jamk

Microk8s hallintatyökalut

Anton Jylhä

Opinnäytetyö, AMK
Toukokuu 2025
Tieto- ja viestintätekniikan tutkinto-ohjelma

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Jylhä Anton

Microk8s hallintatyökalut

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu 2025, 66 sivua

Tieto- ja viestintätekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Tietotekniikan kehitys on lähtenyt liikkeelle suurista fyysisistä laskentatietokoneista, jonka jälkeen ajan myötä on siirrytty virtuaalikoneisiin ja sitä kautta vielä konttiteknologian pariin. Keksinnöt ovat mullistavia meidän maailmassamme, jossa energia- ja kustannustehokkuus ovat avainasemassa. Kehittäjiä ajansaatossa on ollut valtaisasti ja kun tekniikka on stabiloitunut, niin nykyaikaisten kontitettujen palvelujen tuottamiseen tarvitaan palvelun hallintatyökaluja kuten työssä käytetty Kubernetes.

Kun puhutaan IT-palvelun käyttöönotosta, on nykyään välttämätöntä seurata jonkinlaisia standardeja, ohjeistuksia ja prosesseja, jotta ympäristöstä saa tehtyä vakaan, turvallisen ja helposti lähestyttävän. Erilaisia yleiskattavia ohjeita ja toimintamalleja löytyy paljon ja muutamat niistä ovat vakiinnuttanut paikkansa IT-palvelutuotannon piireissä. Näitä ovat ITIL ja DevOps. Nämä kaksi käsitettä ja niiden toimintamallit ovat monille yrityksille elintärkeitä, kun halutaan lähteä luomaan kaikin puolin stabiilia IT-ympäristöä.

Kubernetekselleki, kun on kehitetty huima määrä erinäisiä hallinta- ja monitorointityökaluja, on näistä osattava valita sopivammat eri tilanteisiin. Kun organisaatio haluaa löytää sopivan ohjelman käyttöönsä, löytyy vähäisesti tietoa siitä, mikä ohjelma olisi paras vaihtoehto, kun halutaan ajatella palvelutuotannon kehittämistä myös ITIL:in ja DevOpsin näkökulmasta.

Tässä työssä, kun tutustuttiin viiteen Kubernetekselle suunnattuun työkaluun, on siinä osattu ottaa huomioon myös nämä keskeiset palvelutuotannon näkökulmat. Ohjelmia vertaillaan näiden käsitteiden kautta yksinkertaisessa, yksiklusterisessa ympäristössä, johon on otettu käyttöön verkkokauppa.

Lopputuloksena syntyi kattava vertailu erilaisista tunnetuista ohjelmista ja pohdintaa siitä, minkä kokoiseen Kubernetes-ympäristöön mikäkin sopisi.

Avainsanat (asiasanat)

Kubernetes, konttiteknologia, Linux, ITIL, DevOps, palvelutuotanto, k8s, monitorointi, Microk8s

Muut tiedot (salassa pidettävät liitteet)

-

jamk

Description

Jylhä Anton

Microk8s management tools

Jyväskylä: JAMK University of Applied Sciences, May 2025, 66 pages.

Degree Programme in Information and Communication Technology. Bachelor´s thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

The development of information technology started with large physical computing machines, and over time progressed to virtual machines and eventually to container technology. These innovations have been revolutionary in our world, where energy and cost efficiency are key priorities. Throughout this journey, countless developers have contributed, and as the technology has stabilized, modern containerized service production now requires management tools such as Kubernetes, which was used in this project.

When discussing the deployment of IT services, it has become essential to follow certain standards, guidelines, and processes to create an environment that is stable, secure, and accessible. There are many general frameworks and operational models available, and some have established their place in the field of IT service management. Among these are ITIL and DevOps. These two concepts and their practices are vital for many companies seeking to build a stable IT environment in every respect.

Since a vast array of management and monitoring tools have been developed for Kubernetes, it is important to choose the right ones for different situations. When an organization wants to select a suitable tool, there is little information available on which solution would be the best choice when considering service production from both an ITIL and DevOps perspective.

In this work, five Kubernetes-focused tools were examined, taking into account these key service management perspectives. The programs were compared through these frameworks in a simple, single-cluster environment running an e-commerce platform.

The result was a comprehensive comparison of various well-known programs and an analysis of which Kubernetes environment each would be best suited for.

Keywords/tags (subjects)

Kubernetes, container technology, Linux, ITIL, DevOps, service production, k8s, monitoring, Microk8s

Miscellaneous (Confidential information)

-

Sisältö

Kuviot

Taulukot

Käsitteitä

ITIL = Information Technology Infrastructure Library. Selitetään alempana.

