Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Eero Ketonen

OpenShift-klusterin asennus

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikka

Insinöörityö

22.02.2019

Tekijä Otsikko Sivumäärä Aika	Eero Ketonen OpenShift-klusterin asennus 33 sivua 22.02.2019
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	Tieto- ja viestintätekniikka
Ammatillinen pääaine	Tietoverkot
Ohjaajat	Lehtori Marko Uusitalo

Insinöörityön aiheena oli asentaa testiympäristökäyttöön Red Hatin OpenShift Container Platform. Asennuksen lisäksi tavoitteena oli julkaista testisovellus pyörimään sovelluskonteissa sekä tutkia ympäristön toimivuutta.

Työ aloitettiin käymällä läpi virtualisointia sekä konttivirtualisointia. Näiden teknologioiden pohjalta tutustuttiin Dockeriin ja Platform as a Service (PaaS) -alustoihin Kubernetekseen sekä OpenShiftiin. Työssä rakennettiin kolmen Linux-virtuaalikoneen klusteroitu OpenShift-ympäristö. Ympäristöön julkaistiin yksinkertainen verkkosovellus, joka oli ajossa useissa kloonatuissa sovelluskonteissa. Ympäristöä myös testattiin sekä tutkittiin yleisesti.

Työn tuloksena saatiin toimiva testiympäristö sekä pintaraapaisu siitä, mitä kaikkea OpenShift Container Platformilla voi tehdä. Työssä todettiin, että ympäristö on täydellinen omien sovelluksien testaamiseen sekä sovelluskonttiteknologiaan tutustumiseen. Mikäli haluttaisiin tuotantoympäristöön soveltuva klusteri, tulisi lisätä virtuaalikoneiden sekä fyysisien palvelimien määrää, jotta kaikki komponentit olisivat vähintään kahdennettu.

Avainsanat	OpenShift, Kubernetes, Docker

Author Title Number of Pages Date	Eero Ketonen OpenShift-cluster installation 33 pages 22 February 2019
Degree	Bachelor of Engineering
Degree Programme	Information and Communication Technology
Professional Major	Data Networks
Instructors	Marko Uusitalo, Senior Lecturer

The subject of this thesis was to install and configure Red Hat's OpenShift Container Platform. Second goal was to deploy a container-based application and investigate the functionality of the environment.

The thesis was started by explaining virtualization and container virtualization. Based on these technologies, Docker and Platform as a Service (PaaS) -based platforms Kubernetes and OpenShift were familiarized. OpenShift cluster of three Linux virtual computers were built and configured. After this, a simple web-based application was deployed to the environment. The application was running on multiple cloned containers. The environment was also tested and investigated generally.

As the result of this thesis, a working test environment was built. Also, it was realized that this was only a light scratch of the surface what you can do with OpenShift Container Platform. It was discovered that the environment that was built is perfect for testing and deploying applications and getting familiarized with container technology. To make a production environment grade cluster, more virtual machines and physical servers would be needed to exclude all single point of failures.

Keywords	Openshift, Kubernetes, Docker

Sisällys

Lyhenteet

API	Application Programmer Interface. Ohjelmointirajapinta.
CD	Continuous Delivery. Toimintatapa, jossa pyritään julkaisemaan ohjelmia lyhyin väliajoin.
CI	Continuous Integration. Toimintatapa, jossa ohjelmoijat yhdistävät koodinsa jaetussa versionhallintajärjestelmässä.
CLI	Command-line Interface. Komentorivi.
CPU	Central Processing Unit. Tietokoneen prosessori.
DHCP	Dynamic host configuration protocol. Protokolla IP-osoitteiden jakoon.
DNS	Domain Name Service. IP-osoitteen ja nimen yhdistäminen.
EPEL	Extra Packages for Enterprise Linux. Fedora Special Interest Groupin ylläpitämä pakettikirjasto Enterprise Linuxille.
GUI	Graphical User Interface. Graafinen käyttöliittymä.
HTTP	Hypertext Transfer Protocol. Protokolla WWW-palvelimille sekä selaimille.
HTTPS	Hypertext Transfer Protocol Secure. Transport Layer Security (TLS) protokollalla salattu HTTP liikenne.
LDAP	Lightweight Directory Access Protocol. Hakemistopalvelujen käyttöön tarkoitettu protokolla.
NAT	Network address translation. Yksityisen IP-osoitteen kääntö julkiseen IP-osoitteeseen.
OS	Operating System. Käyttöjärjestelmä.

PaaS	Platform as a Service. Palvelualusta sovelluksille.
RAM	Random Access Memory. Tietokoneen muisti.
RHEL	Red Hat Enterprise Linux. Unixin kaltainen käyttöjärjestelmä.
SDN	Software Defined Networking. Ohjelmisto-ohjatut verkot.
SSH	Secure Shell. Tietoliikenneprotokolla turvalliseen etäyhteyteen.
TLS	Transport Layer Security. Salausprotokolla, jolla suojataan Internet-sovellusten liikenne IP verkkojen yli. Aiemmin kulkenut nimellä Secure Sockets Layer (SSL)
vCPU	Virtual CPU. Virtuaalinen prosessori.
VNID	VXLAN Network Identifier. Teknologia, jolla voidaan tunnistaa ja erottaa loogisia verkkoja toisistaan.
VPN	Virtual Private Network. Salattu tunneli käyttäjän sekä kohteen välillä.

1 Johdanto

Virtualisointi on ollut pinnalla jo pitkään, mutta konttiteknologia on kasvanut räjähdysmäisesti viime vuosien aikana. Yrityksillä on nykyään ajossa tuhansia kontitettuja sovelluksia, todellisuudessa korkeiden määrien vuoksi niiden hallinta on hankalaa. Laajojen sovelluskonttiympäristöjen hallintaa sekä orkestraatiota varten on luotu useita työkaluja eri valmistajien toimesta.

Insinöörityön tarkoituksena oli tutustua sovelluskonttiteknologiaan sekä tarkemmin konttiympäristöjen hallintatyökaluun Red Hatin OpenShift Container Platformiin. Tavoitteena oli asentaa vähintään kolmen virtuaalikoneen OpenShift-klusteri sekä julkaista tähän ympäristöön jokin kontitettu sovellus pyörimään useisiin eri sovelluskontteihin.

Ennen asennusta tutustuttiin OpenShiftin alla pyöriviin teknologioihin, Linux Dockeriin sekä Googlen Kubernetekseen. OpenShift-ympäristössä Kubernetes hoitaa Docker-pohjaisten sovelluskonttien orkestrointia ja OpenShift tuo päälle käyttäjäystävällisen hallinnan sekä sovellusten skaalaamisen, julkaisun ja päivittämisen helppouden.

Asennusvaiheessa täytyi ottaa huomioon oman testiympäristön resurssit ja miettiä tarkkaan, kuinka iso OpenShift-ympäristö sinne mahtuu. Asennuksen jälkeen ympäristöön julkaistiin ajoon yksinkertainen kontitettu verkkosovellus. Jälkeenpäin tutkittiin myös ympäristön vikasietoisuutta ja sen mahdollisia parannuksia. Tarkoitus ei ollut rakentaa ympäristöä, joka sopisi yrityksen tuotantoympäristöksi.

2 Teoria

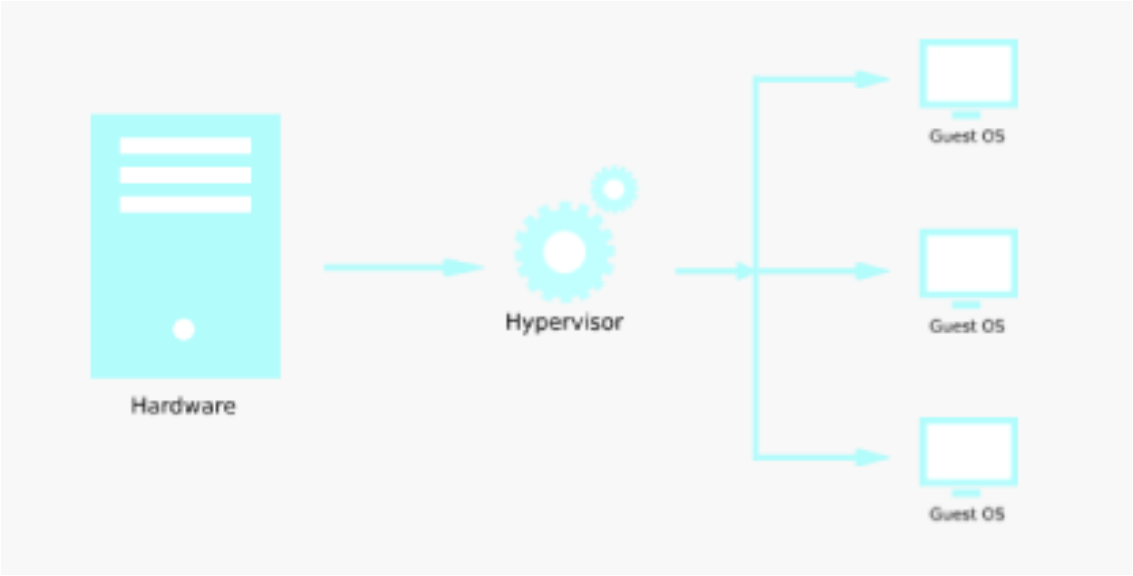
2.1 Virtualisointi yleisesti

Virtualisointi on laaja termi. Tietotekniikassa se tarkoittaa fyysisen resurssin jakamista loogisiksi resursseiksi. Teknologiana tämä tarkoittaa virtuaalisen kerroksen lisäämistä fyysisen laitteiston ja käyttöjärjestelmän väliin. Tämän virtuaalisen kerroksen avulla monta eri käyttöjärjestelmää voi pyöriä yhdellä fyysisellä koneella jakaen tämän koneen resursseja. [1, s. 2.]

2.2 Hypervisor

Hypervisor on käyttöjärjestelmän tavoin toimiva ohjelma, joka erottaa käyttöjärjestelmän ja fyysisen laitteiston. Se voidaan asentaa toisen käyttöjärjestelmän tai fyysisen laitteiston päälle. Tämä ohjelmisto jakaa fyysiset resurssit osiin ja sitä myötä virtuaalikoneille.

Tyypin 1 hypervisoria kutsutaan ”bare metal” -hypervisoriksi (kuva 1). Tämä asennetaan suoraan fyysisen laitteiston päälle eikä alla ole käyttöjärjestelmää. Tällä teknologialla on suora yhteys fyysisiin resursseihin, mikä tekee tyypin 1. hypervisorista tehokkaamman ja skaalautuvamman kuin tyypin 2 hypervisorista. [2, s. 4.]



Kuva 1. Tyypin 1 hypervisor

Tyypin 2 hypervisoria kutsutaan ”hosted”-hypervisoriksi (kuva 2). Tämä asennetaan ohjelmana käyttöjärjestelmän päälle ja turvautuu isäntäkäyttöjärjestelmän fyysisten resurssien hallintaan. [2, s. 4.]

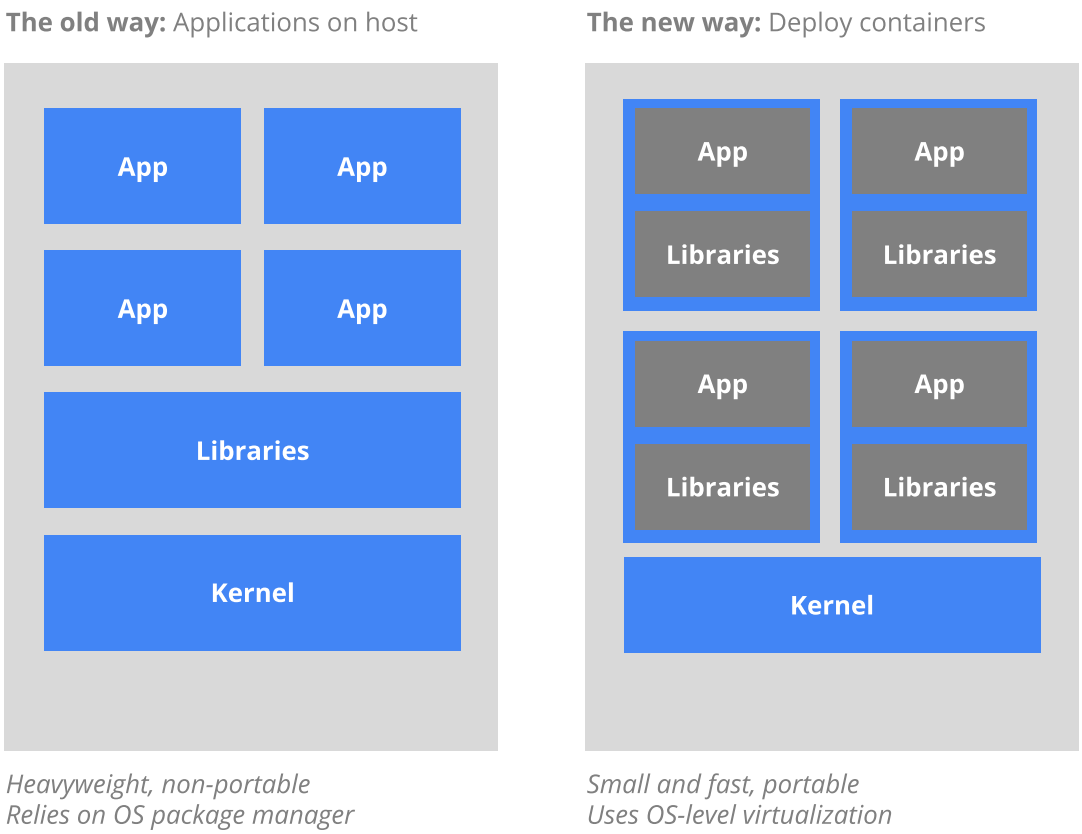
Kuva 2. Tyypin 2 hypervisor

2.3 Virtualisoinnin hyödyt

Virtualisoinnin tarkoituksena on optimoida fyysisten resurssien käyttöä. Ennen virtualisointia monet palvelimet konesaleissa käyttivät vain noin kymmenen prosenttia resursseistaan. Toisin sanoen yksittäisen palvelimen täydestä potentiaalista jäi 90 prosenttia käyttämättä. Virtualisointikerroksen avulla palvelimet pystyvät jakamaan resurssejaan monelle eri virtuaalikoneelle ja fyysisten palvelinten resurssienkäyttö saatiin nostettua 80-90 prosenttiin. Tämän ansiosta fyysisten palvelinten määrää saatiin vähennettyä, sähkönkulutus laski ja konesaleihin vapautui tilaa. [3, s. 4-8.]

