# Ohjeet oman Zero Trust -ympäristön rakentamiseen – mallina Software-Defined Perimeter (SDP)

Sisällys

[Ohjeet oman Zero Trust -ympäristön rakentamiseen – mallina Software-Defined Perimeter (SDP) 1](#_Toc123681517)

[Ohjeen kohderyhmä 2](#_Toc123681518)

[TL;DR 2](#_Toc123681519)

[Lisenssitiedot (sivulta https://github.com/WaverleyLabs; ohje ei sisällä uutta lähdekoodia) 3](#_Toc123681520)

[Esittely 3](#_Toc123681521)

[Miten SDP toimii? 4](#_Toc123681522)

[Käytössä tarvittavat teknologiat 5](#_Toc123681523)

[1. git ja GitHub 5](#_Toc123681524)

[2. node.js ja npm 5](#_Toc123681525)

[3. MySQL-tietokanta 5](#_Toc123681526)

[4. OpenSSL 5](#_Toc123681527)

[5. iptables 6](#_Toc123681528)

[6. conntrack 6](#_Toc123681529)

[7. SSH 7](#_Toc123681530)

[8. Linux 7](#_Toc123681531)

[Asennusympäristöstä huomioitavaa 7](#_Toc123681532)

[Ohjeen seuraaminen 8](#_Toc123681533)

[Controller-moduulin asentaminen 8](#_Toc123681534)

[1. Node 9](#_Toc123681535)

[2. NPM 9](#_Toc123681536)

[3. Git-projektin kloonaus 9](#_Toc123681537)

[4. Node-pakettien asennus 10](#_Toc123681538)

[5. MySQL 10](#_Toc123681539)

[6. Esimerkkitietokannan tuominen 10](#_Toc123681540)

[7. Luodaan uusi käyttäjä tietokantaan 12](#_Toc123681541)

[8. Valmiit tiedot taulukoihin 13](#_Toc123681542)

[9. Config.js-tiedoston asetukset 15](#_Toc123681543)

[10. OpenSSL-asennus 15](#_Toc123681544)

[11. OpenSSL-varmenteet ja PKI-infrastruktuurin rakentaminen 15](#_Toc123681545)

[12. Palvelinvarmenteiden ja -avainten luonti 16](#_Toc123681546)

[13. Controllerin käynnistys 17](#_Toc123681547)

[Gateway-moduulin asentaminen 18](#_Toc123681548)

[1. Linuxin palomuurin asetukset: iptables 19](#_Toc123681549)

[2. Connection tracking ja iptables 19](#_Toc123681550)

[3. fwknop: taustakirjastojen asentaminen 20](#_Toc123681551)

[4. Projektin kloonaus 20](#_Toc123681552)

[5. Mennään oikeaan hakemistoon 20](#_Toc123681553)

[6. Kirjoitetaan komennot 21](#_Toc123681554)

[7. Konfiguroidaan projekti kääntämistä varten 21](#_Toc123681555)

[8. Kääntäminen 21](#_Toc123681556)

[9. Asentaminen 21](#_Toc123681557)

[10. Tiedostojen editointi 22](#_Toc123681558)

[11. Gatewayn käynnistys 24](#_Toc123681559)

[Client-moduulin asentaminen 24](#_Toc123681560)

[12. Client-moduuli 24](#_Toc123681561)

[Lopuksi 24](#_Toc123681562)

## Ohjeen kohderyhmä

Tämä suomenkielinen ohje on suunnattu erityisesti ICT- tai kyberturvallisuuden opiskelijoille tai junior-tason ammattilaisille. Se on suunnattu lukijalle, joka haluaa harjoitella oman tietoturvallisen Zero Trust -ympäristönsä rakentamista. Nollaluottamus- eli Zero Trust -mallien käytännön osaamisesta on monenlaista hyötyä tietoturvasta kiinnostuneille, ja tässä ohjeessa ICT- ja turvaosaamistaan voi kehittää avoimeen lähdekoodiin perustuvan Software-Defined Perimeter -toteutuksen avulla.

## TL;DR

**Jos sinulla ei ole mahdollisuutta mihinkään muuhun, skrollaa ohjeen loppuun ja anna minulle palautetta siellä näkyvien muutaman kysymyksen pohjalta sähköpostiosoitteeseeni sami.kivela@myy.haaga-helia.fi. Lyhytkin palaute on äärimmäisen arvokas! Mutta uskon ohjeesta olevan sinulle myös hyötyä. > POISTETAAN, job done.**

## Lisenssitiedot (sivulta <https://github.com/WaverleyLabs>; ohje ei sisällä uutta lähdekoodia)

Copyright 2016 Waverley Labs, LLC

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see http://www.gnu.org/licenses/.

The fwknop project is released as open source software under the terms of the GNU General Public License (GPL v2).

## 

## Johdanto

Nollaluottamus eli Zero Trust on tietoturvallisten järjestelmien rakentamisen kulmakiviä. Sillä pyritään todentamaan aiempaa tarkemmin kaikkea verkkoliikennettä ja hallitsemaan pääsyä tietoihin dynaamisesti määriteltävän käyttöpolitiikan keinoin.

Software-Defined Perimeter (SDP) on yksi Yhdysvaltain standardisointi- ja teknologiainstituutin (NIST) suosittelemista Zero Trust -toteutustavoista. ja se soveltuu sekä eri pilviratkaisuihin että erillisympäristöihin. Nopea verkkohaku voi todentaa sen, että moni huomattava ICT- ja tietoturvaratkaisujen toimittaja on rakentanut teknologisesti SDP:n kaltaisia ratkaisuja, joista monet elävät vain toisilla nimillä. Tämä yksityishenkilöille suunnattu toteutus tapahtuu mahdollisimman yksinkertaisessa muutaman virtuaalikoneen Ubuntu Linux -ympäristössä.

Software-Defined Perimeter (SDP) eli ns. ”black cloud” -malli kehitettiin julkisen ja yksityisen sektorin yhteistyönä 2000-luvun lopulla. Waverley Labs -niminen riskienhallintayritys oli yksi alan pioneereista, ja heidän GitHub-sivuillaan oleva SDP-toteutus auttaa ymmärtämään niin Zero Trust -teknologiaa kuin muitakin turvallisen tietojenkäsittelyn perusteita (tietokannat, palomuurit, tunnistautumisen ja käyttölupien hallinta, versionhallinta jne.).

Tämä ohje syventää Waverley Labsin GitHub-sivuilla olevaa avoimen lähdekoodin Software-Defined Perimeter (SDP) -asennus- ja konfigurointiohjetta (versio 0.2, heinäkuu 2020): <https://github.com/WaverleyLabs/fwknop/blob/master/Waverley%20Labs%20OpenSDP%20Installation%20and%20Configuration.pdf>

Ohjeen mukaisesti asennettava SDP-toteutus sisältää kolme moduulia: Controller, Gateway ja Client. Nämä ovat SDP-toteutuksen vähimmäisvaatimukset, ja niiden välinen arkkitehtuuri voidaan rakentaa eri tavoin (arkkitehtuurista lisää Cloud Security Alliancen SDP Working Groupin sivuilla: <https://cloudsecurityalliance.org/research/working-groups/software-defined-perimeter>).

