Labra 2

TTTW0430.7S0V5 Pilvipalvelut

Toni Ahola

Aleksi Ahvenjärvi

Antti Lauttaanaho

Aapo Ratas

Laboratorioharjoitus

11 2017

Tietoverkkotekniikka

Tieto- ja viestintätekniikan koulutusohjelma

Sisältö

[1 Johdanto 5](#_Toc498958864)

[2 Teoria 5](#_Toc498958865)

[2.1 Amazon Web Services 5](#_Toc498958866)

[2.1.1 Amazon EC2 6](#_Toc498958867)

[2.2 HAProxy 6](#_Toc498958868)

[2.3 Docker 7](#_Toc498958869)

[2.4 Contriboard 8](#_Toc498958870)

[3 Labra 1 8](#_Toc498958871)

[3.1 HAProxy asennus 12](#_Toc498958872)

[3.2 Docker virtuaalikone 15](#_Toc498958873)

[3.3 Docker asennus 16](#_Toc498958874)

[3.4 .txt -tiedostot 17](#_Toc498958875)

[3.5 Konttien asennus 18](#_Toc498958876)

[3.6 Contriboard 21](#_Toc498958877)

[3.7 HAProxy static 22](#_Toc498958878)

[3.8 Pohdinta 23](#_Toc498958879)

[4 Labra 2 24](#_Toc498958880)

[4.1 Docker Registry ja komponentit 24](#_Toc498958881)

[4.2 API - kuormantasaus 29](#_Toc498958882)

[4.3 Tietoturva – Security groups 30](#_Toc498958883)

[4.4 Ssh hostnamella 31](#_Toc498958884)

[4.5 Kuormantasaus IO-komponentille 35](#_Toc498958885)

[4.6 Backup suunnittelu Mongolle 36](#_Toc498958886)

[4.7 Pohdinta 36](#_Toc498958887)

[Lähteet 36](#_Toc498958888)

**Kuviot**

[﷟HYPERLINK "bookmark://\_Toc498956933" 7](#_Toc498956933)7

[Kuvio 2. Security Group 8](#_Toc498956934)

[Kuvio 3. Saapuvat säännöt 9](#_Toc498956935)

[Kuvio 4 Lähtevät säännöt 9](#_Toc498956936)

[Kuvio 5. EC2 hallintapaneeli 9](#_Toc498956937)

[Kuvio 6. Private keyn luonti 10](#_Toc498956938)

[Kuvio 7. Privaatin avaimen lisääminen PuTTY:n 11](#_Toc498956939)

[Kuvio 8. SSH yhteyden luonti virtuaalikoneeseen 11](#_Toc498956940)

[Kuvio 9. Virtuaalikoneen konsolinäkymä 12](#_Toc498956941)

[Kuvio 10. Ubuntuserverin päivitys 12](#_Toc498956942)

[Kuvio 11. HAProxyn asennus 13](#_Toc498956943)

[Kuvio 12. HAProxyn tilan tarkistus 13](https://jamkstudent-my.sharepoint.com/personal/k2117_student_jamk_fi/Documents/Pilvipalvelut/Ryhma11_Labra2.docx#_Toc498956944)

[Kuvio 13. HTTP sääntö 13](#_Toc498956945)

[Kuvio 14. haproxy.cfg 14](#_Toc498956946)

[Kuvio 15. HAProxyn uudelleen käynnistys 15](#_Toc498956947)

[Kuvio 16. Sisäverkon koneeseen kirjautuminen 15](#_Toc498956948)

[Kuvio 17. Virtuaalikoneen päivitys 16](#_Toc498956949)

[Kuvio 18. Docker asennus 1 16](#_Toc498956950)

[Kuvio 19. Docker asennus 2 16](#_Toc498956951)

[Kuvio 20. Docker asennus 3 16](#_Toc498956952)

[Kuvio 21. Dockerin tilan tarkistus 16](#_Toc498956953)

[Kuvio 22. API:n envit tiedosto 17](#_Toc498956954)

[Kuvio 23. IO:n envit tiedosto 17](#_Toc498956955)

[Kuvio 24. Client:n envit tiedosto 17](#_Toc498956956)

[Kuvio 25. Mongodb:n asennus 18](#_Toc498956957)

[Kuvio 26. Redis -kontin asennus 19](#_Toc498956958)

[Kuvio 27. API -kontin asennus 19](#_Toc498956959)

[Kuvio 28. IO -kontin asennus 19](#_Toc498956960)

[Kuvio 29. Client -kontin asennus 20](#_Toc498956961)

[Kuvio 30. Asennetut kontit 20](#_Toc498956962)

[Kuvio 31. Contriboardin etusivu 21](#_Toc498956963)

[Kuvio 32. Contriboard 22](#_Toc498956964)

[Kuvio 33. Stats -sivun osoitteen ja käyttäjätunnusten määritys 22](#_Toc498956965)

[Kuvio 34. ACL määritys 23](#_Toc498956966)

[Kuvio 35. HAProxyn statistiikkasivu 23](#_Toc498956967)

[Kuvio 36. Koneiden jako 24](#_Toc498956968)

[Kuvio 37. Certtien luonti 25](#_Toc498956969)

[Kuvio 38. Registry -kontin luonti 25](#_Toc498956970)

[Kuvio 39. Certin lisääminen luotetuksi 25](#_Toc498956971)

[Kuvio 40. Imagen pullaus docker hubista 26](#_Toc498956972)

[Kuvio 41. Imagen tagaaminen 26](#_Toc498956973)

[Kuvio 42. Imagen pushaaminen rekisteriin 26](#_Toc498956974)

[Kuvio 43. id\_rsa -tiedoston luonti 27](#_Toc498956975)

[Kuvio 44. Certin siirto 27](#_Toc498956976)

[Kuvio 45. Luotettu certti 27](#_Toc498956977)

[Kuvio 46. /etc/hosts 28](#_Toc498956978)

[Kuvio 47. Imagen pullaaminen rekisteristä 28](#_Toc498956979)

[Kuvio 48. Frontend konfiguraatio 29](#_Toc498956980)

[Kuvio 49. Backend API 29](#_Toc498956981)

[Kuvio 50. Stats -API 30](#_Toc498956982)

[Kuvio 51. Security groups 30](#_Toc498956983)

[Kuvio 52. Yhteys contriboardiin 31](#_Toc498956984)

[Kuvio 53. Hosts 32](#_Toc498956985)

[Kuvio 54. Hostname 32](#_Toc498956986)

[Kuvio 55. Hostname -HaProxy 32](#_Toc498956987)

[Kuvio 56. Network restart 32](#_Toc498956988)

[Kuvio 57. SSH hostnamella 33](#_Toc498956989)

[Kuvio 58. Avain.pem nimen muuttaminen 33](https://jamkstudent-my.sharepoint.com/personal/k2117_student_jamk_fi/Documents/Pilvipalvelut/Ryhma11_Labra2.docx#_Toc498956990)

[Kuvio 59. Id\_rsa -tunnistettu avain 34](#_Toc498956991)

[Kuvio 60. Id\_rsa oikeudet ja ssh yhteys 34](#_Toc498956992)

[Kuvio 61. Backend WS määrittelyt 35](#_Toc498956993)

[Kuvio 62. Kuormantasaus 1 35](#_Toc498956994)

[Kuvio 63. Kuormantasaus 2 36](#_Toc498956995)

# Johdanto

Labratyö on toteutettu *Pilvipalvelut* opintojaksolla. Labrassa käytetään Amazonin AWS (Amazon Web Services) pilvilaskenta-alustaa, jossa labratyö toteutetaan. AWS:n palveluna toimii Amazon EC2. Labrassa otetaan käyttöön kaksi Ubuntu virtuaalipalvelinta, joista toiseen asennetaan HAProxy ja toiseen Docker. HAProxy kone konfiguroidaan oikeille asetuksille, jotta se osaa kommunikoida Docker koneella olevien konttien kanssa. Kontit sisältävät Contriboard tuotteen, joka on avoimen lähdekoodin web-pohjainen muistiinpano-ohjelmisto. Labrassa on siis tarkoituksena saada Contriboard toimimaan oikein, ja myös lisänä saada HAProxyn statistiikkasivu toimimaan.

# Teoria

## Amazon Web Services

AWS on Amazon.com:in tarjoama pilvilaskenta-alusta. AWS:ään kuuluu monia pilvipalvelu -palveluita, joista tunnetuimmat ovat Amazon EC2 (Elastic Computer Cloud) ja Amazon S3 (Simple Storage Service). Pilvipalveluiden hallinnointiin ja luomiseen käytetään AWS-hallintapaneelia, mutta virtuaalikoneiden hallintaan voidaan käyttää Amazonin hallintapaneeelista löytyviä graafisia tai komentorivipohjaisia työvälineitä. Labrassa virtuaalikoneiden hallintaan käytettiin ohjelmaa nimeltä Putty. (Amazon Web Services, 2017.)

