Opinnäytetyö (AMK)

Tieto- ja viestintätekniikka

2025

Rafael Ackalin

Teollisuusjärjestelmän virtuaalinen harjoitusympäristö

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

Opinnäytetyö (AMK) | Tiivistelmä

Turun ammattikorkeakoulu

Tieto- ja viestintätekniikka

2025 | 27 sivua

Rafael Ackalin

Teollisuusjärjestelmän virtuaalinen harjoitusympäristö

Kyberturvallisuusuhat teollisuusverkkoja vastaan ovat nousseet viime vuosina. Etenkin valtiolliset toimijat ovat kiinnostuneita teollisten prosessien haavoittuvuuksien hyödyntämisestä. Teollisuusverkkojen komponentit ja protokollat ovat usein vuosikymmeniä vanhoja, eikä kyberturvallisuus ole ollut prioriteetti niiden kehityksessä. Vanhojen järjestelmien päivittäminen tai korvaaminen on haastavaa. Pelkästään järjestelmien testaaminen voi olla hankalaa toteuttaa.

Työn tavoitteena oli toteuttaa kevyt virtuaalinen tehdasympäristö, jossa on mahdollista testata Modbus-protokollan haavoittuvuuksia. Työ tehtiin, jotta Turun ammattikorkeakoulun Cybersecurity for Industrial Networks -opintojaksolla olisi harjoitusympäristö testauksia varten. Ympäristön virtuaalisointiin käytettiin Docker-konttiteknologiaa ja avoimen lähdekoodin projekteja SCADASim- ja ScadaBR-tehtaan simulointiin.

Tuloksena saatiin virtuaalinen ympäristö, jossa on mahdollista luoda oma tehdasskenaario testauksia varten. Ympäristön avulla voi simuloida yleisellä tasolla komponentteja ja niiden yhteyksiä sekä valvoa ja hallita niitä käyttöliittymässä.

Asiasanat:

kyberturvallisuus, ICS, konttiteknologia, digitaalinen kaksonen, teollisuusverkko

Bachelor’s | Abstract

Turku University of Applied Sciences

Information and Communications Technology

2025 | 27 pages

Rafael Ackalin

Virtual training environment for industrial systems

Cybersecurity threats against industrial networks have been increasing in recent years. Nation-state actors especially are interested in exploiting vulnerabilities found in industrial processes. The components and protocols used in these networks are often decades old and were not created with cybersecurity in mind. Replacing or updating old systems is difficult. Just doing tests on these systems can be hard to do.

The goal of this thesis was to implement a resource-light virtual factory environment to test vulnerabilities in the Modbus-protocol. The thesis was made to create a testing environment for Turku University of Applied Sciences Cybersecurity for Industrial Networks course. Docker container technology was used to virtualize this environment. Two open-source projects called SCADASim and ScadaBR were used to simulate a factory.

The result is a virtual environment where it is possible to create your own factory scenario for testing purposes. The environment can be used to simulate components on a generic level and to monitor and control them with an interface.

Keywords:

cybersecurity, ICS, containerization, digital twin, industrial network

Sisällys

Kuvat

Lyhenteet

HMI	Human Machine Interface. Käyttöliittymä, jolla ohjataan ja valvotaan koneita tai prosesseja.
ICS	Industrial Control System. Teollinen ohjausjärjestelmä. Laitteita ja ohjelmistoa, jolla ohjataan teollisia prosesseja.
OT	Operatiivinen teknologia. Laitteita ja ohjelmistoa, jota käytettään fyysisten prosessien hallintaan.
PDU	Protocol Data Unit. Yhteiskäytännön datayksikkö, tietoliikenteessä käytettävä datayksikkö tiedon siirtämiseen.
PLC	Programmable Logic Controller. Ohjelmoitava logiikka, laite, joka ohjaa toimilaitteita.
SCADA	Supervisory Control and Data Acquisition. Järjestelmä, jolla valvotaan ja ohjataan tehtaan laitteita.

1 Johdanto

Viime vuosina kyberturvallisuusuhat ovat kasvaneet teollisuushallintajärjestelmiä kohtaan. Valtiolliset toimijat ovat kiinnostuneita hyödyntämään näiden järjestelmien haavoittuvuuksia omien intressiensä edistämiseksi. Onnistuneet kyberhyökkäykset teollisuusjärjestelmiä kohtaan voivat aiheuttaa vakavia vaikutuksia yhteiskunnallisella tasolla. Valtiollisia toimijoita kiinnostavat erityisesti infrastruktuurijärjestelmät. Vuoden 2015 kyberhyökkäys ukrainalaiseen sähköverkkoon vaikutti sähkökatkoksilla sähköverkon asiakkaisiin (CISA, 2021). Hyökkäyksien uhka on kasvanut nykyisen geopoliittisen tilanteiden seurauksena (ENISA, 2024 s. 10).

Tämän opinnäytetyön tarkoituksena oli luoda virtuaalinen ympäristö teollisuuden hallintajärjestelmien kyberturvallisuuden testausta ja havainnointia varten käyttäen Docker-konttiteknologiaa. Opinnäytetyössä käytettiin kahta avoimen lähdekoodin projektia simuloimaan tehdasympäristöä. Konttiteknologiaa hyödyntämällä simulointi ei vaadi paljon resursseja, ja se voidaan eristää päälaitteesta.

Työssä pyrittiin ratkaisemaan Turun ammattikorkeakoulun Cybersecurity for Industrial Networks -opintojakson harjoitusympäristön puute toteuttamalla kevyt virtualisoitu tehdasympäristö. Opintojaksolla testataan teollisuusverkkojen toiminnallisuutta, teollisuusprotokollien heikkouksia ja analysoidaan ohjelmoitavan logiikan eli PLC:n lähettämää telemetriadataa. Opintojaksolla ei ole saatavilla fyysistä tehdasympäristöä. Tarvittavat laitteet ovat kalliita ja niitä tarvittaisiin useita luokkaympäristöön. Fyysisen testausympäristön luomiseen on varattava myös tila, johon laitteet voitaisiin asentaa.

Tavoitteena oli tehdä resurssikevyt virtuaalinen ympäristö, jossa voitaisiin testata teollisuusverkkojen ja laitteiden haavoittuvuuksia helposti ja nopeasti. Ympäristön tuli sisältää useita valmiita skenaarioita, mutta myös mahdollisuuden luoda uusia.

