HAMK
Hämeen ammatti-
korkeakoulu

Reititinverkon Varmennepohjainen Turvallisuus

Varmenteet verkkolaitteiden autentikointivälineenä

Ammattikorkeakoulututkinto, opinnäytetyö
Tieto- ja viestintätekniikka, Insinööri (AMK)
Kevät 2024
Risto Lievonen

HAMK
Hämeen ammatti-
korkeakoulu

Tieto- ja viestintätekniikan insinööri (AMK) Tiivistelmä
Tekijä Risto Lievonen Vuosi 2024
Työn nimi Reititinverkon varmennepohjainen turvallisuus
Ohjaaja Teemu Järvenpää

Opinnäytetyön tarkoituksena oli rakentaa laboratorioympäristöön varmennehierarkian tarvitsema palvelininfra sekä testiverkko, jossa verkkolaitteiden varmenteita pystytään testaamaan. Työn taustana ovat PKI-varmenteet, IEEE 802.1X-autentikointi ja IPsec-tietoliikennesalaus.

Työn tavoitteena oli selvittää ja dokumentoida tarvittavat toimenpiteet ja konfiguraatiot verkkolaitteiden varmenteiden käyttöönottoon, elinkaaren hallintaan ja varmenteiden hyödyntämiseen laitteiden autentikointivälineenä. Käyttökohteiksi valikoitui laitetunnistuksen toteutus 802.1X-protokollalla ja liikenteen salaus IPsec-protokollapinolla.

Opinnäytetyössä käytiin läpi x509-varmenteiden rakennetta, varmennehierarkiaa sekä julkisen avaimen salausta. Lisäksi perehdyttiin laitetunnistukseen ja tietoliikenteen salaukseen verkkolaitteilla. Laboratorioympäristön varmenneratkaisusta kirjoitettiin käyttöönotto-ohje, jota voidaan soveltaa laitevarmenteiden käyttöönottoon tuotantoympäristössä. Käyttöönotto-ohjeessa esitellään tarvittavat komponentit, konfiguraatiot ja prosessit varmenteiden elinkaaren aikaisiin toimintoihin, laitetunnistuksen käyttöön ja tietoliikenteen salaukseen.

Työn tuloksena syntyi toimiva ja testattu kokonaisuus laboratorioympäristöön, jolla pystytään todentamaan määritellyt käyttötapaukset ja jota voidaan jatkojalostaa muihin PKI- ja 802.1X testitapauksiin. Kirjoitettu dokumentaatio kattaa seikkaperäisesti ympäristön olennaiset komponentit ja käytettävät toiminnot.

Opinnäytetyö tehtiin nimettömäksi jäävän toimeksiantajan tilauksesta laboratorioympäristöön.

Avainsanat Varmenteet, tietoliikennelaitteet, todentaminen
Sivut 32 sivua ja liitteitä 29 sivua

HAMK
Hämeen ammatti-
korkeakoulu

Information and Communication Technology
Author Risto Lievonen
Subject Certificate-based Security on Router Network
Supervisors Teemu Järvenpää

Abstract
Year 2024

The overall aim of this thesis was to build and configure the necessary server infrastructure and simulated network in a laboratory environment, where certificates for network devices can be tested. The background for the thesis is PKI certificates, IEEE 802.1X authentication and IPsec traffic encryption.

More specifically, the aim of the thesis was to discover and document the needed actions and configurations for the introduction, lifecycle management and utilization of x509 certificates as an authentication tool for network devices. The use cases selected were device identification and authentication with 802.1X protocol and traffic encryption with IPsec protocol stack.

The theory part of the thesis consists of the structure of x509 certificates, certificate hierarchy and public key encryption. It also includes device authentication and traffic encryption with network devices. The practical part of the thesis contains server infrastructure built in laboratory environment, which provides the necessary services for certificate management. An implementation guide for implementing the solution for production environments was developed from the server infrastructure. The guide presents the needed components, configurations and processes for all functions related to certificate lifecycle management, device authentication and traffic encryption.

As a result of the thesis, a functioning and tested set of services for laboratory environment was produced enabling the validation of the defined use cases. The environment can be refined for future PKI and 802.1X testing. The written guide thoroughly covers the essential components and functions of the environment.

The thesis was done for a client’s lab environment. The client wishes to stay anonymous.

Keywords Certificates, network devices, authentication
Pages 32 pages and appendices 29 pages

Sisällys

Kuvat, taulukot ja kaavat

Kuvat

Kaavat

Liitteet

Lyhenteet

AAA	Authentication, Authorization, Accounting protokolla
AD	Active Directory, windowsin hakemistopalvelu
ASN.1	Abstract Syntax Notation One, mm. varmenteiden tietorakennestandardi
BER	Basic Encoding Rules
CA	Certification Authority, varmenteiden myöntäjä
CCITT	Comité Consultatif International Téléphonique et Télégraphique, kansainvälinen televiestintäliitto, nykyisin ITU
CER	Canonical Encoding Rules
CHAP	Challenge Handshake Authentication Protocol
CRL	Certificate Revocation List, varmenteiden sulkulista
CRLDP	CRL Distribution Point, sulkulistan jakelupiste
DER	Distinguished Enconding Rules, varmenteiden binäärimuotoinen koodaus
DNS	Domain Name Service, nimipalvelu
DTLS	Datagram Transport Layer Security
EAP	Extensible Authentication Protocol, autentikointiprotokolla
EAPOL	EAP Over LAN
ESP	Encapsulation Security Payload, IPsecin salattu liikennöintiprotokolla
FTP	File Transfer Protocol, tiedonsiirtoprotokolla
HTTP	Hypertext Transfer Protocol. internetin tiedonsiirtoprotokolla
IEEE	Institute of Electrical and Electronics Engineers

IETF	Internet Engineering Task Force, Internet-protokollien standardisoinnista vastaava organisaatio
IKE	Internet Key Exchange, v1 ja v2, avainneuvotteluprotokolla
IP	Internet Protocol
IPsec	IP Security Architecture, kokoelma IP-liikenteen salaukseen liittyviä protokollia
ITU	International Telecommunication Union, kansainvälinen televiestintäliitto
LAN	Local Area Network, lähiverkko
LDAP	Lightweight Directory Access Protocol, yleinen hakemistoprotokolla
MD5	Message-Digest algorithm, tiivistefunktio
MTU	Maximum Transmission Unit, TCP-liikenteen suurin sallittu pakettikoko
NTP	Network Time Protocol, aikapalveluprotokolla
OCSP	Online Certificate Status Protocol, varmenteen tilatietoprotokolla
OSI	Open Systems Interconnection, yleensä OSI-malli eli OSI reference model, tietoliikenteen kerroksellisuutta kuvaava malli
PAE	Port Access Entity, porttiautentikoinnin toimija
PAP	Password Authentication Protocol, salasanapohjainen autentikointiprotokolla
PEAP	Protected EAP
PEM	Privacy Enhanced Mail, varmenteiden base64-muotoinen koodaus
PKCS	Public Key Cryptographic Standards, kokoelma RSA Laboratoriesin kehittämiä standardeja
PKI	Public Key Infrastructure, julkisen avaimen varmennerakenne
PSK	Pre-Shared Key, jaettu salaisuus autentikointiperusteena
PTP	Precision Time Protocol, tarkempi aikapulssiprotokolla synkronointia vaativiin ympäristöihin.
RA	Register Authority, varmenteiden myöntäjän delegoima toimija
RADIUS	Remote Authentication Dial In User Service, laiteautentikointiprotokolla
RFC	Request For Comments, IETF:n julkaisut, ml. standardit ja muut asiakirjat
RSA	Rivest, Shamir, Adleman. Salausalgoritmi, joka on nimetty kehittäjiensä mukaan
SA	Security Association, IPsec-salauksen komponentti
SCTP	Stream Control Transmit Protocol
SHA	Secure Hash Algorithm, kryptografinen tiivistefunktio
TCP	Transmission Control Protocol, yhteydellinen tiedonsiirtoprotokolla
TLS	Transport Layer Security, internetin salausprotokolla
UDP	User Datagram Protocol, yhteydetön tietoliikenneprotokolla

1 Johdanto

Nykyaikana tietoverkot ovat keskeisessä roolissa erityisesti länsimaisissa yhteiskunnissa. Yhteiskunnan rattaat, ihmisten yksityiselämä ja liike-elämä toimivat aina enemmän verkottuneessa ja datapohjaisessa maailmassa. Tietoverkot ovat useissa tilanteissa näkymättömissä, mutta välttämättömiä taustapalveluita.

Suurimmalla osalla yrityksistä, valtionhallinnosta ja kolmannesta sektorista on jonkinlainen tietoverkko. Yksinkertaisimmillaan verkko on yrityksen toimipaikan lähiverkko, joka kytkeytyy teleoperaattorin reitittimen kautta internetiin. Valtionhallinnolla ja suuryrityksillä, esimerkiksi suurilla tehdaskomplekseilla on jo monimutkaisia, varmennettuja reititinverkkoja.

Tietoverkkojen ja dataan pohjautuvien toimintojen yleistyessä myös verkkojen riskit ovat yleistyneet. Tietomurrot, haittaohjelmat ja verkossa tapahtuva vakoilu ovat lisääntyneet ja ammattimaistuneet. Siinä missä hakkerointia tekivät 2000-luvun alussa taidoillaan leveilevät nörtit huvikseen, nykypäivänä sitä tekevät ammattimaiset rikollisjärjestöt ja valtiolliset toimijat hyötymistarkoituksessa.

Suurin osa verkkohyökkäyksistä tapahtuu verkon ulkopuolelta, ja usein niissä hyödynnetään ulkoverkkoon avoinna olevien palveluiden haavoittuvuuksia. Ulko- ja sisäverkon rajapintojen suojaamiseen laitetaan usein myös paljon huomiota. Usein varsinkin suuremmissa yrityksissä verkot ovatkin hyvin ja asianmukaisesti suojattu palomuurilla ulkoa tulevia uhkia vastaan.

Kuitenkin erityisesti laajojen verkkojen myötä myös verkon sisältä päin tulevat riskit kasvavat. Esimerkiksi laajassa tehdaskompleksissa alueen sisäverkon alueella voi työskennellä useita yrityksiä, alihankkijoita ja vierailevia toimijoita. Mahdollinen hyökkääjä voi soluttautua alueelle sisään ja kytkeytyä lähiverkkoon, jolloin useassa tapauksessa yrityksen palvelut kuten verkkolevyt ovat suoraan avoinna, koska ollaan jo sisäverkossa.

Sisäiseen uhkaan on käytetty jo pitkään lähiverkon laitetunnistusta, jossa verkkoon liittyvä päätelaite, kuten työasema, tunnistetaan AD:hen (Active Directory) liitetyn varmenteen perusteella, ja sen liikennöinti hyväksytään tai hylätään työryhmäkytkimessä. Mutta joissain käyttötapauksissa verkkoon liittyvä laite ei olekaan työasema, vaan toinen reititin.

Opinnäytetyössä käsitellään verkkolaitteiden varmennepohjaisia turvallisuusratkaisuja. Reitittimille voidaan luoda varmenteita aivan kuten työasemillekin, ja niitä voidaan käyttää paitsi laitteen tunnistamisessa, myös IPsec-salauksen neuvottelun autentikoinnissa.

Opinnäytetyö vastaa seuraaviin kysymyksiin:

- Miten verkkolaitteille saadaan luotua organisaatioon kytketyt varmenteet?
- Miten verkkolaitteiden varmenteita voidaan hyödyntää laitetunnistuksessa?
- Miten verkkolaitteiden varmenteita voidaan hyödyntää liikenteen salaamisessa ja mitä etua varmenteiden käyttämisellä on salauksessa?

2 Työn lähtökohdat

Työ tehtiin toimeksiantona nimettömälle toimeksiantajalle, jolla on käytössään useista reitittimistä koostuva verkko, jolta vaaditaan korkeaa tietoturvaa ja luotettavuutta.

Toimeksiantajalla oli käytössä Cisco Systemsin verkkolaitteita, joiden tietoturvaa haluttiin parantaa varmennepohjaisella laitetunnistuksella ja IPsec-salauksella. (IP Security Architecture)

2.1 Toimeksianto

Työksi annettiin reititinverkon varmennepohjaisen laitetunnistuksen ja IPsec-salauksen toteutus. Koska käytettävissä ei ollut valmista varmennehierarkiaa ja sen vaatimaa palvelinrakennetta, sisällytettiin toimeksiantoon myös sen rakentaminen.

Lopputuloksena toimeksiantaja halusi käyttöönotto-ohjeen verkkolaitteiden varmenteiden elinkaaren mukaisiin toimiin sekä varmenteiden hyödyntämiseen laitetunnistuksessa ja IPsec-salauksessa.

2.2 Käyttötapaukset

Työlle määriteltiin toteutettavat tehtävät käyttötapausten kautta. Käyttötapauksilla kuvattiin tarvittavia toimintoja, joita yrityksen tulee verkossa toimiessaan saada tehtyä. Käyttötapaukset jaettiin varmenteiden elinkaareen, laitetunnistukseen ja IPsec-salauksen autentikointiin.

Toimeksiantajan laboratorioverkkoon rakennettiin käyttötapausten vaatima ympäristö ja käyttötapausten toiminnallisuudet toteutettiin ympäristössä. Toteutuksesta laadittiin käyttöönotto-ohje, jonka perusteella ympäristön pystyy rakentamaan tuotantoympäristöön.

2.2.1 Varmenteiden elinkaari

Varmenteiden elinkaari lähtee varmennehierarkiasta, jossa määritetään juurivarmenne, mahdolliset välivarmenteet ja laitevarmenteet sekä näiden keskinäiset suhteet. Laitevarmenteen näkökulmasta varmenteen elinkaari on seuraavan kaltainen:

1. Laite luo avainparin ja varmennepyynnön
2. Varmentaja allekirjoittaa pyynnön ja luo varmenteen
3. Varmenne tuodaan laitteelle ja otetaan käyttöön
4. Varmenteen voimassaolo tarkastetaan sulkulistalta
5. Mahdollisesti vanhentuva varmenne uusitaan
6. Varmenne peruutetaan tai vanhentuu

2.2.2 Varmennepohjainen laitetunnistus

Varmennepohjainen laitetunnistus perustuu 802.1x-standardiin, joka voi käyttää useita eri protokollia laitteen tunnistamiseen ja käyttöoikeuden varmentamiseen. Yleisimmin käytetty protokolla on RADIUS. (Remote Authentication Dial In User Service)

Laitetunnistuksesta tunnistettiin seuraavat käyttötapaukset:

- Sallittu reititin liitetään kytkimeen
- Reititin, jolla ei ole varmennetta, liitetään kytkimeen
- Sulkulistalla oleva reititin liitetään kytkimeen
- Kytkimessä kytkettynä olevan reitittimen varmenne lisätään sulkulistalle

2.2.3 Varmenteilla autentikoitu IPsec

Tietoliikenteen salauksessa yleisin protokollapino on IPsec. Se sisältää kokoelman protokollia, jotka tuottavat luottamuksellisuutta IP-tiedonsiirrolle salaamalla liikenteen. Toisin kuin verkkoselaimissa OSI-kerroksilla 4-7 (Open Systems Interconnection), eli sovelluskerroksilla, käytettävä TLS-salaus (Transport Layer Security), IPsec salaa kaiken IP-liikenteen toimiessaan OSI-kerroksella 3.

IPsec-salaus muodostetaan IKE-avainneuvotteluprotokollaa (Internet Key Exchange) käyttämällä, joka neuvottelee salauksessa käytettävät algoritmit ja salausavaimen.

Neuvottelu suojataan salaamalla neuvottelupaketit joko yhteisesti tiedossa olevalla, etukäteen jaetulla salasanalla (PSK, Pre-Shared Key), tai varmenteiden luottosuhteeseen perustuvalla salauksella.

IPsec-salaukselle määriteltiin seuraavat käyttötapaukset:

- IPsec-tunneli muodostetaan PSK:lla
- IPsec-tunneli muodostetaan hyväksytyillä varmenteilla
- IPsec-tunneli muodostetaan hyväksytyillä, eri välivarmenteiden julkaisemilla varmenteilla
- IPsec-tunnelia yritetään muodostaa sulkulistalla olevalla varmenteella
- Muodostetun IPsec-tunnelin toinen laite lisätään sulkulistalle (CRL, Certificate Revocation List)
- Muodostetun IPsec-tunnelin toinen laite lisätään sulkulistalle (OCSP, Online Certificate Status Protocol)

2.3 Työn tavoitteet

Työn tavoitteena oli rakentaa varmenteiden tuottamiseen tarvittava palvelinrakenne, testata varmenteiden elinkaaren käyttötapaukset verkkolaitteilla ja kirjoittaa toteutuksesta verkkolaitteiden varmenteiden käyttöönotto-ohje, jota toimeksiantaja pystyy soveltamaan tuotantoympäristössä.

Työ toteutettiin valmiina olevaan laboratorioverkkoon, jossa oli palvelimia VMware-virtualisointialustalla ja sekä simuloituja verkkolaitteita. Simuloidut verkkolaitteet oli toteutettu GNS3-verkkosimulaattorilla, ja ne sisälsivät laitevalmistajan täysiä ohjelmistoja, jotka vastaavat toiminnoiltaan täysin fyysisiä laitteita. Simuloidut verkkolaitteet käyttivät Cisco Systemsin ohjelmistoa.

3 Tietoperusta

Opinnäytetyö käsittelee varmenteita ja niiden käyttöä autentikoinnissa kahdessa eri käyttötapauksessa. Tietoperustan ensimmäisessä osassa syvennytään itse varmenteeseen ja käsitellään x.509-standardin mukaiset PKI-varmenteet.

Seuraavissa osissa käsitellään opinnäytetyön käyttötapausten tietoperustat. 802.1X-standardi määrittelee laitetunnistuksen varmenteiden perusteella, ja IPsec-protokolla määrittelee tietoliikenteen salaukseen käytetyt protokollat ja sen autentikoinnin varmenteilla.