Tässä ohjeessa on luotu kolme itsenäistä Ubuntu Linux -virtuaalikonetta, joista ensimmäinen toimii Controller-koneena, toinen Gateway-koneena ja kolmas Client-koneena. Ohje on tehty ammattikorkeakoulun toiminnallisena opinnäytetyönä. Se kuvaa myös tekijän omaa asennusprosessia niin pitkälle kuin projektin aikana oli mahdollista edetä. Viimeistä Client-konetta ohje käsittelee vain pintapuolisesti.

## SDP:n toimintaperiaate

Controller on SDP:n toiminnan ”aivot”. Siihen rakennetaan tietokanta, joka sisältää konetunnukset, suojattavat palvelut ja yhdyskäytävät, jotka suojelevat tietyn organisaation arvokasta tietoa. Controller-koneelle annettujen tietojen pohjalta tehdään palomuurin tarkkaan hallintaan perustuvat päätökset siitä, mikä asiakaskone saa ottaa yhteyden mihinkin tarvitsemaansa yksittäiseen palveluun. Tällainen voi olla esim. SSH, HTTPS tai FTP, mutta laajemmin lähes mikä tahansa muu palvelu.

SDP-toteutuksen toiminta perustuu 2000-luvun alussa kehitettyihin Linux-palomuurin käsittelytapoihin. Palomuuri suodattaa liikennettä sisään ja ulos sille määriteltyjen sääntöjen mukaan. Monen kyberhyökkäyksen suunnittelu on alkanut avointen porttien skannauksella nmap-työkalulla. Jos kaikki portit ovat kiinni, hyökkäyksen toteutus esimerkiksi Metasploit-työkalua hyödyntäen on huomattavasti vaikeampaa. Cloud Security Alliance järjesti vuosia sitten murtautumishaasteen, jossa moni taitava eettinen hakkeri ei päässyt murtautumaan SDP:llä suojattuun kohdeympäristöön.

Yritysympäristössä koneiden luvallinen etähallinta vaikeutuu kohtuuttomasti, jos esimerkiksi SSH-yhteyttä ei voi ottaa palomuurin suljettua kaikki portit. Tämän estämiseksi on kehitetty port knocking -toimintatapa, jossa yhteyksiä voidaan avata täysin suljetunkin palomuurin läpi ”kolkuttamalla” portteja tietyssä järjestyksessä. Tämä koputtelu avaa palomuuriin oven ja mahdollistaa vaikkapa SSH-hallintayhteyden autentikoinnin niin, että heti varmennetun yhteyden avaamisen jälkeen koko palomuuri sulkeutuu taas.

Tietoturvatutkija Michael Rash kirjoitti 2000-luvun alussa port knocking -teknologiaan perustuvan fwknop-työkalun (Firewall Knock Operator). Se käyttää Linuxin käyttäjätilan puolella (user space) netfilter-palomuurin hallintaan tarkoitettua iptables-työkalua, tarkemmin tämän lokitietoja, päästääkseen vain toivotut ja oikein autentikoidut yhteydenotot palomuurin läpi. Lisäksi fwknop osaa tunnistaa yhteyttä pyytävän käyttöjärjestelmäversion.

Fwknop kehittyi sittemmin edelleen Single-Packet Authorization -protokollaksi (SPA). Nimensä mukaisesti se antaa kohdepalveluun pääsyoikeudet vain yhdelle tiedonsiirtopaketille kerrallaan. SPA enkoodaa autentikaatiotiedon sovelluskerroksen dataan, tarkemmin lähetettävään hyötykuormaan (packet payload), kun taas aiempi fwknop tallensi tiedot otsikkotietoihin (packet header). Tässä esiteltävä Software-Defined Perimeter -toteutus perustuu SPA-protokollaan.

Mainituista teknologioista on useita versiota. Jo vuonna 2008 Rash arvioi löytyvän 25 port knocking -implementaatiota ja viisi SPA-versiota. SDP-toteutuksia on Zero Trust -ajattelun valtavirtaistumisen myötä hyvin paljon, ja nyt esiteltävä open source -ohje esittelee vain yhden, perusmuotoisen järjestelmän rakentamisen.

Tämä ohje perustuu erityisesti kahteen lähteeseen:

* Waverley Labsin GitHubissa julkaisemaan asennusohjeeseen Open Source Software Defined Perimeter Installation and Configuration, Version 0.2, July 2020
* Michael Rashin kotisivuillaan julkaisemaan ohjeeseen Single Packet Authorization - A Comprehensive Guide to Strong Service Concealment with fwknop, updated January 2016

Näistä edellinen on tiivis step by step -asennusohje, jonka pohjalta SDP-asennuksen pitäisi olla mahdollinen. Tämä ohje seuraa alkuperäisen ohjeen numerointia.

Käytännössä matkan varrella voi tulla haasteita, joita helpottamaan ja tiedon jakamisen edistämiseksi tämä ohje on kirjoitettu. Waverley Labsin toteutus on julkaistu GNU General Public -lisenssillä, jonka puitteissa sitä saa jakaa edelleen ja muokata. Ohjelman toimivuudesta ei lisenssin puitteissa anneta erikseen takuuta.

Jälkimmäinen eli Rashin ohje on pidempi ja avaa lukijalle erityisesti Gateway- ja Client-koneisiin vaikuttavia taustateknologioita syvällisemmin. Ensimmäistä kertaa SDP-asennusta tekevän olisi hyvä ymmärtää mahdollisimman hyvin molempia ohjeita, mutta niihin perehtyminen voi viedä aikaa.

Käsillä oleva ohje pyrkii sujuvoittamaan ja nopeuttamaan asennusta tai ainakin sen harjoittelemista. Ohjeessa on pyritty selventämään mahdollisia vastaan tulevia ongelmakohtia, mutta sen noudattaminen ei itsessään takaa SDP-järjestelmän toimivuutta. Kuulen erittäin mielelläni palautettasi: Saitko järjestelmän toimimaan? Mitä haasteita matkan varrella vielä kohtasit? Miten ne ratkaisit?

Asennuksessa tarvittavia teknologioita

Käytännössä seuraavia taustateknologioita tarvitaan asennuksessa.

### git ja GitHub

Parissa asennuksen kohdassa siirretään GitHubissa oleva materiaali omalle koneelle. Itse käytin tähän mahdollisimman yksinkertaista mallia. Olettaen, että git on jo asennettu koneellesi, Linuxin komentorivillä tarvitsee kirjoittaa SDP-moduulien kohdehakemistossa:  
  
git clone [Code-kohdassa annettu https-osoite; ensin Controller-hakemistossa ja sitten fwknop-hakemistossa]

### node.js ja npm

Controller-moduuli on kirjoitettu JavaScriptillä ja sen ajamiseen tarvitaan node.js-ajoympäristö sekä node package manager (npm). Näiden asennuksen pitäisi sujua ongelmitta ohjetta noudattaen.

### MySQL-tietokanta

Controller-koneelle luodaan MySQL-tietokanta, jonka tauluihin täytetään SDP-moduulien olennaisia tietoja. Riippuen taustastasi tämä voi olla helppoa tai hankalampaa. Itse käytin tässä vaiheessa melko paljon aikaa, joten työtäsi helpottaakseni olen luonut mallikomentoja, joilla saat varsin nopeasti täytettyä taulut. Toki voit tehdä omatkin taulusi.