IP = Internet Protocol. Protokolla, joka huolehtii IP-pakettien toimittamisesta perille tietoverkossa.

VUH = Virtual User Hours. Virtuaalikäyttäjien käyttämä aika. Lasketaan (virtuaalikäyttäjien määrä * käyttöaika) / 60 minuuttia.

API = Application Programming Interface. Ohjelmointirajapinta, jonka avulla ohjelmat voivat keskustella keskenään.

YAML = YAML Ain't Markup Language. YAML on ihmisystävällinen tietojen serialisointi kieli kaikille ohjelmointikielille. käytetään usein konfiguraatiotiedostoissa

URL = Uniform Resource Locator. Merkkijono, jolla kerrotaan tietyn tiedon paikka.

1 Johdanto

Virtualisointi ja konttiteknologia ovat mullistavia keksintöjä, joita hyödynnetään hyvin laajasti nyky-yhteiskunnassa. Konttiteknologia myös tuo mukanaan erinäisiä ohjelmistoja, joista tunnetuimpien joukossa ovat esimerkiksi Docker ja Kubernetes. Kuberneteksen hallintatyökalujen kehitys on mennyt eteenpäin myös useiden kehittäjien halusta. Tätä myötä työkaluja on julkaistu kaikkien käyttöön valtaisa määrä, mutta näistä on hyvin vähän tehty kattavaa vertailua.

Monet ohjelmat ovat hyvin kattavia ominaisuuksiltaan ja on tehtävä kattava selvitys, että mitkä niistä lopulta soveltuu kellekin parhaiten eri kriteerien perusteella. Cloud Native Computing Foundation (CNCF) vuoden 2020 tekemän kyselyn mukaan 92 % yrityksistä käyttää konttiteknologiaa ja yrityksistä 83 % valitsi Kuberneteksen palvelutuotannon käyttöön (CNCF SURVEY 2020), joka tekee siitä todella suositun teknologian palvelutuotannossa. Siksi on hyvä tietää, mitä välineitä kannattaa käyttää.

Tässä teoksessa tutkittiin ja vertailtiin viittä erilaista työkalua perin pohjin; kubectl, K9s, KDash, Prometheus/Grafana ja Kubernetes Dashboard ja mietitään niitä myös palvelutuotannon, ITIL:in ja DevOpsin näkökulmasta.

Tutkiessa työkaluja opinnäytetyöhön laadittiin niiden asennusohjeet ja listattiin niiden tärkeimmät ominaisuudet ja kuinka ne olisivat hyödyllisiä palvelutuotannon, ITIL:in ja DevOpsin näkökulmasta. Tutkimista varten konfiguroitiin käyttöön yksinkertainen, yksiklusterinen Kubernetes-ympäristö, johon oli pystytetty Prestashop-verkkokauppa, jotta hallinnointityökalut saisi käyttöön tällaisessa konkreettisessa ympäristössä. Työn teoriaosuuksien lähteiden etsimisessä hyödynnettiin Perplexity-tekoälyä.

2 Tutkimusasetelma

Tutkimusasetelmalla tarkoitetaan toimeksiantajasta, työn tavoitteista sekä kehittämishaasteista muodostunutta kokonaisuutta. Lopussa vielä käydään läpi työn luotettavuus sekä eettisyys.

2.1 Toimeksiantaja

Opinnäytetyön toimeksiantajana toimii WIMMA Capstone projektin edustaja Marko Rintamäki. Opinnäytetyön ohjelmia tarkasteltiin palvelutuotannon näkökulmasta ja opinnäytetyön tavoite oli kartoittaa menetelmiä ja välineitä ohjelmistopalveluhallinnan opetukseen.

2.2 Työn tavoitteet

Tavoitteena työlle oli löytää sopivin työkalu Kubernetes klusterin hallinnointiin tai yhdistelmä työkaluja. Työkaluista tutkittiin, että kuinka hyviä ne ovat vianmäärityksen, käytettävyyden, kustannusten, läpinäkyvyyden osalta, sekä kuinka paljon niiden käyttö kuormittaa järjestelmää muistin ja prosessorin käytön osalta. Tutkittiin myös, kuinka tehokkaasti työkalut auttavat käyttäjään ymmärtämään enemmän Kuberneteksen hallintaa, perusrakennetta ja palvelunhallinnan periaatteita. Palveluhallinnan näkökulmasta otetaan erityisesti huomioon ITIL-kehyksen mukaisia toiminnallisuuksia, kuten valvonta, häiriönhallinta ja kapasiteetin hallinta sekä DevOps-toimintamallin toiminnallisuuksia, kuten automaatio ja läpinäkyvyys.

