Tampereen ammattikorkeakoulu

Turvallinen yhdyskäytäväratkaisu

Pauliina Hovila

OPINNÄYTETYÖ
Kesäkuu 2021

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

HOVILA, PAULIINA:
Turvallinen yhdyskäytäväratkaisu

Opinnäytetyö 19 sivua, joista liitteitä 0 sivua
Kesäkuu 2021

Opinnäytetyön tarkoituksena oli tarkastella turvallista yhdyskäytävätoteutusta, jonka toteutukseen opinnäytetyöntekijä osallistui työhön liittyvässä asiakasprojektissa. Yhdyskäytävä toteutettiin osana isompaa ohjelmistokehitysprojektia, joka asetti tarkat vaatimukset toteutuksen tietoturvalle. Turvallisen yhdyskäytävän tarkoituksena oli suodattaa sovelluskokonaisuudessa välitettyä tietoa kahden eri turvaluokitellun ympäristön välillä. Näin yhdyskäytävä pystyy estämään korkeamman turvaluokituksen ympäristön tiedon pääsyn matalamman turvaluokituksen ympäristöön käyttäen alkiotunnistukseen perustuvaa sisällönsuodatusratkaisua.

Sovellus toteutettiin muusta sovelluskokonaisuudesta erillisenä Spring Boot -sovelluksena, jossa välitettyjä viestejä suodatettiin sisällönsuodatussovelluksen lisäksi verkkotason suojauksella muun muassa IP-rajauksella. Toteuttamalla rajausta sekä sovellus- että verkkotasolla ympäristö saavutti projektilta vaaditun turvaluokitustason. Ottaen huomioon useimpien yhdyskäytäväratkaisuiden tunnetut heikkoudet, yhdensuuntaisen liikenteen ratkaisuiden vaatimat sovellusmuutokset ja laiteperusteisten ratkaisuiden rajallisuudet, alkiotunnistukseen perustuva ratkaisu oli oikea valinta kyseiseen tarpeeseen ja vaatimuksiin.

Asiasanat: yhdyskäytävä, sisällönsuodatus, tietoturvallisuus

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Software Engineering

HOVILA, PAULIINA:
Secure Gateway Solution

Bachelor's thesis 19 pages, appendices 0 pages
June 2021

The purpose of this thesis was to examine the implemented secure gateway solution. The gateway was implemented as part of a larger software development project, which set strict requirements for the information security of the application. The purpose of the secure gateway was to filter the transmitted messages within the application environment that covered two different security classification levels to prevent higher level environment information reaching lower-level environment using filtering based on object identification.

Application was implemented separately from the rest of the applications as a standalone Spring Boot application, where transmitted messages were filtered in the content filtering application as well as in the network level using IP restrictions. Using filtering both in application and network levels the environment reached the required security classification. Considering the known weaknesses of most other gateway solutions, application modifications needed for the one-way communication solutions and the limitations of hardware solutions, object identification-based solution was the correct selection in this particular need and requirements.

Key words: gateway, content filtering, information security

SISÄLLYS

LYHENTEET JA TERMIT

AWS	Amazon Web Services
HTTPS	HyperText Transfer Protocol
IP	Internet Protocol
JSON	JavaScpript Object Notation
JVM	Java Virtual Machine
KVM	Keyboard, Video, Mouse
OSI	Open Systems Interconnection
TSL	Transport Layer Security
VPN	Virtual Private Network
XML	Extensible Markup Language

1 JOHDANTO

Opinnäytetyössä tutkitaan tietojen välitystä suljettuihin, korkeamman turvaluokituksen verkkoihin alkiotunnistukseen perustuvan sisällönsuodatusratkaisun avulla. Opinnäytetyöhön liittyen toteutetaan Spring Boot -pohjainen sovellus, joka validoi läpikulkevat viestit ja välittää oikeelliset viestit läpi. Opinnäytetyössä tutkitaan myös vaihtoehtoisia tapoja toteuttaa turvallinen yhdyskäytäväratkaisu ja valittujen toteutustapojen turvallisuutta viranomaisten asettamiin kriteereihin nähden.