### OpenSSL

SDP-projektin avulla saa väistämättä harjoitella myös kryptografiaa. Projektissa käytetään avoimen lähdekoodin OpenSSL-kirjastoa, jonka avulla hallitaan usein esimerkiksi web-selainten suojattua TLS-liikennettä.

Salaus voi olla symmetrinen tai asymmetrinen riippuen siitä, käyttävätkö lähettäjä ja vastaanottaja samaa vai eri salausavainta. Tässä projektissa käytetään asymmetristä eli julkisen avaimen salausta. Controller-koneen asennuksen yhteydessä luodaan oma juurivarmenne, jonka pohjalta tarjotaan käyttöön tuleva varmenne ja allekirjoitetaan eri laitteiden käyttöön tulevat palvelinvarmenteet. Tässä käytetään komentorivillä OpenSSL:ää, hyvin monipuolista avoimen lähdekoodin kryptografista alustaa, jonka tunteminen on järjestelmän ylläpitäjille ja tietoturvan toteuttajille hyödyllistä.

Oma OpenSSL-osaamiseni ei ollut projektin aikana kovin vahvaa, joten törmäsin varsin kiusallisiin ongelmiin. Alusta myös kehittyy koko ajan, eikä sen dokumentaatio ole aina kovin selkeää. Pyrin asennusohjeessa käsittelemään näitä haasteita ja toivon, että saat oman autentikointisi ja sähköisen allekirjoituksesi toimimaan kunnialla.

### iptables

Linux-käyttöjärjestelmän ytimessä sijaitseva tehokas pakettisuodatinjärjestelmä on nimeltään netfilter. Tätä kernel-pohjaista suodatusta on käyttäjätilassa (user space) perinteisesti hallinnoitu iptables-käyttöliittymällä, jota tässäkin ohjeessa käytetään. Netfilter-projektin kehittäjät pyrkivät tosin saamaan käyttäjäkunnan siirtymään edistyneempään nftables-käyttöliittymään, joka on kuitenkin jätetty tämän ohjeen ulkopuolelle, koska alkuperäinen ohje käytti iptablesia.

Iptables-käyttöliittymän hallintalogiikka vaatii oman harjoittelunsa. Jos käyttö ei ole sinulle ennestään tuttua, yksi varoituksen sana tässä asennuksessa: Jos teet asetuksia SSH-etäyhteyden kautta, seuraamalla orjallisesti Waverley Labsin ohjetta voit vahingossa sulkea itsesi koko koneen ulkopuolelle (myönnän – tämä tapahtui aluksi minulle). Ohjeen kronologia voi tässä viedä harhaan, sillä ensimmäisessä kohdassa ohjeistetaan luomaan kaiken liikenteen estävä palomuurisääntö. Jos et tätä ennen ole varmistanut pääsyä sisään koneelle, tämä sääntö blokkaa sinut ulos. Uusi yhteys on mahdollista luoda esimerkiksi konsoliyhteydellä tai asentamalla käyttöjärjestelmä/virtuaalikone uudelleen.

Iptables-konfiguroinnissa voin suositella käyttämään Michael Rashin kirjoittamaa skriptiä, johon palataan tarkemmin alla. Sanottakoon, että fwknop-asennus luo iptables-tauluja vastaavat uudet taulut, joiden tila voidaan tarkistaa syslog-viestien avulla. Rashin SPA/fwknop-tutoriaali näyttää esimerkkejä tästä.

### conntrack

Toinen iptablesin hallintaan liittyvä tärkeä sääntö liittyy käsitteeseen connection tracking (conntrack). Conntrack pitää käytännössä yllä kernelissä olevia tilatietoja käynnissä olevista yhteyksistä. Se on user spacen puolella oleva työkalu, joka tulee olla käytössä iptables-sääntöjä tehdessä. Käytännössä juuri conntrack syöttää fwknopin palomuurihallinnalle tiedon siitä, onko autentikoitu yhteys kerran luotu (ESTABLISHED), ja tämän kernel-tason tiedon perusteella palomuuri osaa jatkaa juuri tämän yhteyden sallimista sulkien kaikki muut käyttäjätilan tasolta tulevat yhteysyritykset (esim. uusi autentikoimaton SSH-yhteys tai nmap-skannausyritys).

Connection trackingin toiminnan varmistaminen voi olla työlästä. Valmiissa SDP-ohjeessa mielestäni toimitaan hieman takaperoisesti, sillä sitä orjallisesti seuraten conntrackin pitäisi olla toiminnassa jo ennen asennusta. Itse käytin mainittua Michael Rashin asennusskriptiä, joka huolehtii myös conntrack-asetukset kuntoon. Skriptin tulkinta on hyödyllinen oppimiskokemus itsessään.

### SSH

Secure Shell (SSH) -yhteyttä käytetään koneiden väliseen turvalliseen etäyhteyteen. SSH:n käyttö liittyy teemoiltaan OpenSSL:ään: luodaan yksityinen ja julkinen avain, pidetään yksityinen avain vain omana tietona ja toimitetaan julkinen avain autentikointia varten kohdekoneelle.

Kohdekoneella on sekä ssh client että ssh server eli sshd-ohjelma (daemon), joka aktiivisesti kuuntelee yhteydenottoja. Tarvittaessa asennus voidaan tehdä:

sudo apt install openssh-server

sudo systemctl enable ssh

Avainten luonnissa voi käyttää eri algoritmeja. Ohjeessa käytössä on RSA-algoritmi, jonka jälkeen suositellummaksi tavaksi on tullut elliptisen käyrän kryptografiaa (ECC) hyödyntävä Ed25519-algoritmi. SSH-avaimen luonti sillä tapahtuu näin:

ssh-keygen -t ed25519

### 8. Linux

Tämän ohjeen seuraamiseksi on hyvä olla jonkinlainen peruskokemus Linux- komentojen suorittamisesta, sillä valtaosa toiminnasta tapahtuu Linuxin komentorivillä. Tässä ohjeessa ei oteta kantaa eri Linux-jakeluiden toiminnallisuuksiin tai paketinhallintaratkaisuihin.

## Asennusympäristöstä huomioitavaa

Tämä ohje on toteutettu Linux Ubuntu Desktop -virtuaalikoneympäristössä. Ohjetta varten oli olemassa kaksi ympäristöä, joita yhdisti yhteinen Client-kone:

* Windows 11 -isäntäkoneessa VirtualBox-alustalla Ubuntu-virtuaalikone nimeltä SDP Client, joka simuloi mitä tahansa etäympäristöstä SDP-yhdyskäytävän suojaamaan ympäristöön yhteyttä ottavaa asiakasta. Tämä SDP Client otti SSH-yhteyden internetin yli rakennettuun etäympäristöön, jossa oli palomuurin takana useampi kone: SDP Gateway, Controller, Client ja lisäksi Gatewayn suojaama ns. VIP Asset -kone, joka edusti organisaation arvo-omaisuutta.
* Edellisen VirtualBox-ympäristön sisäinen SDP-ympäristö, jossa oli Client-koneen lisäksi Gateway- ja Controller-koneet. Näiden avulla pystyi kokeilemaan toteutusta ilman verkon yli menevää SSH-etäyhteyttä.

