jamk.fi

SCADA-protokollien haavoittuvuudet ja vaikutusmahdollisuudet

Pauli Kela

Opinnäytetyö
Maaliskuu 2017
Tekniikan ja liikenteen ala
Insinööri (AMK) ICT
Tietotekniikka
Tietoverkkotekniikka

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Kela, Pauli	Opinnäytetyö, AMK	Maaliskuu 2017
	Sivumäärä 75	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi

SCADA-protokollien haavoittuvuudet ja vaikutusmahdollisuudet

Tutkinto-ohjelma

Tietotekniikan koulutusohjelma

Työn ohjaaja(t)

Mika Rantonen

Antti Häkkinen

Toimeksiantaja(t)

JYVSECTEC

Marko Vatanen

Tiivistelmä

Opinnäytetyön toimeksiantajana oli kyberturvallisuuden tutkimus-, kehitys- ja koulutuskeskus JYVSECTEC, joka tuottaa kyberturvallisuusharjoituksia, konsultointi-, tutkimus-, testaus- ja koulutuspalveluita kansallisille ja kansainvälisille yhteystyökumppaneille. JYVSECTEC –projektin tavoitteena on tietoturvallisuuden johtamisjärjestelmien ja niiden käytön osaamisen kehittäminen sekä tietoturvallisuuteen liittyvän tutkimustyön ja yritysmaailman tietämyksen tukeminen.

Tavoitteena työssä oli tutkia sekä testata tuotanto- ja teollisuusverkoissa laajalti käytössä olevien OPC-, Modbus TCP- ja IEC 60870-5-104 -SCADA-protokollien tietoturvahaavoittuvuuksia. Lisäksi niitä hyödyntämällä työssä pyrittiin vaikuttamaan SCADA-protokollia käyttävien simuloitujen SCADA-järjestelmien toimintaan ja kommunikointiin.

Työ toteutettiin virtuaalikoneista koostuvassa testiympäristössä, jossa SCADA-laitteiden toimintaa simuloitiin protokollasimulaattoreilla. Simuloituja SCADA-järjestelmiä vastaan tehtiin erityyppisiä kyberhyökkäyksiä. Erilaisia kyberhyökkäysmenetelmiä ja hyökkäysvaiheita jäljittelemällä pyrittiin kattamaan useita protokolliin liittyviä haavoittuvuuksia. Nämä liittyivät oikean hyökkäyskohteen löytämiseen ja kohteen suoraan vaikuttamiseen.

Työn tuloksena saatiin suoritettua jokaisen kolmen protokollan kohdalla onnistunut haavoittuvuuden testaus ja todennus.

Avainsanat (asiasanat)

JYVSECTEC, SCADA-protokolla, haavoittuvuudet, OPC, Modbus TCP, IEC 60870-5-104

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Kela, Pauli	Bachelor's thesis	March 2017
		Language of publication: Finnish
	Number of pages 75	Permission for web publication: x

Title of publication
Vulnerabilities in SCADA protocols

Degree programme
Data Network Technology

Supervisor(s)
Rantonen, Mika
Häkkinen, Antti

Assigned by
JYVSECTEC
Vatanen, Marko

Abstract

This thesis was assigned by JYVSECTEC, a cyber security research, development and educational center, which provides cyber security related training, consultation, research, testing and educational services to both national and international collaborators. The main goal of the JYVSECTEC project is to develop management systems and the expertise of their use in the field of cyber security. Additionally, their goal is to support the research and awareness in the business world regarding cyber security.

The goal in this thesis was to study and test the vulnerabilities of OPC, Modbus TCP and IEC 60870-5-104 SCADA-protocols, which are still widely used in industrial and production networks. In addition to exploiting these vulnerabilities, the objective of this study was to gain means of influencing the operation and communications of simulated SCADA systems.

The research was carried out in a test environment consisting of virtual machines, which simulated the operation of SCADA devices using protocol simulation software. Various cyber attack methods and stages were then used against this simulated test environment. Multiple vulnerabilities of protocols were covered by imitating different attack methods and stages. These vulnerabilities relate to finding the right target for attack and attacking it.

As a result, the vulnerabilities of all three protocols were successfully tested and confirmed in this thesis.

Keywords/tags (subjects)
JYVSECTEC, SCADA protocols, vulnerabilities, OPC, Modbus TCP, IEC 60870-5-104

Miscellaneous

Sisältö

Kuviot

LYHENTEET

ADU	Application Data Unit
APCI	Application Service Control Information
ARP	Address Resolution Protocol
ASDU	Application Service Data Unit
COT	Cause of Transmission
CRC	Cyclic Redundancy Check
(D)COM	(Distributed) Component Object Model
DLL	Dynamic-Link Library
HMI	Human-Machine Interface
HTTP(S)	Hypertext Transfer Protocol (Secure)
ICS	Industrial Control System
IED	Intelligent Electronic Device
IP	Internet Protocol
MAC	Media Access Control
MITM	Man-in-the-Middle
MTU	Master Terminal Unit
OPC	OLE (Object Linking and Embedding) for Process Control
PLC	Programmable Logic Controller
RAT	Remote Access Trojan
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SMB	Server Message Block
SPI	Single Point Information

SQL	Structured Query Language
TCP	Transmission Control Protocol
TLS	Transport Layer Security
VPN	Virtual Private Network

1 TYÖN LÄHTÖKOHDAT

1.1 Toimeksiantaja

JYVSECTEC on kyberturvallisuuden tutkimus-, kehitys- ja koulutuskeskus, joka tuottaa kyberturvallisuusharjoituksia, konsultointi-, tutkimus-, testaus- ja koulutuspalveluita kansallisille ja kansainvälisille yhteystyökumppaneille. Jyväskylän ammattikorkeakoulun IT-instituutti JYVSECTEC:n kehittäminen käynnistyi syyskuussa 2011 projektina, jonka tavoitteena oli luoda Keski-Suomeen yksi Suomen johtavista kyberturvallisuuden tutkimus-, kehitys- ja koulutuskeskuksista sekä kehittää niin kansallista kuin kansainvälistäkin yritysten ja toimijoiden yhteistyötä ja verkostoa.

Keskeisessä osassa näiden tavoitteiden saavuttamista on ollut JYVSECTEC:n RGCE-kehitysympäristö (Realistic Global Cyber Environment), jonka myötä JYVSECTEC on kyennyt tarjoamaan ajanmukaiset puitteet, tarvittavat järjestelmät sekä asiantuntijuutta yhteistyöorganisaatioiden kyberturvallisuustietämyksen kehittämiseen ja harjoitusten toteutukseen. RGCE-verkko on julkisesta verkosta eristetty kybertoimintaympäristö, joka tarjoaa riskittömän kehitysympäristön tietoturvahyökkäysten ja -haavoittuvuuksien sekä haittaohjelmien tutkimiselle. Näin ollen kyberturvallisuuden eri ilmiöitä voidaan tutkia ilman, että vaarannetaan oikeita tuotantoympäristöjä, -verkkoja tai –järjestelmiä. (JYVSECTEC n.d)

1.2 Työn tavoitteet ja vaatimusmäärittely

Kriittiseen kansalliseen infrastruktuuriin kuuluvien teollisuuden alojen tuotantoprosesseissa käytettävät SCADA-protokollat on kehitetty vuosikymmeniä sitten, kun SCADA-verkot oli vielä eristetty omiksi teollisuusverkoikseen. TCP/IP-protokollan yleistymisen myötä myös tuotantoympäristöissä on kuitenkin siirrytty käyttämään tätä ”de facto” –protokollaa SCADA-järjestelmien väliseen kommunikointiin, mutta SCADA-protokollat itsessään eivät useinkaan sisällä tietoturvaominaisuuksia kuten eheyden tarkistusta tai autentikointia. Simuloimalla eri protokollia käyttäviä SCADA-järjestelmiä voidaan perehtyä niiden sisäisiin tietoturvahaavoittuvuuksiin ja testata haavoittuvuuksien hyödyntämistä. Altistamalla simuloidut SCADA-järjestelmät eri vaikutusmahdollisuuksille voidaan riskittömästi

havainnoida niiden vaikutuksia ja peilata seurauksia oikean maailman teollisuusympäristöihin. Ymmärtämällä haavoittuvuuksien vakavuudet ja niiden hyödyntämisestä johtuvat seuraukset voidaan puolestaan luoda pohjaa niiltä suojaavien tietoturvaratkaisujen tutkimiselle ja kehittämiselle.

Tässä työssä on testattu erilaisia hyökkäyksiä maailmanlaajuisesti käytössä olevia OPC-, Modbus TCP- ja IEC 60870-5-104 -teollisuusprotokollia vastaan. Työn tavoitteena on peilata hyökkäyksen tuloksia ja niiden vaikutuksia oikean maailman SCADA-ympäristöihin.

2 HAAVOITTUVUUSTESTAUS

2.1 Mitä on haavoittuvuustestaus?

Haavoittuvuustestaus on tietojärjestelmään, sen komponenttiin tai muuhun resurssiin kohdistuvaa eettisesti hyväksyttävää tietomurtojen tekemistä ja tietoturvallisuustason määrittämistä. Haavoittuvuustestauksissa pyrkimyksenä on päästä käsiksi testattavaan järjestelmään käyttämättä perinteisiä pääsykeinoja kuten käyttäjänimiä, salasanoja tai pääsyoikeuksia. Haavoittuvuustestauksessa käytetään kyberhyökkäyksien menetelmiä, mutta toisin kuin oikealla hyökkääjällä, haavoittuvuustestaajalla on resurssien omistajan lupa tunkeutumisen suorittamiseksi. Lupaa vastaan testaaja raportoi resurssin omistajalle suojattavan kohteen haavoittuvuuksista ja niiden suojaamiskeinoista. (Northcutt, S., Shenk, J., Shackleford, D., Rosenberg, T., Siles, R. & Mancini, S. 2006.)

Testauksen tavoite on riippuvainen suojeltavasta kohteesta, esimerkiksi dataresurssien tapauksessa onnistunessa haavoittuvuustestauksessa saavutetaan pääsy tietokantoihin, luottamuksellisiin dokumentteihin sekä muuhun salassa pidettävään tietoon. Haavoittuvuustestauksen tavoitteena on parantaa testattavan tietojärjestelmän tai sen komponentin tietoturvallisuutta. (Northcutt, S., Shenk, J., Shackleford, D., Rosenberg, T., Siles, R. & Mancini, S. 2006.)

2.2 Terminologiaa

Tietoturvallisuudesta ja haavoittuvuustestaamisesta puhuttaessa käytetään usein totutusti erityismerkityksen omaavia tai vierasperäisiä termejä. Tämän työn kannalta oleellisia termejä käydään lyhyesti läpi seuraavassa listauksessa:

Ad hoc = “Varta vasten” tapahtuva toiminta, joka suoritetaan tilanteen sitä vaatiessa, ilman sen aiempaa suunnittelua (esimerkiksi ad hoc –testaus).

Brute force = ”Raakaa (laskenta)voimaa” hyödyntävä kryptografinen hyökkäystekniikka, johon liittyy kaikkien merkkijonojen läpikäyminen systemaattisesti ja yksitellen, kunnes oikea merkkiyhdistelmä löytyy.

DoS (Denial-of-Service): Palvelunestohyökkäys, jossa hyökkääjän tavoitteena on estää laitteen, palvelun tai muun verkkoresurssin tarkoitettu käyttö.

Exploit: Haavoittuvuuden hyödyntäminen.

Hakkeri: Henkilö, joka edistynyttä tietoteknistä osaamistaan hyödyntämällä saavuttaa pääsyn suojattuun tietoresurssiin.

Hyökkäys: Pyrkimys ohittaa tietojärjestelmän turvallisuusmekanismit.

Hyökkäyspotentiaali: Havainnoitava todennäköisyys hyökkäyksen onnistumiselle, joka on riippuvainen hyökkääjän taitotasosta, käytettävissä olevissa resursseista ja motivaatiosta.

Hyökkäysvektori: Hyökkäyskeino tai –tapa.

Sertifikaatio: Valtuuttamisprosessin yhteydessä käytettävä turvallisuusarviointi, joka ilmoittaa kuinka hyvin sertifioitu osapuoli täyttää turvallisuusvaatimukset

Penetraatio: Tietojärjestelmän tai -resurssin tietoturvamekanismien onnistunut, auktorisoimaton ohittaminen.

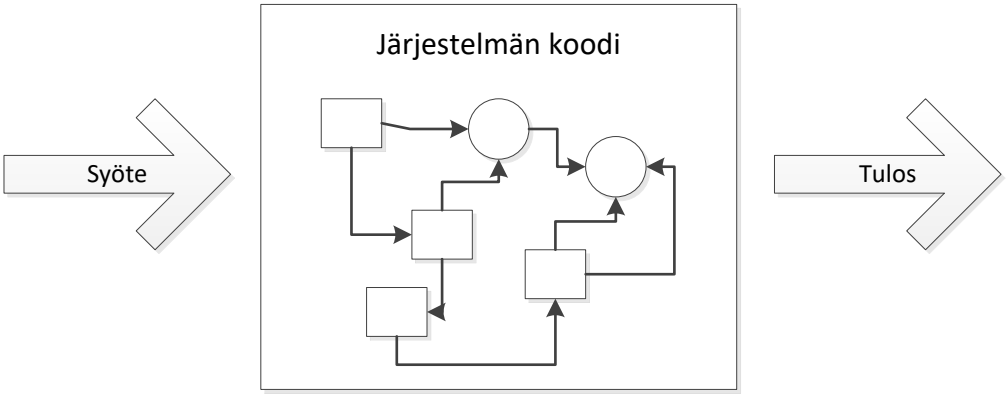
Protokolla: Prosessien, laitteiden tai muiden komponenttien väliseen kommunikointiin käytettävä, yhteisesti sovittu menettelykäytäntö.

Troijalainen: Hyödylliseksi tai viattomaksi ohjelmaksi naamioitu haittaohjelma tai koodi, joka suoritettaessa avaa haavoittuvuuden tietojärjestelmään.

(Glossary of Vulnerability Testing Terminology n.d.)

2.3 Testausmenetelmät

White-box testing on järjestelmän tai ohjelman testaustapa, joka testauksen onnistumiseksi vaatii testaajalta paljon tietoa tai kokemusta testattavan ohjelman suunnittelusta, sisäisestä rakenteesta, toimintatavasta ja/tai sen implementointiin liittyvistä asioista. White-box -testauksen keskipisteessä ovat järjestelmän sisäiset komponentit, koodi sekä toimintatapa, joiden kautta ohjelman kokonaisvaltaista toimivuutta testataan. White-box –testaustapaa on havainnollistettu kuviossa 1. (White Box Testing Fundamentals n.d.)



Kuvio 1. White-box –testaus

Black-box testing on myös käyttäytymistestaukseksi nimitetty, sovellustasolla tapahtuva testaustapa, jossa testaajalla ei ole mitään aiempaa tietoa tai kokemusta testattavan kohteen sisäisestä rakenteesta, suunnittelusta tai implementoinnista. Black-box –testausmetodia on havainnollistettu kuviossa 2. Tällä testausmetodilla pyritään usein havaitsemaan seuraaviin asioihin liittyviä vikoja:

- Väärät tai puuttuvat toiminnot
- Rajapintaongelmat
- Viat datarakenteissa tai tietokannoissa
- Käyttäytymiseen tai suorituskykyyn liittyvät ongelmat
- Käynnistämiseen ja sulkemiseen liittyvät viat (Black Box Testing Fundamentals n.d.)

Kuvio 2. Black-box -testaus

Gray-box testing on testausmenetelmä, joka on yhdistelmä White- ja Black-Box –metodeja. Gray-box –testauksessa testattavan kohteen sisäiset rakenteet ovat osittain testaajien tiedossa. Usein tämä tarkoittaa järjestelmän testauksessa sitä, että

järjestelmää testataan niin sanotun ”peruskäyttäjän” roolissa kuten Black-box -testauksessa, mutta testauksen suunnitteluvaiheessa testaajilla on pääsy järjestelmän sisäisiin rakenteisiin tai tietämys niistä. (Gray Box Testing Fundamentals n.d.)

3 KYBERHYÖKKÄYKSET

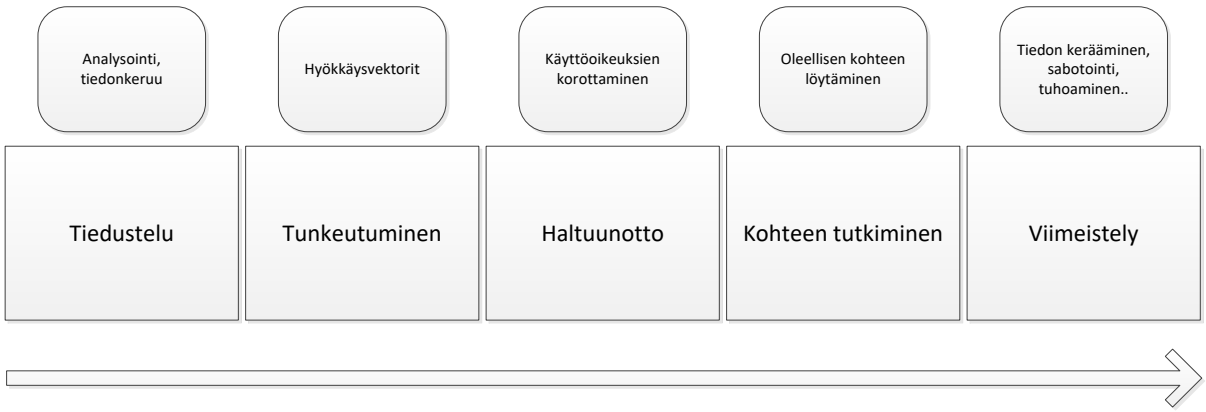
3.1 Yleistä

Kyberhyökkäykset ovat poliittisista, taloudellisista tai sosiaalisista syistä Internetin välityksellä suoritettuja tietoverkkohyökkäyksiä, jotka kohdistuvat joko valtaväestöön, yrityksiin tai kansallisiin organisaatioihin. (What Constitutes a Cyber Attack? n.d.) Hyökkäyksen motiivit voidaan luokitella poliittisiin, taloudellisiin ja sosiokulttuurisiin motiiveihin ja ne voivat ilmentyä vahingon tuottamisena, salassapidettävien tietojen varastamisena, toiminnan häiritsemisenä, vakoiluna tai julkisuuden tavoitteluna. (Gandhi, R. 2011.) Kyberhyökkäyksessä käytettävät tekniikat ja hyökkäysvaiheet riippuvat paljolti sen kohteesta ja hyökkäyksen tavoitteista. Eri hyökkäystekniikoista ja -vaiheista yleisimpiä kuvaillaan tarkemmin luvuissa 3.2 ja 3.3.

3.2 Kyberhyökkäyksen vaiheet

Kyberhyökkäyksen ensimmäinen vaihe koostuu hyökkäyskohteen mahdollisimman tarkasta analysoinnista sekä taustatiedon keräämisestä. **Tiedusteluvaihe** on hyökkäyksen kriittisin ja samalla aikaavievin, sillä se näyttelee suurta roolia siinä, kuinka hyvin hyökkäyksen tavoitteet toteutuvat. Tietoa kerätään muun muassa kohteessa käytössä olevista järjestelmistä, laitteista ja protokollista, joista mahdollisesti löytyviä haavoittuvuuksia ja heikkouksia voisi käyttää hyväksi sekä järjestelmään murtautuessa että myöhemmissä hyökkäyksen vaiheissa. **Tunkeutumisvaiheessa** avainasemaan nousevat kohteessa käytössä olevat teknologiat sekä niiden haavoittuvuudet. Eri tunkeutumismenetelmiin lukeutuvat muun muassa ”waterholing-”, phishing- sekä SQL-injektiomenetelmät, joilla hyökkääjät tavoittelevat väylää kohdeverkkoon. Kohdeverkkoon tunkeutumisen jälkeen hyökkääjä pyrkii saamaan jalansijaa verkossa korottamalla pääsy- ja käyttöoikeuksiaan ja näin tekemällä **ottamaan verkon osia haltuunsa**. Korotetuin oikeuksin hyökkääjä kykenee liikkumaan kohdeverkossa vapaammin ja **etsimään verkosta** hyökkäyksen tavoitteen kannalta otollisinta hyökkäyskohdetta. Kohteen löytyessä suoritetaan varsinainen **hyökkäyksen viimeistely,** oli kyseessä sitten

tietomurto salatun informaation toivossa, palvelun tai muun resurssin sabotointi tai muu tavoite. (Rhoades, T. 2016.) Edellä kuvattuja tyypillisimpiä hyökkäyksen vaiheita on havainnollistettu kuviossa 3.



Kuvio 3. Kyberhyökkäyksen vaiheet

3.3 Hyökkäystyypit

3.3.1 DoS (Denial of Service)

DoS -hyökkäyksessä hyökkääjän tavoitteena on estää palvelun, informaation tai muun verkkoresurssin käyttö sen tarkoitetulta käyttäjäryhmältä. Yleisimmissä DoS-hyökkäyksissä hyökkääjä pyrkii ylikuormittamaan kohdeverkkoa suurilla määrillä dataa, tavoitteenaan hidastaa sitä tai kokonaan lamaannuttamaan sen toiminnan. Onnistuneessa DoS-hyökkäyksessä kohdepalvelin ylikuormittuu käsitellessään hyökkääjän dataryöppyä, eikä se näin ollen kykene käsittelemään muita palvelimelle saapuvia, oikeita palvelupyyntöjä. (Understanding Denial-of-Service Attacks 2013.)