2 KÄSITTEET

2.1 Yhdyskäytävä

Yleisesti yhdyskäytävällä tarkoitetaan tietoliikenteessä verkossa olevaa kohtaa, joka mahdollistaa liikennöinnin verkosta toiseen. Tässä opinnäytetyössä yhdyskäytävällä tarkoitetaan ratkaisua, jolla mahdollistetaan suojaustasoltaan erilaisten ympäristöjen liittäminen toisiinsa. Yhdyskäytävän tehtävä on liikennöinnissä sallia vain matalamman suojaustason tiedon siirtyminen korkeamman suojaustason ympäristöstä matalamman tason ympäristöön (Ohje yhdyskäytäväratkaisujen… 2018, 4).

Yhdyskäytävien suunnittelun tavoitteena on tavallisesti seurata Bell-LaPadula -mallin sääntöjä tietojen lukemisesta ja kirjoittamisesta (Ohje yhdyskäytäväratkaisujen… 2018, 4). Mallin mukaan hyväksyttävässä yhdyskäytäväratkaisussa ylemmän suojaustason informaatio ei saa siirtyä matalamman suojaustason ympäristöön, ja täten yhdyskäytäväratkaisun pitää toteuttaa mallin perusperiaatteet ”No Read Up” sekä ”No Write Down”. Jotta nämä säännöt toteutuisivat, Bell-LaPadula -mallin hyväksymät yhdyskäytäväratkaisut toteutetaan joko yksisuuntaisina suodatusratkaisuina, jolloin liikennöinti tapahtuu vain matalammalta tasolta korkeammin suojatulle tasolle tai sisällönsuodatusratkaisuina, joissa informaatio suodatetaan tunnistamisen jälkeen, niin että vain matalamman tason informaation kulku ylemmältä tasolta matalamman tason ympäristöön sallitaan.

2.2 Turvallisuusluokitukset

Katakri eli Kansallinen turvallisuusauditointikriteeristö määrittelee muun muassa tietoliikenneturvallisuuteen liittyvät turvallisuusluokitukset, jotka pyrkivät yhtenäistämään ja tehostamaan turvallisuustoiminnan toiminta- ja toteutustapoja (Katakri 2011, 73).

Turvallisuusluokitukset on tietoturvan osalta jaettu kolmeen tasoon: perustason vaatimukset (ST4), korotetun tason vaatimukset (ST3) ja korkean tason vaatimukset (ST2). Kriteerit vastaavat yleisimpiin kysymyksiin vaaditusta tietoturvaratkaisusta liittyen kaikkiin kolmeen vaatimustasoon. Näin ollen toteuttaessa näihin turvaluokituksiin peilattavaa toteutusta, käytännön toteutuksen ominaisuuksia on helppo peilata turvallisuusvaatimuksiin tai käyttää kriteereitä hyväksi esimerkiksi jo sovelluksen toiminnallisuuksia määriteltäessä. Katakri määrittelee kaikille suojaustasoille teknisen tietoturvallisuuden arviointiin käytettävät vaatimukset erikseen tietoliikenne-, tietojärjestelmä-, tietoaineisto- ja käyttöturvallisuudelle (Katakri 2015, 29).

3 TEKNOLOGIAT

3.1 Kotlin

Kotlin on erilaisille alustoille sopiva avoimen lähdekoodin staattisesti tyypitetty ohjelmointikieli (Kotlin Programming Language 2021). Kotlin on suunniteltu yhteensopivaksi Java-ohjelmointikielen kanssa ja koska se kääntyy muun muassa JVM-tavukoodiksi, sitä pystytään ajamaan ympäristöissä, jotka tukevat Java-kielen määrittelyn mukaisia virtuaalikoneita. Java-yhteensopivuuden vuoksi Kotlin on yhteensopiva kaikkien olemassa olevien Java-viitekehyksien ja työkalujen kanssa.

