Santeri Nurminen

MIKROPALVELUARKKITEHTUURIN TIETOTUR-VAUHAT

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2021

TIIVISTELMÄ

Nurminen, Santeri
Mikropalveluarkkitehtuurin tietoturvauhat
Jyväskylä: Jyväskylän yliopisto, 2021, 33 s.
Tietojärjestelmätiede, Kandidaatintutkielma
Ohjaaja: Marttiin, Pentti

Mikropalveluarkkitehtuuri on ohjelmistoarkkitehtuurin suuntaus, jossa ohjelmisto jaetaan pieniin, itsenäisesti suoritettaviin palveluihin. Mikropalveluarkkitehtuurien hyödyistä, kuten skaalautuvuudesta, huolimatta arkkitehtuuri lisää myös hyökkäysrajapintaa tietoturvauhille. Tässä kandidaatintutkielmassa perehdytään näihin tietoturvauhkiin kuvailevan kirjallisuuskatsauksen menetelmin. Tutkielmassa löydettiin monia yleisesti arkkitehtuuriin päteviä tietoturvauhkia, joita jaoteltiin mikropalveluarkkitehtuurista muodostettujen eri kerrosten mukaan. Havaitut tietoturvauhat olivat pääsääntöisesti vakavia ja monessa tapauksessa saattoivat johtaa koko järjestelmän hallinnan menettämiseen. Tutkielman tuloksena pystyttiin muodostamaan yleinen katsaus mikropalveluarkkitehtuurin tietoturvauhista sekä näiden mahdollisista vaikutuksista.

Asiasanat: mikropalveluarkkitehtuuri, mikropalvelut, järjestelmäarkkitehtuuri, tietoturva, tietoturvauhka

ABSTRACT

Nurminen, Santeri
Microservice architecture's information security threats
Jyväskylä: University of Jyväskylä, 2021, 33 pp.
Information Systems, Bachelor's Thesis
Supervisor: Marttiin, Pentti

Microservice architecture is a systems architecture style where the system is divided into small, independent services. In spite of the advantages of using this architecture, such as scalability, it also provides an increased attack surface for information security threats. In this bachelor's thesis these threats are analyzed through a descriptive literature review. In the review many general security threats affecting the architecture were observed, which were classified into different levels of the architecture found earlier in the thesis. The observed threats were in general severe and in many cases could cause loss of control of the entire system. As a result of the literature review, an universally applicable overview of microservice architecture's information security threats was established.

Keywords: microservice architecture, microservices, systems architecture, information security, information security threat

KUVIOT

TAULUKOT

SISÄLLYS

1 JOHDANTO

Mikropalvelut ovat yhden vastuun omistavia ohjelmiston palasia, mitä voidaan ajaa, skaalata ja testata itsenäisesti riippumattomana muusta järjestelmästä (Larrucea, Santamaria, Colomo-Palacios & Ebert, 2018). Monoliittisen järjestelmätoteutuksen vastakohtana mikropalvelujen hyödyntäminen edesauttaa organisaatioita saattamaan tuotteensa markkinoille nopeammin (Larrucea ym., 2018) sekä mahdollistaa muun muassa DevOps -käytänteiden hyödyntämisen helpottumisen ja kehittäjätiimin paremman organisoinnin palveluiden ympärille (Balalaie, Heydarnoori & Jamshidi, 2016). Järjestelmän näkökulmasta mikropalveluarkkitehtuurin suurimpana hyötynä on mahdollisuus skaalata järjestelmää helposti vastaamaan käyttäjien määrän tarpeisiin (Dragoni ym., 2018). Vastatakseen kasvavaan käyttäjämäärään, monet suuret palveluntarjoajat, kuten Netflix, Amazon ja Uber ovat ottaneet käyttöönsä mikropalveluarkkitehtuurin (Hassan, Ali & Bahsoon, 2017).

Mikropalveluiden ja mikropalveluarkkitehtuurin hyödyntäminen ohjelmistokehityksessä voi kuitenkin koitua ongelmalliseksi tietoturvan näkökulmasta. Monoliittiseen toteutukseen verrattuna mikropalveluita hyödyntävä ohjelmisto lisää pinta-alaa tietoturvahyökkäyksille riippumatta yksittäisten palveluiden tietoturvan tasosta (Pereira-Vale, Fernandez, Astudillo & Márquez, 2021). Vuonna 2020 tietoturvamurron keskimääräinen hinta toimijalle on ollut 3,86 miljoonaa dollaria (IBM, 2020). Täten, siirryttäessä käyttämään mikropalveluarkkitehtuuria, on oleellista tiedostaa arkkitehtuurin hyödyntämisestä koituvat tietoturvauhat, jotta niiltä pystytään suojautumaan.

Tässä työssä tutkitaan mikropalveluarkkitehtuurin tietoturvauhkia kuvailevan kirjallisuuskatsauksen menetelmin. Tutkielmassa tarkastellaan kirjallisuudessa esiintyneitä yleisimpiä tietoturvauhkia, jotka voidaan yhdistää mikropalveluarkkitehtuuriin. Tämän lisäksi tarkastelun kohteena ovat arkkitehtuurin toteutuksessa tarvittavista menetelmistä nousevat tietoturvauhat. Tarkoituksena on muodostaa kokoava yleiskatsaus sekä antaa esimerkkejä mikropalveluarkkitehtuurin eri kerroksia koskevista tietoturvauhista. Tutkielman tutkimuskysymyksenä toimii ”*Mitä tietoturvauhkia mikropalveluarkkitehtuurin hyödyntäminen sisältää tai mahdollistaa?*”.

Tutkielmassa kerätään aineistoa Jyväskylän Yliopiston tarjoamista tietokannoista, kuten IEEE Xplore, minkä lisäksi hyödynnetään Google Scholar -tietokantaa. Lähdekirjallisuudeksi pyritään valitsemaan vertaisarvioitua, JUFO-luokituksen omistavaa kirjallisuutta lähteiden laadun takaamiseksi.

Tutkielmassa käsitellään mikropalveluarkkitehtuuria yleiseltä tasolta, ottamatta kantaa esimerkiksi teknologiavalintoihin. Tällä tavalla menettelemällä pystytään kokoamaan yleispätevä katsaus tietoturvauhkiin, mikä ei ole sidosteinen tiettyyn teknologiaratkaisuun. Tutkimuksen ulkopuolelle rajataan kyberfyysiset järjestelmät, joihin liittyy suuri määrä omia tietoturvaongelmiaan. Tämän lisäksi rajauksia tehdään muun muassa tietoturvan käsitettä koskien, joita käydään tarkemmin läpi kappaleessa kolme.

Tutkielma koostuu viidestä kappaleesta. Johdantokappaleen jälkeen, kappaleessa kaksi, käydään läpi mikropalveluarkkitehtuurin määritelmä sekä rakenne yleisellä tasolla. Kappaleella pyritään muodostamaan lukijalle selkeä kuva siitä, mitä mikropalveluarkkitehtuurilla tarkoitetaan. Tämän jälkeen, kappaleessa kolme, käsitellään tietoturvan käsitettä. Kappaleessa rajataan tietoturva ja tietoturvauhat tutkielman kontekstiin, jonka jälkeen käydään läpi erilaisia tapoja luokitella uhkia. Kappaleessa muodostetaan lopuksi tutkielmassa käytettävä tietoturvauhkien luokittelukehys hyödyntämällä kirjallisuudesta löydettyjä viitekehyksiä. Tämän jälkeen, kappaleessa neljä, tarkastellaan mikropalveluarkkitehtuurin tietoturvauhkia. Uhkia käsitellään kappaleessa kaksi muodostettujen kerrosten järjestyksessä. Kappaleen lopuksi muodostetaan yhteenvetona taulukko kaikista uhista sekä käydään tiivistävää pohdintaa tuloksista. Viidennessä kappaleessa käydään läpi oleellisia huomioita tutkimuksesta, analysoidaan tulosten luotettavuutta sekä pohditaan jatkotutkimusaiheita.

2 MIKROPALVELUARKKITEHTUURI

Tässä luvussa esitellään mikropalveluarkkitehtuurin käsite sekä rakenne. Ensin annetaan lyhyt kuvaus monoliittisesta arkkitehtuurista, millä pyritään edesauttamaan mikropalveluarkkitehtuurin hahmottamista vastakohdan kautta. Tämän jälkeen esitellään mikropalveluarkkitehtuurin sekä mikropalveluiden määritelmät ja arkkitehtuurin yleinen rakenne. Koska mikropalveluarkkitehtuurilla ei ole tiettyä vakiintunutta toteutustapaa, esitellään arkkitehtuurin toteutusta ohjaavia vaihtoehtoisia toimintatapoja ja malleja eri näkökulmista. Kappaleen tarkoituksena on muodostaa yleiskatsaus mikropalveluarkkitehtuuriin sekä sen rakenteeseen, jotta tietoturvaa tarkasteltaessa pystytään hahmottamaan arkkitehtuuri kokonaisuutena.

2.1 Monoliittinen arkkitehtuuri

Mikropalveluarkkitehtuuria voidaan kuvailla monoliittisen arkkitehtuurin vastakohtana. Stephens (2015) kuvailee monoliittista arkkitehtuuria järjestelmäarkkitehtuuriksi, missä yksi ohjelma tekee kaikki ohjelmiston operaatiot, kuten näyttää käyttöliittymän ja käsittelee informaatiota.

Monoliittiseen arkkitehtuuriin liittyy ohjelmiston toiminnan kannalta tiettyjä rajoituksia, joita on pyritty ratkaisemaan käyttämällä erilaisia vaihtoehtoisia ohjelmistoarkkitehtuureja. Näiden rajoitusten tiedostaminen on hyödyllistä, kun pyritään hahmottamaan mikropalveluarkkitehtuurin käyttämisen hyötyjä sekä perusteluja. Näistä rajoituksista suurin on järjestelmän joustavuuden puute (Stephens, 2015). Vaikka Stephens (2015) viittaa joustavuudella lähinnä ohjelmiston toteutuksen (ohjelmoinnin) helpottumiseen, monoliittisen arkkitehtuurin kontekstissa joustamattomuus näkyy myös ohjelmiston suorituksessa. Monoliittinen arkkitehtuuri nojaa abstraktiokerroksissa resurssien jakamiseen samasta suorituslaitteesta, joten ohjelmiston moduuleja ei voida suorittaa itsenäisesti (Bucchiarone, Dragoni, Dustdar, Larsen & Mazzara, 2018). Tästä puolestaan seuraa ohjelmiston käyttämien resurssien skaalaamisen vaikeutuminen

sekä muutosten tekemisen vaikeudet (Bucchiarone ym., 2018). Toisaalta monoliittisen arkkitehtuurin valitseminen sisältää myös mahdollisia hyötyjä: esimerkiksi tarve tietoverkkojen yli tapahtuvaan kommunikaatioon ohjelmiston moduulien välillä poistuu sekä arkkitehtuuri voi osoittautua sopivammaksi valinnaksi pienille projekteille (Stephens, 2015).