SDP-toteutuksen koodi on testattu vain \*nix-tyyppisissä järjestelmissä. Ohjeen oletuksena on Linux Ubuntu -ympäristö.

## Ohjeen seuraaminen

Pidä saatavilla alkuperäinen ohje:

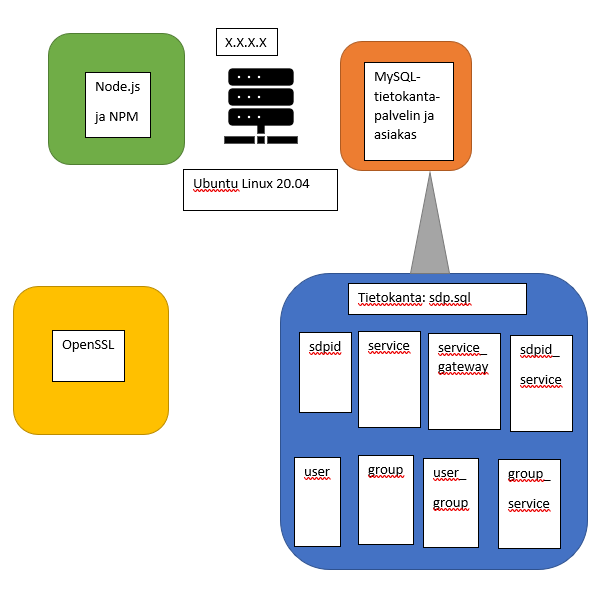
<https://github.com/WaverleyLabs/fwknop/blob/master/Waverley%20Labs%20OpenSDP%20Installation%20and%20Configuration.pdf>

Ohjeesta on myös toinen versio osoitteessa <https://github.com/WaverleyLabs/SDPcontroller>. Tässä erona on nähdäkseni vain lyhennetty OpenSSL-komento Certificate Authority -varmenteen luomisessa (tähän palataan kohdassa 11).

Kannattaa pitää lähellä myös Michael Rashin fwknop-ohje <http://www.cipherdyne.org/fwknop/docs/>. Sitä ei tarvita juuri alussa Controllerin asennuksessa mutta myöhemmin kyllä.

## Controller-moduulin asentaminen

Controller koostuu muutamasta peruskomponentista, jotka on kuvattu alle. Asennuskoneet kannattaa muutoin pitää mahdollisimman puhtaina erillisistä ohjelmista – peruspäivitykset toki on syytä tehdä.



### Node

Aluksi kannattaa tehdä tavalliset Ubuntun pakettitilanteen päivitykset (tässä käyttäen [listaoperaattoria &&,](https://www.gnu.org/savannah-checkouts/gnu/bash/manual/bash.html#Lists) mutta toki voi kirjoittaa erikseenkin):

sudo apt update && sudo apt upgrade

Asennus lähtee liikkeelle Node.js:n asennuksella.

sudo apt install nodejs

Itselläni tämä ei aiheuttanut ongelmia puhtaassa Ubuntu-koneessa. Tarkistetaan Noden versio ja varmistetaan samalla asennus:

node -v

### NPM

Sitten kannattaa tarkistaa, onko Node.js packace manager asentunut:

npm

Tässä tapauksessa ei, joten asennan:

sudo apt install npm

Tarkistan komennolla npm, että on asentunut ja versio näkyy. Huom! Snapin kautta asentaessa voi tulla eroavaisuuksia, joten itse suosittelen asentamaan apt:n avulla.

### Git-projektin kloonaus

Lataa tai kloonaa SDP Controller GitHubista. Käytän itse tähän kloonausta. Tarkista, että Git toimii. Sitten tee Controller-koneelle Git-repositorio haluamastasi hakemistosta käsin, joka minulla on /home/samik/:

sudo git clone <https://github.com/WaverleyLabs/SDPcontroller.git>

Tämä luo SDPcontroller-alihakemiston ja Gitin versionhallinnan.

### Node-pakettien asennus

Seuraavaksi siirrytään cd-komennolla tähän hakemistoon home/samik/SDPcontroller, jossa asennetaan node-packaget:

npm install

### MySQL

Seuraavaksi asennetaan MySQL-palvelin. Ollaan edelleen samassa hakemistossa SDPcontroller:

sudo apt install mysql-server

(Setting up mysql-server-8.0 (8.0.31-0ubuntu0.22.04.1)

Voit tarkistaa, että server on päällä:

sudo systemctl status mysql.service

tai

sudo service mysql status

Toivottavasti MySQL-palvelin on ongelmitta päällä. Seuraavat vaiheet voivat tapahtua hieman eri tavoin riippuen järjestelmästä ja toimintatavasta. Tässä vaiheessa alkuperäinen ohje antaa niukat tiedot, joita nyt laajennetaan. Lisäohjeita voit katsoa mm. täältä:

<https://ubuntu.com/server/docs/databases-mysql>

<https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04>

<https://www.digitalocean.com/community/tutorials/how-to-import-and-export-databases-in-mysql-or-mariadb>

### Esimerkkitietokannan tuominen

Ennen sample-kannan hakua tulee tehdä tietokantaan turva-asetukset. Avataan MySQL-kehote:

sudo mysql

Voimme nähdä kaikki tietokannat (MySQL-komentojen jälkeen tulee aina puolipiste):

SHOW DATABASES;

Katsotaan MySQL-nimisestä tietokannasta tiedot käyttäjistä ja tunnistautumistavoista:

SELECT user, plugin FROM mysql.user;

Tein tämän muutoksen:  
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'password';

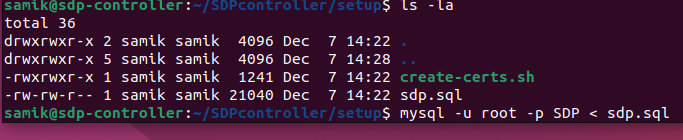
Tässä pääsyoikeudet saaneen root-käyttäjän tunnistautumistavaksi määritellään MySQL:n natiivisalasana, joka tässä tapauksessa on ’password’.

MySQL-kehotteessa voimme sanoa ’exit’. Seuraavaksi suojaamme tietokannan:

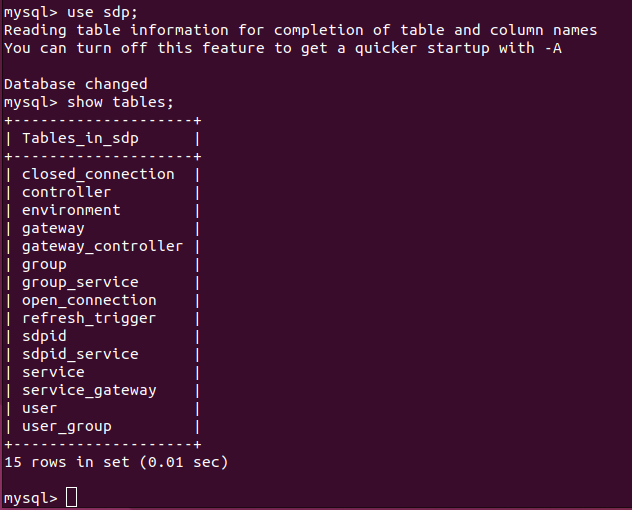
sudo mysql\_secure\_installation

Aluksi määritellään rootille uusi salasana. Seuraaviin turvakovennuksiin voi vastata kyllä eli Y.