2.3 Tutkimuskysymykset

Opinnäytetyö on tärkeä ja ajankohtainen, sillä aiheen osalta ei vielä löytynyt suomeksi juurikaan lähteitä, jossa pääasiassa keskityttäisi nimenomaan ohjelmien vertailuun ITIL:n, DevOpsin tai palvelutuotannon näkökulmista. Nykyään miltei kaikilla isommilla yrityksillä on oma IT-osasto. Tästä syystä on hyvä olla tietoinen siitä, mitkä ohjelmat yritykselle sopii parhaiten, kun mietitään palvelutuotantoon sopivaa kokonaisuutta. Tässä työssä vertailtiinkin ohjelmia seuraavien tutkimuskysymysten kautta:

1. Mitä tarkoittaa palvelutuotannon käsite ITIL ja DevOpsin näkökulmasta?
2. Mikä ohjelma sopisi minkäkin Kubernetes ympäristön hallintaan?

Kysymykset muovautuivat työn tavoitteiden pohjalta, jotka tulivat työn tilaajalta.

2.4 Luotettavuus ja eettisyys

Opinnäytetyön luotettavuuden varmistamiseksi menetelmän valinta ja työn suunnittelu perusteltiin huolellisesti. Työssä käytetty vertailumenetelmä rakennettiin siten, että se mahdollisti johdonmukaisen arvioinnin eri Kubernetes-hallinnointi- ja monitorointityökalujen välillä. Menetelmien valintaan vaikutti niiden soveltuvuus aiheen tekniseen kontekstiin sekä käytännön toteutettavuus työssä käytetyssä ympäristössä.

Tutkimuksen luotettavuutta arvioitiin toistuvuudella, objektiivisuudella, validiteetilla ja yhtenäisellä testiympäristöllä. Luotettavuus arvioitiin esimerkiksi toistettavuuden kannalta. Tutkimuksessa pyrittiin varmistamaan dokumentoimalla kaikki testausprosessit, käyttöönotetut työkalut ja ympäristön määrittelyt tarkasti. Luotettavuutta edisti myös objektiivisuus, jossa varmistettiin käyttämällä samoja kriteerejä kaikille arvioiduille työkaluille. Validiteettia tukee se, että arviointikriteerit valittiin suoraan työkalujen käyttötarkoitusta ja käytännön merkitystä tukien.

Työtä tehdessä otettiin myös huomioon ympäristön konfiguroinnin vaikutus työkalujen suorituskykyyn tai työkalujen version eroavaisuudet. Näiden vaikutuksia pyrittiin minimoimaan valitsemalla ohjelmistoista viimeisimmät stabiilit versiot ja testaukseen yhtenäinen testiympäristö.

Lähdeaineiston luotettavuus varmistettiin käyttämällä ensisijaisesti riippumattomia teknisiä julkaisuja sekä tunnettuja alan asiantuntijalähteitä. Kaikki käytetyt lähteet on arvioitu kriittisesti niiden ajantasaisuuden ja asiantuntijuuden perusteella. Joitain lähteitä karsittiin pois, sillä niissä niiden teoriaosuuksissa oli poikkeavuuksia muihin vastaaviin verrattuna.

Tekoälyä työssä hyödynnettiin lähteiden etsimisessä, kuitenkin niin, että ennen niiden käyttöä varmistettiin, että lähde vaikuttaa validilta vaihtoehdolta opinnäytetyön lähteeksi. Tekoälyn avulla myös pohdittiin, mistä näkökulmista työkaluja kannattaa lähteä tutkimaan.

3 Tietoperusta

3.1 ITIL

ITIL eli Information Technology Infrastructure Library on maailman käytetyin palveluhallinnan viitekehys, jonka tavoite on tuottaa kokonaisvaltainen toimintamalli palvelujen pystyttämiseen, kehittämiseen ja toimittamiseen. ITIL:n uusin versio ITIL v4 julkaistiin vuonna 2019. (ITIL 2023.)

ITIL on globaalisti tunnustettu viitekehys parhaita käytäntöjä ja malleja IT-alan johtamista varten. ITIL-kehyksen kehitys alkoi yli 20 vuotta sitten Englannissa valtiohallinnon hankkeena 1980-luvulla ja sittemmin sen kehitystä on ajansaatossa jatkettu ympäri maailman. (ITIL 2023.)