3.3.2 Phishing

Phishing-hyökkäys tarkoittaa nimensä mukaisesti henkilökohtaisen tai muun salassa pidettävän informaation kalastelua, jossa informaatiota tavoitteleva hyökkääjä esiintyy luotettavana lähteenä. Phishing-menetelmää käytetään sähköpostiviestinnässä, sosiaalisessa mediassa tai matkapuhelinviestinnässä joko puheluiden tai tekstiviestien muodossa. Spear phishing –menetelmä toimii samalla

periaatteella, mutta kohdistuu yrityksiin tai organisaatioihin. Jotta informaatiopyyntö vaikuttaisi mahdollisimman aidolta, yhteydenotossa käytetään kohdeyritykseen tai sen toimialaan liittyviä yksityiskohtia. (Scamwatch n.d.)

3.3.3 Haittaohjelmat

Haittaohjelma eli malware (malicious software) on haitallinen tai ärsyttävä ohjelma, joka on tarkoitettu asentuvan kohdelaitteelle huomaamattomasti, usein jonkin muun sovelluksen asennuksen tai saastuneen sähköpostiliitteen avaamisen yhteydessä. Eri haittaohjelmatyyppejä ovat muun muassa vakoilu- ja mainosohjelmat, virukset, troijalaiset ja rootkitit. (What is Malware? n.d.) Kyberhyökkäyksissä malwareohjelmia käytetään useimmiten vakoilutarkoituksiin, haitallista koodia suorittavien ohjelmien levittämiseen kohdeverkossa sekä hyökkääjän käyttöoikeuksien korottamiseen.

3.3.4 SQL-injektio

SQL-injektio (Structured Query Language) on hyökkäysmenetelmä, jossa hyökkääjä antaa poikkeavia syötteitä SQL-kyselyjä käyttävän ohjelman käsiteltäväksi ja seuraa, kuinka ohjelma vastaa kyselyihin. SQL-haavoittuvuuden omaava ohjelma voi virheellisesti lukea hyökkääjän kyselyn komentona, jolloin hyökkääjä voi päästä käsiksi ohjelman tietokantaan. Tietokantaan päästyään hyökkääjä voi esimerkiksi manipuloida ohjelmaa sallimaan tietokantaan perustuvan sisäänkirjautumisen tai luvatta muokata tai poistaa tietokannan dataa. (SQL Injection Cheat Sheet & Tutorial: Vulnerabilities & How to Prevent SQL Injection Attacks n.d.)

3.3.5 Man-in-the-Middle

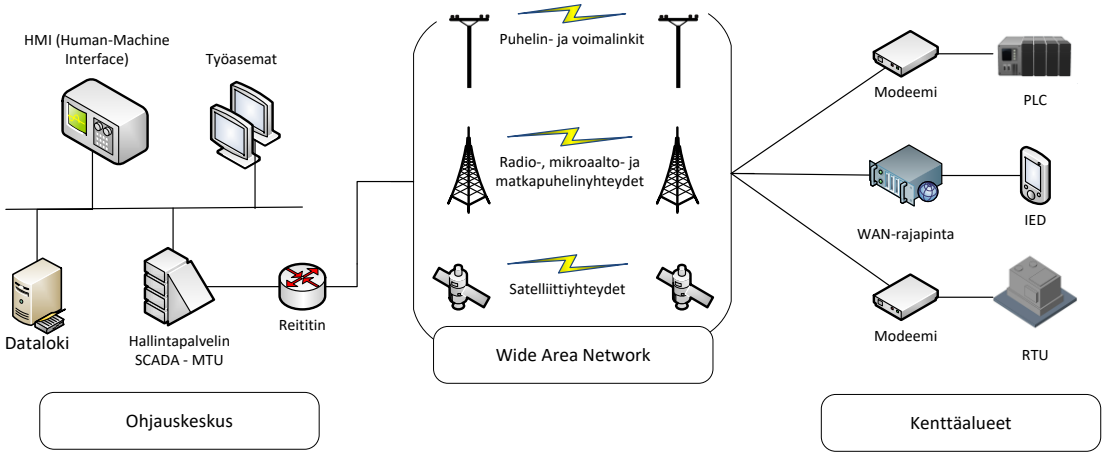
Man-in-the-Middle –hyökkäyksessä hyökkääjä sieppaa kahden laitteen välisen kommunikaatioyhteyden esittämällä molempia osapuolia, jotta nämä ”luulevat” keskustelevansa toistensa kanssa. Tämä tapahtuu kohdelaitteiden ARP (Address Resolution Protocol) -tietoja väärentämällä, jota käsitellään yksityiskohtaisemmin luvussa 10.1.1. Tosiasiassa kaikki laitteiden välinen kommunikaatio kiertää kuitenkin hyökkääjän kautta, jolloin tämä kykenee salakuuntelemaan ja hyökkäyskohteen turvaominaisuuksista riippuen jopa muokkaamaan kommunikaation sisältöä. Man-in-the-Middle –hyökkäyksen tehokkuus perustuu http-protokollaan ja tiedonsiirtoon,

jotka molemmat ovat ASCII-pohjaisia. Näin ollen onnistuneessa MITM-hyökkäyksessä hyökkääjän on helppo tulkita kohdelaitteiden välistä kommunikaatiota. (Cagalaban, G., So, Y. & Kim, S. 2009.)

4 SCADA-JÄRJESTELMÄT

4.1 Tarkoitus ja käyttökohteet

SCADA-protokollia ja niitä käyttäviä järjestelmiä käytetään monitoroimaan ja ohjaamaan kriittisten teollisuuden ja tuotannon alojen järjestelmiä vaihtelevien välimatkojen päästä. Näitä ovat esimerkiksi energia-, vesivoima-, liikenne- sekä öljyteollisuuden alat, joissa SCADA-protokollia käytetään ohjaamaan esimerkiksi siirtoputkien, tuotantolaitteiston sekä ilmastointijärjestelmien toimintaa. SCADA-järjestelmän toiminta koostuu tuotantojärjestelmien ja niiden prosesseihin liittyvän informaation mittaamisesta ja keräämisestä sekä sen siirtämisestä hallintakeskukselle, jossa kerätty informaatio kootaan visuaalisesti ylläpitäjän hallintapäätteille analysoitavaksi. Helposti tulkittavan informaation pohjalta järjestelmää voidaan ylläpitää lähettämällä komentoja takaisin mittaus-ja ohjauslaitteistolle. Esimerkki tyypillisestä SCADA-järjestelmästä on esitetty kuviossa 4. (Stouffer, K., Falco, J. & Kent, K. 2006.)



Kuvio 4. Tyypillisen SCADA-järjestelmän rakenteita

4.2 Tietoturvallisuus

SCADA-järjestelmät olivat ennen vanhaan eristetty muusta infrastruktuurista sekä sijaintinsa että verkkoympäristönsäkin puolesta, joten luonnollisesti ne olivat myös nykyistä tietoturvallisempia. Ethernet- ja TCP/IP-protokollien yleistymisen myötä myös SCADA-järjestelmät siirtyivät käyttämään näitä tiedonsiirrossaan

virtaviivaistaakseen toimintaansa. Langattomat verkkotekniikat ovat entisestään vähentäneet SCADA-järjestelmien eristyneisyyttä muusta julkisesta verkkoympäristöstä. Yleisimpiin SCADA-protokolliin ei kuulu sisäisiä tietoturvaominaisuuksia, sillä ne suunniteltiin vuosikymmeniä sitten, jolloin SCADA-järjestelmät toimivat eristetyissä ympäristöissä ja uhkanäkymät olivat paljon nykyistä vähäisemmät. Tästä syystä tietoturva-aspekteja ei juurikaan otettu protokollien suunnittelussa huomioon ja tietoturvaominaisuudet kuten tiedon eheys ja autentikointi jäivät puuttumaan protokollamäärityksistä. Nykyisin SCADA-järjestelmät ovat usein suorassa yhteydessä organisaation sisäiseen TCP/IP (Transfer Control Protocol/Internet Protocol) -paikallisverkkoon, jotta niille voidaan suorittaa ylläpidollisia toimenpiteitä ja hallintohenkilöstö voi tarkastella liiketoiminnalle oleellista informaatiota. Tämä tarkoittaa sitä, että SCADA-verkot voivat olla samalla epäsuorasti yhteydessä julkiseen verkkoon, mikä on kansallisen infrastruktuurin turvallisuuden kohdalla erittäin suuri riski. (Stouffer, K., Falco, J. & Kent, K. 2006.)

4.3 Tunnettuja SCADA-hyökkäyksiä

SCADA-järjestelmiin liittyvät uhkanäkymät eivät ole pelkästään teoreettisia. Kuten lukuisat oikean maailman esimerkit ovat osoittaneet, kyberhyökkäyksillä voi olla merkittäviä vaikutusmahdollisuuksia epäturvallisiin SCADA-järjestelmiin ja niitä hyödyntävään kansalliseen infrastruktuuriin. (Nigam 2016.) Luvuissa 3.3.1 – 3.3.3 käydään läpi hyökkäyksiä, jotka olivat erityisesti suunniteltu ja kohdistettu SCADA-järjestelmiä vastaan.

4.3.1 Night Dragon (2009)

”Night Dragon” oli kyberhyökkäysten sarja, jotka kohdistettiin Exxonin, Shellin ja BP:n ohella lukuisiin globaaleihin öljy-, energia-, ja petrokemikaaliyrityksiin. Hyökkäysten tavoitteena oli kerätä yritysten kilpailukykystrategioihin ja projektirahoituksiin liittyvää arkaluontoista informaatiota. Raporttien mukaan hyökkäyksen lopputulemana hyökkääjät saivat käsiinsä edellä mainitun informaation lisäksi myös yritysten SCADA-järjestelmien rakenteeseen ja toimintaan liittyviä piirustuksia, joita hyödynnettiin myöhemmin hyökätessä näitä vastaan lisäinformaation varastamiseksi. (Global Energy Cyberattacks: “Night Dragon” 2011.)

4.3.2 Stuxnet (2010)

”Stuxnet” oli Yhdysvaltain ja Israelin yhteistyönä toteutettu kyberhyökkäys iranilaista Natanz-ydinvoimalaa vastaan. Stuxnet-viruksen avulla kyettiin kaappaamaan dataa ydinvoimalan SCADA-järjestelmien PLC (Programmable Logic Controller) -laitteilta sekä kirjoittamaan niihin muutoksia. Hyökkäyksen tavoitteina oli selvittää uraanin rikastamiseen käytettävien sentrifugien ja niitä ohjaavien tietokoneiden toimintaa, lyhentää sentrifugien elinikää pyörimisnopeuden vaihtelusta aiheutuvan värinän avulla sekä väärentää voimalan valvontahuoneen tilannenäkymää toistamalla vanhoja raportteja normaalitilanteesta. On arvioitu, että hyökkäyksen seurauksena viidennes Iranin ydinsentrifugeista tuhoutui ja Iranin ydinaseohjelma viivästyi vuosilla. (Schneier 2010.)

Epäillään, että alun perin virus levitettiin ydinvoimalan sisäverkkoon USB-tikun avulla kaksoisagentin toimesta. Kohdeverkkoon päästyään virus alkaa levittää itseään muihin saman verkon laitteisiin, joissa se piilottaa itsensä rootkitin avulla ja antaa käyttöjärjestelmähaavoittuvuuksien avulla korotetut käyttäjäoikeudet. Stuxnetin tarkoituksena ei ole vahingoittaa itse Windows-koneita, joihin se leviää, vaan se ainoastaan etsii niiltä tietynlaista Siemens-valmisteisen PLC-laitteen ohjaussovellusta nimeltä SIMATIC WinCC/Step 7. Jos virus löytää kyseisen ohjelman koneelta, se saastuttaa sen hyödyntäen ohjelman omaa haavoittuvuutta. Saastutettuaan PLC-laitteita ohjaavan tietokoneen, se lukee ja muuttaa tiettyjä bittiarvoja koneella ohjattavan PLC:n datassa. (Schneier 2010.)

4.3.3 Havex (2014)

”Havex” on RAT-tiedostosta (Remote Access Trojan) ja PHP-kielellä kirjoitetusta C&C-palvelimista (Command-and-Control) koostuva haittaohjelmaperhe, jota on käytetty eri teollisuussektoreilla toimivien laitevalmistajien SCADA-laitteistojen ja tiedostojen vakoiluun. Havex-haittaohjelman on tutkittu olevan erityisen kiinnostunut energiasektorin kohteista. (Havex Hunts For ICS/SCADA Systems 2014.)

F-Securen raportin mukaan hyökkääjät hyödynsivät haavoittuvuuksia ICS-ohjelmistoja tarjoavien yritysten verkkosivuja ylläpitävässä ohjelmistossa ja lopulta pääsivät käsiksi verkkosivuilta ladattavien ICS-ohjelmistojen asennuspaketteihin, joita

kohdeyrityksen asiakkaat voivat ladata omille ICS-järjestelmilleen. Oikeat asennuspaketit korvattiin troijalaisversioilla, joihin oli hyökkääjien toimesta lisätty mbcheck.dll –niminen RAT-tiedosto. Ohjelmisto asentuu ja toimii normaalisti, mutta DLL (Dynamic-Link Library) -tiedoston ansiosta hyökkääjälle avautuu takaovi tietokoneelle, jolle manipuloitu ICS (Industrial Control System) -ohjelmisto on asennettu. (Havex Hunts For ICS/SCADA Systems 2014.)

Saastutetun ohjelmiston toiminnallisuuteen kuuluu lisäkomponenttien lataaminen ja suorittaminen kohdekoneella. Yksi näistä komponenteista tutkii ja listaa kohdelaitteen paikallisverkkoa, etsien siihen yhdistettyjä OPC (OLE for Process Control) -resursseja ja –palvelimia. OPC:n DCOM (Distributed Component Object Model) -teknologiaan pohjautuen komponentti kerää tietoa yhdistetyistä OPC-resursseista ja lähettää ne hyökkääjien hallussa oleville C&C-palvelimille analysoitavaksi. Troijalaisen toiminnallisuudesta ei F-Securen tutkimusten perusteella ole löytynyt komponentteja, joilla OPC-protokollaa käyttäviä teollisuuslaitteita olisi voinut suoraan ohjata, vaan hyökkäystä on käytetty toistaiseksi vain vakoilutarkoituksiin. (Havex Hunts For ICS/SCADA Systems 2014.)

5 KÄYTETYT TEKNOLOGIAT

5.1 VMware vCloud

Työn testausympäristö luotiin virtuaalikoneilla JYVSECTEC:in hallinnoimaan VMwaren vCloud –pilviympäristöön. vCloud on virtualisointialusta, joka yhdistää VMwaren pilvi-infrastruktuuriin perustuvia ohjelmistoja yhteen pakettiin. vCloudin ominaisuuksiin kuuluvat datacenter-palvelut, virtualisointi, käytänteiden perusteella suoritettava resurssienjako sekä sovellusten ja toimintojen hallinta. VMware vCloudiin kuuluu seuraavat VMware-tuotteet ja niiden ominaisuudet:

- VMware vSphere –palvelinvirtualisointialusta
- VMware vRealize –automaation ja suorituskyvyn hallintasovellus (VMware vCloud Suite Datasheet n.d.)

5.2 Protokollasimulaattorit

SCADA-protokollien ja niitä käyttävien laitteiden toimintaa voidaan simuloida protokollasimulaattoreilla. Protokollasta ja ohjelmasta riippuen niillä voidaan mallintaa joko Master- tai Slave-laitteiden toimintaa, joissain tapauksissa samalla ohjelmalla voidaan simuloida molempia. Slave-simulaattorilla voidaan mallintaa SCADA-järjestelmän PLC- ja RTU (Remote Terminal Unit) -laitteiden toiminnallisuutta, johon kuuluu tila- ja mittausarvojen kerääminen rekisteriin sekä näiden tietojen välittäminen Master-laitteelle. Näitä arvoja simuloidakseen ne generoivat satunnaisarvoja kunkin muuttujatyypin raja-arvojen puitteissa. Master-simulaattoreiden operointitapa on pääpiirteittäin aina sama; Master-simulaattorilla otetaan yhteys Slave-laitteen IP (Internet Protocol) -osoitteeseen ja määritetään protokollaa käyttävä portti. Master-simulaattorilla määritetään muuttuja, minkä arvoja Slavelta haetaan, jolloin simulaattori kyselee näitä arvoja Slavelta joko automaattisesti käyttäjän määritetyin aikavälein tai manuaalisesti komentoja lähettämällä. Kutakin työssä tutkittavaa SCADA-protokolla käsitellään yksityiskohtaisesti kappaleissa 6-8.

5.3 Metasploit Framework

Metasploit Framework on Metasploit –penetraatiotestausprojektiin kuuluva Open Source –työkalu ja hyökkäysalusta. Se on suunniteltu haavoittuvuuksia hyödyntävän koodin kehittämiseen, testaamiseen ja suorittamiseen haavoittuvuustestauksen ja siihen liittyvän tutkimustyön edistämiseksi. Se sisältää noin 900 valmiiksi integroitua haavoittuvuusmoduulia ja on siten laajin yksittäinen tarjolla oleva kokoelma laadukkaita, hyödynnettäviä haavoittuvuuksia. (Metasploit n.d.)

Metasploit Framework on moduulaarinen, joten se yhdistää haavoittuvuuden, jonka avulla päästään käsiksi kohdejärjestelmään, sekä itse koodin joka kohteeseen pääsyn jälkeen suoritetaan. Sekä haavoittuvuusmoduuleja että kohdejärjestelmässä suoritettavaa koodia (payload) voidaan muokata käyttäjän tarpeiden mukaisesti. Metasploit Frameworkin käyttämistä käsitellään luvussa 10.1.3.

6 OPC

6.1 OPC Classic

OPC on teollisuuden yhteentoimivuusstandardien kokonaisuus, joka määrittelee datakommunikoinnin eri laitevalmistajien laitteiden välillä. Se noudattaa yksinkertaista Client-Server –toimintamallia, jossa OPC-Client noutaa dataa yhdeltä tai useammalta OPC-Serveriltä. OPC-palvelin on käytännössä ohjelma, joka kääntää teollisuuslaitteiden kuten PLC:iden käyttämät protokollat muiden järjestelmien kanssa yhteensopivaksi OPC-protokollaksi. OPC Classic –standardit pohjautuvat Microsoft Windowsin COM/DCOM-teknologioihin, jotka vastaavat datansiirrosta software-rajapintojen välillä. OPC Classic-protokollaan lukeutuu eri datatyyppien käyttöön tarkoitettuja spesifikaatioita, joita ovat muun muassa

- OPC DA (Data Access): Yleisimmin käytetty OPC-spesifikaatio, jota käytetään luettaessa ja kirjoitettaessa reaaliaikaista dataa kuten lukuarvo-, aika- ja laatuinformaatiota. Yleisesti OPC:sta puhuttaessa viitataan DA-spesifikaation toiminnallisuuteen.
- OPC HDA (Historical Data Access): Käytetään käsiteltäessä arkistoitua ja laskelmia sisältävää aggregaattidataa, esimerkiksi tietokantoja.
- OPC A&E (Alarms & Events): Käytetään ympäristön olosuhteiden monitorointiin OPC-Serverillä. Spesifikaatio määrittää tila- ja raja-arvot sekä muuttujat, joiden perusteella voidaan luoda ilmoituksia ja hälytyksiä OPC-Clientille lähetettäväksi. (OPC Foundation n.d.)

6.2 Toimintaperiaate

6.2.1 Yleistä

Tyypillisin OPC-konfiguraatio on yksittäinen Client-Server –yhteys, mutta sen lisäksi OPC:tä voidaan käyttää seuraavissa konfiguraatioissa:

- OPC-aggregointi: OPC-Client yhdistyy useaan OPC-palvelimeen. Tämä mahdollistaa SCADA-informaation keräämisen ja esittämisen usealta lähteeltä.

- OPC-tunnelointi: OPC-Client yhdistyy OPC-palvelimeen verkon ylitse.
- OPC-siltaus: Kaksi OPC-palvelinta yhdistyy toisiinsa jakaakseen dataa sekä ilmoittaakseen arvo- ja tilamuutoksista verkossa.
  (OPC DataHub n.d.)