3.2 Spring

Spring -viitekehys tarjoaa kattavan infrastruktuurituen Java-pohjaisille sovelluksille (Comparison Between… 2021). Spring -viitekehyksen avulla ohjelmistojen kehitystä voidaan helposti nopeuttaa ja tehostaa, koska monet sovelluksissa toistuvat rakenteet ja toiminnallisuudet, jotka ilman viitekehystä pitäisi kirjoittaa alusta alkaen, voidaan lisätä koodiin käyttämällä viitekehyksen tarjoamia valmiita moduuleita. Valmis moduuli voidaan lisätä koodiin helposti, ja sen konfiguraatioita voidaan säätää tapaukseen sopivaksi vain muutamalla rivillä koodia.

3.3 Spring Boot

Spring Boot on Spring -viitekehyksen yksinkertaistetumpi laajennus, jonka tavoitteena on tehdä kehitysympäristöstä nopeampi ja tehokkaampi karsimalla ylimääräistä konfiguraatiota, jota alkuperäinen Spring -viitekehys käyttää (Spring n.d.). Spring Boot -viitekehys tarjoaa valmiiksi ns. aloituspaketin, johon on koottu yleisimmät riippuvuudet, ja jonka konfiguraatioita on pyritty automatisoimaan mahdollisimman paljon (Comparison Between… 2021).

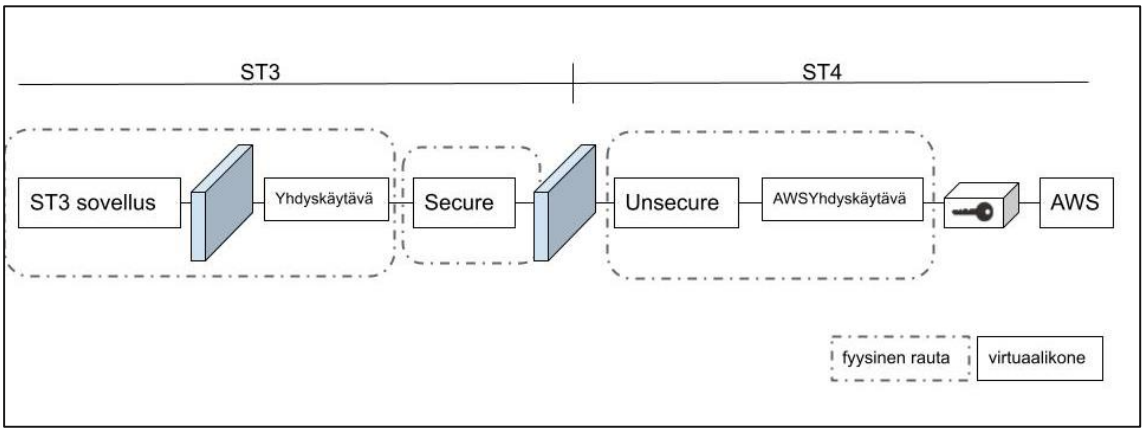
3.4 Apache Camel

Apache Camel on avoimen lähdekoodin integraatioviitekehys, jolla pystytään määrittelemään muun muassa integraatioiden väliohjelmistojen viestitoteutusten reititysten ja välitysten sääntöjä (What is Camel? n.d.). Camelin toiminta perustuu vallitsevaan Enterprise Integration Patterns -malliin, joka sisältää useita integraatioiden yhteydessä käytettäviä viestintämalleja. Camel mahdollistaa integraatiot useiden järjestelmien kanssa käyttäen samaa rajapintaa viestintätavasta riippumatta ja se tukee myös erimuotoisia viestintäprotokollia ja datatyyppejä kuten esimerkiksi HTTP-protokolaa (Introduction to… 2020).

4 TOTEUTUS

4.1 ARKKITEHTUURI

Kuvio 1 kuvaa toteutuksen kokonaisarkkitehtuuria, jossa nähdään sovelluskokonaisuuteen kuuluvat virtuaalikoneilla ajettavat sovellukset, erilliset yhdyskäytävät, palomuurit sekä VPN-yhteys, jota kuvataan avain -kuvaajalla AWS-ympäristön edessä. Kuviossa on nähtävissä myös suojaustasot ja niihin liittyvät fyysiset erottelut toteutuksessa.



