

30 Commission announces next steps on cybersecurity of 5G networks in complement to latest progress report by Member States. 2023. Verkkoaineisto. Euroopan komissio. <https://ec.europa.eu/commission/presscorner/detail/en/ip_23_3309/>. Päivitetty 15.6.2023. Luettu 26.12.2023.

31 Suomeen kohdistuu tiedustelua ja vaikuttamista. Verkkoaineisto. Suojelupoliisi. <https://supo.fi/tiedustelu-ja-vaikuttaminen/>. Luettu 26.12.2023.

32 Turvallisen sovelluskehityksen käsikirja. 2020. Verkkoaineisto. Digi- ja väestötietovirasto. <https://www.suomidigi.fi/sites/default/files/2020-05/Turvallisen%20sovelluskehityksen%20k%C3%A4sikirja.pdf>. 19.5.2020. Luettu 11.1.2024.

33 Kesäniemi, Ari. Mobile Application Threat Analysis. Verkkoaineisto. OWASP. <https://owasp.org/www-pdf-archive/Mobile-threat-analysis-short-presentation_owasp.pdf>. Luettu 11.1.2024.

34 Geib, Jeremy; Kess, Barbara; Berry, David; Santos, Brittany & Baldwin M. 2020. Microsoft Threat Modeling Tool Threats. Verkkoaineisto. Microsoft Learn. <https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats/>. 25.8.2020. Luettu 11.1.2024.

35 What is REST API? 2020. Verkkoaineisto. Red Hat. <https://www.redhat.com/en/topics/api/what-is-a-rest-api/>. 8.5.2020. Luettu 20.1.2024.

36 SSRF: What every beginner pentester should now. 2023. Verkkoaineisto. Medium. <https://medium.com/@TheCS_student/ssrf-what-every-beginner-pentester-should-know-ba94a80b8457/>. 15.1.2023. Luettu 2.2.2024.

37 Grassi, Paul; Fenton, James; Newton, Elaine; Perlner, Ray; Regenscheid, Andrew; Burr, William; Richer, Justin; Lefkovitz, Naomi; Danker, Jamie; Choong, Yie-Yin; Greene, Kristen & Theofanos, Mary. 2017. Digital Identity Guidelines: Authentication and Lifecycle Management. NIST Special Publication 800-63b. Gaithersburg: National Institute of Standards and Technology.

38 How does SSL work? Verkkoaineisto. Cloudflare. <https://www.cloudflare.com/learning/ssl/how-does-ssl-work/>. Luettu 10.1.2024.

39 What is an SSL/TLS certificate? Verkkoaineisto. Amazon Web Services. <https://aws.amazon.com/what-is/ssl-certificate/>. Luettu 10.1.2024.

40 Singh, Prabhnit. 2018. Opitimizing Web Performance with TLS 1.3. Verkkoaineisto. Thousand Eyes. <https://www.thousandeyes.com/blog/optimizing-web-performance-tls-1-3/>. 28.11.2018. Luettu 10.1.2024.

41 Dasgupta, Rono. Securing Mobile Application with Cert Pinning. Verkkoaineisto. DZone. <https://dzone.com/refcardz/securing-mobile-applications-with-cert-pinning/>. Luettu 10.1.2024.

42 Mobile App Authentication Architectures. Verkkoaineisto. OWASP Mobile Application Security. <https://mas.owasp.org/MASTG/General/0x04e-Testing-Authentication-and-Session-Management/>. Luettu 14.1.2024.

43 Kryptografiset vahvuusvaatimukset luottamuksellisuuden suojaamiseen - kansalliset turvallisuusluokat. 2022. Verkkoaineisto. Liikenne- ja viestintävirasto Traficom. <https://www.kyberturvallisuuskeskus.fi/sites/default/files/media/file/ohje-kryptografiset-vahvuusvaatimukset-kansalliset-suojaustasot.pdf>. 26.10.2022. Luettu 2.2.2024.

44 Data and file storage overview. Verkkoaineisto. Android Developers. <https://developer.android.com/training/data-storage/>. Luettu 15.1.2024.

45 What is the cloud? Verkkoaineisto. Cloudflare. <https://www.cloudflare.com/learning/cloud/what-is-the-cloud/>. Luettu 20.12.2023.

46 Mell, Peter & Grance, Timothy. 2011. The NIST Definition of Cloud Computing. NIST Special Publication 800-145. Gaithersburg: National Institute of Standards and Technology.

47 Valtiohallinnon pilvipalvelulinjaukset. 2023. Valtiovarainministeriön julkaisuja 2023:75. Helsinki: Valtiovarainministeriö.