6.2.2 Datatyypit

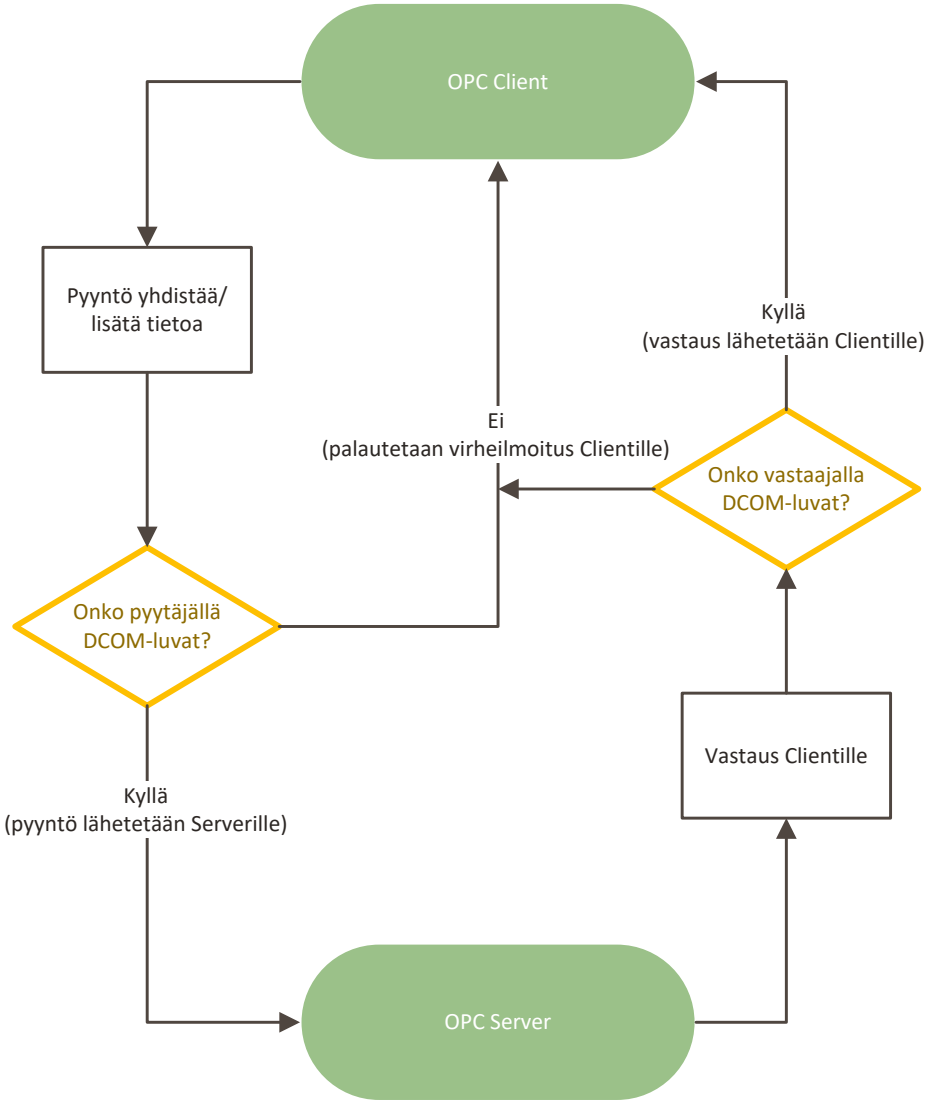
Taulukossa 1 on esitelty OPC-protokollan tukemat dataformaatit, joissa OPC-Client pyytää dataa sekä OPC-Server palauttaa Clientin pyytämää dataa.

Taulukko 1. OPC-protokollan datatyypit

ID	Datatyyppi	Kuvaus
0	VT_EMPTY	Oletusarvo/Tyhjä arvo
2	VT_I2	2-bit. merkattu kokonaisluku
3	VT_I4	4-bit. merkattu kokonaisluku
4	VT_R4	4-bit. reaaliluku
5	VT_R8	8-bit. reaaliluku
6	VT_CY	Valuutta
7	VT_DATE	Päivämäärä
8	VT_BSTR	Teksti
10	VT_ERROR	Virhekoodi
11	VT_BOOL	Boolean (TRUE = -1, FALSE = 0)
17	VT_I1	1-bit. merkitty char-arvo
18	VT_UI1	1-bit. merkitty char-arvo
19	VT_UI2	2-bit. merkitsemätön ko-konaisluku
20	VT_UI4	4-bit. merkitsemätön ko-konaisluku
+8192	VT_ARRAY	Arvojono

6.3 Tietoturvallisuus

OPC Classic –spesifikaatioissa ei ole sisäänrakennettuja tietoturvaominaisuuksia, vaan kaikki tietoturva on delegoitu kommunikaatiosta vastaavalle COM/DCOM-kerrokselle. Sen toiminnallisuuteen kuuluu tarkistaa, millä Windows-käyttäjillä on tarvittavat pääsy- ja käynnistysoikeudet DCOM:ia käyttävissä sovelluksissa joko paikallisella tietokoneella tai paikallisverkon/toimialueen tietokoneilla. DCOM:in turvallisuusasetukset ja pääsyoikeudet ovat sidoksissa kuhunkin tietokoneeseen ja/tai paikallisverkkoon tai toimialueeseen, jossa se sijaitsee. (DCOM – Secure by Default 2009, 5). DCOM:in tietoturvaa ja sen toimintaa on kuvattu vuokaaviona kuviossa 5.



Kuvio 5. DCOM:in tietoturvan toimintalogiikka OPC-protokollassa

6.4 Haavoittuvuudet OPC:ssa

6.4.1 Haavoittuvuudet salasanoissa

Koska OPC tukeutuu Microsoftin autentikointimekanismeihin, heikot salasanat ovat OPC-palvelimien kriittisimpiä heikkouksia. OPC:n edellyttämän DCOM:in autentikointi tapahtuu paikallisesti tai käyttämällä domainin/AD:n jakamia tunnistetietoja (Byres & Peterson 2007.)

6.4.2 Riittämätön lokikirjaus

Windowsin 2000- ja XP-versiot eivät oletuksena pidä lokia DCOM:in yhteyspyynnöistä, SMB (Server Message Block) -sisäänkirjautumisista tai yrityksistä päästä käsiksi järjestelmäobjekteihin. Ilman näiden lokiasetusten käyttöönottoa ei voida mitenkään havaita tunkeutumisyrityksiä tai niiden alkuperää. (Byres & Peterson 2007.)

6.4.3 Rekisterin selaaminen etänä

Windowsin rekisteri on todennäköinen hyökkäyksen kohde, sillä sinne varastoidaan niin paikallisia kuin muualta verkosta peräisin olevia konfiguraatiotietoja. Lukuoikeuksien salliminen rekisteriin etänä mahdollistaa hyökkääjän pääsyn merkittävään määrään tietoa. OPC-protokollan kohdalla hyökkääjä voisi tunnistaa aktiiviset OPC-palvelimet sekä muita asennettuja DCOM-objekteja. (Byres & Peterson 2007.)

6.4.4 Eheydentarkistuksen puute OPC-kommunikaatiossa

DCOM-asetukset eivät oletusarvoisesti sisällä viestien eheyden tarkistusta. Jos OPC-kommunikaatiota pohjustavan verkon infrastruktuuri on murrettu ja hyökkääjä pääsee kuuntelemaan sekä syöttämään verkkoon omaa liikennettä, on tämän mahdollista myös generoida haittaliikennettä palvelimen ja clientin välisen

autentikointivaiheen jälkeen. SMB-protokollalle kehitettyjä MITM-työkaluja muokaten ja käyttäen on mahdollista suorittaa MITM-hyökkäyksiä myös OPC-kommunikaatiota vastaan. (Byres & Peterson 2007.)

6.4.5 CLSID-rekisteriavain

CLSID tai Class Identifier on numeroita ja kirjaimia sisältävä tunnistenumero, jota käytetään eri ohjelmainstanssien tunnistamiseen COM-pohjaisessa verkkoympäristössä. Se sallii Windowsin käyttöjärjestelmien ja ohjelmien tunnistaa eri ohjelmistokomponentteja yksilöllisellä rekisteriavaimella pelkän nimen sijaan. Tietyt OPC:tä käyttävät Client-sovellukset pystyvät näkemään verkon aktiivisten OPC-komponenttien CLSID-avaimen ilman, että ovat näihin suoraan yhteydessä. CLSID:stä voi olla hyökkääjälle hyötyä sopivan kohdelaitteen tunnistamisessa, sillä pelkän verkkohaun avulla hyökkääjä voi onnistua selvittämään esimerkiksi komponentin valmistajan, mallin ja/tai nimitiedot. (What is CLSID Registy Key? n.d.)

6.5 OPC UA (Unified Architecture)

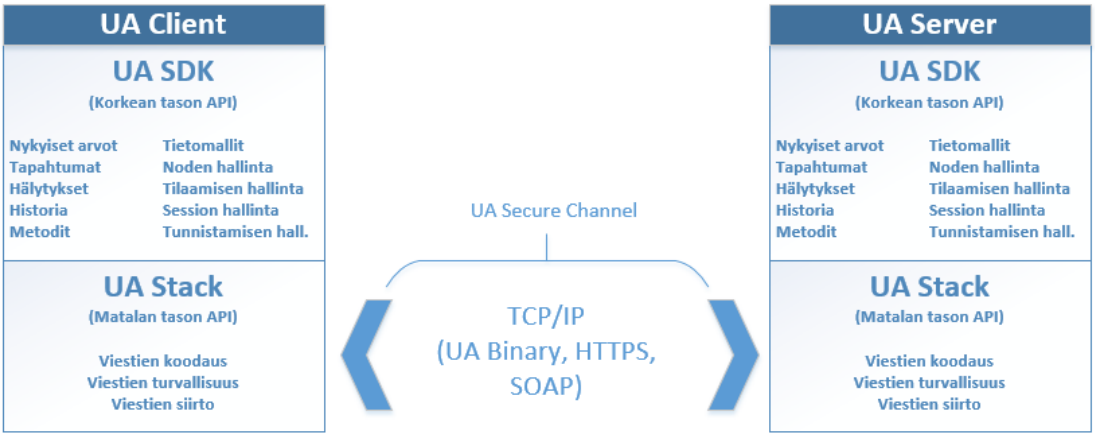
6.5.1 Yleistä

OPC Unified Architecture on vuonna 2008 julkistettu uuden sukupolven OPC-protokolla, jossa on yhdistetty eri OPC Classic -spesifikaatioiden toiminnallisuudet ja Windows-pohjainen DCOM-kommunikaatio on korvattu palvelupohjaisella ja alustariippumattomalla kommunikaatio-arkkitehtuurilla. Muun muassa näiden muutosten avulla saavutettiin seuraavat kehitystavoitteet OPC Classic –versioihin verrattuna:

- Toiminnallinen vastaavuus (DA:n, HDA:n ja A&E:n toiminnallisuus yhdistetty, taaksepäin yhteensopiva OPC Classic -sovellusten kanssa)
- Alustariippumattomuus (Microsoftin DCOM:ista siirrytty monikerroksiseen palvelupohjaiseen kommunikaatio-arkkitehtuuriin)
- Tietoturvallisuus (lisätty tiedon salaus, autentikointi ja auditointi)
- Laajennettavuus (mahdollista lisätä ominaisuuksia vaikuttamatta toiminnallisuuteen)
- Kattava tiedonmallinnus (monimutkaisen informaation määritykseen)

Toistaiseksi OPC UA on enimmäkseen käytössä sulautetuille järjestelmille alustariippumattomuutensa takia sekä välittäjäprotokollana OPC Classic –järjestelmille, jotka ovat edelleen laajalti käytössä. (OPC Unified Architecture n.d)

Kuvio 6 havainnollistaa OPC UA:n toiminnallisuutta sen toimintaketjussa. Kerrostetussa mallissa Stack hoitaa protokollan kommunikaation laitteiden välillä, kun taas SDK vastaa OPC UA:n perustoiminnoista ja objektimäärityksistä.



Kuvio 6. OPC UA:n toiminnallisuus kerrosmallissa

6.5.2 Tietoturvallisuus

OPC UA sisältää vanhempaan Classic-spesikaatioon verrattuna useita julkisen avaimen infrastruktuuriin perustuvia tietoturvaominaisuuksia, joita käytetään sekä OPC-Clienteilla että –Servereillä:

- Clientit ja Serverit tunnistetaan käyttämällä X.509-sertifikaatteja, jotka vaihdetaan kolmivaiheisessa kättelyssä ennen kommunikoinnin aloittamista osapuolien välillä
- Käyttäjien autentikointi tapahtuu joko X.509-sertifikaateilla, käyttäjänimi-salasana –yhdistelmällä tai ulkoisilla tiketeillä
- Vaihtoehtoisesti voidaan ottaa käyttöön yksittäisten OPC-viestien allekirjoitus ja salaus, johon on käytettävissä useita vaihtoehtoisia salausalgoritmeja

Binääri- ja HTTPS (Hypertext Transfer Protocol Secure) -kommunikaatio tapahtuu yhdellä konfiguroitavalla TCP/IP-portilla, joka mahdollistaa TLS (Transport Layer Security) -salausprotokollan käytön. Lisäksi OPC UA on taaksepäin yhteensopiva Classic-spesifikaatioiden kanssa myös tietoturvallisuuden suhteen, sillä DCOM-pohjainen kommunikaatio voidaan tunneloida OPC UA:n Proxy- ja Wrapper-komponenttien, kuten kaupallisen UA Gatewayn avulla. OPC UA voi myös kommunikoida minkä tahansa http tai UA TCP-portin kautta, minkä ansiosta OPC UA-laitteet voidaan yhdistää turvallisesti VPN (Virtual Private Network) -yhteyden yli ja sallia käytettävät portit palomuureissa muodostaessa client-to-server –etäyhteyksiä. (Unified Architecture n.d.)

6.5.3 Haavoittuvuudet OPC UA:ssa

Itseallekirjoitetut X.509 –sertifikaatit

OPC UA –spesifikaation mukaan Client- ja Server-sovellusten tulee asennuksen yhteydessä luoda näiden itsensä allekirjoittamat sertifikaatit. Kuka tai mikä osapuoli tahansa voi tehdä tämän, joten tunnistamisen perustaminen itseallekirjoitettuihin sertifikaatteihin on sama kuin kysyisi henkilöllisyyden varmentamisen yhteydessä henkilöltä itseltään, onko hän kuka väittää olevansa. (Peterson 2008.)

Spesifikaatio sallii OPC UA:n Client- ja Server-sovellusten hyväksyä niin itseallekirjoitetut kuin muutkin sertifikaatit varmentamatta näiden luotettavuutta kolmannen osapuolen avulla. Tämän seurauksena OPC UA-palvelimen voi vaarantaa kuka tahansa, jolla on pääsy OPC UA:n Client-sovellukseen, vaikka palvelin vaatisikin tietojen salausta ja vastapuolen allekirjoituksia. Hyökkääjä voisi täten yksinkertaisesti hyödyntää salattua ja autentikoitua Secure Channel –yhteyttä välittääkseen hyökkäyksiä palvelimelle. (Peterson 2008.)

Secure Channel –asetukset

Secure Channel –yhteyttä käytetään yhdistämään OPC UA –palvelin Discovery-palvelimeen, joka puolestaan mahdollistaa saatavilla olevien OPC UA-palvelimen näkymisen OPC UA –clienteille. Jos Secure Channelin tietoturva-asetuksissa ”Sign and Encrypt” –asetus on poissa käytöstä, sallitaan OPC UA Clientien yhdistää myös ei-

sertifioituihin OPC UA –palvelimiin. Tämän ominaisuuden ansiosta hyökkääjän voisi onnistua houkuttelemaan Clienteja yhdistämään tämän hallussa olevaan OPC UA –palvelimeen. (Peterson 2008.)

7 MODBUS TCP

7.1 Yleistä

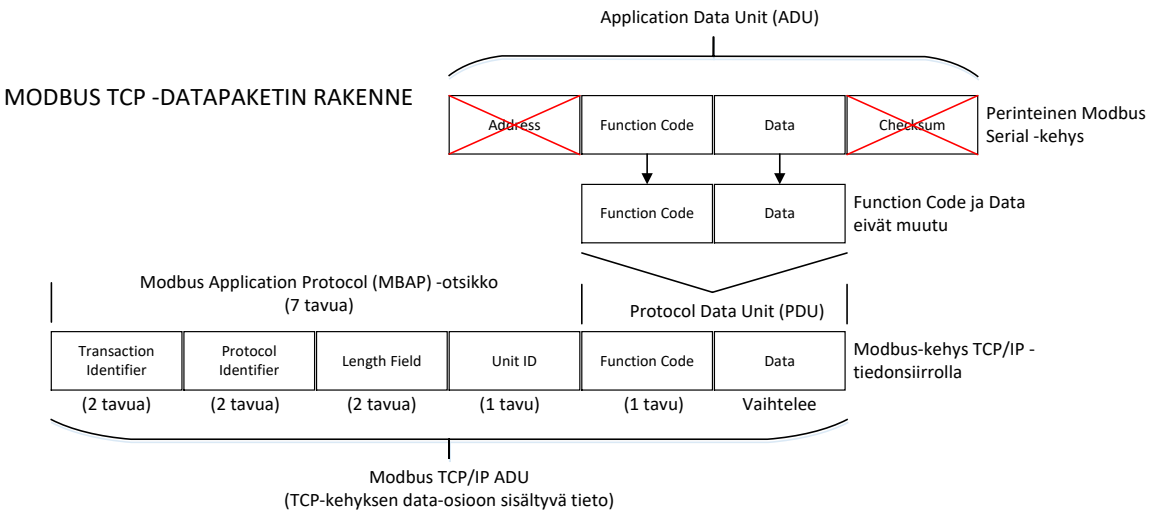
Modbus TCP on tuotantoympäristöissä käytettävä avoin telekommunikointiprotokolla, jolla voidaan yhdistää verkon kytkin (Server) ja verkkoon kytketty HMI-laite (Human Machine Interface) tai muu IED-laite (Intelligent Electronic Device) kuten kontrolleri, reititin, IP-puhelin, langaton yhteyspiste ja saada ne kommunikoimaan keskenään. Kytkin paketoi Request-pyyntöviestin tai Response-vastausviestin ADU-paketiksi (Application Data Unit), jonka jälkeen Client-laite lähettää viestin kytkimen TCP-porttiin, joka on oletuksena 502. Modbus TCP-protokolla on käytännössä perinteinen Modbus RTU-protokolla, sillä erotuksella, että Modbus-viestien siirtämiseen käytetään Ethernet-, TCP- ja IP-protokollia vanhempien Serial-yhteyksien sijaan. Lähetettävä informaatio siirtyy Modbus-kehyksessä TCP:lle, joka varmistaa, että kaikki datapaketit vastaanotetaan oikein. Tämän jälkeen perinteisen TCP-paketin tietoihin lisätään lisädatana Modbus-kehyksen sisältämä informaatio. Lopuksi IP-protokolla muodostaa saaduista tiedoista IP-paketin ja lähettää sen vastaanottajalle. (Introduction to Modbus TCP/IP 2005, 3-4.)

7.2 Toimintaperiaate

7.2.1 Yleistä

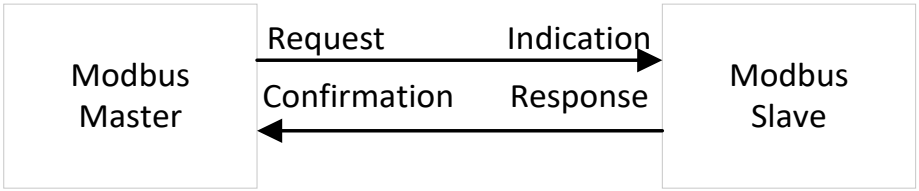
Modbus TCP-protokolla yhdistää perinteisen Modbus-datakehyksen TCP-kehykseen ilman Modbus-protokollan käyttämää tarkistussummaa. Tietojen eheys tarkistetaan sen sijaan hyödyntämällä TCP/IP-protokollien tarkistussummia. Lisäksi perinteisen Modbus-kehyksen osoitekenttä korvataan Unit Identifier –tunnistekentällä Modbus TCP -kehyksessä. Sitä käytetään kommunikoimaan verkon siltaavien ja reitittävien laitteiden kanssa, jotka käyttävät yksittäistä IP-osoitetta. Kun Modbus-payload yhdistetään TCP-kehykseen, ylimääräiset bitit kuljetetaan MBAP-otsikossa, jolloin pyyntö- ja vastausviestien vastaanottaja tunnistaa yksittäisen viestin kokonaispituuden, vaikka viesti olisi jaettu useampaan pakettiin sen kuljetusta varten. Käyttämällä Ethernet-tason 32-bittistä CRC (Cyclic Redundancy Check) -

tarkistussummaa yhdessä tarkoin määriteltyjä viestin pituuksia varmistetaan, että havaitsemattomien bittivirheiden todennäköisyys on Modbus-kommunikoinnissa äärimmäisen pieni. (Introduction to Modbus TCP/IP 2005, 4-5). Modbus TCP-protokollan kehysrakennetta on havainnollistettu kuviossa 7.



Kuvio 7. Modbus TCP -kehyksen rakentuminen

Modbus TCP-protokolla toimii Client/Server (Master/Slave) –periaatteella. Client-laitteet muodostavat yhteyden yhdelle tai useammalle verkon Server-laitteelle, joilta ne pyytävät haluttuja tietoja. Verkon Modbus-laitteet keskustelevat käyttäen neljää eri viestityyppiä: Request, Confirmation, Indication ja Response. Modbus TCP-liikennettä kuunnellaan ja vastaanotetaan Modbus-protokollalle varatussa portissa 502. (Modbus Messaging on TCP/IP Implementation Guide 2002, 6). Client (Master)- ja Server (Slave) -laitteiden välistä kommunikointia on havainnollistettu kuviossa 8.



Kuvio 8. Modbus-laitteiden kommunikointi ja viestityypit

Kommunikointiin Modbus-laitteet käyttävät seuraavantyyppisiä viestejä:

- Request: Client aloittaa kommunikoinnin (pyytää tietoja) yhden tai useamman Server-laitteen kanssa.
- Indication: Viesti, jolla Server kuittaa Clientin lähettämän Request-pyyntöviestin vastaanotetuksi.
- Response: Viesti, jolla Server vastaa (lähettää tietoja) Clientin lähettämään Request-viestiin.
- Confirmation: Server ilmoittaa vastaanottaneensa Serverin Response-vastausviestin.