KUVIO 1. Toteutuksen arkkitehtuuri

Arkkitehtuurissa huomattavaa on tietoturvan vuoksi erillisraudoille sijoitetut sovelluskokonaisuudet. Vaikka toteutuksessa pyrittiinkin käyttämään mahdollisimman vähän erillisiä rautoja, virtuaalikoneiden erottelu ei ollut tietoturvan kannalta riittävä vaan ST3-sovelluksen olemassa olevaan rautaan liittämisen sijaan myös Secure -komponentti vaati erillisen raudan alleen ja ST4-toteutus piti sijoittaa myös erillisraudalle.

4.2 YHDYSKÄYTÄVÄTOTEUTUS

Projektin toteutukseen valittiin yhdyskäytävän ratkaisuvaihtoehdoksi alkiotunnistukseen perustuva sisällönsuodatusratkaisu, joka mahdollisti tietoliikenteen ylemmän suojaustason ympäristöstä matalamman tason ympäristöön, sekä myös matalammalta tasolta ylemmän tason ympäristöön yhden suojaustason ylittävänä ratkaisuna (Ohje yhdyskäytäväratkaisujen…

2018, 6). Sovelluskokonaisuuteen kuuluu ST3-suojaustason sovellus sekä toisessa päässä AWS-pilvialustan päälle rakennettu sovellus (kuvio 1). AWS-pohjaiseen sovellukseen välitettävän tiedon pitää olla ST4-turvallisuusluokiteltua, jotta se voidaan esittää pilviympäristössä (PiTuKri 2020, 33), eli kokonaistoteutuksen tietoliikenne on suojaustasojen ST3 ja ST4 välinen.

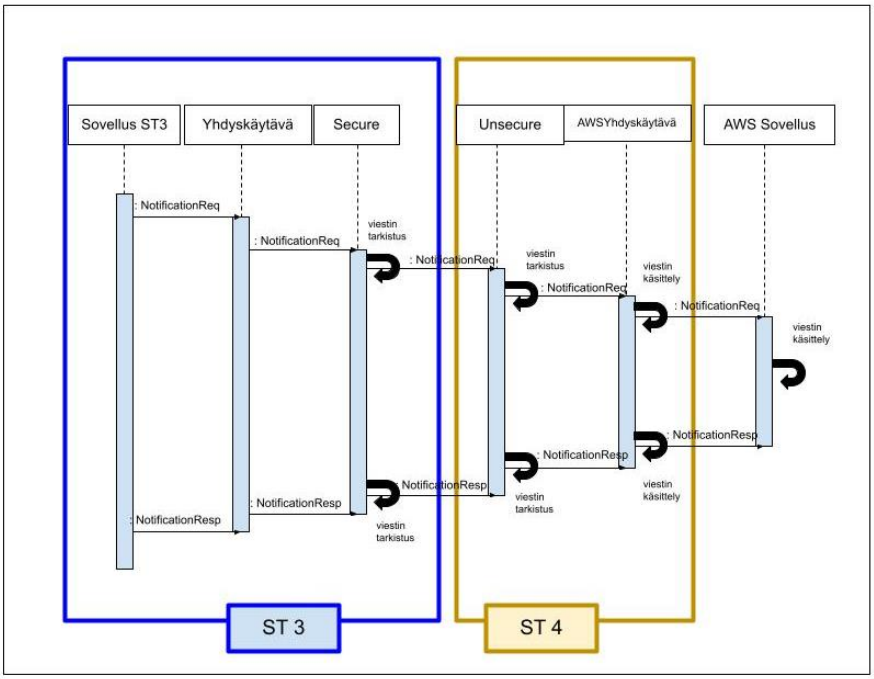
Sovelluksen tavoitteena on suodattaa tietoliikennettä järjestelmien välillä määriteltyyn tietojoukkoon perustuen. Toteutuksessa suodatus tehdään siten, että siirretystä informaatiosta tarkistetaan tiettyjä parametreja, joita verrataan määriteltyyn skeemaan ja vertailuun perustuen päätellään, kuuluuko siirrettävä tieto sovelluksen käsittelemään tietojoukkoon vai ei. Tämä tarkoittaa myös sitä, että suodatus on räätälöitävä erikseen eri toteutuksissa erityyppisille tietotyypeille käyttäen esimerkiksi luokkakirjastoja tai XML-skeemaa. Lisäksi suodatuksessa voidaan käyttää myös tiedon kokoon kantaa ottavia parametreja.