3.1 X.509 PKI Varmenne

Internetin laajentuessa 1980-luvulla kansainvälisessä televiestintäliitto CCITT:ssä (Comité Consultatif International Téléphonique et Télégraphique), joka nykyisin tunnetaan nimellä ITU (International Telecommunication Union) pohdittiin internetin laajuisen osoitteiston laatimista, jotta kaikki internetin käyttäjät voisivat löytää toisensa. Tähän käyttötarkoitukseen kehitettiin standardi X.500, jossa määriteltiin internetin hakemistopalvelut. (Weider & Reynolds, 1992, s. 1)

X.500 standardi ei koskaan saavuttanut alkuperäistä tavoitettaan internetin hakemistopalvelusta, mutta sen mukana kehitettiin myös muita standardeja, joista useita on vielä nykypäivänäkin hyvin laajassa käytössä, kuten hakemistoprotokolla LDAP (Lightweight Directory Access Protocol) ja varmennestandardi X.509, PKI (Public Key Infrastructure). X.509 julkaistiin CCITT standardina vuonna 1988, ja päivitettiin useaan otteeseen päätyen vuonna 1996 julkaistuun versio 3 standardiin, joka on nykyisinkin käytössä. (Cooper ym., 2008, ss. 9–10)

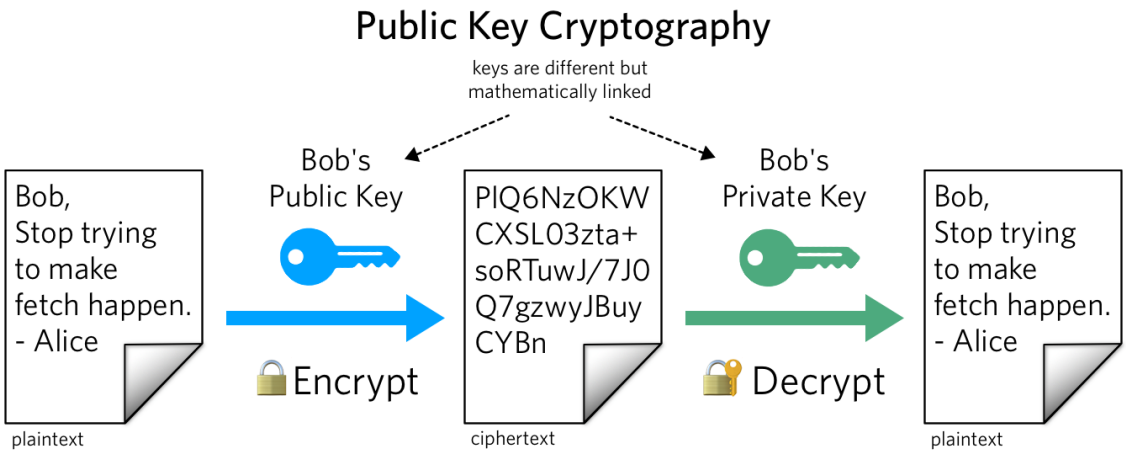
3.1.1 Julkisen avaimen salaus

Ron Rivestin, Adi Shamirin ja Len Adlemanin kehittämä ja mukaan nimetty julkisen avaimen RSA-salaus perustuu matemaattiseen yhtälöön, jossa kaksi suurta, satunnaista alkulukua, p ja q kertomalla luodaan julkinen avain n. Samoista alkuluvuista lasketaan yhteisestä, julkisesta kantaluvusta E, esimerkiksi TLS-salauksessa 65537, käänteinen eksponentti, jonka jälkeen otetaan julkisen avaimen tekijöinä käytetyt alkuluvut, vähennetään molemmista 1, kerrotaan nämä luvut keskenään ja lasketaan aiemman käänteisen eksponentin jakojäännös tälle tulolle. Näin saadaan yksityinen avain D. (Hellman, 2002, s. 45)

Salakirjoitettu viesti C saadaan ottamalla desimaaleiksi muutettu viesti P, nostamalla se potenssiin E ja laskemalla siitä jakojäännös julkiseen avaimeen n. Salauksen purku taas tapahtuu nostamalla salattu viesti C potenssiin yksityinen avain E ja laskemalla saadusta luvusta jakojäännös julkisella avaimella n. (Hellman, 2002, s. 45)

Kuvassa 1 on esitetty viestin salaus julkisella avaimella ja salatun viestin purku yksityisellä avaimella.

Kuva 1. Julkisen avaimen salaus (Robinson, 2018).



Kaava 1. Julkisen avaimen laskentakaava (Hellman, 2002, s. 45).

$$n = p \cdot q$$

Kaava 2. Yksityisen avaimen kaava (Hellman, 2002, s. 45).

$$D = E^{-1}\ mod\ (p-1) \cdot (q-1)$$

Kaava 3. Julkisen avaimen salauksen laskentakaava (Hellman, 2002, s. 45).

$$C = P^E\ mod\ n$$

Kaava 4. Julkisen avaimen salauksen purkamisen laskentakaava (Hellman, 2002, s. 45).

$$P = C^D\ mod\ n.$$

Salauksen vahvuus perustuu siihen, että tuotettu julkinen avain on riittävän suuri, esimerkiksi 500 bittinen, jolloin salausavaimen pituus on $2^{500} = 3{,}2 \cdot 10^{150}$ merkkiä. Tämän luvun tekijöihin, eli alkulukuihin p ja q, jako vaatisi $1{,}8 \cdot 10^{75}$ operaatiota, joihin 1 µs laskenta-ajalla kuluisi $6 \cdot 10^{61}$ vuotta, mutta tuntemalla alkuperäisistä alkuluvuista muodostettu yksityinen avain salauksen purku on helppoa. Nykyiset salausavaimet käyttävät 2048 tai 4096-bittisiä avaimia. (Hellman, 2002, ss. 44–46)

3.1.2 Digitaalinen allekirjoitus

Digitaalinen allekirjoitus perustuu allekirjoitetun viestin tiivistefunktioon, ja sen salaamiseen käyttäjän yksityisellä avaimella. (Hellman, 2002, s. 47) Tiivistefunktio, esimerkiksi SHA-256, (Secure Hash Algorithm) laskee viestin tai tiedoston sisällöstä tiivistetyn, vakiopituisen merkkijonon, joka on aina sama, kun viesti on sama, mutta muuttuu merkittävästi, jos viestin yksikin bitti muuttuu. (Pittalia, 2019, s. 147)

Allekirjoittaessa lasketaan viestin tiiviste ja salataan se käyttäjän yksityisellä avaimella, jolloin vastaanottajan on helppo varmistua viestin aitoudesta vertaamalla itse laskemaansa viestin tiivistefunktiota lähettäjän allekirjoitukseen, jonka vastaanottaja purkaa lähettäjän julkisella avaimella. (Hellman, 2002, s. 47)

3.1.3 Varmenne

X.509 standardi määrittelee PKI-varmenteet. Varmenteen tehtävä on yhdistää julkinen salausavain luotettavasti sen käyttäjään, joka voi olla järjestelmä tai henkilö. Varmenne sisältää tiedon varmenteen allekirjoituksesta, sen algoritmista, varmenteen myöntäjästä, varmenteen voimassaolosta, varmenteen käyttötarkoituksista ja käyttäjän julkisen avaimen ja sen algoritmin. Lisäksi varmenteeseen voidaan liittää laajennuksia. Varmenteen rakenne määritellään ASN.1 (Abstract Syntax Notation One) syntaksilla, joka on määritelty osana alkuperäistä X.500-standardia. Usein varmenne kuitenkin kirjataan PEM-muodossa (Privacy Enhanced Mail). (Cooper ym., 2008, ss. 16–22)

Varmenne itsessään ei pääsääntöisesti sisällä yksityistä avainta, joten se voidaan julkaista sellaisenaan ja välittää julkisia kanavia pitkin. Varmenteen perusteella voidaan tarkastaa varmenteen luotettavuus, salata varmenteen omistajalle tarkoitettu viesti ja varmistua varmenteen omistajan allekirjoituksen aitoudesta. (Cooper ym., 2008, ss. 9–10, 29–31)

3.1.4 Varmennepolku

Varmenteiden luotettavuus perustuu varmennepolkuun, jossa tunnettu ja luotettu juurivarmentaja, CA (Certificate Authority) tai CA:n delegoima RA (Register Authority) myöntää käyttäjän varmenteen. Myöntäminen tapahtuu, kun juurivarmentaja allekirjoittaa alivarmenteen omalla yksityisellä avaimellaan. Luottamalla päävarmenteeseen luotetaan automaattisesti myös sen myöntämiin alivarmenteisiin. Internetissä luotetut juurivarmenteet on lisätty selainohjelmistoihin ohjelmiston valmistajan toimesta, mutta yksityisissä ympäristöissä luottosuhde tulee muodostaa erikseen. (Cooper ym., 2008, ss. 10–13)

Koska julkisen avaimen salauksen luotettavuus perustuu yksityisen avaimen pysymiseen salassa, on ylin juurivarmenne usein turvallisuussyistä erillisellä, verkosta irrotetulla palvelimella. Tämä turvallisuusjärjestely luo kuitenkin haasteen, koska uusien varmenteiden myöntäminen pitää tapahtua juurivarmenteen yksityisellä avaimella, jolloin kaikkien varmenteiden myöntäminen pitäisi tehdä manuaalisesti tällä erillispalvelimella. Tästä syystä varmennepolkuun lisätään yleensä välivarmentaja, joka on juurivarmentajan myöntämä varmenne, joka puolestaan myöntää käyttäjille varmenteita. Tällä järjestelyllä varmistutaan siitä, että juurivarmenne pysyy salaisena, mutta varmenteita pystytään silti tuottamaan verkon kautta. (Cooper ym., 2008, ss. 10–13)

3.1.5 Sulkulistaus

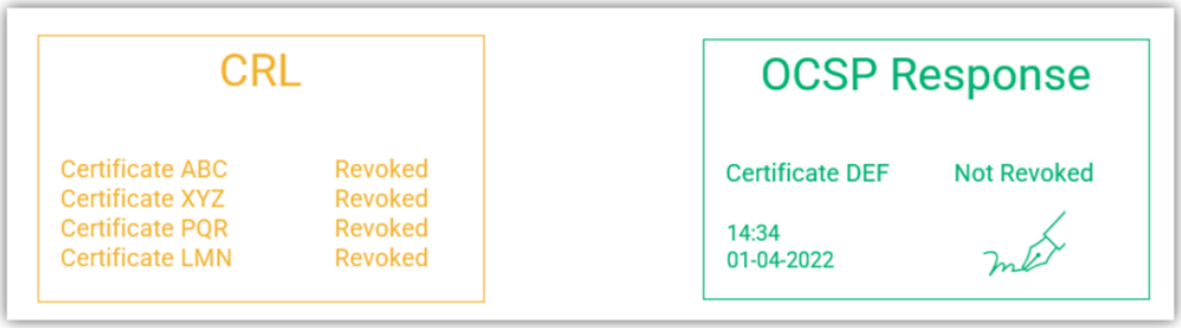
Varmenne on voimassa pääsääntöisesti koko siihen merkityn voimassaoloajan. Joissain tilanteissa voi kuitenkin olla tarvetta poistaa varmenteen luottamus kesken sen voimassaolon. Esimerkiksi käyttäjän irtisanoutuessa työnantajansa palveluksesta tai luotetun laitteen tullessa varastetuksi on tärkeää, että on olemassa mekanismi varmenteen luottamuksen purkuun. Tähän tarkoitukseen X.509-standardissa on määritelty sulkulistaus eli CRL. CRL on aikaleimattu lista, jonka varmenteen CA tai erillinen CRL-myöntäjä allekirjoittaa. Lista sisältää kaikkien peruttujen varmenteiden sarjanumerot. (Cooper ym., 2008, ss. 13–14)

Sulkulistat jaetaan julkisesti, ja sulkulistan osoite on yleensä kerrottu CA:n varmenteessa. Sulkulistan jakamiseen on useita keinoja, mutta useimmiten listat jaetaan julkiselta HTTP- (Hypertext Transport Protocol), FTP- (File Transfer Protocol) tai LDAP-palvelimelta. Myös alkuperäinen X.500-palvelin on mahdollinen, mutta harvinaisempi. Tätä CRL:n jakelupistettä kutsutaan CRL Distribution Pointiksi (CRLDP). (Cooper ym., 2008, ss. 13–14)

CRL-sulkulistaus perustuu varmenteiden käyttäjän ajoittain tekemään tarkastukseen, jossa käyttäjä aika ajoin noutaa sulkulistan CRLDP:stä ja varmistaa sen kautta varmenteen voimassaolon. Tämä jättää kuitenkin mahdollisuuden käyttää peruttua varmennetta vanhan sulkulistan voimassaoloajan, jos käyttäjä ei ole noutanut uudempaa sulkulistaa, jossa varmenne olisi peruttu. Tähän tarpeeseen on kehitetty OCSP, joka antaa mahdollisuuden reaaliaikaisempaan varmenteiden tilatietoon kuin CRL. (Santesson ym., 2013, s. 5)

OCSP perustuu HTTP-kyselyyn, jolla käyttäjä varmistuu yksittäisen varmenteen voimassaolosta joka kerta, kun varmennetta käytetään. Tällä tavoin varmenteen peruminen johtaa luottamuksen menetykseen välittömästi. OCSP ja CRL eivät ole toisiaan poissulkevia, vaan CRL voi edelleen toimia OCSP:n varamenetelmänä, tai toisin päin. (Santesson ym., 2013, ss. 6–8) Kuvassa 2 on esitelty CRL:n ja OCSP:n olennaiset erot.

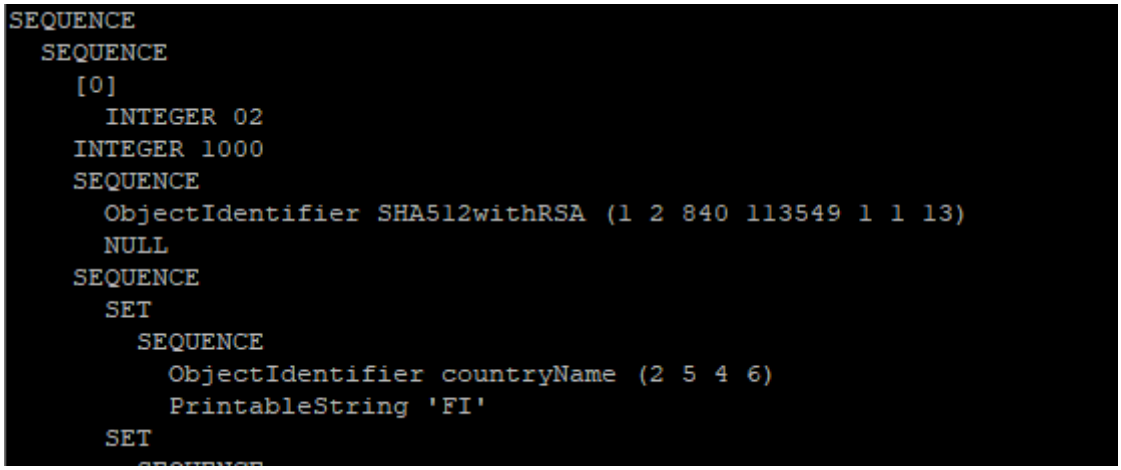
Kuva 2. CRL ja OCSP erot (Thakkar, 2022).



3.1.6 Tiedostomuodot

ITU:n standardi x.690 määrittelee ASN.1-syntaksin ja sen hierarkiat, muuttujatyypit ja rakenteen. Standardissa määritellään myös eri koodaussäännöt ASN.1-syntaksin mukaisen tiedon tallentamiseksi tiedostomuotoon. Nämä tavat ovat DER (Distinguished Encoding Rules), BER (Basic Encoding Rules) ja CER (Canonical Encoding Rules). (ITU-T, 2021) Kaikki x.509-standardin tiedot on kirjoitettu ASN.1-syntaksilla. Standardi määrittelee, että PKI-tiedostot koodataan DER-koodauksella. (Cooper ym., 2008, s. 16) Kuvassa 3 on esimerkki ASN.1-syntaksin mukaisen varmenteen alusta.

Kuva 3. ASN.1 muotoinen varmenne.



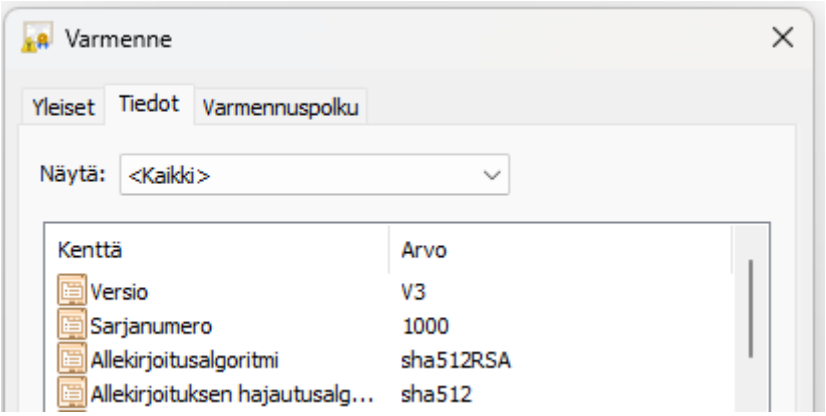
DER-muodossa ASN.1 tiedot kirjoitetaan heksana binääritiedostoon. Tiedosto voidaan ennen tai jälkeen tallennuksen muuntaa PEM-koodaukseksi, joka on käytännössä vain base64-koodaus DER-muotoisesta tiedostosta. PEM-koodauksessa ennen ja jälkeen tietosisältöä lisätään otsikot. Otsikot ovat määrämuotoisia sisältäen viisi miinusmerkkiä (-), välilyönnin, isoilla kirjaimilla sanat BEGIN tai END, isoilla kirjaimilla tiedoston tyypin ja lopuksi taas viisi miinusmerkkiä. Esimerkiksi ”----- BEGIN CERTIFICATE -----” ja ”----- END CERTIFICATE -----”. (Josefsson & Leonard, 2015, ss. 3–7) Kuvassa 4 on kuvassa 3 ollut ASN.1-muotoisen varmenteen alkuosa koodattuna PEM-muotoon.

Kuva 4. PEM-muotoinen varmenne.

Tiedostot tallennetaan levylle eri tiedostomuodoissa, joita käytetään kuitenkin eri työkaluissa eri tavoilla. DER-muotoiset varmenteet on määritelty ”.cer” päätteisiksi RFC 2585:ssä, ja PEM-koodatut ”.crt” päätteisiksi RFC7468:ssa. Eri työkalut kuitenkin käyttävät näitä ristiin, ja lisäksi näkee ”.pem” ja ”.der” päätteisiä. Muita kuin varmenteita näkee esimerkiksi ”.key”, ”.crl”, ”.csr” päätteillä. Useimmin kyse on PEM-koodatusta datasta. Käytettävä pääte riippuukin usein käytettävästä työkalusta, eikä pääte vaikuta varsinaiseen toimintaan.

(Josefsson & Leonard, 2015, s. 9) Esimerkki windowsin tiedostonhallinnassa avatusta varmenteesta löytyy kuvasta 5.

Kuva 5. Windowsin tiedostonhallinnassa avattu varmenne.