AWS:n alusta pohjautuu IaaS (Infrastructure as a Service) pilvipalvelumuotoon, jossa Amazon tarjoaa käyttäjälle ydinpalvelut: laskentateho, levykapasiteetti, tietoverkot, muisti ja käyttöjärjestelmä. Amazonin asiakas voi näin rakentaa itse kokonaisia ympäristöjä AWS:n päälle. Pilvipalvelumuodon raja on vuosien aikana muuttunut, koska heidän palveluistaan löytyy myös PaaS- ja SaaS kategoriaan kuuluvia palveluita.

### Amazon EC2

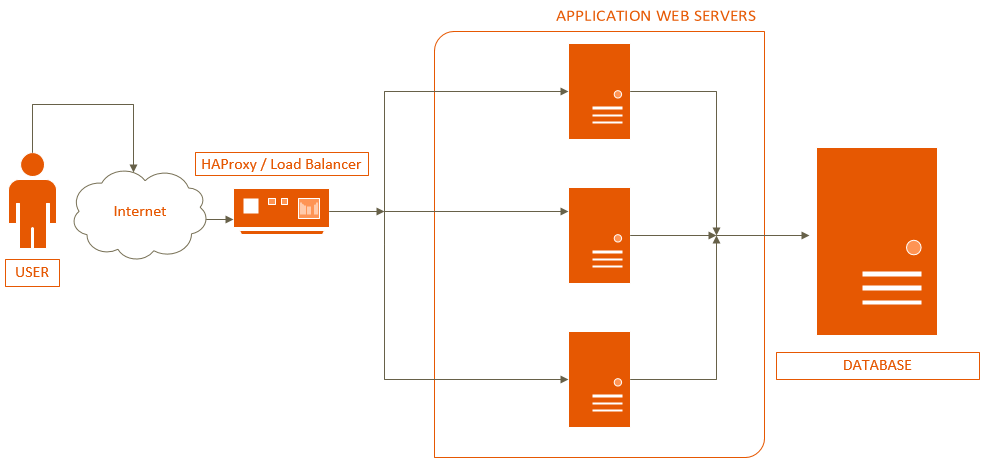
Amazon.com:n EC2 palvelu on keskeinen osa Amazon.com AWS:n pilvilaskenta-alustaa. Käyttäjät ostavat EC2 palvelininstanssin, jonka jälkeen he voivat luoda omia virtuaalikoneita Amazonin pilveen. Palvelininstanssin asentamisen jälkeen virtuaalikoneelle asennetaan käyttöjärjestelmä ja tarvittavat palvelinsovellukset. Eli kun käyttäjä on luonut virtuaalikoneen, hän voi ajaa omia palveluitaan virtuaalikoneessa.

Amazon tarjoaa myös valmiiksi optimoituja Amazon Machine Image (AMI) –levykuvia näille virtuaalikoneille. Valmiita AMI –levykuvia ovat mm. Amazon Linux, Ubuntu Server ja Microsoft Windows Server. Tässä labrassa käytetään juuri Amazonin EC2 palvelua ja valmista Ubuntu Server AMI –levykuvaketta. (Amazon Elastic Compute Cloud, 2017.)

## HAProxy

HAProxy eli High Availability Proxy on ilmainen, avoimen lähdekoodin ohjelmisto. HAProxy on erittäin suosittu TCP/HTTP Load Balancer eli kuormantasaaja, ja proxy –palvelin. Sen käytöllä parannetaan palvelinympäristön suorituskykyä ja luotettavuutta. Tämä toteutetaan jakamalla TCP ja HTTP pohjaisten sovelluksien pyyntöjä monille palvelimille. (ks. Kuvio 1.)

HAProxy on myös siksi tehokas, koska se on kirjoitettu C-kielellä. C-kielellä on hyvä maine tehokkaasta ja nopeasta muistin ja prosessorin käytöstä. Korkean tason nettisivut, kuten mm. GitHub, Instagram, Imgur, Twitter ja Reddit käyttävät myös HAProxya. (An Introduction to HAProxy and Load Balancing Concepts, 2014.)



Kuvio 1. HAProxy esitettynä

## Docker

Docker on avoin ohjelmistoalusta, jonka avulla ylläpitäjät voivat kehittää, käyttää ja siirtää sovelluksia ja kokonaisia ohjelmistoja konttitenkologian (Container technology) avulla. Dockerin avulla sovellukset voidaan hajauttaa infrastruktuurista, jotta ohjelmistoja voi siirtää nopeasti. Dockerin avulla infrastruktuuria voidaan silti hallita samallalailla kuin hallitsisi sovelluksia.

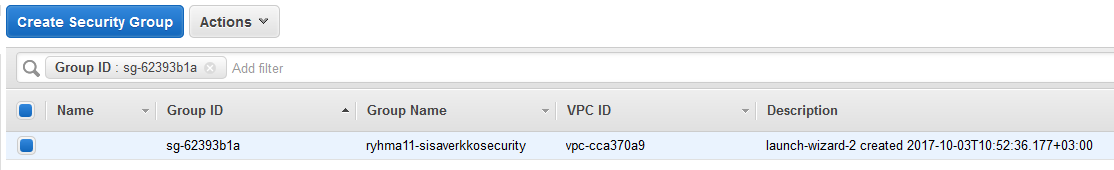
Ohjelmistojen nopea pakkaaminen, suoritus ja siirtäminen perustuu siis konttiteknologiaan. Kontit ovat eristetyssä ja suojattussa ympäristössä, ja tämän avulla kontit jakavat samanaikaisesti yhteisen isäntäkoneen (Host). Docker toimii siis yhteisen käyttöjärjestelmän päällä. Konttiteknologian avulla ohjelmistoista tehdään kevyitä ja näin ne eivät tarvitse hypervisorin ylimääräistä kuormitusta, vaan pyörivät ja jakavat yhteisen host-koneen kernelin. Eli kontteja voidaan tuottaa enemmän ja tehokkaammin kuin kokonaisia virtuaalikoneita. Kontteja käytetään siis host-koneessa todellisina virtuaalikoneina. (Docker overview, 2017.)

## Contriboard

Contriboard on N4S@JAMK projektiryhmän kehittämä reaaliaikainen avoimenlähdekoodin web-pohjainen ja digitaalinen yhteis- ja muistiinpanotyökalu. Tuote on tarkoitettu tabletti- ja työpöytä käyttäjille. Erilaisiset projektiryhmät, tiimit tai vaikka yksityiset henkilöt voivat käyttää sovellusta ja näin helpottaa yhteistä tai yksityistä työntekoaan. Contriboardia kehitetään nykyaikasilla tekniikoilla, ja on näin mahdollista ottaa käyttöön nopeasti ja varmasti. (Nelimarkka P., 2014.)

# Labra 1

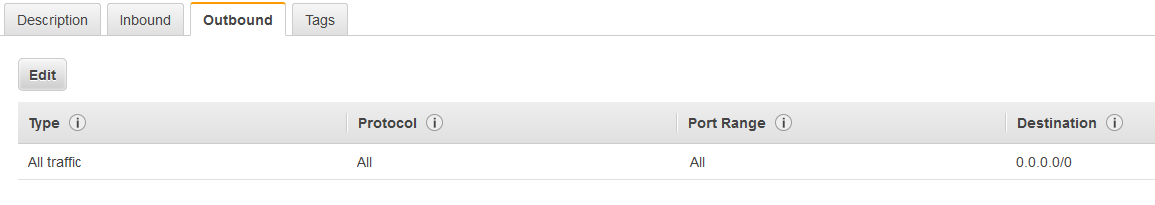
Ensiksi suunnattiin Amazonin EC2 pilvipalveluun ryhmän omilla tunnuksilla ja luotiin uusi instanssi. Virtuaali koneeksi valittiin Amazonin valmis AMI-levykuvake, Ubuntu Server AMI. Instanssin asennuksessa määritellään paljonko halutaan mm. CPUita tai Memorya käyttöön. Kun tämä oli tehty, luotiin omalle ryhmälle Security Group. (ks. Kuvio 2.) Security Groupin avulla sallitaan sisäverkon saapuvaa ja lähtevää liikennettä. Sallittiin kaikki liikenne, niin tulevalle kuin lähtevälle. Saapuville säänöille lisättiin vielä SSH. Valittiin protokolla TCP ja vakioportti SSH -portti 22. (ks. Kuvio 3. ja 4.) Seuraavaksi voidaan laittaa virtuaalikoneet päälle. (ks. Kuvio 5.)