Voidaan huomata, että monoliittinen arkkitehtuuri ei sovellu tilanteisiin, joissa ohjelmistolta vaaditaan joko kykyä skaalautua käytön mukaan, tai joustavuutta muutokseen. Internetin käytön lisäännyttyä huomattavasti viime vuosina esimerkiksi mobiililaitteiden yleistyttyä, nykypäivän ohjelmistot vaativat skaalautuvuutta käyttäjämäärien kasvaessa. Mikropalveluarkkitehtuuri on noussut viime aikoina vaihtoehdoksi ratkaisemaan muun muassa edellä mainittuja haasteita.

2.2 Mikropalveluarkkitehtuurin määritelmä

Mikropalveluarkkitehtuurilla ei ole tieteellisessä keskustelussa yksimielistä tarkkaa määritelmää. Dragoni, ym. (2017) määrittelevät mikropalveluarkkitehtuurin ”hajautetuksi sovellukseksi, missä kaikki sen moduulit ovat mikropalveluja”. Mikropalvelun he puolestaan määrittelevät seuraavasti: ”koheesioinen, itsenäinen prosessi, joka kommunikoi viestien avulla”. (Dragoni ym., 2017)

Mikropalveluarkkitehtuuria on kuvailtu myös monoliitin hajotelmaksi (Larrucea ym., 2018). Vaikka tarkka määritelmä voi vaihdella yksityiskohdiltaan lähteen mukaan, useissa tutkimuksissa mikropalveluarkkitehtuurin kuvaus mukailee Dragonin ym. (2018) esittämää kolmea perusperiaatetta arkkitehtuurin rakenteesta:

- Sidottu konteksti, eli liiketoimintalogiikan komponentteja toteuttaa yksi mikropalvelu.
- Mikropalvelut pyritään toteuttamaan mahdollisimman pienikokoisina.
- Mikropalvelut toteutetaan itsenäisinä palasinaan eristettynä toisista palveluista. Palveluiden välinen kommunikointi tapahtuu rajapintojen kautta. (Dragoni ym., 2018)

Näitä perusperiaatteita toteuttamalla mikropalveluarkkitehtuuri mahdollistaa varteenotettavia hyötyjä, kuten skaalattavuuden ja ylläpidon helpottumista (Dragoni ym., 2018), nopeampaa ohjelmiston markkinoille saattoa (Larrucea ym., 2018), sekä mahdollisuutta parempaan kehitystiimin jäsentelyyn (Balalaie ym., 2016).

2.3 Mikropalveluarkkitehtuurin rakenne

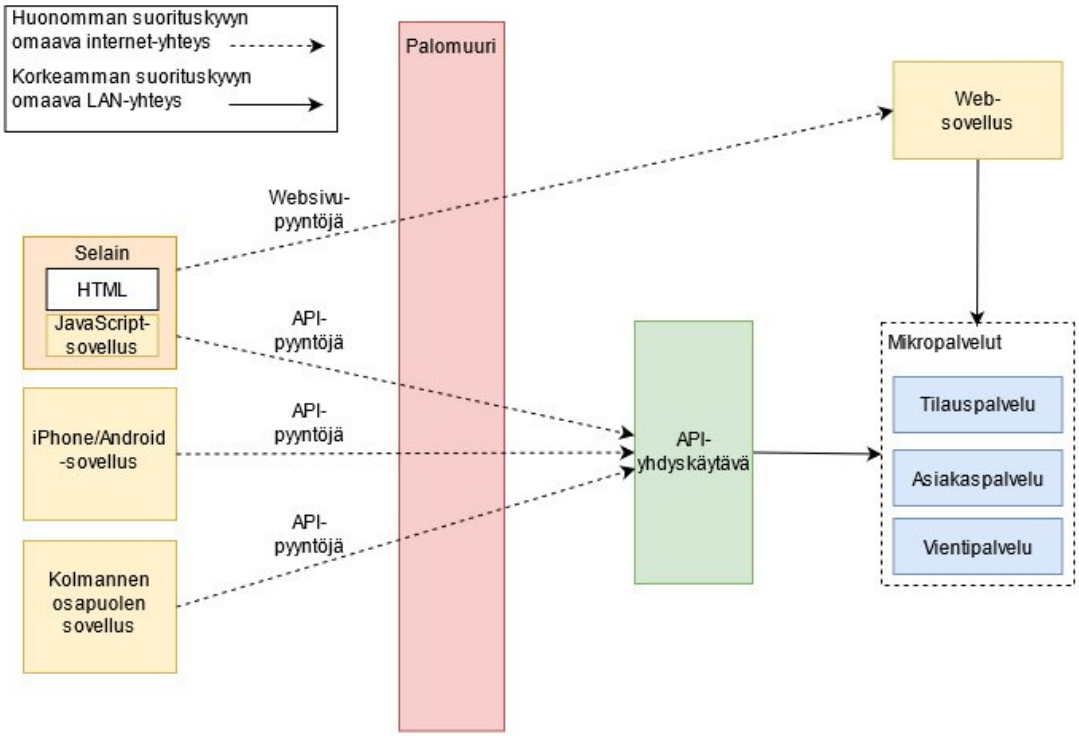
Mikropalveluarkkitehtuuria toteuttamalla pyritään siis hajauttamaan monimutkainen monoliitti pienemmiksi, helposti hallittaviksi osiksi. Vaikka mikropalveluarkkitehtuurin määritelmä ei itsessään ota kantaa arkkitehtuuria toteuttavaan rakenteeseen, ovat Taibi, Lenarduzzi ja Pahl (2018) löytäneet kirjallisuuskatsauksessaan arkkitehtuurin toteutukseen liittyviä yleisesti toistuvia toimintatapoja. Näitä toimintatapoja on jaoteltu kolmeen eri ryhmään, joita ovat orkestraatio ja koordinaatio (orchestration & coordination), käyttöönotto (deployment) sekä datan varastointi (data storage) (Taibi ym., 2018).

2.3.1 Orkestraatio ja koordinaatio

Orkestraatioon sekä koordinaatioon liittyvillä arkkitehtuurin osilla toteutetaan mikropalveluiden kommunikaatiota. Koska mikropalvelut kommunikoivat pääsääntöisesti julkisten rajapintojen kautta (Dragoni ym., 2017), orkestraatio ja koordinaatio on erittäin relevantti arkkitehtuurin osa tietoturvan näkökulmasta. Mikropalveluarkkitehtuurin tarvitsee käsitellä karkeasti kahdentyyppistä kommunikaatiota: ulkoapäin palveluille pyyntöjä tekevää kommunikaatiota sekä sisäistä, palveluiden välistä kommunikaatiota.

Koska mikropalvelut toteutetaan itsenäisinä palasinaan (Dragoni ym., 2018), tulee järjestelmään yhteyttä ottavalle toimijalle tarjota abstraktiorajapinta tukemaan kommunikaatiota järjestelmän kanssa. Ulkoapäin järjestelmään saapuvia pyyntöjä käsittelemään Taibi, ym. (2018) suosittelevat API-yhdyskäytävää (API-gateway). He määrittelevät API-yhdyskäytävän järjestelmän pääsypisteeksi (entry point), minkä kautta järjestelmään saapuva kutsu reititetään joko oikealle mikropalvelulle tai monille palveluille, joiden vastaukset yhdyskäytävä kokoaa yhteen (Taibi ym., 2018).

API-yhdyskäytävä mahdollistaa näin ollen palvelun ulkopuolisen toimijan kommunikaation mikropalveluille. Yhdyskäytävää hyödyntämällä estetään ulkopuolisten toimijoiden suora kommunikaatio mikropalveluille, millä saavutetaan hyötyjä molemmille osapuolille. Ulkopuolisen toimijan ei tarvitse tietää, mikä mikropalvelu vastaa mistäkin palvelun logiikan osa-alueesta, joten palveluun on kokonaisuudessaan helpompaa integroitua pienemmän rajapintamäärän takia. Palveluntarjoajan ei tarvitse puolestaan paljastaa järjestelmänsä arkkitehtuuria muille toimijoille, millä voidaan vähentää tietoturvan näkökulmasta hyökkäyspinta-alaa. Seuraavassa kuviossa (kuvio 1) havainnollistetaan API-yhdyskäytävän tekemää ulkoa tulevien kutsujen välitystä palveluille.



KUVIO 1 API-yhdyskäytävä (Richardson, 2019, s. 260) (suomennettu, piirretty uudelleen)

Yksi mikropalveluarkkitehtuurin suurimmista vahvuusalueista on skaalautuvuus. Skaalautuvuudella käytännössä tarkoitetaan, että samasta mikropalvelusta tarjotaan tarpeen mukaan monta rinnakkaista instanssia samanaikaisesti (Dragoni ym., 2018). Rinnakkaisuuden johdosta tarvitaan tapa, jolla tunnistetaan saatavilla oleva mikropalvelun instanssi ja välitetään kommunikaatio tälle instanssille. Taibi, ym. (2018) ovat tunnistaneet kaksi yleisesti käytössä olevaa tapaa tämän sisäisen kommunikaation toteuttamiselle: asiakaspuolen löytämismallin (client-side discovery pattern) sekä palvelinpuolen löytämismallin (server-side discovery pattern).

Molempia malleja yhdistää palvelurekisteri, mikä yksinkertaisesti esitettynä on tietokanta palveluista, palveluiden instansseista sekä niiden sijainneista verkossa (Richardson, 2019, s. 81). Perusperiaatteena on lähettää kysely asiakkaalta (tässä tapauksessa API-yhdyskäytävältä) palvelurekisteriin, mistä selviää tarjolla olevat mikropalvelujen instanssit sekä näiden osoitteet (Richardson, 2019, s. 81). Tämän jälkeen kuormituksen tasaaja ohjaa kyselyn valitsemalleen mikropalvelulle. Mallien välinen ero tulee esiin mikropalveluille suoraan lähtevän kommunikoinnin sijainnista: kommunikointi voi lähteä joko suoraan asiakkaalta tai erilliseltä, palvelimella sijaitsevalta reitittimeltä (Taibi ym., 2018).