2.4 Konttivirtualisointi

Konttivirtualisointi tarkoittaa sovelluskonttien virtualisointia käyttöjärjestelmän päällä. Kontti on kevyt ja eristetty ympäristö, joka sisältää ohjelman ja kaikki mitä ohjelma tarvitsee pyöriäkseen. Samaa sovelluskonttia voidaan ajaa eri käyttöjärjestelmillä, mikä tekee siitä käyttöjärjestelmäriippumattoman. Kontit jakavat isäntäkäyttöjärjestelmänsä ydintä, mutta kontin sisällä pyörivät prosessit ovat eristetty muusta ympäristöstä. Sovelluskonttien teknologia auttaa varmistamaan, että esimerkiksi ohjelmoijan tietokoneella luotu kontti toimii myös tuotantoympäristössä. [4.]



Kuva 3. Miksi konttivirtualisointi? [5]

Kuvan 3 vasemmalla puolella havainnollistetaan sovellukset käyttöjärjestelmän päällä ja oikealla puolella sovelluskontit. Käyttöjärjestelmän päällä olevat sovellukset ovat riippuvaisia isäntäkäyttöjärjestelmän kirjastoista ja paketeista. Tämä tekee ohjelmien siirtämisen vaikeaksi ympäristöstä toiseen. Oikealla puolella sovelluskontit sisältävät omat kirjastonsa ja käyttävät vain isäntänsä ydintä. Tämän teknologian ansiosta kontitettu sovellus voi pyöriä minkä käyttöjärjestelmän päällä tahansa ympäristömuutoksista huolimatta. [5.]

2.5 Klusteri

Klusteri tarkoittaa ryhmää koneita, jotka toimivat yhtenä koneena. Klusteri voi koostua esimerkiksi kolmesta tai kymmenestä koneesta. Kaikki koneet yhdistävät resurssinsa ja työskentelevät yhtenä koneena. Klusterin yksittäistä konetta kutsutaan nodeksi. Klustereita rakennetaan myös korkean käytettävyyden takia (high availability). Mikäli yksi kone

klusterista hajoaa, on klusterissa vielä toimivia koneita. Tämän teknologian avulla häiriöaikaa ei synny yhden noden rikkoutumisesta. [6.]

2.6 Docker

Docker on sovelluskonttiteknologian päälle luotu sovellus, joka julkaistiin avoimen lähdekoodin ohjelmana vuonna 2013 dotCloud-yrityksen toimesta. Docker luottaa Linux-käyttöjärjestelmän ominaisuuksiin, kuten nimiavaruuksiin ja kontrolliryhmiin, joilla varmistetaan resurssien eristys ja ohjelman pakkaaminen kaikkien osien kanssa, mistä ohjelma on riippuvainen. [7.]

Dockerin julkaisun jälkeen tuhannet ohjelmistokehittäjät huomasivat sovelluksen potentiaalin ja alkoivat käyttää sitä. Ensimmäisen vuoden sisällä yritykset kuten Red Hat ja Amazon aloittivat kaupallisen tuen Dockerille, vaikka Dockerin hallitus varoitti tuotteen käyttämistä tuotantoympäristössä. Kun Docker julkaisi version 1.0 kesäkuussa 2014, oli sovellusta ladattu jo 2,75 miljoonaa kertaa. [7.]

Terminologiaa

Container image tarkoittaa pakettia, joka sisältää kaiken koodin ja tiedon mitä tarvitaan sovelluskontin luomiseen. Kun container image on luotu, sitä ei voi enää muokata.

Container tarkoittaa instanssia, joka on luotu container imagesta. Tämä instanssi kuvaa yhden ohjelman, prosessin tai palvelun suorittamista. Container sisältää container imagen sisällön, suoritusympäristön sekä muita komentoja. Palvelun, prosessin tai ohjelman skaalaamisessa voidaan samasta container imagesta luoda useita instansseja. [8.]

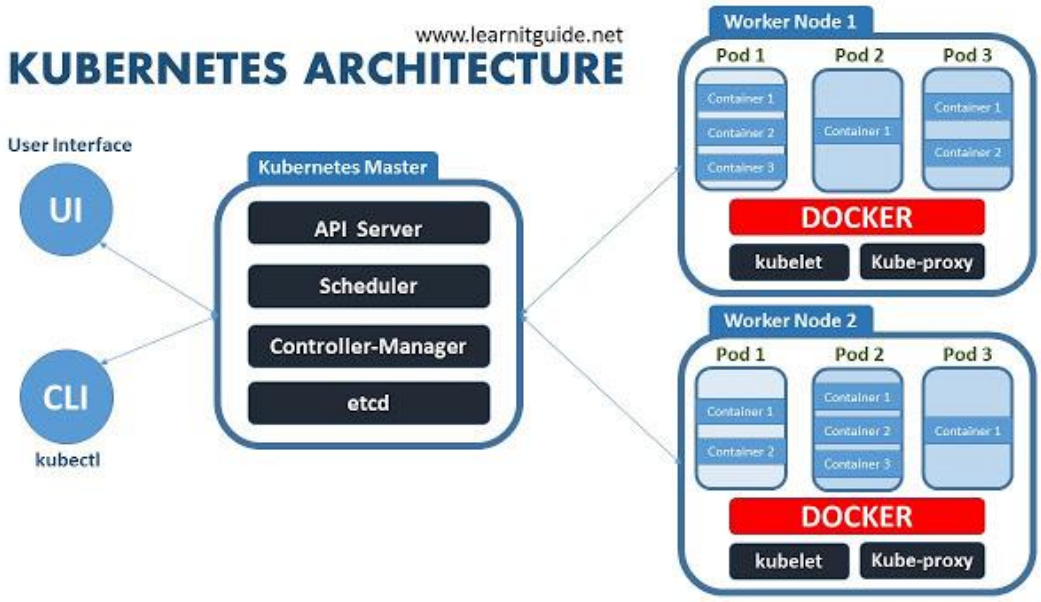
Dockerfile on tekstitiedosto, joka sisältää ohjeet ja komennot docker imagen luomiselle. Se on vähän kuin skripti: ensimmäinen rivi kertoo pohjalevykuvan. Tämän jälkeen tiedostossa on määritetty komennot esimerkiksi tarvittavien ohjelmien asentamiseen ja tiedostojen kopiointiin. [8.]

2.7 Kubernetes

Kubernetes on avoimen lähdekoodin projekti, jonka Google julkaisi vuonna 2014. Kubernetes on tarkoitettu konttipohjaisten sovellusten ja palvelujen hallintaan. Kubernetes tarjoaa konttikeskeisen hallintaympäristön ja hoitaa käyttäjän puolesta konttiorkestroinnin, reitityksen ja tietojenkäsittelyn. [5.]

Kubernetes ei kuitenkaan ole perinteinen ja kaikenkattava Platform as a Service -alusta. Kubernetes operoi konttitasolla eikä laitteistotasolla. Se tarjoaa PaaS-alustalle yleisiä ominaisuuksia, kuten käyttöönottoa, skaalausta, kuormanjakoa, lokitusta ja valvontaa. Kuitenkaan Kubernetes ei ole yhtenäinen, vaan nämä oletusratkaisut ovat valinnaisia ja alustaan lisättäviä. [5.]

Komponentit ja terminologiaa



Kuva 4. Kuberneteksen arkkitehtuuri [9].

Kubernetes-klusteri tarvitsee monia eri komponentteja toimiakseen. Jos rakennetaan esimerkiksi kolmen koneen klusteri, yksi näistä on master-node, loput ovat nodeja.

Kuberneteksen pod on pienin mahdollinen ohjattava ja hallittava objekti. Yksi pod sisältää yhden tai useamman containerin. Jokaiselle podille Kubernetes määrittää oman IP-osoitteensa Kuberneteksen sisäverkosta. Podeja ei ole suunniteltu kestäviksi. Ne eivät kestä aikataulutusvirheitä, nodejen hajoamista tai resurssien loppumista. Jos pod hajoaa, ei sitä yritetä korjata vaan uusi luodaan tilalle. [10.]

Kubelet on node-palvelimilla pyörivä palvelu, joka kommunikoi masterin komponenttien kanssa. Kubelet valvoo omalla nodellaan olevia podeja ja tarvittaessa tuhoaa ja luo uusia. Se ottaa vastaan komentoja ja työtehtäviä ja toteuttaa nämä. Kubelet vastaanottaa manifestin muodossa tehtävät sekä työkuormat. Kun kaikki on tehty, kubelet jää valvomaan nykyistä tilaa. Mikäli jokin poikkeaa sille ilmoitetusta tilasta, kubelet korjaa tilanteen. [11.]

Masterin komponentit luovat yhdessä klusterin ohjaustason (control plane). Master tekee klusterissa kaikki päätökset, valvoo ja reagoi klusterin tapahtumiin. Esimerkiksi jos yksi podi lakkaa toimimasta, uuden luomisesta vastaavat master-nodella pyörivät komponentit.

Kube-apiserver on Masterilla pyörivä komponentti, joka tarjoaa Kuberneteksen Application Programming Interface (API) -ohjelmointirajapinnan. Tämä palvelu toimii frontendinä ohjaustasolle eli käyttäjälle näkyvänä hallintasovelluksena. Kube-scheduler seuraa juuri luotuja podeja ja siirtää nämä päättämälleen nodelle.

Etcd on tallennuspalvelu, johon tallennetaan klusterin kriittinen data, esimerkiksi konfiguraatio. Etcd on hajautettu avainarvopalvelu, joka nopeasti ja luotettavasti tallentaa kriittiset tiedostot.

Kube-controller-manager on komponentti, joka pyörii kontrollereilla. Kontrollereita on seuraavanlaisia: Node Controller, joka on vastuussa nodejen valvonnasta, Replication Controller pitää huolen, että podeja on oikea määrä, Service Account & Token Controller luo oletuskäyttäjiä ja ohjelmointirajapintaan pääsyvaltuuksia uusille nimiryhmille. [12.]

2.8 Ansible

Ansible on työkalu, jolla tehdään konfiguraatiohallintaa sekä automaatiota eri ympäristöissä. Ansiblen julkaisi Michael DeHaan vuonna 2012 avoimen lähdekoodin projektina GitHubiin. Lyhyessä ajassa Ansiblen lähdekoodi sai yli 17 000 tähteä sekä yli 1700 uniikkia myötävaikuttajaa GitHubissa [13, s. 2]. Ansiblen kaltaisia konfiguraatiohallintatyökaluja ovat muun muassa Puppet tai Chef. Näistä Ansible erottuu yksinkertaisuudensa vuoksi. Ansible ei tarvitse omaa sovellusagenttiaan etähallittavilla koneilla vaan käyttää sen sijaan sopivaa protokollaa päästäkseen muihin koneisiin, esimerkiksi Secure Shelliä (SSH), jolla voidaan muodostaa salattu etäyhteys. Lisäksi Ansiblelle täytyy määrittää oikeudet. [14.]