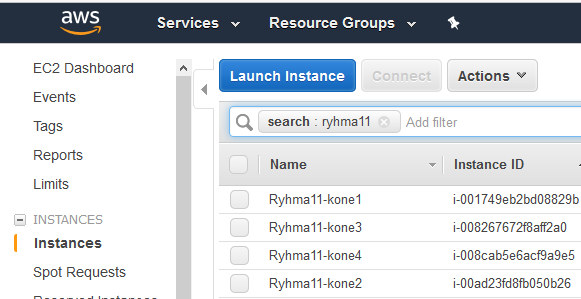
Kuvio 2. Security Group



Kuvio 3. Saapuvat säännöt

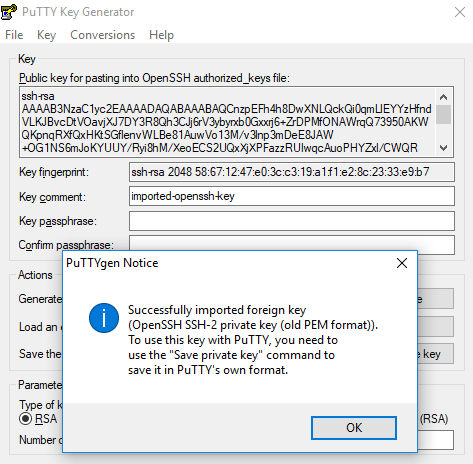


Kuvio 4 Lähtevät säännöt



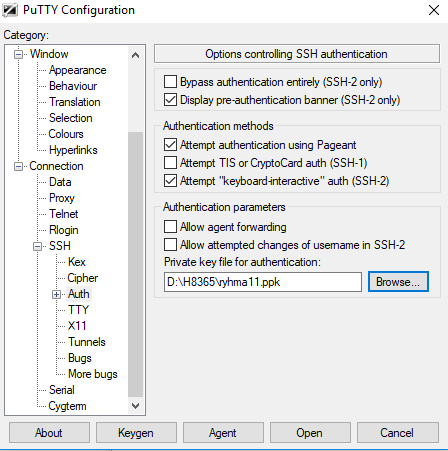
Kuvio 5. EC2 hallintapaneeli

Seuraavaksi luotiin privaatti avain ryhmälle annetusta julkisesta avaimesta, PuttyGen:n ohjelman avulla. (ks. Kuvio 6.) Privaatin avaimen luonti oli välttämätöntä, jotta päästiin käsiksi ryhmälle tarkoitettuihin Amazonin pilvipalvelussa oleviin virtuaalikoneisiin.



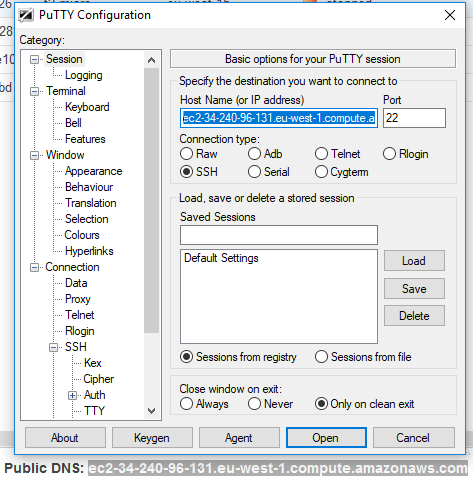
Kuvio 6. Private keyn luonti

Privaatti avain lisätään PuTTYn tietoihin, jonka jälkeen sitä voidaan käyttää SSH yhteyden autentikointiin. (ks. Kuvio 7.)

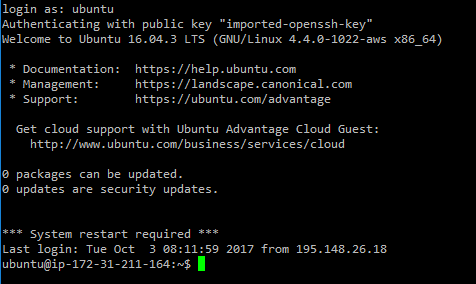


Kuvio 7. Privaatin avaimen lisääminen PuTTY:n

Nyt SSH-yhteys voidaan luoda virtuaalikoneeseen, jolla on Amazonin antama julkinen IP-osoite. (ks. Kuvio 8.) Onnistuneen yhteyden luonnin jälkeen päästään sisälle virtuaalikoneeseen. (ks. Kuvio 9.)



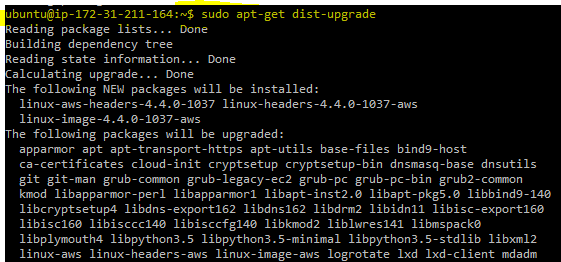
Kuvio 8. SSH yhteyden luonti virtuaalikoneeseen



Kuvio 9. Virtuaalikoneen konsolinäkymä

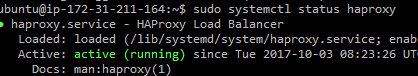
## HAProxy asennus

Päivitetään Ubuntu Server ja asennetaan HAProxy halutulle virtuaalikoneelle. (ks. Kuvio 10. ja 11.) Tarkistetaan HAProxyn tila systemctl –komennolla jotta voidaan todeta se toimivaksi, ja että HAProxy on käynnissä. (ks. Kuvio 12.)



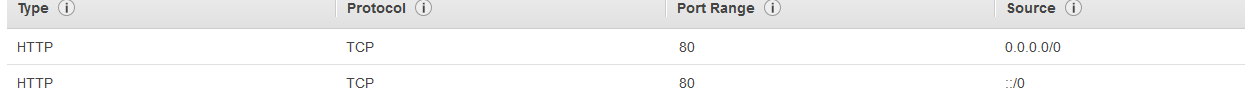
Kuvio 10. Ubuntuserverin päivitys



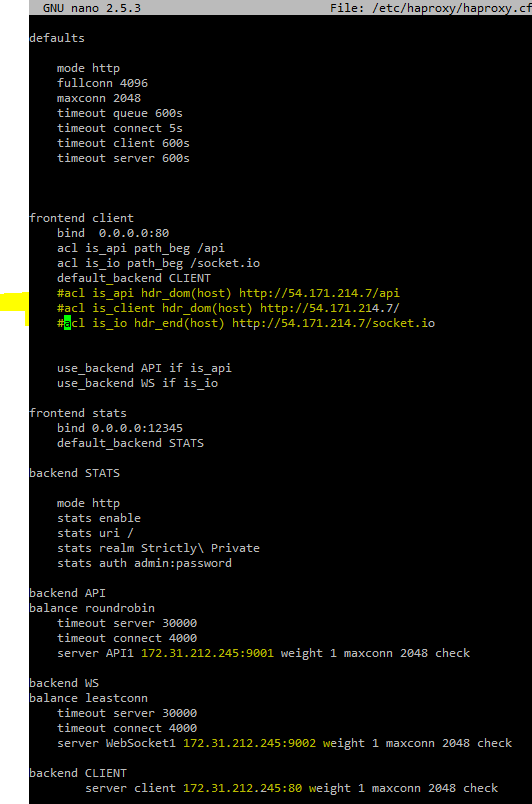
Kuvio 11. HAProxyn asennus

Kuvio 12. HAProxyn tilan tarkistus

Luodaan Amazonin web-käyttöliittymässä sääntö, jossa sallitaan kaikki 80 porttiin tuleva http liikenne. (ks. Kuvio 13.)

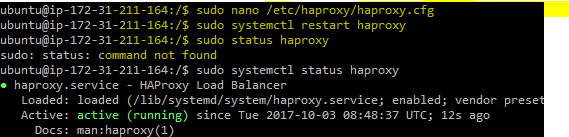
Kuvio 13. HTTP sääntö

Seuraavaksi tutkitaan HAProxyllä olevaa haproxy.cfg –tiedostoa, joka sijaitsee */etc/haproxy/* polussa. Tiedosto muokataan tarpeen mukaan sopivaksi. Olennaista on kiinnittää huomiota backend API, WS ja Client IP-osoitteiden oikeellisuuteen. (ks. Kuvio 13.) Joihinka tulee toisen (Docker) virtuaalikoneen IP-osoitteet. Kuviossa 13 olevista kommentoiduista (#) keltaisista riveistä ei tarvitse välittää. Nämä on värjätty vahingossa.