Asiakaspuolen löytämismallissa asiakas on tietoinen palvelurekisterin olemassaolosta. Asiakas tällöin pyytää suoraan palvelurekisteriltä saatavilla olevat palvelut, minkä jälkeen asiakkaaseen sisäänrakennettu kuormantasaaja välittää kutsun suoraan halutulle mikropalvelulle (Richardson, 2019, s. 82). Palvelinpuolen löytämismallissa puolestaan asiakas välittää kutsunsa erilliselle

reitittimelle, joka kommunikoi palvelurekisterin kanssa sekä tasapainottaa kuormaa erillään asiakkaasta (Richardson, 2019, s. 84).

2.3.2 Käyttöönotto

Käyttöönotolla tarkoitetaan tässä kontekstissa tapaa tai tapoja, joilla mikropalveluja konkreettisesti suoritetaan esimerkiksi pilvipalvelussa. Mikropalveluiden käyttöönotto on teknisellä tasolla suhteellisen monimutkainen prosessi, eikä tälle tutkielmalle ole tarkoituksenmukaista käydä yksityiskohtaisesti läpi mikropalveluarkkitehtuurin käyttöönoton toteutusta. Näin ollen tässä osiossa käydään lyhyesti läpi käyttöönottoa tukevia teknologioita sekä strategiaa, jotta pystytään muodostamaan yleinen kuva käyttöönotosta.

Mikropalvelujen käsitteeseen kuuluvat olennaisina ominaisuuksina palveluiden eristettävyys sekä pienikokoisuus. Käyttöönoton näkökulmasta yksi vaihtoehto tukemaan näitä ominaisuuksia ovat konttiteknologiat (container). Kontilla tarkoitetaan kevennettyä sekä eristettyä versiota virtuaalikoneesta, mikä pitää sisällään pakatun, ajovalmiin sovelluksen tai sovelluksen osasen sekä ajoon vaadittavat muut lisäykset (Pahl, 2015). Esimerkiksi yhdessä kontissa voidaan ajaa mikropalvelua, mikä vastaa käyttäjien nimien hallinnasta. Käytännössä yhteen konttiin voitaisiin sijoittaa koko ohjelmisto monoliittisena toteutuksena, kuten virtuaalikoneeseenkin.

Konttiteknologiat ovat yleisesti käytetty vaihtoehto mikropalveluiden ajoon, koska konttienhallintajärjestelmät, kuten Docker, sekä konttien orkestrointijärjestelmät, kuten Kubernetes, mahdollistavat palveluiden skaalautuvuuden, luotettavuuden sekä reaktiivisuuden (Douglis & Nieh, 2019). Tämä toteutetaan pakkaamalla palvelut kontteihin, joita ajetaan halutussa ajoympäristössä, tässä tapauksessa Docker, jonka jälkeen konttien orkestrointijärjestelmä, kuten Kubernetes, hallinnoi konttirykelmää (cluster). Konttien orkestrointijärjestelmän tarkoituksena on muun muassa automaattisesti hallita konttien uudelleenkäynnistystä kaatumistilanteissa, jakaa suoritusresursseja palveluiden instansseille, taata terveiden instanssien halutun määrän saatavuus sekä tasapainottaa kutsuja instansseille (Richardson, 2019, s. 399). Näin menettelemällä saavutetaan skaalautuvuus: monta instanssia, eli konttia, samanaikaisessa suorituksessa, luotettavuus: orkestrointijärjestelmä käynnistää uudelleen kaatuneet kontit sekä reaktiivisuus: ohjelmisto reagoi resurssien saatavuuteen (jos esimerkiksi palvelin kaatuu, orkestrointijärjestelmä luo palveluiden instanssit toiselle palvelimelle).

Konttiteknologioiden lisäksi on myös olemassa muita vaihtoehtoja mikropalveluiden suoritusalustaksi. Mikropalvelut voidaan eristää omiin virtuaalikoneinstansseihinsa (Richardson, 2019, s. 390). Virtuaalikoneet ovat kontteja raskaampi vaihtoehto, sisältäen enemmän valmiiksi olemassa olevia resursseja, kuten täyden käyttöjärjestelmän version. Virtuaalikoneita hyödyntämällä saavutetaan muun muassa totaalinen palveluiden eristys laskentaresursseista lähtien, mutta toisaalta esimerkiksi resurssien tehokas hyödyntäminen vaikeutuu

(Richardson, 2019, s. 392). Tämän lisäksi mikropalveluita voidaan ajaa suoraan käytetyn ohjelmointikielen paketointijärjestelmällä (Richardson, 2019, s. 386), tai palvelittomissa (serverless) ympäristöissä, kuten AWS Lambdassa (Richardson, 2019).

Mikropalvelut, kuten kaikki muutkin sovellukset, tarvitsevat suoritusresursseja, kuten laskentatehoa sekä muistia, jotta niitä voidaan ajaa. Näin ollen mikropalveluille tulee tarjota resursseja joko fyysiseltä tai virtuaalikoneelta, eli hostikoneelta. Taibi, ym. (2018) tunnistivat kaksi strategiaa sijoittaa palveluita ajoon: monta palvelua hostilla (Multiple Service per Host) ja yksi palvelu hostilla (Single Service per Host). He kuitenkin toteavat, että hostin varaaminen yhdelle mikropalvelulle rikkoo mikropalveluiden perusperiaatteita, samalla laskien huomattavasti suoritustehoa sekä skaalautuvuutta (Taibi ym., 2018). Täten ainoaksi varteenotettavaksi käyttöönottostrategiaksi jää sijoittaa monta eri palvelua samalle hostille suoritettavaksi.

2.3.3 Datan varastointi

Mikropalvelut tarvitsevat myös paikan varastoida dataa, eli käytännössä tietokannan. Tässä osiossa tarkastellaan erilaisia tapoja lähestyä datan varastointia mikropalveluarkkitehtuurin rakenteen kannalta. Taibi, ym. (2018) tunnistivat kolme toisistaan eroaavaa käytäntöä varastoida dataa: tietokanta palvelua kohti (database-per-service), tietokantaklusteri (database cluster) ja jaettu tietokanta (shared database). Koska tietokantaparadigmalla ei ole suoranaista vaikutusta itse arkkitehtuurin rakenteeseen, vaan pikemminkin sovelluksen sisäiseen toteutukseen, ei tässä osiossa oteta kantaa valintoihin esimerkiksi relaatio- tai dokumenttitietokantojen välillä.

Tietokanta palvelua kohti -lähestymistapa on hyvin yksiselitteinen. Jokaiselle palvelulle luodaan oma yksityinen tietokanta, mihin vain kyseisellä palvelulla on pääsy (Taibi ym., 2018). Tällä lähestymistavalla edistetään palveluiden eristeisyyttä sekä helpotetaan esimerkiksi palveluiden kehitystyötä (Richardson, 2019, s. 12). Koska palveluiden eristeisyys on yksi mikropalveluarkkitehtuurin tärkeimmistä lähtökohdista, voidaan tätä lähestymistapaa pitää varteenotettavimpana valintana datan varastoinnin kannalta. Jakamalla jokaiselle palvelulle oma tietokantansa vältytään myös ristiriitaisuuksilta tietokantaparadigman ja tietokannanhallintajärjestelmän valinnan kanssa. Tällöin palveluille voidaan valita juuri palvelun tarpeisiin vastaava tiedonvarastointimenetelmä, eikä kompromisseja tarvitse tehdä (Hofmann, Schnabel & Stanley, 2016, s. 45).

Käytännössä ohjelmiston dataa on mahdollista varastoida myös tietokantaklusterille tai jaetulle tietokannalle. Mikropalveluiden näkökulmasta nämä vaihtoehdot näyttäytyvät samankaltaisina. Koska molemmissa tapauksissa tietokantaa käytetään samalla tavalla, tietokannat näkyvät mikropalveluille yhtenä tietokantana (Taibi ym., 2018). Tietokantaklusteri soveltuu skaalautuvuutensa takia tapauksiin, missä ohjelmiston dataliikenne on erittäin suurta ja jaetun tietokannan hyödyntäminen voi helpottaa siirtymistä monoliittisesta arkkitehtuurista mikropalveluihin (Taibi ym., 2018). Kuitenkin, koska näillä lähestymis-

tavoilla menetetään merkittävästi mikropalveluille oleellista datan eristeisyyttä, eivät tietokantaklusteri tai jaettu tietokanta ole ihanteellinen ratkaisu mikropalveluarkkitehtuurin datan varastointiin.

3 TIETOTURVA JA UHAT

Tässä kappaleessa määritellään tietoturva sekä tietoturvauhat tutkielman kontekstiin ja esitellään viitekehys näiden uhkien arviointiin. Aluksi määritellään tietoturva, jotta käsitteestä voidaan rajata pois tutkielmaan kuulumattomat näkökulmat. Tämän jälkeen määritellään tietoturvauhka, jonka avuksi esitellään lyhyesti kolme eri näkökulmaa luokitella tietoturvauhkia. Kolmannesta näkökulmasta, uhan kohteen mukaan luokittelusta, johdetaan tutkielmassa käytettävä viitekehys mikropalveluarkkitehtuurin tietoturvauhkien luokitteluun. Kappaleen tarkoituksena on muodostaa lukijalle yleinen kuva tietoturvasta, tietoturvauhista sekä perustella tutkielmassa käytettävää luokittelukehystä.

3.1 Tietoturvan määritelmä

Tietoturva on nykypäivänä erittäin laaja, poikkitieteellinen sekä globaalisti merkittävä ilmiö. Tietoturvan laaja-alaisen ulottuvuuden takia myös sen tarkka määritelmä voi vaihdella huomattavasti kontekstista riippuen. ISO/IEC 27000:2018 -standardissa tietoturva määritellään seuraavasti: ”tiedon luottamuksellisuuden, eheyden ja saatavuuden säilyttäminen” (2018). Tämän lisäksi huomautetaan, että määreitä kuten aitous, vastuuvelvollisuus, kiistämättömyys ja luotettavuus voidaan ottaa huomioon (ISO, 2018). Tietoturvalla pyritään siis suojaamaan toimijan omistamaa informaatiota. Pietras (2019) erittelee tietoturvalle neljä komponenttia: ICT-turvallisuuden, fyysisen turvallisuuden, henkilökohtais-organisaationaalisen turvallisuuden sekä laillisen turvallisuuden.