Playbookit ja Inventory

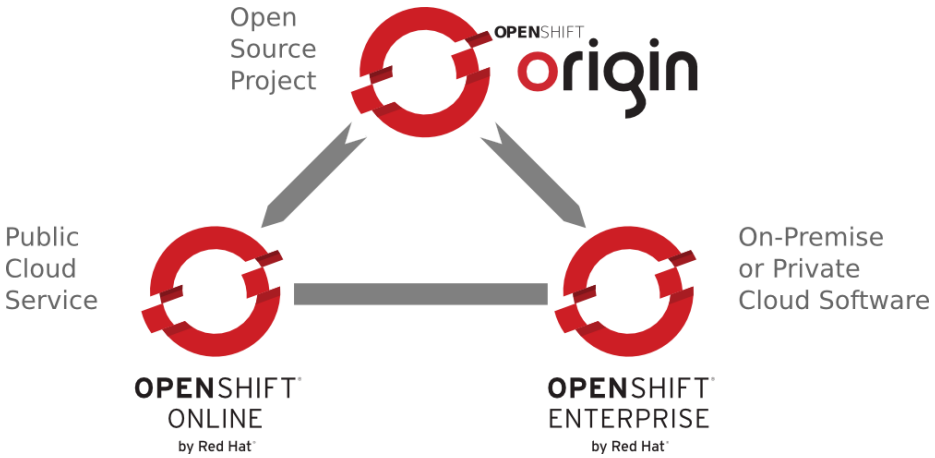
Playbookit ovat YAML-tiedostoja, joita Ansible käyttää ohjeina. YAML on merkintäkieli, jota käytetään usein konfiguraatiotiedostoissa. Yksinkertaisesti playbookit ovat vain rivejä komentoja sekä argumentteja, jotka kaikki suoritetaan synkronoidusti tai ei-synkronoidusti.

Konfiguraatiohallinnassa käytettävä ohjelma tarvitsee ohjeet, millä kaikilla laitteilla täytyy tehdä muutoksia. Tätä kutsutaan inventoryksi. Inventory-tiedosto sisältää kaikki laitteet, joille Ansiblen on tarkoitus tehdä muutoksia. Oletuksena inventory-tiedosto sijaitsee /etc/hosts/-hakemistossa. Tätä ei kuitenkaan ole suositeltavaa käyttää, vaan jokaiselle projektille kannattaa luoda oma inventory-tiedostonsa. [14.]

2.9 OpenShift

OpenShift on Red Hatin kehittämä PaaS-alusta sovelluskonttien hallintaan, luomiseen ja käyttöönottoon. Red Hat tarjoaa erilaisia vaihtoehtoja alustastaan yrityksille. OpenShift

Container Platform on yrityksen toimitiloissa (on-premises) pyörivä alusta, OpenShift Online on julkisessa pilvipalvelussa pyörivä alusta, OpenShift Dedicated on Red Hatin ylläpitämä alusta julkisessa pilvipalvelussa ja OpenShift Origin (nykyisin OKD) on avoimen lähdekoodin versio alustasta, jota yhteisö ylläpitää. Kaikki lähdekoodi on saatavilla GitHubista Apache 2 -lisenssin alla. OpenShiftin Online ja Container Platform pohjautuvat avoimen lähdekoodin versiosta Origin. [15.] [16.]

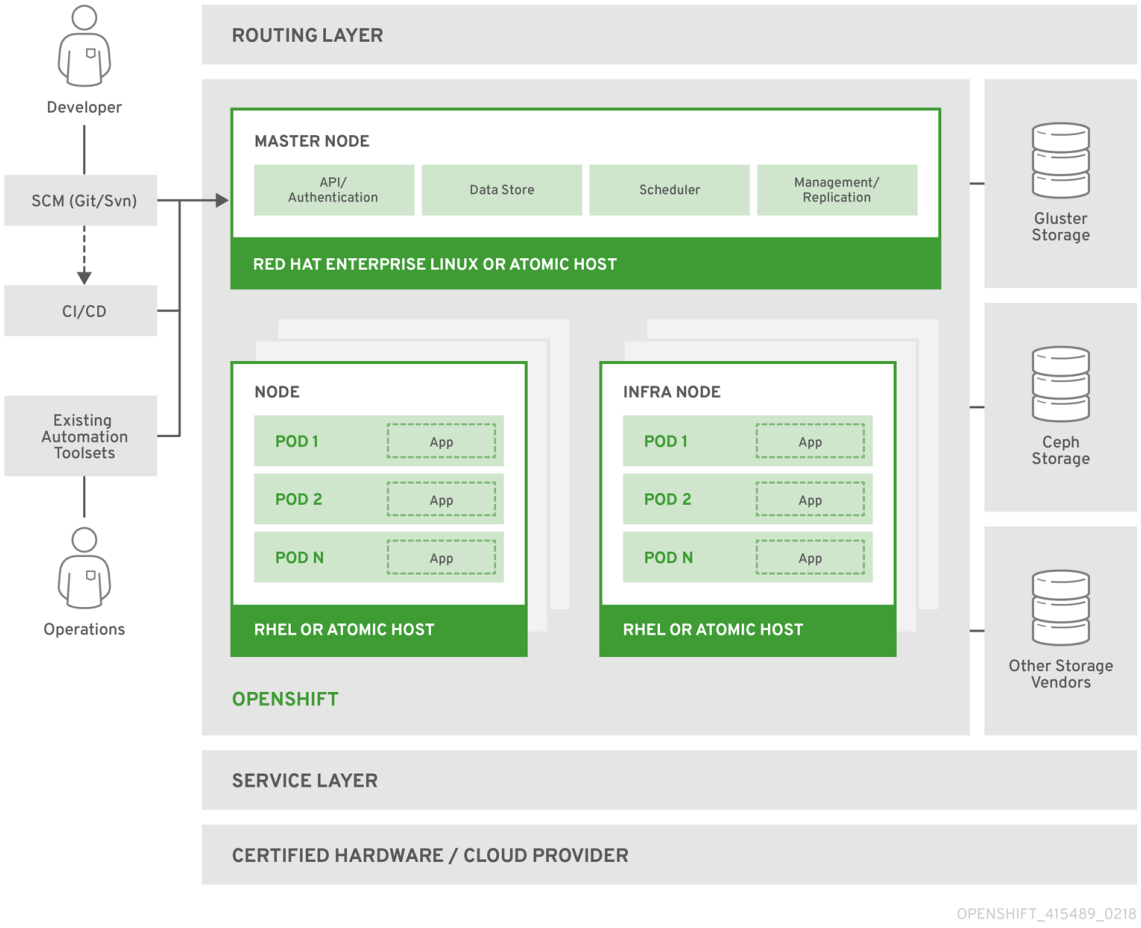


Kuva 5. OpenShiftin versiot [16].

OpenShiftissä sovellustasolla Docker vastaa sovelluskonteista ja Kubernetes hoitaa sisäisesti konttien orkestroinnin sekä hallinnan. OpenShift tekee vaivattomaksi julkaista uusia ohjelmia konttipohjaiseen ympäristöön sekä päivittää ohjelmistokoodia ilman käyttökatkoja. Tämä helpottaa yritysten kehitystiimien työtä tekemällä sovellussuunnittelun ja testauksen vaivattomaksi sekä nopeaksi. OpenShiftiä voi hallita selainpohjaiselta hallintasivulta sekä Command-line Interfacelta (CLI) eli käyttöjärjestelmän komentoriviltä. [17.]

2.9.1 Arkkitehtuuri

Koska OpenShiftin sisällä toimii Kubernetes, on OpenShiftillä samantapainen arkkitehtuuri sekä komponentit. Masterilla pyörivät omat komponentit sekä palvelut ja nodet toimivat klusterin työntekijöinä, joita master ohjaa. Masterin komponentit pyörivät masternodella staattisina podeina.



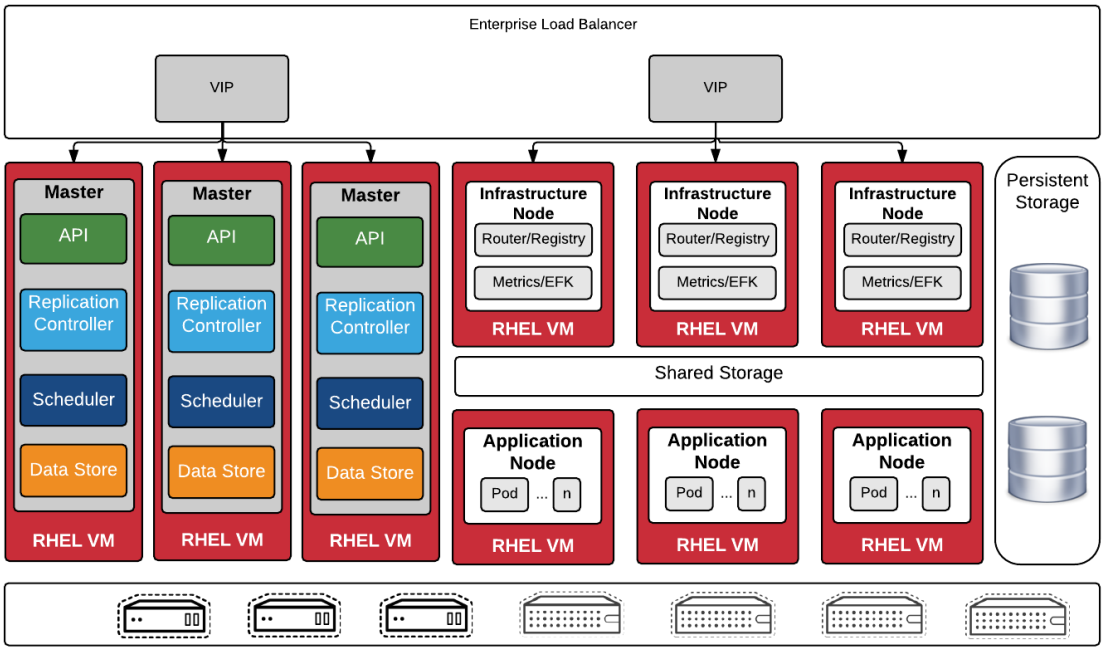
Kuva 6. OpenShiftin yleiskatsaus

Kuvassa 5 havainnollistetaan OpenShiftin arkkitehtuuri. Ylempänä näkyy master, alla näkyy sovellusnode ja infrastruktuurinode. Infrastruktuurinode on tarkoitettu kaikille palveluille, jotka ympäristön omistaja luokittelee infrastruktuuriin, esimerkiksi HAProxy tai OpenShiftin reititin. sovellusnodet on tarkoitettu vastaanottamaan kaikki työkuorma sovelluskonteista.

Arkkitehtuuristen vaihtoehtojen kohdalla täytyy tietää, millaiseen käyttöön oma ympäristö tulee. Halutaanko ympäristö julkiseen pilvipalveluun vai yrityksen omiin tiloihin? Kuinka monta podia ympäristössä tarvitsee pyöriä? Kuinka paljon kuormaa oletettuun ympäristöön kohdistuu? [18.]

Pienin mahdollinen ympäristö on ”all in one” -ympäristö, jossa kaikki komponentit ovat yhdellä isäntäkoneella, eli master ja node ovat samassa. Tätä ympäristöä ei käytetä tuotantoympäristönä, vaan se sopii pieneksi testiympäristöksi. Masterien ja nodejen lukumäärät ovat miltei vapaavalintaisia. OpenShiftin eri versiot tukevat tiettyyn pisteeseen asti, esimerkiksi 3.9.0 versio tukee 2000 nodeen ja 120 000 podiin asti. [19.]

Esimerkki tuotantoon soveltuvasta ympäristöstä löytyi contained.io-sivustolta. Tämä sivusto on Red Hat -yhteisön perustama projekti, jolla on tarkoitus jakaa tietämystä OpenShiftistä kiinnostuneille.



Kuva 7. v1.contained.io-sivuston esimerkkiarkkitehtuuri [20].

Ympäristön yhdeksän konetta ovat Red Hat Enterprise Linux -käyttöjärjestelmällä varustettuja virtuaalikoneita. Näistä kolme on mastereita ja loput kuusi nodeja. Nodet jaetaan kahtia ja saadaan kolme infrastruktuurinodea ja kolme sovellusnodea. Infrastruktuurinodeilla pyörii levykuvarekisteri sekä OpenShiftin reititin. Kolme sovellusnodea ovat koneita, joilla pyörii alustalla ajetut sovellukset. [20.]

Tuotantoympäristöön on suositeltavaa lisätä erillinen yrityskäyttöön suunniteltu kuormanjakaja. Suosituimpia kuormanjakajia ovat F5 Networksin BIG IP -kuormanjakajat, jota myös tässä ympäristössä on käytetty. F5:lle on määritetty kaksi eri virtuaalista IP-osoitetta (VIP), joiden alle määritetään palvelinten IP-osoitteet ja kaikki liikenne, mikä virtuaaliseen IP-osoitteeseen tulee, jaetaan tasaisesti sen alla oleville palvelimille. Tässä ympäristössä toisen VIP:n alle on määritetty kolme master-konetta ja toisen alle kolme infrastruktuurinodea. [20.]

Kubernetes jakaa jokaiselle podille oman IP-osoitteensa, jonka avulla podit pystyvät keskustelemaan toistensa kanssa Kuberneteksen sisäverkossa. Tämän ansiosta podeja voidaan kohdella samalla tavalla kuin virtuaalikoneita, esimerkiksi nimeämisessä, kuormanjaossa, migraatiossa tai porttiallokoinnissa. [21.]