Opinnäytetyön luvussa 2 käsitellään teollisuusverkkojen kyberturvallisuushaasteita ja pohditaan, miksi vanhentuneiden laitteistojen korvaaminen on haastavaa. Luvuissa 3 ja 4 esitetään työssä käytetyt teknologiat ja ohjelmat, niiden ominaisuudet ja toiminta. Luvussa 5 käydään läpi Docker-konttien ja harjoitusympäristön asennukset ja testaaminen.

2 Kyberturvallisuus haasteet teollisuusverkoissa

Teollisuusverkot ovat keskeinen osa operatiivista teknologiaa (OT). OT kattaa ohjelmoitavia järjestelmiä ja laitteita, jotka ovat vuorovaikutuksessa fyysisen ympäristön kanssa. OT-järjestelmillä on erilaiset riskit ja prioriteetit IT-järjestelmiin verrattuna. OT-järjestelmien riskit voivat vaikuttaa vakavasti ihmisten terveyteen ja turvallisuuteen sekä aiheuttaa ympäristövahinkoa. OT-järjestelmät vaativat eri suorituskykyä ja luotettavuutta kuin tyypilliset IT-verkot. OT-järjestelmät ovat yleisesti aikakriittisiä ja hyväksyvät vain tietyn tasoisia viiveitä ja viivevaihtelua. (Stouffer ym., 2023 ss. 8, 28–29.)

2.1 Perinnejärjestelmien haasteet

Teollisuusympäristöjen komponenttien päivittäminen on usein haastavaa, sillä tehtaan prosesseja ei voida pysäyttää väliaikaisesti. Tuotannon lopettaminen tuo tehtaalle menetyksiä eikä palveluja tai tuotteita voida tuottaa. OT-järjestelmien elinkaari voi ylittää 20 vuotta. Näitä järjestelmiä kutsutaan perinnejärjestelmiksi. Perinnejärjestelmät saattavat käyttää laitteistoa ja ohjelmistoa, joita ei enää tueta. Järjestelmien päivittäminen ei ole mahdollista, jos tuotteen myyjä ei enää tue tuotetta. (Stouffer ym., 2023 s. 37.) Vanha järjestelmä voi esimerkiksi käyttää ohjelmaa, joka on yhteensopiva vain Windows XP -käyttöjärjestelmässä. Uudempien käyttöjärjestelmien tietoturvallisuuspäivitysten hyödyntäminen vaatisi ohjelman uudelleenkirjoittamisen.

Uusien laitteiden sekä teknologian kehittäminen tai hankkiminen ja tuotannon lopettaminen tulee kalliiksi. Vanhentuneen järjestelmän pitäminen on edullisempaa ja varmempaa kuin uuden hankinta. Suuret muutokset uusista laitteista saattavat luoda lisää negatiivisia vaikutuksia kyberturvallisuudelle (Flaus, 2019 s. 22). Esimerkiksi uudet langattomat IT-laitteet ja yleisesti uusien IT-teknologioiden käyttöönotto, kuten etäyhteysmahdollisuudet, vähentävät OT-laitteiden eristyneisyyttä ulkomaailmaan (Stouffer ym., 2023 s. 28). Näiden

haavoittuvuuksien testaaminen on vaikeaa fyysisillä laitteilla oikeassa ympäristössä.

2.2 Käytettävyysvaatimukset estävät päivityksiä

Laitteiden päivityksissä tulee ottaa huomioon teollisuusjärjestelmien käytettävyysvaatimukset. Järjestelmät ovat yleisesti jatkuvasti päällä ilman katkoja. Saatavuusvaatimukset estävät laitteiden uudelleenkäynnistämisen ja kaikki katkokset pitää suunnitella ja ajoittaa päiviä tai viikkoja etukäteen. Käyttöjärjestelmien päivittäminen turvallisuuskorjauksia varten vaatii suunnittelua ja testaamista ennen implementointia. (Stouffer ym., 2023 ss. 29–30.)

Koska käyttöjärjestelmä- tai ohjelmapäivitykset eivät välttämättä ole yhteensopivia perinnejärjestelmien kanssa, on oltava varma päivitysten toiminnallisuudesta. Päivityksen testaaminen ja suunnittelu on mahdollista, jos identtisiä järjestelmiä on useampi ja eikä niillä ole samoja saatavuusvaatimuksia tai ne eivät ole käytössä. Varalaitteita ei mahdollisesti ole saatavilla kustannusten takia, joten riski pitkään kestävään katkokseen on olemassa. Vanhojen ohjelmien sekä vanhentuneiden käyttöjärjestelmien käyttöä pidetään hyväksyttävänä, jotta tuotantokatkoksia voidaan välttää.

Perinnejärjestelmillä on pieni määrä resursseja, koska ne on valmistettu vuosikymmeniä sitten. Perinnejärjestelmiltä puuttuvat esimerkiksi salaus, virhelokikirjanpito ja salasanan suojaus. Pieni laskentateho voi estää näiden ominaisuuksien jälkiasentamisen. Turvallisuusominaisuudet saattavat myös häiritä järjestelmien saatavuutta ja ajoituksia. (Stouffer ym., 2023 s. 29.)

2.3 Teollisuusjärjestelmien ohjausprotokollat

Alussa OT-järjestelmät käyttivät suljettuja ohjausprotokollia ja olivat eristettyjä. IT-ratkaisujen käyttöönoton yleistyessä järjestelmät ovat vähemmän eristettyjä ulkopuolelta. (Stouffer ym., 2023 s. 28.) Monet käytetyistä protokollista on

kehitetty ilman turvamenetelmiä, kuten salausta tai todennusta. Näille ei ollut tarvetta, kun järjestelmät olivat eristettyjä.

ICS eli teollisuusohjausjärjestelmät käyttävät useita eri protokollia, joista jotkin ovat kehitetty tiettyyn käyttötarkoitukseen. Protokollan käyttöönottoon vaikuttaa, mille teollisuusalueelle protokolla tulee käyttöön. Esimerkiksi kemikaaliteollisuudella on eri vaatimukset ja tarpeet kuin sähköjakelujärjestelmillä. (Flaus, 2019 ss. 43–44.)

2.3.1 Modbus-protokolla

Tehdasverkoissa yleisesti käytetään Modbus/TCP-protokollaa. Modbus on asiakas/palvelin-protokolla, jossa yksi laitteista on asiakas, joka ohjaa ja kommunikoi asiakaslaitteen kanssa. Palvelin vastaa asiakaslaitteen luku- ja kirjoituspyyntöihin. (Flaus, 2019 s. 46.)