Tässä tutkielmassa tarkastellaan tietoturvaa sovellusarkkitehtuurin näkökulmasta. Näin ollen tietoturvan käsitteestä tutkielman kontekstissa rajataan pois fyysinen turvallisuus, henkilökohtais-organisaationaalinen turvallisuus sekä laillinen turvallisuus. Fyysisellä turvallisuudella tarkoitetaan fyysisen omaisuuden suojelemista esimerkiksi vartiointihenkilökunnalla tai palohälytinjärjestelmällä (Pietras, 2019). Henkilökohtais-organisaationaalinen turvallisuus kattaa toimintatavat, joilla hallinnoidaan organisaation henkilöstön fyysinen

pääsy suojatun datan tiloihin, kuten palvelinhuoneisiin (Pietras, 2019). Fyysiset uhat, kuten tulipalot tai tulvat, sekä henkilöstöstä johtuvat tietoturvauhat vaikuttavat kaikkiin järjestelmiin arkkitehtuurista riippumatta. Näin ollen tutkielman kannalta ei ole mielekästä tarkastella näitä tietoturvan komponentteja. Myöskään laillinen turvallisuus, mihin kuuluvat muun muassa lakisäädännölliset tekijät tietoturvaan liittyen (Pietras, 2019), eivät liity tutkielman aihepiiriin. Tällöin tutkielman kontekstissa tietoturvalla tarkoitetaan tarkemmin ottaen ICT-turvallisuuden komponenttia. ICT-turvallisuudella tarkoitetaan tietojärjestelmien, elektronisen datan sekä datan siirron suojaamista (Pietras, 2019). Koska tässä tutkielmassa tarkastellaan tietoturvaa tietyn ohjelmistoarkkitehtuurin näkökulmasta, tietoturvan käsite tarkentuu edelleen ICT-turvallisuudesta mikropalveluarkkitehtuurin menetelmin toteutetun järjestelmän, sen datan sekä datan siirron suojaamiseen.

3.2 Tietoturvauhat

Tietoturvauhalla tarkoitetaan uhkaa toimijan omistamaa informaatiota kohtaan. Tietoturvauhkia voidaan tarkastella monesta eri näkökulmasta, riippuen halutaanko esimerkiksi määrittää uhalle riskiarvio, tarkastella mistä uhka on lähtöisin, tai mitä järjestelmän osaa uhka koskee. Näin ollen tietoturvauhalle on haastavaa antaa tarkkaa määritelmää, koska uhat näyttäytyvät aina tapaus- ja näkökulmakohtaisesti. Koska tietoturvauhat ovat monimutkaisia sekä mahdollisesti vaikeasti tunnistettavia, seuraavaksi käydään lyhyesti läpi kolme esimerkkiä uhkia luokittelevista viitekehyksistä. Tarkoituksena on muodostaa kuva uhkien monimutkaisesta luonteesta sekä antaa esimerkkejä eri näkökulmista, joista uhkia voidaan tarkastella ja määritellä.

3.2.1 Luokittelu uhan lähteen mukaan

Jouini, Rabai ja Aissa (2014) esittelevät artikkelissaan ”puumaisen” mallin, jolla uhkia voidaan luokitella sen lähteestä alkaen, päättyen uhan vaikutukseen. Aluksi uhka luokitellaan joko ulkoiseen tai sisäiseen uhkaan, jonka jälkeen se luokitellaan edelleen joko ihmisen, ympäristön tai teknologian aiheuttamaksi. Tästä uhka luokitellaan pahansuovaksi (malicious) tai ei-pahansuovaksi (non-malicious), josta edelleen tahalliseksi tai tahattomaksi. Lopuksi uhalle määritetään vaikutus, joihin lukeutuvat esimerkiksi informaation tuhoutuminen tai informaation korruptoituminen. (Jouini ym., 2014)

Malli soveltuu hyvin uhkien luokitteluun tilanteissa, missä uhkien lähteellä sekä laajalla kokonaiskuvalla on merkitystä. Käytännön esimerkkinä voisi olla tilanne, missä halutaan selvittää organisaation tietoturvauhkien laajuus ja vaikutukset. Koska tässä tutkielmassa tarkastellaan tietoturvauhkia hyvin suppealta alueelta, tämän mallin hyödyntäminen uhkien luokittelussa ei tarjoa mielekästä tulosta.

3.2.2 Luokittelu riskin mukaan

Tietoturvauhkien riskiarvioiden muodostamiseksi on luotu monia erilaisia viitekehyksiä. Tuoreempana esimerkkinä toimii Rea-Guamanin, Mejían, San Feliun ja Calvo-Manzanon (2020) artikkelissaan esittämä AVARCIBER. AVARCIBER koostuu kuudesta eri aktiviteetista, jotka puolestaan koostuvat pienemmistä tehtävistä (task). Kun viitekehystä hyödynnetään onnistuneesti, saa toimija hyvin yksityiskohtaista sekä kokonaisvaltaista tietoa kokemistaan tietoturvauhista. Tähän sisältyvät muun muassa uhkien vaikutusarviot, toteutumisarviot sekä koituvien vahinkojen kustannusarviot. (Rea-Guaman ym., 2020)

AVARCIBER:illä pystytään muodostamaan erinomainen riskiarvio tietoturvauhista. Viitekehys soveltuu hyvin tilanteisiin, missä toimijan tietoturvakenttä sekä muut varat ovat jo tiedossa, koska viitekehyksen eri vaiheet tarvitsevat yksityiskohtaista tietoa esimerkiksi riskien toteutumisen rahallisista kustannuksista. Tässä tutkielmassa tarkastellaan tietoturvaa yleisemmältä tasolta ilman tietyn organisaation toiminnan analyysiä, jolloin tarkkoja tietoja esimerkiksi kustannuksista ei ole saatavilla. Täten riskin mukaan luokittelua on haasteellista soveltaa tässä tutkielmassa.

3.2.3 Luokittelu uhan kohteen mukaan

Sinha, Rai ja Bhushan (2019) luokittelevat tietoturvauhkia artikkelissaan kolmeen eri kategoriaan: verkkouhkiin (network threat), hostiuhkiin (host threat) ja sovelluskerroksen uhkiin (software threat). He muodostavat jokaisesta uhkakategoriasta taulukon, johon sijoitetaan kategoriaa koskevia uhkia sekä uhkien yleisiä piirteitä. Tämän lisäksi artikkelissa annetaan suuri määrä esimerkkejä kategorioita tyypillisesti koskevista tietoturvauhista. (Sinha ym., 2019)

Tällä lähestymistavalla pystytään muodostamaan yleinen kokonaiskuva tietoturvauhista sekä uhkien kohteina olevista järjestelmän osista. Näin ollen kohteen mukaan luokittelu soveltuu erittäin hyvin tämän tutkielman tarkoitukseen, koska tavoitteena on muodostaa yleinen käsitys mikropalveluarkkitehtuurin kohtaamista tietoturvauhista. Seuraavaksi esitellään tutkielmassa käytettävä uhkien luokittelun viitekehys.

Luvussa 2 on määritelty mikropalveluarkkitehtuurin kolme ”kerrosta” (orkestraatio ja koordinaatio, käyttöönotto ja datan varastointi). Tietoturvauhkien luokittelu näiden kerroksien mukaan on erittäin luontevaa, joten uhkia luokitellaan aluksi uhkakerroksen mukaan. Eri kerroksien kokemat samantyyppiset uhat (kuten palvelunestohyökkäykset) saattavat aiheuttaa järjestelmään erilaisia vaikutuksia, mikä tukee tarvetta luokitella uhkia kerroksien mukaan. Tarvittaessa kerroksista eritellään pienemmät osat sulkumerkinnällä helpottamaan uhkien jaottelua. Kerroksen määrittämisen jälkeen esitetään kerrosta koskevat tietoturvauhat. Uhkien nimeämisessä hyödynnetään muun muassa Sinhan, ym. (2019) kokoamaa listaa esimerkkiuhista. Uhkien nimeämisen jälkeen pyritään määrittämään uhille vaikutukset, missä hyödynnetään Jouinin,

ym. (2014) viitekehyksessä esitettyjä uhkien vaikutuksia. Mahdollisia vaikutuksia ovat:

- Informaation tuhoutuminen
- Informaation korruptoituminen
- Informaation paljastuminen luvattomille osapuolille
- Palvelun luvaton käyttö
- Käytön estyminen
- Käyttöoikeuksien muokkaaminen (elevation of privilege)
- Laiton käyttö (järjestelmän normaalien toimintojen käyttäminen hyökätessä muihin toimintoihin) (Jouini ym., 2014)

Edellä mainittujen vaikutusten lisäksi uhkiin voidaan tapauskohtaisesti liittää myös muita vaikutuksia, kuten muiden uhkatyyppien mahdollistamista. Seuraavana esimerkki taulukkomuodossa olevasta tietoturvauhkien luokittelusta tutkielman kontekstissa (taulukko 1).

TAULUKKO 1 Esimerkki tietoturvauhkien luokittelusta

Uhkakerros	**Uhka**	**Vaikutukset**
Orkestraatio ja koordinaatio (API-yhdyskäytävä)	Uhka 1 Uhka 2	Informaation tuhoutuminen
Orkestraatio ja koordinaatio (Palveluiden kommunikaatio)	Uhka 3	Palvelun luvaton käyttö
Käyttöönotto	Uhka 4	Käytön estyminen
	Uhka 5	Muiden uhkien mahdollistaminen
Datan varastointi	Uhka 6	Informaation korruptoituminen

On todettava, että tietoturvan elävän luonteen takia kaikkia uhkia on lähes mahdotonta listata, koska uusia uhkia ilmenee nopeaan tahtiin. Tämän johdosta tutkielmassa pyritään löytämään relevantteja esimerkkejä kerrosten kokemista uhista kaikenkattavan listauksen sijaan. Tällä lähestymistavalla pystytään silti muodostamaan hyvä kokonaiskuva mikropalveluarkkitehtuurin tietoturvauhista sekä uhkien vaikutuksista niiden toteutuessa.

4 MIKROPALVELUARKKITEHTUURIN TIETOTURVAUHAT

Tässä kappaleessa käydään läpi kirjallisuudesta löytyneitä mikropalveluarkkitehtuurin tietoturvauhkia kappaleessa kaksi esitetyn jaottelun järjestyksessä. Jokaisessa osiossa uhista annetaan ensin kirjallinen selitys, minkä jälkeen kerroksen tietoturvauhat tiivistetään taulukkomuotoon. Lopuksi esitetään yhteenvetotaulukko löytyneistä uhista sekä esitetään tiivistävää pohdintaa tuloksista sekä tutkielman kattavuudesta.