Tämän jälkeen loin root-käyttäjänä kantaan esimerkkitietokannan tällaisella komennolla:

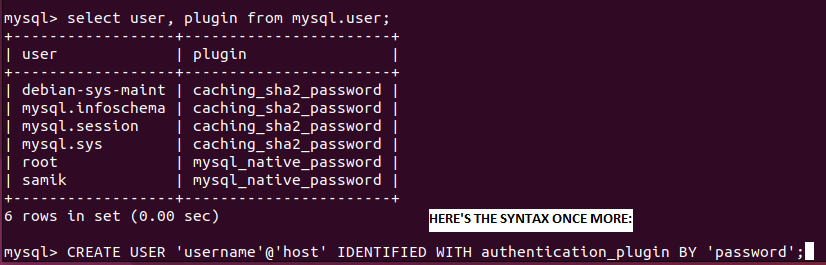


Menemällä takaisin MySQL-kehotteeseen voimme katsoa, mitä tauluja tuodussa kannassa on:



### Luodaan uusi käyttäjä tietokantaan

Tavoitteemme on luoda rootin lisäksi toinen käyttäjä käyttämään sdp.sql-tietokantaa. Se tapahtuu CREATE USER -komennolla, itse luon käyttäjän ’samik’. Seuraavassa tämän käyttäjän kirjautumistavaksi laitetaan authentication\_plugin (katso aiemmista MySQL-ohjeista lisätietoja):

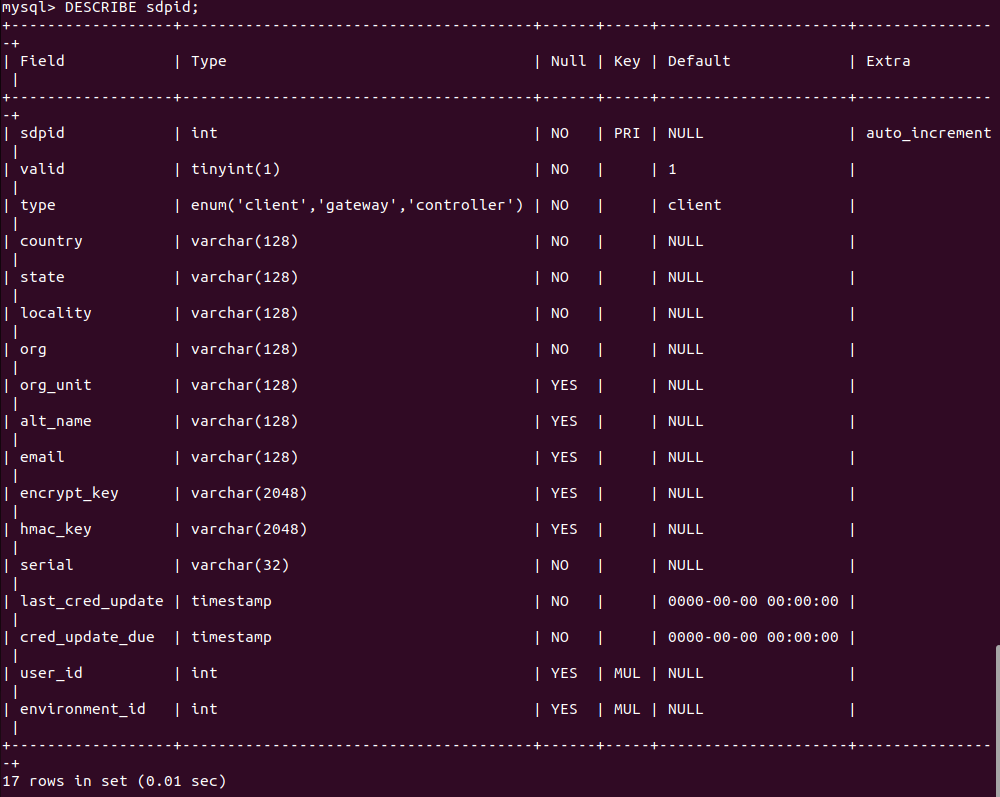


Uuden käyttäjän pitäisi nähdä hieman eri tietokannat kuin root-käyttäjän:

Kuva, joka sisältää kohteen teksti

Kuvaus luotu automaattisesti

Komennolla DESCRIBE voi nähdä taulujen kenttien vaadittavia tyyppejä ja esimerkiksi olennaisen Primary Keyn, kuten seuraavasta taulusta sdpid:



### Valmiit tiedot taulukoihin

Waverley Labsin ohjeen kohdissa 8a – 8d on eritelty vähintään neljä täytettävää taulua. Tein itse ensin suunnitelman aiemmassa kuvassa näkyvistä 15 taulusta (nro 1 closed\_connection – nro 15 user\_group), mutta päätin sitten täyttää vain vähimmäismäärän neljä taulua, jotka ovat kuvan listauksessa taulut:

* 10 (sdpid)
* 11 (sdpid\_service)
* 12 (service) ja
* 13 (service\_gateway)

Seuraavassa on mallina arvot näihin neljään tauluun. Arvoja voi joutua muuttamaan myöhemmin, mutta se lienee helpompaa valmiiden arvojen pohjalta. Lisäykseen on varmasti parempiakin tapoja, mutta omalla SQL-osaamisellani tein nyt näin.

**Taulu 10:**

INSERT INTO sdpid VALUES ('100', 1, 'controller', 'US', 'California', 'Los Angeles', 'Corp Corp', NULL, NULL, NULL, NULL, NULL, '12345', '2022-12-08 16:16:00', '2023-06-05 23:59:00', NULL, NULL);

INSERT INTO sdpid VALUES ('200', 1, 'gateway', 'US', 'California', 'Los Angeles', 'Corp Corp', NULL, NULL, NULL, NULL, NULL, '23456', '2022-12-08 16:16:00', '2023-06-05 23:59:00', NULL, NULL);

INSERT INTO sdpid VALUES ('300', 1, 'client', 'US', 'California', 'Los Angeles', 'Corp Corp', NULL, NULL, NULL, NULL, NULL, '34567', '2022-12-08 16:16:00', '2023-06-05 23:59:00', NULL, NULL);

INSERT INTO sdpid VALUES ('400', 1, 'client', 'FI', 'Uusimaa', 'Helsinki', 'Distance Corp', NULL, NULL, NULL, NULL, NULL, '45678', '2022-12-08 16:16:00', '2023-06-05 23:59:00', NULL, NULL);

**Taulu 11:**

INSERT INTO sdpid\_service VALUES ('1', '1', '2');

INSERT INTO sdpid\_service VALUES ('2', '1', '3');

INSERT INTO sdpid\_service VALUES ('3', '2', '1');

INSERT INTO sdpid\_service VALUES ('4', '2', '2');

INSERT INTO sdpid\_service VALUES ('5', '2', '3');

INSERT INTO sdpid\_service VALUES ('6', '3', '1');

INSERT INTO sdpid\_service VALUES ('7', '3', '2');

INSERT INTO sdpid\_service VALUES ('8', '3', '3');

INSERT INTO sdpid\_service VALUES ('9', '4', '1');

INSERT INTO sdpid\_service VALUES ('10', '4', '2');

INSERT INTO sdpid\_service VALUES ('11', '4', '3');

**Taulu 12:**

INSERT INTO service VALUES ('1', 'controller', 'controller service');

INSERT INTO service VALUES ('2', 'data\_service1\_lin', 'VIP asset Linux');

INSERT INTO service VALUES ('3', 'data\_service2\_win', 'VIP asset Windows ');

**Taulu 13:**

INSERT INTO service\_gateway VALUES ('1', '1', '1', 'udp', '443', '0.0.0.0', '443');

INSERT INTO service\_gateway VALUES ('2', '2', '1', 'tcp', '443', '0.0.0.0', '443');

INSERT INTO service\_gateway VALUES ('3', '2', '1', 'tcp', '443', '0.0.0.0', '443');

Sitten vain exit.