Kuvio 14. haproxy.cfg

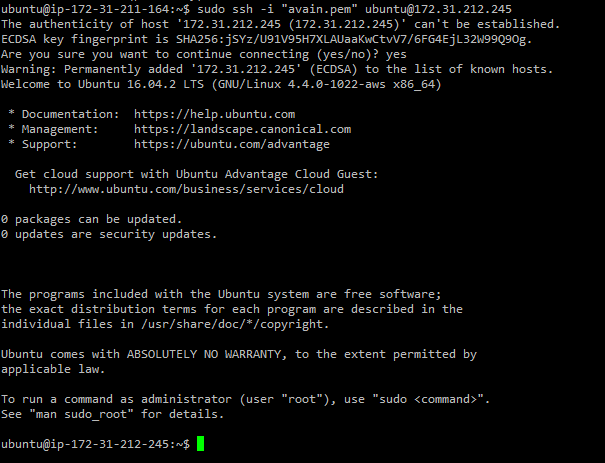
.cfg -tiedoston muokkaamisen jälkeen HAProxy tulee käynnistää uudelleen ja tarkistaa sen tila. (ks. Kuvio 15.)



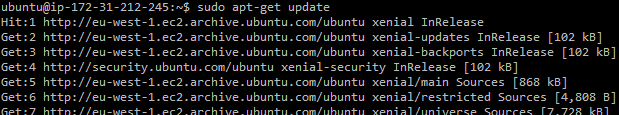
Kuvio 15. HAProxyn uudelleen käynnistys

## Docker virtuaalikone

Koska virtuaalikone, johon Docker asennetaan, sijaitsee sisäverkossa, täytyy sinne kulkea HAProxy -koneen kautta. (ks. Kuvio 16.) HAProxy koneelta otetaan SSH-yhteys sisäverkossa olevalle koneelle, jolloin se tarvitsee myös private keyn. Labrassa private key -tiedosto on luotu ubuntu -käyttäjän kotikansioon nimellä ”avain.pem”. Lisäksi uusi virtuaalikone päivitetään rutiininomaisesti. (ks. Kuvio 17.)



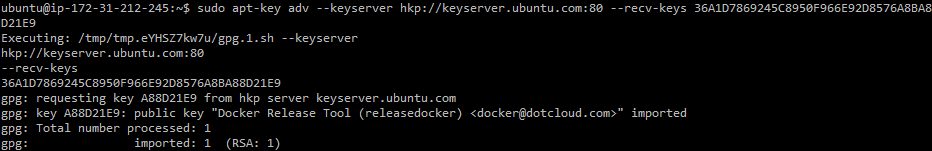
Kuvio 16. Sisäverkon koneeseen kirjautuminen



Kuvio 17. Virtuaalikoneen päivitys

## Docker asennus

Ensiksi asennetaan Docker ja tarkistetaan sen tila. (ks. Kuvio 18.,19.,20. ja 21.)



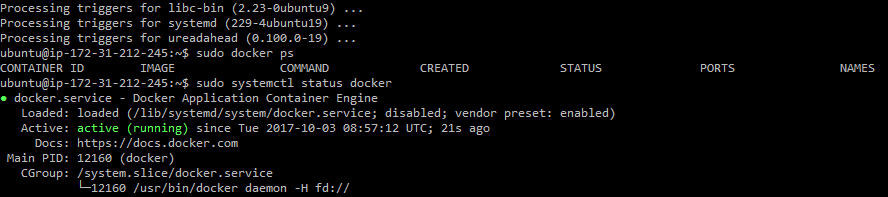
Kuvio 18. Docker asennus 1



Kuvio 19. Docker asennus 2



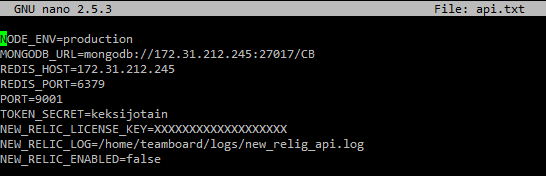
Kuvio 20. Docker asennus 3



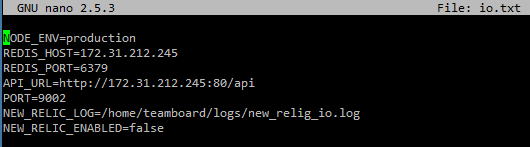
Kuvio 21. Dockerin tilan tarkistus

## .txt -tiedostot

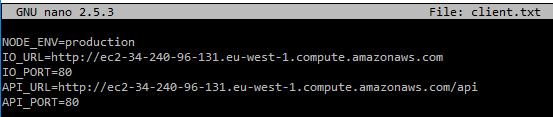
Ennen konttien asennusta tulee jokaiselle Contriboardin kontille luoda tarvittavat envit tiedostot. Tiedostot ovat tavallisia .txt -tiedostoja ja niiden luonnissa tulee kiinnittää huomiota mongodb:n ja redis:n IP -osoitteisiin. Tiedostot sijoitettiin */envit* kansioon. Tässä harjoituksessa kaikki kontit asennetaan samalle virtuaalikoneelle, joten IP -osoitteeksi määritetään sen hetkisen virtuaalikoneen osoite. Contriboardin kontteja on yhteensä kolme: api, io ja client. Näille jokaiselle tulee luoda oma envit -tiedosto. (ks. Kuvio 22., 23., ja 24.)



Kuvio 22. API:n envit tiedosto



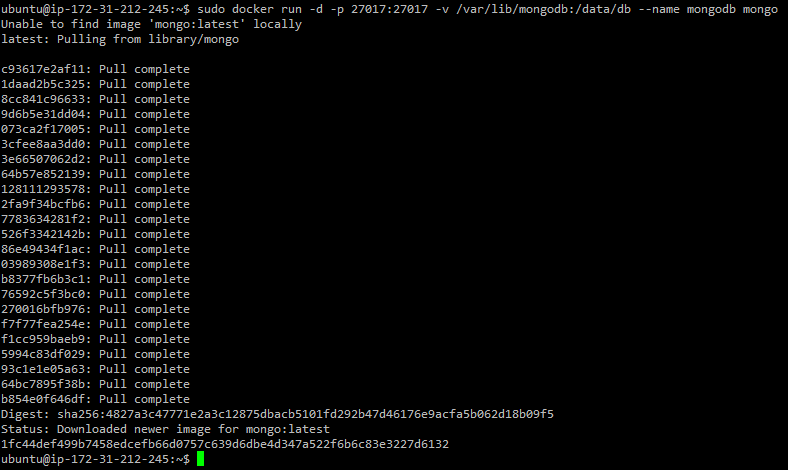
Kuvio 23. IO:n envit tiedosto



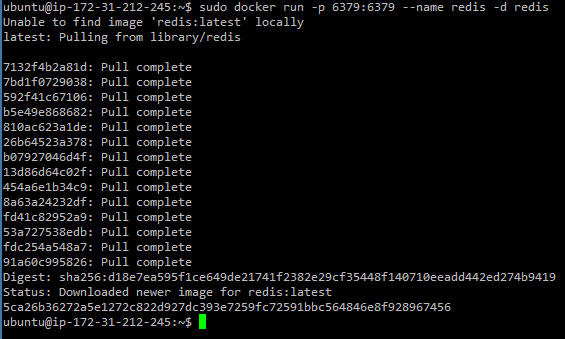
Kuvio 24. Client:n envit tiedosto

## Konttien asennus

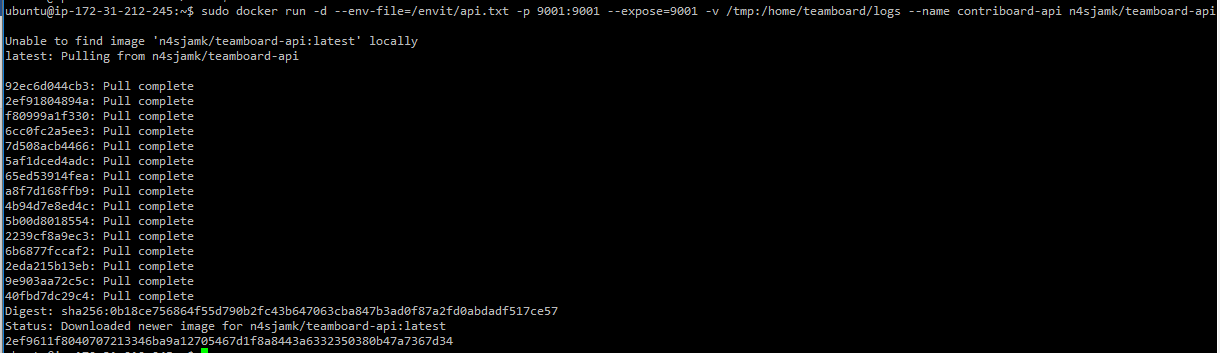
Tarvittavia kontteja on siis yhteensä viisi: Mongo, Redis, Api, IO ja Client. Jokainen kontti asennetaan yksitellen, jolloin tulee määrittää kaikki kontin asetukset. Jos esimerkiksi kontin envit tiedostoon tehdään muutoksia, on kyseinen kontti pysäytettävä, poistettava ja asennettava uudelleen. Ohessa tarvittavat komennot eri konttien asennuksiin. (ks. Kuvio 25., 26., 27., 28. ja 29.)