4.1 Orkestraatio ja koordinaatio

Orkestraation ja koordinaation kerros on mikropalveluarkkitehtuurin tietoturvan kannalta kriittinen tarkastelun alue. Mikropalveluiden välinen sekä järjestelmästä ulos lähtevä kommunikaatio toteutetaan tietoverkkojen kautta, mikä lisää hyökkäysrajapintaa verrattuna monoliittiseen arkkitehtuuriin (Pereira-Vale ym., 2021). Koska kerros toteuttaa käytännössä kaiken järjestelmässä tapahtuvan kommunikaation, on erittäin tärkeää, että kerrokseen kohdistuvat tietoturvauhat tiedostetaan.

4.1.1 API-yhdyskäytävää koskevat uhat

Järjestelmän pääsypisteenä toimiva API-yhdyskäytävä on kriittinen järjestelmän toiminnan kannalta, huolehtien järjestelmän kommunikaatiosta ulospäin. Näin ollen API-yhdyskäytävä toimii erinomaisena kohteena hyökkääjille, jotka haluavat esimerkiksi estää pääsyn järjestelmään, saada pääsyn järjestelmään tai kaapata järjestelmässä liikkuvaa kommunikaatiota. API-yhdyskäytävää koskevat uhat liittyvät erityisesti tietoverkkoja koskeviin uhkiin. Näistä uhista perinteisesti tärkeimpinä pidetään ARP-väärennöstä, välistävetohyökkäystä (Man in the middle, MITM) ja palvelunestohyökkäystä (DoS) (Yu, Yike, Yuqun & Xi, 2019).

ARP-väärennöksellä hyökätään kohteen ARP (Address Resolution Protocol) -tauluun, millä kartoitetaan kohteen IP- ja MAC-osoitteita (Lin, Koo & Wang, 2013). Ilman yksityiskohtaista tuntemusta tietoverkkoprotokollien toiminnasta hyökkäyksen toimintamekanismia on vaikea havainnollistaa, mutta käytännössä ARP-väärennöksellä voidaan vaikuttaa tietoverkossa tapahtuvan kommunikaation kohteisiin. ARP-väärennöksellä joko häiritään kohteen kommunikointia (toisin sanottuna toteutetaan palvelunestohyökkäys) tai mahdollistetaan välistävetohyökkäys (MITM) (Lin ym., 2013). Tämän lisäksi ARP-väärennöksellä voidaan matkia lähettäjää vastaamalla pyyntöihin kohteen sijasta (Abad & Bonilla, 2007). Mikropalveluarkkitehtuurin näkökulmasta ARP-väärennökset siis 1) mahdollistavat DoS- ja MITM- hyökkäyksiä sekä 2) voivat aiheuttaa datan korruptoitumista lähettäjää matkimalla. Etenkin tilanteessa, jossa mikropalvelun lähettämän datan muoto (esimerkiksi JSON) tunnetaan, lähettäjää matkimalla voidaan helposti vaikuttaa datan vastaanottajan näkemään informaatioon ilman, että sovellukseen sisäänrakennetut validointimenetelmät huomaavat väärennöstä. ARP-väärennösten tunnistaminen sekä niiltä suojautuminen on haasteellista (Lin ym., 2013), mutta onnistuneen hyökkäyksen seuraamuksien johdosta erittäin tärkeää.

Välistävetohyökkäyksessä hyökkääjä asettaa itsensä kohteen ja kohteen kanssa kommunikoivan toimijan väliin, välittäen heidän kommunikointiaan ”välikätenä” (Salifu, 2012). Tällöin hyökkääjä pääsee tarkastelemaan kaikkea kahden toimijan välillä tapahtuvaa kommunikaatiota, mistä voi seurata informaation paljastumista luvattomille osapuolille. Tästä voi puolestaan seurata esimerkiksi salasanojen tai suojausavainten vuotamista, joiden avulla voidaan murtautua syvemmälle järjestelmään (Salifu, 2012). Erityisesti API-yhdyskäytävän tapauksessa välistävetohyökkäyksellä voi olla erittäin laajoja vaikutuksia, koska yhdyskäytävä toimii kaiken järjestelmän ulkopuolisen kommunikaation välittäjänä. Näin ollen, pahimmassa tapauksessa, hyökkäyksen toteuttaja pääsee seuraamaan kaikkea järjestelmään saapuvaa sekä siitä lähtevää kommunikaatiota. Välistävetohyökkäykseen on kuitenkin olemassa toimivia suojamekanismeja, kuten palomuurit sekä datan salaaminen (Salifu, 2012). Datan salaamiseen liittyy kuitenkin omia ongelmiaan, kuten tehokkaiden salausalgoritmien vaatima prosessointiteho (Yu ym., 2019). Koska API-yhdyskäytävä käsittelee hyvin paljon kommunikaatiota eri mikropalveluilta verrattuna esimerkiksi monoliittisten järjestelmien kommunikaatioon, datan salauksen vaatima prosessointiteho kasvaa entisestään.

Palvelunestohyökkäyksien tarkoituksena on estää käyttäjien pääsy järjestelmään (Carl, Kesidis, Brooks & Suresh, 2006). Hyökkäyksen onnistuessa käyttäjien järjestelmään pääsy voi olla estynyt minuuteista jopa moniin päiviin (Zhiyuan, Jamdagni, Xiangjian, Nanda & Ren, 2014). Tutkielman viitekehyksen näkökulmassa palvelunestohyökkäyksen vaikutuksena on siis käytön estyminen. Edellä mainitun ARP-väärennöksen lisäksi hyökkäystapoja on yleisesti kaksi: heikkoushyökkäys (vulnerability attack) sekä tulvimishyökkäys (flooding attack) (Carl ym., 2006). Heikkoushyökkäyksessä järjestelmään lähetetään korruptoituneita pyyntöjä, joilla pyritään kaatamaan järjestelmä, kun taas tulvi-

mishyökkäyksessä pyritään ylikuormittamaan järjestelmä lähettämällä mahdollisimman monta pyyntöä jatkuvalla aikavälillä (Carl ym., 2006). Koska API-yhdyskäytävä toimii järjestelmän ainoana pääsypisteenä ulkoapäin, onnistunut palvelunestohyökkäys yhdyskäytävään estää pääsyn järjestelmään kokonaan. Tilanteessa, jossa yhdyskäytävään ei saada yhteyttä moneen päivään, seuraamukset järjestelmän omistajan liiketoiminnan kannalta voivat olla erittäin huomattavat.

Voidaan huomata, että API-yhdyskäytävä on altis monelle erilaiselle vihamieliselle hyökkäykselle. Koska yhdyskäytävä on ainoa pääsypiste järjestelmään ulkopuolelta, muodostuu siitä selkeä ”pullonkaula” järjestelmälle. Uhkien toteutuessa, eteenkin palvelunestohyökkäyksen tapauksessa, seuraamukset voivat olla vakavat. Vaikka listaus ei olisi täysin kattava eri hyökkäysmenetelmien kannalta, voidaan huomata, että jo näiden tietoturvauhkien vaikutukset ovat huomattavat. Täten voidaan todeta, että API-yhdyskäytävä on mikropalveluarkkitehtuurin tietoturvan kannalta erittäin kriittinen tarkastelun kohde, jonka suojaamiseen tulee varata tarvittava määrä resursseja. Seuraavassa taulukossa (taulukko 2) esitetään API-yhdyskäytävään kohdistuvat tietoturvauhat yhteenvetona tutkielmassa käytettävän luokittelukehyksen muodossa.

TAULUKKO 2 API-yhdyskäytävän tietoturvauhat

Uhkakerros	**Uhka**	**Vaikutukset**
Orkestraatio ja koordinaatio (API-yhdyskäytävä)	ARP-väärennös	Datan korruptoituminen, muiden hyökkäysmenetelmien mahdollistaminen
	Välistävetohyökkäys	Informaation paljastuminen luvattomille osapuolille
	Palvelunestohyökkäys	Käytön estyminen

4.1.2 Palveluiden kommunikaatiota koskevat uhat

Koska mikropalveluiden välille tarvitsee toteuttaa kommunikaatiota, muodostuu tästä arkkitehtuurin erityispiirteestä uusi hyökkäysrajapinta verrattuna monoliittiseen arkkitehtuuriin. Edellä mainittujen tietoverkkoja koskevien uhkien lisäksi palveluihin liittyy uhkia käyttöoikeuksien ja varmennuksen näkökulmista.

Tietoverkkoja koskevat uhat näyttäytyvät myös palveluiden välisessä kommunikaatiossa, koska kommunikaatio toteutetaan tietoverkkojen yli. Uhkien vaikutukset jäävät kuitenkin vähemmiksi verrattuna API-yhdyskäytävän tapaukseen, koska koko järjestelmän sijaan uhka vaikuttaa vain yhden palvelun ja yhdyskäytävän väliseen kommunikaatioon. Esimerkiksi palvelunestohyökkäyksen tapauksessa yhden palvelun instanssin ylikuormittaminen ei aiheuta kokonaiskuvassa suurta vaikutusta, koska skaalautuvuuden takia palveluista ylläpidetään monta instanssia samanaikaisesti. Toisaalta uhat ovat silti olemas-

sa, eikä esimerkiksi MITM-hyökkäyksellä kaapattujen salausavainten uhkaa tule sivuuttaa.

Palvelujen käyttöoikeuksien hallinta sekä varmennus on mainittu haastavaksi tietoturvan kannalta monessa lähteessä (Pereira-Vale ym., 2021; Yarygina & Bagge, 2018; Yu ym., 2019). Palveluiden varmennuksella (authentication) tarkoitetaan toimintatapoja, joilla palvelut varmennetaan juuri halutuiksi palveluiksi, eikä esimerkiksi luvattoman osapuolen järjestelmään sijoittamiksi tai hallitsemiksi. Jokainen palvelu tulisi varmentaa, koska jos järjestelmään päätyy hyökkääjän ohjaama palvelu, saattaa se lähettää tartuttavia pyyntöjä muille palveluille (Yu ym., 2019). Tartuttavien pyyntöjen seurauksena koko järjestelmä voi päätyä hyökkääjän haltuun, jos varmentamaton palvelu on kriittinen järjestelmän toiminnan kannalta. Täten varmentamattoman palvelun vaikutukset tietoturvaan ilmenevät palvelun (tässä tapauksessa mahdollisesti koko järjestelmän) luvattomana käyttönä. Luvattoman käytön seuraamukset vaihtelevat tapauskohtaisesti, riippuen muun muassa järjestelmän riippuvuudesta kyseisestä palvelusta, mutta esimerkiksi informaation korruptoituminen tai paljastuminen luvattomille osapuolille ovat hyvin realistisia uhkakuvia.