Käyttötarkoituksesta riippuen Modbus-protokollaa voidaan soveltaa myös laajemmissa verkkokokonaisuuksissa eri verkkojen välillä. Tällöin Modbus-protokollaa käyttävä verkko koostuu Master- ja Slave-laitteiden lisäksi siltaavista tai reitittävistä laitteista, joilla voidaan muodostaa eri verkkojen välisiä Serial-yhteyksiä Master- ja Slave-laitteiden välille. (Modbus Messaging on TCP/IP Implementation Guide 2002, 8). Kuviossa 3 on esitetty Modbus TCP- protokollan toiminta-arkkitehtuuria.

7.2.2 Datatyypit

Modbus TCP:n datatyypit koostuvat diskreeteistä Input- ja Output-arvoista (ON/OFF) sekä 16-bittisistä rekisteriarvoista syötteitä datan tallennusta ja lukemista varten (Modbus Messaging on TCP/IP Implementation Guide 2002, 13). Taulukossa 2 on esitetty Modbus TCP –protokollan eri datatyypit, joita Modbus TCP:tä käyttävät laitteet voivat tulkita ja käyttää kommunikointiin.

Taulukko 2. Modbus TCP:n datatyypit

Nimi	Tyyppi	Käyttö
Discrete Input	Yksittäinen bitti	Read-only
Discrete Output (Coils)	Yksittäinen bitti	Read-write
Input Registers	16-bittinen sana	Read-only
Holding Registers	16-bittinen sana	Read-write

7.3 Haavoittuvuudet

7.3.1 Protokollaskannerit

Modbus TCP-protokollaa varten on kehitetty useita rekisteriskannereita, jotka kykenevät hakemaan tietoja Modbus-laitteiden rekisteritiedoista ja tilatiedoista. Näitä tietoja voidaan tarkastella muun muassa binääreinä, heksadesimaaleina, Int16/32, Uint16/32 sekä muilla formaateilla. Lisäksi skannerit voivat havaita Modbus-laitteiden IP-osoitteet verkosta, jos ne eivät ole entuudestaan tiedossa.

7.3.2 Luvaton komentojen suoritus

Autentikoinnin puute Master- ja Slave-solmujen välillä mahdollistaa hyökkääjän lähettää väärennettyjä Modbus-viestejä useille Slave-laitteille. Suorittaakseen tämän tyypin hyökkäyksen, hyökkääjälle tulee olla pääsy verkkoon, jossa Master- tai Slave-laitteet sijaitsevat. (Audit: TCP Modbus – Unauthorized Write/Read Request n.d.)

7.3.3 Palvelunestohyökkäykset

Hyökkääjä voi esimerkiksi esittää Modbus –Slave-laitetta ja ”pommittaa” sen roolissa Master-laitteita TCP SYN-yhteyspyynnöillä ja muilla turhilla viesteillä, joiden ansiosta näiden prosessointikyky ylittyy. Tästä seurauksena ne voivat laitteesta riippuen joko lakata vastaamasta komentoihin, palauttaa kyselyiden yhteydessä vääriä arvoja tai käynnistyä kokonaan uudelleen. (Chen, Pattanaik, Goulart, Butler-Purry & Kundur 2014, 5.)

7.3.4 Man-in-the-Middle –hyökkäykset

Eheyden tarkistuksen, salauksen ja autentikoinnin puuttuminen Modbus TCP-protokollassa tekee siitä haavoittuvan kaikentyyppisiä Man-in-the-Middle –hyökkäyksiä vastaan. Tietoturvaominaisuuksien puuttuminen protokollamäärityksistä mahdollistaa kohdeverkkoon päässeen hyökkääjän kaapata, muokata ja toistaa verkon oikeita Modbus-viestejä tai lähettää väärennettyjä viestejä slave-laitteille.

Salauksen puuttuessa Modbus TCP-viestien sisältö on hyökkääjän helposti luettavissa. Havaituksi tulemisen välttämiseksi ja kommunikaatioyhteyden

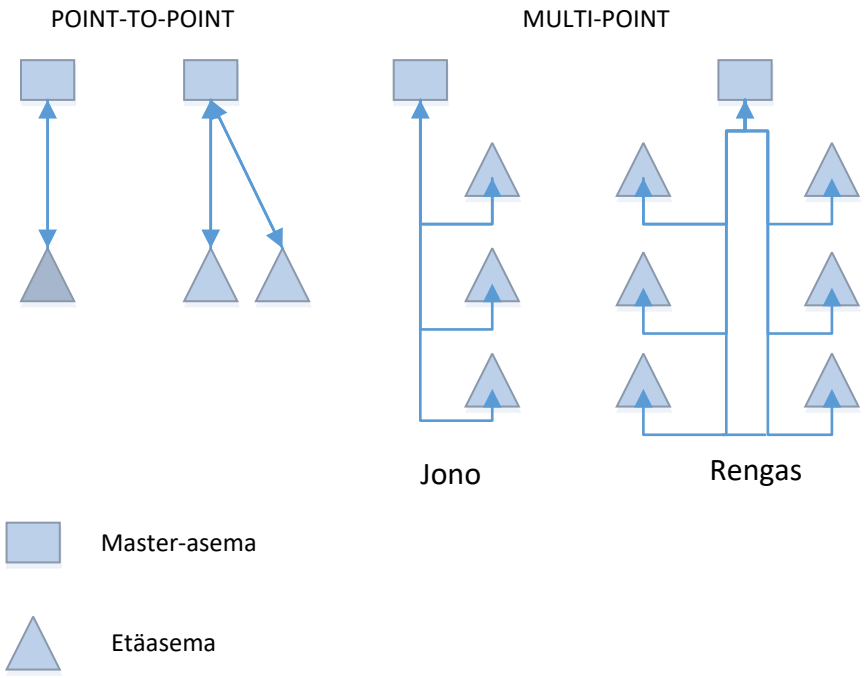
ylläpitämiseksi hyökkääjä kuitenkin välittää osapuolien lähettämän liikenteen sen tarkoitetulle vastaanottajalle. Hyökkääjä jakaa alkuperäisen yhteyden kahteen uuteen yhteyteen (Master/Hyökkääjä) sekä (Hyökkääjä/Slave).

8 IEC 60870-5-104

8.1 Yleistä

IEC Technical Commitee 57:n kehittämä IEC 60870-5-104 -protokolla on IEC 60870-5 -telekontrolli- ja –kommunikaatioprotokollan laajennos, joka laajentaa alkuperäisen protokollan toiminnallisuutta telekontrollilaittelle ja koodattua bittidataa hyödyntäville TCP/IP-järjestelmille. Se mahdollistaa Master-ohjausaseman ja Slave-etäaseman kommunikoinnin perinteisissä TCP/IP-verkoissa. Se on kehitetty monitoroimaan ja ohjaamaan TCP/IP-verkon avulla hallittavia prosesseja, jotka kattavat maantieteellisesti laajoja alueita. (IEC 60870-5-104 Master Protocol Configuration Manual 2006.)

IEC 104 –protokolla toimii joko Point-to-Point tai Multi-Point –topologiassa. Point-to-Point –konfiguraatioissa joko Master tai Outstation-etäasema voi lähettää viestejä. Multi-Point –konfiguraatioissa puolestaan verkon Outstation-laitteet jakavat viestintäkanavan Masterille päin, joten ne voivat lähettää viestejä vain yksi kerrallaan. T104 -protokollaa käytettäessä Point-to-Point –konfiguraatiossa viestinnän voi aloittaa kumpi tahansa laitteista. Multi-Point –konfiguraatiossa viestinnän voi aloittaa ainoastaan Master-laite. IEC 60870-5-104 -protokollan eri topologiat on esitetty kuviossa 9.

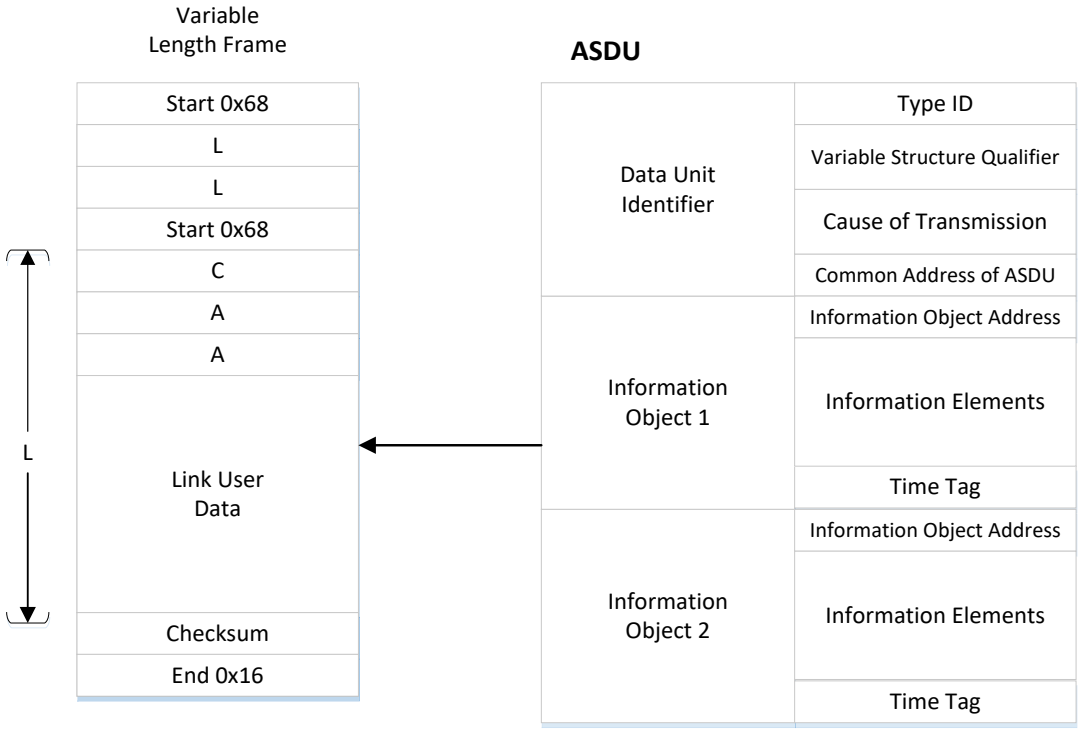


Kuvio 9. IEC 60860-5-104:n topologiat

8.2 Viestirakenne

IEC 60860-5-104:n viesti muodostuu APCI (Application Protocol Control Information) -ohjauskehyksestä (Application Service Control Information) sekä ASDU-datakehyksestä (Application Service Data Unit). APCI sisältää data link layer –kehyksen kantaman linkkiosoitteen ja ohjausinformaation sekä lipun, joka ilmoittaa Class 1 –datan saatavuuden ASDU-kehyksessä. Kukin APCI-kehys voi kantaa enintään yhden ASDU:n. (IEC 60870-5-104: Telegram structure n.d.)

ASDU puolestaan sisältää sovellustason datan, kuten laitteen osoitetiedot ja laitteen mittaamat lukemat, jotka sisältyvät ASDU:n tieto-objektikenttiin (Maynard, P., McLaughlin, K. & Haberler, B. 2014.) Kuviossa 10 nähdään datalinkkikehyksen rakenne sekä sen varsinaista mittausdataa kuljettavan ASDU:n rakenne tieto-objektikenttineen.



Kuvio 10. IEC 60870-5-104:n datalinkkikehyksen ja ASDU:n rakenne

8.3 Dataobjektit

ASDU:ssa kuljetettavat dataobjektit voivat olla desimaali-, kokonaisluku-, boolean-, tai bittijonoarvoja mitattavasta datasta ja sille sopivasta dataobjektista riippuen. IEC 60870-5-104 –protokollan tukemat yleisimmät dataobjektit selityksineen ovat listattu taulukossa 3. (Standard IEC 60870-5-104 Data Types n.d.) Täydellinen listaus IEC 60870-5-104 –protokollan tukemista viesteistä löytyvät liitteestä 1.

Taulukko 3. IEC 60870-5-104:n dataobjektit

Tyyppi-ID	Kuvaus
1–2, 30	Single indication without / with 24-bit timestamps
3–4, 31	Double indication without / with 24-bit timestamps
5–6, 32	Step position information without / with 24-bit timestamps
7–8, 33	Bit-string of 32-bit without / with timestamps
9–14, 34-36	Measured value (normalized/scaled/short floating point without/with timestamps)
15–16. 37	Integrated totals (counters) without / with timestamps

17-21, 39-40	Packed events (start & tripping) of protection equipment

8.4 Tietoturvallisuus

Modbus TCP:n tavoin, myös IEC-60870-5-104 –protokolla on suora porttaus vanhoista Serial-linkeistä TCP/IP:hen. Tästä syystä protokollaan itsessään ei sisälly mitään sisäänrakennettuja tietoturvaominaisuuksia kuten autentikaatiota tai tiedonsalausta. Se hyödyntää whitelisting-periaatetta Slave-laitteella, toisin sanoen Slave-laite hyväksyy yhteyden muodostamisen vain ennalta määritettyihin IP-osoitteisiin. ARP-tietoja väärentämällä on kuitenkin mahdollista esiintyä esimerkiksi verkon Master-laitteena, kun hyökkääjän MAC-osoite yhdistetään Slave-laitteen ARP-tiedoissa vastaamaan oikean Master-laitteen IP-osoitetta, jolloin kaikki oikealle Master-laitteelle ohjatut paketit saapuvat hyökkäävälle koneelle. (Maynard, P., McLaughlin, K. & Haberler, B. 2014.)

8.5 Haavoittuvuudet

Kuten Modbus TCP:n ja useiden muiden SCADA-protokollien tapauksessa, tietoturvaominaisuuksien puuttuminen datalinkki- ja sovelluskerrokselta tekee myös IEC-60870-5-104:stä haavoittuvan monen tyyppisille hyökkäyksille kuten spooffaus-, salakuuntelu-, datanmuokkaus- ja toistohyökkäyksille. MITM-tyyppiset hyökkäykset ovat tästä syystä poikkeuksellisen tehokkaita, sillä IEC 104-laitteet eivät autentikointi- ja verifiointiominaisuuksien puuttuessa pysty havaitsemaan tunnistautuneita kommunikaatio-osapuolia eikä IEC 104 –datapaketteihin tehtyjä muutoksia. (Maynard, P., McLaughlin, K. & Haberler, B. 2014.)

9 HAAVOITTUVUUSTESTAUSTEN SUUNNITTELU

9.1 Toteutettavat haavoittuvuustestaukset

Koska tässä työssä käsitellään protokollia, joita koskevat pääpiirteittäin samoista ominaisuuksista (tai niiden puuttumisesta) johtuvat haavoittuvuudet, tullaan eri protokollien kohdalla tekemään erityyppisiä haavoittuvuustestauksia aina oikean hyökkäyskohteen tunnistamisesta kohteen suoraan vaikuttamiseen Man-in-the-Middle –hyökkäyksin.

9.1.1 OPC: Hyökkäyskohteen tunnistaminen

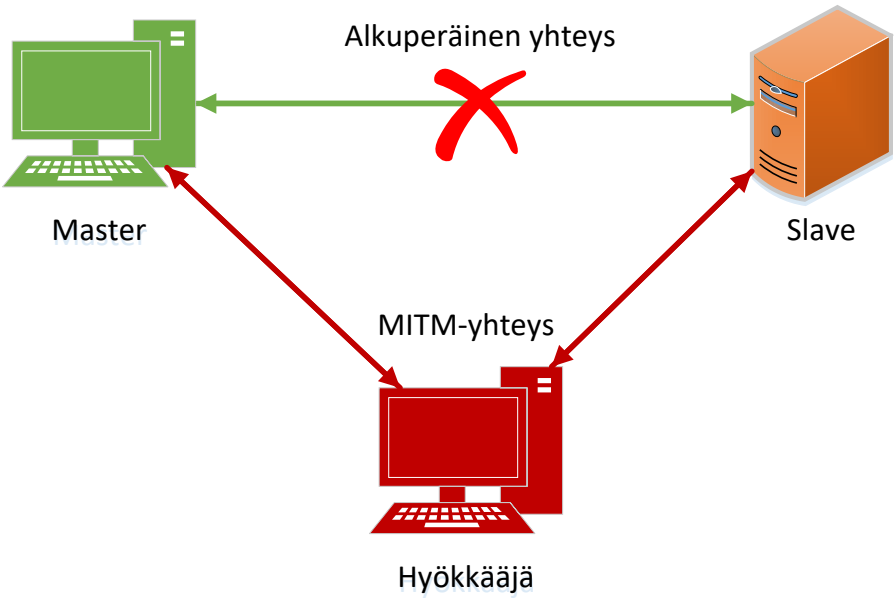
Kuten kyberhyökkäyksen tyypillisiä vaiheita kuvaavassa luvussa 3.2 todettiin, jokainen kyberhyökkäys lähtee liikkeelle kohteen tunnistamisesta. Tämän hyökkäysvaiheen testaukseen käytetään OPC-protokollaa, joka Windowsin DCOM-teknologiaan pohjautuen jakaa lähiverkkoon kuuluville osapuolille broadcast-liikenteenä poikkeuksellisen paljon informaatiota verkon resursseista (Peterson 2008.)

OPC-protokollan kohdalla ei ole tarpeen luoda monimutkaista usean laitteen testausympäristöä, koska testataan vain hyökkääjän käytössä olevia keinoja kerätä verkosta informaatiota, jonka avulla tämä voisi tunnistaa verkosta oikean kohdelaitteen varsinaista hyökkäystä varten. Näin ollen testaus tehdään yhdellä Windows Server 2008 R2-koneella, jossa OPC-palvelimen toimintaa simuloidaan MatrikonOPC:n Server for Simulation-protokollasimulaattorilla. Samalla koneella testataan ohjelmia ja työkaluja, jotka kykenevät havaitsemaan verkon OPC-komponentit, kuten käynnissä olevan paikallisen Matrikonin OPC-palvelimen.

Ideaalinen kohdelaite hyökkäykselle voi olla esimerkiksi OPC-palvelin, johon vaikuttamalla hyökkääjä voisi saada koko OPC-järjestelmän otettua haltuun. Näillä ohjelmilla testataan, mitä informaatiota koneen OPC-komponenteista saadaan irti ja voitaisiinko saatujen tietojen perusteella päätellä kyseessä olevan OPC-palvelin.

9.1.2 Modbus TCP: Komennonsyöttöhyökkäys

Modbus TCP-protokollan haavoittuvuutta MITM-hyökkäykselle testataan kuvion 11 mukaisessa Master-Slave –tyyppisessä testausympäristössä, jonka kommunikointiin vaikutetaan kolmannen osapuolen, hyökkääjän, toimesta. Haavoittuvuustestauksen tavoitteena on salakuunnella Modbus-laitteiden välistä liikennettä sekä päästä Modbus-järjestelmän ulkopuolisella laitteella käsiksi coil- ja rekisteriarvoihin, joita Master- ja Slave-laite vaihtavat keskenään. Lisäksi näitä arvoja pyritään muokkaamaan järjestelmän ulkoa käsin niin sanotun command injection –hyökkäyksen avulla. MITM-hyökkäyksen seurauksena tapahtuvaa muutosta dataliikenteessä on havainnollistettu kuviossa 11.

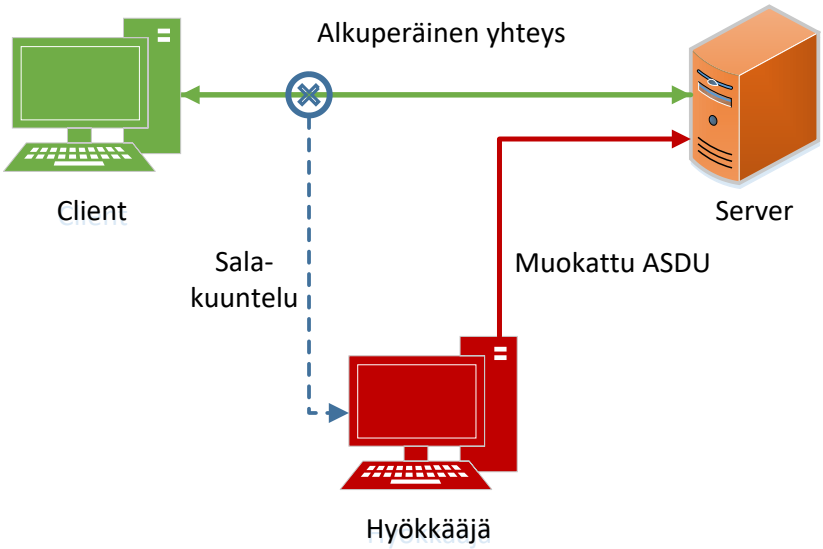


Kuvio 11. Man-in-the-Middle–hyökkäys

9.1.3 IEC 60860-5-104: Modifiointihyökkäys

IEC 60870-5-104 –protokollan kohdalla testataan sen haavoittuvuutta modifiointihyökkäyksille, joissa hyökkääjä salakuuntelee ja väärentää suoraan laitteiden välisiä viestejä huijatakseen Master-laitetta. Tämän hyökkäystekniikan käyttämisen tekee mahdolliseksi se, ettei IEC 104 –protokolla muiden työssä käsiteltyjen SCADA-protokollien tavoin sisällä tietoturvaominaisuuksia kuten tiedon

eheyden tarkistamista tai autentikointia. Tästä johtuen Master-laite ei havaitse viestien sisällössä tapahtuneita muutoksia niitä vastaanottaessaan. Modifiointihyökkäyksen prosessia on havainnollistettu kuviossa 12.