Esimerkiksi Windowsin tiedostonhallinta tunnistaa ”.crt”, ”.cer” ja ”.der” tiedostot varmenteiksi ja ”.crl” tiedostot sulkulistaksi. RSA Laboratories on luonut ryhmän standardeja nimeltään PKCS (Public Key Cryptography Standards), jotka määrittelevät osaltaan käytettäviä tiedostomuotoja. Näistä muodoista nykypäivänä yleisin käytössä oleva on PKCS#12, joka on salatun yksityisen avaimen sisältävä varmenne. PKCS#12-tiedostoista käytetään usein päätettä ”.p12” tai ”.pfx”. (Moriarty ym., 2014, s.4)

3.2 IEEE 802.1X

Porttiperusteinen laitetunnistus ja autentikointi antaa verkon hallinnoijalle mahdollisuuden rajoittaa verkkoon kytkettyjä laitteita ja sen myötä parantaa verkon turvallisuutta, kun luvattomien laitteiden kytkeminen verkkoon on estetty. IEEE (the Institute of Electrical and Electronics Engineers) on standardisoinut porttiperusteisen laitetunnistuksen standardilla 802.1X. (IEEE, 2020, s. 2)

Standardissa määritellään EAP (Extensible Authentication Protocol), joka käyttää keskitettyä pääsynhallintapalvelinta. Koska EAP on suunniteltu toimimaan ilman IP-yhteyttä, yhteys verkkolaitteen ja pääsynhallintapalvelimen kautta toteutetaan EAPOL (EAP Over LAN) protokollaa käyttäen. EAPOL nojaa oletukseen, jossa yhteys pääsynhallintapalvelimen ja verkkolaitteen välillä on turvallinen. Usein näin ei kuitenkaan ole, jolloin hyödynnetään AAA-protokollia (Authentication, Authorization, Accounting), kuten Diameter tai RADIUS autentikoimaan laitteita. (IEEE, 2020, s. 17)

3.2.1 Laitetunnistuksen toiminta

Port Access Entity (PAE) tarkoittaa käytännössä verkkolaitteen porttia, jossa laitetunnistus on käytössä. PAE voi olla joko verkossa kiinni olevassa verkkolaitteessa, joka valvoo liittyviä laitteita, tai laitteessa, joka yrittää liittyä verkkoon. Näitä kutsutaan autentikoijaksi (Authenticator) ja anojaksi (Supplicant). Kukin PAE voi toimia molemmissa rooleissa, eri autentikointitapahtumissa. EAP määrittelee vielä kolmannen roolin, eli pääsynhallintapalvelimen. Kaikki kolme roolia ovat olennaiset EAP-autentikointitapahtumassa. (IEEE, 2020, s. 43)

Autentikointitapahtuma EAP:n kautta tapahtuu seuraavasti. Kun PAE:en kytketään uusi laite (anoja), autentikoijana toimiva PAE lähettää laitteelle autentikointipyynnön, jossa kysytään esimerkiksi identiteettiä, MD5-haastetta, etukäteen jaettua salasanaa (PSK) tms. Joissain tapauksissa identiteetiksi riittää portti, johon on kytkeydytty tai anojan MAC-osoite. Anoja vastaa pyyntöön omalla identiteetillään, johon autentikoija vastaa tarvittaessa lisäpyynnöillä. Autentikoija tarkastaa identiteetin pääsynhallintapalvelimelta, kunnes autentikoija joko saa riittävät tiedot yhteyden hyväksymiseen tai hylkää yhteyden ja sulkee portin. (Aboba ym., 2004, ss. 7–8)

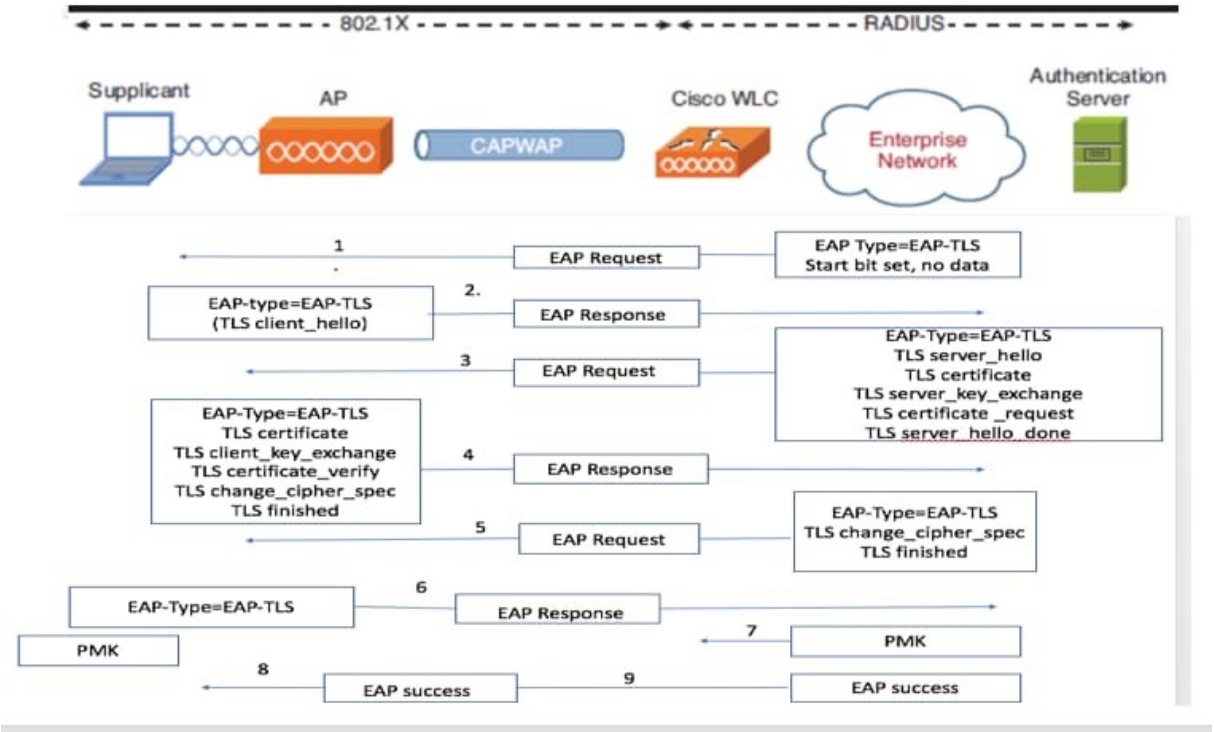
EAP toimii kaksisuuntaisena protokollana, jolloin erillinen, mutta samanaikainen autentikointitapahtuma voi tapahtua myös verkkoon liittyvän laitteen puolella. Tässä

tapahtumassa roolit kääntyvät päinvastoin, ja autentikoijana toimii liittyvä laite, ja anojana kiinteän verkon laite. Luonnollisesti myös liittyvä laite tarvitsee yhteyden pääsynhallintapalvelimelle autentikoinnin onnistumiseksi. EAP itsessään ei tarvitse IP-yhteyttä pääsynhallintapalvelimen kanssa, jolloin autentikointitapahtuma voi tapahtua saman palvelimen kautta kuin ensimmäisessä tapahtumassa. Useimmiten kuitenkin autentikointi tapahtuu AAA-protokollalla kuten RADIUS, jolloin IP-yhteys tarvitaan, tai verkko tulee olla konfiguroitu siten, että kiinteän verkon laite välittää autentikointipyynnöt pääsynhallintapalvelimelle. (Aboba ym., 2004, s. 14)

3.2.2 EAP-TLS

EAP:in on kehitetty useita laajennuksia tukemaan erilaisia autentikointitapauksia. Yksi näistä laajennuksista on EAP-TLS, joka mahdollistaa varmenteiden käytön sekä anojan, että autentikoijan identiteetin tunnistamiseksi. EAP-TLS-keskustelu alkaa perinteisellä EAP-pyynnöllä, jonka jälkeen pääsynhallintapalvelin aloittaa anojan kanssa EAP-TLS-keskustelun. Keskustelu rakennetaan TLSv1.0 tai myöhemmän salauksen päälle, jossa pääsynhallintapalvelin varmentaa anojan identiteetin tarjotun varmenteen perusteella. Vasta autentikoinnin jälkeen palvelin lähettää autentikoijalle hyväksynnän portin avaukseen. (Simon ym., 2008, ss. 4–6) Kuvassa 6 on Cisco Systemsin havainnekuva EAP-TLS-keskustelun vaiheista.

Kuva 6. EAP-TLS-autentikaation toiminta (Cisco Systems, 2020).



3.2.3 RADIUS

Useimmiten pääsynhallintapalvelin sijaitsee IP-verkon takana keskitetysti konesalissa palomuurien suojassa. Tässä tapauksessa EAP ei sellaisenaan sovi yhteysprotokollaksi, koska sen liikenne tapahtuu lähiverkon sisällä, eikä käytä IP-yhteyttä. Tästä syystä autentikointiin käytetään yleensä AAA-protokollaa, ja näistä yleisin on RADIUS. Myös muita vaihtoehtoja, kuten Diameter on olemassa. (IEEE, 2020, s. 17)

RADIUS-autentikoinnissa autentikoija luo pääsypyynnön (Access-Request), joka saattaa sisältää esimerkiksi käyttäjänimen, porttitiedon ja vaikka käyttäjältä pyydetyn salasanan. Salasana koodataan MD5-tiivisteeksi, ja pääsypyyntö välitetään RADIUS-palvelimelle. RADIUS-palvelin tarkistaa pyynnön ja vastaa siihen oman autentikointisäännöstönsä perusteella. Autentikoijan tulee olla RADIUS-palvelimelle hyväksytty autentikoija, ja tuntea palvelimelle määritelty salasana. RADIUS-palvelin vastaa pyyntöön joko lisätietopyynnöllä (Access-Challenge), hyväksynnällä (Access-Accept) tai hylkäyksellä (Access-Reject). (Rigney ym., 2000, ss. 5–7)

RADIUS ei ole osa 802.1X-standardia, vaan oma protokollansa, joka pystyy kuljettamaan standardin mukaista EAP-protokollaa. Muita RADIUS-protokollan tukemia

autentikointiprotokollia ovat esimerkiksi CHAP (Challenge Handshake Authentication Protocol) tai PAP (Password Authentication Protocol). RADIUS kattaa AAA-palvelimen muutkin toiminteet, kuten valtuutuksen (Authorization) ja kirjanpidon (Accounting). Tästä syystä RADIUS-protokollalla voidaan välittää muutakin tietoa kuin pelkkä autentikointi. Valtuutustiedolla voidaan määrittää esimerkiksi tietty VLAN (Virtual LAN), johon portti kytketään, tai käyttäjän oikeudet verkkolaitteen hallinnoinnissa. Kirjanpito välittää pääsynhallintapalvelimelle tietoja liitetystä laitteesta ja tapahtumista valvontaa varten. (Rigney ym., 2000, ss. 5–8)

Diameter-protokolla on RADIUS-protokollasta jatkokehitetty autentikointiprotokolla. Siinä missä RADIUS on alun perin kehitetty autentikoimaan puhelinverkon yli tapahtuvia yhteyksiä ja terminaaliyhteyksiä, on Diameter kehitetty alusta alkaen IP-maailmaan. Diameter käyttää siirtotienä TCP- tai SCTP (Stream Control Transmit Protocol), jotka ovat yhteydellisiä protokollia ja siten varmempia kuin RADIUS-protokollan käyttämä UDP. Lisäksi Diameter tukee yhteyden salausta TLS/TCP ja DTLS/SCTP (Datagram Transport Layer Security) protokollilla. Diameter on taaksepäin yhteensopiva RADIUS-protokollaa käyttävien ympäristöjen kanssa. (Fajardo ym., 2012, ss. 7–8)

3.3 IPsec-protokolla

IPsec on kokoelma protokollia, joiden tarkoitus on tuottaa yhteensopivia ja korkealaatuisia kryptografisia salausmenetelmiä IP-liikenteelle. IPsec muodostuu pohjimmiltaan neuvotteluprotokollasta IKE ja sen uudemmasta versiosta IKEv2, sekä liikenteen salauskomponenteista AH (Authentication Header) ja ESP (Encapsulation Security Payload), jotka salaavat liikenteen. (Kent & Seo, 2005, ss. 5–6)

IPsecin protokollat ovat rakenteeltaan modulaarisia ja suunniteltu käytettäväksi monien algoritmien kanssa. Tämä antaa mahdollisuuden toteuttaa IPsec mahdollisimman tarkoituksenmukaisella algoritmilla. IPsec ei myöskään yleensä näy verkon käyttäjille mitenkään, jolloin esimerkiksi sovellusten ei tarvitse tietää liikennöivänsä salatusti. (Kent & Seo, 2005, s. 6)

SA (Security Association) on IPsec-salatun yhteyden perusrakennuspalikka. SA on yksisuuntainen yhteys, joka tarjoaa turvallisuutta sen kuljettamalle liikenteelle. Tyypillisessä tapauksessa laitteiden välillä on kaksisuuntainen liikenne, jolloin laitteelle tulee muodostaa kaksi SA:ta. IKE-protokolla vastaa SA-parien luomisesta ja ylläpitämisestä. SA voi olla joko kuljetus- tai tunnelointimoodissa. Kuljetusmoodin SA salaa koko liikennepaketin SA:n lähdeosoitteesta kohdeosoitteeseen, kun taas tunnelimoodin SA luo IP-tunnelin, jonka sisällä liikenne salataan. Tunnelimoodin SA ei salaa tunnelin sisäisiä lähde- ja kohdeosoitteita, ainoastaan datan. (Kent & Seo, 2005, ss. 11–16)

AH on protokolla, joka tuottaa tietoliikenteelle muuttumattomuuden tarkastettavuuden. AH ei itsessään salaa liikennettä, vaan toimii ainoastaan IP-osoitelohkon aitouden varmentajana, jotta varmistutaan lähettäjän aitoudesta ja paketin muuttumattomuudesta. AH myös estää saman paketin uudelleenlähettämisen linkittymällä IKE:llä neuvoteltuun SA:han. AH:ta voi käyttää yksinään tai yhdessä ESP:n kanssa, mutta tällöin molemmat vaativat oman SA:n. (Kent, 2005a, s. 3)

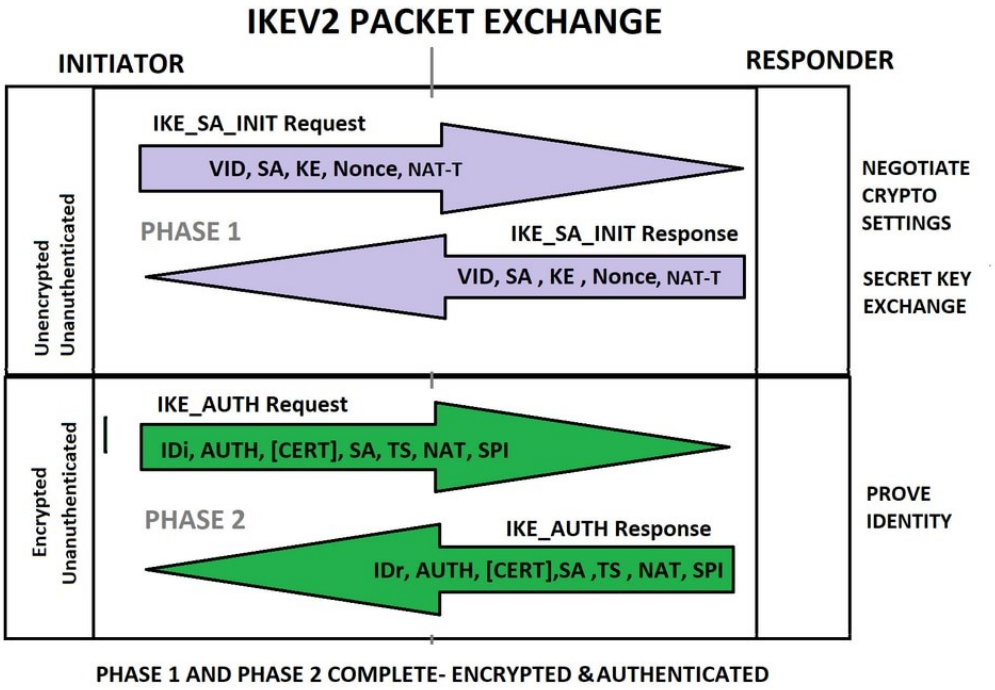
ESP on OSI-mallin 3-tason protokolla, joka tuottaa tietoliikenteelle luottamuksellisuutta salaamalla liikenteen tai muuttumattomuuden tarkastettavuutta, tai molempia. ESP ei itsessään ota kantaa salauksessa tai muuttumattomuuden tarkastuksessa käytettävään salausalgoritmiin tai avaimeen, vaan ne on neuvoteltu IKE:n toimesta ja sidottu SA:han. Tyypillisesti ESP:tä käytetään molempien osien tuottamiseen, jolloin ei tarvita AH:ta erikseen. (Kent, 2005b, ss. 3–4)

3.3.1 IKEv2

IKE-protokollan tärkein tehtävä on autentikoida kahden toimijan välinen yhteys sekä luoda ja ylläpitää SA:t. IKE neuvottelee käytettävät algoritmit ja salausavaimet ESP:tä tai AH:ta varten. IKE myös tarkkailee yhteyden tilaa ja muodostaa yhteyden uudestaan tarvittaessa. (Kaufman ym., 2014, ss. 5–7)

Kuvassa 7 kuvataan IKEv2-neuvottelun vaiheet. IKEv2 suorittaa neuvottelun pyyntö–vastaus viestipareilla (Request–Response). Ensimmäinen viestipari IKE_SA_INIT neuvottelee salausalgoritmit, vaihtaa kertakäyttöavainta (nonce) ja suorittaa avaintenvaihdon Diffie-Hellman metodilla. Seuraavassa viestiparissa IKE_AUTH suoritetaan autentikointi ja vaihdetaan identiteettejä, salasanoja (PSK) tai varmenteita. (Kaufman ym., 2014, ss. 9–10)

Kuva 7. IKEv2 neuvottelu (Cisco Systems, 2021).



Kaikki loput IKEv2-paketit on salattu näiden pakettien neuvottelemalla salauksella. Kun molemmat osapuolet ovat saaneet autentikoitua toisen osapuolen ja hyväksyttyä salauksen, luodaan SA:t CREATE_CHILD_SA viestipareilla. Tämän jälkeen muodostetaan AH tai ESP ja aletaan liikennöimään näillä osapuolten välillä. Yhteyden katketessa tai salausavaimen

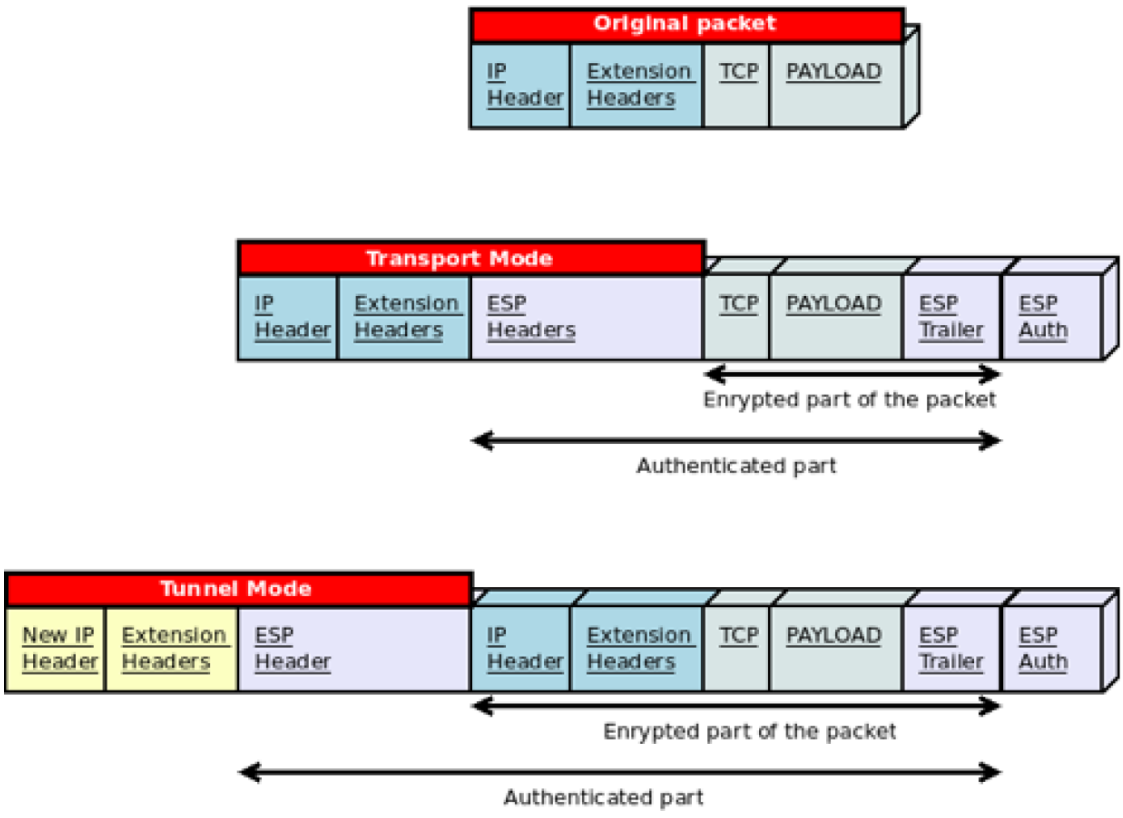
uusinnassa käydään uudelleen IKEv2-neuvottelu tarvittavilta osin. (Kaufman ym., 2014, ss. 9–17)

3.3.2 MTU

MTU (Maximum Transmission Unit) on TCP:n (Transmission Control Protocol) ominaisarvo, joka määrittää suurimman välitettävän pakettikoon. Tätä arvoa suuremmat paketit tulee pilkkoa pienempiin paketteihin, jotta ne pääsevät verkosta läpi. Tätä prosessia kutsutaan fragmentoinniksi. Fragmentointi voidaan tehdä päätelaitteessa tai jossakin reitin varrella olevassa reitittimessä. (Eddy ym., 2022, s. 3.7.)