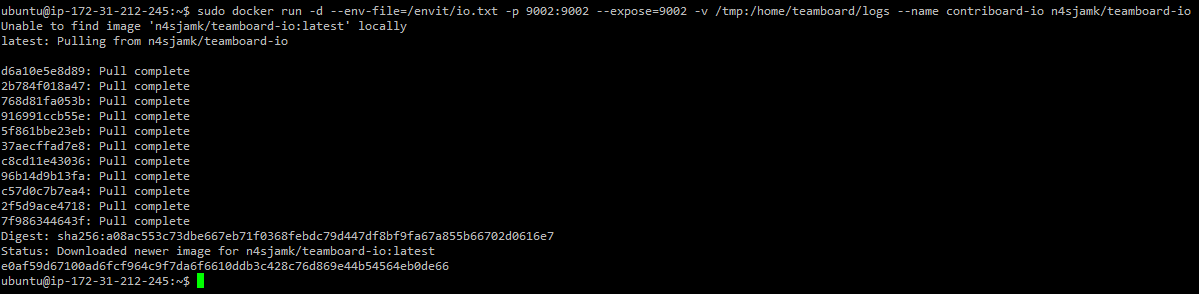
Kuvio 25. Mongodb:n asennus



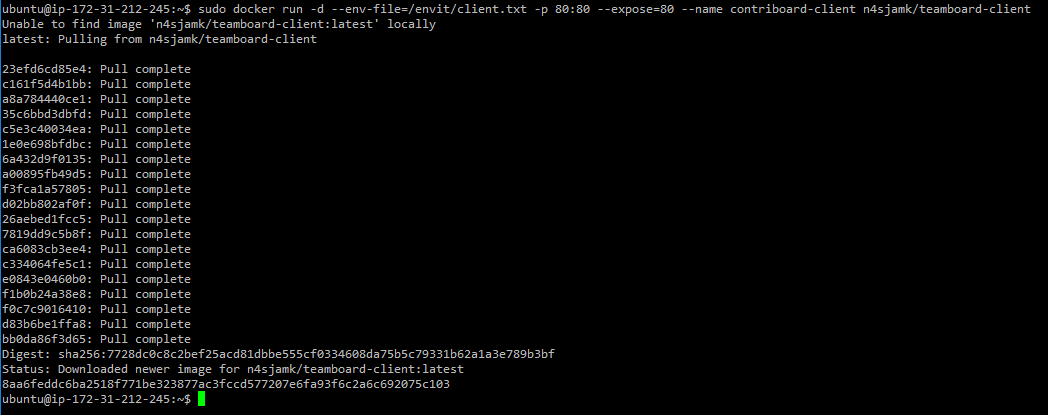
Kuvio 26. Redis -kontin asennus



Kuvio 27. API -kontin asennus

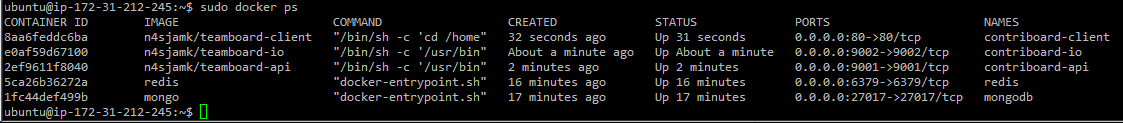


Kuvio 28. IO -kontin asennus



Kuvio 29. Client -kontin asennus

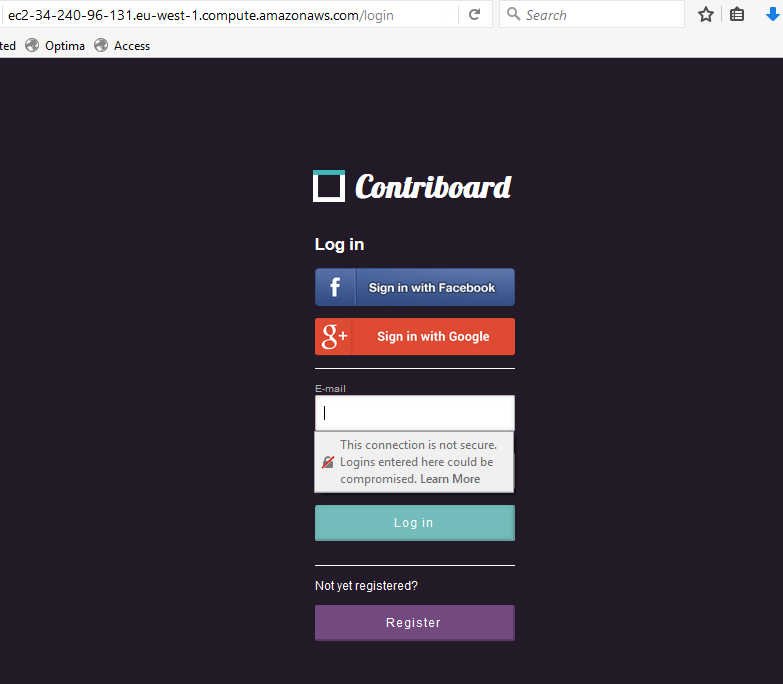
Lopuksi voidaan tarkistaa kaikki asennetut kontit *docker ps -*komennolla. (ks. Kuvio 30.)



Kuvio 30. Asennetut kontit

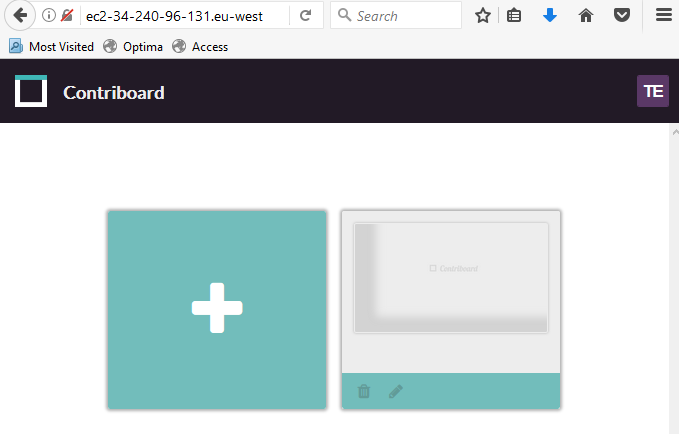
## Contriboard

Jos konttien asennus onnistui toivotusti, HAProxyn julkiseen IP -osoitteeseen meneminen selaimella ohjaa käyttäjän Contriboard:in. (ks. Kuvio 31.)



Kuvio 31. Contriboardin etusivu

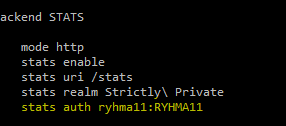
Jos käyttäjä haluaa testata Contriboardia, hänen tulee luoda itselleen tunnukset verkkosivulla, minkä jälkeen hän voi kirjautua sisään. (ks. Kuvio 32.)



Kuvio 32. Contriboard

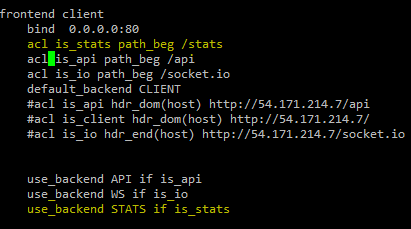
## HAProxy static

Lisäksi harjoituksessa otettiin käyttöön HAProxyn statistiikkasivu, jossa voidaan tarkastella esim. konttien toimintaa. Tätä varten HAProxyn .cfg tiedostoon on tehtävä muutoksia. Ensiksi .cfg tiedostossa määritetään osoite, johon yhdistämällä päästään statistiikkasivuille, sekä käyttätunnukset sivulle. Pääte voi olla jokin portti tai osoite, tässä työssä se määritettiin olevaksi /stats. (ks. Kuvio 33.)