Protokolla on turvaton, koska Modbus-protokollan paketteja on helppo muokata ja lukea. Esimerkiksi palvelunestohyökkäys on helppo toteuttaa, koska joitain Modbus-komentoja on mahdollista lähettää yleislähetystilassa. (Flaus, 2019 s. 46.) Komennot lähtevät kaikille vastaanottajille, mikä ylikuormittaa saatavilla olevia resursseja tai mahdollisesti kaataa prosesseja.

2.3.2 Profibus-väylä ja Profinet-standardi

Profibus-väylä on kenttäväylästandardi, jota käytetään sarjamuotoiseen kommunikaatioon sarjakaapelin tai valokuidun kautta. Profibus-väylä on asiakas-palvelinprotokolla, joka on suunniteltu liittämään kenttälaitteita PLC:hen. Profibus-väylä ei ole haavoittuvainen TCP/IP-hyökkäyksille. (Flaus, 2019 s. 46.)

Profinet on Profibus-väylään perustuva standardi, joka käyttää Ethernetiä kommunikaatioon sarjakaapelin sijaan. Profinet käyttää tunnistukseen perustuvaa toistojärjestelmä reaaliaikaiseen toiminnallisuuteen sekä protokollan

muokkaamiseen lyhyellä aikavälillä. Protokollaa on mahdollista käyttää langattomissa verkoissa käyttämällä Profinetin TCP/IP-toiminnallisuutta. Järjestelmät, jotka käyttävät Profinetiä, ovat tarkoitettu luotettavaan ja reaaliaikaiseen kommunikaatioon. Profinet protokolla pyrkii takamaan järjestelmien saatavuuteen ja luotettavuuteen, protokollalla ei ole turvaominaisuuksia ja se on eristettävä verkosta estääkseen hyökkäyksiä. (Flaus, 2019 ss. 46–47.)

2.4 SCADA-järjestelmät

Laitteita ja niiden prosesseja ohjataan ja valvotaan SCADA-valvomo-ohjelmistolla. SCADA-järjestelmillä ohjataan maantieteellisesti hajautettuja laitteita, joiden data keskitetään SCADA:lla. SCADA-järjestelmät yhdistävät tiedonhankintalaitteet tiedonsiirtolaitteilla sekä käyttöliittymäohjelmiston tarjotakseen keskitetyn valvonta- ja hallintajärjestelmän. (Stouffer ym., 2023 s. 12.)

SCADA-järjestelmään kuuluvat työasemat, jotka käyttävät erikoistunutta ohjelmistoa graafisella käyttöliittymällä. Työasemat ovat verkon kautta suoraan yhdistetty fyysisiin laitteisiin, kuten PLC:hin, syöttö- tai tulostuskomponentteihin sekä etäkäyttöliittymiin. (Flaus, 2019 s. 5.)

2.5 ICS-komponentit

2.5.1 PLC-laite

PLC on komponentti, joka tarjoaa automaatiojärjestelmälle olennaisia toimintoja. PLC:llä on mahdollista muokata, aktivoida tai moduloida fyysisiä toimintoja siihen ohjelmoidun logiikan mukaan. PLC:n on operoitava reaaliajassa muiden fyysisten järjestelmien rinnalla. Viiveet tuovat ei-toivottuja tuloksia prosessissa. Yksinkertaisesti PLC on sisään- ja ulostulolaite, joka lukee sisään tulevaa dataa, ajaa siihen kirjoitetun logiikkaohjelman saaduilla arvoilla

ja tulostaa antureilta saatuja arvoja sekä lähettää komentoja toimilaitteille. (Flaus, 2019 ss. 7–8, 10.)

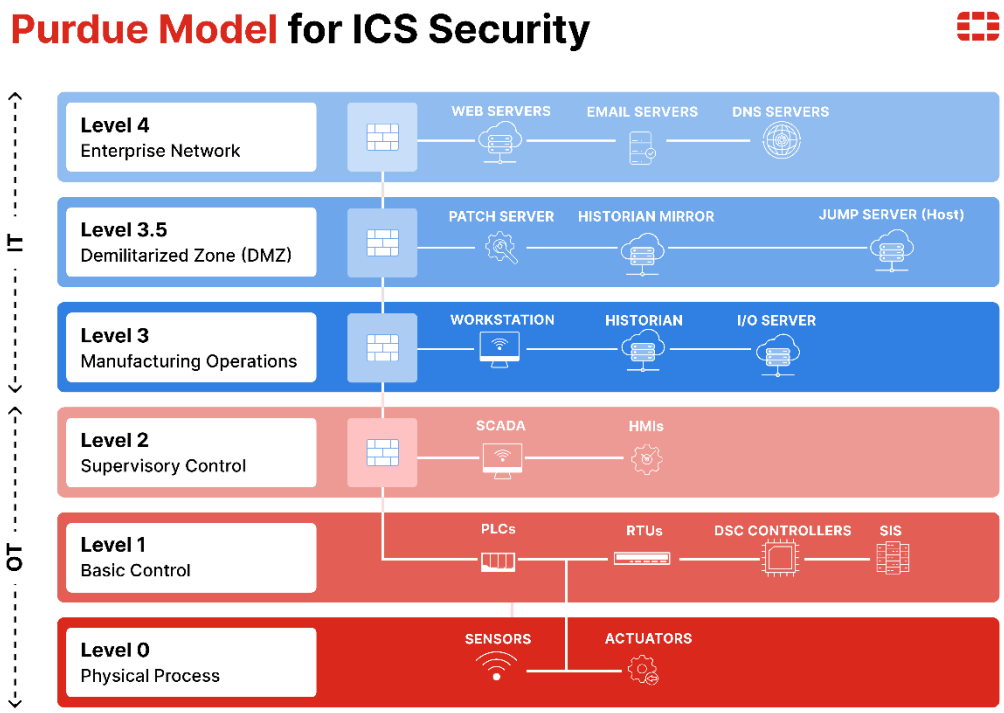
2.5.2 Käyttöliittymä

HMI eli käyttöliittymä on ohjelma, jolla visualisoidaan järjestelmiä niiden valvontaa ja ohjausta varten. HMI voi olla työasema, jossa on useita näyttöjä, joilla näkyy järjestelmien nykyinen tila tai aiempaa dataa (Stouffer;ym., 2023 s. 10). HMI voi myös olla sulautettu järjestelmä, joka toimii usein fyysisen järjestelmän lähellä (Flaus, 2019 s. 15).