IPsec-salaus lisää TCP-paketin otsikkolohkon kokoa, mikä tarkoittaa sitä, että paketin koko kasvaa. Kun verkossa on matkalla yhteyksiä rajoitetulla MTU:lla, täytyy paketin datasisältöä pienentää, jotta salatut paketit menisivät läpi. Tunnelimoodin SA osaa fragmentoida ja koota paketteja, mutta sillä on vaikutusta salauksen suorituskykyyn. Eri SA-moodien vaikutus pakettikokoon ilmenee hyvin kuvasta 8. (Kent & Seo, 2005, ss. 15–16) IPsecin vaikutus pakettikokoon riippuu käytettävistä algoritmeista, mutta opinnäytetyön tekijän kokemuksen perusteella hyvä nyrkkisääntö on 100 bittiä per salauskerros.

Kuva 8. ESP-paketin kehys eri SA-moodeissa. (Bajrami, 2019).



4 Opinnäytetyön toteutus

Opinnäytetyö toteutettiin rakentamalla laboratorioympäristöön virtuaalinen testiverkko ja sille varmennetuotannon tarvitsemat tukipalvelut. Testiverkon laitteille luotiin varmenteet ja niitä käytettiin eri käyttötapausten toteuttamiseksi.

Testiverkko ja tarvitut tukipalvelut toteutettiin kokonaisuudessaan virtuaalisina. Palvelimet virtualisoitiin VMware-virtualisointialustalla ja verkkolaitteet GNS3-verkkosimulaattorilla.

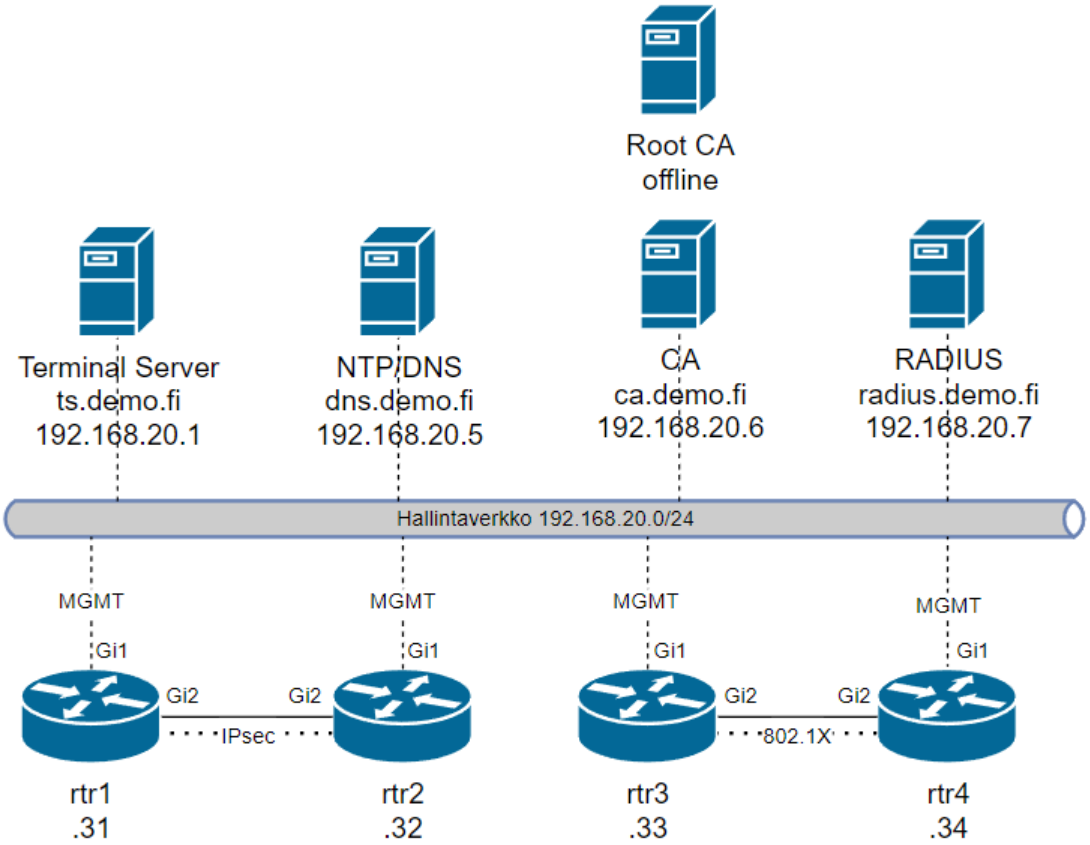
4.1 Testiympäristö

Testiympäristö rakennettiin fyysisen palvelimen sisällä olevien VMware-virtuaalikoneiden varaan. Käytettävissä oli Ubuntu 22.04 -käyttöjärjestelmä, GNS3-verkkosimulaattori ja laiteohjelmistot Cisco CSR1000V ja Cisco Catalyst 8000V. Varmennepalvelu on lähtökohtaisesti laitevalmistajariippumaton, kunhan käytettävästä laitteesta löytyy tuki varmenteille.

Molemmat laitemallit tukevat varmenteita, mutta IPsec-tuki löytyy ainoastaan CSR1000V laitteesta, kun taas 802.1X-tuki on ainoastaan C8000V laitteella. Tästä syystä testiympäristöön rakennettiin 2 kahden reitittimen verkkoa, joilla liikennettä pystytään testaamaan. 802.1X-autentikoija on yleensä kytkin, eikä reititin, mutta kytkinlaitteiden virtualisointi ei ole ainakaan GNS3:lla kovin realistinen. Varmenteiden tuominen eri valmistajien laitteille tapahtuu samoilla periaatteilla, mutta komentojen ja valikoiden osalta tulee tukeutua valmistajan dokumentaatioon.

Testiympäristöön tehtiin joitakin oletusarvoja ja arkkitehtuuriratkaisuja. Ympäristön päädomainiksi valittiin demo.fi. Tähän domainiin luotiin DNS-alue ja kaikki tukipalvelimet sekä verkkolaitteet. Ympäristöön asennettiin virtuaaliset tukipalvelimet ja niiden ohjelmistot internetistä, jonka jälkeen ympäristö irrotettiin internetistä kokonaan. Palvelimia ei kahdennettu, eikä palvelimia suojattu erillisellä palomuurilla. Palvelimien ohjelmistopalomuureista sallittiin vain tarvittavat protokollat. Tuotantoympäristössä suojaus tulee luonnollisesti ottaa paremmin huomioon. Testiympäristön järjestelyt on esitetty kuvassa 9.

Kuva 9. Testiympäristön rakenne.

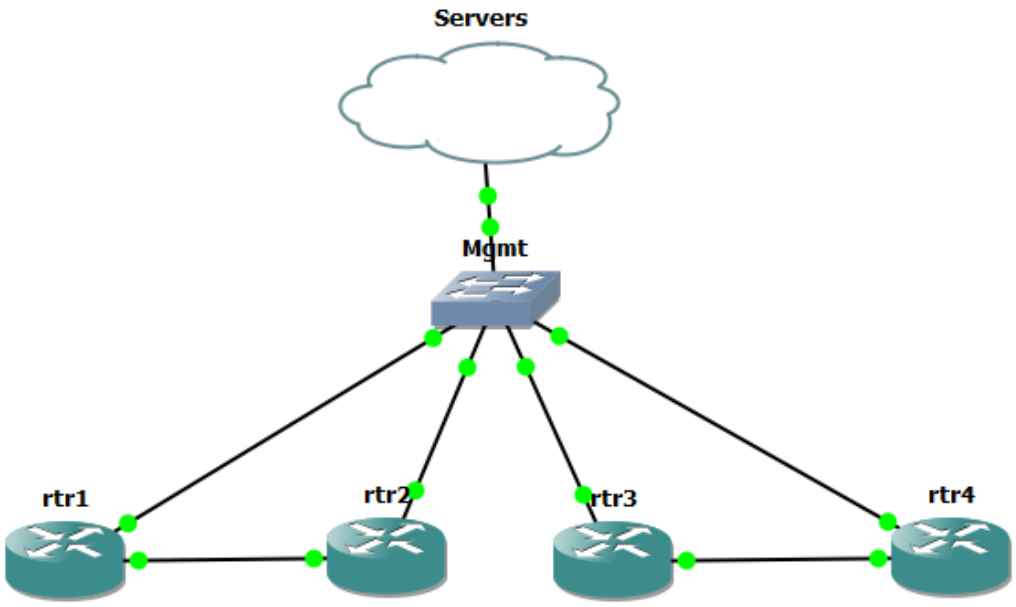


4.1.1 GNS3 simulaatioympäristö

GNS3 on ilmainen reaaliaikainen verkkosimulaattori, joka tarjoaa työkaluja useiden eri valmistajien verkkolaitteiden simulointiin ja konfiguraatioiden testaamiseen ilman tarvetta fyysisille laitteille ja kytkennöille. GNS3:lle on tarjolla ohjelmistopaketteja niin reitittimille, palomuureille kuin palvelimille ja yksittäisille sovelluksillekin. Osa paketeista on saatavilla ilmaiseksi, mutta osa vaatii laitevalmistajan sopimuksen tai käyttäjätilin paketin lataamiseksi. Ladattavissa sovelluksissa hyödynnettiin toimeksiantajan sopimuksia.

Simulaattorissa verkkotopologian muuttaminen onnistuu graafisen käyttöliittymän kautta hiiren klikkauksella, eikä fyysisiä muutoksia tarvita. Kuvakaappaus laboratorioverkon rakenteesta GNS3-simulaattorissa on kuvassa 10.

Kuva 10. Laboratorioverkko GNS3-simulaattorissa.



4.1.2 Tukipalvelimet virtuaalikoneilla

Varmenteiden elinkaaren hallintaa varten joidenkin tukipalveluiden olemassaolo on vähintäänkin suositeltavaa, mutta usein jopa välttämätöntä järjestelmän toimivuuden varmistamiseksi.

Varmenteiden voimassaolo tarkistetaan sulkulistalta. Sulkulista on varmentajan allekirjoittama lista, joka julkaistaan HTTP-, LDAP- tai FTP-palvelimella. Testiympäristössä tavaksi valittiin helpoimmin toteutettava HTTP. HTTP-palvelimeksi valittiin NGINX. Toinen tapa sulkulistan jakamiseen on OCSP-protokolla, joka toteutettiin testejä varten OpenSSL-kirjaston ocsp-työkalulla. Tuotantoympäristöissä on suositeltua tukeutua kaupallisiin tuettuihin sovelluksiin.

Sulkulistan sijainti ilmoitetaan varmenteessa itsessään web-osoitteena, joka perustuu DNS-nimeen. Tästä syystä yhtenä tukipalveluna tarvittiin DNS-palvelu. DNS rakennettiin Ubuntu-käyttöjärjestelmään Bind9-sovelluksella.

Varmenteiden voimassaolo määritellään varmenteen luontihetkellä aikaan sidottuna. Näin ollen laitteiden kellonaika on oleellisessa osassa varmenteiden toimivuutta. NTP-palvelulla

varmistetaan, että verkon laitteilla ja varmennepalvelimella on synkronoitu aika. NTP-palvelu rakennettiin yhdessä DNS-palvelun kanssa samalle palvelimelle. Ubuntun ntp-sovellus huolehti aikapalvelusta.

NTP-palvelusta on huomattava, että tässä käyttötapauksessa olennaista ei ole niinkään oikea kellonaika kuin se, että kaikki laitteet ovat samassa ajassa. Tuotantoympäristössä, varsinkin ulkomaailmaan yhteydessä olevassa, täsmällinen kellonaika on luonnollisesti välttämätön, ja ympäristössä on syytä olla suunniteltu NTP-hierarkia. Joissain sovelluksissa myös tarkempi PTP-aika (Precision Time Protocol) voi olla tarpeen, mutta varmenteille riittää NTP-ajan tarkkuus.

4.1.3 Simuloitu verkko

GNS3-verkkosimulaattorilla rakennettiin pieni verkko, joka liitettiin simulaattorin Cloud-toiminnolla palvelimen VMwaren sisäisiin verkkorajapintoihin, joissa tukipalvelut julkaistiin. Näin verkkolaitteet saatiin keskustelemaan hallintarajapinnan kautta suoraan tukipalvelimien kanssa. Todellisessa tilanteessa verkkolaitteiden hallinta olisi todennäköisesti rakennettu verkkolaitteiden reititystä hyödyntäen ns. in-band hallintana, mutta testiympäristössä nähtiin järkevämmäksi toimia näin. Verkon hallinta ei muutenkaan kuulu työn sisältöön.

Verkko rakentui 2 Cisco CSR-1000V virtuaalireitittimestä rtr1 ja rtr2 sekä 2 Cisco C8000V virtuaalireitittimestä rtr3 ja rtr4. IPsec-toteutus tehtiin rtr1 ja rtr2 välille johtuen mallin tuesta IPsecille, kun taas 802.1X tehtiin rtr3 ja rtr4 välille. Kaikille reitittimille luotiin varmenteet tukipalveluiden välivarmentajalla.

4.2 Varmennetuotanto

Varmennepolussa noudatettiin kolmiportaista varmennepolkua. Verkosta irti olevalla juurivarmenteella allekirjoitettiin välivarmenne, jolla puolestaan allekirjoitettiin laitevarmenteet. Välivarmenne tuotti sulkulistat verkkoon, joiden perusteella laitteet muodostivat luottosuhteita. Kaikki varmennepalvelimet toimivat Ubuntu-alustalla ja käyttivät varmennekirjastona OpenSSL-kirjastoa.

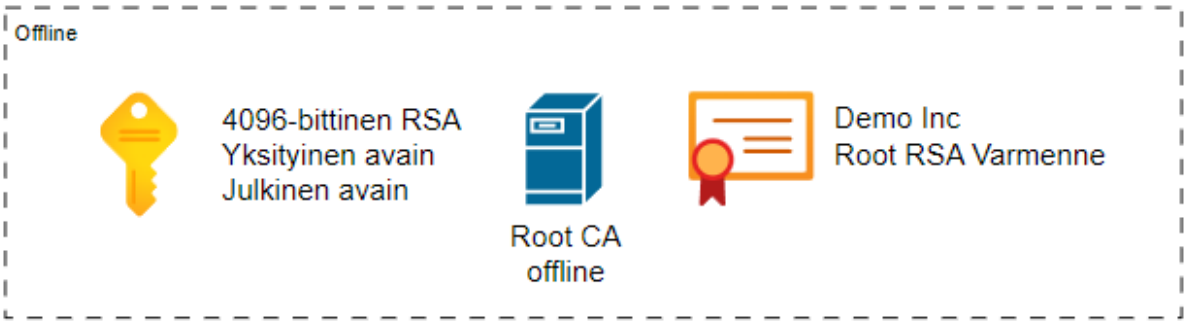
Varmenteiden luontiprosessi on tarkemmin kuvattu liitteessä 1. Toimeksiantajan varmennetuotanto ei perustunut OpenSSL-kirjastoon, mutta testiympäristössä toteutus oli helpompi tehdä avoimen lähdekoodin tuotteella.

4.2.1 Juurivarmenne

Juurivarmenne luotiin erilliselle virtuaalikoneelle, jota ei kytketty verkkoon lainkaan. Palvelimella tuotettiin OpenSSL-kirjastolla yksityinen avain 4096-bittisenä RSA-avaimena ja salattiin AES256-salauksella salasanalla, joka kirjattiin dokumentaatioon ylös. Seuraavaksi luotiin konfiguraatio juurivarmenteelle kopioimalla OpenSSL oletuskonfiguraatio uudeksi tiedostoksi ja muokkaamalla sitä. Tämän jälkeen generoitiin juurivarmenne 10 vuoden ajalle. Juurivarmenteen komponentit on kuvattu kuvassa 11.

Testimielessä luotiin myös elliptisiin käyriin perustuvat EC-avaimet ja niille varmenteet, mutta koska prosessi ei muulla tavoin eroa RSA-avaimien käytöstä, ei nähty tarkoituksenmukaiseksi luoda erillistä EC-varmennehierarkiaa. EC-avainten käyttö dokumentoitiin kuitenkin jokaiseen vaiheeseen liitteessä 1.

Kuva 11. Juurivarmennepalvelin Root CA.