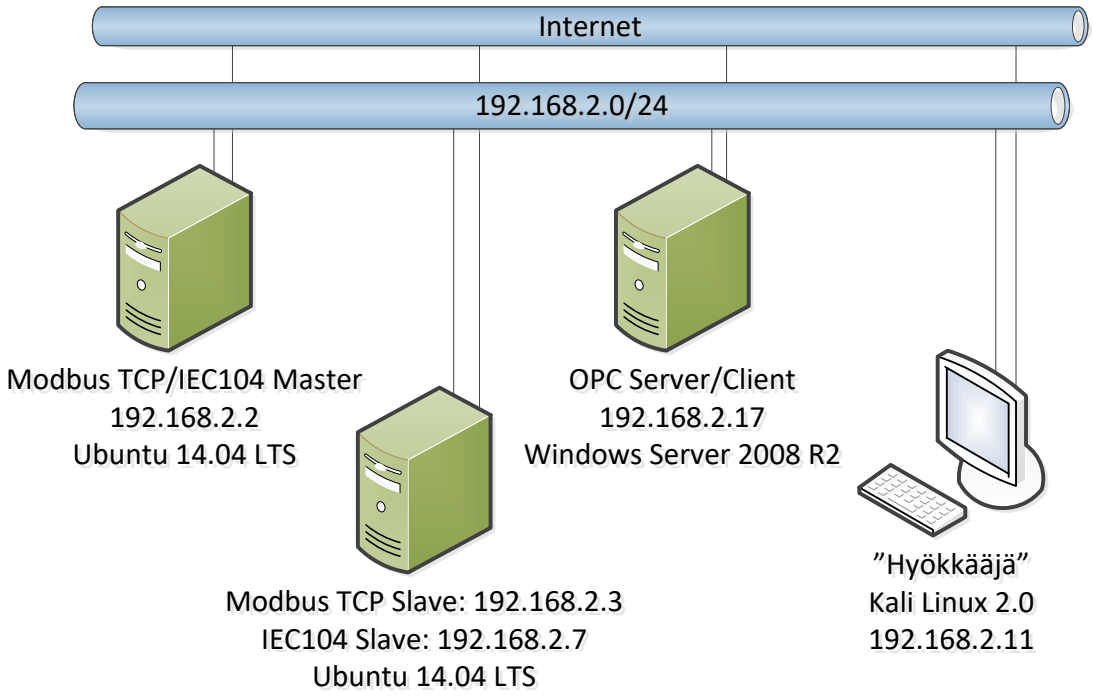


Kuvio 12. Modifiointihyökkäys

10 HAAVOITTUVUUSTESTAUSTEN TOTEUTUS

10.1 Testausympäristö

Testausympäristö koostuu neljästä virtuaalikoneesta, joilla simuloidaan eri protokollien Server- ja Client-toiminnallisuuksia. Samassa 192.168.2.0 –aliverkossa simulointikoneiden kanssa on Kali Linux –käyttöjärjestelmää käyttävä kone, jolla mallinnetaan hyökkääjää. Kali Linux –koneelta käsin Server- ja Client-laitteet altistetaan luvuissa 10.2 – 10.4 kuvailluille hyökkäyksille. Testausympäristöä on havainnollistettu kuviossa 13.



Kuvio 13. Työn testausympäristö

10.2 OPC: Hyökkäyskohteen tunnistaminen

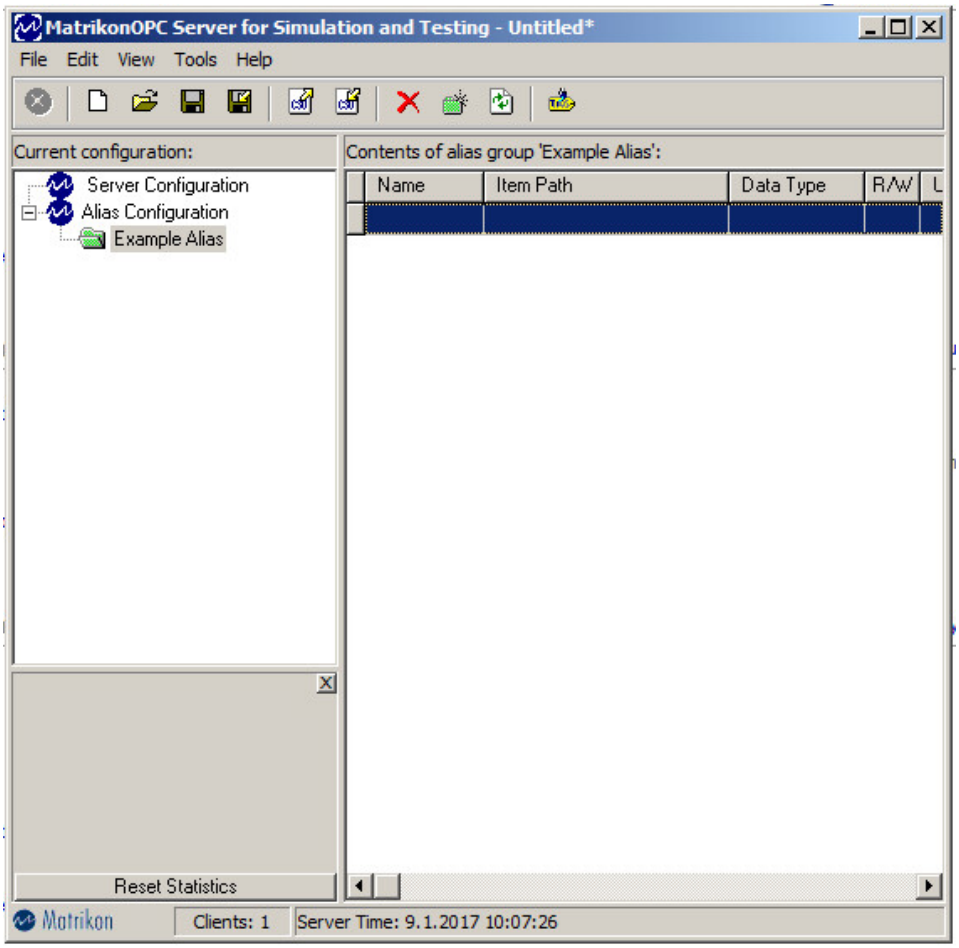
OPC-protokollan haavoittuvuustestauksen tavoitteena on selvittää, kuinka paljon ja minkälaatuista informaatiota OPC-palvelimesta saadaan irti ilman, että yhdistetään Client-puolen ohjelmalla OPC-palvelimeen. Näin jäljitellään hyökkääjää, joka käyttää OPC Client-sovellusta verkon OPC-resurssien vakoilutarkoituksiin ja koittaa pysytellä havaitsemattomana. Itse protokollasimulaattoreiden toiminnallisuuksia ei käsitellä

tässä kappaleessa, sillä ne ovat pääperiaatteiltaan samanlaisia muiden protokollien toimintaa mallintavien simulaattorien kanssa, eivätkä sisällä oleellista tietoa testauksen tavoitteeseen nähden.

OPC-protokollan rekisteripohjaisia haavoittuvuuksia testataan yhdellä Windows Server 2008 R2-pohjaisella tietokoneella, joka toimii OPC-palvelimena. OPC-palvelimen toiminnallisuudesta vastaa MatrikonOPC:n Server for Simulation –ohjelma. Kun OPC-palvelin on toiminnassa ja valmis yhdistettäväksi OPC Clienteihin, tarkastellaan mitä tietoa siitä näkyy muualle sisäverkkoon. Tämä tehdään testaamalla erilaisia vapaasti saatavilla olevia OPC Client- ja Explorer –ohjelmia, jotka ensin skannaavat verkon saatavilla olevat OPC-resurssit, jonka jälkeen niihin voi halutessaan yhdistää.

10.2.1 OPC-palvelin

OPC-palvelin luodaan käynnistämällä MatrikonOPC:n simulaattoriohjelma. Ohjelman käynnistyttyä palvelin on toimintakunnossa ja siihen voidaan ottaa yhteys Client-ohjelmilla. Käynnissä oleva OPC-palvelimen käyttöliittymä on esitetty kuviossa 14.

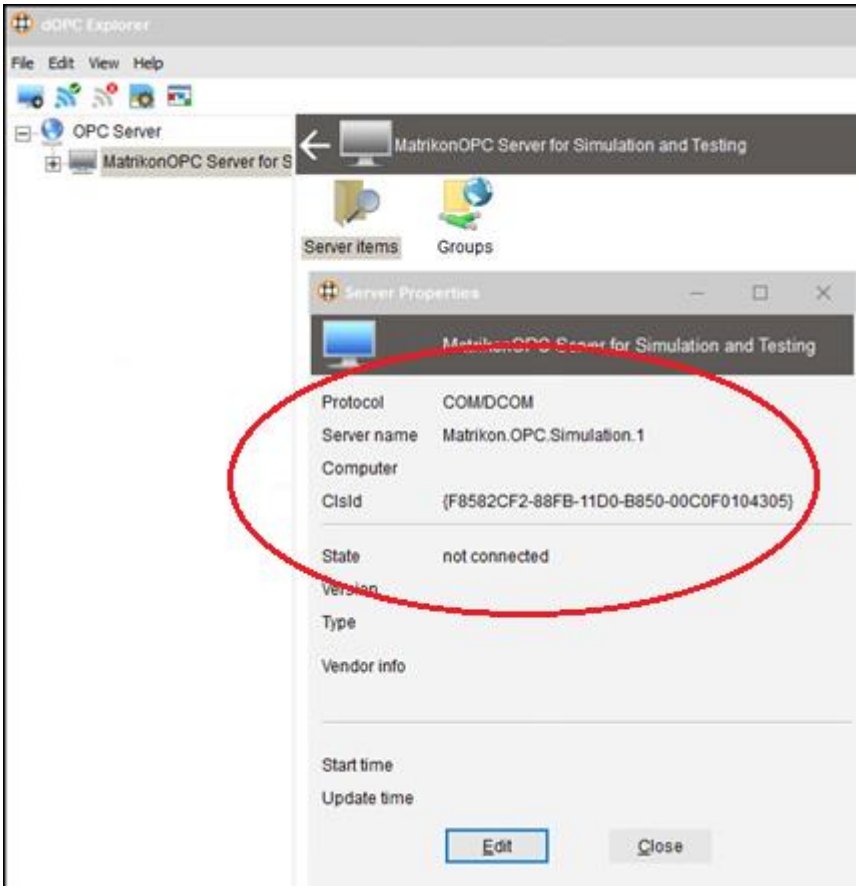


Kuvio 14. Näkymä OPC-palvelinsimulaattorista

10.2.2 OPC Client –puoli

MatrikonOPC Explorer & Kassl dOPC Explorer

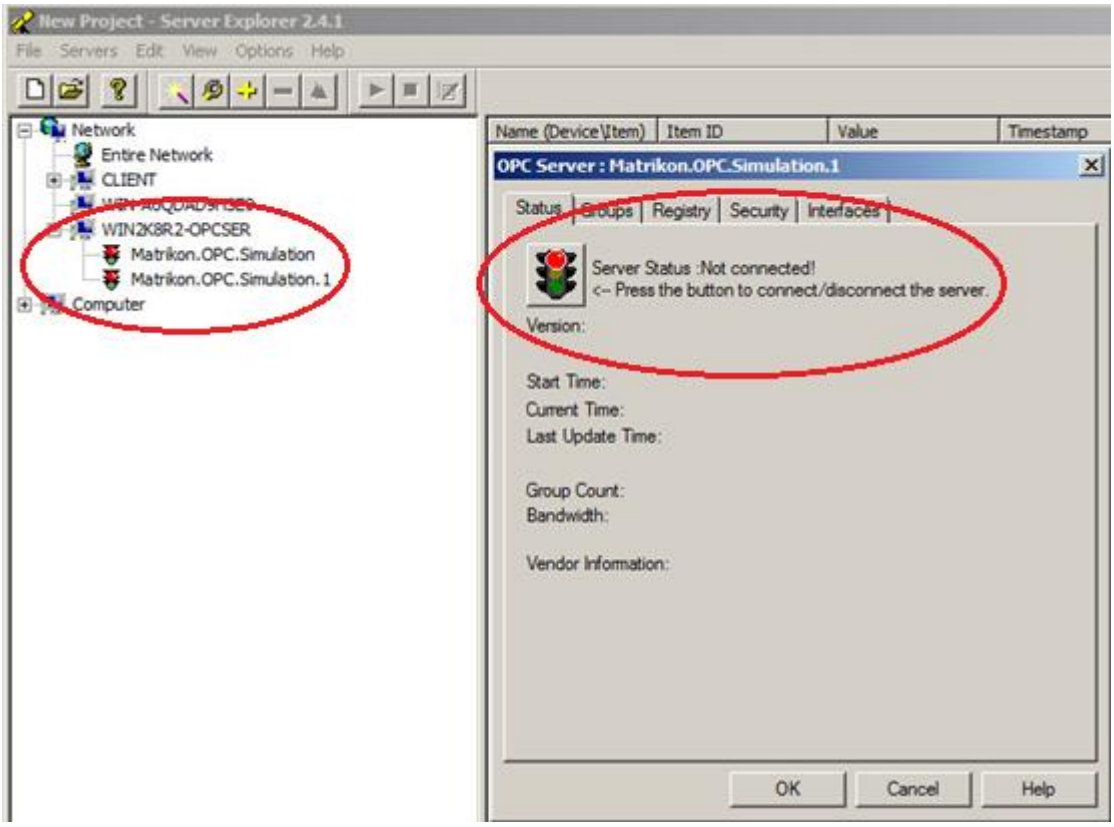
OPC-palvelimen ollessa käynnissä testattiin, mitä tietoa Client-ohjelmat näkevät siitä. Koska MatrikonOPC –palvelinsimulaattorin mukana tuli Explorer-sovellus, oli luontevaa kokeilla ensin sitä. Lisäksi asennettiin Kassl:n dOPC Explorer. Molempien ohjelmien tapauksessa tulos oli sama; OPC-palvelimeen yhdistämättä siitä näkyy lähiverkkoon käytettävä protokolla, palvelimen nimi ”Matrikon.OPC.Simulation.1” ja CLSID-rekisteriavain. Nämä OPC-palvelimen tiedot dOPC Explorer –client-ohjelmasta tarkasteltuna näkyvät kuviossa 15.



Kuvio 15. OPC-palvelimen tiedot dOPC Explorer -clientista tarkasteltuna

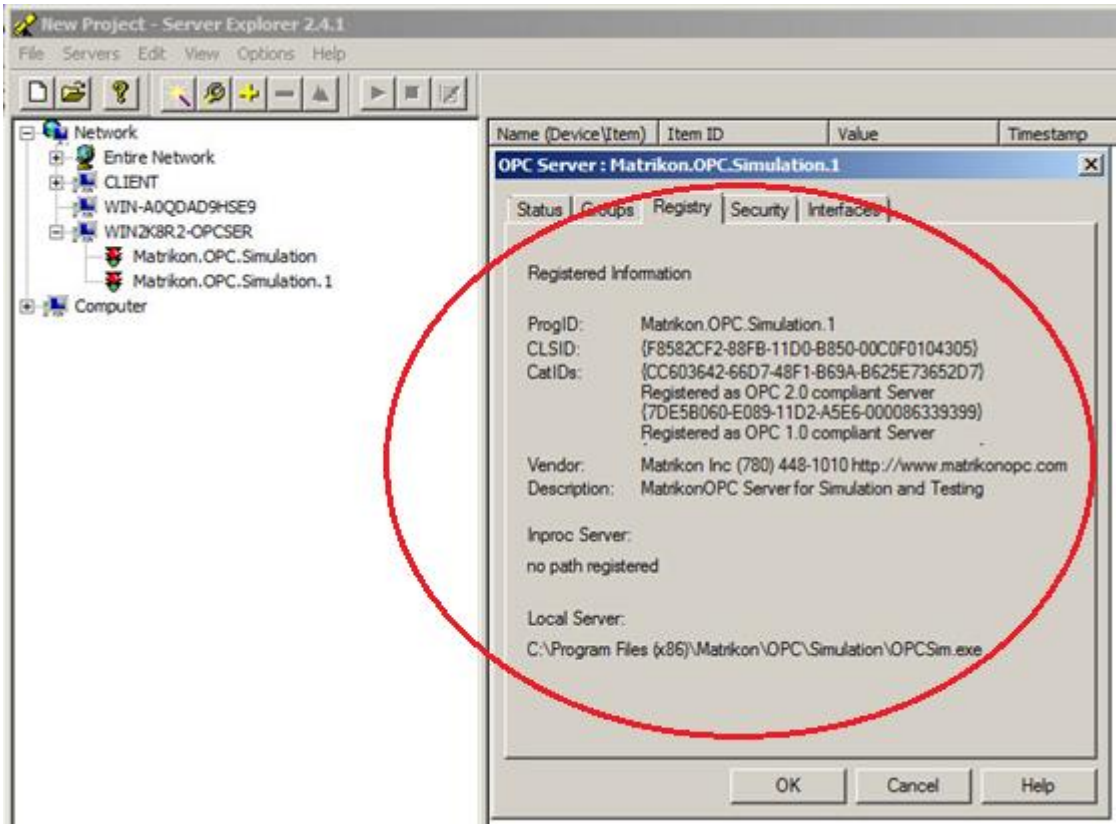
National Instruments Server Explorer

National Instrumentsin Server Explorer -ohjelma paljasti vertailluista client-ohjelmista eniten tietoa testikoneen OPC-komponenteista. Ohjelma listaa verkosta löydetyt OPC-palvelimet NetBIOS-nimellä, jonka alle eritellään palvelinkoneella ajetut palvelininstanssit niiden yksilöllisen ProgID:n mukaan, tässä tapauksessa nimellä ”Matrikon.OPC.Simulation”. Palvelimien listaus on nähtävissä kuviossa 16.



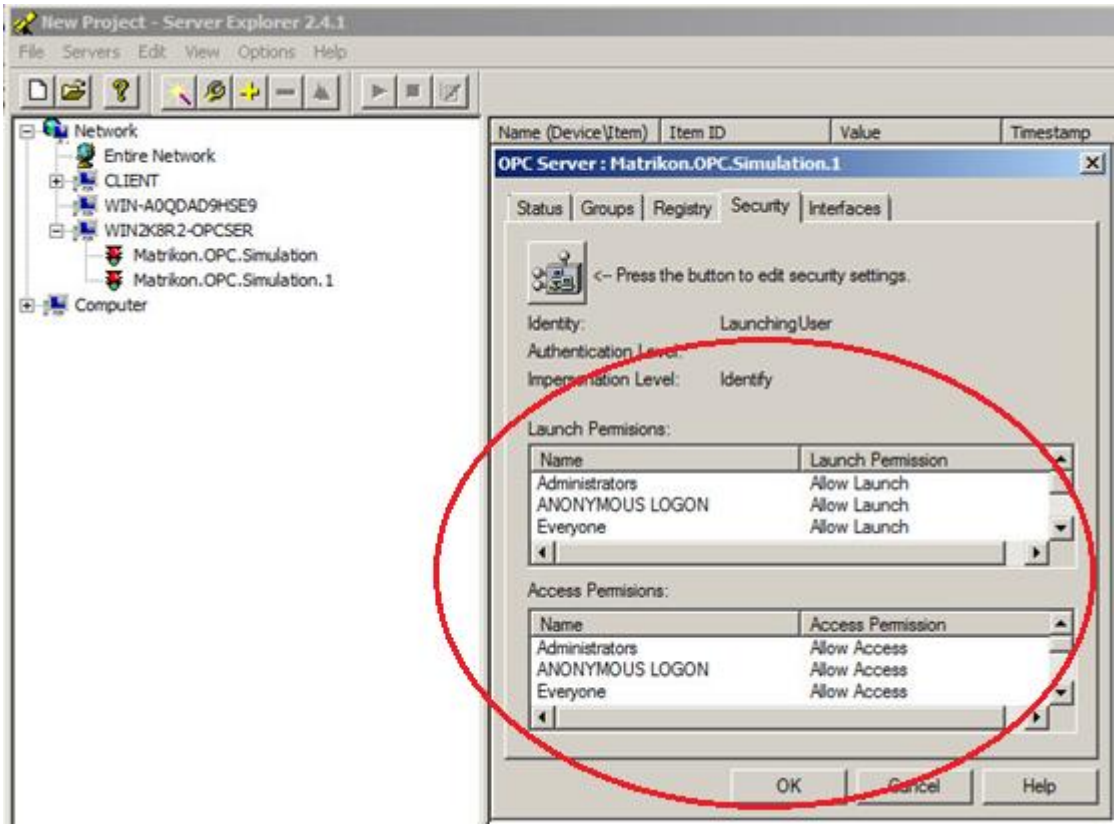
Kuvio 16. OPC-palvelimien listaus ja tilatiedot

Palvelimeen yhdistämättä Server Explorer -ohjelmalla on mahdollista tarkastella palvelimen yksityiskohtia, joihin kuuluvat rekisteritiedot kuten palvelimen nimi ja laitteen kuvaus, CLSID- ja CatID –tunnisteet, laitteen valmistaja sekä palvelinohjelmiston asennuspolku (kuviossa 17)



Kuvio 17. OPC-palvelimen rekisteritiedot NI:n Server Explorerissa

Security -välilehdellä on mahdollista tarkastella palvelimen pääsy- ja hallintaoikeuksia, joista selviää millä käyttäjätileillä on palvelinohjelmiston pääsy- ja käynnistämisoikeudet (kuvio 18).