OpenShift tarvitsee kaksi sisäistä verkkoa toimiakseen. Toisesta verkosta, esimerkiksi 10.128.0.0/14, klusteri jakaa osoitteet podeille. Tämä verkko on oletuksena käytössä ja siihen mahtuu 262 142 host-osoitetta. Toisen verkon OpenShift tarvitsee klusterin palveluille. Näille annetaan oletuksena IP-osoitteet verkosta 172.30.0.0/16, johon mahtuu 65 534 osoitetta. Nämä verkot voi myös päättää itse asennusvaiheessa. [21.]

2.9.2 OpenShift SDN

OpenShift tarjoaa kolmea erilaista vaihtoehtoa verkon rakenteelle, joista kaikki ovat Software Defined Networking (SDN) -tyypin ratkaisuja sekä pohjautuvat OVS:ään (Open vSwitch). Alkuperäisistä vaihtoehdoista *ovs-subnet* on yksinkertaisin. Yksi verkko, jossa kaikki podit ja palvelut voivat keskustella kaikkien kanssa. Tässä insinöörityössä tehdyssä ympäristössä käytetään vaihtoehtoa *ovs-multitenant*, joka tarjoaa projektikohtaista eristystä podeille ja palveluille. Jokaisella projektilla on oma VXLAN Network Identifier

(VNID), jolla tunnistetaan liikenne podien ja palveluiden välillä. VNID:t ovat 24-bittisiä ja loogisia eri verkkoja voi olla yhteensä yli 16 miljoonaa. Tämä eroaa Virtual Local Area Network (VLAN):sta siten, että eri VLAN:eja voi olla maksimissaan hieman yli 4000. Tämä teknologian avulla podit eivät voi kommunikoida eri projekteihin kuuluvien podien kanssa. Kolmas vaihtoehto on *ovs-networkpolicy*, jolla projektin ylläpitäjät voivat itse konfiguroida eristyspolitiikat NetworkPolicy-objektien avulla. [22.]

OpenShift SDN luo jokaiselle nodelle kolme verkkokorttia. Jotta podit voivat saada oman IP-osoitteensa, nodelle tehdään sillattu verkkokortti *br0* (OVS bridge). Kaikki nodella pyörivät sovelluskontit saavat tämän kautta IP-osoitteensa. Tun0 on OVS:n sisäinen portti, jonka avulla voidaan liikennöidä klusterin verkosta ulkoverkkoihin. Tun0-verkkokortille määritetään klusterin verkon oletusyhdyskäytävä (default gateway). Lopuksi OpenShift SDN konfiguroi reitityssääntöjä ja sallii pääsyn klusterin verkosta ulkoverkkoon Network Address Translationin (NAT) avulla, joka kääntää yksityisen IP-osoitteen julkiseksi IP-osoitteeksi. Jokainen kerta, kun uusi pod ilmestyy nodelle, tapahtuu seuraavat:

1. Pod saa IP-osoitteensa klusterille allokoidusta verkosta.

2. Pod kiinnittää *veth*-verkkokorttinsa noden OVS bridge *br0*-verkkokorttiin.

3. OpenShift SDN lisää säännön OVS-tietokantaan, jossa määritetään reititys uudesta podista oikeaan OVS-porttiin. [22.]

2.9.3 Tietoturva

Red Hat käyttää lukuisia teknologioita pitääkseen OpenShiftin tietoturvallisena ratkaisuna. Jotta käyttöjärjestelmä pystyy suojaamaan sovelluskontit, pitää käyttöjärjestelmän pystyä suojaamaan isäntänsä ydin. Kontit ovat vain tietyntyyppisiä prosesseja ja ne suojataan samalla tavalla kuin mikä tahansa muu prosessi. SELinux (Security-Enhanced Linux) tarjoaa ylimääräisen turvakerroksen, joka eristää kontit toisistaan sekä isännäs-

tään. SELinux mahdollistaa tarkemman järjestelmän valvonnan. Nimiavaruudet (namespace) sekä kontrolliryhmät (CGroups) mahdollistavat ja eristävät konttien resurssienkäytön. [23.]

Kaikki kommunikointi ja liikenne API-rajapinnan kanssa sekä master-komponenttien välillä suojataan Transport Layer Security (TLS) -protokollalla. TLS tarjoaa vahvan salauksen nykypäivän standardien mukaisesti. OpenShiftillä on oletuksena käytössä versio TLS 1.2, mutta muiden versioiden käyttäminen on myös mahdollista. Mikäli käytetään OpenShiftin valmista Ansible-asennuspakettia, se luo automaattisesti sisäisen Public Key Infrastructuren (PKI). Sisäinen PKI käyttää 2048-bittisiä RSA-avaimia sekä SHA256-signeerauksia. Nämä kaikki ovat self-signed-sertifikaatteja eivätkä ole Certificate Authority (CA) hyväksymiä. Omat mukautetut sertifikaatit ovat myös tuettuja OpenShiftissä. [24, s. 9-10.]

OpenShiftissä autentikointitapoja on monia. Esimerkiksi yksinkertainen paikallisesti toteutettu autentikointi htpasswd-työkalulla. Master-koneelle luodaan paikallisesti .htpasswd-tiedosto ja lisätään tähän tiedostoon polku OpenShiftin konfiguraatioon. Htpasswd-tiedosto sisältää käyttäjätunnukset ja salasanat. Toinen esimerkkitapa on käyttää ulkoista Lightweight Directory Access Protocol (LDAP) -palvelinta. LDAP-palvelimen yleisin käyttötapa on käyttäjätunnusten sekä oikeuksien tarkistaminen. Nämä molemmat vaihtoehdot ovat yksinkertaisesti konfiguroitavissa master-config.yaml-tiedoston identityProviders-osion alle. Alla olevassa esimerkissä käytetty htpasswd-vaihtoehtoa. [25.]

```
identityProviders:
- challenge: true
  login: true
  mappingMethod: claim
  name: htpasswd_auth
  provider:
    apiVersion: v1
    file: /etc/origin/master/htpasswd
    kind: HTPasswdPasswordIdentityProvider
```

2.10 Muiden valmistajien vaihtoehdot

Vaihtoehtoja OpenShiftille löytyy runsaasti muilta valmistajilta. Nykypäivän suurimmat pilvipalveluyritykset kuten Amazon Web Services (AWS) ja Microsoft Azure sekä pääasiassa verkkolaitteita valmistava Cisco Systems tarjoavat sovelluskonttipohjautuvia hallinta-alustoja.

Cisco Container Platform on OpenShiftin kaltaisesti Kuberneteksen teknologian kanssa rakennettu hallintatyökalu, jonka tarkoitus on helpottaa Kubernetes-klustereiden rakentamista, sovellusten julkaisua sekä niiden päivittämistä ilman käyttökatkoja. Cisco Container Platform ja OpenShift Container Platform ovat Continuous Integration / Continuous Delivery (CI/CD) -tyypin ratkaisuja. Cisco Container Platformilla voidaan ohjata ja valvoa lukuisia Kubernetes-klustereita samanaikaisesti esimerkiksi eri palveluntarjoajien ympäristöissä. [26.]

Azure Kubernetes Service (AKS) on Microsoft Azure -pilvipalvelussa sijaitseva työkalu, jolla on helpotettu Kubernetes-klusterin luonti Azure-pilvessä. AKS myös hallitsee Kubernetes-ympäristöä ja tekee sovellusten julkaisun ja hallinnan helpoksi ilman konttiorkestraatio-osaamista. AKS tarjoaa ylläpitoa sekä automaattista valvontaa kriittisten palvelujen suhteen. AKS:ssä erilaista muihin verrattuna on se, että Azure pitää huolta klusterin master-nodeista. Loppukäyttäjän tarvitsee vain huolehtia sovellusnodeista ja niillä pyörivistä omista palveluistaan. AKS:ssä master-koneista ei tarvitse maksaa Azurelle, ainoastaan sovellusnodeista peritään maksu. [27.]

Amazon Elastic Container Service (Amazon ECS) on Amazonin kehittämä konttiorkestraatioalusta kontitettujen sovellusten julkaisuun, hallintaan sekä skaalaukseen [28]. Amazon ECS -alustaan on mahdollista lisätä Amazon Fargate, mikä hallitsee klusterin koneita loppukäyttäjän puolesta. Amazon ECS:stä on kaksi käynnistystyyppiä: Fargate tai EC2. Fargate-käynnistystyypin avulla loppukäyttäjän ei tarvitse tehdä mitään muuta kuin luoda sovelluskontti, määrittää CPU ja muistivaatimukset sekä verkkoasetukset. Fargate hoitaa itse klusterin ja sen koneet. EC2-käynnistystyypillä loppukäyttäjä voi hallita klusterin koneita. [29.]

3 Klusterin asennus

Tässä työssä on käytetty OpenShiftin versiota 3.9.0. Asennus on toteutettu hyödyntämällä konfiguraatiohallintatyökalua Ansiblea.

Ympäristönä toimii varsin yksinkertainen VMware-ympäristö VMware Inc. on yritys, joka tarjoaa virtualisointiratkaisuja. HP ProLiant DL360 -palvelimelle asennettu VMwaren tyypin 1 hypervisor ESXi 6.5 Enterprise. Palvelimella on kokonaismuistia 48 GB ja suoritintehoa 19,2 GHz. Tämä palvelin sijaitsee reitittimen takana ainoana fyysisenä sisäverkon laitteena.

Tässä työssä oikea domain on peitetty esimerkkidomainilla example.com. Myös julkinen IP-osoite on keksitty. Domain example.com on hankittu Domain Name Service (DNS) -palveluntarjoajalta. Tämän domainin DNS-sääntöihin on luotu seuraavat säännöt:

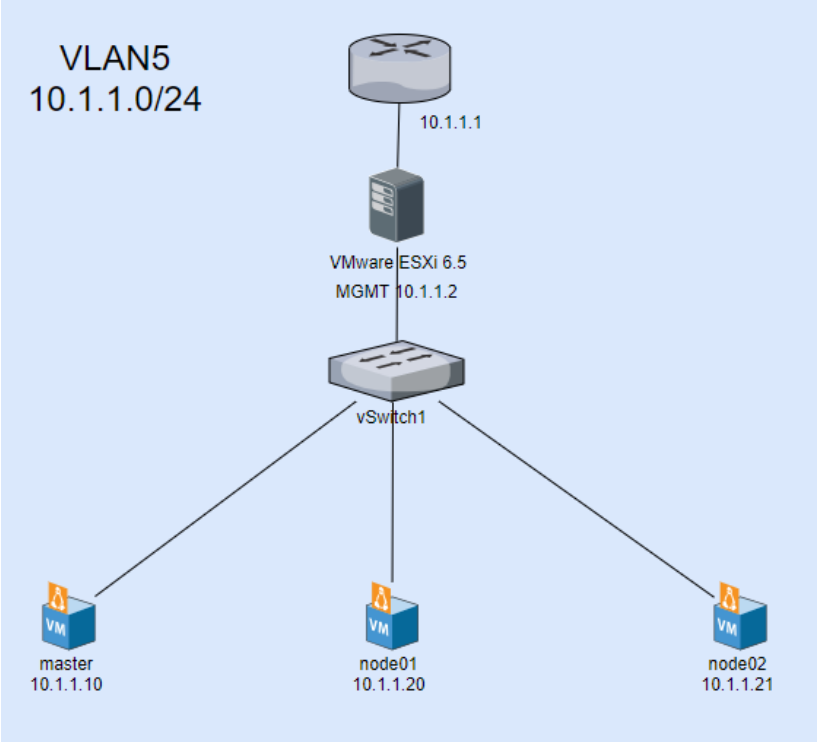
```
* 420 IN CNAME apps.console
@ 10800 IN A 88.241.220.10
apps.console 10800 IN A 88.241.220.10
console 10800 IN A 88.241.220.10
node01 10800 IN A 88.241.220.10
node02 10800 IN A 88.241.220.10
```

Säännöillä on osoitettu example.com-domainin alidomainit console, node01, node02, apps.console sekä koko apps.console.example.com-alidomainin juuri kotiympäristön julkiseen IP-osoitteeseen 88.241.220.10.

Pääsy ESXi:n hallintaan sekä virtuaalikoneille on rajoitettu ulkoverkosta. Reitittimelle on asennettu OpenVPN-työkalu, jonka kautta voidaan muodostaa salattu yhteys käyttäjän ja reitittimen välille. Kun tunneli on muodostettu, saadaan yhteys kaikkiin virtuaalikoneisiin ja palvelimeen. Toisien sanoen, palvelimia voidaan operoida ainoastaan sisäverkosta. Ainoa pääsy ulkoverkosta ympäristöön on alidomainin console.example.com, josta pääsee OpenShiftin selainpohjaiseen hallintaan.