Nykyisin vuodesta 2013 lähtien, ITIL viitekehyksen omistaa Axelos, joka muodostuu kansainvälisestä yrityksestä nimeltä Capita sekä Britannian hallituskansliasta. Axelos ylläpitää ja päivittää ITIL-kokoelmaa sekä lisensoi yrityksille oikeuden käyttää ITIL:in immateriaalioikeuksia ja valtuuttaa yrityksiä antamaan sertifikaatteja ITIL osaamisesta ja sen soveltamisesta. (Kesanto 2021.)

ITIL ei ole standardi, jota täytyy noudattaa sellaisenaan, vaan se on kokoelma käytäntöjä, joista jokainen yritys ja organisaatio voi ottaa itselleen käyttöön ne mieluisimmat menetelmät. ITIL mallin avulla voi varmistua siitä, että asiakkaalle tuotettu palvelu on laadukasta ja stabiilia. Yksi ITIL-mallin tavoitteista on vähentää yritykselle aiheutuvia kustannuksia pitkällä tähtäimellä tekemällä prosessista yksinkertaisen ja tehokkaan. (Salo 2018, 2.)

3.1.1 ITIL v1

Ensimmäinen ITIL versio on kokoelma kirjoja parhaista palvelu- ja johtamiskäytännöistä. Kirjoja on julkaistu vuodesta 1989 lähtien, minkä jälkeen niitä on vuosien mittaan laitettu jakoon yhteensä 34 kappaletta. (Salo 2018, 2.)

3.1.2 ITIL v2

ITIL v2 julkaistiin vuonna 2000 ja tällä versiolla pyrittiin parantamaan sovellus- tietoturva ja infrastruktuurin hallintaa sekä tuki- ja toimitusprosesseja. Toisessa versiossa keskitytään palvelutuotannon osalta kysymykseen, mitä pitäisi tehdä palvelun parantamiseksi, kun taas ITIL v3 vastaa kysymykseen, miten palvelua parannetaan. (Salo 2018, 2–3.)

3.1.3 ITIL v3

ITIL v3 on julkaistu vuonna 2007 ja sitä on parannettu vuonna 2011. ITIL v3 koostuu viidestä eri kirjasta, jotka ovat:

- Palvelustrategia (service strategy)
- Palvelusuunnittelu (service design)
- Palvelutransitio (service transition)
- Palvelutuotanto (service operation)
- Palvelun jatkuva parantaminen (continual service improvement)

ITIL v3 on tiivistetty selkeäksi prosessikirjastoksi. ITIL:n versio 2 sisälsi kahdeksan kirjaa, ja ensimmäinen versio 34 kirjaa. Yritys, jota hyödyntää ITIL:iä käyttää sitä ottamalla käyttöön vain ne ohjeistukset, joita kokee kokoelmasta tarvitsevansa. ITIL avustaa parantamaan palvelujen laatua ja täyttämään ISO 20000 -laatustandardin vaatimukset. (Mts. 3–4.)