Opinnäytetyössä keskityttiin toteutukseen, jossa siirrettävä tieto ainoastaan suodatetaan, eikä informaation sisältöä muokata suodatuksen perusteella turvaluokitukseen sopivaksi. Yhdyskäytävään tuleva viesti joko menee läpi halutulle tasolle tai yhdyskäytävä estää virheellisen viestin pääsyn tasolle, jolloin se palauttaa lähteeseen virheviestin.

4.2.1 Spring Boot -sovellus

Yhdyskäytävätoteutus on Spring Boot -sovellus, joka tarjoaa rajapinnat, jotka vastaavat yhdyskäytävän yhdistämien sovellusten jo olemassa olevia rajapintoja. Tämä mahdollistaa sen, että yhdyskäytävätoteutus on näkymätön yhdistettäville turvaluokituksiltaan eroaville sovelluskokonaisuuden osille. Tavoitteena oli luoda sovellus, jota pystyttäisiin käyttämään kummankin turvaluokituksen ympäristöissä, eli monistamalla yhdyskäytäväsovellusta voitaisiin luoda ns. yhdyskäytäväputki, jossa kaksi yhdyskäytäväsovellusta toimii yhdessä. Vaikka kyseessä on kahden eri suojaustason kattava ratkaisu, voidaan kummallakin puolella silti käyttää samoja validointisääntöjä ja rajapintoja.

Kaikki yhdyskäytävään liittyvät toiminnot käynnistetään kuviossa 2 näkyvän ST3-turvaluokitetun sovelluksen toimesta, joka lähettää yhdyskäytävän lävitse pyynnön käytävän toisella puolella sijaitsevan sovelluksen rajapinnalle. Kuviossa on esimerkin vuoksi käytetty viestinä ilmoituksen lähetystä sovellusten välillä. Yhdyskäytävään tuleva tieto on muodoltaan JSON:ia, joka muutetaan Kotlin-luokiksi. Tämän muunnoksen aikana tietosisällön rakenne ja sisältö validoidaan määriteltyä skeemaa vasten. Muunnos JSON:ista Kotliniksi onnistuu vain oikeinmuotoisella JSON-tietosisällöllä. Validoinnin jälkeen Kotlin -luokka muutetaan jälleen takaisin JSON:iksi ja lähetetään edelleen rajapintaan. Kaikki viestintä yhdyskäytävien toimesta tapahtuu käyttäen HTTPS-protokollaa.



KUVIO 2. Sekvenssikaavio esimerkkiviestin kulusta

ST3-turvaluokitettu sovellus lähettää ajastetusti palomuurin lävitse viestin ensimmäiselle yhdyskäytävälle, joka tarkistaa viestin merkintää varten, tekee lokimerkinnän ja välittää viestin eteenpäin Secure -komponentille. Secure -komponentti tarkistaa, että viesti on skeeman muotoinen ja rakenne on oikea ja tekee lokimerkinnän validoinnista. Jos viestin validointi onnistuu, komponentti kutsuu ST4-suojaustasolla olevaa Unsecure -komponenttia. ST4-suojaustasolla Unsecure-komponentti tarkistaa jälleen saadun viestin rakenteen, validoi sen suojaustason määritellyn skeeman perusteella ja kirjaa lokimerkinnän viestin validoinnista. Validoinnin onnistuessa komponentti kutsuu saman suojaustason

yhdyskäytävää. AWSYhdyskäytävä kutsuu VPN:n läpi pilviympäristön sovelluksen rajapintaa, joka hakee viestistä halutut tiedot ja palauttaa sen vastauksena AWSYhdyskäytävälle, jolloin viesti lähtee kulkemaan takaisin kohti ST3-sovellusta. Unsecure- ja Secure- komponenttien välillä on myös palomuuritoteutus, jossa lisäsuojana HTTPS-protokollan tukena on myös IP-rajaus, jolloin vain tietystä IP-osoitteesta tulevat vastaukset sallitaan. Secure -komponentin validoima viesti välitetään edelleen ST3-yhdyskäytävään. Jos viestin validointi ei onnistu jossain tilanteessa eli viesti on virheellinen, tuntematon tai väärän rakenteinen, virheviesti palautetaan kutsujalle.