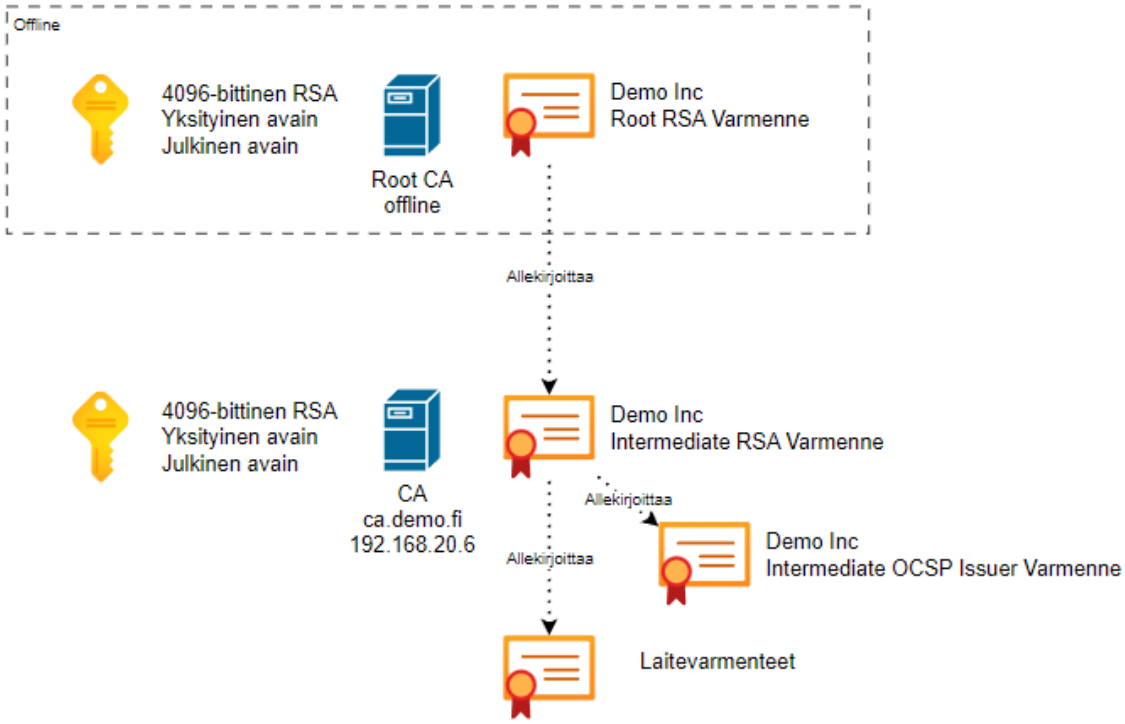


4.2.2 Välivarmenne

Välivarmenne luotiin lähiverkkoon kytkettyyn virtuaalikoneeseen nimeltään ca.demo.fi. Varmenteen luomisessa noudatettiin muuten samoja vaiheita kuin juurivarmenteessa, mutta varmenteen generoinnin sijaan generoitiin CSR (Certificate Signing Request), joka kopioitiin manuaalisesti juurivarmenteen palvelimelle, allekirjoitettiin juurivarmenteen yksityisellä avaimella ja allekirjoitettu varmenne palautettiin välivarmentajan palvelimelle. Varmennehierarkia on kuvattu kuvaan 12.

Valmis varmenne kopioitiin takaisin välivarmentajan koneelle. Sen jälkeen luotiin välivarmentajalla allekirjoitettu varmenne OCSP palvelua varten. Tätä varten generoitiin oma yksityinen avain ja CSR, jonka välivarmentaja allekirjoitti ja jonka varmenteeseen lisättiin OCSP-laajennus.

Kuva 12. Testiympäristön varmennehierarkia.



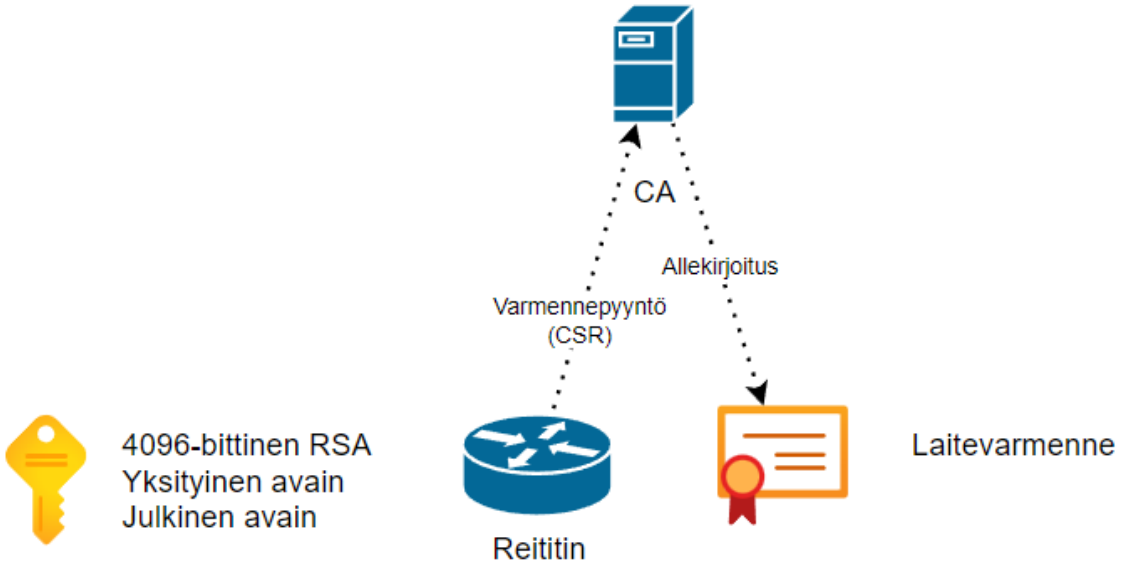
Lopuksi luotiin juurivarmenteen ja välivarmentajan CRL sulkulista, jotka julkaistiin CA-palvelimella HTTP-sivulla.

4.2.3 Laitevarmenteiden luominen

Ennen laitekohtaisen varmenteen luomista kerrottiin jokaiselle laitteelle varmenteiden hierarkia ja ladattiin laitteelle juuri- ja välivarmenteet. Tämän jälkeen luotiin laitteelle salausavainpari sekä välivarmenteelle osoitettu varmennuspyyntö. Kuvassa 13 on esitetty laitevarmenteen luomisen vaiheet.

Luotu varmennuspyyntö siirrettiin välivarmenteen palvelimelle ja allekirjoitettiin siellä välivarmentajan yksityisellä avaimella. Luotu laitevarmenne kopioitiin sen jälkeen takaisin laitteelle. Laite varmisti ennen varmenteen käyttöönottoa vielä välivarmentajan sulkulistalta, että saatu varmenne oli validi.

Kuva 13. Laitevarmenteen luominen.



4.2.4 Sulkulistaus

Laitevarmenne lisätään sulkulistalle, jos sen voimassaoloaikana epäillään laitteen tai sen yksityisen avaimen joutuneen luvattoman käyttäjän haltuun. Varmennetta, jonka voimassaoloaika on päättynyt, ei tarvitse laittaa sulkulistaan, ja erääntyneet varmenteet poistuvat myös sulkulistalta automaattisesti. Sulkulistalla olevaa varmennetta ei pitäisi hyväksyä autentikointimenetelmäksi, joskin tämä riippuu verkkolaitteiden konfiguraatiosta.

Sulkulistausta testattiin niin laitetunnistuksessa kuin IPsec-tunneloinnin autentikoinnissa. Molemmissa tapauksissa toiminto toteutettiin sekä voimassa olevalla, että sulkulistalle lisätyllä varmenteella. Sulkulistauksen testit tehtiin sekä CRL-jakelulla että OCSP-protokollalla. Sulkulistaus on dokumentoitu tarkemmin liitteessä 1.

4.2.5 Varmenteiden uusiminen

Vanhentunut tai sulkulistalle lisätty varmenne tulee uusia ennen kuin sitä voidaan käyttää verkossa uudestaan laitetunnistukseen tai IPsec-tunnelointiin. Varmenteiden uusiminen voidaan tehdä joko samalla avainparilla tai uudelleen generoidulla avainparilla. Avaimen uudelleen generointi tulee ehdottomasti tehdä, jos on epäilys yksityisen avaimen vuotamisesta tai sen murtamisen onnistumisesta. Uusi avain myös pakottaa mahdollisen hyökkääjän aloittamaan avaimen murtamisen alusta.

Käytännössä kuitenkin vähintään 2048-bittinen avain on nykyteknologialla mahdoton murtaa ihmisen elinaikana, joten avaimen generoinnin järkevyys on hieman kyseenalaista. Lisäksi avaimen vaihtuminen saattaa vaikuttaa joidenkin sovellusten toimintaan hetkittäin. Erityisesti pitää huomioida, että jos jokin tiedosto on salattu vanhalla avaimella, ei sitä saa uudella auki. Tietoliikennesalauksessa tai laitetunnistuksessa tämä ei kuitenkaan ole ongelma, koska salaus puretaan käytännössä reaaliaikaisesti, eikä tietoa säilötä liikennöintiavaimilla salattuna.

Varmenteen uusiminen tehdään käytännössä samalla tavalla kuin varmenteen luominenkin. Yksityisellä avaimella luodaan varmennuspyyntö. jonka välivarmentaja allekirjoittaa. Allekirjoitettu varmenne siirretään laitteelle samalla tavoin kuin aiemminkin. Uusimisprosessi on kuvattu tarkemmin liitteessä 1.

4.3 Autentikointi

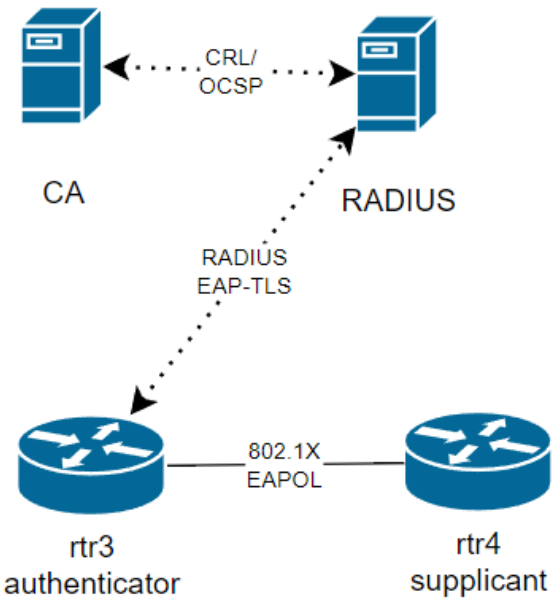
Laitevarmenteita käytettiin työssä autentikointiin sekä laitetunnistuksessa että IPsec-tunneloinnin avainneuvottelussa. Molempien käyttötapausten vaatimat ratkaisut rakennettiin laboratorioympäristöön ja testattiin toimivaksi. Rakennetun ympäristön toteutus dokumentoitiin käyttöönotto-ohjeeseen siten, että se on helposti sovellettavissa tuotantoympäristössä.

4.3.1 Laitetunnistus

Laitetunnistustestissä rtr4-niminen reititin toimii verkkoon liittyvänä reitittimenä, joka tunnistautuu 802.1X protokollalla rtr3-reitittimen tuottamalla autentikoijalla. Rtr3 pyytää rtr4:ltä varmenteen, jonka se toimittaa RADIUS-protokollalla RADIUS-palvelimelle EAP-TLS-autentikointia varten. RADIUS-palvelin tarkistaa, että varmenne on allekirjoitettu hyväksytyllä ketjulla (välivarmenteella), että varmenne on voimassa ja vielä CA-palvelimen sulkulistalta, että varmenne ei ole sulkulistalla. Jos kaikki on kunnossa, palvelin vastaa rtr3:lle, että rtr4 on luotettu ja voidaan päästää verkkoon. Kuvaan 14 on kuvattu laitetunnistuksen toiminta tässä käyttötapauksessa.

Palvelimen ja laitteen konfiguraatiot ja tarkempi toiminta on dokumentoitu liitteessä 1.

Kuva 14. Laitetunnistuksen toiminta.



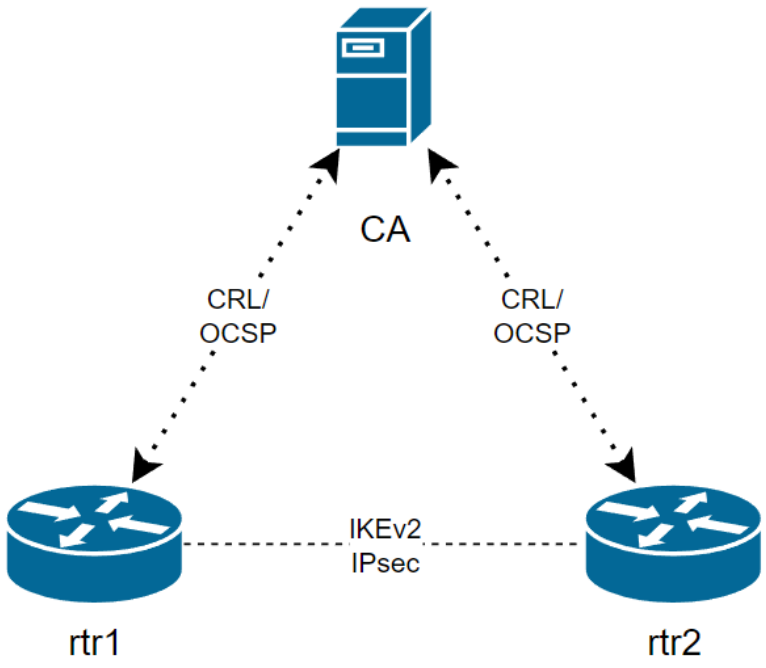
4.3.2 Salaustunneli

IPsec-salauksessa vastapään laite autentikoidaan IKEv2-neuvotteluprotokollan IKE_AUTH viestipaketeilla. Autentikointiin on käytettävissä erilaisia metodeja, joista yleisimmin käytetyt ovat Digitaalinen allekirjoitus joko RSA- tai ECDSA-avaimella, etukäteen jaettu salasana (PSK) tai kokonaan puuttuva autentikointi (NULL).

Jaetun salasanan autentikoinnissa salasanan vahvuus määrittää myös salauksen vahvuutta. Salasanat tulee myös konfiguroida joko jokaiselle linkille erikseen, tai yksi yhteinen salasana kaikille linkeille. Ensimmäinen vaihtoehto lisää konfiguraation monimutkaisuutta ja hallinnoinnin määrää erityisesti isossa verkossa. Jälkimmäinen ratkaisu taas heikentää turvallisuutta. Salasanaa käytettäessä myös vaarantuneen laitteen sulkeminen verkosta on hankalampaa, erityisesti jos linkkejä on paljon.

Varmenteita käytettäessä laitteen konfiguraatiossa IKEv2-profiiliin konfiguroidaan omalle identiteetille käytettävä varmenne, sekä vastapään autentikointiin käytettävä ylemmän tason varmenne. Laitteelle ei tarvitse kertoa vastapään varmennetta, ainoastaan ylemmän tason varmenne. Kuvaan 15 on piirretty laitteiden ja palvelimen väliset yhteydet IKEv2-neuvottelun aikana.

Kuva 15. IKEv2-neuvottelun toiminta.



IKEv2 konfiguroidaan tarkastamaan varmenteen voimassaolo sulkulistalta. Tarkastus tapahtuu aina kun IKEv2-neuvottelu käydään. Jos varmenne lisätään sulkulistalle, yhteys kuitenkin toimii ja liikennöinti jatkuu, kunnes neuvottelu käydään uudestaan ja sulkulista tarkastetaan. IKEv2 protokollan voikin useimmissa laitteissa konfiguroida katkaisemaan yhteyden, jos varmenne joutuu sulkulistalle. Tällöin katkaisu riippuu sulkulistan tarkastuksen aikavälistä. Aikaväliä voidaan säätää verkkolaitteiden konfiguraatiolla, mutta varmempi tapa on määritellä sulkulistalle voimassaoloaika.

Varmenteita käytettäessä liikenteen salauksen autentikointiin parannetaan turvallisuutta siis kahdella tavalla. Ensinnäkään salauksen autentikoinnin salasanoja ei tarvitse hallinnoida linkki- tai laitekohtaisesti eikä salauksen vahvuus ole riippuvainen salasanan vahvuudesta. Toiseksi varastetun tai muuten vääriin käsiin joutuneen laitteen liikennöinti verkkoon pystytään helposti estämään sulkulistalla.

Laitteiden konfiguraatio on dokumentoitu tarkemmin liitteessä 1.

4.4 Käyttöönotto-ohje

Toimeksiantajalle luovutettiin testausympäristön dokumentaatio käyttöönotto-ohjeena. Ohjeessa on kerrottu laboratorioympäristön palvelimien rakenne ja asennetut ohjelmistot sekä kuvattu vaiheittain käyttötapausten toteutus. Käyttöönotto-ohje löytyy opinnäytetyön liitteestä 1.

4.4.1 Tukipalvelut

Tukipalveluista ohjeessa on mainittu vaadittuina palveluina DNS ja NTP, sekä sulkulistojen jakelupalvelu HTTP-protokollalla. Tukipalveluiden konfiguraatiota ei ole tarkemmin eritelty, koska kyseessä on perustason verkkopalvelut, joiden toteutus riippuu joka tapauksessa ympäristöstä.

4.4.2 Varmennepalvelut

Varmennepalvelimista ohjeessa on laaja kuvaus ja yksityiskohtaiset käyttöohjeet kaikkiin toimenpiteisiin varmenneympäristön pystyttämiseksi. Ohjeessa käsitellään juuri- ja välivarmenteiden luominen, laitevarmenteiden allekirjoitus, varmenteiden lisääminen sulkulistalle ja sulkulistojen generointi.

4.4.3 Laiteautentikointi

Käyttöönotto-ohjeessa kuvataan RADIUS-palvelimen konfiguraatio ja käyttöönotto laitteilla. Autentikointiin käytetään EAP-TLS-protokollaa, jonka paketit kuljetetaan RADIUS-protokollalla. Ohjeessa käsitellään laitekonfiguraatiot sekä tunnistettavalle että tunnistavalle laitteelle.

4.4.4 Laitekonfiguraatiot

Pääpaino käyttöönotto-ohjeessa on laitevarmenteiden käsittelyssä. Ohjeessa on kerrottu tarvittavat konfiguraatiot ja erilaisia toteutusvaihtoehtoja laitevarmenteiden luomiseksi. Ohjeeseen on koottu ylätason varmenteiden luottosuhteiden muodostus ja niiden tuominen laitteille sekä varmenteiden käyttö laiteautentikoinnissa ja salauksen autentikoinnissa.

5 Johtopäätökset ja yhteenveto

Opinnäytetyön tavoitteena oli rakentaa laboratorioympäristöön palvelin- ja verkkorakenne laitevarmenteiden tuottamiseen ja niiden elinkaaren hallintaan Ciscon verkkolaitteille. Lopputuloksena toimeksiantajalle syntyi tarvittava testausympäristö ja käyttöönotto-ohje, jossa ympäristö on dokumentoitu ja jolla vastaavan ympäristön pystyttäminen tuotantoympäristöön onnistuu. Olennaisena osana ohjetta on verkkolaitteiden varmenteiden hallinnointi, joka toimii myös muunlaisten PKI-ympäristöjen kanssa.

Opinnäytetyön ensimmäinen tutkimuskysymys oli, miten verkkolaitteille saadaan luotua organisaatioon kytketyt varmenteet. Työssä on laajasti teoriapohjaa PKI varmenteen taustasta. Työssä on kerrottu varmenteiden suunnittelun, pystyttämisen ja hallinnoinnin prosessit yleisellä tasolla, ja liitetiedostona olevassa käyttöönotto-ohjeessa on myös selostettu yksityiskohtaisesti varmenteiden käyttö avoimen lähdekoodin OpenSSL-kirjastolla. Käyttöönotto-ohjeessa on myös yksityiskohtainen kuvaus laitevarmenteiden luomisesta ja asentamisesta laitteelle.

Toisena tutkimuskysymyksenä oli, miten verkkolaitteiden varmenteita voidaan hyödyntää laitetunnistuksessa. Opinnäytetyössä on yleiskatsaus teoriapohjaan varmenteiden käyttöön laitteiden tunnistamisessa RADIUS- ja EAP-TLS-protokollilla. Käyttöönotto-ohjeessa on yksityiskohtaisemmat ohjeet avoimen lähdekoodin FreeRADIUS-palvelimen konfigurointiin ja sen käyttöön verkkolaitteilla varmennepohjaiseen laitetunnistukseen.