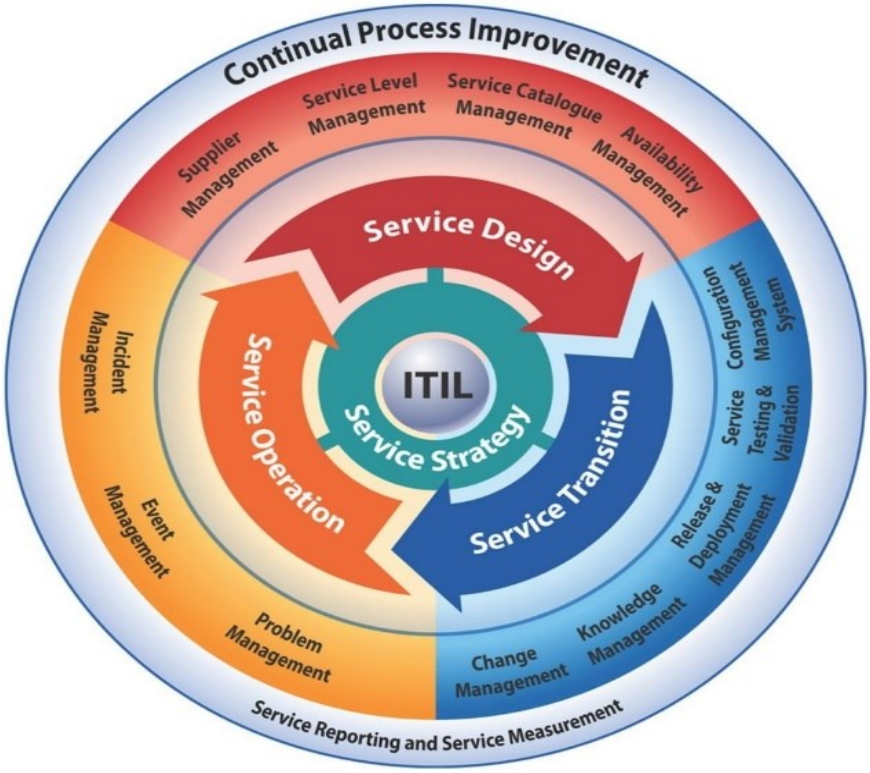
3.1.4 ITIL v4

ITIL:in versio 3 kuvaa palvelun elinkaarta, johon kuuluu 26 prosessia. ITIL 4 muuttaa näkökulmaa niin, että sitä voi paremmin sovittaa yhteen IT-operaatioiden ja modernin liiketoiminnan vaatimusten pohjalta. (The Evolution of ITIL: From ITIL v3 to ITIL 4 – What’s Changed and Why It Matters 2024)

ITIL v3:n prosessit ja funktiot on muutettu 34 käytäntöön kolmen kategorian alle: Yleisen hallinnan käytännöt (General management practices), palvelunhallinnan käytännöt (Service management practices) ja teknisen hallinnan käytännöt (Technical management practices). Muutoksessa tapahtuvat kaksi keskeistä muutosta, joista ensimmäisessä prosessit on muokattu ITIL 4:ään käytännöiksi, jotta ne olisivat helpommin muokattavissa organisaatioiden käyttötarpeisiin. Toisessa muutoksessa yksittäiset käytännöt on poistettu tietyistä vaiheista palvelun arvoketjua, jotta on voitu antaa tilaa käytännön tulkinnalle ja sille, että miten käytännöt soveltuisivat parhaiten tiettyihin tilanteisiin. (Kaikkonen 2021, 13.)

Vuosien saatossa ITIL on kehittynyt nykyiseen versioon 4, jossa korostetaan enemmän asiakkaan merkitystä, palvelun laatua ja arvon tuottamisen merkitystä kaikkien sidosryhmien kanssa (Kotala 2022, 11).

Katso ITILin rakennetta visualisoiva kuvio (ks. kuvio 1.) Vertailtavat ohjelmat sijoittuisivat ITIL-mallin kuvassa kohtaan Service Operation ja pääasiassa painottuu Incident/Event Managementiin.



Kuvio 1. ITIL-mallin rakenne (What is ITIL? 2013)