4.2.2 Viestien toteutus

Yhdyskäytävän viestien välityksessä käytettiin Apache Camel – viitekehystä. Sovelluksen toteutuksessa piti ottaa huomioon, että toteutuksen täytyi tukea kaikkia projektin kontekstissa olemassa olevia viestejä. Viestit voivat olla esimerkiksi tietopaketteja tai ilmoituksia. Viestintä on toteutettu käyttämällä HTTPS-protokolaa, jossa viestintä on salattu käyttäen TSL-protokolaa.

4.2.3 Lokitus

Viestien siirrosta tehdään lokikirjauksia, jotta tiedonsiirtoa ja sen toimintaa voidaan halutessa seurata. Viestin validoinnin onnistumisesta lokitetaan joka tapauksessa riippumassa siitä, oliko viesti oikeellinen vai poikkesiko se määritellystä skeemasta. Viestistä merkitään ylös sen ID ja validoinnin tulos, mutta viestin sisällöstä, turvaluokituksesta tai sen rakenteesta ei tehdä mitään merkintöjä. Virhetilanteessa merkitään ID:n lisäksi erikseen vikalokimerkintä.

5 JOHTOPÄÄTÖKSET JA POHDINTA

5.1 Analyysi toteutuksen turvallisuudesta

Toteutuksen tavoitteena oli saada aikaan yksinkertainen ja turvallinen yhdyskäytäväratkaisu, joita jo olemassa olleiden sovellusten rajapinnat tukisivat. Projektin turvaluokitukset huomioiden alun perin tavoitteena oli saada ST3-hyväksyntä ympäristölle valitulla yhdyskäytäväratkaisulla. Toteutuksessa pystyttiin käyttämään lähes identtistä yhdyskäytäväratkaisua sekä ST3-ympäristössä, että ST4 -ympäristössä.

Verraten ratkaisua Kyberturvallisuuskeskuksen antamaan ohjeistukseen turvallisen alkiotunnistuksen sisältösuodatukseen perustuvan yhdyskäytävän kriteereistä (Ohje yhdyskäytäväratkaisujen… 2018, 6) toteutettu ratkaisu täyttää ST3-kriteerit soveltuvin osin. Ohjeistuksen mainitsemat ehdot 1-3 täyttyvät ratkaisussa, kun yhdyskäytävän vastaanotetun tiedon rakenne, sisältö ja sovellustason sanomamuoto on etukäteen määritelty ja vain määritykseen sopiva tieto päästetään lävitse. Tietoalkiot voitaisiin halutessaan myös merkitä tunnistettaessa esimerkiksi tiettyyn suojaustasoon sopiviksi tai tiedon omistaja, salassapitoaika tai jakelu voitaisiin yhdistää tietoon. Tähän yhdyskäytäväratkaisuohjeen mainitsemaan tiedon merkintään ei tässä tapauksessa ollut tarvetta, koska vain oikeellinen määriteltyyn skeemaan sopiva tieto päästettiin lävitse, eikä edellä mainitut merkintätiedot tuoneet sovelluksen toimintaan mitään lisäarvoa.

Sovellus ei mahdollista väärän muotoisen JSON-tietosisällön muuntamista Kotlin-luokaksi tai päästä virheellisiä viestejä käytävässä eteenpäin. Se tekee myös epäonnistuneista validoinneista lokimerkinnän. Tällöin suodatuksen voidaan olettaa toimivan luotettavasti sekä oikean muotoisten, että virheellisten tietosisältöjen tapauksessa ja täyttävän näin ratkaisuohjeen ehdon 4 (Ohje yhdyskäytäväratkaisujen… 2018, 7).