Viimeinen tutkimuskysymys oli, miten verkkolaitteiden varmenteita voidaan hyödyntää liikenteen salaamisessa ja mitä etua varmenteiden käyttämisellä on salauksessa. Tietoliikenteen salaus IPsec-protokollapinolla on opinnäytetyön teoriaosuudessa selostettu ylätasolla, ja käyttöönotto-ohjeessa on yksityiskohtaiset konfigurointiohjeet varmenteiden käyttöön salauksen autentikoinnissa.

Varmenteiden käyttöönotto tuo kiistattomia hyötyjä verkon turvallisuudelle. Varmenteet vähentävät laitteiden konfiguraationhallinnan tarvetta ja parantavat tietoturvaa ja laitteiden hallittavuutta. Haittapuoleksi varmenteiden käyttöönotossa voidaan nähdä hieman monimutkaisempi ympäristön pystytys ja ylläpito.

Lähdeluettelo

Aboba, B., Blunk, L. J., Vollbrecht, J. R., Carlson, J. & Levkowetz, H. (Kesäkuu 2004). *IETF RFC 3748*. Extensible Authentication Protocol (EAP). https://datatracker.ietf.org/doc/html/rfc3748

Bajrami, V. (26.9.2019). *RedHat Enable Sysadmin.* An introduction to IPv6 packets and IPSec. https://www.redhat.com/sysadmin/ipv6-packets-and-ipsec

Cisco Systems. (27.8.2020). *Understand and Configure EAP-TLS with Mobility Express and ISE*. https://www.cisco.com/c/en/us/support/docs/wireless/mobility-express/213579-understand-and-configure-eap-tls-with-mo.html

Cisco Systems. (18.11.2021). *Troubleshoot IPsec Issues for Service Tunnels on vEdges with IKEv2*. https://www.cisco.com/c/en/us/support/docs/ip/internet-key-exchange-ike/217418-troubleshoot-ipsec-issues-for-service-tu.html

Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R. & Polk, T. (Toukokuu 2008). *IETF RFC 5280*. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. https://datatracker.ietf.org/doc/html/rfc5280

Eddy, W. M., Postel, J. & Oppermann, A. (Elokuu 2022). *IETF RFC 9293*. Transmission Control Protocol (TCP). https://datatracker.ietf.org/doc/html/rfc9293

Fajardo, V., Arkko, J., Loughney, J. & Zorn, G. (Lokakuu 2012). *IETF RFC 6733.* Diameter Base Protocol. https://datatracker.ietf.org/doc/html/rfc6733

Hellman, M. E. (5.5.2002). An overview of public key cryptography. *IEEE Communications Magazine 40.5*, 42-49. https://doi.org/10.1109/MCOM.2002.1006971

IEEE. (28.2.2020). *IEEE Standard for Local and Metropolitan Area Networks--Port-Based Network Access Control*. https://doi.org/10.1109/IEEESTD.2020.9018454

ITU-T. (13.2.2021). *Recommendation X.680-X.693 (02/21).* https://www.itu.int/rec/T-REC-X.680-X.693-202102-I/en

Josefsson, S. & Leonard, S. (Huhtikuu 2015). *IETF RFC 7468*. Textual Encodings of PKIX, PKCS, and CMS Structures. https://datatracker.ietf.org/doc/html/rfc7468

Kaufman, C., Hoffman, P., Yoav, N., Eronen, P. & Kivinen, T. (Lokakuu 2014). *IETF RFC 7296*. Internet Key Exchange Protocol Version 2 (IKEv2). https://datatracker.ietf.org/doc/html/rfc7296

Kent, S. (Joulukuu 2005a). *IETF RFC 4302*. IP Authentication Header. https://datatracker.ietf.org/doc/html/rfc4302

Kent, S. (Joulukuu 2005b). *IETF RFC 4303*. IP Encapsulating Security Payload (ESP). https://datatracker.ietf.org/doc/html/rfc4303

Kent, S. & Seo, K. (Joulukuu 2005). *IETF RFC 4301*. Security Architecture for the Internet. https://datatracker.ietf.org/doc/html/rfc4301

Moriarty, K., Nystrom, M., Parkinson, S., Rusch, A. & Scott, M. (Heinäkuu 2014). *IETF RFC 7292*. PKCS #12: Personal Information Exchange Syntax v1.1. https://datatracker.ietf.org/doc/html/rfc7292

Pittalia, P. P. (2019). A comparative study of hash algorithms in cryptography. *International Journal of Computer Science and Mobile Computing, 8(6)*, 147-152.

Rigney, C., Rubens, A. C., Simpson, W. A. & Willens, S. (Kesäkuu 2000). *IETF RFC 2865*. Remote Authentication Dial In User Service (RADIUS). https://datatracker.ietf.org/doc/html/rfc2865

Robinson, K. (21.9.2018). *Twilio Blog*. What is Public Key Cryptography. https://www.twilio.com/en-us/blog/what-is-public-key-cryptography

Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S. & Adams, C. (Kesäkuu 2013). *IETF RFC 6960*. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol. https://datatracker.ietf.org/doc/html/rfc6960

Simon, D., Aboba, B. & Hurst, R. (Maaliskuu 2008). *IETF RFC 5216*. The EAP-TLS Authentication Protocol. https://datatracker.ietf.org/doc/html/rfc5216

Thakkar, M. (30.6.2022). *Infosec Insights*. by Sectigo Store. https://sectigostore.com/blog/what-is-ocsp-ocsp-security-explained/

Weider, C. & Reynolds, J. K. (Maaliskuu 1992). *IETF RFC 1308*. Executive Introduction to Directory Services Using the X.500 Protocol. https://datatracker.ietf.org/doc/html/rfc1308

Liite 1. PKI-ympäristön käyttöönotto-ohje

Tukipalvelut

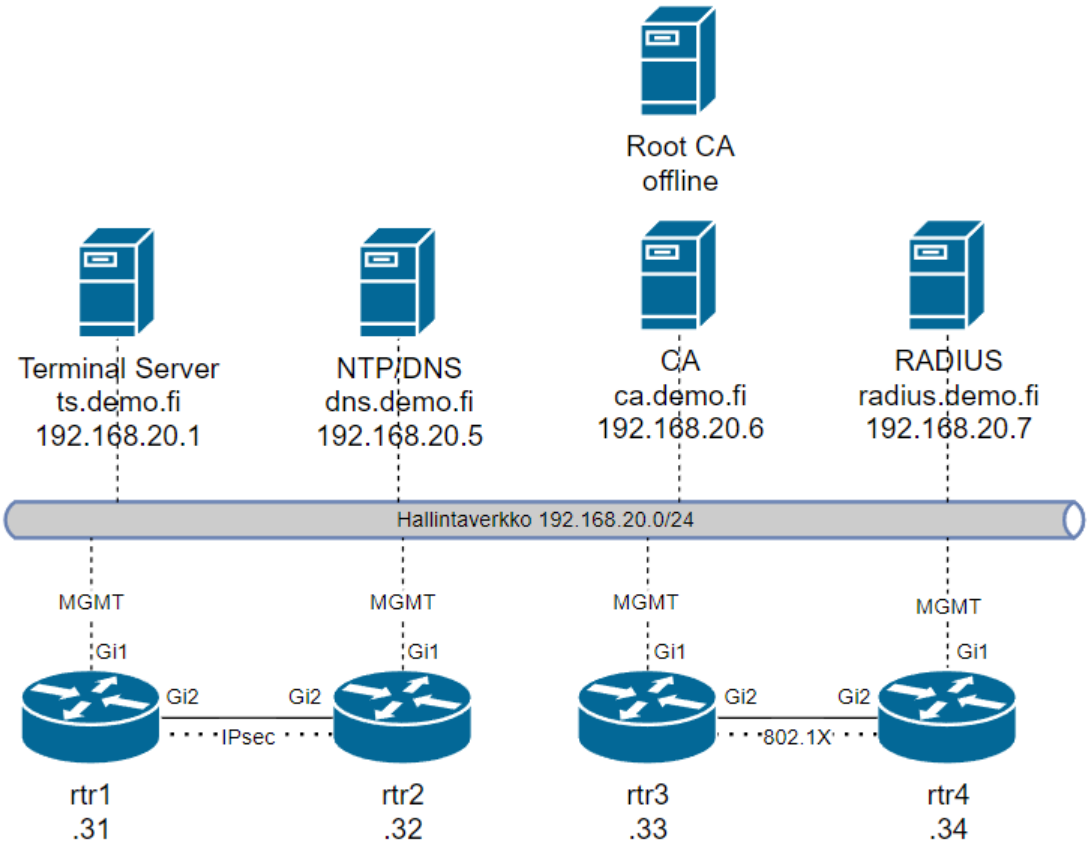
Jotta varmenteet toimisivat oikealla tavalla, tulee verkossa olla laitteille tietyt tukipalvelut olemassa. Tukipalvelut on asennettu Demoverkon GNS3-palvelimelle virtuaalikoneina, joiden pohjana toimii Ubuntu LTS 22.04.

Varmenteissa määritellään mm. palvelun DNS-nimi ja sulkulistojen (CRL ja OCSP) jakeluosoite FQDN-muodossa, jolloin DNS-nimipalvelun tulee olla toiminnassa.

Varmenteiden voimassaolo perustuu myös aikaan, jolloin verkkolaitteiden aika on myös syytä olla oikeassa ja samassa ajassa palvelinten kanssa. Tästä syystä NTP-palvelin tulee olla saatavilla.

Testiympäristön verkko on IP-osoitelohkossa 192.168.20.0/24. Testiympäristöön on rakennettu DNS-palvelin käyttäen Bind9 sovellusta, domain nimi on demo.fi. Ympäristöstä löytyy seuraavat palvelimet:

- Hyppypalvelin ts.demo.fi, 192.168.20.2
- DNS- ja NTP-palvelin dns.demo.fi sekä ntp.demo.fi, 192.168.20.5
- CA-palvelin, ca.demo.fi, crl.demo.fi ja ocsp.demo.fi, 192.168.20.6
- RADIUS-palvelin radius.demo.fi, 192.168.20.7
- Juurivarmennepalvelin root-ca, ei verkossa



Varmennepalvelut

Varmennepalvelut on asennettu kahdelle palvelimelle. Juurivarmenne on verkosta irti olevalla palvelimella ja välivarmenne tuotetaan ca-palvelimelta, joka toimii myös sulkulistojen jakelupalvelimena. Molemmille palvelimille on asennettu OpenSSL-työkalu, jolla varmenteet luodaan.

Palvelun konfiguraatiotiedosto on muokattu ohjelmiston vakiokonfiguraatiosta, johon on lisätty kokonaisuuden toimivuuden kannalta olennaiset osiot. Sama konfiguraatiotiedosto on käytössä sekä juurivarmennepalvelimella että välivarmennepalvelimella. Konfiguraatiotiedosto on tarkemmin kuvattu kappaleessa 2.3.

Juurivarmenne

Juurivarmennepalvelimella *root-ca* on PKI:ta varten seuraava kansiorakenne:

```
/pki
```

```
/pki/backup
/pki/conf
/pki/ca
/pki/ca/private
/pki/ca/requests
/pki/ca/certs
/pki/ca/crl
/pki/ca/newcerts
```

Kansiorakenteen lisäksi on luotu myös seuraavat tiedostot

```
/pki/ca/crlnumber
/pki/ca/index
/pki/ca/serial
```

crlnumber sisältää sulkulistan numeron, joka on aluksi asetettu lukuun 1000. Samoin *serial* tiedosto sisältää varmenteen sarjanumeron, joka on aluksi asetettu myös lukuun 1000. *index* sisältää tietokannan varmenteen myöntämistä alivarmenteista ja se on aluksi tyhjä tiedosto.

Avainpari ja varmenne

/pki/conf/openssl.cnf tiedostosta löytyy OpenSSL-konfiguraatio juurivarmenteelle. Juurivarmenne luodaan seuraavilla toimenpiteillä. Ensin luodaan varmenteelle salausavaimet komennolla

```
openssl genrsa -aes256 -out /pki/ca/private/cakey-4096.pem 4096
```

Komento luo 4096-bittisen RSA-avaimen tiedostoon */pki/ca/private/cakey-4096.pem* ja salaa sen AES256-salauksella. Komennon jälkeen ohjelma kysyy avaimen salaukselle salasanaa, jolla yksityistä avainta voi käyttää. Toinen vaihtoehto salasanan käytölle on luoda satunnainen salasana ja tallentaa se tiedostoon, josta se voidaan ladata aina käytön yhteydessä. Satunnainen 256-merkkinen salasana luodaan komennolla

```
openssl rand --base64 -out /pki/ca/private/capass 256
```

Ja avainta luodessa salasanatiedosto otetaan käyttöön muokkaamalla avainkomentoa seuraavasti

```
openssl genrsa -aes256 -out /pki/root/private/cakey-4096.pem -passout
file:/pki/ca/private/capass 4096
```

Tulevissa komennoissa luotu salasanatiedosto otetaan käyttöön lisäämällä komentoon vipu

```
–passin file:/pki/ca/private/capass
```

Jos halutaan RSA-avaimien sijaan käyttää EC-avaimia, komento on seuraava

```
openssl ecparam -genkey -name secp384r1 | openssl ec -aes256 -out
/pki/ca/private/cakey-secp384r1.pem –passout file:/pki/ca/private/capass
```

Komento luo EC-avaimen käyttäen elliptistä käyrää secp384r1. Mahdolliset vaihtoehtoiset käyrät saa komennolla **`openssl ecparam -list_curves`**. Käyrän luomisen jälkeen se salataan AES256-salauksella tiedostoon */pki/ca/private/cakey-secp384r1.pem*. Komennon lopussa oleva **`-passout`** vipu kertoo, että avaimen salasana haetaan tiedostosta *capass*. Ilman tätä vipua komento kysyy jälleen yksityisen avaimen salasanaa kahteen kertaan.

Huomioi, että jos juurivarmenne on luotu RSA-avaimilla, tulee kaikki alivarmenteet myös luoda RSA-avaimilla. Vastaavasti taas EC-avaimilla luotu juurivarmenne tarkoittaa, että alivarmenteet tulee myös luoda EC-varmenteella.

Avainten generoimisen jälkeen luodaan self-signed juurivarmenne komennolla

```
openssl req -config /pki/conf/openssl.cnf -new -x509 –sha512 -extensions
v3_ca -days 3650 -key /pki/ca/private/cakey-4096.pem –passin
file:/pki/ca/private/capass -out /pki/ca/certs/root.crt
```

Komento luo uuden varmennepyynnön (req) ja allekirjoittaa sen x509-varmenteeksi, joka on voimassa 10 vuotta (days). Varmenteen luomisessa otetaan laajennus v3_ca käyttöön, joka kertoo varmenteen olevan juurivarmenne. Laajennuksen yksityiskohtia voidaan muuttaa */pki/conf/openssl.cnf* tiedoston v3_ca osiossa. Käytettävä avain määritellään -key osiossa, tässä voidaan myös käyttää luotua EC-avainta. Varmenne tallennetaan -out osion tiedostoon. Varmennetta luodessa ohjelma kysyy tarkempia tietoja varmenteesta, kuten Common name (CN) jotka tallennetaan varmenteeseen. SHA512 kertoo, millä algoritmilla varmenteen allekirjoitus on luotu.

Varmenteen tiedot voi tarkastaa komennolla

```
openssl x509 -noout -text -in /pki/ca/certs/root.crt
```

Tämän jälkeen varmenteen voi kopioida käyttöön. Varmenne on ympäristössä kopioitu välivarmentajan palvelimelle, josta se on myös laitteiden käytettävissä.

Yksityinen avain on kopioitu myös */pki/backup* kansioon.

Sulkulista

Sulkulistan generointi tapahtuu komennolla

```
openssl ca -config /pki/conf/openssl.cnf -gencrl –passin
file:/pki/ca/private/capass -out /pki/ca/crl/root.crl -crldays 180
```

Juurivarmenteen sulkulista generoidaan pitkäksi ajaksi, koska juurivarmenteella allekirjoitetaan vain välivarmenne, ja välivarmenne joudutaan perumaan vain, jos se vaarantuu. Oletettavasti siis juurivarmenteen sulkulista on pääsääntöisesti tyhjä. Sulkulista kannattaa luoda vasta välivarmenteen luomisen jälkeen.

Välivarmenne

Välivarmennepalvelimella, eli *ca.demo.fi* on seuraava kansiorakenne:

```
/pki
/pki/backup
/pki/conf
/pki/html
/pki/intermediate
/pki/intermediate/private
/pki/intermediate/requests
/pki/intermediate/certs
/pki/intermediate/crl
/pki/intermediate/newcerts
```

Kansiorakenteen lisäksi luodaan myös seuraavat tiedostot

```
/pki/intermediate/crlnumber
/pki/intermediate/index
/pki/intermediate/serial
```

crlnumber sisältää sulkulistan numeron, joka on aluksi asetettu lukuun 2000. Samoin *serial* tiedosto sisältää varmenteen sarjanumeron, joka on aluksi asetettu myös lukuun 2000. *index* sisältää tietokannan varmenteen myöntämistä alivarmenteista ja se on aluksi tyhjä tiedosto.

Varmennuspyyntö

/pki/conf/openssl.cnf tiedostosta löytyy konfiguraatio välivarmenteelle. Välivarmenteen salausavaimet luodaan samalla tavoin kuin juurivarmennekin, huomioitavana että avaimen tulee olla juurivarmennetta vastaava. Eli RSA-juurelle RSA-välivarmenne ja EC-juurelle EC-välivarmenne.

Välivarmenteelle tehdään varmennuspyyntö (CSR) komennolla

```
openssl req -config /pki/conf/openssl.cnf -new -key
/pki/intermediate/private/cakey-4096.pem –passin
file:/pki/intermediate/private/capass -out
/pki/intermediate/requests/intermediate.csr
```

Huomioitava, että toisin kuin juurivarmenteessa, välivarmenteessa ei käytetä x509 valitsinta, eli varmennetta ei allekirjoiteta itse, vaan luodaan vain varmennepyyntö. Tämän jälkeen varmennepyyntö kopioidaan juuripalvelimelle ja allekirjoitetaan siellä komennolla

```
openssl ca -config /pki/conf/openssl.cnf -extensions v3_intermediate_ca
-days 3650 -md sha512 -in /pki/ca/requests/intermediate.csr –passin
file:/pki/ca/private/capass -out /pki/ca/certs/intermediate.crt
```

Komennolla varmennepyyntö allekirjoitetaan ja varmenteelle lisätään Intermediate laajennus, joka tekee varmenteesta välivarmenteen. Varmenne luodaan 1 vuoden ajaksi (days) ja käyttää allekirjoitukseen SHA512 algoritmiä. Varmenne pohjautuu välivarmenteen CSR-tiedostoon ja kirjoitetaan */pki/ca/certs/intermediate.crt* tiedostoon. Tämän jälkeen varmenne kopioidaan välivarmenteen palvelimelle.

Tässä vaiheessa kannattaa myös luoda juurivarmenteen sulkulista, ja molemmat tiedostot siirretään samalla välivarmenteen palvelimelle. Välivarmenne kopioidaan palvelimen */pki/intermediate/certs* kansioon, ja juuren sulkulista */pki/intermediate/crl* kansioon.