### Config.js-tiedoston asetukset

Seuraavaksi katsotaan kansiossa home/samik/SDPcontroller olevaa määritystiedostoa:

nano config.js

Itse muutin tämän asetuksia vasta myöhemmin, mutta toki jo nyt voi löytämiään asetuksia muuttaa.

### OpenSSL-asennus

Asenna tarvittaessa openssl, vaikka yleensä se tulee Linux-asennuksen mukana. Tämän voi tarkistaa:

openssl version

OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

### OpenSSL-varmenteet ja PKI-infrastruktuurin rakentaminen

Sitten siirrymme vaiheeseen, jossa oma rajallinen kokemukseni tuli vastaan. Annetut OpenSSL-asennusohjeet johtivat useaan virhetilanteeseen, joiden ratkomisessa päädyin seuraamaan useita JavaScript-funktiokutsuja tiedostosta toiseen. Jos saat varmenteet normaalisti toimimaan, hienoa! Valitettavasti ohjeen tämä kohta aiheutti kuitenkin itselleni suuria ongelmia. Mennään kuitenkin eteenpäin. Hyvä OpenSSL-ohje on esimerkiksi Ivan Ristićin OpenSSL Cookbook: <https://www.feistyduck.com/books/openssl-cookbook/>

Waverley Labsin SDP-ohjeen PDF-versiossa on kohdassa 11 kaksi komentoa ja Controller-moduulin GitHub-kuvauksen README.md-tiedostossa puolestaan yksi koottu komento. Seurataan aluksi ensimmäistä PDF-versiota.

openssl genrsa -des3 -out ca.key 4096

Tämä luo RSA-salausalgoritmiin perustuvan yksityisen avaimen, joka salataan symmetrisellä Triple DES -salausalgoritmilla. Lopussa RSA:n avainparin laskemiseen tarvittavan moduluksen pituus määritellään kaksinkertaiseksi (= 4096) ohjeen tekoaikana ilmeisesti OpenSSL:ssä tyypilliseen 2048 bittiin nähden. Määre   
-out ohjaa luotavan yksityisen avaimen kirjoitettavaksi tiedostoon nimeltä ca.key.

Seuraava komento

openssl req -new -x509 -days 365 -key ca.key -out ca.crt

luo varmenteen luontipyynnön. Varmenne on voimassa 365 päivää, sen yksityinen avain on äsken luotu ca.key, ja varmenne kirjoitetaan tiedostoon ca.crt.

RSA-algoritmi ei ole enää OpenSSL-projektin suositus, vaan sen on korvannut Ed25519. Muitakin versiomuutoksia tapahtuu jatkuvasti. Itse sain tässä vaiheessa virheen:

Error: RSA private key not found from openssl output

Tätä selviteltiin pitkään ja hartaasti. Tässä ohjeessa ei valitettavasti voida syventyä OpenSSL:n komentojen kirjoon, koska kyseessä on todellinen kryptografian Swiss Army Knife. Lopulta sain varmenteet luotua.

Toinen vaihtoehto SDP-ohjeessa on käyttää komentoa

openssl req -x509 -sha256 -nodes -days 365 -newkey rsa:4096 -keyout ca.key -out ca.crt

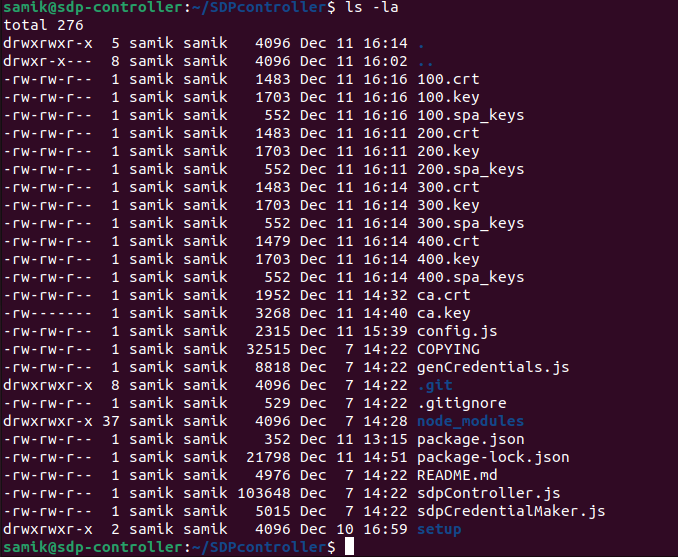
jota voi halutessaan käyttää myös (kumpi sinulla toimii paremmin?).

### 12. Palvelinvarmenteiden ja -avainten luonti

Tässä luodaan esimerkkiavaimet SDP:n Client-, Gateway- ja Controller-komponenteille. Itse käytin SDPID-arvoja 100 (Controller), 200 (Gateway), 300 (sisäinen Client) ja 400 (ulkoinen Client). Kaikki komennot näkyvät tässä:



Palvelinvarmenteet näkyvät kansiolistauksessa:

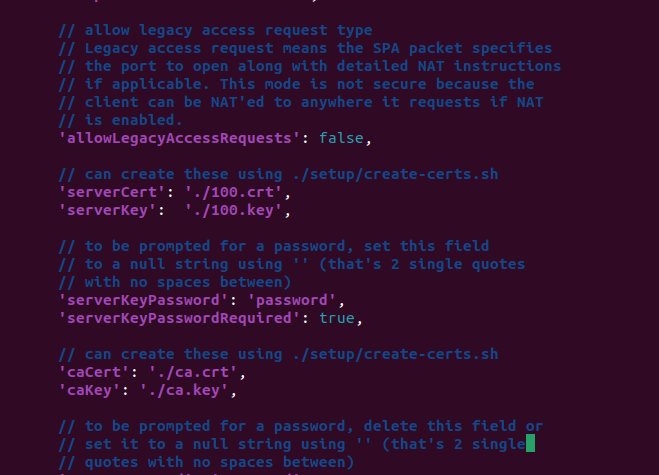


### 13. Controllerin käynnistys

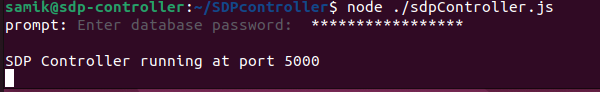
Controller käynnistetään komennolla

node ./sdpController.js

Viimeistään tässä vaiheessa voi olla syytä tarkistaa config.js-tiedoston asetuksia, esim. varmenteiden sijaintipolkuja.