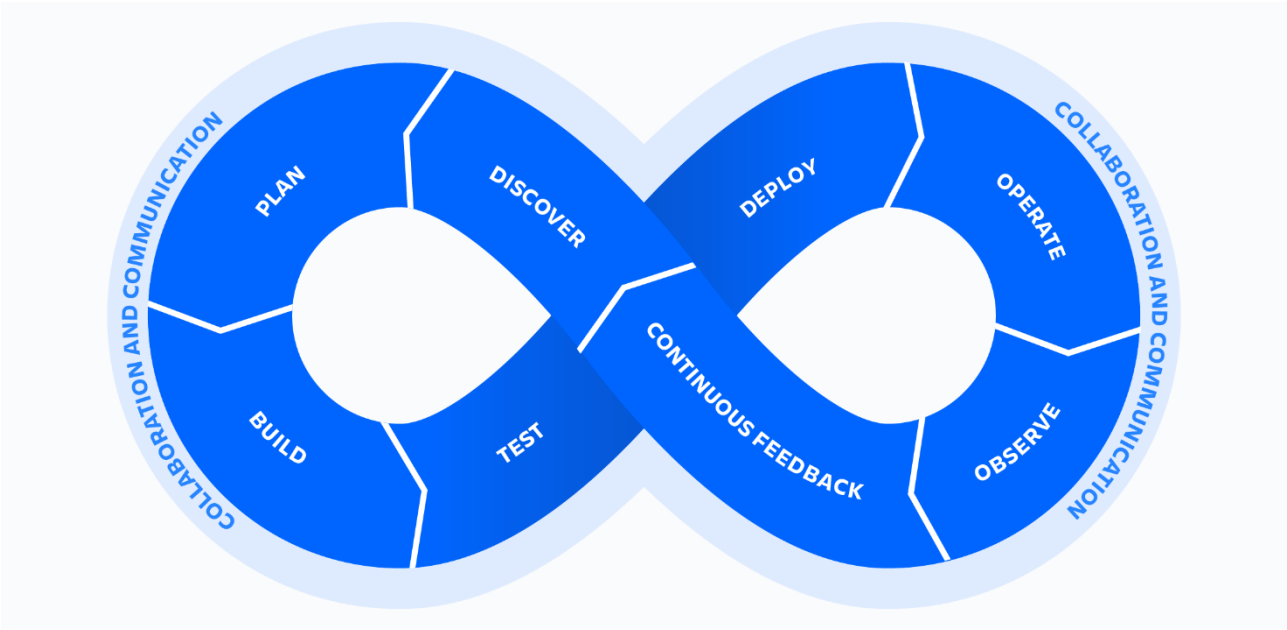
3.2 DevOps

DevOps on menetelmä IT-toiminnoille, erityisesti ohjelmistokehitykselle, jossa automatisoidaan työnkulkua yhdistämällä kehityksen ja toiminnot. DevOps lyhenne tulee sanoista Development, eli kehitys ja operations eli operatiiviset toiminnot. DevOps on toimintamalli, jonka perusperiaate onkin siis sulauttaa kehitys ja toiminnat tiiviiksi paketiksi. (Mitä tarkoittaa DevOps? 2024.)

DevOps on myös osa ITIL:iä ja aihetta, jonka takia sen ymmärtäminen on olennaista. DevOps on digitaalisten palveluiden kehittämiseen ja tuottamiseen tarkoitettu toimintamalli, joka perustuu ketteriin menetelmiin, jatkuvaan integraatioon ja toimitukseen. Mallissa korostuvat myös testausprosessien sekä käyttöympäristöjen konfiguroinnin automatisointi, mikä mahdollistaa tehokkaamman ja sujuvamman kehitystyön. Tyypillisesti ohjelmistokehitys etenee toistuvina vaiheina, joihin kuuluvat suunnittelu, koodaus, kääntäminen, testaus, julkaisun valmistelu ja asennus, tuotantokäyttö sekä järjestelmän jatkuva seuranta. (Winter n.d; DevOps n.d.)

DevOps-malli tähtää näiden vaiheiden mahdollisimman pitkälle vietyyn automatisointiin erityisesti kehityksen, testauksen ja ylläpidon osalta. Sen avulla ohjelmiston rakentaminen, testaus ja julkaisu voidaan toteuttaa nopeasti ja luotettavasti. DevOpsin hyödyntämisestä on raportoitu monia etuja, kuten korkeampi asiakastyytyväisyys, laadukkaammat tuotteet, vakaammat julkaisuprosessit sekä nopeampi kyky tuoda uusi ratkaisu testattavaksi. (DevOps n.d.)

DevOpsia pystyy havainnollistamaan hyvin siitä tehdyn kuvion avulla. (Ks. kuvio 2.) Työkalujen, kuten Kubectl ja K9s pääpaino DevOps-mallissa on operate-kohdassa, KDashin sekä Prometheus/Grafanan pääpaino on kohdassa Observe ja Kubernetes Dashboard sijoittuu sekä Operate että Observe-akselille.



Kuvio 2. DevOps-malli

3.3 DevOps ja ITIL

DevOps ja ITIL eivät sulje toisiaan pois. Ne voivat olla toisiaan täydentäviä lähestymistapoja – molemmat tuovat mukanaan omat etunsa. Ketteryys ja yhteistyö. Prosessi ja hallinta. Yhdistetty lähestymistapa voi hyödyntää molempien vahvuuksia. (DevOps vs. ITIL -- Which matters for your team? 2024.)

ITILiä ja DevOpsia voidaan hyödyntää rinnakkain, ja parhaimmillaan niiden yhdistäminen auttaa yrityksiä löytämään tasapainon perinteisten toimintamallien ja modernin ketteryyden välillä. Kun ymmärrämme näiden lähestymistapojen tavoitteet ja otamme ne käyttöön tarkoituksenmukaisesti, voi koko liiketoiminta hyötyä.

ITIL ja DevOps tähtäävät lopulta samoihin päämääriin – tehokkuuden lisäämiseen ja yhteistyön parantamiseen. Jotta nämä menetelmät toimisivat saumattomasti yhdessä, on tärkeää sopeuttaa ITIL vastaamaan DevOpsin toimintaympäristöä ja samalla karsia pois kaikki toiminnan sujuvuutta hidastavat tekijät. (Hoogenraad 2022.)

Kuten parhaiten menestyvät tiimit tietävätkin, IT-ala vaatii sekä ITIL:iä, että DevOpsia. (DevOps vs. ITIL -- Which matters for your team? 2024.)