OCSP

Välivarmenteessa otetaan käyttöön myös OCSP sulkulistaus. Tätä varten tulee luoda oma varmenne, jota käytetään vain OCSP sulkulistojen allekirjoitukseen. Luodaan siis jälleen kerran uusi salausavain, tiedostoon */pki/intermediate/private/oscpkey-4096.pem*. Avaimella luodaan varmennepyyntö samalla tavoin kuin välivarmenteellekin, -key valintaan äsken luotu *oscpkey-4096.pem* ja -out valintaan *ocsp.csr*. Varmennepyyntö allekirjoitetaan välivarmenteen palvelimella samalla komennolla kuin välivarmenne allekirjoitettiin juurivarmenteella. -config valitsimeen *intermediate.cnf*, -in valitsimeen *ocsp.csr* ja -out valitsimeen *ocsp.crt*.

Sulkulistan jakelu

Välivarmenteen palvelimelle on asennettu myös NGINX-palvelu, jonka tehtävä on jakaa varmenteet ja sulkulistat. NGINX ohjaa vakiona kyselyt */var/www/html* kansioon. Asetuksen voi muuttaa */etc/nginx/sites-available/default* tiedostosta. Tiedostosta on muutettu server -osion alla root-kansio */pki/html* kansioon. Tämä ohjaa kyselyt kyseiseen kansioon.

Kansioon on linkitetty sekä kaikki tähän asti luodut varmenteet, että sulkulistat. Linkitys näille tehdään symbolisilla linkeillä komennoilla

```
ln -s /pki/intermediate/certs/root.crt /pki/html/root.crt
ln -s /pki/intermediate/certs/intermediate.crt
/pki/html/intermediate.crt
ln -s /pki/intermediate/certs/ocsp.crt /pki/html/ocsp.crt
ln -s /pki/intermediate/crl/root.crl /pki/html/root.crl
ln -s /pki/intermediate/crl/intermediate.crl /pki/html/intermediate.crl
```

Tiedostoille tulee tehdä oikeusmuutokset, jotta ne ovat HTML-palvelimelta käytettävissä. Sama koskee kansiorakennetta. Tässä tulee kuitenkin olla tarkkana, jottei palvelimen private-kansiota pääse lukemaan HTTP:llä. Oikeudet alla olevan listan mukaan

```
/pki                                    0655
```

```
/pki/backup                          0600
/pki/conf                            0600
/pki/html                            0655
/pki/intermediate                    0655
/pki/intermediate/private            0600
/pki/intermediate/requests           0600
/pki/intermediate/certs              0655
/pki/intermediate/crl                0655
/pki/intermediate/newcerts           0600
```

html-kansioon on myös luotu index.html, jossa luodaan linkit ladattaviin tiedostoihin. Tämä on vain käyttäjän iloksi, laitteet eivät tällaista tiedostoa tarvitse. Tiedoston sisältö on seuraava

```
<html>
<head></head>
<body>
<ul>
<li><a href="root.crt">Root CRT</a></li>
<li><a href="intermediate.crt">Intermediate CRT</a></li>
<li><a href="ocsp.crt">OCSP CRT</a></li>
</ul>
<ul>
<li><a href="root.crl">Root CRL</a></li>
<li><a href="intermediate.crl">Intermediate CRL</a></li>
</ul>
</body>
</html>
```

Konfiguraatiotiedosto

Varmennepalvelimille on luotu yhtyeinen konfiguraatiotiedosto, joka löytyy polusta */pki/conf/openssl.cnf*. Tiedoston olennaiset osiot on esitelty tässä.

`[ CA_default ]` osio pitää sisällään kansiopolut ja vakioasetuksia esimerkiksi sulkulistan voimassaoliin ja varmenteiden pituuteen. Lisäksi osiossa määritellään varmenteiden politiikat, jotka pyynnön tulee täyttää, jotta se voidaan allekirjoittaa.

`[ v3_ca ]` osiolla määritellään juurivarmenteen ominaisuudet. basicConstraints asetuksen CA:true on tärkein ominaisuus, jolla määritetään että luotavalla varmenteella voidaan allekirjoittaa muita varmenteita. Tämä pitää olla sekä juuri- että välivarmenteessa.

`[ v3_intermediate_ca ]` määrittää välivarmenteen ominaisuudet. CRL jakelupiste ja OCSP jakelupiste kerrotaan erillisillä osioilla, joihin tässä kohdassa viitataan.

`[ usr_cert ]`, `[ server_cert ]` ja `[ network_cert ]` määrittävät eri käyttötarkoituksiin tulevien varmenteiden asetukset. Näillä asetuksillä määritelläänj varmenteille sallitut käyttötavat.

`[ radius_cert ]` ja `[ ocsp ]` määrittävät RADIUS- ja OCSP palvelimien varmenteiden asetukset.

`[ crl_info ]` ja `[ ocsp_info ]` määrittävät sulkulistojen jakelupisteiden osoitteet.

Laitevarmenteet

Laitevarmenteiden osalta prosessi on kutakuinkin sama kuin väli- ja OCSP-varmenteen. Laitteelle generoidaan salausavaimet ja niiden pohjalta varmennuspyyntö. Pyyntö allekirjoitetaan välivarmenteella ja varmenne siirretään takaisin laitteelle. Prosessi voidaan toteuttaa manuaalisesti laitteen komentoriviltä tai automaatiolla SCEP-protokollalla. SCEP-protokollaa varten tarvitaan SCEP-palvelin, jona voi toimia esimerkiksi Ciscon reititin tai ISE-palvelin. Palvelimen toteutusta ei ole kuvattu tässä dokumentissa, ainoastaan sen käyttö laitteelta.

Manuaalinen prosessi

Manuaalinen prosessi tapahtuu komentoriviltä, jossa luodaan CSR ja kopioidaan se ruudulta palvelimelle. Palvelimelta varmenne siirretään laitteelle myös kopioimalla komentoriville.

Verkkolaitteen konfiguraatio

Ciscon laitteissa varmenteet määritellään trustpointeina. Koko varmenneketju juurivarmenteesta alaspäin kuvataan truspointeihiin, jonka jälkeen kyseiset varmenteen tuodaan laitteelle ja varmennuspyyntö luodaan trustpointiin liittyen.

Ensin määritellään tukipalvelut DNS ja NTP.

```
ip domain name demo.fi
ip name-server 192.168.20.5
ntp server ntp.demo.fi prefer
```

Tämän jälkeen luodaan salausavaimet varmennetta varten. Avaimet luodaan varmenteiden mukaisesti RSA tai EC muotoisina.

```
crypto key generate rsa general-keys exportable label certRSA modulus
4096
crypto key generate ec keysize 384 exportable label certEC
```

Juurivarmenne

Sen jälkeen luodaan trustpoint juurivarmenteelle.

```
crypto pki trustpoint root
 enrollment terminal pem
 fingerprint 0167A7B1E8E9360E7967CC041714B114956AE81F
 revocation-check none
```

Enrollment tarkoittaa sitä, millä tavalla varmenne laitteelle tuodaan. Valittu vaihtoehto tarkoittaa, että varmenne tuodaan kopioimalla se komentoriville. Fingerprint komento kertoo varmenteen sormenjäljen, johon ladattua varmennetta verrataan. Sormenjäljen saa varmennepalvelimella komennolla

```
openssl x509 -noout -in root.crt -fingerprint
```

On huomioitava, että komento antaa sormenjäljen SHA1-muodossa kaksoispisteellä eroteltuna hexana, esim BF:34:24… Cisco ei kuitenkaan tätä muotoa hyväksy, vaan kaksoispisteet tulee poistaa välistä. Fingerprint komennon voi myös jättää pois, tässä

tapauksessa laite näyttää sormenjäljen ja kysyy, hyväksytäänkö se. Sormenjäljen tulosteen voi muuntaa reitittimelle sopivaan muotoon myös esim sed-komennolla seuraavasti:

```
openssl x509 -noout -in root.crt –fingerprint | sed ’s/://g’
```

HTTP-palvelimelta nouto

Varmenteen pystyy myös noutamaan HTTP-protokollalla. Tätä varten tulee tehdä erillinen enrollment profiili, jossa noutopaikka määritetään

```
crypto pki profile enrollment root-http
 authentication url http://ca.demo.fi
 authentication command GET /root.crt
```

Profiilissa määritellään palvelimen url, josta varmenteen voi noutaa. `authentication` tarkoittaa trustpointiin liitettyä allekirjoittajaa, ja `enrollment` laitteen varmennetta, joka allekirjoituksella luodaan. Profiili otetaan käyttöön trustpointissa muuttamalla enrollment asetusta seuraavasti

```
crypto pki trustpoint root
 enrollment profile root-http
```

Varmenteen asennus

Tämän jälkeen juurivarmenne ladataan laitteeseen komennolla `crypto pki authenticate root`. Jos yhteys palvelimeen on kunnossa ja nimipalvelu toimii, laite hakee palvelimelta root.crt tiedoston ja liittää sen trustpointiin. Onnistunut tuonti näkyy alla olevana viestinä.

```
Certificate has the following attributes:
       Fingerprint MD5: F99505E7 F271BE11 DC8000CE 24B1011F
      Fingerprint SHA1: 0167A7B1 E8E9360E 7967CC04 1714B114 956AE81F
Trustpoint Fingerprint: 0167A7B1 E8E9360E 7967CC04 1714B114 956AE81F
Certificate validated - fingerprints matched.
Trustpoint CA certificate accepted.
```

Välivarmenne

Myös välivarmenteelle luodaan trustpoint. Tässä trustpointissa määritellään myös laitteen varmenteeseen liittyvät asiat.

```
crypto pki trustpoint intermediate
 enrollment terminal pem
 usage ike
 usage ssl-server
 usage ssl-client
 serial-number
 fqdn rtr1.demo.fi
 subject-name cn=rtr1.demo.fi
 chain-validation continue root
 crl cache delete-after 60
 revocation-check crl
 hash sha512
 rsakeypair certRSA
 eckeypair certEC
```

HUOM! Varmenteessa käytetään vain jompaakumpaa, **`rsakeypair`** tai **`eckeypair`**. **`usage`**-komennot määrittävät varmenteen käyttötapaukset, **`serial-number`** kertoo että varmenteelle luodaan sarjanumero, **`fqdn`** ja **`subject-name`** ovat varmenteeseen koodattavia tietoja laitteen identiteetistä. **`chain-validation`** kertoo, että varmenne on allekirjoitettu juurivarmenteella ja tulee tarkastaa sitä vasten. **`revocation-check`** kertoo, että sulkulistat noudetaan välivarmenteen CRL-osoitteesta ja poistetaan 60 minuutin kuluttua (**`crl cache`**) ja **`hash`** kertoo allekirjoitukseen käytettävän algoritmin.

`revocation-check` komennolle voi antaa useamman vaihtoehdon, ja laite tarkastaa sulkulistat annetussa järjestyksessä. Jos listalla viimeisenä on **`none`**, laite tarkastaa ensin sulkulistat ja jos se ei saa yhteyttä mihinkään sulkulistaan, se lopuksi hyväksyy varmenteen, jos se on voimassa.

Sulkulistoista on huomioitava, että Cisco ei hyväksy PEM-muotoista sulkulistaa, vaan sen tulee olla DER-muodossa. OpenSSL tuottaa vakiona PEM muotoisen sulkulistan, joka pitää muuntaa erikseen DER muotoon komennolla

```
openssl crl -in /pki/intermediate/crl/intermediate.crl -outform DER -out
/pki/intermediate/crl/intermediate.crl
```

Kuten juurivarmenteessakin, myös välivarmenne voidaan hakea HTTP-protokollalla. Välivarmenteelle luodaan oma profiili

```
crypto pki profile enrollment intermediate-http
 authentication url http://ca.demo.fi
 authentication command GET /intermediate.crt
 enrollment terminal
```

Ja se otetaan jälleen käyttöön välivarmenteen trustpointissa

```
crypto pki trustpoint intermediate
 enrollment profile intermediate-http
```

Tämän jälkeen noudetaan välivarmenne palvelimelta komennolla **`crypto pki authenticate intermediate`**. Varmenne tarkistetaan automaattisesti juurivarmennetta vasten ja hyväksytään tai hylätään sen mukaan.

Laitevarmenne

Laitevarmennetta varten tehdään ensin varmennuspyyntö komennolla **`crypto pki enroll intermediate`**. Pyyntö luo ja tulostaa ruudulle CSR-pyynnön, joka tulee kopioida ruudulta ja viedä varmennepalvelimelle. Pyyntö on PEM-muodossa, mutta laite jättää siitä otsikkokentän pois, ja OpenSSL ei ymmärrä pyyntöä. Pyyntöä pitää muokata siten, että ennen pyyntöä on otsikko ja pyynnön lopussa alaotsikko seuraavasti

```
-----BEGIN CERTIFICATE REQUEST-----
MIIErDCCApQCAQAwRjEU……I3TTAW0jhe4onWuCf2FhIyd0k
-----END CERTIFICATE REQUEST-----
```

Tämän jälkeen pyyntö voidaan allekirjoittaa välivarmenteella ja varmenne kopioida laitteelle

```
openssl ca -config /pki/conf/openssl.cnf -extensions network_cert -days
365 -md sha512 –passin file:/pki/intermediate/private/capass -in
/pki/intermediate/requests/rtr1.csr -out
/pki/intermediate/certs/rtr1.crt
```

Kun varmenne on luotu, se tulostetaan komennolla **`cat /pki/intermediate/certs/rtr1.crt`** ja kopioidaan leikepöydälle. Laitteella annetaan komento **`crypto pki import intermediate certificate`**, jonka jälkeen laite pyytää liittämään varmenteen komentoriville.

Varmenteen liittämisessä on huomioitava, että OpenSSL tuottaa varmennetiedostoon sekä tekstimuotoisen, että PEM-muotoisen varmenteen. Vain PEM-muotoinen osa syötetään laitteelle. PEM-muotoinen osa näyttää tältä:

```
-----BEGIN CERTIFICATE-----
MIIGjDCCBHSgAwIBAgICIAEwDQYJKoZ……ah8Yj8GrQHMQaQXKmLSnmL7nE
-----END CERTIFICATE-----
```

Varmenne tarkistetaan automaattisesti välivarmennetta vasten ja hyväksytään tai hylätään. Asennetut varmenteet voi tarkistaa laitteelta komennolla **`show crypto pki certificates`**. Asennettu varmenne näyttää esimerkiksi tältä

```
Certificate
  Status: Available
  Certificate Serial Number (hex): 2000
  Certificate Usage: General Purpose
  Issuer:
    cn=ca.demo.fi
    o=Demo Inc
    st=Finland
    c=FI
  Subject:
    Name: rtr1.demo.fi
    cn=rtr1.demo.fi
  CRL Distribution Points:
    http://crl.demo.fi/intermediate.crl
  Validity Date:
    start date: 13:21:00 UTC Apr 2 2024
    end   date: 13:21:00 UTC Apr 2 2025
  Associated Trustpoints: intermediate
```

Automaattinen prosessi SCEP:llä

SCEP-protokolla on Ciscon alun perin kehittämä protokolla varmenteiden automaattiseen noutoon ja uusimiseen. SCEP on vanha protokolla, joka lähettää palvelimelle PKCS#7 muotoisia pyyntöjä, joiden sisälle on koodattu CSR PKCS#10 muodossa. Pyyntöön lisätään ennalta määritelty salasana, jonka perusteella varmenne myönnetään.

Ciscon dokumentaatio SCEP:stä löytyy linkistä https://www.cisco.com/c/en/us/support/docs/security-vpn/public-key-infrastructure-pki/116167-technote-scep-00.html.

SCEP:lle on nykyaikaisempia ja turvallisempia protokollia, kuten ACME, EST ja CMS. Laitteiden tuki eri protokollille kuitenkin vaihtelee, ja esimerkiksi Cisco ja Juniper tukevat yhteisesti vain SCEPiä.

Verkkolaitteen konfiguraatio

Ciscon laitteille varmenteiden tuominen SCEP:llä tapahtuu hyvin samalla tavoin kuin terminaalinkin kautta. SCEP:llä kuitenkin määritellään trustpointille enrollment url, joka vastaa SCEP-palvelimen osoitetta. Lisäksi määritellään käytettävä salasana.

```
crypto pki trustpoint intermediate
 enrollment url http://scep.demo.fi
 password ********
```

SCEP-protokollalla voidaan noutaa niin juuri- kuin välivarmennekin samoin kuin http-palvelimelta komennolla `crypto pki authenticate intermediate`. On huomioitava, että laite lähettää pyynnön varmenteesta trustpointin nimellä, eli tässä tapauksessa SCEP-palvelimelle lähtee pyyntö intermediate varmenteesta. Konfiguraation pitää siis täsmätä palvelimen konfiguraatioon.

Jos trustpointin konfiguraatioon on annettu komento `auto-enroll`, haetaan laitevarmennetta automaattisesti heti autentikoinnin jälkeen, jolloin käyttäjän toimia ei enää tarvita. Jos tämä kuitenkin puuttuu, haetaan varmenne laitteelle samalla komennolla kuin aiemminkin, eli `crypto pki enroll intermediate`. Varmenne haetaan palvelimelta automaattisesti, eikä import-komentoa tarvita. Varmenne myös uusitaan automaattisesti sen vanhentuessa.

Sulkulistaus

Sulkulistaus tehdään CA-palvelimella. Vaarantuneen tai varastetun laitteen lisätään sulkulistalle seuraavalla komennolla

```
openssl ca -config /pki/conf/openssl.cnf -revoke
/pki/intermediate/certs/rtr1.crt
```

Komento lisää rtr1.crt varmenteen sulkulistalle. Luodessa varmenteita openssl tallentaa uudet varmenteet sarjanumeron perusteella openssl.cnf tiedoston määrittämään newcerts kansioon, tässä tapauksessa /pki/intermediate/newcerts. Jos jostain syystä certs kansiossa ei ole oikeaa varmennetta, sen pystyy lisäämään sulkulistalle myös newcerts kansiosta.

Sulkulistalle lisäämisen jälkeen sulkulista pitää myös jakaa laitteille. Se tapahtuu joko generoimalla CRL tai suoraan OCSP-palvelulla. Palvelut eivät ole toisiaan poissulkevia, vaan niitä voidaan hyödyntää varmentavina palveluina.

CRL

CRL sulkulistan generointi vaatii erikseen generoinnin, ja se tapahtuu alla olevalla komennolla

```
openssl ca -config /pki/conf/openssl.cnf -gencrl –passin
file:/pki/intermediate/private/capass -out
/pki/intermediate/crl/intermediate.crl -crldays 30
```

Sulkulistan generoinnin jälkeen se pitää vielä muuntaa DER-muotoon Ciscon laitteita varten. Muunnos tapahtuu seuraavalla komennolla.

```
openssl crl -in /pki/intermediate/crl/intermediate.crl -outform DER -out
/pki/intermediate/crl/intermediate.crl
```

Sulkulistan nimeämisessä on huomioitava, että sen tulee täsmätä varmenteessa ilmoitettuun tiedostopolkuun, joka määritellään */pki/conf/openssl.cnf* tiedostossa. Muussa tapauksessa sulkulistan polku joudutaan erikseen määrittelemään jokaiselle laitteelle.

OCSP

OCSP on automaattinen protokolla, joka vastaa laitteiden sulkulistakyselyihin varmennepalvelun indeksointitiedoston perusteella. Näin erillistä sulkulistaa ei tarvitse generoida.