Palveluiden käyttöoikeuksilla tarkoitetaan palveluille annettuja oikeuksia lähettää pyyntöjä muille palveluille (Yarygina & Bagge, 2018). Käyttöoikeuksia jakamalla pyritään estämään muun muassa tilanne, jossa yhden palvelun hyökkääjän haltuun joutumisesta seuraa ”lumipalloefektillä” koko järjestelmän hallinnan menettäminen (Yarygina & Bagge, 2018). Käyttöoikeudet ulottuvat myös järjestelmän loppukäyttäjien puolelle, jolloin pyritään estämään luvattomien pyyntöjen lähettäminen järjestelmään (Yu ym., 2019). Jos käyttöoikeuksien jakamista ei toteuteta onnistuneesti, uhaksi muodostuu palvelun luvaton käyttö. Tämän lisäksi, jos hyökkääjä saa haltuunsa käyttöoikeuksia hallinnoivan järjestelmän, seuraukset ovat erittäin vakavat. Jos hyökkääjä pystyy mielivaltaisesti hallinnoimaan käyttöoikeuksia, pystyy hän käytännössä ottamaan koko järjestelmän haltuunsa esimerkiksi antamalla omille palveluilleen täydet oikeudet kutsua muita järjestelmän palveluita.

Käyttöoikeuksien hallinta sekä palveluiden varmennus ovat selvästi hyvin toisiinsa sitoutuneita käsitteitä. Molempien perimmäisenä tarkoituksena on estää järjestelmän luvaton käyttö, joko sisä- tai ulkopuolelta. Voidaan huomata, että luvattoman käytön seuraukset voivat kasvaa pahimmassa tapauksessa koko järjestelmän hallinnan menettämiseen, joten myös tämän mikropalveluarkkitehtuurin tietoturvan osa-alueen tarkastelu on erittäin tärkeää. Seuraavassa taulukossa (taulukko 3) esitetään palveluiden kommunikaatiota koskevat tietoturvauhat tutkielmassa käytettävän luokittelukehyksen muodossa.

TAULUKKO 3 Palveluiden kommunikaatiota koskevat tietoturvauhat

Uhkakerros	**Uhka**	**Vaikutukset**
Orkestraatio ja koordinaatio	Riittämätön varmennus	Palvelun luvaton käyttö
	Käyttöoikeuksien jakamisen epäonnistuminen	Palvelun luvaton käyttö

4.1.3 Huomioita

Voidaan huomata, että orkestraatio ja koordinaatio on mikropalveluarkkitehtuurin tietoturvan kannalta erittäin tärkeä osa-alue. Tietoturvauhat painottuvat haitallisiin hyökkäyksiin järjestelmää kohtaan, johtuen arkkitehtuurin lisääntyneestä kommunikaation tarpeesta verrattuna perinteisiin järjestelmiin. Tulee huomata, että myös muita uhkia on olemassa, kuten datan korruptoituminen sovelluksessa esiintyvän vian johdosta. Näiden uhkien tunnistaminen sekä korjaaminen on kuitenkin pääsääntöisesti sovelluskerroksen vastuu, eivätkä tämänkaltaisten uhkien mahdolliset vaikutukset yllä yhtä vakaviksi kuin edellä esitettyjen hyökkäysten. Näin ollen sovellusten epäonnistuneista toteutuksista johtuvat uhat on rajattu pois tästä osiosta. Tällöin kuitenkaan listaus ei ole täysin kattava, mutta siitä selviää vakavimmat kirjallisuudesta löydetyt tietoturvauhat orkestraation ja koordinaation kerrosta kohtaan. Seuraavassa taulukossa (taulukko 4) esitetään yhteenveto löydetyistä orkestraatiota ja koordinaatiota koskevista tietoturvauhista.

TAULUKKO 4 Orkestraation ja koordinaation tietoturvauhat

Uhkakerros	**Uhka**	**Vaikutukset**
Orkestraatio ja koordinaatio (API-yhdyskäytävä)	ARP-väärennös	Datan korruptoituminen, muiden hyökkäysmenetelmien mahdollistaminen
	Välistävetohyökkäys	Informaation paljastuminen luvattomille osapuolille
	Palvelunestohyökkäys	Käytön estyminen
Orkestraatio ja koordinaatio (Palveluiden kommunikaatio)	Riittämätön varmennus	Palvelun luvaton käyttö
	Käyttöoikeuksien jakamisen epäonnistuminen	Palvelun luvaton käyttö

4.2 Käyttöönotto

Käyttöönottoon liittyvät tietoturvaongelmat ovat paljolti sidottuja palveluita suorittaviin ja hallinnoiviin teknologioihin. Tässä osiossa käyttöönoton tietoturvauhkia tarkastellaan kolmesta eri näkökulmasta, joita ovat ajo, hallinta ja suoritusympäristö.

Koska mikropalveluita voidaan ajaa monella eri teknologiavaihtoehdolla, kuten konteissa tai virtuaalikoneilla, tulee arkkitehtuuria kehitettäessä ottaa huomioon teknologiavalinnasta koituvat tietoturvauhat. Tiettyjen teknologioiden tietoturvauhkien tarkastelu ei kuitenkaan ole tämän tutkielman kannalta tarkoituksenmukaista, joten suoritusteknologioiden tietoturvauhat käsitellään yhtenä kokonaisuutena arkkitehtuurin tietoturvan näkökulmasta. Tulee kuitenkin huomata, että suoritusteknologioiden tietoturvauhat ovat pääsääntöisesti

erittäin vakavia ja saattavat johtaa koko järjestelmän hallinnan menetykseen. Esimerkiksi kappaleessa 2.3.2 esitellyn Dockerin tietoturvasta on tehty laajaa tutkimusta, jossa on havaittu uhkia, joiden vaikutuksiin sisältyy muun muassa palvelunestohyökkäyksiä sekä palveluiden haltuunottoa (Combe, Martin & Pietro, 2016). Esimerkkejä ajoon kohdistuvien tietoturvauhkien vaikutuksista ovat näin ollen käytön estyminen sekä palvelun luvaton käyttö. Tämän lisäksi voidaan olettaa, että lähes kaikki muut vaikutukset ovat mahdollisia, mutta näitä ei listata esimerkkien puuteen takia.

Palveluiden hallintajärjestelmien tietoturvasta puolestaan ei olla tehty varteenotettavaa tutkimusta. Hallintajärjestelmien, kuten Kubernetes, vastuulle jää kuitenkin muun muassa palvelun halutun instanssimäärän ylläpitäminen sekä kutsujen tasapainotus instanssien välillä. Tällöin, jos hyökkääjä saa haltuunsa palvelun hallintajärjestelmän, kykenee hän esimerkiksi sammuttamaan haluamansa palvelun tai korvaamaan sen omallaan. Näin ollen, kuten ajon uhkien tapauksessa, esimerkkivaikutuksia ovat käytön estyminen sekä palvelun luvaton käyttö, joiden lisäksi lähes kaikki muut vaikutukset voivat myös olla mahdollisia. Maininnan arvoista on myös hallintajärjestelmien konfiguraation vaikeus mikropalveluarkkitehtuurin kompleksisuuden vuoksi. Jos konfiguraatio tehdään väärin, aukeaa siitä hyökkäysrajapinta, jonka kautta informaatiota voi vuotaa tai järjestelmän osat voivat joutua hyökkääjän haltuun (Pereira-Vale ym., 2021).

Mikropalveluarkkitehtuuriin pohjautuvan järjestelmän, kuten kaikkien muidenkin järjestelmien, suoritukseen tarvitaan laskentaresursseja. Näitä resursseja jaetaan suoritusympäristöstä, joihin lukeutuvat on-premise-ratkaisut sekä pilviympäristöt. Koska mikropalveluarkkitehtuuri on ”pilvinatiivi”, eli pilvessä suoritettavaksi suunniteltu ratkaisu (Balalaie ym., 2016), käsitellään tässä tutkielmassa suoritusalustana pilviympäristöjä. Pilviympäristöjen käyttöön liittyy hyötyjen lisäksi monia selkeitä tietoturvauhkia (Takabi, Joshi & Ahn, 2010), jotka tulee ottaa huomioon suoritusympäristöä valittaessa. Nämä uhat tiivistyvät tietoturvan hallinnan menetyksenä: ulkoista palveluntarjoajaa käytettäessä palveluntarjoajalla on täysi hallinta kaikesta, mitä pilvessä suoritetaan (Yarygina & Bagge, 2018).

Koska käyttöönoton tietoturva on hyvin riippuvaa teknologiavalinnoista, on tietoturvauhkien listaaminen tämän tutkielman kontekstiin haastavaa. Tässä osiossa on pyritty käsittelemään käyttöönottoon liittyviä tietoturvauhkia yleisellä tasolla. Näin menettelemällä on pystytty antamaan yleiskuva tietoturvan haasteista, jotka tulee ottaa huomioon arkkitehtuuria suunniteltaessa. Seuraavassa taulukossa (taulukko 5) esitetään yhteenveto osiossa ilmenneistä tietoturvauhista.

TAULUKKO 5 Käyttöönoton tietoturvauhat

Uhkakerros	Uhka	Vaikutukset
Käyttöönotto	Ajojärjestelmien uhat	Käytön estyminen, Palvelun luvaton käyttö, Mahdollisia muita vaikutuksia
	Hallintajärjestelmien uhat	Käytön estyminen, Palvelun luvaton käyttö, Mahdollisia muita vaikutuksia
	Väärin tehty konfiguraatio	Informaation paljastuminen luvattomille osapuolille, Palvelun luvaton käyttö, Mahdollisia muita vaikutuksia
	Suoritusympäristön uhat	Suorituksen hallinnan menetys

4.3 Datan varastointi

Datan varastoinnin kerrokseen liittyy arkkitehtuurin näkökulmasta määrällisesti vähiten tietoturvauhkia. Tietokantoja perinteisesti koskevat uhat, kuten SQL-injektiot ovat käytännössä sovelluskerroksen tietoturvauhkia, joten niitä ei tule käsitellä arkkitehtuuria koskevina uhkina. Myös esimerkiksi datan salaaminen tietokannan sisällä jää tietokannanhallintajärjestelmän tehtäväksi. Tämän lisäksi dataan liittyvät uhat, joita ovat datan kaappaaminen, salaisuuksien vuotaminen, tietokannan tuhoutuminen sekä datan peukalointi (data tamper) (Yu ym., 2019) on tietokannan tuhoutumista lukuun ottamatta käsitelty muiden kerrosten uhkina.