3.4 Palvelutuotanto

Palvelutuotanto tarkoittaa organisoitua toimintaa, jonka tavoitteena on tuottaa palveluita tai palvelutuotteita sekä täyttää palvelusopimusten velvoitteet. Se on interaktiivinen prosessi, jossa palvelun tuottaja ja asiakas toimivat vuorovaikutuksessa. (Palveluliiketoiminnan sanasto 2010, 10.)

Palvelutuotanto on vaihe, jossa organisaation strategiset tavoitteet konkretisoituvat käytännön toimiksi. Siinä huolehditaan palveluiden ja niitä tukevan teknologian ylläpidosta ja hallinnasta sekä kerätään tietoa päätöksenteon tueksi. Palvelutuotantoon kuuluu useita prosesseja, kuten herätteiden hallinta, häiriönhallinta, ongelmanhallinta, palvelupyyntöjen käsittely ja pääsynhallinta. Lisäksi siihen sisältyvät keskeiset toiminnot, kuten palvelupiste, tekninen hallinta, IT-käyttöpalvelun hallinta ja sovellushallinta. (Haara 2016, 14.) Työssä keskityttiin ITIL-prosesseista hyvin vahvasti nimenomaan palvelutuotannon prosesseihin.

Palvelun hallinnointi ja stabiili olemus ovat yksi keskeisimmistä asioista palvelutuotannossa. Palvelutuotannon häiriöt voivat heijastua asiakkaille huonolaatuisena tai epäjohdonmukaisena palveluna. Asiakastyytyväisyyttä pidetään tärkeänä mittarina palveluprosessien suorituskyvyn arvioinnissa (Saviranta 2019, 1) ja tämä tukee myös ITIL version 4 periaatteita.

4 Kontit ja Kubernetes

Tässä työn vaiheessa keskitytään konttiteknologian historiaan, Kubernetekseen sekä sen yleisimpiin käsitteisiin. Käsitteet on hyvä sisäistää jokaisen, joka mielii työskennellä Kuberneteksen parissa. Lopuksi kerrotaan vielä itse Microk8s:stä

Kuberneteksesta puhuttaessa kontit on kaiken tämän pohja. Kontti tarkoittaa kokoelmaa komponentteja, joita vaaditaan jonkin sovelluksen ajamiseen (Wallenius 2022a). Kun tässä yhteydessä pakataan kokoelma kirjastoja, sovelluksia ja konfiguraatiotiedostoja yhteen paikkaan voidaan puhua kontituksesta. Kontteja voi käyttää eri ympäristöissä eli ne toimivat eri käyttöjärjestelmillä ja sekä testaus- että tuotantoympäristössä samalla tavalla. Tämä ratkaisu poistaa tutun ongelman,

jossa esim. jokin ohjelma toimii yhdessä ympäristössä, mutta ei toisessa. Kontit ovat huomattavasti kevyempiä kuin virtuaalikoneet ja niitä on erittäin helppo siirtää eri ympäristöjen välillä. (Leskinen 2019, 14.)

Kaikilla konteilla on oma verkko, muisti ja prosessit, mutta niitä kuitenkin yhdistää yksi yhteinen käyttöjärjestelmä. Kontteja voi ajaa sekä pilvessä, virtuaalikoneessa tai suoraan fyysisellä tietokoneella. Konttien vaatimat resurssit kuitenkin otetaan siltä käyttöjärjestelmältä, minkä päälle ne on asennettu. (Ezinne 2023.)

Konttiorkestrointi taas on prosessi, jossa ympäristöön otetaan käyttöön useita kontteja, joita hallinnoidaan jollain konttiorkestraation hallinnointityökalulla. Työkalun avulla automatisoidaan konttien käyttöönotto, jolloin prosessia ei tarvitse itse manuaalisesta tehdä. Erityisen hyödyllisiä nämä ovat isoissa organisaatioissa, joissa voidaan ottaa käyttöön tuhansia kontteja päivässä. (Ezinne 2023.) Tällaisia hallinnointityökaluja ovat muun muassa Docker sekä Kubernetes.

Konttien kehitys otti ensiaskeleensa vuonna 1979, kun Unixille suunnattua ohjelmaa nimeltä chroot (change root) alettiin kehittämään. Tällä ohjelmalla konttien kaltainen teknologia lähti liikkeelle, sillä chrootin avulla prosessi voitiin eristää sen juurihakemistoa muuttamalla. Chroot julkaistiin Unix 7:lle vuonna 1982. (Marquez 2023.)