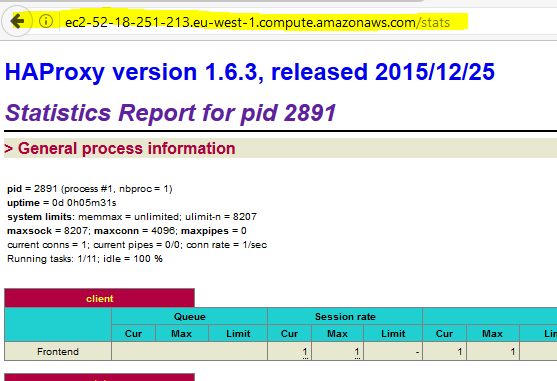
Kuvio 33. Stats -sivun osoitteen ja käyttäjätunnusten määritys

Seuraavaksi määritetään samassa .cfg tiedostossa access control list sääntö, jotta liikenne /stats sivulle sallitaan. (ks. Kuvio 34.)



Kuvio 34. ACL määritys

Onnistuneen konfigutaation jälkeen /stats sivulle meneminen onnistuu selaimella yhdistämällä HAProxyn julkiseen osoitteeseen. (ks. Kuvio 35.)



Kuvio 35. HAProxyn statistiikkasivu

## Pohdinta

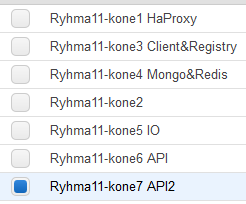
Labratyö onnistui hyvin ja kaikki saatiin toimimaan. Kaikki olivat myös labrassa paikalla ja kaikki tekivät oman osansa työstä. HAProxy ja Docker koneet saatiin asennettua ja conffattua oikein ensinmäisellä labra kerralla, ja HAProxyn Static sivu saatiin toisella labra kerralla. Eniten aikaa kului raportin kirjoittamiseen. Vaadittavat teoriat työhön aukaisivat vielä hyvin erikseen labrassa käytettyjä ”komponentteja”, mm. Amazon ja HAProxy.

Ennen labratöitä tehdyt Docker –harjoitukset auttoivat suuresti labratyön suorituksessa. Ainoana poikkeuksena labrassa oli, että koneet laitettiin pyörimään Amazonin pilveen. Niin kuin oikeassa elämässä tehtäisiin. Muuten labra vastasi edellisiä Docker –harjoituksia. Amazonin pilvipalvelu alustaa oli yllättävän helppo käyttää. Kokonaisuudessaan hyvä käytännön labra ja onnistunut sellainen.

# Labra 2

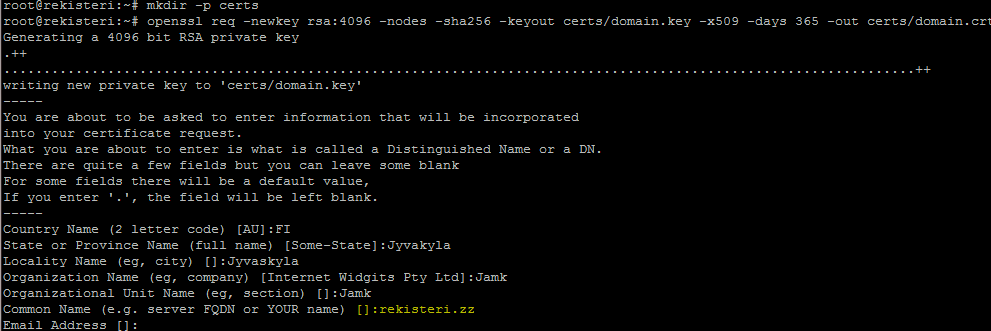
## Docker Registry ja komponentit

Labran tavoite on asentaa Docker Registry yhdelle koneelle, sekä contriboard toimimaan neljällä koneella. Lisäksi jokaisella koneella vain yksi komponentti, poislukien redis ja mongo, jotka ovat samalla koneella. (Ks. Kuvio 36.)



Kuvio 36. Koneiden jako

Ensinksi luodaan kansio certeille ja luodaan certti. (ks. kuvio 37) Certin luonnissa on tärkeää huomata rekisterin FQDN (tässä tapauksessa rekisteri.zz), joka on oltava joka paikassa sama mm. /etc/hosts -kansiossa.



Kuvio 37. Certtien luonti

Luodaan registry -kontti, joka käyttää juuri luotuja certtejä. (ks. Kuvio 38.)



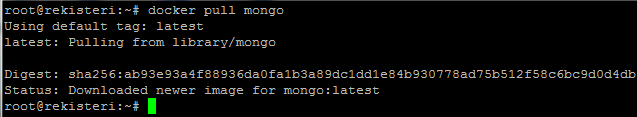
Kuvio 38. Registry -kontin luonti

Luodaan certs.d ja rekisteri.zz -kansiot docker -kansion alle, johon siirretään luotu certti. Tämä toimenpide tekee certistä luotetun. (ks. Kuvio 39.)



Kuvio 39. Certin lisääminen luotetuksi

Pullataan image rekisterikoneelle docker hubista. (ks. Kuvio 40.)



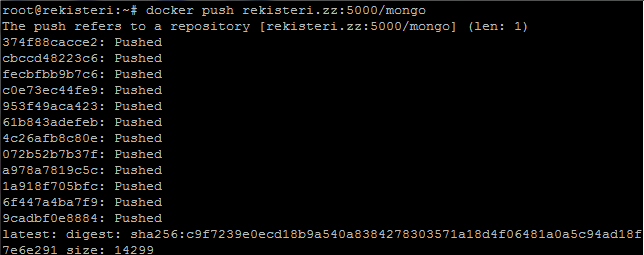
Kuvio 40. Imagen pullaus docker hubista

Tagataan image, jotta se voidaan pushata oikeaan konttiin/porttiin. (ks. Kuvio 41.)



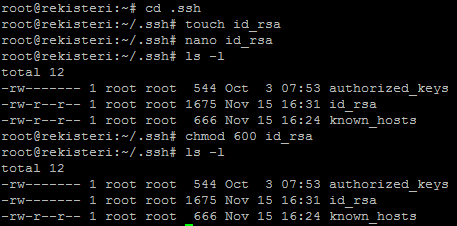
Kuvio 41. Imagen tagaaminen

Pushataan image rekisteriin. (ks. Kuvio 42.)



Kuvio 42. Imagen pushaaminen rekisteriin

Luodaan id\_rsa tiedosto rekisteri koneelle, jotta voidaan siirtää certifikaatti toiselle koneelle. Id\_rsa tiedosto on sama kuin koneella, johon otetaan yhteys ulkoverkosta. (ks. Kuvio 43.)



Kuvio 43. id\_rsa -tiedoston luonti

Siirretään certti koneelle, johon halutaan pullata imageja rekisteristä. (ks. Kuvio 44.)



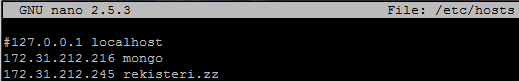
Kuvio 44. Certin siirto

Lisätään certti luotettuihin certteihin. (ks. Kuvio 45.)



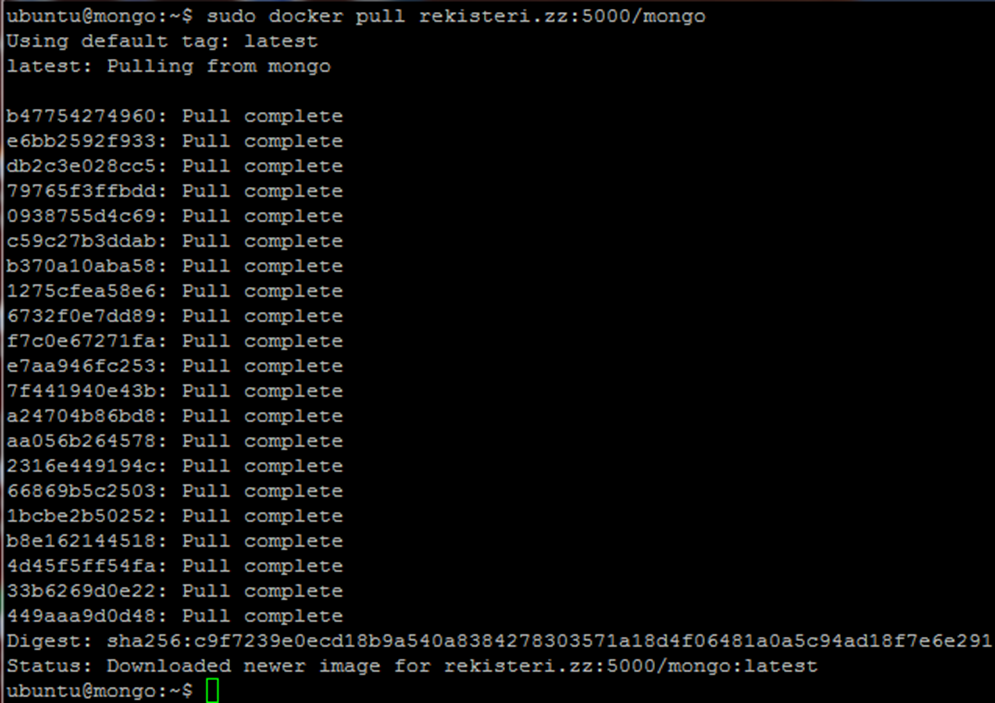
Kuvio 45. Luotettu certti

Lisätään rekisterikone tunnettuihin hosteihin. (ks. Kuvio 46.)