2.5.3 Kenttälaitteet

Kenttälaitteita ovat esimerkiksi anturit ja toimilaitteet. Anturit ovat laitteita, jotka mittaavat fyysisiä ominaisuuksia ja lähettävät tiedot PLC:lle. Toimilaitteita ovat esimerkiksi venttiilit, kytkimet ja moottorit. Toimilaitteita käytetään prosessien operoimiseen saatujen komentojen perusteella. (Stouffer ym., 2023 s. 10.)

Kuvassa 1 on purdue mallilla segmentoitu ICS-verkko. Mallissa on merkitty mihin kenttälaitteet kuten anturit ja toimilaitteet sijoittuvat. Kuvassa näkyy myös muiden ICS-komponenttien segmentit sekä niiden väliset yhteydet.



Kuva 1. Purdue malli ICS-verkolle (Fortinet Inc., n.d.)

2.6 Kybertestaaminen

Kybertestaus alkaa yleisesti tutkimalla järjestelmän dokumentaatiota. Testattavien laitteiden ja järjestelmien tuntemus on tärkeää suunnittelu- ja toteutusvaiheessa. Tavoitteena on yleensä haavoittuvuuksien löytäminen ja niiden penetraatiotestaaminen. Haavoittuvuustestaamisella tarkoitetaan kaikkien potentiaalisten heikkouksien etsimistä järjestelmästä. Penetraatiotestaaminen on löydettyjen haavoittuvuuksien hyödyntämistä. (Mowbray, 2014 ss. 139–140.)

Haavoittuvuuksien etsimisessä skannataan verkosta löytyviä laitteita ja portteja. Skannaukset luovat verkkoon liikennettä ja ovat suoraan vuorovaikutuksessa

verkossa olevien laitteiden kanssa. ICS-verkkojen skannauksessa pitää ottaa huomioon, että ICS-laitteet ovat erittäin herkkiä tällaiselle verkkoliikenteelle. Skannaaminen voi aiheuttaa epävakautta tai vaikuttaa laitteen prosessitilaan. Perinnejärjestelmä voi esimerkiksi kaatua skannaamisen aikana, mikä pysäyttää prosessin. Tästä syystä ICS-laitteiden kyberturvallisuustestaukset tulisi tehdä suunniteltujen katkojen aikana, jos niitä halutaan tehdä oikeassa ympäristössä. (Stouffer ym., 2023 s. 209.)

2.7 Teollisuusverkkojen virtuaalisointi

Virtuaalisointi auttaa testaamaan teollisuusverkkoja ilman siihen vaadittuja laitteita. Virtuaalinen testaaminen ei tarvitse samoja resursseja kuin fyysinen testaus. Kaikki verkossa olevat laitteet voidaan sammuttaa tai testata tavalla, joka voisi aiheuttaa vahinkoa fyysisessä maailmassa.

Eristetyssä ympäristössä voi luoda turvallisesti skenaarioita, joissa tapahtuu vahinkoa laitteille ja teoriassa myös henkilövahinkoja. Digitaalinen ympäristö ei myöskään aiheuta minkään oikean prosessin pysäyttämistä. Verkon eri protokollia on helpompi seurata ja testata eri korjausmenetelmiä nopeasti.

3 Harjoitusympäristön teknologiat

3.1 Konttiteknologia

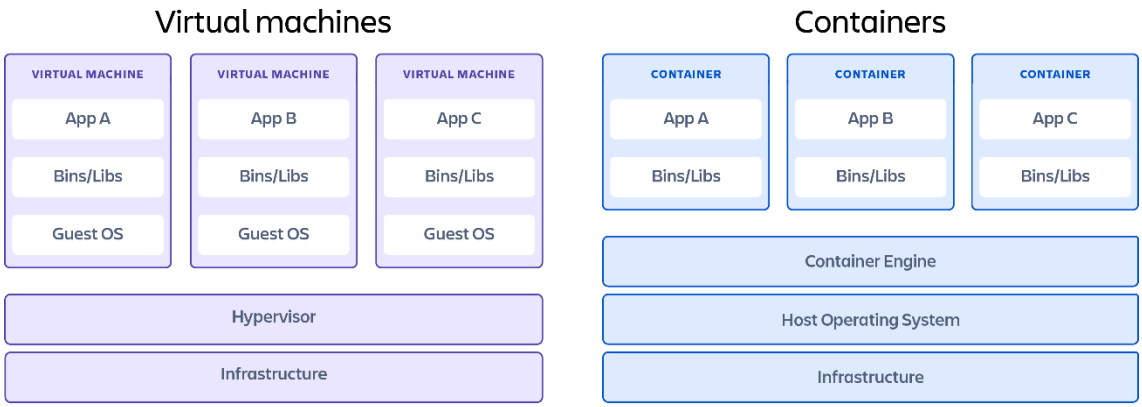
Kontit ovat virtualisointiteknologiaa, jota käytetään ohjelmien luomiseen, testaamiseen ja käyttöönottoon. Kontit paketoivat koodin, kirjastot, kehysympäristön ja muut riippuvuudet eristettyyn konttiin. Kontteja on helppo siirtää ja ajaa eri ympäristöissä ja infrastruktuureissa, koska ne ovat itsenäisiä niistä. (Red Hat, 2021.)

Konttiteknologia on yleisessä käytössä pilvipalveluissa erityisesti DevOps kehitys- ja tuotantotarkoituksiin. Amazon Web Services ja Google Cloud Platform esimerkiksi hyödyntävät konttiteknologiaa palveluissaan. (Muli, 2018 s. 4.)

Koska kontit pitävät kaikki riippuvuudet sisällään, konttien sisällä olevaa ohjelmaa ei tarvitse konfiguroida uudelleen ympäristön vaihtuessa. Konttien siirrettävyys ja pieni koko sopii esimerkiksi DevOpsiin ja mikropalveluihin. (Susnjara ym., 2024.)