Suodatus suoritetaan muusta sovelluskokonaisuudesta eriytetyssä sovelluskokonaisuudessa, joka koostuu Secure- ja Unsecure -komponenteista,

eikä se ole näkyvä ympäröiville sovelluksille. Toiminnallisuus on rajattu omille edustapalvelimilleen ja kokonaisuuksia rajaa IP- ja porttirajaukset, jossa suoritetaan myös suodatusta eli suodatusta suoritetaan sovellustason lisäksi myös verkkotasolla. Useammassa tasolla toteutettavalla, eriytetyssä toiminnallisuudessa toimivalla suodatuksella täytetään ratkaisuohjeen ehdot 5-6 (Ohje yhdyskäytäväratkaisujen… 2018, 7).

5.2 Vaihtoehtoiset toteutustavat

5.2.1 Yksisuuntaiset suodatusratkaisut

Yksisuuntaiset suodatusratkaisut rajaavat eri tavoin liikennöinnin yksisuuntaiseksi. Liikennöinnin rajaaminen voi tapahtua esimerkiksi fyysisellä OSI-mallin kerroksella. Tällöin puhutaan niin sanotusta datadiodiratkaisusta, joka mahdollistaa tiedonsiirron ainoastaan matalan tason ympäristöstä ylemmälle tasolle. Datadiodiratkaisussa täytyy myös toteuttaa informaation eheyden tarkistus. Datadiodi ratkaisu mahdollistaa toimimisen korkean turvaluokituksen ympäristöissä, mutta rajaa liikennöinnin yksisuuntaiseksi. Kyseistä ratkaisua voidaan käyttää myös useamman kuin yhden suojaustason läpi toteutettavassa yhdyskäytävässä.

Datadiodin lisäksi yksisuuntaisiin suodatusratkaisuihin lukeutuvat myös esimerkiksi ajastettujen palomuurisääntöjen avulla eristetty yhteys, jossa suojaustasojen välille luodaan eristetty vyöhyke, jossa liikennöinti sallitaan rajatussa aikaikkunassa yhteen suuntaan kerrallaan ja vain tunnistetut tietotyypit sallitaan yhteyden läpi. Kyseinen ratkaisu tuo lisäsuojaa eriyttämällä yhdyskäytävän erikseen hallinnoiduksi vyöhykkeeksi, jos riskinä on, että hyökkääjä pääsee käsiksi alemmalle tai ylemmälle suojaustasolle. Ratkaisussa pitää kuitenkin varmistaa, että yhteys katkeaa aina kun sallittu aikaikkuna umpeutuu. Palomuuriratkaisu on toteutettu OSI-mallin verkko- ja sovelluskerroksilla ja sitä voidaan käyttää ainoastaan yhden suojaustason ylittävissä yhdyskäytäväratkaisuissa.

Alun perin toteutettavan yhdyskäytävän ratkaisumallia valittaessa, mietittiin mahdollisuutta käyttää datadiodin mallia suodatukseen. Datadiodin kyvykkyyksien epävarmuuden ja muutosherkkyydestä johtuvien ylläpitovaatimusten vuoksi datadiodia ei nähty optimaaliseksi vaihtoehdoksi tähän toteutukseen.

5.2.2 Muut yhdyskäytäväratkaisumallit

Yhdyskäytävätoteutukselle on yleisesti olemassa monia muita ratkaisumalleja. Turvallisuusvaatimusten luonteen vuoksi toteutuksessa ei kuitenkaan pidetty tarpeellisena tutkia useampaa toteutustapaa, koska niiden tunnettujen heikkouksien vuoksi nämä ratkaisumallit eivät ole Kyberturvallisuuskeskuksen yleisesti hyväksymiä (Ohje yhdyskäytäväratkaisujen… 2018, 13). Kyseiset yhdyskäytävätoteutukset voidaan hyväksyä tilanteesta riippuen, jos välitettävien tietojen omistaja on tyytyväinen saavutettavaan tietoturvan tasoon riskianalyysien perusteella.