Kuvio 46. /etc/hosts

Pullataan rekisterissä oleva image. (ks. Kuvio 47.)



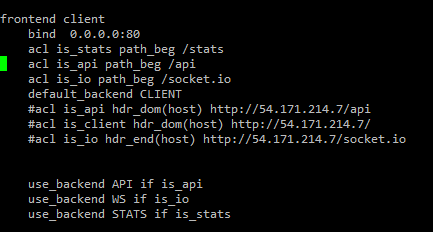
Kuvio 47. Imagen pullaaminen rekisteristä

## API - kuormantasaus

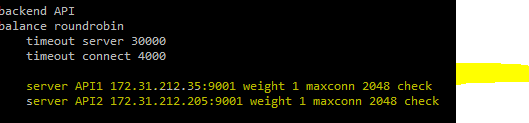
Kun koneille ollaan asennettu komponentit ja todennettu niiden toimivuus, lisätään kaksi API-konetta, joista toinen samalle koneelle, missä API jo on, ja toinen uudelle koneelle.

Tämän jälkeen suoritetaan kuorman jakoa HaProxylta käsin. Ideana on, että kuorma(liikenne) jaetaan API koneiden kesken. Tämä tapahtuu siten, että muokataan ''/etc/haproxy/haproxy.cfg'' tiedostoa, siten että määritetään molemmat API koneet backend API:iin. (ks. Kuvio 49.)

Käytännössä ensin määrittelemme frontendiin: Jos liikenne = API, niin ohjataan se backend APIin.(ks. Kuvio 49.) Backend API:ssa taas määritetään molempien API koneiden IP, jolloin kuorma jaetaan näiden koneiden välillä.

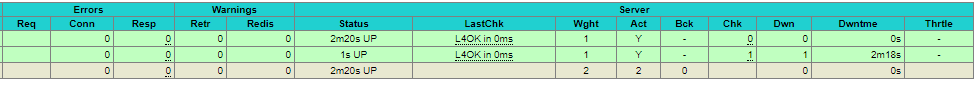
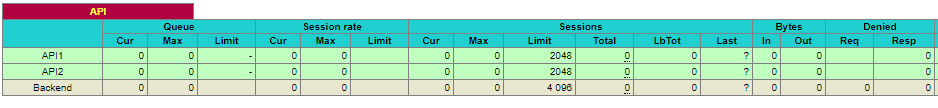


Kuvio 48. Frontend konfiguraatio



Kuvio 49. Backend API

Tämän jälkeen todennetaan, että kuormantasaus toimii halutulla tavalla. Tähän käytämme haproxyn /stats sivusto, jolta näemme liikenteen serverien välillä.(ks. Kuvio 50.)



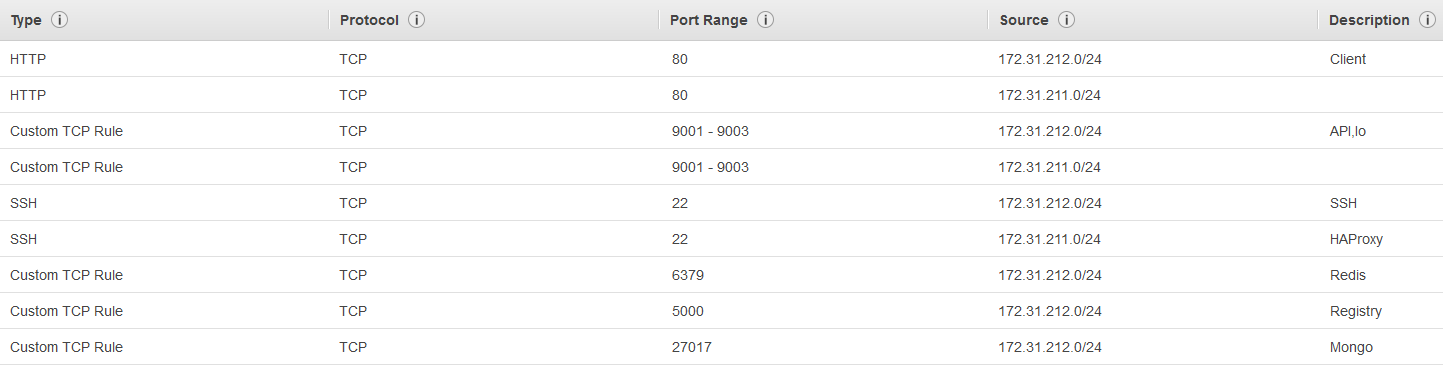
Kuvio 50. Stats -API

Nähdään, että molemmat API-koneet ovat käytössä, ja että kuorma jakaantuu niiden välillä.

## Tietoturva – Security groups

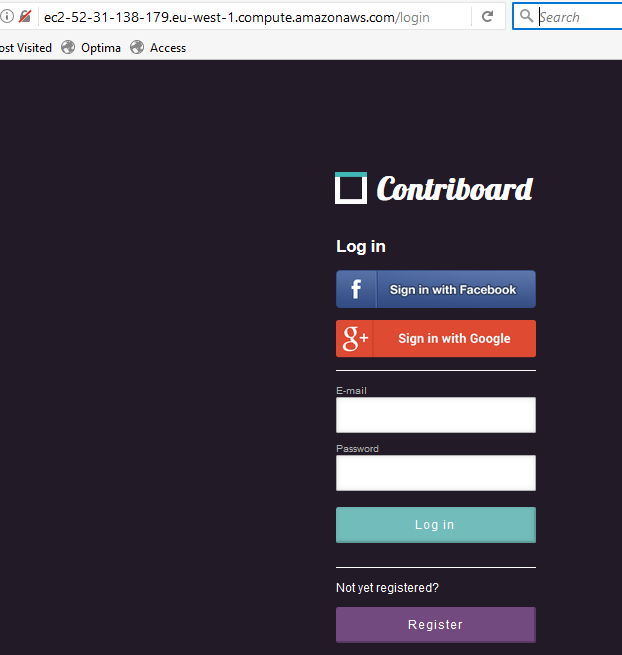
Muokataan security grouppien asetuksia siten, että vain tarvittava liikenne sallitaan ryhmämme subnetistä, poislukien HaProxy, jolle tulee sallia liikennettä myös muualta.

Luodaan siis seuraavanlaiset säännöt, jolla sallimme liikenteen vain sisäverkosta, ja vain tarvittaviin portteihin. (Ks. Kuvio 51.)



Kuvio 51. Security groups

Todennetaan vielä sääntöjen jälkeen, että yhteys toimii sisäverkossa(Ks. Kuvio 52.)



Kuvio 52. Yhteys contriboardiin

## Ssh hostnamella

Seuraavaksi lisättiin proxy koneelle avain, jonka avulla saatiin sshyhteys muihin koneisiin yksinkertaisesti antamalla käsky: '' ssh \*koneen nimi\* ''.

Ensiksi vaihdettiin docker koneiden hostnamet yksinkertaisimmiksi ''/etc/hosts'' tiedostossa. Tämän avulla hostien muistaminen on helpompaa. Lisäksi koneet nyt tunnistavat, että kyseiset hostit ovat olemassa. Nimetään koneet mahdollisimman yksinkertaisesti: Esim. API-kone = api , Client-kone = client ja niin edelleen.(Ks. Kuvio 53-54.)