3.2 Konttien ja virtuaalikoneiden erot

Pääero konttien ja virtuaalikoneiden välillä on käyttöjärjestelmän ytimen käyttö. Kontit toimivat jaetun käyttöjärjestelmän ytimen päällä. Jokainen virtuaalikone ajaa omaa ydintä. Globaalin ytimen resurssien jako mahdollistaa usean kontin ajamisen yhteisen ytimen päällä samalla käyttäen vähemmän prosessori-, muisti- ja verkkoresursseja. (Watada ym., 2019 s. 2.) Kuvan 2 vasemmalla puolella on kolme virtuaalikonetta, joista jokainen ajaa omaa käyttöjärjestelmää. Oikealla puolella kolme konttia jakaa isäntäkoneen käyttöjärjestelmää.



Kuva 2. Virtuaalikoneen ja kontin ero virtuaalisoinnissa (Buchanan, n.d.).

Käyttöjärjestelmän ytimen jakaminen toimii myös, kun isäntäkone käyttää esimerkiksi Windows-käyttöjärjestelmää, mutta ajaa Linux-kontteja. Kun Linux-kontteja pyöritetään Windows-isäntäkoneella, Dockeri käyttää WSL2:ta ajaakseen kevyttä Linux-virtuaalikonetta, jonka ydintä kontit jakavat.

3.3 Modbus TCP/IP

Modbus on OSI-mallin sovelluskerroksen protokolla, joka toimittaa asiakas-palvelin-kommunikaatiota laitteiden välillä. Modbussille on varattu portti 502. Modbus on pyyntö-vastausprotokolla ja tarjoaa palveluita, jotka ovat määritettyjä toimintakoodeilla. Toimintakoodit ovat pyyntövastaus PDU:n eli yhteiskäytännön datayksikön elementtejä. Toimintakoodeja ovat esimerkiksi 'Read Coils', joka lukee laitteen käämien tilan, joka voi olla päällä tai pois. 'Write Coil' sen sijaan kirjoittaa päällä ja pois -pyynnön laitteelle. (The Modbus Organization, 2012 ss. 2, 11.)

Toimintakoodeja voidaan käyttää esimerkiksi nestesäilön nestepinnan tason lukemiseen. 'Read Holding Register' -toiminto lukisi nykyisen nestepinnan tason. Jos taso ylittää määritetyn rajan, 'Write Coil' -toiminnolla voidaan lähettää signaali sulkea venttiili. Kun nestepinnan taso on laskenut, venttiili avataan.

4 Harjoitusympäristön ohjelmistot

Opinnäytetyössä käytettiin neljää eri ohjelmaa: Docker Desktop virtualisointiin, SCADASim laitteiden simulointiin, ScadaBR käyttöliittymänä ja pyModbusTCP haavoittuvuustestaukseen.

4.1 Docker Desktop -sovellus

Docker Desktop on sovellus, jonka voi asentaa Mac-, Linux- tai Windows-ympäristöihin. Docker Desktop on käyttöliittymä, jolla voit hallita ja luoda Docker-kontteja sekä niissä ajettuja sovelluksia. Dockeria voi käyttää myös komentokehotteella. (Docker Inc., n.d.)

Docker tarjoaa useita työkaluja eri tarkoituksiin. Docker Compose yksinkertaistaa monikontti ohjelman hallitsemista. Docker Kubernetes yksinkertaistaa paikallisten konttien käyttöönottoa, skaalausta, testaamista ja hallintaa. Docker Volume tarjoaa konttien data hallintaa ja datan jakamista konttien kesken. (Docker Inc., n.d.)

Konttiteknologian tarkoitus on luoda eristetty ympäristö, jossa voi vapaasti suorittaa sovelluksia. Tämä toteutustapa valittiin, koska kontit ovat kevyt, nopea ja luotettava tapa kehittää ja julkaista sovelluksia. Kontit ovat sopiva tapa luoda eri simuloiduille ICS-laitteille oma eristetty ympäristö.

4.2 SCADASim-simulaattori

SCADASim on Software Engineering Instituution luoma avoimen lähdekoodin SCADA-simulaattori, jolla voi testata tehdas- tai infrastruktuuriympäristöjä. SCADA on ohjelmistojärjestelmä, jolla valvotaan ja ohjataan eri prosesseja ja laitteita (Flaus, 2019 s. 4).

Opinnäytetyössä käytettiin GitHub-käyttäjän Kamran Hasanin haaroittamaa SCADASimiä. Alkuperäinen SCADASim on kirjoitettu Python2:lla, joka voi olla

haastava saada toimimaan. Hasanin koodi on kirjoitettu Python3-versiolla, joten projektin asennus on helpompaa. (Hasan, 2021.)

4.3 ScadaBR-järjestelmä

ScadaBR on avoimen lähdekoodin SCADA-järjestelmä, jolla luodaan HMI:tä eli käyttöliittymiä. ScadaBR kommunikoi eri PLC- eli ohjelmoitavien logiikkalaitteiden kanssa. Tässä työssä käytettiin GitHub-käyttäjän Thiago Alvesin julkaistua ScadaBR-asentajaa (Alves, 2021.).