Kappaleessa 2.3.3 esitettiin kolme tiedon varastointimallia, joita ovat tietokanta palvelua kohden, tietokantaklusteri ja jaettu tietokanta. Tietoturvan näkökulmasta näistä lähestymistavoista turvallisin on tietokanta palvelua kohden, koska tietokantaklusterin tai jaetun tietokannan tapauksissa uhat vaikuttavat yhden palvelun sijaan moneen palveluun. Tiedon eristeisyyden poistumisen johdosta jaetun tietokannan tai tietokantaklusterin käyttö on itsessään tietoturvauhka, jonka vaikutuksena on tietoturvauhkien vaikutusten leviäminen sekä eristeisyyden poistuminen. Tämä ilmenee esimerkiksi tietokannan tuhoutumisen uhan tilanteessa: Jos käytössä on tietokantaklusteri tai jaettu tietokanta, informaatiota tuhoutuu suurempi määrä, kuin jos käytössä olisi tietokanta palvelua kohden.

Varastointimallien lisäksi uhkia nousee esiin datan käsittelyssä. Datan lähde (tietokanta) tulee suojata valtuuttamattomilta kutsuilta sekä kutsujen te-

kijän (mikropalvelun) tulee varmistua, että datan lähde on oikea (Yu ym., 2019). Jos arkkitehtuurissa käytetään tietokanta palvelua kohti -varastointimallia, tietokannan on vaivatonta tunnistaa, miltä palvelulta kutsut tulevat ja onko palvelu valtuutettu tekemään kutsuja. Kuitenkin ottamatta kantaa varastointimalliin, jos kutsujan tunnistus on toteutettu väärin, nousee uhkakuvaksi luvattoman osapuolen pääsy kantakyselyjen tekoon. Tästä voi seurata informaation tuhoutumista, korruptoitumista sekä paljastumista luvattomille osapuolille. Myös datan lähde pitää varmentaa kutsujalle, jotta kutsuja voi varmistua datan oikeellisuudesta (Esposito, Castiglione & Choo, 2016). Jos datan lähdettä ei varmisteta, voi palveluun päätyä väärää dataa ja näin ollen seurata informaation korruptoitumista.

Vaikka datan varastointiin liittyy arkkitehtuurin näkökulmasta määrällisesti vähemmän tietoturvauhkia verrattuna muihin kerroksiin, tulee datan varastoinnin tietoturva silti taata järjestelmää suunniteltaessa. Datan suojaus on ehdotonta toimijan liiketoiminnan varmistamiseksi. Seuraavassa taulukossa (taulukko 6) esitetään tässä datan varastointiin liittyvät löydetyt tietoturvauhat yhteenvetona.

TAULUKKO 6 Datan varastoinnin tietoturvauhat

Uhkakerros	**Uhka**	**Vaikutukset**
Datan varastointi	Jaettu tietokanta / tietokantaklusteri	Tietoturvauhkien vaikutusten leviäminen, Eristeisyyden poistuminen
	Tietokannan suojauksen epäonnistuminen luvattomilta osapuolilta	Informaation tuhoutuminen, Informaation korruptoituminen, Informaation paljastuminen luvattomille osapuolille
	Datan lähteen varmentamisen epäonnistuminen	Informaation korruptoituminen

4.4 Uhkien yhteenveto ja pohdinta

Tässä osiossa tarjotaan yhteenveto tutkielmassa löytyneistä mikropalveluarkkitehtuurin tietoturvauhista. Aluksi esitetään taulukkomuodossa kaikki löytyneet tietoturvauhat, minkä tarkoituksena on muodostaa yhteinen kokonaisuus tehdyn tutkimuksen tuloksista. Tämän jälkeen käydään pohdintaa tuloksista sekä tehdyistä huomioista. Seuraavassa taulukossa (taulukko 7) tutkielmassa löydetyt tietoturvauhat yhteenvetona.

TAULUKKO 7 Mikropalveluarkkitehtuurin tietoturvauhat

Uhkakerros	Uhka	Vaikutukset
Orkestraatio ja koordinaatio (API-yhdyskäytävä)	ARP-väärennös	Datan korruptoituminen, Muiden hyökkäysmenetelmien mahdollistaminen
	Välistävetohyökkäys	Informaation paljastuminen luvattomille osapuolille
	Palvelunestohyökkäys	Käytön estyminen
Orkestraatio ja koordinaatio (Palveluiden kommunikaatio)	Riittämätön varmennus	Palvelun luvaton käyttö
	Käyttöoikeuksien jakamisen epäonnistuminen	Palvelun luvaton käyttö
Käyttöönotto	Ajojärjestelmien uhat	Käytön estyminen, Palvelun luvaton käyttö, Mahdollisia muita vaikutuksia
	Hallintajärjestelmien uhat	Käytön estyminen, Palvelun luvaton käyttö, Mahdollisia muita vaikutuksia
	Väärin tehty konfiguraatio	Informaation paljastuminen luvattomille osapuolille, Palvelun luvaton käyttö, Mahdollisia muita vaikutuksia
	Suoritusympäristön uhat	Suorituksen hallinnan menetys
Datan varastointi	Jaettu tietokanta / tietokantaklusteri	Tietoturvauhkien vaikutusten leviäminen, Eristeisyyden poistuminen
	Tietokannan suojauksen epäonnistuminen luvattomilta osapuolilta	Informaation tuhoutuminen, Informaation korruptoituminen, Informaation paljastuminen luvattomille osapuolille
	Datan lähteen varmentamisen epäonnistuminen	Informaation korruptoituminen

Voidaan huomata, että mikropalveluarkkitehtuuri todellakin lisää hyökkäysrajapintaa järjestelmää kohtaan. Palvelujen eristämisen johdosta tarvittavat ratkaisut sekä teknologiat tuovat mukanaan omia kriittisiä tietoturvauhkiaan, jotka vaativat erityistä huomiota arkkitehtuuria suunniteltaessa. Löydetyt uhat ovat pääosin vaikutuksiltaan hyvin vakavia, mikä tukee tarvetta panostaa resursseja arkkitehtuurin tietoturvaan järjestelmän toiminnan turvaamiseksi.

Orkestraation ja koordinaation kerroksen tietoturvauhat näyttäytyivät ulkopuolelta järjestelmään kohdistettujen hyökkäysten muodossa. Syynä tälle on kommunikaatioon vaadittavat tietoverkot, jotka toimivat selvästi erinomaisena hyökkäysrajapintana. Vaikka tietoverkkoja koskevat uhat koskevat yleisesti

ottaen kaikkia järjestelmiä, sisällytettiin ne tähän tutkielmaan vaikutusten vakavuuden vuoksi. Huomattiin, että tietoverkkoja koskevien uhkien hyökkäysrajapinta käytännössä kasvaa huomattavasti mikropalveluarkkitehtuurissa verrattuna esimerkiksi monoliittiseen arkkitehtuuriin, koska ulkopuolisen kommunikaation lisäksi palveluiden välinen kommunikaatio täytyy toteuttaa tietoverkoilla. Tämän lisäksi mielenkiintoisena voidaan pitää uhkien ”lumipalloefektiä”. Tällä tarkoitetaan uhkien vaikutuksia tarkastellessa huomattua ilmiötä, missä pieneltäkin vaikuttavan murron onnistuessa saattaa koko järjestelmä joutua hyökkääjän hallintaan.

Käyttöönoton kerroksen tietoturvauhkia tarkasteltaessa huomattiin, että uhat ovat sidonnaisia käyttöönotossa käytettäviin teknologioihin. Tämän johdosta tietoturvauhkien tarkastelu täytyi suorittaa hyvin yleisellä tasolla, lähestyen käyttöönoton tietoturvaa ryhmitellen teknologioita ajojärjestelmiin, hallintajärjestelmiin sekä suoritusympäristöihin. Näin menettelemällä pystyttiin muodostamaan yleiskuva käyttöönottoon liittyvistä tietoturvauhista ottamatta kantaa käytettyihin teknologioihin. Löydetyt tietoturvauhat ja vaikutukset olivat mahdollisesti hyvin vakavia, joten voidaan todeta, että käyttöönottoteknologioita valittaessa tulee näiden tietoturvaan tutustua hyvin yksityiskohtaisesti uhilta suojautuakseen. Myös käyttöönoton tietoturvauhissa huomattiin ”lumipalloefektiä”, kuten orkestraation ja koordinaation tapauksessa.

Datan varastoinnin kerroksesta löydettiin määrällisesti vähiten tietoturvauhkia arkkitehtuurin näkökulmasta. Kuitenkin datan ollessa liiketoiminnan ytimessä, ovat datan varastointiin liittyvät tietoturvauhat erittäin tärkeitä järjestelmän yleisen tietoturvan kannalta. Voitiin huomata, että tietoturvan näkökulmasta jaetun tietokannan tai tietokantaklusterin käyttö on selvästi huonompi vaihtoehto verrattuna tietokanta palvelua kohden -malliin. Yleisenä huomiona voidaan todeta, että datan varastointia perinteisesti koskevat tietoturvauhat jäävät suurimmaksi osaksi itse sovellusten ja tietokannanhallintajärjestelmän vastuulle, jotka jäävät tutkielman näkökulman ulkopuolelle.

Kuten kappaleen kolme lopussa todettiin, tietoturvauhkien täydellinen listaaminen on erittäin haastavaa. Tutkielmassa pystyttiin tästä huolimatta muodostamaan kattava yleiskatsaus mikropalveluarkkitehtuurin tietoturvauhista, niiden suuresta määrästä sekä niiden vakavuudesta.

5 YHTEENVETO

Tässä tutkielmassa tarkasteltiin mikropalveluarkkitehtuuria, tietoturvaa sekä tietoturvauhkia lähdekirjallisuuteen perustuen. Aluksi käytiin läpi mikropalveluarkkitehtuurin rakenne, jonka jälkeen siirryttiin tarkastelemaan tietoturvan käsitettä. Lopuksi nämä näkökulmat yhdistettiin mikropalveluarkkitehtuurin tietoturvauhkien tarkasteluun. Tutkielman tutkimuskysymyksenä toimi ”*Mitä tietoturvauhkia mikropalveluarkkitehtuurin hyödyntäminen sisältää tai mahdollistaa?*”.