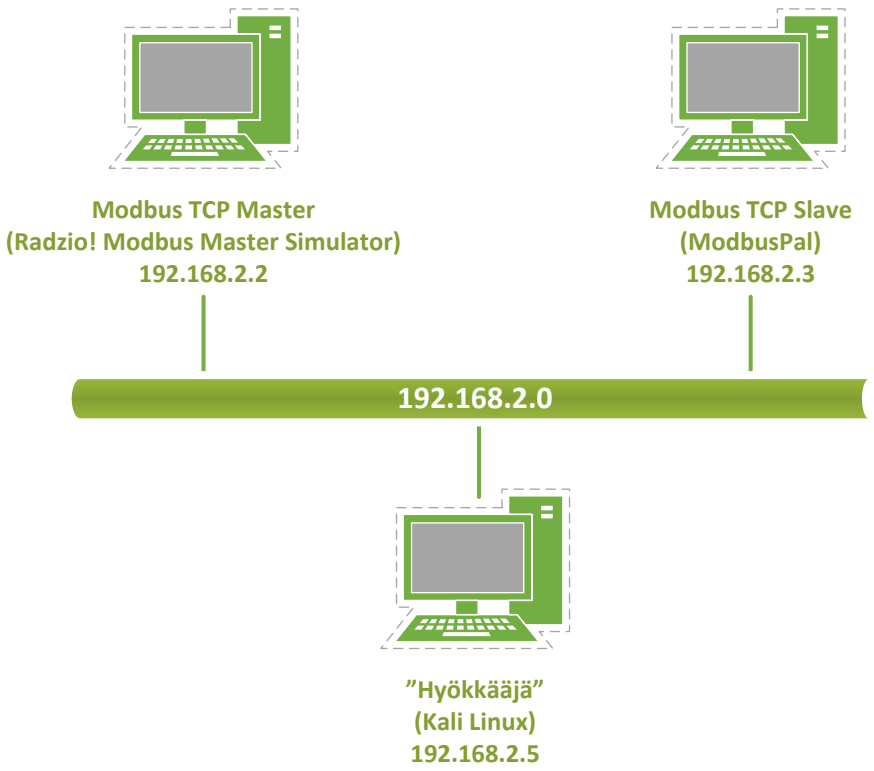
Kuvio 18. OPC-palvelimen turvallisuusasetukset

10.2.3 Johtopäätökset

Useimmat Client-puolen OPC-sovellukset eivät tämän testin perusteella sovellu hyvin OPC-kohdeverkon tiedustelutarkoituksiin, sillä ne useimmiten vaativat yhdistämisen OPC-palvelimeen ennen kuin saavat tästä mitään hyödyllistä informaatiota irti. Tästä huolimatta, jo pelkkä CLSID voi olla hyökkääjälle johtolanka, jolla tämä pääsee OPC-palvelimen jäljille. Ainakin MatrikonOPC-palvelinsimulaattorin tapauksessa CLSID-avaimen verkkohaulla saa selville valmistajan ja ohjelman nimen, vaikka palvelimen NetBIOS- ja ProgID-nimet olisikin muutettu verkkoympäristössä palvelimen toiminnallisuuden salaamiseksi. Kuten luvussa 10.2.2 jo todettiin, NI:n Server Explorer sen sijaan paljasti OPC-palvelimesta reilusti enemmän tietoa ilman tarvetta ottaa tähän yhteyttä. Näistä vertailuista saatujen tietojen pohjalta voidaan todeta tämän olevan paras työkalu OPC-verkon resurssien tiedusteluun.

10.3 Modbus TCP: Komennonsyöttöhyökkäys

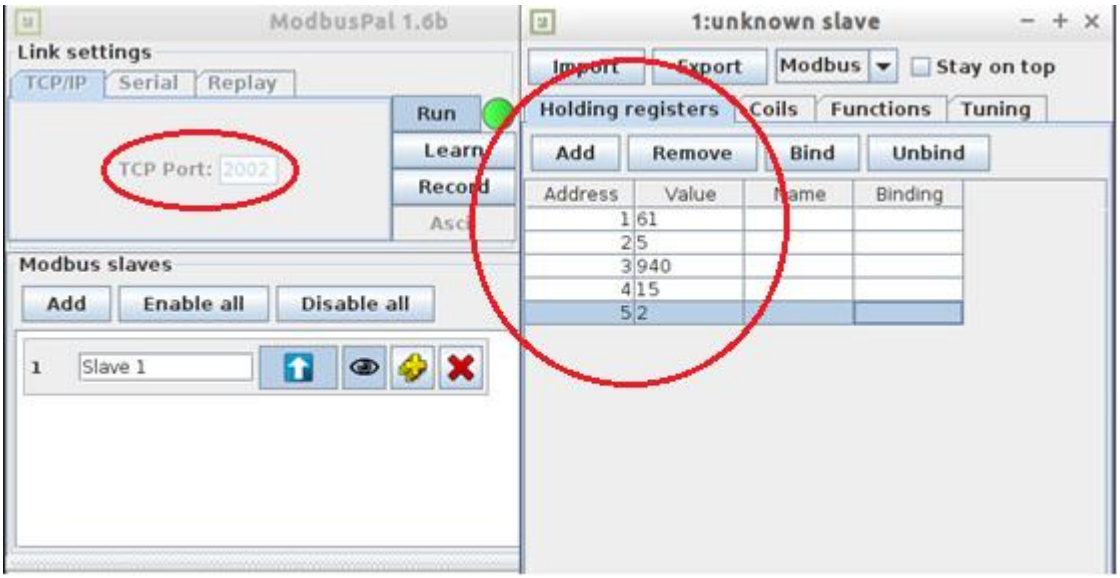
Modbus TCP-protokollan kohdalla testataan lukuun 3.2 viitaten hyökkäysvaiheista viimeistä eli suoraa hyökkäystä ja kohteeseen vaikuttamista. Samassa aliverkossa SCADA-koneiden kanssa on hyökkäysalustana toimiva Kali Linux –käyttöjärjestelmää käyttävä virtuaalikone, jolta käsin vaikutetaan Modbus TCP-protokollaa käyttävien Master- ja Slave -laitteiden väliseen viestintään hyödyntäen Metasploit Frameworkin modbusclient –haavoittuvuusmoduulia. Modbus TCP-testausympäristö on esitetty kuviossa 19.



Kuvio 19. Modbus TCP –haavoittuvuustestausympäristö

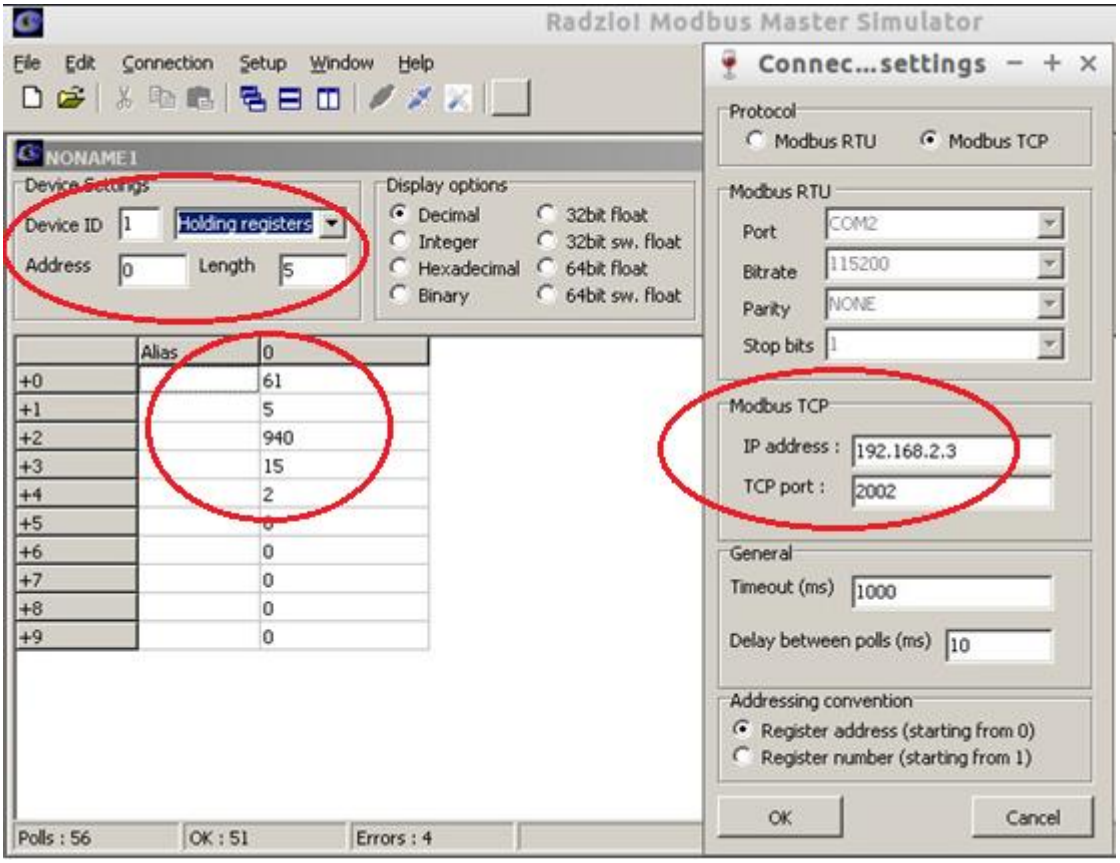
Modbus TCP:n kohdalla protokollasimulaattoreiksi valikoituivat Radzio! Modbus Master Simulator (Master) ja ModbusPal (Slave). ModbusPalilla voidaan simuloida haluttu määrä Slave-laitteita, joille kullekin voidaan erikseen määrittää haluttu määrä rekisteri- ja coil-arvoja vastaamaan oikeissa teollisuus- ja tuotantoympäristöissä käytettäviä määreitä kuten jännite, tilavuus, päällä/pois, lämpötila ja niin edelleen.

Ensimmäisenä Slavena toimivan ModbusPaliin määritetään TCP-portti, jota käytetään informaation siirtämiseksi Masterille. Samaa porttinumeroa käytetään luodessa yhteys Masterilta Slavelle päin. SCADA-verkon PLC-laitteita simuloidaan lisäämällä haluttu määrä Slave-laitteita, joista kullekin voidaan määrittää Slave-editorissa (oik.) omat register- ja coil-arvot. Simulointi alkaa painamalla ”Run”-painiketta. Painikkeen viereinen vihreämerkkivalo ilmoittaa Master-laitteelta vastaanotetuista Request-viesteistä ja näin ollen myös yhteyden toimivuudesta. ModbusPalin toimintaa ja ominaisuuksia on havainnollistettu kuviossa 20.



Kuvio 20. Näkymä Modbus Slave -simulaattorista (ModbusPal)

Kuvion 21 Radzio! Master-simulaattoriin määritetään yhteysparametrit ”Connection Settings”-ikkunasta lisäämällä Slave-laitteen IP-osoite 192.168.2.3 sekä TCP-portti 2002, jota Slave-simulaattori käyttää TCP-porttina. Yhteys Slave-laitteelle muodostetaan ”Connect”-painikkeesta. Pudotusvalikosta valitaan tarkasteltavaksi haluttu muuttuja (Coil/input status, holding/input registers) ja ”Length”-kohtaan Slave-simulaattoriin lisättyjen muuttujien määrä. Näiden parametrimääritysten jälkeen Master-laite alkaa vastaanottamaan Slave-laitteen rekisteriarvoja sisältäviä Response-viestejä. Rekisteriarvojen siirtyminen Slavelta Masterille käy ilmi kuvioista 20 ja 21. Kaikkia Slave-laitteen coil- ja rekisteriarvoja voidaan muuttaa myös Masterilta käsin.

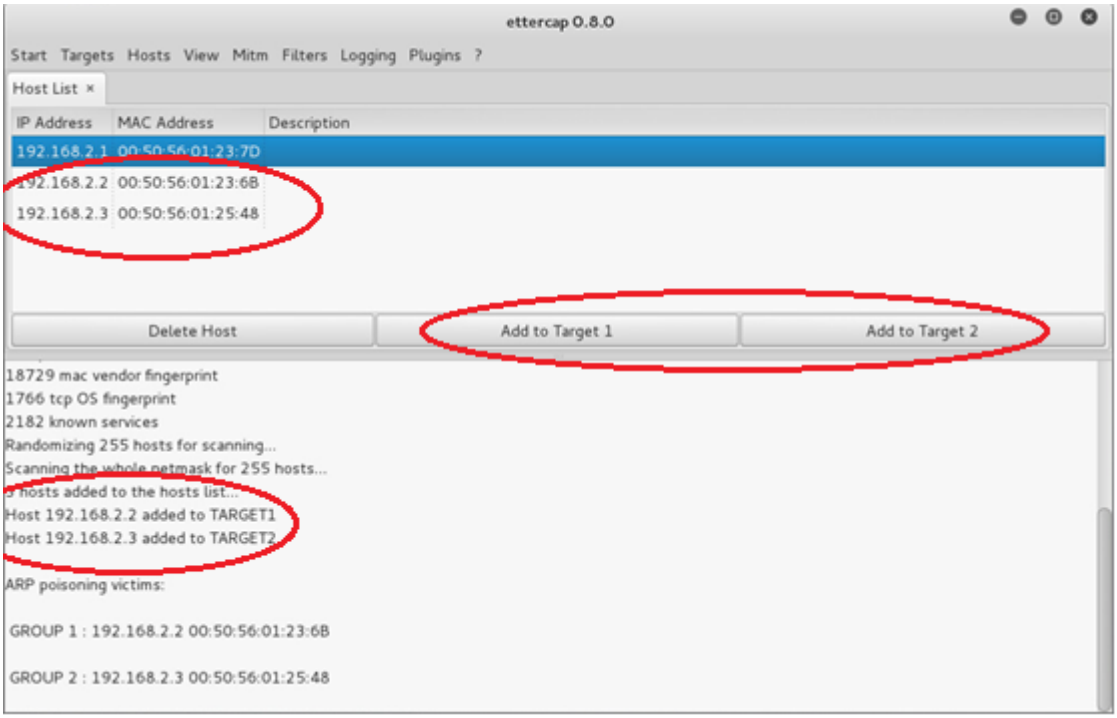


Kuvio 21. Näkymä Modbus Master -simulaattorista (Radzio!)

10.3.1 ARP-reititystietojen väärentäminen (Ettercap)

Jotta Modbus-laitteiden välistä viestintää voidaan salakuunnella, niiden ARP-tietoja täytyy väärentää, jotta niiden välinen liikenne saadaan kiertämään hyökkääjäkoneen kautta MITM-hyökkäyksen periaatteen mukaisesti. Tähän tarkoitukseen käytetään Ettercap-ohjelmaa ja sen ARP poisoning –moduulia.

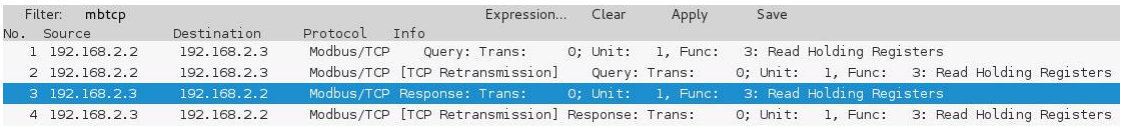
Ensin Ettercapilla skannataan kaikki luodun testiympäristön host-laitteet, jossa myös Modbus-laitteet sijaitsevat. Tämän Modbus Master- ja Slave-laitteet merkitään kohteiksi hyökkäystä varten. Lopuksi aktivoidaan ARP poisoning –hyökkäys MITM-valikosta ja aloitaan yhteyden kuuntelu painamalla ”Start sniffing”. Tämä yhdistää hyökkääjäkoneen MAC-osoitteen Modbus-laitteiden IP-osoitteisiin sekä Master- että Slave-laitteiden ARP-tiedoissa ja saa täten kaiken liikenteen kiertämään hyökkääjän koneen kautta. ARP-tietojen väärentämisprosessia Ettercapilla on havainnollistettu kuviossa 22.



Kuvio 22. Modbus-laitteiden ARP-tietojen väärentäminen Ettercapilla

10.3.2 Liikenteen salakuuntelu (Wireshark)

Modbus TCP-liikenteen kiertäessä hyökkääjäkoneen kautta sen sisältöä voidaan lukea selkokielisenä Wireshark-pakettianalysaattorill. Tämä tapahtuu Modbus-verkon suuntaan osoittavan rajapinnan liikennettä kuuntelemalla ja suodattamalla sitä Modbus TCP-suotimella nimeltä ”mbtcp”. Kuviossa 23 nähdään yksi sykli Master-laitteen lähettämästä rekisterikyselystä sekä Slave-laitteen vastauksena palauttamasta Response-viestistä.



Filter: mbtcp Expression... Clear Apply Save

No.	Source	Destination	Protocol	Info
1	192.168.2.2	192.168.2.3	Modbus/TCP	Query: Trans: 0; Unit: 1, Func: 3: Read Holding Registers
2	192.168.2.2	192.168.2.3	Modbus/TCP	[TCP Retransmission] Query: Trans: 0; Unit: 1, Func: 3: Read Holding Registers
3	192.168.2.3	192.168.2.2	Modbus/TCP	Response: Trans: 0; Unit: 1, Func: 3: Read Holding Registers
4	192.168.2.3	192.168.2.2	Modbus/TCP	[TCP Retransmission] Response: Trans: 0; Unit: 1, Func: 3: Read Holding Registers

Kuvio 23. Wireshark-kaappaus Modbus-laitteiden viesteistä

Tarkastellessa tarkemmin kuvion 10 Slave-laitteen Response-viestiä, nähdään sen sisältävän selkokielisenä kaikki Masterille ilmoitettavat rekisteriarvot. Samalla

nähdään, että arvot vastaavat Slave-simulaattoriin määritettyjä rekisteriarvoja (kuvio 24).

```
▸ Transmission Control Protocol, Src Port: 2002 (2002), Dst Port: 36764 (36764)
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 13
    Unit Identifier: 1
▾ Modbus
    Function Code: Read Holding Registers (3)
    Byte Count: 10
    Register 0 (UINT16): 61
    Register 1 (UINT16): 5
    Register 2 (UINT16): 940
    Register 3 (UINT16): 15
    Register 4 (UINT16): 2
```

Kuvio 24. Wireshark-tarkastelu Modbus Slave -laitteen Response-viestistä

10.3.3 Komennon syöttö (Metasploit Framework+ modbusclient)

Metasploit-penetraatiotestaustyökaluun kuuluvalla modbusclient-moduulilla voidaan suorittaa tavallisesti Master-laitteelle kuuluvia toimintoja kuten Slave-laitteen coil- ja rekisteriarvojen lukemista ja niiden kirjoittamista. Toimintamoodi asetetaan komennolla ”set ACTION READ/WRITE_REGISTER/COIL”. Moduuliin määritetään kohdelaitteen IP-osoite, kirjoitettava data, dataosoite, johon uusi data kirjoitetaan sekä portti, jota kohdelaite käyttää Modbus-datan siirtoon. Nämä määritykset ovat nähtävissä kuviossa 25.

```
msf auxiliary(modbusclient) > show options

Module options (auxiliary/scanner/scada/modbusclient):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   DATA          666              no        Data to write (WRITE_COIL and WRITE_REGISTER modes only)
   DATA_ADDRESS  0                yes       Modbus data address
   RHOST         192.168.2.3      yes       The target address
   RPORT         2002             yes       The target port
   UNIT_NUMBER   1                no        Modbus unit number


Auxiliary action:

   Name            Description
   ----            -----------
   WRITE_REGISTER  Write one word to a register


msf auxiliary(modbusclient) > set ACTION WRITE_REGISTER
ACTION => WRITE_REGISTER
msf auxiliary(modbusclient) >
```

Kuvio 25. Modbus Slave-laitteen määritykset rekisteriarvon muuttamiseksi

Kun oikeat parametrit on määritetty, voidaan ajaa Metasploitin modbusclient-moduuli komennolla ”run”. Kuten kuviosta 26 käy ilmi, moduuli tämän jälkeen palauttaa ilmoituksen onnistuneesta rekisteriarvon 666 kirjoittamisesta Slave-laitteen rekisteriosoitteeseen 0 kuvion 26 mukaisesti.

```
msf auxiliary(modbusclient) > run

[*] Sending WRITE REGISTER...
[+] Value 666 successfully written at registry address 0
[*] Auxiliary module execution completed
msf auxiliary(modbusclient) >
```

Kuvio 26. Slave-laitteen rekisteriarvon muutos Metasploitin modbusclient-moduulilla

Wiresharkissa tarkasteltuna Modbus TCP-paketin sisältö on kuvion 27 mukainen. Paketin yksityiskohdista käy ilmi olennaisimpana toiminnon nimi ”Write Single Register”, protokollan mukainen funktiokoodi yhden rekisteriarvon kirjoittamiselle (06) sekä Unit Identifier, jota käytetään yksilöimään Modbus-verkon Slave-laitteet. Rekisteriin kirjoitettu arvo 666 on kuitenkin itsessään heksadesimaalimuodossa 0x29A.

```
▸ Frame 710: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
▸ Ethernet II, Src: Vmware_01:23:b7 (00:50:56:01:23:b7), Dst: Vmware_01:25:48 (00:50:56:01:25:48)
▸ Internet Protocol Version 4, Src: 192.168.2.5  (192.168.2.5 ), Dst: 192.168.2.3 (192.168.2.3)
▸ Transmission Control Protocol, Src Port: 57835 (57835), Dst Port: 2002 (2002), Seq: 1, Ack: 1, Len: 12
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
▾ Modbus
    Function Code: Write Single Register (6)
    Reference Number: 2
    Data: 029a
```