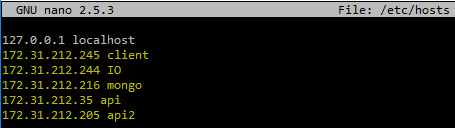


Kuvio 53. Hosts



Kuvio 54. Hostname

Lisättiin myöskin Haproxylle uudet hostnamet. (ks. Kuvio 55.)



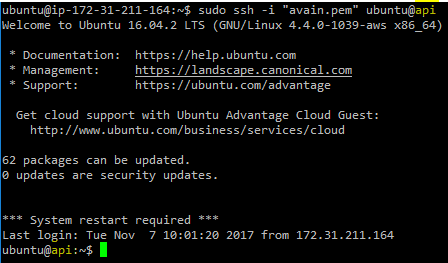
Kuvio 55. Hostname -HaProxy

Seuraavaksi käynnistettiin network.service uudestaan, jotta muutokset tulevat voimaan (Ks. Kuvio 56.)



Kuvio 56. Network restart

Todennettiin, että hostien nimet tulivat voimaan ottamalla yhteys määritettyihin hostnameihin(api). (Ks. Kuvio 57.)



Kuvio 57. SSH hostnamella

Seuraavaksi, jotta saatiin yhteys ilman, että avainta tulee erikseen määritellä, kopioitiin avain.pem tiedosto .ssh/ kansioon ja muutettiin sen nimeksi id\_rsa. (Ks. Kuvio 58.)



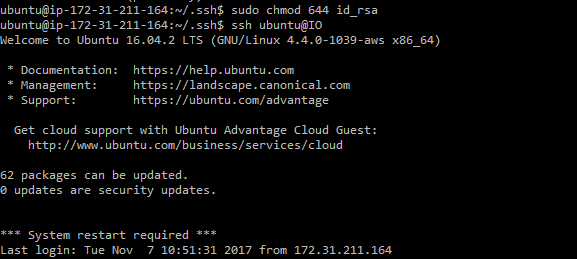
Kuvio 58. Avain.pem nimen muuttaminen

Tämän jälkeen ''Sudo cat id\_rsa >> authorized\_keys'' - komennolla lisättiin id\_rsa – avain tunnistettujen avainten joukkoon. ( ks. Kuvio 59.)



Kuvio 59. Id\_rsa -tunnistettu avain

Lisäksi muutettiin id\_rsa:n oikeudet jotta ssh yhteyttä luodessa avain voidaan lukea ja todentaa yhteys. Yhdistetään IO koneelle ja huomataan että yhdistäminen toimii. (Ks.Kuvio 60.)



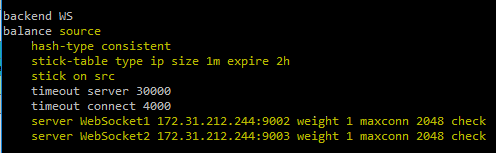
Kuvio 60. Id\_rsa oikeudet ja ssh yhteys

## Kuormantasaus IO-komponentille

Tässä osiossa tarkoituksena on asentaa jollekkin koneelle toinen IO-kontti ja suorittaa kuormantasaus näiden välillä. IO:lle juttelu tapahtuu websockettia, eli tilallista yhteyttä käyttäen. Käytännössä tämä toimii siten, että yhden asiakkaan kaikki yhteydet päätyvät samalle koneelle, mutta useamman asiakkaan yhteydet jakaantuvat eri koneiden välillä.

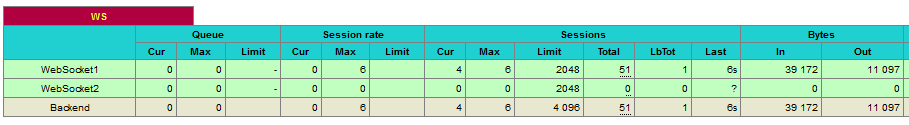
Muokataan siis taas HaProxyn conffi tiedostoa " /etc/haproxy/haproxy.cfg"

Backend WS kohtaan lisätään uuden IO-koneen osoite, sekä määritellään kuormantasauksen tyyppi. (ks. Kuvio 61.)



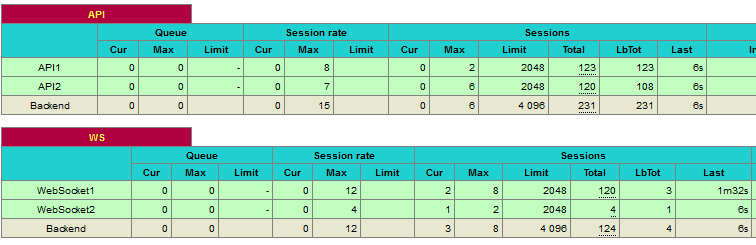
Kuvio 61. Backend WS määrittelyt

Todennetaan kuormantasausta: Ensiksi kaikki liikenne näkyy WebSocket1 koneella.(Ks.Kuvio 62.)



Kuvio 62. Kuormantasaus 1

Nyt kuitenkin, kun otamme yhteyden Contriboardiin toiselta koneelta, näemmä kuinka kuorma tasaantuu molempien WebSockettien välillä.(Ks.Kuvio 62.)



Kuvio 63. Kuormantasaus 2

## Backup suunnittelu Mongolle

Backup toteutetaan erilaisilla skripteillä ja erillisellä backup koneella, johon backupit säilötään. Skriptit ovat ohjelmoitu ajamaan seuraavia käskyjä. Ensimmäisellä skriptillä paikallisella koneella ajetaan *docker commit -p container-ID backup-name*. Käsky tekee kontista snapshotin ja tämä ajastetaan ajamaan itsensä kerrasta kahteen päivässä. Jotta snapshotin voisi ottaa tästä paikalliselta koneelta backup koneelle pitää siitä tehdä .tar tiedosto. Skripti tekee kontista .tar tiedoston seuraavalla käskyllä *docker save -o backup-name.tar backup-name*. Backup koneella oleva skripti hakee .tar backupit käyttäen *rsync* tai *scp* komentoa. Näin kontit ovat kahdennettuna, jolloin tietokanta kestää yhden docker -koneen hajoamisen. (Mathews R, 2016.)

## Pohdinta

Docker ja HaProxy olivat tuttuja labra 1 jäljiltä. Konttien järjesteleminen kone kohtaisesti (yksi kontti per kone) ei ollut vaikeaa. Myöskin API kontin kahdennus ja security grouppien määrittely onnistui lähes vaivattomasti. Ensimmäinen mietinnän paikka tulikin ssh-yhteyden luomisessa aliverkon koneille pelkällä hostnamella, mutta noin tunnin googlailun ja tutkiskelun jälkeen homma saatiin toimimaan. Seuraavan päänvaivan loi IO-kontin kahdennus; tilallisen yhteyden kahdennus oli uusi juttu kaikille (niin kuin pitikin), mutta onneksi Viinikanojan opinnäytetyöstä saatiin hyviä vinkkejä ja tehtävä saatiin suoritettua (pienten mutkien kautta). Viimeisimmäksi jätetty registry oli ehkä vaikein osa koko työtä. Suoraa ohjetta oli vaikea löytää mistään, mutta muutaman ohjeen ristiinlukeminen tuotti toivotun lopputuloksen. Samalla registryn luonnissa tuli kerrattua certtien luontia ja käyttöä. Kaiken kaikkiaan labrasta jäi hyvä maku suuhun ja työ saatiin valmiiksi.

# Lähteet

Amazon AWS. 2017. Amazon AWS nettisivu. Viitattu 16.10.2017 https://aws.amazon.com/what-is-aws/

Amazon Elastic Compute Cloud. 2017. Wikipedia –artikkeli Amazonin EC2 palvelusta. Viittattu 17.10.2017 https://en.wikipedia.org/wiki/Amazon\_Elastic\_Compute\_Cloud

Amazon Web Services. 2017. Wikipedia –artikkeli Amazonin pilvipalveluista. Viitattu 16.10.2017. https://en.wikipedia.org/wiki/Amazon\_Web\_Services

An Introduction to HAProxy and Load Balancing Concepts. 2014. DigitalOcean artikkeli HAProxysta ja kuormantasauksesta. Viitattu 17.10.2017 https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts

Docker overview. 2017. Docs Docker nettisivu. Viitattu 18.10.2017 https://docs.docker.com/engine/docker-overview/

Mathews R. 2016. Artikkeli Docker backupien teosta. Viitattu 20.11.2017 https://bobcares.com/blog/docker-backup/

Nelimarkka P. 2014. Artikkeli Contriboardista GitHub nettisivulla. Viitattu 18.10.2017 https://github.com/N4SJAMK/teamboard-meta/wiki/about-product