Tutkimuskysymykseen pystyttiin muodostamaan yleiskattava vastaus kappaleessa neljä. Koska kappaleen neljä lopussa annettiin koostava pohdinta tuloksista, tuloksia ei eritellä yksityiskohtaisesti enää tässä kappaleessa. Tärkeimpinä löytöinä esiin nousivat kuitenkin tietoturvauhkien suuri hyökkäysrajapinta, yleinen vakavuus sekä uhkien vaikutusten ”lumipalloefekti”. Tutkielman tarkoituksena oli muodostaa kokoava yleiskatsaus sekä antaa esimerkkejä mikropalveluarkkitehtuurin eri kerroksia koskevista tietoturvauhista. Tästä näkökulmasta tutkielman tarkoituksessa onnistuttiin.

Tutkielman tuloksia voidaan pitää pääsääntöisesti luotettavina. Tietoturva on luonteeltaan hyvin tapauskohtainen käsite, joten kaikkia esitettyjä uhkia ei luultavimmin ilmene jokaisessa yksittäisessä tarkasteltavassa tapauksessa. Joihinkin esitetyistä uhista, kuten esimerkiksi palvelunestohyökkäyksiin, on olemassa sisäänrakennettuna toteutettuja ratkaisuja esimerkiksi pilvipalveluntarjoajan tarjoamana. Tämä ei kuitenkaan poista uhan olemassaoloa, joten uhasta on hyvä olla tietoinen.

Tutkielmaprosessin aikana aihepiiriin tehtiin monia rajauksia, jotka vaikuttivat lopulliseen tutkimustulokseen. Heti alussa tutkielman aihepiiristä rajattiin pois kyberfyysiset järjestelmät, kuten IoT-laitteet. Perusteena tälle oli kyberfyysisten järjestelmien kokemat moninaiset tietoturvauhat, joita ei ole yleistettävissä muihin järjestelmiin. Toinen huomattava rajaus tehtiin tietoturvan käsitteessä, jossa näkökulma kavennettiin ICT-turvallisuuteen. Jos tutkielmaan olisi sisällytetty muitakin tietoturvan näkökulmia, olisi tuloksista saattanut muodostua liian yleisiä, jolloin painotus mikropalveluarkkitehtuuriin olisi voinut heikentyä. Tulee kuitenkin huomata, että tietoturvan ”heikoin lenkki” on yleensä ihminen, jolloin järjestelmän kohtaamat vakavimmat tietoturvauhat

koituvat luultavimmin teknologiasta riippumattomista tekijöistä. Näiden rajausten lisäksi myös mikropalveluarkkitehtuurin rakenteesta jätettiin tietoisesti käsittelemättä tiettyjä yksityiskohtia. Palveluiden välisen kommunikaation käytännön toteutukset, joihin lukeutuvat muun muassa viestiväylät (message bus) ja palveluverkot (service mesh), sekä palomuurin toiminta jätettiin käsittelemättä. Näiden tarkka ymmärtäminen ei kuitenkaan ole ollut edellytys tietoturvauhkien analysointiin, joten tulosten voidaan olettaa olevan luotettavia tältäkin osin.

Tutkielman tuloksia voidaan olettaa pystyttävän käyttämään käytännön päätöksenteon tukena. Tutkielma tarjoaa arkkitehtuuria suunnittelevalle toimijalle yleisen katsauksen tietoturvan näkökulmasta pohdintaa vaativista kohdista. Tämän lisäksi tutkielman tulokset voivat herätellä toimijaa tarkastelemaan järjestelmänsä tietoturvaa yksityiskohtaisemmin jo yleisellä tasolla löydettyjen uhkien määrän sekä vakavuuden johdosta. Tutkielman tieteellisenä kontribuutiona toimii tiiviisti kirjallisuuskatsauksen menetelmillä koottu listaus yleisimmistä mikropalveluarkkitehtuurin tietoturvauhista, jonka pohjalta voidaan alkaa analysoimaan tietoturvauhkia tarkemmin. Tämän lisäksi tutkielmassa johdettu viitekehys soveltuu myös muiden järjestelmien tietoturvauhkien tiiviiseen kokoamiseen.

Jatkotutkimusaiheita pohdittaessa esiin nousee heti ensimmäisenä uhkien ratkaisujen etsiminen. Lähdekirjallisuutta etsittäessä löydettiin suuri määrä esitetyiltä tietoturvauhkilta suojautumiseen tehtyjä ehdotuksia, joista pystyttäisiin muodostamaan mielekäs kirjallisuuskatsaus. Käytännön tutkimuksessa voitaisiin ottaa tarkastelun alle tietty olemassa oleva järjestelmä, jonka tietoturvauhkia voitaisiin analysoida. Tämän lisäksi tutkimuksen kohteeksi voitaisiin ottaa yksi tutkielmassa esitetty teknologia, kuten Docker tai Kubernetes, joiden tietoturvaa voitaisiin tutkia yksityiskohtaisemmin. Myös mikropalveluarkkitehtuurin tarkempi vertailu muihin arkkitehtuurimalleihin tietoturvan näkökulmasta voisi olla mielekäs tutkimuksen kohde.

LÄHTEET

Abad, C. L. & Bonilla, R. I. (2007). An analysis on the schemes for detecting and preventing ARP cache poisoning attacks. Teoksessa *27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07)* (60-66). Toronto, Canada.

Balalaie, A., Heydarnoori, A. & Jamshidi, P. (2016). Microservices architecture enables DevOps: Migration to a cloud-native architecture. *IEEE Software, 33*(3), 42-52.

Bucchiarone, A., Dragoni, N., Dustdar, S., Larsen, S. T. & Mazzara, M. (2018). From monolithic to microservices an experience report from the banking domain. *IEEE Software, 35*(3), 50-55.

Carl, G., Kesidis, G., Brooks, R. R. & Suresh, R. (2006). Denial-of-service attack-detection techniques. *IEEE Internet Computing, 10*(1), 82-89.

Combe, T., Martin, A. & Pietro, R. (2016). To docker or not to docker: A security perspective. *IEEE Cloud Computing, 3*, 54-62.

Douglis, F. & Nieh, J. (2019). Microservices and containers. *IEEE Internet Computing, 23*(6), 5-6.

Dragoni, N., Giallorenzo, S., Lluch-Lafuente, A., Mazzara, M., Montesi, F., Mustafin, R. & Safina, L. (2017). Microservices: Yesterday, today, and tomorrow. Teoksessa Mazzara, M. & Meyer, B. (toim.), *Present and Ulterior Software Engineering* (195-216). Springer.

Dragoni, N., Lanese, I., Thordal Larsen, S., Mazzara, M., Mustafin, R. & Safina, L. (2018). Microservices: How to make your application scale. Teoksessa Petrenko, A. K. & Voronkov, A. (toim.), *Perspectives of System Informatics: 11th International Andrei P. Ershow Informatics Conference* (95-104). Moscow, Russia: Springer.

Esposito, C., Castiglione, A. & Choo, K. R. (2016). Challenges in delivering software in the cloud as microservices. *IEEE Cloud Computing, 3*(5), 10-14.

Hassan, S., Ali, N. & Bahsoon, R. (2017). Microservice ambients: An architectural meta-modelling approach for microservice granularity. Teoksessa *2017 IEEE International Conference on Software Architecture (ICSA)* (1-10). Gothenburg, Sweden.

Hofmann, M., Schnabel, E. & Stanley, K. (2016). *Microservices best practices for java* (First edition). Poughkeepsie, NY: IBM Corporation, International Technical Support Organization.

IBM. (2020). IBM cost of a data breach report 2020 . Haettu 15.5.2021 osoitteesta https://www.ibm.com/security/digital-assets/cost-data-breach-report/

Jouini, M., Rabai, L. B. A. & Aissa, A. B. (2014). Classification of security threats in information systems. *Procedia Computer Science, 32*(C), 489-496.

International Organization for Standardization. (2018). *Information technology — Security techniques — Information security management systems — Overview and vocabulary* (ISO/IEC Standard No. 27000:2018). Haettu osoitteesta https://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

Larrucea, X., Santamaria, I., Colomo-Palacios, R. & Ebert, C. (2018). Microservices. *IEEE Software, 35*(3), 96-100.

Lin, J. C., Koo, M. J. & Wang, C. S. (2013). A proposal for a schema for ARP spoofing protection. *Applied Mechanics and Materials, 284-287*, 3275-3279.

Pahl, C. (2015). Containerization and the PaaS cloud. *IEEE Cloud Computing, 2*(3), 24-31.

Pereira-Vale, A., Fernandez, E. B., Astudillo, H. & Márquez, G. (2021). Security in microservice-based systems: A multivocal literature review. *Computers & Security, 103*.

Pietras, E. (2019). Information security – its essence and threats. *Scientific Journal of the Military University of Land Forces, 191*(1), 26-35.

Rea-Guaman, A. M., Mejía, J., Feliu, T. S. & Calvo-Manzano, J. A. (2020). AVARCIBER: A framework for assessing cybersecurity risks. *Cluster Computing, 23*(3), 1827-1843.

Richardson, C. (2019). *Microservices patterns : With examples in java* (1st edition). Shelter Island, NY: Manning Publications.

Salifu, A. (2012). Detection of man-in-the-middle attack in computer networks. *I-Manager's Journal on Communication Engineering and Systems, 2*(1), 1-8.

Sinha, P., Rai, A. k. & Bhushan, B. (2019). Information security threats and attacks with conceivable counteraction. Teoksessa *2019 2nd International Con-*

ference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT) (1208-1213).

Stephens, R. (2015). *Beginning software engineering*. Indianapolis, IN: Wrox, a Wiley Brand.

Taibi, D., Lenarduzzi, V. & Pahl, C. (2018). Architectural patterns for microservices: A systematic mapping study. Teoksessa Méndez Muñoz, V., Ferguson, D., Helfert, M. & Pahl, C. (toim.), *Proceedings of the 8th International Conference on Cloud Computing and Services Science* (221-232). Funchal, Madeira, Portugal: SciTePress.

Takabi, H., Joshi, J. B. D. & Ahn, G. (2010). Security and privacy challenges in cloud computing environments. *IEEE Security & Privacy, 8*(6), 24-31.

Yarygina, T. & Bagge, A. H. (2018). Overcoming security challenges in microservice architectures. Teoksessa *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)* (11-20). Bamberg, Germany.

Yu, D., Yike, J., Yuqun, Z. & Xi, Z. (2019). A survey on security issues in services communication of microservices-enabled fog applications. *Concurrency and Computation: Practice and Experience, 31*(22).

Zhiyuan, T., Jamdagni, A., Xiangjian, H., Nanda, P. & Ren, P. L. (2014). A system for denial-of-service attack detection based on multivariate correlation analysis. *IEEE Transactions on Parallel and Distributed Systems, 25*(2), 447-456.