ESXi:llä virtualisoituja koneita on useampia, OpenShift-klusteriin liittyviä on kolme. Jokainen virtuaalikone on samasta sisäverkosta, 10.1.1.0/24, missä oletusyhdyskäytävä

on 10.1.1.1. Dynamic Host Configuration Protocol (DHCP) -osoiteavaruuteen on määritetty tästä verkosta osoitteet 10.1.1.30 – 10.1.1.254.



Kuva 8. Testiympäristö

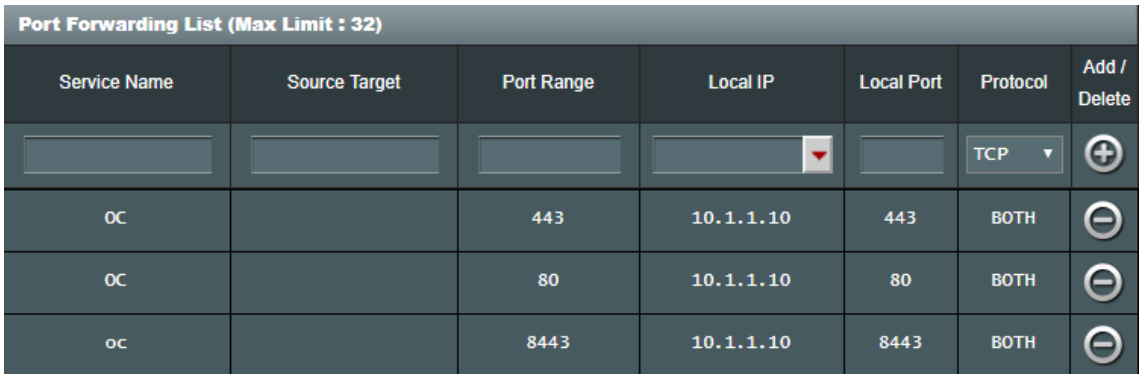
3.1 Klusterin koneet

OpenShift Origin on toteutettu kolmella CentOS 7 x86 64-bittisellä virtuaalikoneella. Yksi näistä on master ja loput kaksi nodeja. Masterille määritetty staattinen IP-osoite 10.1.1.10, node01:lle 10.1.1.20 ja node02:lle 10.1.1.21.

Minimivaatimukset masterille ovat 16GB RAM, 4vCPU ja 40 GB:n kiintolevy. Nodeille minimivaatimukset ovat 8GB RAM, 1vCPU ja 15 GB:n levy. Masterille on asetettu minimivaatimusten mukaiset resurssit sekä molemmilla nodeilla on 10GB RAM, 2vCPU ja 30 GB:n levyt.

3.2 Klusterin valmistelu

Jotta OpenShiftin selainpohjaiseen hallintaan pääsee ulkoverkosta, täytyy reitittimellä tehdä tarvittavat porttien uudelleenohjaukset. OpenShift tarvitsee portit 443 (HTTPS), 80 (HTTP) ja 8443 (HTTPS). Nämä portit uudelleenohjataan ulkoverkosta masterin yksityiseen IP-osoitteeseen 10.1.1.10, joka tarkoittaa käytännössä sitä, että kaikki liikenne mikä tulee portteihin 80,443 ja 8443 ohjataan osoitteeseen 10.1.1.10.



Kuva 9. Porttiuudelleenohjaukset reitittimellä

Koska testiympäristössä ei ole käytössä erillistä DNS-palvelinta, pitää kaikkien kolmen koneen /etc/hosts -tiedostosta löytyä seuraavat rivit. Näiden avulla koneet osaavat yhdistää koneiden nimet IP-osoitteisiin.

```
[root@master ~]# cat /etc/hosts
10.1.1.10 master console console.example.com
10.1.1.20 node01 node01.example.com
10.1.1.21 node02 node02.example.com

root@node01 ~]# cat /etc/hosts
10.1.1.10 master console console.example.com
10.1.1.20 node01 node01.example.com
10.1.1.21 node02 node02.example.com

[root@node02 ~]# cat /etc/hosts
10.1.1.10 master console console.example.com
10.1.1.20 node01 node01.example.com
10.1.1.21 node02 node02.example.com
```

Kaikilla kolmella CentOS-virtuaalikoneella täytyy ensiksi päivittää nykyiset asennetut pakettikirjastot. Tämän jälkeen käynnistetään uudelleen jokainen virtuaalikone.

```
yum- y update
reboot
```

Myös paketteja, joita ei ole valmiiksi, asennetaan kaikille koneille komennolla:

```
yum install -y wget git net-tools docker-1.13.1 bind-utils iptables-ser-
vices bridge-utils bash-completion kexec-tools psacct openssl-devel
httpd-tools NetworkManager python-cryptography python2-pip python-devel
python-passlib java-1.8.0-openjdk-headless "@Development Tools"
```

Käynnistetään ohjelma NetworkManager sekä sallitaan ohjelma, jotta se käynnistyy automaattisesti virtuaalikoneen käynnistyksessä. Tämä tehdään kaikilla koneilla.

```
systemctl start NetworkManager
systemctl enable NetworkManager
```

Asennetaan kaikille koneille viimeisin Extra Packages for Enterprise Linux (EPEL) -pakettikirjasto. Tällä varmistetaan, että tarvittavat pakettikirjastot ovat ajan tasalla ja asennus onnistuu normaalisti.

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-
7.noarch.rpm
```

Master-koneelle asennettavat paketit asennetaan komennolla:

```
yum -y --enablerepo=epel install ansible pyOpenSSL
```

Master-koneelle kloonataan OpenShiftin virallisesta GitHub-repositorysta Ansiblelle tarvittavat tiedostot sekä varmistetaan versio:

```
git clone https://github.com/openshift/openshift-ansible.git
cd openshift-ansible && git fetch && git checkout release-3.9 && cd ..
```

Pysäytetään ja käynnistetään Docker-palvelu jokaisella koneella. Sallitaan myös palvelu, jotta se käynnistyy automaattisesti palvelimen käynnistyksessä.

```
systemctl stop docker
systemctl restart docker
systemctl enable docker
```

Jotta OpenShift sekä Ansible pystyvät toimimaan, jokaiselta koneelta toiselle pitää sallia SSH-pääsy. Kun SSH-avaimet on jaettu palvelinten välillä, ei enää tarvitse kirjoittaa salasanaa ottaessaan yhteyttä palvelimelta toiselle. SSH-avaimet voidaan jakaa alla olevilla komennoilla:

```
master
ssh-copy-id -i ~/.ssh/id_rsa.pub 10.1.1.20
ssh-copy-id -i ~/.ssh/id_rsa.pub 10.1.1.21

node01
ssh-copy-id -i ~/.ssh/id_rsa.pub 10.1.1.10
ssh-copy-id -i ~/.ssh/id_rsa.pub 10.1.1.21

node02
ssh-copy-id -i ~/.ssh/id_rsa.pub 10.1.1.10
ssh-copy-id -i ~/.ssh/id_rsa.pub 10.1.1.20
```

3.3 Asennuskonfiguraatio

Kun kaikki valmistelut virtuaalikoneille on tehty, luodaan Ansiblelle inventory.ini -tiedosto, joka sisältää kaiken tiedon ja konfiguraation, minkä pohjalta OpenShift-klusteri luodaan. Tätä tiedostoa Ansible-ohjelma käyttää asentaessaan klusterin.

inventory.ini-tiedostossa ensimmäisenä määritetään ryhmät, joita klusterissa käytetään. Tässä työssä tarvitaan masters, nodes sekä etcd.

```
[OSEv3:children]
masters
nodes
etcd
```

Seuraavat rivit määrittelevät, mitkä koneet kuuluvat mihinkin ryhmään klusterissa. Halutun master-koneen täytyy löytyä sekä masters- että nodes-osion alta. Tässä klusterissa myös etcd asentuu masterille.

```
[masters]
10.1.1.10 openshift_ip=10.1.1.10 openshift_schedulable=true
[etcd]
10.1.1.10 openshift_ip=10.1.1.10
[nodes]
10.1.1.10 openshift_ip=10.1.1.10 openshift_schedulable=true
openshift_node_labels="{'region': 'infra', 'zone': 'default'}"
10.1.1.20 openshift_ip=10.1.1.20 openshift_schedulable=true
openshift_node_labels="{'region': 'primary', 'zone': 'east'}"
```

```
10.1.1.21 openshift_ip=10.1.1.21 openshift_schedulable=true
openshift_node_labels="{'region': 'primary', 'zone': 'east'}"
```

Muuttujat määritetään [OSEv3:vars]-osion alle. Tässä klusterissa tarvitsee määrittää seuraavat muuttujat.

Ansible-ohjelmalle käyttäjä, jota ohjelma käyttää:

```
ansible_ssh_user=root
```

Tarvittavat muuttujat ympäristön tyyppiin sekä versioon:

```
containerized=True
deployment_type=origin
openshift_deployment_type=origin
openshift_release=v3.9.0
openshift_pkg_version=-3.9.0
openshift_image_tag=v3.9.0
openshift_service_catalog_image_version=v3.9.0
template_service_broker_image_version=v3.9.0
```

OpenShift-klusterille kerrotaan, että htpasswd-ohjelma määrittää käyttäjätunnukset klusterille ja missä käyttäjätunnustiedosto sijaitsee:

```
openshift_master_identity_providers=[{'name': 'htpasswd_auth', 'login':
'true', 'challenge': 'true', 'kind': 'HTPasswdPasswordIdentityProvider',
'filename': '/etc/origin/master/htpasswd'}]
```

Viimeisimpänä kerrotaan, että alidomain console.example.com on OpenShiftin julkinen nimi sekä oletusalidomain on apps.example.com.

```
openshift_public_hostname=console.example.com
openshift_master_default_subdomain=apps.example.com
```

Tämän jälkeen voidaan aloittaa klusterin luonti Ansiblen avulla. OpenShiftin viralliselta GitHub-repositorystä ladattu openshift-ansible-paketti /opt/-osioon sisältää kaiken tarvittavan. Tästä paketista käytämme kahta Ansible playbook -konfiguraatiotiedostoa:

```
/opt/openshift-ansible/playbooks/prerequisites.yml
/opt/openshift-ansible/playbooks/deploy_cluster.yml
```

Ensimmäisenä ajetaan prerequisites.yml alla olevalla komennolla. Argumentti -i määrittää Ansiblen tarvittavan inventory-tiedoston.

```
ansible-playbook -i inventory.ini openshift-ansible/playbooks/prerequi-
sites.yml
```

Tämä valmistelee kaiken klusterin luomiseen, muun muassa Docker-ohjelman muuttujia. Kun prerequisites.yml on suoriutunut onnistuneesti, voidaan ajaa deploy_cluster.yml:

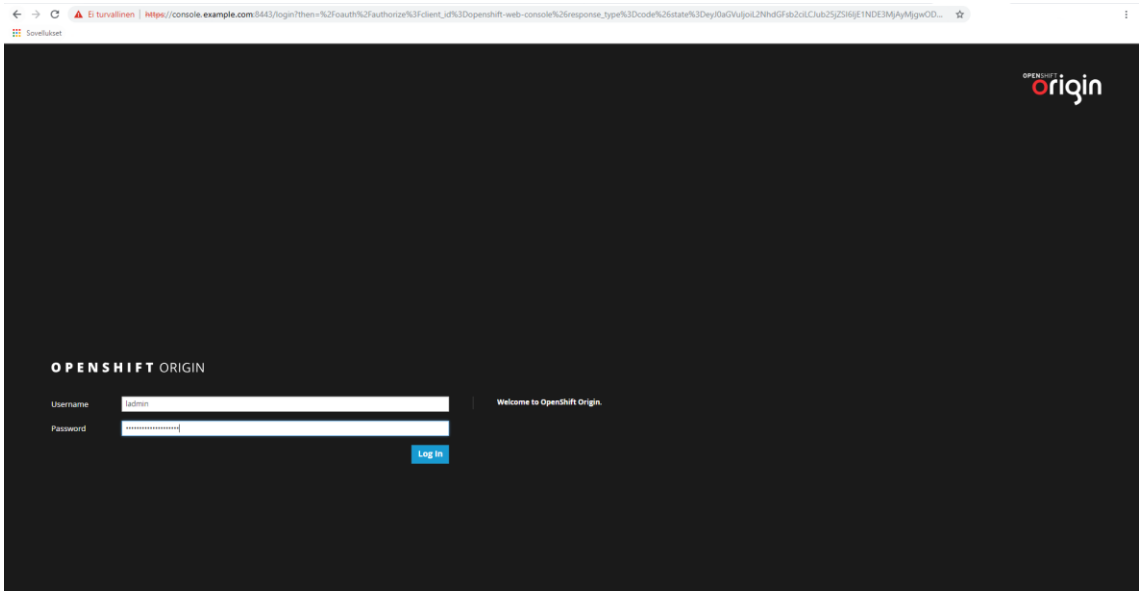
```
ansible-playbook -i inventory.ini openshift-ansible/playbooks/de-
ploy_cluster.yml
```

Asennus kestää useita minuutteja riippuen ympäristön laajuudesta, klusterin koosta ja koneiden resursseista. Mikäli jokin muuttuja on pielessä, asennus epäonnistuu ja Ansible keskeyttää sekä ilmoittaa, missä ongelma on. Mikäli asennus onnistuu, ilmoittaa Ansible asennuksen jälkeen. Kun asennus on suoritettu onnistuneesti, voidaan luoda vapaavalintainen käyttäjä OpenShift-ympäristöön. Tämä tapahtuu htpasswd-ohjelmalla ja alla olevalla komennolla luodaan ladmin-niminen käyttäjä, jolle toisen rivin komennolla annetaan cluster-admin rooli:

```
[root@master ~]# htpasswd -c /etc/origin/master/htpasswd ladmin
[root@master ~]# oc adm policy add-cluster-role-to-user cluster-admin
ladmin
```

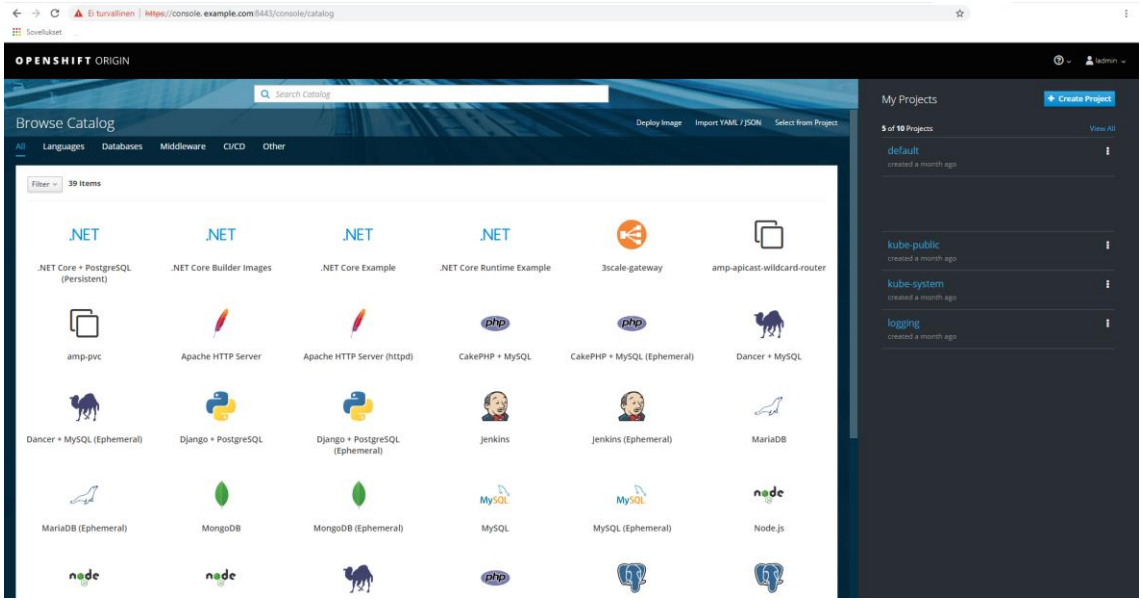
Tämän jälkeen kaikki on valmista. OpenShiftin selainpohjainen hallinta löytyy osoitteesta https://console.example.com:8443. Asennuksessa on automaattisesti otettu käyttöön Secure Sockets Layer (SSL) -protokolla, joka salaa liikenteen käyttäjän ja sivuston välillä. Käytössä on self-signed sertifikaatti, jonka takia selain varoittaa sivustosta.

Kun sertifikaatti hyväksytään, tulee OpenShift-selainpohjaisen hallinnan kirjautumissivu. Tähän voimme käyttää aikaisemmin luotuja ladmin-tunnuksia:



Kuva 10. Hallintasivusto

Kirjautumisen jälkeen OpenShift-selainpohjaisen hallinnan sivusto näyttää oletuksena seuraavalta:



Kuva 11. Hallintasivuston oletusnäkymä

CLI:ltä voidaan myös hallita OpenShift-klusteria samalla tavalla kuin hallintasivuston kautta. Esimerkiksi projekteja, julkaisuja sekä päivityksiä voidaan tehdä CLI:ltä. OpenShiftin CLI:tä voidaan ohjata vain masterilta. Nodeilla komennot eivät toimi.

```
[root@master ~]# oc whoami
ladmin

[root@master ~]# oc get nodes
NAME      STATUS    ROLES     AGE       VERSION
master    Ready     master    65d       v1.9.1+a0ce1bc657
node01    Ready     compute   65d       v1.9.1+a0ce1bc657
node02    Ready     compute   65d       v1.9.1+a0ce1bc657
```

”oc get nodes” -komennolla näkee kaikki klusterin jäsenet, tällä kertaa masterin ja kaksi nodea. Kuten koneiden rooleista huomaa, nodet ovat compute-tyypin koneita, eli niin sanottuja klusterin työntekijöitä.

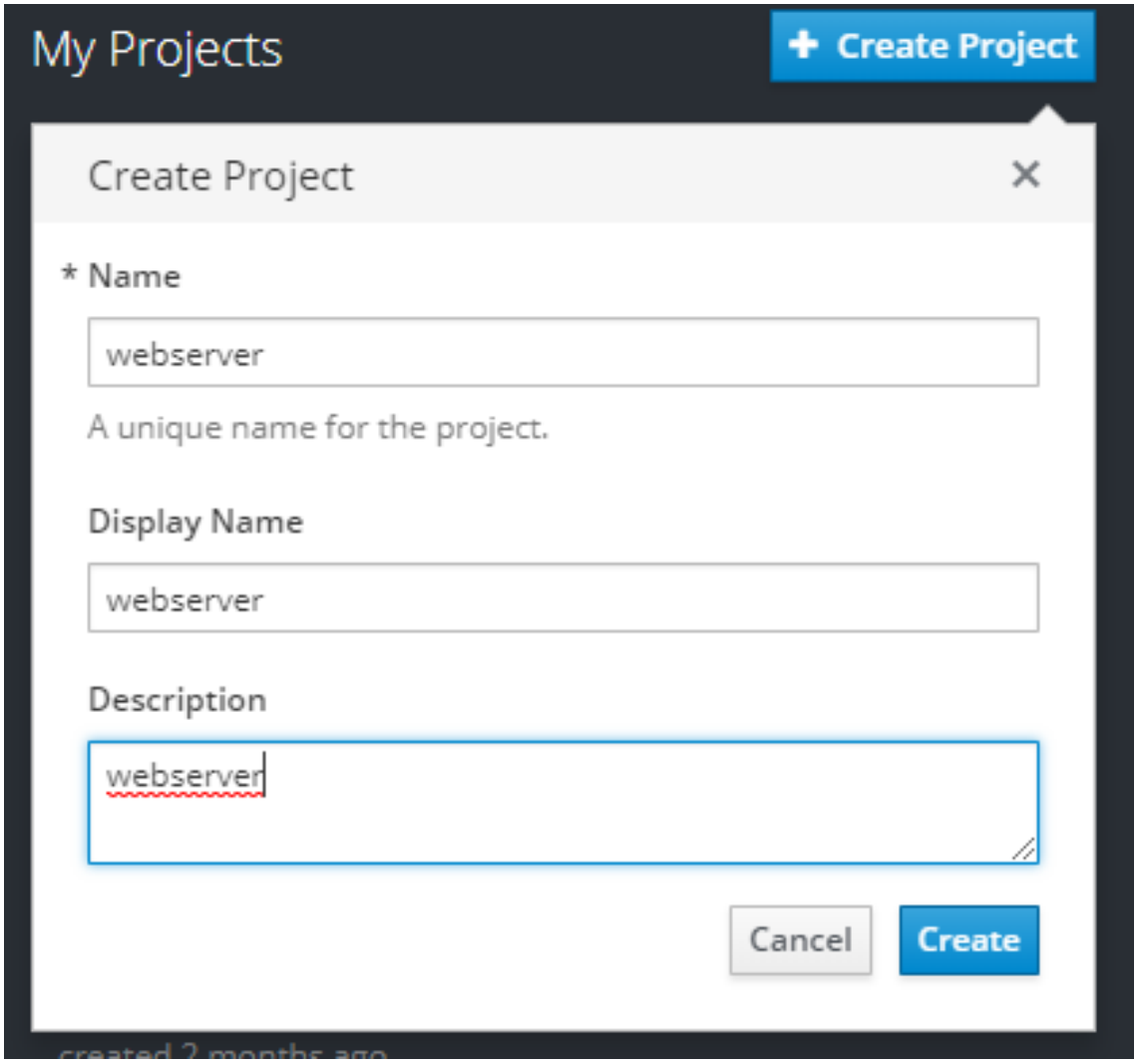
Esimerkiksi uusi projekti voidaan luoda helposti komennolla oc new-project *projektin nimi*.

```
[root@master ~]# oc new-project testproject
Now using project "testproject" on server "https://master:8443"