OCSP palvelun tulee kuitenkin olla päällä. OpenSSL tarjoaa mahdollisuuden tuottaa ocsp-palvelua testikäyttöön. Palvelu ajetaan alla olevalla komennolla:

```
openssl ocsp -CA /pki/intermediate/certs/intermediate.crt -rsigner
/pki/intermediate/certs/ocsp.crt -rkey
/pki/intermediate/private/ocspkey-4096.pem -passin
file:/pki/intermediate/private/ocsppass -port 8080 -index
/pki/intermediate/index -nmin 15
```

OCSP-palveluun määritetään varmenteiden myöntäjä (CA), OCSP-palvelulle tehty varmenne ja sen yksityinen avain sekä tämän salasanatiedosto. Lisäksi määritellään portti, josta palvelua tarjotaan ja viitataan välivarmenteen indeksitiedostoon. -nmin valinta määrittää, kuinka pitkään varmennetieto on voimassa.

Varmenteiden käyttö

Laitetunnistus

Laitetunnistus 802.1X protokollalla toimii siten, että toinen laitteista toimii autentikoijana (**`authenticator`**) ja toinen anojana (**`supplicant`**). Tunnistuksen voi tehdä myös molempiin suuntiin samanaikaisesti, mutta yleensä tunnistus tehdään vain verkkoon liittyvälle laitteelle.

EAP-TLS protokolla perustuu molemminpuoliseen tunnistukseen varmenteen perusteella. Varmenteissa on listattu käyttötarkoitukset, joita varmenteella on mahdollista tehdä. Käytettäessä varmennetta laitetunnistukseen, tulee laitevarmenteella olla TLS Client Authentication käyttötarkoituksena. Tämä tulee huomioida varmennetta allekirjoittaessa.

FreeRADIUS

Ympäristössä RADIUS-palvelin on toteutettu FreeRADIUS sovelluksella. Toteutuksen kannalta olennaiset konfiguraatiotiedostot ja niiden asetukset on määritelty tässä. Esivalmisteluina palvelimelle asennetaan FreeRADIUS ja sen tarvitsemat sovellukset. Tämän jälkeen luodaan palvelimelle yksityinen avain ja allekirjoitetaan sille välivarmenteella varmenne samaan tapaan kuin OCSP-palvelimellekin. Tämän jälkeen tehdään konfiguraatio kuntoon ja käynnistetään palvelu.

/etc/freeradius/3.0/clients.conf

```
client routers {
        ipaddr = 192.168.20.0/24
        proto = *
        secret = demoradius
        nas_type = cisco
}
```

Clients tiedostolla määritellään RADIUS-palvelimen asiakkaat, eli tunnistuksen tekevät laitteet. Tiedostossa määritellään ip-osoitealue, josta kyselyt voivat tulla. Protokolla voidaan rajata vain TCP- tai UDP-liikenteeseen. Lisäksi määritellään salasana ja mahdollisesti laitetyyppi.

/etc/freeradius/3.0/mods-available/eap

```
eap {
  default_eap_type = tls
  timer_expire = 60
  ignore_unknown_eap_types = no
  cisco_accounting_username_bug = no
  max_sessions = ${max_requests}

  tls-config tls-common {
    private_key_password = radius
    private_key_file = /pki/intermediate/private/radiuskey.pem
    certificate_file = /pki/radius/radius-chain.crt
    ca_file = /pki/radius/root.crt
```

```
    auto_chain = no
    check_crl = yes

    ca_path = /pki/radius
    ca_path_reload_interval = 3600
    allow_expired_crl = no
    cipher_list = "DEFAULT"
    cipher_server_preference = no
    tls_min_version = "1.2"
    tls_max_version = "1.2"
    ecdh_curve = ""
  }

  tls {
    tls = tls-common
    configurable_client_cert = yes
    EAP-TLS-Require-Client-Cert = Yes
  }
}
```

Eap-konfiguraatiotiedostossa määritellään EAP-protokollaan tarvittavat asetukset. Tässä tapauksessa käytetään EAP-TLS protokollaa. Konfiguraatiossa määritellään polku, jonne varmenteet ja sulkulistat laitetaan. Varmenteiden osalta on huomattava, että RADIUS-palvelimen varmennetiedoston tulee sisältää myös välivarmenne, eli kyseessä on ketjuvarmenne. Varmenteiden tulee olla PEM-muodossa. Ketjuvarmenteen saa helposti luotua komennolla

```
cat /pki/radius/radius.crt /pki/radius/intermediate.crt >
/pki/radius/radius-chain.crt
```

Sulkulistoista on huomioitava, että FreeRADIUS ei osaa noutaa sulkulistoja automaattisesti, vaan ne tulee tallentaa konfiguraatiossa mainittuun ca_path kansioon yhdessä varmenteiden kanssa. Tässä tapauksessa polku on */pki/radius*. Sulkulistat tulee päivittää kansioon säännöllisin välein PEM-muodossa. Päivityksen jälkeen tulee ajaa komento

```
c_rehash /pki/radius
```

Jotta EAP:n saa käyttöön, tulee se linkittää symbolisella linkillä */etc/freeradius/3.0/mods-enabled* kansioon. Komento annetaan seuraavasti

```
ln -s /etc/freeradius/3.0/mods-available/eap /etc/freeradius/3.0/mods-enabled/eap
```

/etc/freeradius/3.0/sites-available/demo

```
server radius.demo.fi {
  listen {
    type = auth
    ipaddr = *
    port = 0
    limit {
    max_connections = 16
    lifetime = 0
    idle_timeout = 30
  }

  listen {
    ipaddr = *
    port = 0
    type = acct
  }

  authorize {
    preprocess
    auth_log
    suffix
    eap {
      ok = return
    }
    files
    expiration
    logintime
    Autz-Type New-TLS-Connection {
```

```
      ok
    }
  }

  authenticate {
    eap
  }

  preacct {
    preprocess
    acct_unique
    suffix
    files
  }

  accounting {
    detail
    -sql
    exec
    attr_filter.accounting_response
  }

  post-auth {
    verify_tls_client_common_name
    if (session-state:User-Name && reply:User-Name && request:User-Name
&& (reply:User-Name == request:User-Name)) {
      update reply {
        &User-Name !* ANY
      }
    }
    update {
      &reply: += &session-state:
    }
    -sql
    exec
    remove_reply_message_if_eap
```

```
    Post-Auth-Type REJECT {
      -sql
      attr_filter.access_reject
      eap
      remove_reply_message_if_eap
    }

    if (EAP-Key-Name && &reply:EAP-Session-Id) {
      update reply {
        &EAP-Key-Name := &reply:EAP-Session-Id
      }
    }
  }
}
```

Site-tiedosto sisältää autentikointikyselyjä kuuntelevan palvelimen asetukset. EAP-tiedoston tapaan myös site-tiedosto pitää linkittää sites-enabled-kansioon. Jos muita autentikointipalvelimia ei ole käytössä, kannattaa ensin poistaa sites-enabled kansiosta kaikki linkitykset. Samoja palvelimen portteja ei voi kuunnella useampi palvelin.

Palvelu kannattaa käynnistää taustapalveluna komennolla

```
systemctl start freeradius
systemctl enable freeradius
```

Enable komento varmistaa, että palvelu käynnistyy automaattisesti palvelimen käynnistyessä.

RADIUS-palvelinta voi testata komennolla `freeradius -X` Tällöin palvelimen loki juoksee suoraan näytöllä ja mahdolliset virheet on helppo löytää. Luonnollisesti itse palvelun tulee olla sammutettuna.

Palvelimen lokista löytää melko helposti virheet, ne tulostetaan punaisella. Esimerkiksi sulkulistalla oleva varmenne näyttää tältä

```
(8) eap_tls: (TLS) Creating attributes from client certificate
```

```
(8) eap_tls:   TLS-Client-Cert-Serial := "2031"
(8) eap_tls:   TLS-Client-Cert-Expiration := "240420093936Z"
(8) eap_tls:   TLS-Client-Cert-Valid-Since := "240406093936Z"
(8) eap_tls:   TLS-Client-Cert-Subject := "/CN=rtr4.demo.fi"
(8) eap_tls:   TLS-Client-Cert-Issuer :=
"/C=FI/ST=Finland/O=DemoInc/OU=IT/CN=DemoInc Intermediate CA RSA"
(8) eap_tls:   TLS-Client-Cert-Common-Name := "rtr4.demo.fi"
(8) eap_tls:   ERROR: (TLS) OpenSSL says error 23 : certificate revoked
(8) eap_tls: (TLS) send TLS 1.2 Alert, fatal certificate_revoked
(8) eap_tls: ERROR: (TLS) Alert write:fatal:certificate revoked
(8) eap_tls: ERROR: (TLS) Server : Error in error
(8) eap_tls: ERROR: (TLS) Failed reading from OpenSSL:
error:0A000086:SSL routines::certificate verify failed
(8) eap_tls: ERROR: (TLS) System call (I/O) error (-1)
(8) eap_tls: ERROR: (TLS) EAP Receive handshake failed during operation
(8) eap_tls: ERROR: [eaptls process] = fail
(8) eap: ERROR: Failed continuing EAP TLS (13) session.  EAP sub-module
failed
(8) eap: Sending EAP Failure (code 4) ID 9 length 4
```

Verkkolaitteen konfiguraatio

Ciscon laite voidaan konfiguroida anojaksi, jos laite tukee 802.1X protokollaa. Anojalle laitteelle konfiguroidaan EAP profiili ja 802.1X profiili, jotka otetaan käyttöön portissa, jolla verkkoon liitytään. Autentikoijana toimivan laitteen konfiguraatioon määritetään RADIUS-palvelin ja autentikoivan portin asetukset.

Ensin otetaan yleisesti 802.1X käyttöön komennolla sekä anojassa että autentikoijassa.

```
dot1x system-auth-control
```

Anoja (Supplicant)

Sen jälkeen anojalle luodaan profiilit

```
eap profile eap_profile
```

```
 method tls
 pki-trustpoint intermediate
!
dot1x credentials cert_credentials
 username rtr3.demo.fi
 pki-trustpoint intermediate
```

Ja lopuksi profiilit otetaan käyttöön siinä portissa, jolla autentikoitavaan verkkoon halutaan liittyä.

```
interface GigabitEthernet2
 dot1x pae supplicant
 dot1x credentials cert_credentials
 dot1x supplicant eap profile eap_profile
```

Autentikoija (Authenticator)

Autentikoijalle taas määritellään käytettävät RADIUS-palvelimet

```
radius server demo_radius
 address ipv4 192.168.20.7 auth-port 1812 acct-port 1813
 key demoradius
!
ip radius source-interface GigabitEthernet1
!
aaa authentication dot1x default group radius
aaa authorization network default group radius
```

Tämän jälkeen määritellään portin turvallisuusasetukset

```
interface GigabitEthernet2
 authentication priority dot1x
 authentication port-control auto
 authentication periodic
 authentication timer reauthenticate 180
 authentication violation protect
 dot1x pae authenticator
```

```
dot1x timeout quiet-period 10
dot1x timeout server-timeout 30
```

Porttimäärityksissä määritetään 802.1X pae rooli, joka voi olla **authenticator**, **supplicant** tai **both**. Both roolissa molemmat päät varmistavat toisensa ennen yhteyden avaamista. Tämä edellyttää, että molemmilla on yhteys RADIUS-palvelimeen.

authentication timer reauthenticate on erityisen tärkeä komento. Se pakottaa laitteen autentikoimaan uudestaan tietyin välein. Tämä varmistaa, että jos laite päätyy sulkulistalle, sen liikennöinti myös loppuu joskus.

Myö **authentication violation** on tärkeä komento. Se määrittää toimet, joita epäonnistuneen autentikoinnin seurauksena tehdään. Vaihtoehdot ovat **shutdown**, jolla portti suljetaan, **restrict**, joka tekee syslogiin virheilmoituksen, **protect** pitää portin ylhäällä mutta pudottaa kaiken liikenteen ja **replace** katkaisee autentikointisession ja yrittää uudestaan.

Autentikoinnin tilaa voi tarkastella Ciscon laitteelta komennolla

```
rtr3#show authentication sessions
Interface  MAC Address  Method  Domain  Status Fg  Session ID
---------------------------------------------------------------------------
Gi2  0ca8.5309.0001  dot1x  DATA     Auth      1F14A8C000000017B45EC648
```

Tulosteesta näkee, että portissa Gi2 oleva laite on autentikoitu RADIUS-palvelimen kautta. Autentikoimaton laite jää UnAuth tilaan, ja jos yhteys palvelimelle puuttuu, jää tilaksi Unknown. Lokitiedoissa autentikointi näkyy seuraavasti:

```
*Apr  6 17:09:37.709: %DOT1X-5-SUCCESS: R0/0: sessmgrd: Authentication
successful for client (0c36.08d8.0001) on Interface Gi2 AuditSessionID
2014A8C00000001BB45ED951
*Apr  6 17:09:37.728: %SESSION_MGR-5-SUCCESS: R0/0: sessmgrd:
Authorization succeeded for client (0c36.08d8.0001) on Interface
GigabitEthernet2 AuditSessionID 2014A8C00000001BB45ED951
```

Lokitietoihin autentikoinnit tulevat vain, jos authentication logging verbose konfiguraatio on annettu.

Liikenteen salaus

Varmenteilla voidaan parantaa IPsec-salauksen turvallisuutta suorittamalla salauksen IKEv2-neuvottelun autentikointi varmennepohjaisesti. Seuraavassa kuvataan Ciscon laitteille tarvittavat konfiguraatiot.

Verkkolaitteen konfiguraatio

Liikenteen salaukseen IPsec-protokollalla on useita keinoja. Tässä on esitelty konfiguraatio, joka luo IPsec-salatun GRE-tunnelin laitteesta toiseen.

Ensin luodaan IP-yhteys laitteiden välille. Yhteys voi olla suora tai reititetty jonkin verkon läpi, kunhan paketit laitteiden välillä kulkevat. Tässä tapauksessa laitteiden välillä on suora yhteys ja niille annetaan osoitteet verkosta 10.1.0.0/24

```
! Reititin rtr1
rtr1(config)# interface GigabitEthernet2
rtr1(config-if)# description rtr2
rtr1(config-if)# ip add 10.1.0.1 255.255.255.0
rtr1(config-if)# no shutdown
! Reititin rtr2
rtr2(config)# interface GigabitEthernet2
rtr2(config-if)# description rtr1
rtr2(config-if)# ip add 10.1.0.2 255.255.255.0
rtr2(config-if)# no shutdown
rtr2# ping 10.1.0.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.0.1, timeout is 2 seconds:
.!!!!
```

Nyt molemmilla laitteilla on ip-osoite Gi2 portissa, ja paketit kulkevat, kuten pingistä nähdään. Seuraavaksi luodaan perusteet IKEv2 neuvottelulle, tässä rtr1 laitteelle.

```
crypto ikev2 proposal ike_proposal
 encryption aes-gcm-256
 prf sha512
 group 20
```

```
!
crypto ikev2 policy ike_policy
 match address local 10.1.0.1
 proposal ike_proposal
!
crypto ikev2 disconnect-revoked-peers
```

IKEv2 ehdotus (proposal) antaa laitteelle ehdotuksen käytettävistä algoritmeista IKEv2-neuvottelua varten. Ehdotukseen määritellään salaus- tai allekirjoitusalgoritmi, tai molemmat. GCM-salausta käyttäessä allekirjoitusalgoritmiä ei käytetä. PRF, eli Pseudo-Random Function generoi satunnaista dataa liikenteen salausavaimeen, ja DH group määrittää avainvaihtoalgoritmin vahvuuden. `disconnect-revoked-peers` pitää huolen siitä, että sulkulistalle joutuneet laitteet katkaistaan salaustunnelista.

IKEv2-politiikka määrittää kussakin tilanteessa käytettävän ehdotuksen. `match` komennolla politiikka sidotaan paikalliseen ip-osoitteeseen (Gi2).

```
crypto pki certificate map demo_cert_map 10
 subject-name co demo.fi
```

Varmennekarttaan listataan ne ominaisuudet, joita vastapään varmenteella tulee olla, jotta se voidaan hyväksyä IKEv2-profiilissa vastapään laitteeksi. Tämä ei vielä tarkista varmennetta mitenkään, ainoastaan ohjaa sen oikeaan profiiliin.

```
crypto ikev2 profile ike_profile
 match address local interface GigabitEthernet2
 match certificate demo_cert_map
 identity local dn
 authentication remote rsa-sig
 authentication local rsa-sig
 pki trustpoint intermediate
 lifetime 600
 dpd 10 10 periodic
 reconnect timeout 600
```

IKEv2-profiililla määritellään IKE-neuvottelun autentikointimenetelmä, tässä tapauksessa RSA-allekirjoitus. Varmenteita käytettäessä määritetään myös allekirjoitusvarmenne, jota

vasten vastapää varmennetaan. Lisäksi määritellään muita parametrejä, kuten oman laitteen identiteetti DN, joka tulee käytettävästä varmenteesta.

DPD eli Dead Peer Detection on protokolla, joka testaa vastapään saavutettavuutta tasaisin välein. `lifetime` määrittää IKEv2 SA:lle elinajan, jonka jälkeen se neuvotellaan uudestaan. `reconnect timeout` kertoo, kuinka pian yhteyden katkeamisen jälkeen yritetään uudestaan.

```
crypto ipsec transform-set ipsec_ts esp-aes 256
 mode transport require
!
crypto ipsec profile ipsec_profile
 set security-association lifetime kilobytes 102400
 set transform-set ipsec_ts
 set pfs group20
 set ikev2-profile ike_profile
```

IPsec-profiilissa määritellään uudelleen avaintamisen väli sekä käytettävä salausalgoritmi `transform-set` asetuksella. Lisäksi määritellään IKEv2-profiili ja PFS-avainsalauksen algoritmi. PFS eli Perfect Forward Secrecy varmistaa, että samaa ipsec-avainta ei käytetä uudestaan, vaan se generoidaan joka kerta uudestaan.

```
interface Tunnel1
 ip address 20.1.0.1 255.255.255.0
 tunnel source GigabitEthernet2
 tunnel destination 10.1.0.2
 tunnel protection ipsec profile ipsec_profile
```

Tunneliasetukset ovat minimiasetukset, joita salatun tunnelin tekemiseen tarvitaan. Tunnelin sisäinen ip-osoite on 20.1.0.1/24. Tunneli lähtee Gi2 portista ja menee osoitteeseen 10.1.0.2 (rtr2). Tunnel protection komennolla otetaan käyttöön ipsec-salaus.

Helpoiten salauksen tilanteen voi tarkastaa komennolla

```
rtr1# show crypto session
Crypto session current status

Interface: Tunnel1
```

```
Profile: ike_profile
Session status: UP-ACTIVE
Peer: 10.1.0.2 port 500
  Session ID: 16
  IKEv2 SA: local 10.1.0.1/500 remote 10.1.0.2/500 Active
  IPSEC FLOW: permit 47 host 10.1.0.1 host 10.1.0.2
        Active SAs: 2, origin: crypto map
```

Halutessa yksityiskohtaisempaa tilannetietoa voi käyttää komentoja

```
show interfaces Tunnel1
show crypto ikev2 sa
show crypto ipsec sa
```

Ja salauksen neuvottelun voi pakottaa uudestaan komennolla

```
clear crypto session
```