Poisluettuihin ratkaisuihin lukeutuvat muun muassa liikennevuon sisältöratkaisut, virtualisointiin perustuvat ratkaisut ja KVM-ratkaisumallit. Esimerkiksi KVM-ratkaisussa liikennöinti on yleensä rajattu kytkimen avulla näppäimistön, näytön ja hiiren toiminnallisuuksiin, jossa rajaus on toteutettu joko ohjelmistopohjaisesti tai mekaanisesti. KVM-ratkaisun turvallisuuden perustuessa tuotekohtaisuuteen, kyseinen malli ei ollut edes käyttökelpoinen projektin kontekstissa. Myös poisluetut ohut- ja monitasopääteratkaisut rajaavat sovelluskohteet samalla laitteella tapahtuvaan eri suojaustasojen ympäristöjen käyttöön, joka ei myöskään ollut validi ratkaisumalli toteutukselle.

5.3 Pohdinta

Opinnäytetyön tulos ei ole ristiriidassa projektissa toteutetun yhdyskäytäväratkaisuanalyysin kanssa, vaan työssä päädyttiin tukemaan valittua alkiotunnistukseen perustuvaa sisällönsuodatusratkaisua. Tutkimustyön tuloksena ei löydetty parempaa vaihtoehtoista toteutustapaa, joka olisi täyttänyt

projektin vaatimukset suojaustasojen ja muiden rajoitteiden suhteen. Datadioditoteutuksen lisäksi muut yhdyskäytäväratkaisut olivat helposti rajattavissa pois mahdollisista toteutustavoista projektin asettamien rajausten perusteella.

Opinnäytetyön tuloksena yhdyskäytäväratkaisuiden vertailuiden ja mahdollisten aikaisempien suunniteltujen toteutusten dokumentaatiosta pääosat on koottu yhteen lähteeseen, aikaisemman pirstalaisen dokumentaation sijaan. Pirstaleinen ja epäloogisesti arkistoitu dokumentaatio teki tutkimuksesta myös aika ajoin vaivalloista, koska ristiriitaisten dokumentaatioiden eroavaisuudet nojasivat paljolti muutaman ihmisen tietoon ja näin ollen uutena asiaan perehtyvä henkilö ei pysty pelkästään dokumentaatioon perustuen perehtymään kattavasti valitun toteutuksen perusteisiin tai toteutuksen aikajanaan.

Opinnäytetyössä katettu sovellustoteutus on perusperiaatteiltaan yksinkertainen suodatusratkaisu, joka helpottaa toteutuksen ylläpitoa ja mahdollisia tulevaisuuden muutostarpeita.

LÄHTEET

Comparison Between Spring and Spring Boot. 24.3.2021. Luettu: 30.5.2021. https://www.baeldung.com/spring-vs-spring-boot

Introduction To Apache Camel. 25.3.2020. Luettu 31.5.2021. https://www.baeldung.com/apache-camel-intro

Katakri. Kansallinen turvallisuusauditointikriteeristö. 2011. Helsinki: Puolustusministeriö.

Katakri. Tietoturvallisuuden auditointityökalu viranomaisille. 2015. Helsinki. Puolustusministeriö.

Kotlin Programming Language. 2021. Luettu: 21.5.2021. https://github.com/JetBrains/kotlin

Ohje yhdyskäytäväratkaisujen suunnitteluperiaatteista ja ratkaisumalleista. 2018. Kyberturvallisuuskeskus. Helsinki: Viestintävirasto.

PiTuKri. Pilvipalveluiden turvallisuuden arviointikriteeristö. 2020. Traficom. Liikenne- ja viestintävirasto.

Spring. n.d. Spring Boot. Luettu: 21.5.2021. https://spring.io/projects/spring-boot

Spring Boot. 2021. Luettu: 21.5.2021. https://github.com/spring-projects/spring-boot

What is Camel? n.d. luettu 31.5.2021 https://camel.apache.org/manual/latest/faq/what-is-camel.html