Jos kaikki sujuu hyvin, Controller käynnistyy:



Ohjeen kohdan 12 mukaisesti pitää vielä muistaa siirtää crt- ja key-tiedostot Gateway- ja Client-koneille ja toimia muiden ohjeiden mukaan.

## Gateway-moduulin asentaminen

Tämä asennus on sinänsä suoraviivaisempaa kuin edellinen, mutta Linuxin palomuurin toimintaperiaate on syytä tuntea ainakin jollain tasolla. Olennaista on muistaa, ettei blokkaa itseään ulos asennettavalta Gateway-palvelimelta, jos siihen ottaa SSH-etäyhteyden. SDP-ohjeen kohdan 2 kronologia nimittäin voi johtaa harhaan.

Tämä asennusohje on tehty omaan kotilabraympäristööni aiemmin käytetyn etäympäristön sijaan. Tällä halusin varmistaa toiminnan. Pieniä eroja voi siksi näkyä host-koneiden nimeämistyyleissä.

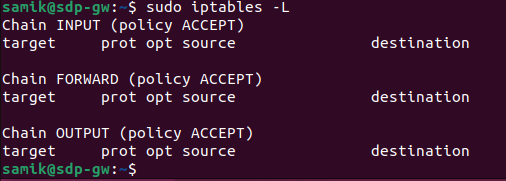
### Linuxin palomuurin asetukset: iptables

Iptablesin voi tarvittaessa asentaa komennolla

sudo apt install iptables

Olemassa olevat palomuurisäännöt (INPUT, FORWARD, OUTPUT) näkee komennolla

sudo iptables -L



### Connection tracking ja iptables

Tässä vaiheessa kaiken liikenteen pitäisi olla sallittua (ACCEPT). Voit toki harjoitella iptables-komentoja, se on hyvää oppia. Etenemisen sujuvoittamiseksi suosittelen ajamaan Gateway-koneelle Michael Rashin skriptin:

https://www.cipherdyne.org/LinuxFirewalls/ch01/iptables.sh.tar.gz

Tämän selitys löytyy kohdasta 1.3 Default-Drop firewall policy sivulta

<https://www.cipherdyne.org/fwknop/docs/fwknop-tutorial.html>

Voi myös käyttää Rashin lyhyempää versiota samalla sivulla:

iptables -I INPUT 1 -i eth0 -p tcp --dport 22 -j DROP

iptables -I INPUT 1 -i eth0 -p tcp --dport 22 -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

Myös SDP-asennusohjeessa on tästä lyhennetty versio ilman protokollan (TCP) ja kohdeportin (22) määrityksiä.

Olennaista on ymmärtää optioiden -A ja -I ero. A eli Append lisää sääntöjä loppuun, kun taas I sijoittaa säännöt määriteltyyn kohtaan, kuten yllä INPUT-taulun riville 1 eli ensimmäiselle riville. Säännöt pätevät ylhäältä alas -järjestyksessä, eli alin ohittaa ylimmän.

Edellä kuvatut Rashin kaksi komentoa taas toimivat seuraavasti. Ensimmäinen hyppää (-j eli Jump) pudottamaan (DROP) kaiken pakettiliikenteen kohdeportista 22 (tässä SSH-oletusportti). Toinen taas hyppää ACCEPT-sääntöön käyttäen kernelin conntrack-ominaisuutta eli olemassa olevien yhteyksien seurantaa. Tämä toinen sääntö hyppää listan ylimmäksi riville 1, eli se toteutetaan ennen DROP-komentoa. Näin varmistetaan, että jo olemassa oleva SSH-etäyhteys pysyy päällä silloinkin, kun DROP-sääntö (tai policy) sen muutoin estää.

Taulun säännöt ovat merkitsevämpiä kuin policyt. Kuitenkin tavoitteena on tuottaa Default Drop policy.

Iptables-asetukset voi tarvittaessa tallettaa:

sudo /sbin/iptables-save

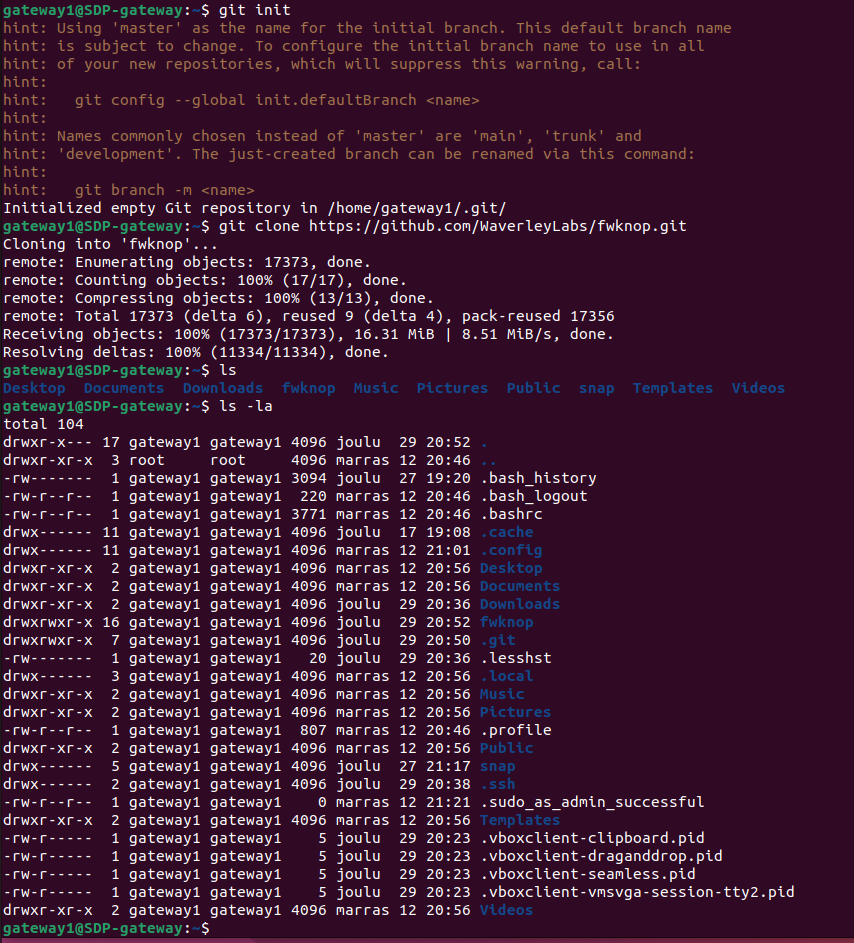
Itse en tehnyt forwarding-asetusta ohjeen kohdan 2 lopussa.

### fwknop: taustakirjastojen asentaminen

Asennetaan seuraavaksi fwknopin tarvitsemat taustakirjastot ohjeen komennon mukaisesti.