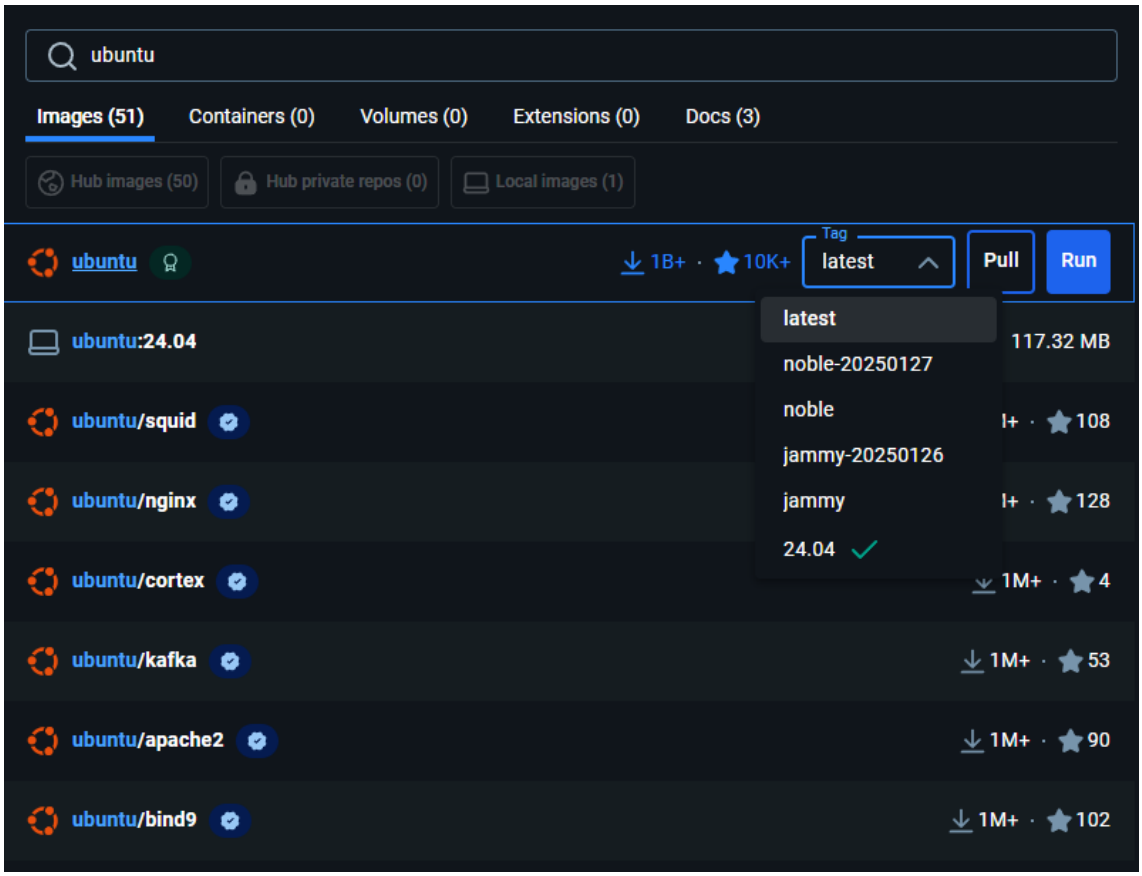
ScadaBR:n käyttöliittymän avulla siihen yhdistettyjä PLC-laitteita pystyy monitoroimaan ja hallitsemaan. Käyttöliittymässä voi luoda loogisen topologian graafisessa näkymässä. ScadaBR:n oma selainkäyttöliittymä on yksinkertainen ja nopeasti opittavissa.

4.4 pyModbusTCP-kirjasto

pyModbusTCP on Pythonissa kirjoitettu Modbus TCP -protokollakirjasto. Sen avulla voi suorittaa yksinkertaisia haavoittuvuustestejä tekemällä luku- tai kirjoituspyyntöjä PLC-laitteelle. pyModbusTCP:tä voi myös hyödyntää lokikirjanpito- tai vianetsintätarkoituksiin.

5 Tuotos

Docker-kontit käyttävät Ubuntu 24.04 -käyttöjärjestelmäkuvaa. Dockerissa on mahdollista käyttää eri käyttöjärjestelmiä, joista monet on luotu erikoistarkoituksiin. Käyttöjärjestelmistä on myös mahdollista käyttää vanhempia versioita, jos ne ovat saatavilla. Kuvassa 3 on eri Ubuntu-kuvia, jotka ovat saatavilla Docker Desktopissa. Kuvassa eivät näy kaikki saatavilla olevat Ubuntu-kuvat.



Kuva 3. Docker-kuvat haulla 'ubuntu'. Hakutuloksena on eri Ubuntu-kuvia, joista ensimmäinen on perus-Ubuntu-kuva ja muut ovat erikoissovellustarkoituksiin tarkoitettuja.

Tavoitteena oli, että SCADASim-kontti simuloi PLC-laitteita ja ScadaBR-käyttöliittymässä voi valvoa ja ohjata niitä. Lisäksi pyModbusTCP:llä pyriittiin

lähettämään Modbus-toimintakoodeja SCADASim-konttiin ja näkemään vaikutukset käyttöliittymässä.

Työssä ei käytty Dockerfile- tai Docker Compose -ominaisuuksia konttien luomiseen. Ohjelmat ja niiden riippuvuudet asennettiin kontin sisällä, ja valmis kontti vietiin ulos TAR-tiedostona. Valmis kontti on helppo tuoda Dockeriin ja ajaa ilman lisäasennuksia.

5.1 SCADASim-kontti

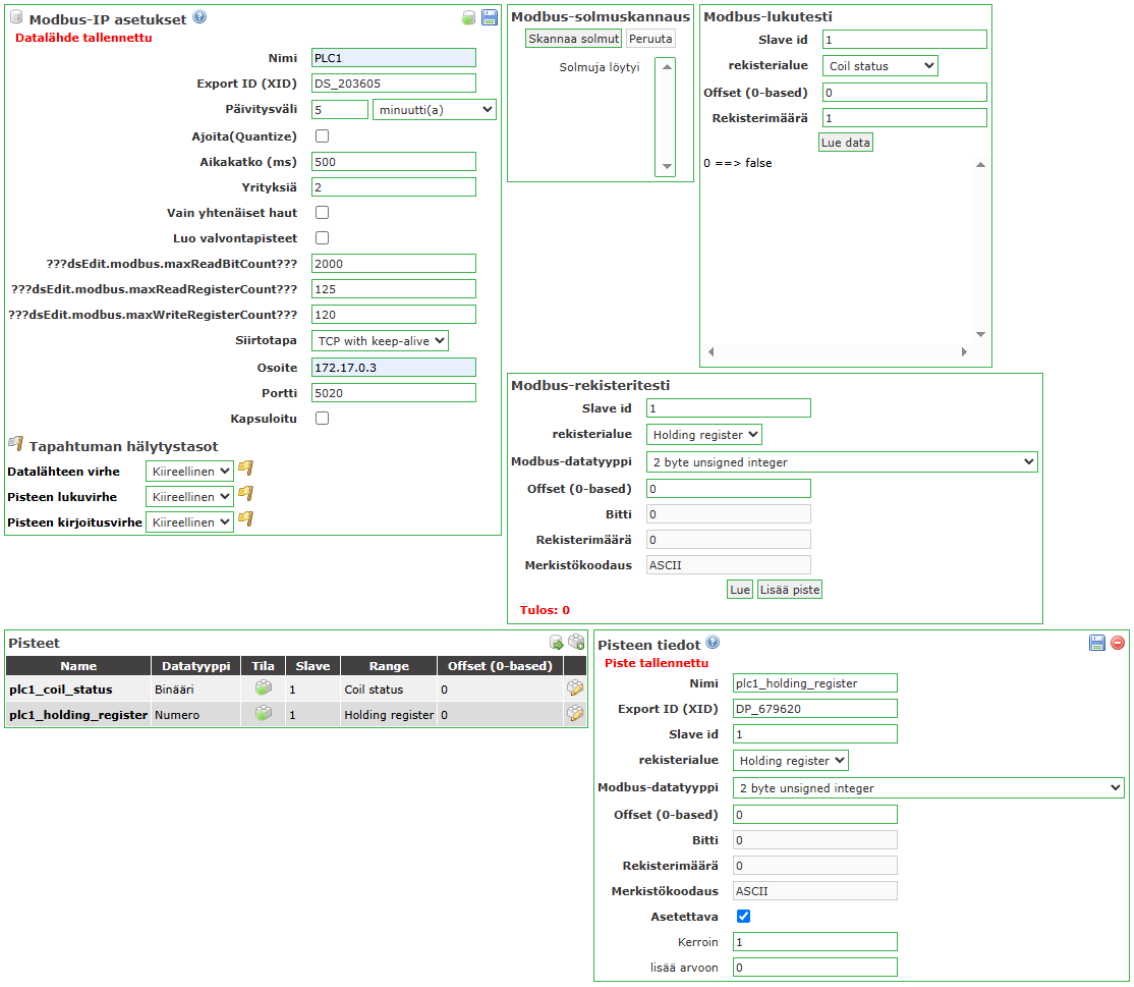
Työ alkoi SCADASimin lataamisella GitHub-tietovarastosta ja Pythonin asentamisella konttiin. SCADASimin riippuvuudet täytyi asentaa, jotta SCADASim toimi. Riippuvuuksiin kuului esimerkiksi pymodbus. Konfiguraatiotiedostot vaativat paljon muokkauksia, koska tiedostopolut olivat usein kovakoodattuja eivätkä käyttäneet erikseen Python3-komentoja.