[root@master ~]# oc get projects
NAME                    DISPLAY NAME   STATUS
default                                Active
kube-public                            Active
kube-system                            Active
logging                                Active
management-infra                       Active
openshift                              Active
openshift-infra                        Active
openshift-node                         Active
openshift-web-console                  Active
testproject                            Active
webserver               webserver      Active
```

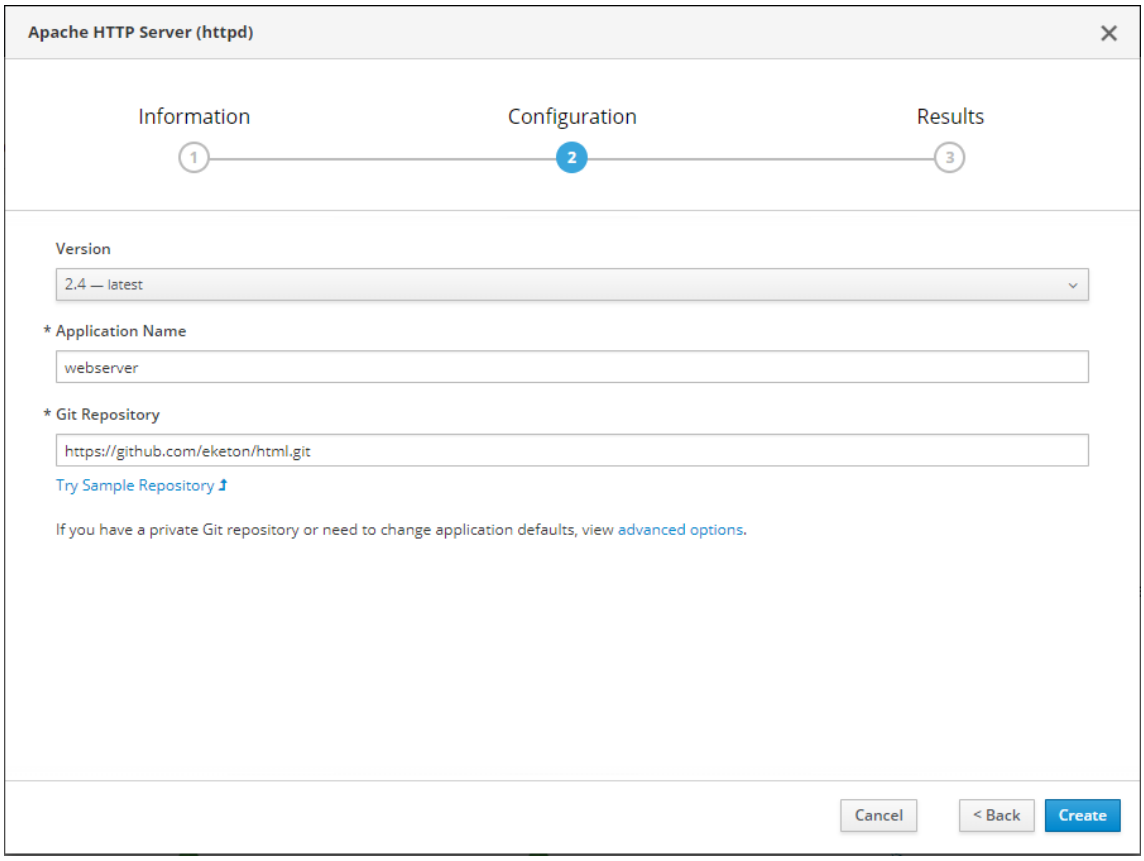
4 Ympäristön testaus

Ympäristön testausta varten täytyi luoda jokin ohjelma pyörimään sovelluskontteihin. Valitsin yksinkertaisen Apache HTTP Web Serverin. OpenShiftissä ensimmäisenä luodaan projekti, jonka sisään luodaan sovellusjulkaisuja. HTTP-palvelinta varten luodaan vapaavalintaisella nimellä projekti.



Kuva 12. Testiprojekti

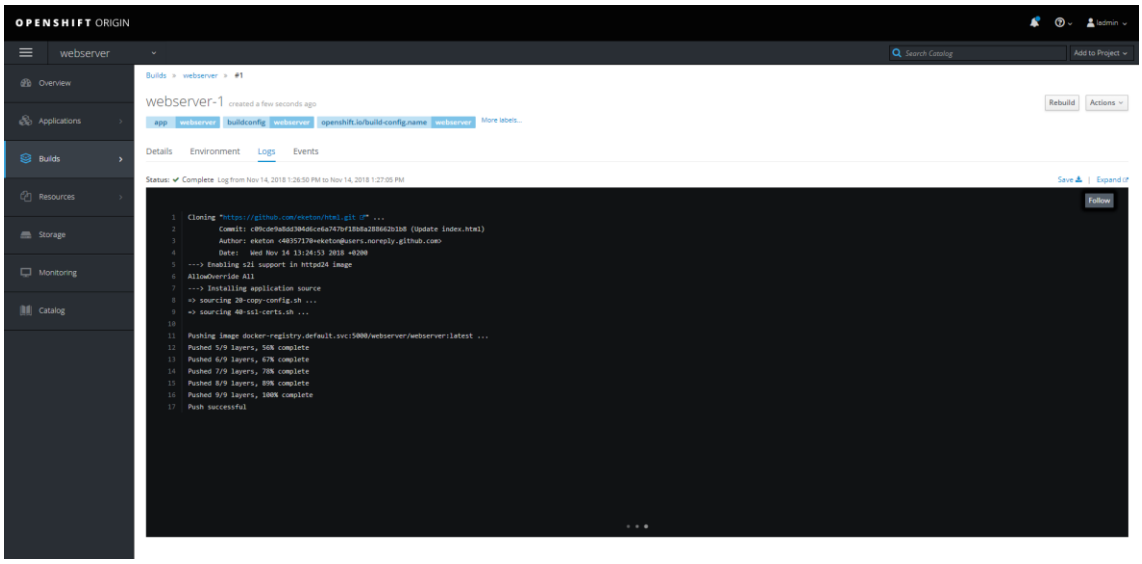
Projektin luomiseen menee noin sekunti. Tämän jälkeen projektin alle voidaan luoda vapaavalintaisia sovelluksia. Testiprojektissa valittiin valmiiksi katalogista Apache HTTP Server (httpd).



Kuva 13. Sovelluksen luonti

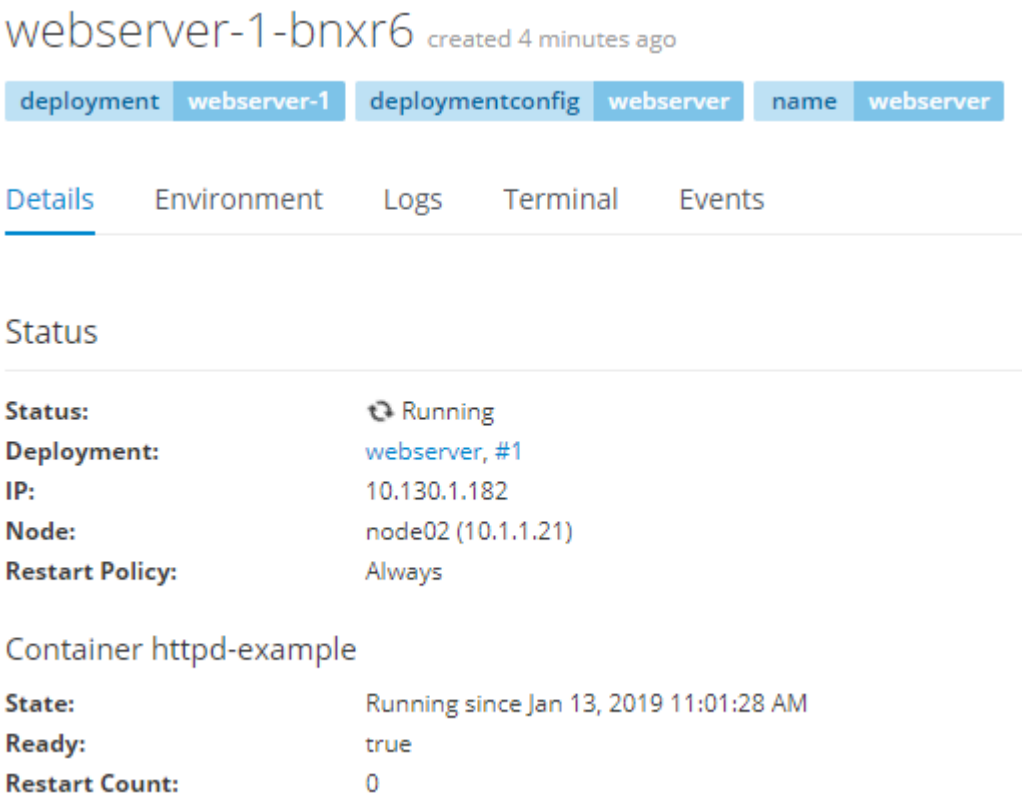
Sovellusta luotaessa määritetään repositoryn sijainti, jota käytetään Apache HTTP -sovelluksessa. Projektia varten github.com:iin luotu hyvin yksinkertainen paketti Apache HTTP -palvelinta varten. Repository saadaan osoitteesta https://github.com/eketon/html.git. Tämä sisältää index.html-tiedoston, jota käytetään HTTP-palvelimen oletussivuna. Lisäksi repository sisältää audiotiedoston sekä pakollisen README.md-tiedoston.

Kun sovellus on luotu, voidaan projektin lokeista seurata edistymistä. HTTP-palvelimen luomiseen ei kulunut paljoa aikaa, alle kymmenen sekuntia.



Kuva 14. Deploymentin logit

Oletuksena OpenShift loi HTTP-palvelimelle yhden podin nimeltä ”webserver-1-bnxr6”. Klikkaamalla podia voidaan nähdä sen tarkemmat tiedot.



Kuva 15. Ensimmäinen pod

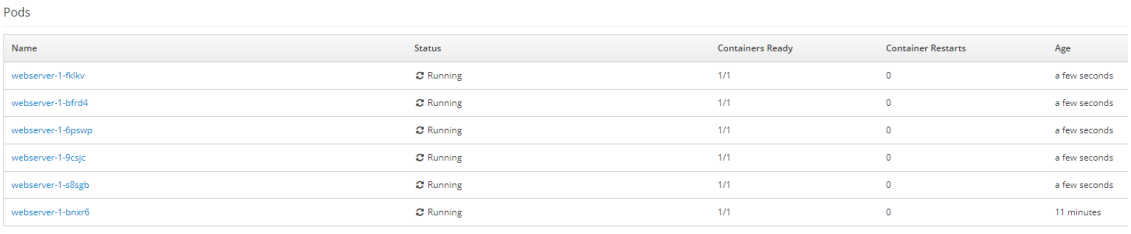
Huomataan, että pod ” webserver-1-bnxr6” on tällä hetkellä toiminnassa, sen IP-osoite on 10.130.1.182 ja pyörii node02:llä. Node02:n komentoriviltä voimme myös tarkastaa komennolla ”docker ps”, mikä näyttää kaikki Docker-sovelluskontit. Suodatetaan hakutuloksia myös nimellä käyttäen grep-työkalua niin vastaukseksi tulee vain tähän podiin liittyvät kontit. Vastaukseksi komentorivi tulostaa kahden kontin tiedot.

```
[root@node02 ~]# docker ps | grep bnxr6

CONTAINER ID:7e843737363f
IMAGE:docker-registry.default.svc:5000/webserver/web-
server@sha256:5d73b93620cf7a30701ed395836059347c1a7c8cce933e3b0b63e4c2afb19b51
COMMAND:"container-entrypo..."
CREATED:7 minutes ago
STATUS:Up 7 minutes
PORTS:
NAMES:k8s_webserver_webserver-1-bnxr6_webserver_63e03c44-e80e-11e8-a4bd-
000c294b9ef9_0

CONTAINER ID:6f7bcf4ca132
IMAGE:openshift/origin-pod:v3.9.0
COMMAND:"/usr/bin/pod"
CREATED:7 minutes ago
STATUS:Up 7 minutes
PORTS:
NAMES:k8s_POD_webserver-1-bnxr6_webserver_63e03c44-e80e-11e8-a4bd-
000c294b9ef9_0
```

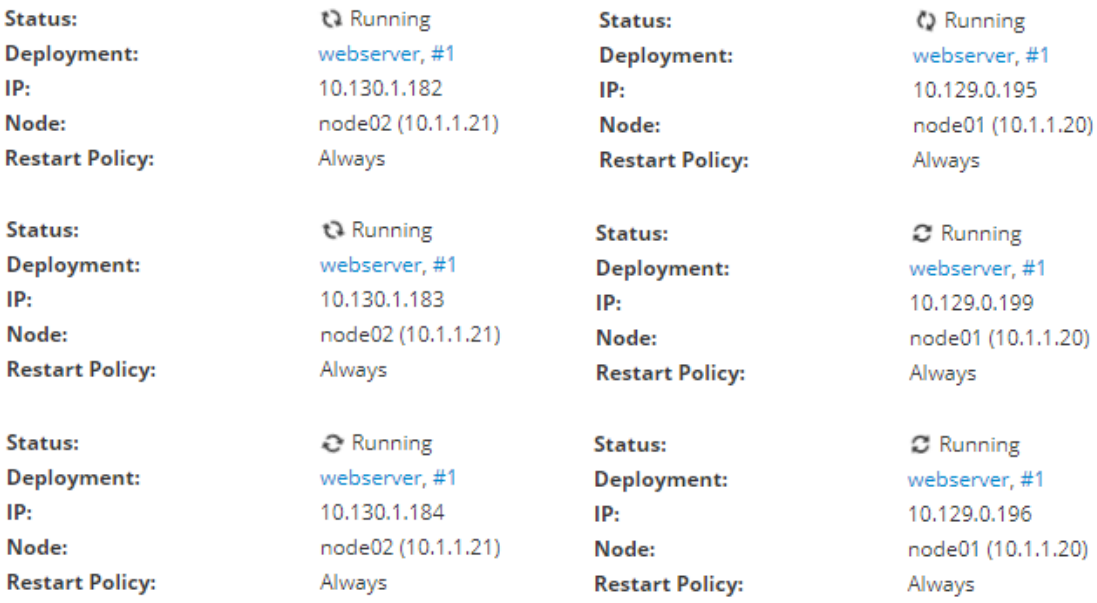
Tämän jälkeen skaalataan sovellusta yhdestä sovelluskontista useampaan. Helpointa skaalaus on tehdä hallintasivuston kautta Applications → Deployments → webserver. Nuolta ylöspäin klikkaamalla skaalaus tapahtuu sekunneissa. Skaalaus voidaan tehdä myös CLI:ltä ”oc scale”-komennolla. HTTP-palvelimen podien määrä on nostettu yhdestä kuuteen. Tähän kului aikaa noin kymmenen sekuntia.


Pods

Name	Status	Containers Ready	Container Restarts	Age
webserver-1-fklkv	Running	1/1	0	a few seconds
webserver-1-bfrd4	Running	1/1	0	a few seconds
webserver-1-6pswp	Running	1/1	0	a few seconds
webserver-1-9csjc	Running	1/1	0	a few seconds
webserver-1-s8sgb	Running	1/1	0	a few seconds
webserver-1-bnxr6	Running	1/1	0	11 minutes

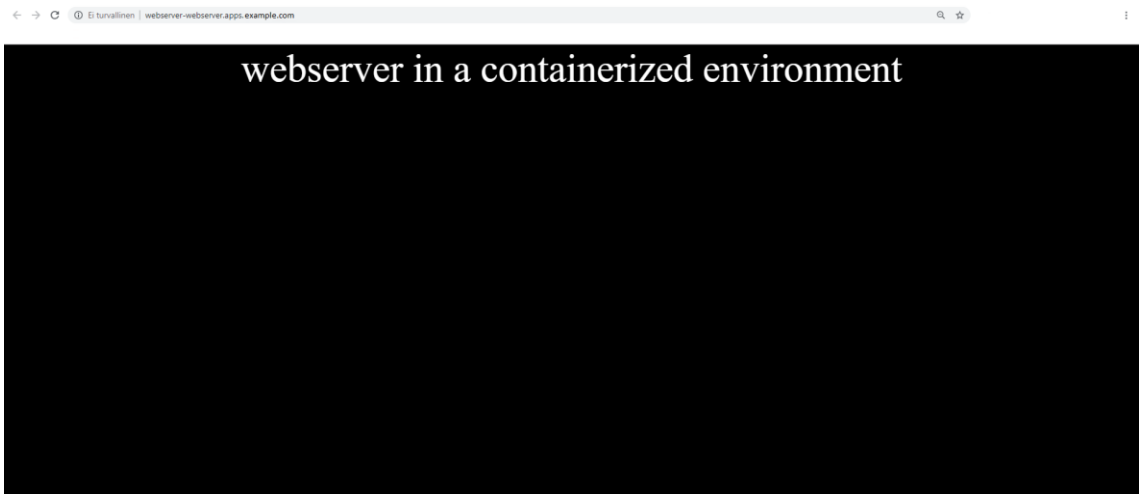
Kuva 16. Skaalaus

Voidaan tarkastella, että OpenShift on jakanut kuorman tasaisesti molemmille nodeille. Molemmilla nodeilla pyörii tällä hetkellä kolme podia.



Kuva 17. Podit

OpenShift loi automaattisesti reitin HTTP-palvelimelle. Tämä on mahdollista luoda myös itse. Automaattisesti luotu URL on http://webserver-webserver.apps.example.com. Automaattinen syntaksi on http://*projekti*.*deployment*.apps.example.com. Tämä reitti ei oletuksena käytä SSL:ää, mutta sen voi helposti lisätä hallintapaneelin kautta menemällä Applications → Routes → webserver → Edit → Secure route. Sinne voi lisätä oman sertifikaatin SSL:lle käytettäväksi.



Kuva 18. Webserver-projektin HTTP-sivusto

Vikasietoisuus

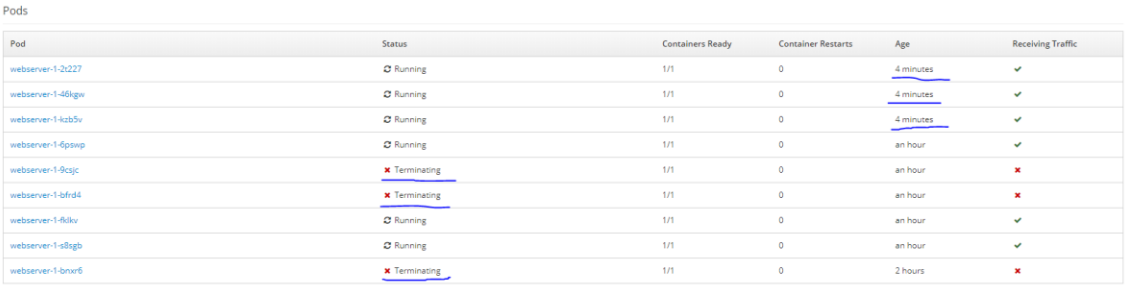
Vikasietoisuus nykypäivänä on hyvin tärkeää yritysten tarjoamissa palveluissa eikä häiriöaikaa saisi olla. OpenShift ja tämän sisällä klusterista ja podien orkestroinnista vastaava Kubernetes ovat varsin älykkäitä ja nopeita reagoimaan häiriöihin. Esimerkkiympäristöksi on nyt rakennettu kolmen koneen klusteri, yksi master ja kaksi nodea. Molemmilla nodeilla on ajossa kolme identtistä podia, jotka kaikki pyörittävät samaa edellä luotua HTTP-palvelinta.

Tässä ympäristössä olisi vikasietoisuuden kannalta huomattavasti parannettavaa. Esimerkiksi master ei ole tällä hetkellä kahdennettu. Mikäli master-virtuaalikone hajoaisi niin koko klusteri lakkaisi toimimasta. Jos vielä otetaan huomioon palvelut, joista OpenShift on riippuvainen, esimerkiksi VMware host, jolla koneet ovat ajossa. Fyysinen host voi myös hajota tai menettää sähköt. Olisi järkevää ajaa klusterin koneita eri fyysisillä hosteilla. Tämän ympäristön vikasietoisuutta olisi voinut parhaiten parantaa lisäämällä toisen master-koneen klusteriin, mutta fyysisen palvelimen resurssit olivat rajalliset. Mikäli olisi ollut toinen fyysinen palvelin, olisivat kaksi palvelinta toimineet klusterina ja OpenShift-klusterin virtuaalikoneet olisi jaettu eri palvelimille.

OpenShiftille on kerrottu, että webserver-projektissa haluttu podien määrä on kuusi. Mikäli määrä putoaa alle halutun määrän, OpenShift korjaa tilanteen ja luo tarvittavan määrän uusia podeja. Voimme tahallisesti aiheuttaa OpenShiftille häiriötä sammuttamalla toisen nodeista VMwaresta. Tämä on siis sama tilanne, kuin jos kone menettäisi sähkönsä.