### Projektin kloonaus

Alustetaan git myös tälle koneelle ja kloonataan projekti GitHubista**: EI INIT**

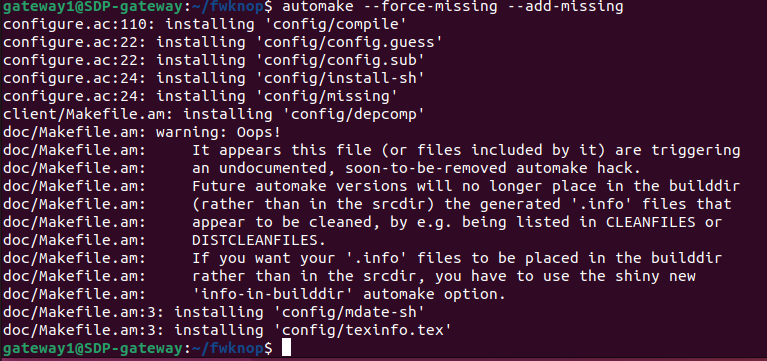


### 5. Mennään oikeaan hakemistoon

Esimerkissä /home/gateway1/fwknop. Kuten mainittiin, Gateway-kone asennetaan tässä kotilabraympäristööni, jossa nimeämiskäytäntö eroaa hieman Controller-esimerkistä.

### 6. Kirjoitetaan komennot

Ohjeen mukaan. Itselläni tuli oireita mahdollisista vaikeuksista, joiden vaikutuksesta kuulen mielelläni tarkemmin asiasta tietäviltä.



### 7. Konfiguroidaan projekti kääntämistä varten

Koska kyseessä on Gatewayn asennus, ei asenneta tälle koneelle Clientia eli käytetään lisämäärettä –disable client:

./configure --prefix=/usr --sysconfdir=/etc --with-iptables=/path/to/iptables --disable-client

### 8. Kääntäminen

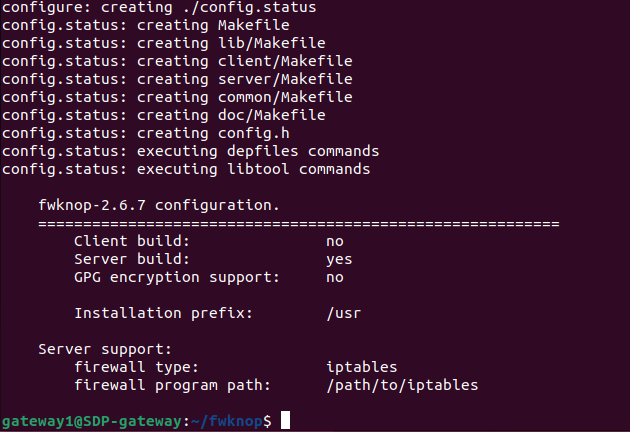
Suoraviivaisesti:

make

### 9. Asentaminen

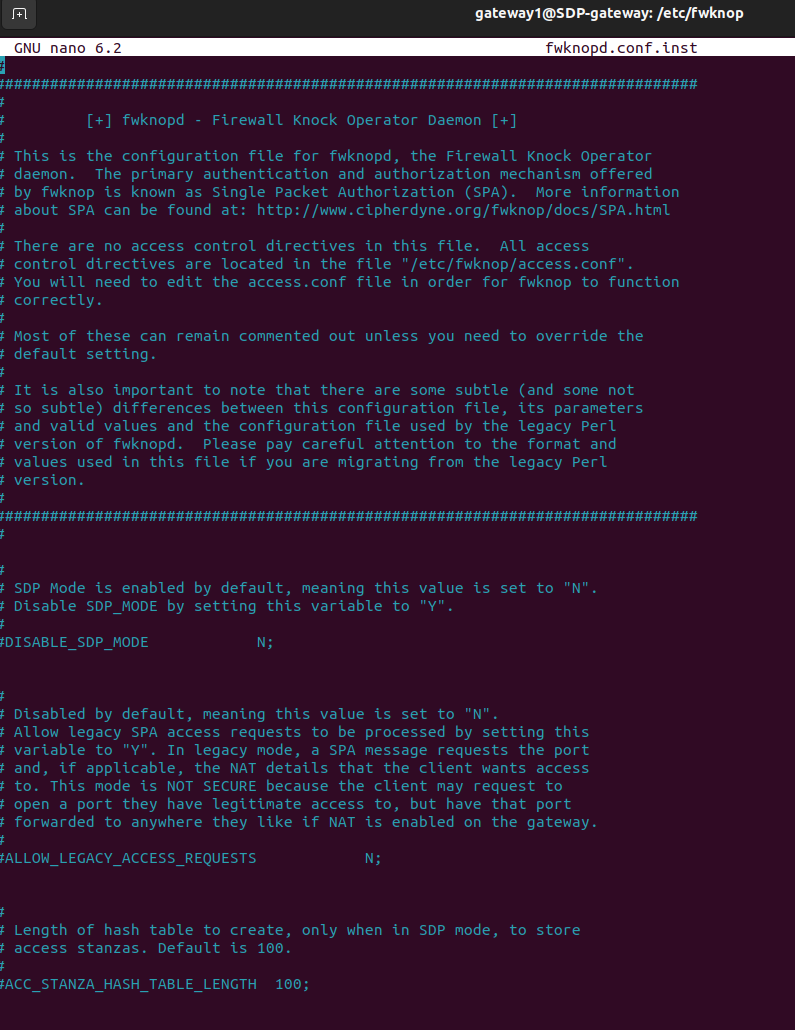
sudo make install

Itselläni tässä tuli erroria ja ajoin ohjeen mukaan tarvittavat komennot uudestaan. Tuloksen tulisi kuitenkin näyttää koko lailla tältä:



### 10. Tiedostojen editointi

Asetustiedostoissa on varsin hyvät selvitykset. Itse puutuin kahteen ensin mainittuun tiedostoon mutta en gate.fwknoprc-tiedostoon.



### 11. Gatewayn käynnistys

Tässä vaiheessa Controller kannattaa pitää käynnissä. Jos kaikki menee hyvin, yhteys koneiden välillä syntyy.

## Client-moduulin asentaminen

### 1.–10. Client-moduuli

Tämän asennus noudattaa pitkälle Gateway-moduulin asennusta. Konfiguraatiotiedostojen muutokset selitetään ohjeessa.

Ohjeen tässä versiossa ei valitettavasti pystytä ottamaan tarkemmin kantaa Client-moduulin asennukseen. Gateway-asennuksen tehneelle Client-asennuksen pitäisi olla varsin tuttua.

## Lopuksi

Tämän ohjeen tekeminen oli ainakin itselleni opettavainen kokemus. Prosessin aikana sain oppia taustateknologioista ja ratkoa monia ongelmia. Omat asennusvaiheeni kestivät muiden kiireiden takia useamman viikon, joten ymmärrän myös lukijoita, joiden fokus matkan varrella herpaantuu.

**Kuulisin joka tapauksessa erittäin mielelläni kokemuksiasi.**

**Mikä toimi? Mikä ehkä ei? Minkä voisi ilmaista paremmin? Saitko kokonaisuuden toimimaan? Haluatko kehittää ohjetta paremmaksi? Mitä erityisesti opit ja mitä et? Mitä ohjeen lukeminen (ja toivottavasti myös käytännössä kokeileminen) opetti Software-Defined Perimeter -teknologiasta, Zero Trustista ja tietoturvasta?**

**Olen äärimmäisen kiitollinen, jos voit antaa tätä palautetta osoitteeseen:**

[**sami.kivela@myy.haaga-helia.fi**](mailto:sami.kivela@myy.haaga-helia.fi)

**Kehitän ohjetta palautteen pohjalta, ja nimeäsi ei tulla julkaisemaan missään tilanteessa.**

**Lämmin kiitos!**