Työssä käytettiin SCADASimin oletuskonfiguraatioita testauksiin, mutta oli ollut myös mahdollista luoda omia skenaarioita tarjotulla konfiguraation luomiskomentosarjalla. Käytetyssä skenaariossa oli neljä vesisäiliötä eri tilavuuksilla. Säiliöitä saattoi täyttää tai tyhjentää vaihtamalla kelan tilaa.

5.2 ScadaBR-kontti

ScadaBR:n mukana tuli asennuskomentosarja, joten riippuvuuksia ei tarvinnut erikseen asentaa. Asennuksen valmistuttua ScadaBR:ää kyettiin hallitsemaan ja seuraamaan selain-käyttöliittymän välityksellä. Sivu toimi käyttöliittymänä seurantaa ja hallintaa varten.

Käyttöliittymässä lisättiin SCADASim-kontissa simuloidut PLC:t IP-osoitteilla ja porteilla. Jokaiseen PLC:hen lisättiin datapisteitä, joita pystyi seuraamaan tai muokaamaan olotilan mukaan. Kuvassa 4 on lisätty PLC-datalähde ja sen datapisteet. PLC-datasiirto tapahtui Modbus TCP/IP:n välityksellä.

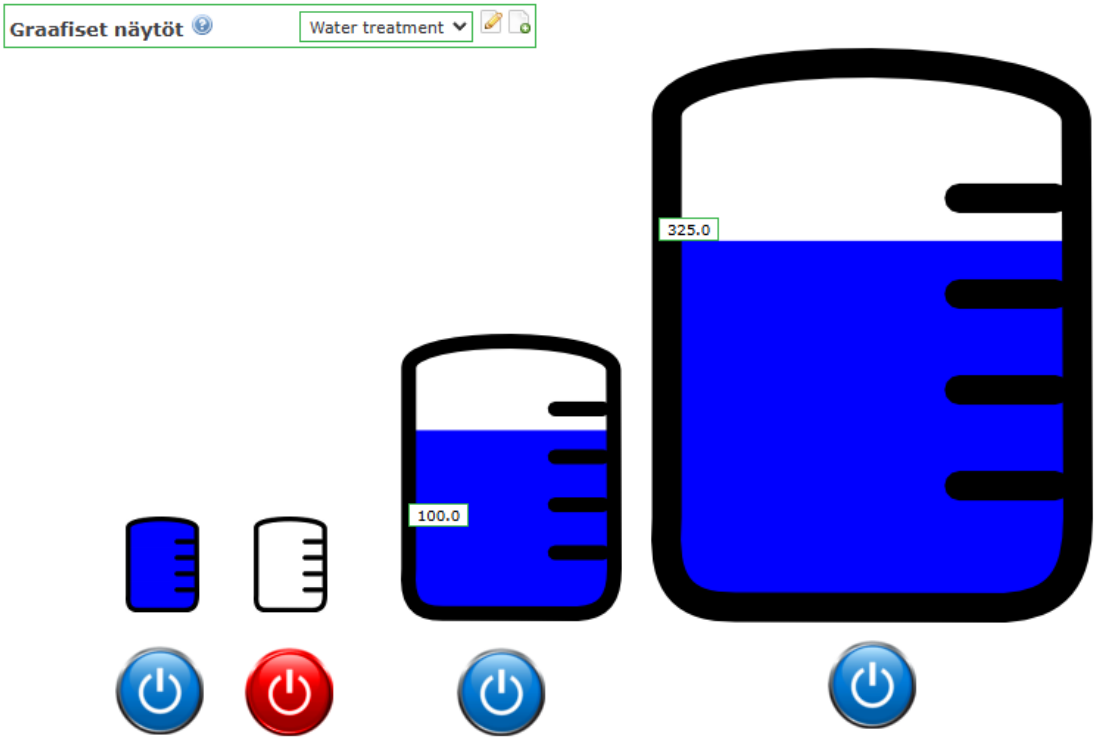


Kuva 4. ScadaBR-käyttöliittymässä on lisätty datalähde ja datapisteet.

Datalähteitä voi lisätä seurantalistaan kuten kuvassa 5. Seurantalista päivittyy automaattisesti ja näyttää valittujen datapisteiden arvot valitun päivitysvälin mukaan.

Kuva 5. Käyttöliittymän seurantalista. Seurantalistassa on eri datapisteitä ja niiden nykyiset arvot.

Datalähteitä voi myös seurata graafisesti. Graafiselle näytölle asetetaan kuvakkeita käyttötarkoituksen mukaan. Koska käytetyssä skenaariossa on vesisäiliöitä, käytetään säiliökuvakkeita. Kuvassa 6 on esimerkki siitä, miltä graafinen näyttö näyttää.



Kuva 6. ScadaBR:n graafiset näytöt neljällä vesisäiliöllä ja kelan päälle/pois -kuvake.