Kuvio 27. Rekisteriarvon muutos Wiresharkissa tarkasteltuna

Rekisteriarvon muuttuminen voidaan varmentaa myös määrittämällä modbusclient lukemaan rekisteriarvo samasta rekisteriosoitteesta 0, johon aiemmin tehty rekisteriarvon kirjoittaminen kohdistettiin. Annetaan komento ”set ACTION READ_REGISTER” ja ajetaan toiminto rekisteriarvon lukemiseksi komennolla ”run”. Vastauksena moduuli palauttaa kuvion 28 mukaisesti arvon 666 rekisteriosoitteessa 0. Read-toiminto antaa täten keinon varmistaa tarkoitettu rekisteriarvon muutos, jos muita keinoja rekisteriarvojen lukemiseen ei ole käytettävissä.

```
[*] Sending WRITE REGISTER...
[+] Value 666 successfully written at registry address 0
[*] Auxiliary module execution completed
msf auxiliary(modbusclient) > set ACTION READ_REGISTER
ACTION => READ_REGISTER
msf auxiliary(modbusclient) > run

[*] Sending READ REGISTER...
[+] Register value at address 0 : 666
[*] Auxiliary module execution completed
msf auxiliary(modbusclient) >
```

Kuvio 28. Rekisteriarvojen lukeminen Metasploit Frameworkin modbusclient-moduulilla

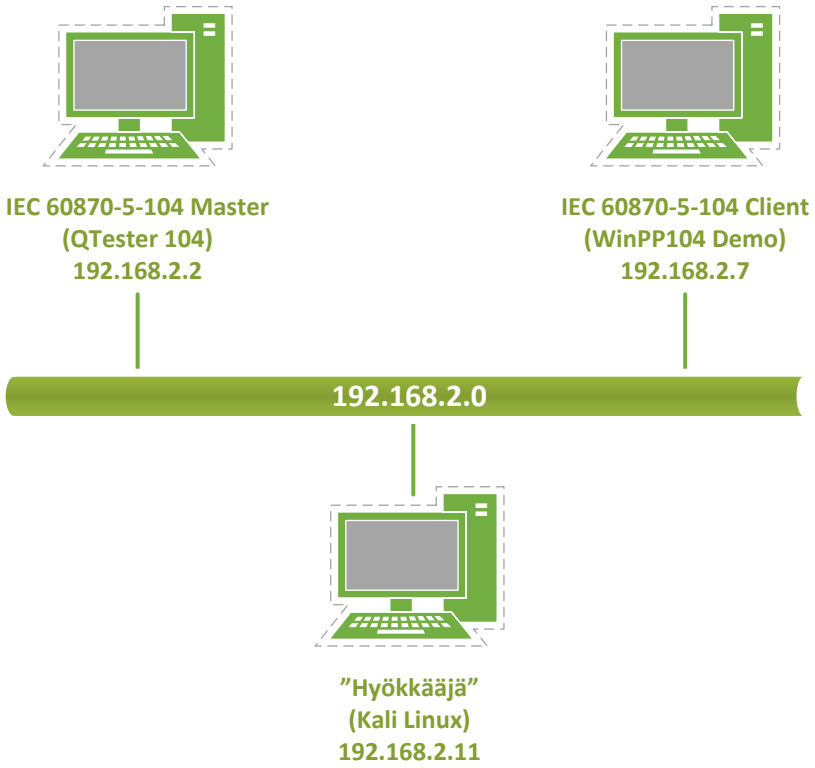
10.3.4 Johtopäätökset

Master-laitteen rooli Modbus TCP-järjestelmässä on toimia ohjaus- ja valvontakeskuksena. Näin ollen rekisteriarvojen kirjoittaminen suoraan RTU- tai Slave-laitteen rekisteriin tarkoittaa käytännössä sitä, että järjestelmää valvovalle Master-laitteelle välittyy Slave-laitteelta virheellistä tietoa aina, kun arvo tietyssä rekisterissä muuttuu. Tämä vääristynyt tilannekuva voi pahimmillaan johtaa järjestelmän ylläpitäjän kohdalla vääriin toimenpiteisiin tai hämätä tätä olemaan ottamatta toimenpiteitä, kun siihen olisi tarvetta.

Tämän menetelmän käyttämisessä piilee se ongelma (tai hyvä puoli), että jos Modbus TCP-järjestelmän Slave-laitteet raportoivat tila- tai arvomuutoksista autonomisesti odottamatta Masterin säännöllistä kyselyä, laitteiden välistä suoritetun komennonsyöttöhyökkäyksen vaikutus ei välttämättä kestä pitkään. Tämä vaikutus on riippuvainen Modbus TCP –kohdejärjestelmästä ja aikavälistä, jona RTU-laiteet päivittävät mittausarvonsa ja välittävät tiedot Masterille.

10.4 IEC 60870-5-104: Modifiointihyökkäys

Tämän haavoittuvuustestauksen tavoitteena on salakuunnella Master- ja Slave-laitteiden välistä liikennettä sekä Ettercap-pluginia käyttämällä muokata Slave-laitteen lähettämiä Single-Point Information –viestityyppejä ”suoraan lennosta” niiden matkatessa Masterille. Tämän ansiosta Masterille välittyy jatkuvasti vääristynyt tieto Slave-laitteen tiedoista ja näin ollen myös protokollaa käyttävän SCADA-järjestelmän tilasta. IEC 104 –protokollan testauksissa testausympäristö on samankaltainen kuin Modbus TCP-protokollan kohdalla lukuunottamatta virtuaalikoneilla ajettavia protokollasimulaattoreita. Modifiointihyökkäystä on havainnollistettu kuviossa 29.

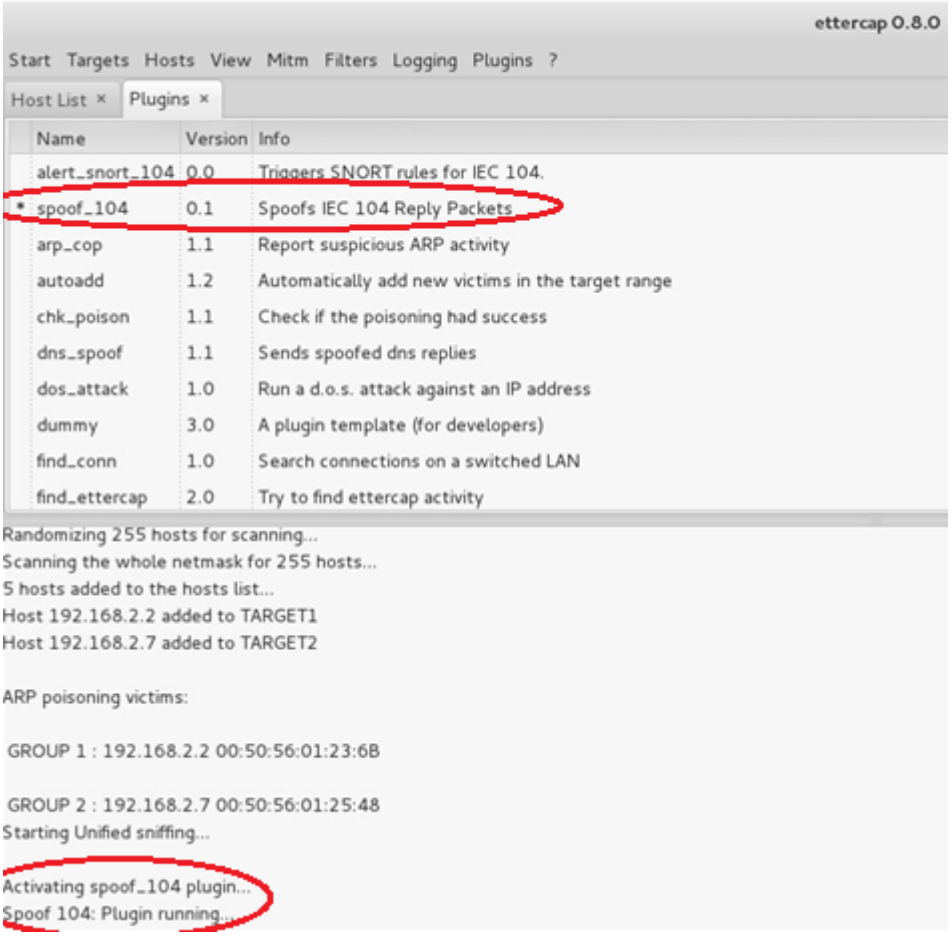


Kuvio 29. IEC 60870-5-104 –haavoittuvuustestausympäristö

10.4.1 Salakuuntelu ja pluginin aktivoiminen (Ettercap)

IEC 104-laitteiden viestintää kuunnellaan kappaleen 10.3.1 mukaisesti hyödyntämällä Ettercapin ARP spoofing –toiminnallisuutta, jonka ansiosta kaikki laitteiden välinen liikenne kulkee hyökkääjän koneen kautta. IEC 104-pluginin löytyessä Ettercapin tiedostokansiosta /ettercap/plug-ins/ se voidaan aktivoida Ettercapin plugin-listasta. Kun plugin on aktiivinen, Ettercap palauttaa viestin ”Spoof 104: Plugin running...” kuvion 30 mukaisesti.

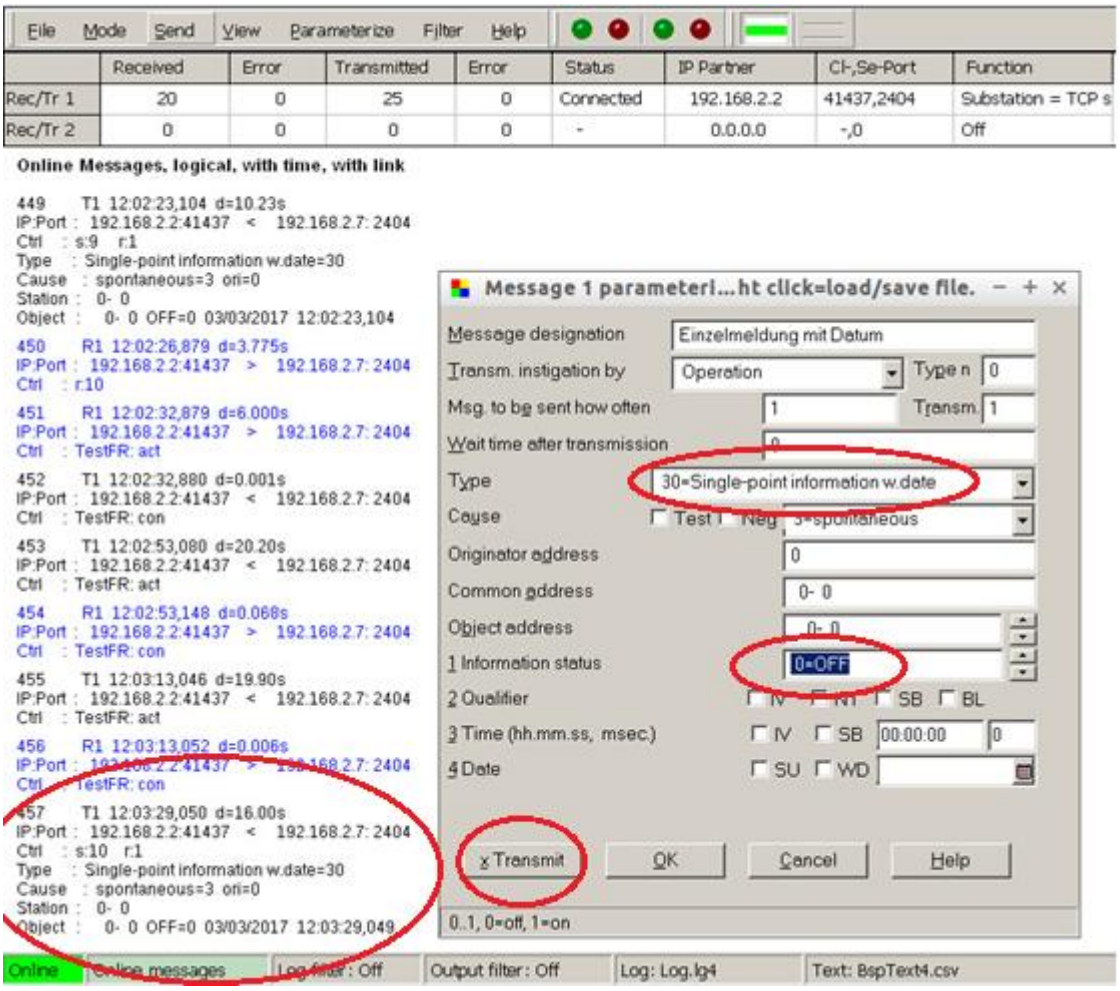
Plugin on suunniteltu etsimään protokollassa määriteltyjä M_SP_TB_1 –viestityyppejä, jollaisen löytäessään se pudottaa alkuperäisen paketin ja lähettää Masterille modifioidun paketin alkuperäisen sijasta. Paketin modifiointi kohdistuu paketin Single-Point Information –tietueen ensimmäiseen SPI-bittiin, jota käytetään ilmaisemaan tilaa (1 = ON, 0 = OFF). (Maynard, P., McLaughlin, K. & Haberler, B. 2014.)



Kuvio 30. IEC 104 -pluginin aktivoiminen Ettercapissa

10.4.2 WinPP104 –protokollasimulaattori (Slave)

Slave-laitteen toiminnallisuutta simuloimaan valittiin kuviossa 31 näkyvä WinPP104 –protokollasimulaattori, sillä se pitää yksityiskohtaista lokia lähetetyistä ja vastaanotetuista paketeista. Sillä voidaan lisäksi manuaalisesti lähettää testauksen tarpeiden mukaisia Single-Point Information –tyypin paketteja Masterille. Kuviossa 31 lähetetään Masterille yksittäinen M_SP_TB_1 –viesti, jonka SPI-arvoksi on valittu 0=OFF.

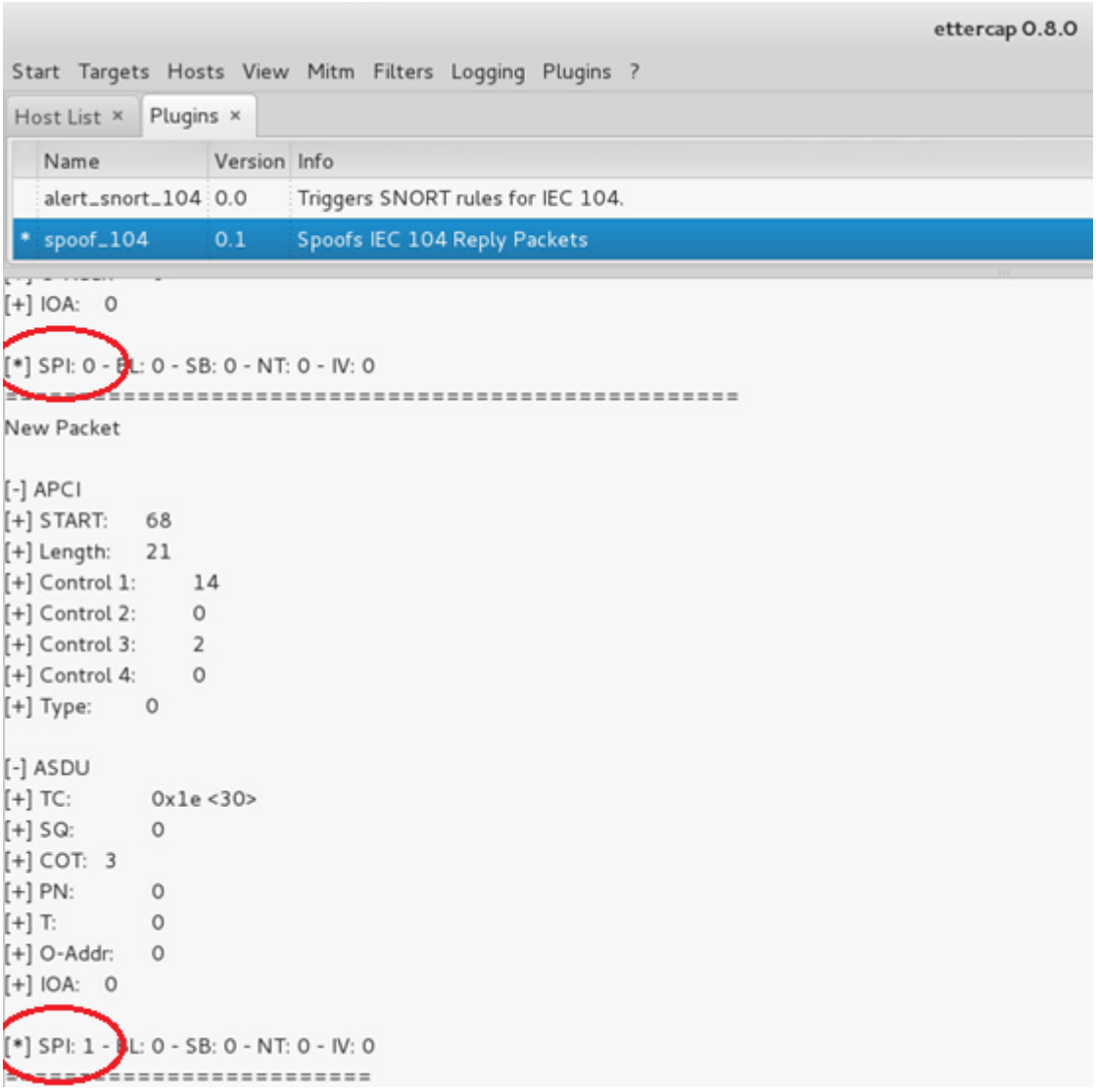


Kuvio 31. Single-point -viestin lähettäminen Slave-laitteelta Masterille

10.4.3 104-plugin toiminnassa

Ettercapin 104-plugin toimii reaaliaikaisesti siten, että aina kun Slave lähettää Masterille M_SP_TB_1 –viestityypillä jollain tila-arvolla, Ettercap havaitsee, pudottaa ja korvaa sen uudella paketilla, jossa SPI-arvo on pluginin asetustiedostossa määritetyn arvon mukainen, joko 0 tai 1. Tässä kokeessa muutetaan SPI-arvo 0:sta 1:een eli OFF-tilasta ON-tilaan.

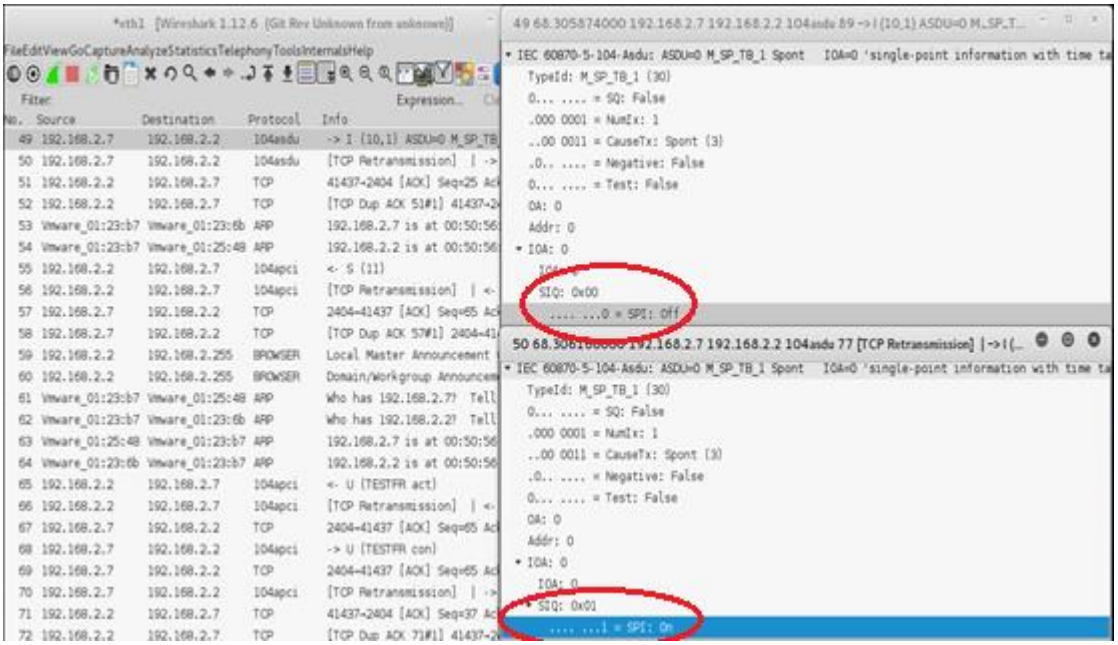
Kuviossa 32 nähdään, kuinka plugin toimieessaan palauttaa sekä alkuperäisen ja korvaavan paketin ASDU:n tiedot yksityiskohtaisesti, muun muassa paketin tyypin 30 (Single point information with time tag CP56Time2a), CoT-arvon 3 (Cause of Transmission, jossa 3 tarkoittaa spontaania, Slaven aloittamaa lähetystä). Alimpana listauksessa näkyy SPI-objekti, joka on pluginin toimesta vaihdettu 0:sta 1:een.



Kuvio 32. SPI-arvon muutos arvosta 0 (OFF) arvoon 1 (ON)

10.4.4 ASDU-paketin uudelleenlähetys (Wireshark)

Tarkastellaan luvussa 10.4.3 kuvattuja pakettimuutoksia myös Wiresharkissa (kuvio 33). Kuviossa vasemmanpuoleisen listauksen ylimmät paketit 49 ja 50 ovat WinPP104:n (Slaven) lähettämä alkuperäinen ASDU SPI-arvolla 0 sekä Ettercap-pluginin uudelleenlähettämä, modifioitu ASDU SPI-arvolla 1, vastaavasti.