```
[root@master ~]# oc get nodes
NAME       STATUS     ROLES     AGE       VERSION
master     Ready      master    65d       v1.9.1+a0ce1bc657
node01     Ready      compute   65d       v1.9.1+a0ce1bc657
node02     NotReady   compute   65d       v1.9.1+a0ce1bc657
```

OpenShift on huomannut, että node02 ei vastaa. OpenShift on myös nopeasti huomannut, että node02:lla olevat podit eivät enää reagoi klusterin omassa valvonnassa. OpenShift on laittanut nämä podit tuhoutumaan ja luonut automaattisesti kolme uutta podia node01:lle. Tällä hetkellä kaikki HTTP-palvelimen kuusi podia pyörivät node01:llä.



Kuva 19. Uusien podien automaattinen luonti

Missään vaiheessa HTTP-palvelin ei lakannut kokonaan toimimasta, koska koko ajan oli vähintään yksi pod, joka tarjosi HTTP-palvelin -palvelua. Kun node02 nostettiin takaisin päälle VMwaresta, ajan myötä OpenShift tasasi podit molemmille nodeille. Molemmilla nodeilla on nyt kolme podia ajossa.

5 Yhteenveto

Opinnäytetyössä tutustuttiin sovelluskonttiteknologiaan sekä sen yhteen suosituimmista hallinta-alustoista. Sovelluskonttiteknologia on ollut jo pitkään näkyvillä, mutta hiljattain sen suosio on kasvanut räjähdysmäisesti. Työssä tutustuttiin vapaan lähdekoodin OpenShift Container Platformiin, Openshift Originiin. Pohjimmaisena sovelluskonttitasolla tutustuttiin Dockeriin sekä konttien orkestrointia hoitavaan Googlen Kubernetekseen.

Tämän työn tavoite oli asentaa Kolmen CentOS-virtuaalikoneen OpenShift Origin -klusteri sekä tutustua OpenShiftin toimintaan. Asennus onnistui pitkien valmistelujen sekä useiden epäonnistumisten jälkeen. Useimmiten asennus epäonnistui jonkin tietyn ohjelmistopaketin puuttumisen vuoksi. Työssä testattiin myös OpenShiftin reagointia vikatilanteisiin ja tulokset olivat odotettuja. Mikäli konteille olisi vielä asentanut tarkemman valvonnan tiheämmällä aikavälillä, olisi klusteri reagoinut nopeammin vikatilanteisiin.

Opinnäytetyössä ei verrattu OpenShiftiä muiden valmistajien alustoihin vaan työssä keskityttiin vain OpenShiftiin ja sen toiminnallisuuksiin sekä käytettävyyteen. Klusterissa ajettiin myös hyvin yksinkertaista kontitettua verkkosovellusta hajautettuna kahdelle virtuaalikoneelle. Sovelluksen julkaisu oli tehty erittäin helpoksi. OpenShift tuo helppouden julkaista sekä päivittää kontitettuja sovelluksia ilman käyttökatkoja.

Kyseinen asennettu ympäristö ei sopisi yrityksen tuotantoympäristöksi vaan enemmänkin kotikäyttäjän tai pienen yrityksen testiympäristöksi. Mikäli fyysisiä resursseja olisi ollut enemmän saatavilla, olisi ympäristöön voinut lisätä enemmän mastereita sekä nodeja. Ympäristöön olisi voinut lisätä myös kuormanjaon. Nykyinen ympäristö on täydellinen omien sovelluksien testaukseen sekä OpenShiftiin ja konttiteknologiaan tutustumiseen.

Lähteet

1 Understanding Full Virtualization, Paravirtualization and Hardware Assist. 2008. Verkkoaineisto. VMware https://www.vmware.com/techpapers/2007/understanding-full-virtualization-paravirtualizat-1008.html. Luettu 30.9.2018.

2 Margaret Rouse. 2016. Hypervisor. Verkkoaineisto. Techtarget. https://searchservervirtualization.techtarget.com/definition/hypervisor. Luettu 3.11.2018.

3 Bernard Golden. 2011. HP Virtualization for dummies. Verkkoaineisto. Wiley Publishing Inc. https://ssl.www8.hp.com/de/de/pdf/virtuallisation_tcm_144_1147500.pdf. Luettu 30.9.2018

4 What's a Linux Container? Verkkoaineisto. Redhat. https://www.redhat.com/en/topics/containers/whats-a-linux-container. Luettu 10.11.2018.

5 What is Kubernetes? Verkkoaineisto. Kubernetes. https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/. Luettu 17.11.2018.

6 Klusteri. 2017. Verkkoaineisto. Wikipedia. https://fi.wikipedia.org/wiki/Klusteri_(tietotekniikka). Luettu 17.11.2018.

7 Nick Martin. 2015. A brief history of Docker Containers' overnight success. Verkkoaineisto. https://searchservervirtualization.techtarget.com/feature/A-brief-history-of-Docker-Containers-overnight-success. Luettu 10.11.2018.

8 Docker terminology. 2019. Verkkoaineisto. https://docs.microsoft.com/en-us/dotnet/standard/microservices-architecture/container-docker-introduction/docker-terminology. Luettu 17.11.2018

9 What is Kubernetes – Learn Kubernetes from Basics. 2018. Verkkoaineisto. https://www.learnitguide.net/2018/08/what-is-kubernetes-learn-kubernetes.html. Luettu 19.2.2019

10 What is a Pod? Verkkoaineisto. Kubernetes. https://kubernetes.io/docs/concepts/workloads/pods/pod/. Luettu 17.11.2018

11 Justin Ellingwood. An Introduction to Kubernetes. 2018. Digitalocean. https://www.digitalocean.com/community/tutorials/an-introduction-to-kubernetes. 17.11.2018.

12 Kubernetes Components. Verkkoaineisto. Kubernetes. https://kubernetes.io/docs/concepts/overview/components/. Luettu 17.11.2018

13 Michael Heap. Ansible From Beginner to Pro. 2016. Michael Heap. Saatavilla maksullisesti osoitteesta: https://www.amazon.com/Ansible-Beginner-Pro-Michael-Heap-ebook/dp/B01LZAW11H. Luettu 31.11.2018.

14 What Is Ansible? Verkkoaineisto. Networklore. https://networklore.com/ansible/. Luettu 31.11.2018.

15 OpenShift. Verkkoaineisto. Wikipedia. https://en.wikipedia.org/wiki/OpenShift#cite_note-5. Luettu 17.11.2018.

16 Katie Miller, Steve Pousty. Getting Started with OpenShift. 2014. O'Reilly. https://www.oreilly.com/library/view/getting-started-with/9781491900468/ch01.html. Luettu 31.11.2018.

17 OpenShift Documentation. Verkkoaineisto. Redhat. https://docs.openshift.com/. Luettu 17.11.2018.

18 Architecture Overview. Verkkoaineisto. Redhat. https://docs.okd.io/3.9/architecture/index.html. Luettu 24.11.2018.

19 Cluster limits. Verkkoaineisto Redhat. https://docs.okd.io/latest/scaling_performance/cluster_limits.html#scaling-performance-cluster-limits. Luettu 6.11.2018.

20 Installing a Highly-Available OpenShift Cluster. Verkkoaineisto. Redhat. http://v1.uncontained.io/playbooks/installation/. Luettu 8.12.2018.

21 Networking. Verkkoaineisto. Redhat. https://docs.okd.io/3.9/architecture/networking/networking.html. Luettu 6.11.2018.

22 OpenShift SDN. Verkkoainesto. Redhat. https://docs.okd.io/3.9/architecture/networking/sdn.html. Luettu 8.12.2018

23 How Containers are Secured on RHEL. Verkkoaineisto. Redhat. https://docs.okd.io/latest/security/hosts_multitenancy.html. Luettu 8.12.2018.

24 OpenShift Container Platform 3.9. Architecture. Verkkoaineisto. Redhat. https://access.redhat.com/documentation/en-us/openshift_container_platform/3.9/pdf/architecture/OpenShift_Container_Platform-3.9-Architecture-en-US.pdf. Luettu 8.12.2018.

25 Configuring authentication. Verkkoainesto. Redhat. https://docs.openshift.com/enterprise/3.0/admin_guide/configuring_authentication.html. Luettu 8.12.2018.

26 Cisco Container Platform 2.2.0 User Guide. Verkkoaineisto. Cisco Systems. https://www.cisco.com/c/en/us/td/docs/net_mgmt/cisco_container_platform/2-2/User_Guide/CCP-User-Guide-2-2-0/CCP-User-Guide-2-2-0_chapter_00.html. Luettu 19.1.2019.

27 Azure Kubernetes Service (AKS). 2018. Verkkoaineisto. Microsoft. https://docs.microsoft.com/en-us/azure/aks/intro-kubernetes. Luettu 19.1.2019.

28 Amazon Elastic Container Service. Verkkoaineisto. Amazon. https://aws.amazon.com/ecs/#. Luettu 19.1.2019.

29 AWS Fargate. Verkkoaineisto. Amazon. https://aws.amazon.com/fargate/. Luettu 19.1.2019.