Kuvakkeiden sininen osio nousee tai laskee rekisteriarvon mukaan. Kelan ollessa päällä säiliö täyttyy, ja ollessaan pois päältä säiliö tyhjentyy. Kuvakkeiden koko on visuaalinen eikä vaikuta arvoihin.

5.3 pyModbusTCP-kontti

pyModbusTCP-kontilla testattiin haavoittuvuuksia Modbus-protokollassa. pyModbusTCP:llä voidaan lähettää luku- ja kirjoitustoimintakoodeja SCADASim-PLC:hin. On mahdollista esimerkiksi lähettää PLC:lle arvo, joka ylittää normaalin maksimirajan, mutta arvo silti kirjoitetaan PLC:n rekisteriin.

Testausskenaariossa PLC2:n maksimirekisteriarvo on kymmenen. PLC2:n porttiin 5021 kirjoitetaan arvo 2500 toistuvasti 0,5 sekunnin välein. Kuvassa 7 on PLC2:n arvot ennen kuin kirjoituskomentosarja ajetaan sekä sen arvot, kun komentosarjaa ajetaan jatkuvasti.

PLC2 - coil_status	1	13:35:31
PLC2 - hr	10.0	13:35:31
PLC2 - coil_status	1	13:35:59
PLC2 - hr	2500.0	13:35:59

Kuva 7. PLC2 seurantalistassa, jossa on rekisteriarvot ennen komentosarjan ajamista ja sen jälkeen.

Kontin tarkoitus on olla helppo tapa testata Modbus-protokollan turvallisuuspuutteita. pyModbusTCP:tä voidaan myös käyttää lokikirjanpitoon tai vianetsintään. Haluttaessa sitä voidaan myös käyttää valvontatarkoituksiin.

6 Johtopäätökset

Työn tavoitteena oli luoda virtuaalinen ympäristö ICS-laitteiden kyberturvallisuustestaukseen käyttäen konttiteknologiaa. Tuloksena on ympäristö, jossa simuloidaan ICS-laitteita, käytetään käyttöliittymää niiden valvontaan ja testataan yhteyksien haavoittuvuuksia. Toteutus hyötyisi jatkokehityksestä, erityisesti SCADASimin kovakoodattujen asioiden korjaamisesta ja aidompien skenaarioiden luomisesta.

Ympäristö eroaa aidosta ympäristöstä, koska komponentit eivät ole samoja kuin virtuaalisessa ympäristössä. SCADASim ei ole tarpeeksi pitkälle kehitetty ratkaisu simuloimaan tiettyjä komponentteja. Käyttötarkoitukseen se on kuitenkin hyvä alusta yleiseen testaamiseen ja ICS-simulointiin. Työn toteutuksessa ei myöskään ole samoja haavoittuvuuksia kuin aidossa ympäristössä. Työstä esimerkiksi puuttuvat mahdolliset perinneohjelmat tai vanhentunut käyttöjärjestelmä, joissa on haavoittuvuuksia.

Työn olisi voinut toteuttaa myös konttiteknologian sijaan virtuaalikoneilla. Proxmox-virtualisointialustalla, VirtualBox-virtuaaliohjelmistolla tai vastaavalla teknologialla pystyisi saavuttamaan saman lopputuloksen kuin Dockerilla. Kuitenkin Dockerin vähäiset resurssivaatimukset vaikuttivat tämän toteutuksen teknologiavalintaan.

Lähteet

Alves, T. 2021. ScadaBR Installer. *Github arkisto.* Viitattu: 25. 3 2025. https://github.com/thiagoralves/ScadaBR_Installer.

Buchanan, I. n.d. Containers vs. virtual machines. Atlassian. Viitattu 3.4.2025. https://www.atlassian.com/microservices/cloud-computing/containers-vs-vms.

CISA. 2021. *Cyber-Attack Against Ukrainian Critical Infrastructure.* 2021. Viitattu 19.4.2025. https://www.cisa.gov/news-events/ics-alerts/ir-alert-h-16-056-01.

Docker. n.d. The #1 containerization software for developers and teams. Viitattu 6.4.2025. https://www.docker.com/products/docker-esktop/.

ENISA. 2024. *ENISA Threat Landscape 2024.* Viitattu 19.4.2025. https://www.enisa.europa.eu/sites/default/files/2024-11/ENISA%20Threat%20Landscape%202024_0.pdf.

Flaus, J. 2019. *Cybersecurity of Industrial Systems. John Wiley & Sons, Incorporated.* Viitattu 7.3.2025. https://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=5813671. Vaatii käyttäjätunnuksen.

Fortinet, Inc. n.d. Purdue Model for ICS Security. Viitattu 27. 5 2025. https://www.fortinet.com/resources/cyberglossary/purdue-model.

Hasan, K. 2021. SCADASim. GitHub arkisto. Viitattu 25.3.2025. https://github.com/kamranhasan/SCADASim.

Mowbray, T. 2014. *Cybersecurity : managing systems, conducting testing, and investigating intrusions.* Hoboken, New Jersey: Wiley. Viitattu 18.4.2025. https://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=1481185. Vaatii käyttäjätunnuksen.

Muli, J. 2018. *Beginning DevOps with Docker: Automate the deployment of your environment with the power of the Docker toolchain (1st edition.). Packt Publishing.* Viitattu 25.3.2025. https://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=5405695. Vaatii käyttäjätunnuksen.

Red Hat. 2021. *What is containerization?* 2021. Viitattu 23.3.2025. https://www.redhat.com/en/topics/cloud-native-apps/what-is-containerization.

Stouffer, K; Pease, M; Tang, C; Zimmerman, T; Pillitteri, V; Lightman, S; Hahn, A; Saravia, S; Sherule, A; Thompson, M. 2023. *Guide to Operational Technology (OT) Security. NIST.* Viitattu 19.3.2025. https://doi.org/10.6028/NIST.SP.800-82r3.

Susnjara, S & Smalley, I. 2024. *What are containers? IBM.* Viitattu 4.4.2025. https://www.ibm.com/think/topics/containers

The Modbus Organization. 2012. *MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b3.* 2012. Viitattu 26.3.2025. https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf.

Watada, J; Roy, A; Kadikar, R; Pham, H; Xu, B. 2019. *Emerging Trends, Techniques and Open Issues of Containerization: A Review.* IEEE Access. Viitattu 23.3.2025. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8861307.