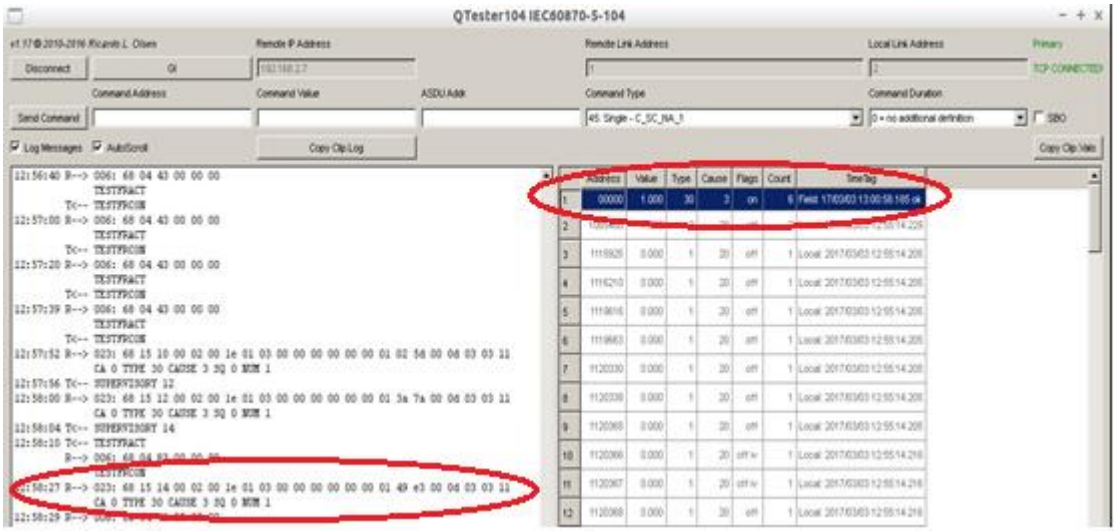
Kuvio 33. SPI-arvon sisältämän ASDU-paketin korvaaminen

10.4.5 QTester104 –protokollasimulaattori (Master)

IEC 104 Master-laitteen toiminnallisuuden simuloimiseen käytetään kuviossa 34 näkyvää QTester104 –protokollasimulaattoria. Sen avulla voidaan lähettää halutun tyyppisiä komentoja Slave-laitteelle Command Type –alasvetovalikosta. Tätä ominaisuutta ei kuitenkaan ollut tarpeen käyttää tässä testauksessa, sillä Slave-simulaattorilla oli mahdollista lähettää M_SP_TB_1 –tyypin viestejä suoraan Masterille ilman tämän kyselyä.

WinPP104- Slave-simulaattorin tavoin se listaa yksityiskohtaista tietoa kaikista lähetetyistä ja vastaanotetuista paketeista sen käyttöliittymän vasemmanpuoleiseen lokiin. Oikeanpuoleissa taulukossa näkyvät Slave-laitteelta peräisin olevat objektitiedot. Kunkintyyppisistä objekteista nähdään suoraan niiden arvo, objektityyppi, lähetyksen syy, flag sekä vastaanotettujen objektiarvojen lukumäärä ja aikaleima.

Listassa tummennettuna ovat tyyppi-ID:n 30 M_SP_TB_1 –viestit, jotka saapuvat aina Masterille flagilla ”on” SPI-bitin muutoksesta johtuen. Tästä syystä Masterilla on jatkuvasti vääristynyt kuva Slave-laitteen tilanteesta. Kuviossa 34 lähetetään Masterille yksittäinen M_SP_TB_1 –viesti, jonka SPI-arvoksi on valittu 0=OFF.



Kuvio 34. Masterin vastaanottamia väärennettyjä M_SP_TB_1 –viestejä

10.4.6 Johtopäätökset

Modifiointihyökkäys on periaatteeltaan samanlainen kuin Modbus TCP:n kohdalla toteutetussa komennonsyöttöhyökkäyksessä, sillä molemmissa tavoitteena on hämätä Master-laitetta, joka valvoo ja ohjaa järjestelmän toimintaa. Modifiointihyökkäyksessä IEC 104 –Masterille saatiin kuitenkin välittymään pysyvästi väärä tilannekuva Slave-laitteen ON/OFF -tyyppisestä boolean-arvoista, koska Masterin vastaanottama ASDU-viesti SPI-arvoineen ei tule Slave-laitteelta, vaan hyökkääjän koneelta, josta käsin hyökkääjä kykenee täysin vaikuttamaan Master-laitteelle välittyvään tilannekuvaan. Huomattakoon, että Ettercap-pluginin koodia muokkaamalla on mahdollista vaihtaa vaikutus päinvastaiseksi tai jopa laajentaa pluginin toiminnallisuutta muihin IEC 104 –protokollan tukemiin tietotyyppeihin.

11 POHDINTA

Kaikki työlle asetetut tavoitteet eri haavoittuvuuksien hyödyntämiseksi saavutettiin. Sain tutkittua ja testattua eri hyökkäyssektoreihin liittyviä haavoittuvuuksia ja näin tekemällä onnistuin lähestymään tutkimusongelmaa eri näkökulmista sen sijaan, että olisin keskittynyt jokaisen protokollan kohdalla saman toistamiseen.

Suurimmaksi haasteeksi työssä muodostui IEC 104 –protokollaan vaikuttaminen. Ensimmäisenä yritin toteuttaa Replay-hyökkäystä, jossa Slave-laitteen Masterille lähettämiä ASDU-viestejä kaapataan ja toistetaan Masterille uudelleen tämän tilannekuvan sekoittamiseksi. Tämä yritys kuitenkin kaatui siihen, että TCPreplay ei osannut sekvensoida Master-laitteelle lähetettäviä paketteja oikein, jolloin ne pudotettiin vastaanottopäässä sovelluskerroksella.

Sekvensoinnin epäonnistuessa tcpreplay-ohjelmalla kokeilin tcpliveplay-ohjelmaa, joka puolestaan sekvensoi paketit oikein, mutta tuki vain TCP-pakettien lähettämistä, eikä huomioinut ASDU-paketteja ollenkaan. Tämän jälkeen kuitenkin löysin keinon Master-laitteen hämäämiseksi Ettercap-pluginia hyödyntämällä, joka lopulta olikin tehokkaampi ja samalla helpompi tapa saman tavoitteen saavuttamiseksi. Replay-hyökkäyksestä poiketen se ei vaadi pakettien manuaalista lähettämistä hyökkäyksen ylläpitämiseksi, vaan plugin muokkaa viestien sisältöä suoraan lennosta.

Myös aiheen teoreettinen tutkiminen oli todella aikaavievää, sillä vaikka työssä käsitellyillä protokollilla oli toiminnallisuuden ja tarkoituksen puolesta paljon yhteistä, oli niiden sisäisissä spesifikaatioissa ja datatyypeissa suuria eroja.

Vaikka työssä käsitellyt protokollat ovat teknisesti vanhentuneita ja eivät sisällä tietoturvaominaisuuksia, ovat ne silti edelleen laajalti käytössä tuotanto –ja teollisuusympäristöissä. Vaikka tieto-ominaisuuksien puolesta uudempia ja tehokkaampia SCADA-protokollia on nykyisin olemassa, niiden käyttö on vielä teollisuudenverkkojen mittakaavalla vähäistä yhteensopivuusongelmien ja teknologioiden päivittämisestä aiheutuvista kustannuksista johtuen (Maynard et al 2014, 31.) Tästä syystä näitä haavoittuvuuksia vastaan olisi suuri tarve kehittää protokollan ulkopuolisia tietoturvamekanismeja, onhan kyseessä kuitenkin kansallista infrastruktuuria pyörittävät järjestelmät

LÄHTEET

Acromag Incorporated, 2005. Introduction to Modbus TCP/IP – Technical Reference. Viitattu 30.1.2017.

https://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf

Advosol, N.d. OPC Unified Architecture. Tietoa eri OPC UA-spesifikaatioista. Viitattu 4.11.2016. http://www.opcua.us/

Audit: TCP Modbus – Unauthorized Write Request. Symantecin hyökkäystunniste Modbus TCP –haavoittuvuudesta. Viitattu 16.11.2016.

https://www.symantec.com/security_response/attacksignatures/detail.jsp?asid=23907

Byres Research, 2007. OPC Security White Paper #2 OPC Exposed. Tietoa OPC-protokollan haavoittuvuuksista. Viitattu 6.11.2016.

https://scadahacker.com/library/Documents/OPC_Security/OPC%20Security%20-%20OPC%20Exposed.pdf

Cagalaban, G., So, Y. & Kim, S. 2009. SCADA Network Insecurity: Securing Critical Infrastructures through SCADA Security Exploitation. Viitattu 27.2.2017.

http://www.sersc.org/journals/JSE/vol6_no6_2009/6.pdf

Cheah, Z. 2008. Testing and Exploring Vulnerabilities of the Applications Implementing IEC 60870-5-104 Protocol. Diplomityö, Tukholman kuninkaallinen teknillinen korkeakoulu. Viitattu 20.2.2017.

http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=0513EED48102FDAD1BD940260EF12B11?doi=10.1.1.548.7490&rep=rep1&type=pdf

Clarke, G. & Reynders, D. 2004. Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems. Tietoa IEC 60870-5 –protokollaperheen spesifikaatioista. Viitattu 5.10.2016.
https://www.fer.unizg.hr/_download/repository/Practical_modern_SCADA_protocols_-_dnp3,_60870-5_and_Related_Systems.pdf

Enigma Software, N.d. What is a CLSID Registry Key? Tietoa OPC-protokollan käyttämistä CLSID-rekisteriavaimista. Viitattu 6.1.2017.

http://www.enigmasoftware.com/what-is-clsid-registry-key/

F-Secure Labs, 2014. Havex Hunts For ICS/SCADA Systems. Raportti Havex-hyökkäyksestä. Viitattu 6.11.2016.

https://www.f-secure.com/weblog/archives/00002718.html

Gandhi, R., Sharma, A., Mahoney, W., Sousan, W., Zhu, Q. & Laplante, P. 2011. Dimensions of Cyber-Attacks. Julkaisu kyberhyökkäysten hyödyistä ja motiiveista. Viitattu 5.3.2017.

http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5725605

IEC 60870-5-104: Telegram structure, N.d. Tietoa IEC 104 –protokollan viestirakenteista. Viitattu 14.12.2016.

http://www.mayor.de/lian98/doc.en/html/u_iec104_struct.htm

Kepware Technologies, 2009. DCOM – Secure by Default. Tietoa OPC:n DCOM-kommunikaation tietoturvasta. Viitattu 6.2.2017.

https://www.kepware.com/en-us/support/resource-library/connectivity-guides/dcom-secure-by-default/

Maynard, P., McLaughlin, K. & Haberler, B. 2014. Towards Understanding Man-In-The-Middle Attacks on IEC 60870-5-104 SCADA Networks. Tutkimus IEC 60870-5-104:n MITM-hyökkäyksistä. Viitattu 28.2.2017.

http://ewic.bcs.org/upload/pdf/ewic_icscsr14_paper5.pdf

McAfee, 2011. Global Energy Cyberattacks: ”Night Dragon”. Raportti Night Dragon –hyökkäyksestä. Viitattu 2.12.2016.

http://www.mcafee.com/us/resources/white-papers/wp-global-energy-cyberattacks-night-dragon.pdf

Metasploit, N.d. Blogi Metasploitista ja sen historiasta. Viitattu 6.3.2017.

https://blgtechn.blogspot.fi/2012/08/metasploit.html

Modbus Messaging on TCP/IP Implementation Guide, 2002. Modbus TCP-protokollan käyttöönotto-opas. Viitattu 14.12.2016.
https://www.honeywellprocess.com/library/support/Public/Documents/51-52-25-121.pdf

Nigam, R. 2015. (Known) SCADA Attacks Over The Years. Blogikirjoitus tunnetuista SCADA-järjestelmiä vastaan kohdistuneista kyberhyökkäyksistä. Viitattu 10.10.2016.

https://blog.fortinet.com/2015/02/12/known-scada-attacks-over-the-years

Northcutt, S., Shenk, J., Shackleford, D., Rosenberg, T., Siles, R. & Mancini, S. 2006. Penetration Testing: Assessing Your Overall Security Before Attackers Do. Tutkimus penetraatiotestauksesta. Viitattu 5.3.2017.

https://www.sans.org/reading-room/whitepapers/analyst/penetration-testing-assessing-security-attackers-34635

OPC Foundation, N.d. OPC Classic. Tietoa eri OPC Classic –spesifikaatioista. Viitattu 22.11.2016.

OPC Foundation, N.d. Unified Architecture. Tietoa OPC UA-protokollasta. Viitattu 4.11.2016.

https://opcfoundation.org/about/opc-technologies/opc-ua/

https://opcfoundation.org/about/opc-technologies/opc-classic/

Oulun Yliopisto, N.d. Glossary of Vulnerability Testing Terminology. Haavoittuvuustestaussanastoa. Viitattu 5.3.2017.

https://www.ee.oulu.fi/roles/ouspg/Glossary

Peterson, D. 2008. OPC UA: Part 3 – Specification Vulnerabilities. Blogikirjoitus OPC UA-protokollan haavoittuvuuksista. Viitattu 19.1.2017.
http://www.digitalbond.com/blog/2008/09/19/opc-ua-part-3-specification-vulnerabilities/

Rhoades, T. 2016. Cyber Security: Understanding the 5 Phases of Intrusion. Graylogin blogikirjoitus kyberhyökkäyksen vaiheista. Viitattu 5.3.2017.

https://www.graylog.org/blog/61-cyber-security-understanding-the-5-phases-of-intrusion

Scamwatch, N.d. Phishing. Tietoa phishing-hyökkäyksistä. Viitattu 5.3.2017.

https://www.scamwatch.gov.au/types-of-scams/attempts-to-gain-your-personal-information/phishing

Schneier, B. 2010. The Story Behind The Stuxnet Virus. Artikkeli Stuxnet -viruksesta. VIitattu 2.12.2016.

http://www.forbes.com/2010/10/06/iran-nuclear-computer-technology-security-stuxnet-worm.html

Software Testing Fundamentals, N.d. Tietoa järjestelmätestausmenetelmistä. Viitattu 5.3.2017.

http://softwaretestingfundamentals.com/

Standard IEC 60870-5-104 Data Types. Taulukko protokollan dataobjekteista. Viitattu 8.1.2017.

https://infosys.beckhoff.com/english.php?content=../content/1033/tcplclibiec870_5_104/html/tcplclibiec870_5_104_objref_overview.htm&id

Stouffer, K., Falco, J. & Kent, K. 2006. Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control Systems Security. Tietoa SCADA-järjestelmistä. Viitattu 3.11.2016.

https://www.dhs.gov/sites/default/files/publications/csd-nist-guidetosupervisoryanddataccquisition-scadaandindustrialcontrolsystemssecurity-2007.pdf

US-Cert, 2013. Understanding Denial-of-Service Attacks. Artikkeli DoS-hyökkäyksistä. Viitattu 5.3.2017.

https://www.us-cert.gov/ncas/tips/ST04-015

Veracode, N.d. SQL Injection Cheat Sheet & Tutorial: Vulnerabilities & How to Prevent SQL Injection Attacks. Tietoa SQL-hyökkäyksistä. Viitattu 5.3.2017.

https://www.veracode.com/security/sql-injection

VMware, N.d. VMware vCloud Suite. PDF-artikkeli vCloudista. Viitattu 6.3.2017.

https://www.virtualizationworks.com/datasheets/VMware-vCloud-Suite-Datasheet.pdf

What is OPC? Tietoa OPC-protokollasta. Viitattu 5.10.2016.

http://www.opcdatahub.com/WhatIsOPC.html

LIITTEET

Liite 1. IEC 60870-5-104 -protokollan datatyypit

Type	Dec	Hex	Description
ASDU_TYPEUNDEF	0	0x00	Not used
M_SP_NA_1	1	0x01	Single-point information
M_SP_TA_1	2	0x02	Single-point information w/ time tag
M_DP_NA_1	3	0x03	Double-point information
M_DP_TA_1	4	0x04	Double-point information with time tag
M_ST_NA_1	5	0x05	Step position information
M_ST_TA_1	6	0x06	Step position information with time tag
M_BO_NA_1	7	0x07	Bitstring of 32 bit
M_BO_TA_1	8	0x08	Bitstring of 32 bit with time tag
M_ME_NA_1	9	0x09	Measured value, normalised value
M_ME_TA_1	10	0x0A	Measured value, normalized value with time tag
M_ME_NB_1	11	0x0B	Measured value, scaled value
M_ME_TB_1	12	0x0C	Measured value, scaled value wit time tag
M_ME_NC_1	13	0x0D	Measured value, short floating point number
M_ME_TC_1	14	0x0E	Measured value, short floating point number with time tag

M_IT_NA_1	15	0x0F	Integrated totals
M_IT_TA_1	16	0x10	Integrated totals with time tag
M_EP_TA_1	17	0x11	Event of protection equipment with time tag
M_EP_TB_1	18	0x12	Packed start events of protection equipment with time tag
M_EP_TC_1	19	0x13	Packed output circuit information of protection equipment with time tag
M_PS_NA_1	20	0x14	Packed single point information with status change detection
M_ME_ND_1	21	0x15	Measured value, normalized value without quality descriptor
ASDU_TYPE_22..29	22..29	0x16..0x1D	Reserved (standard area)
M_SP_TB_1	30	0x1E	Single-point information with time tag CP56Time2a
M_DP_TB_1	31	0x1F	Double-point information with time tag CP56Time2a
M_ST_TB_1	32	0x20	Step position information with time tag CP56Time2a
M_BO_TB_1	33	0x21	Bitstring of 32 bit with time tag CP56Time2a
M_ME_TD_1	34	0x22	Measured value, normalised value with time tag CP56Time2a
M_ME_TE_1	35	0x23	Measured value, scaled value with time tag CP56Time2a
M_ME_TF_1	36	0x24	Measured value, short floating point number with time tag CP56Time2a

M_IT_TB_1	37	0x25	Integrated totals with time tag CP56Time2a
M_EP_TD_1	38	0x26	Event of protection equipment with time tag CP56Time2a
M_EP_TE_1	39	0x27	Packed start events of protection equipment with time tag CP56Time2a
M_EP_TF_1	40	0x28	Packed output circuit information of protection equipment with time tag CP56Time2a
ASDU_TYPE_41..44	41..44	0x29..0x2C	Reserved (standard area)
C_SC_NA_1	45	0x2D	Single command
C_DC_NA_1	46	0x2E	Double command
C_RC_NA_1	47	0x2F	Regulating step command
C_SE_NA_1	48	0x30	Set-point Command, normalised value
C_SE_NB_1	49	0x31	Set-point Command, scaled value
C_SE_NC_1	50	0x32	Set-point Command, short floating point number
C_BO_NA_1	51	0x33	Bitstring 32 bit command
ASDU_TYPE_52..57	52..57	0x34..0x39	Reserved (standard area)
C_SC_TA_1	58	0x3A	Single command with time tag CP56Time2a
C_DC_TA_1	59	0x3B	Double command with time tag CP56Time2a
C_RC_TA_1	60	0x3C	Regulating step command with time tag CP56Time2a
C_SE_TA_1	61	0x3D	Measured value, normalised value command with time

			tag CP56Time2a
C_SE_TB_1	62	0x3E	Measured value, scaled value command with time tag CP56Time2a
C_SE_TC_1	63	0x3F	Measured value, short floating point number command with time tag CP56Time2a
C_BO_TA_1	64	0x40	Bitstring of 32 bit command with time tag CP56Time2a
ASDU_TYPE_65..69	65..69	0x41..0x45	Reserved (standard area)
M_EI_NA_1	70	0x46	End of Initialisation
ASDU_TYPE_71..99	71..99	0x47..0x63	Reserved (standard area)
C_IC_NA_1	100	0x64	Interrogation command
C_CI_NA_1	101	0x65	Counter interrogation command
C_RD_NA_1	102	0x66	Read command
C_CS_NA_1	103	0x67	Clock synchronisation command
C_TS_NA_1	104	0x68	Test command
C_RP_NA_1	105	0x69	Reset process command
C_CD_NA_1	106	0x6A	Delay acquisition command
C_TS_TA_1	107	0x6B	Test command with time tag CP56Time2a
ASDU_TYPE_108..109	108..109	0x6C..0x6D	Reserved (standard area)

P_ME_NA_1	110	0x6E	Parameter of measured values, normalized value
P_ME_NB_1	111	0x6F	Parameter of measured values, scaled value
P_ME_NC_1	112	0x70	Parameter of measured values, short floating point number
P_AC_NA_1	113	0x71	Parameter activation
ASDU_TYPE_114..119	114..119	0x72..0x77	Reserved (standard area)
F_FR_NA_1	120	0x78	File ready
F_SR_NA_1	121	0x79	Section ready
F_SC_NA_1	122	0x7A	Call directory, select file, call file, call section
F_LS_NA_1	123	0x7B	Last section, last segment
F_FA_NA_1	124	0x7C	ACK file, ACK section
F_SG_NA_1	125	0x7D	Segment
F_DR_TA_1	126	0x7E	Directory
ASDU_TYPE_127..255	127..255	0x7F..0xFF	Reserved (user area)