Seuraava askel konttiteknologiassa tapahtui noin 20 vuotta myöhemmin vuonna 2000, kun FreeBSD julkisti jail-komennon heidän käyttöjärjestelmäänsä. Ominaisuus oli hyvin paljon chrootin kaltainen, se lisäksi tarjosi myös tavan eristää tiedostojärjestelmiä, käyttäjiä, verkkoja jne. Jailin avulla pystyi eristetylle ympäristölle antamaan IP-osoitteen, konfiguroimaan kustomoidun sovellusasennuksen ja tekemään muutoksia jokaiseen jailiin. (mt.)

Vuonna 2004 julkaistiin Solaris-kontit, jotka loivat täydellisiä sovellusympäristöjä Solaris Zones -tekniikan avulla. Vyöhykkeiden (zones) avulla sovellukselle voitiin antaa täysi käyttäjä-, prosessi- ja tiedostojärjestelmätila sekä pääsy laitteiston resursseihin. Sovellus näkee kuitenkin vain sen, mikä on sen oman vyöhykkeen sisällä. (mt.)

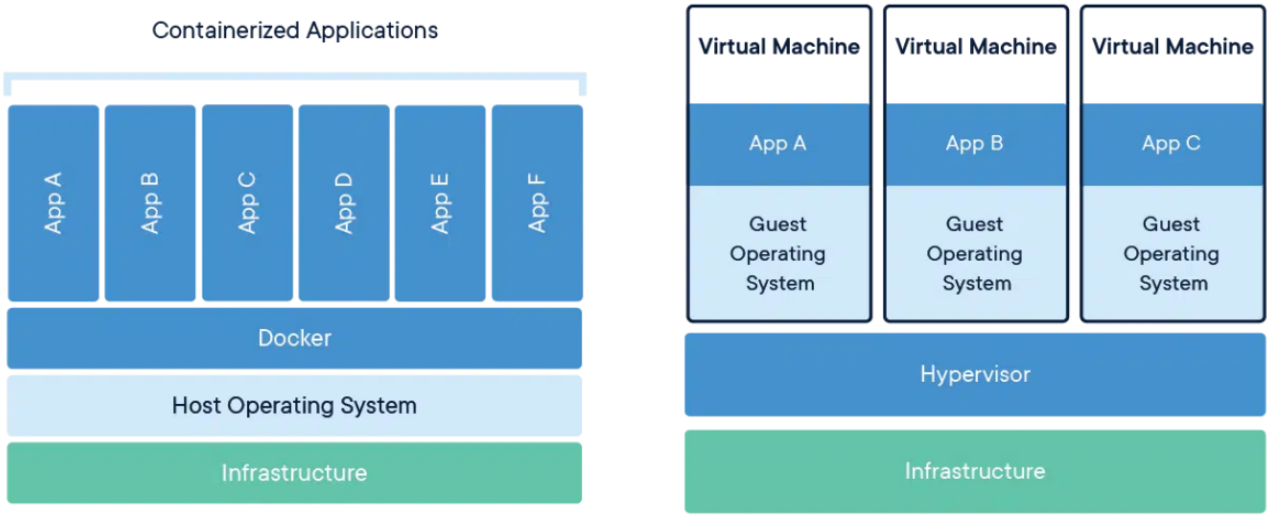
Vuonna 2006 Googlen insinöörit ilmoittivat kehittäneensä prosessikontteja, joiden tarkoituksena oli eristää prosesseja ja rajoittaa niiden resurssien käyttöä. Vuonna 2007 nämä prosessikontit nimettiin uudelleen ohjausryhmiksi (control groups, cgroups), jotta ne eivät sekoittuisi kontti-sanaan. (mt.)

Vuonna 2008 cgroups liitettiin Linuxin ytimeen versiossa 2.6.24, mikä johti LXC-projektin (Linux Containers) syntyyn. LXC mahdollistaa käyttöjärjestelmätason virtualisoinnin, jossa useat eristetyt Linux-ympäristöt (kontit) voivat toimia yhteisen Linux-ytimen päällä. Jokaisella kontilla on oma prosessi- ja verkkotilansa. (Marquez 2023.)

Vuonna 2013 Google muutti konttien kehitystä jälleen julkaisemalla avoimen lähdekoodin konttipinonsa nimellä Let Me Contain That For You (LMCTFY). LMCTFY:n avulla sovellukset voitiin ohjelmoida ymmärtämään konttiympäristöjä ja luomaan sekä hallinnoimaan omia alikonttejaan. (mt.) Yksi tunnetuimmista konttien hallintatyökaluista on Docker, avoimen lähdekoodin sovellus, joka julkaistiin vuonna 2013. Dockerin keskeinen idea on, että kehittäjä voi pakata kaikki sovelluksen tarvitsemat kirjastot ja riippuvuudet yhdeksi kokonaisuudeksi, jota kutsutaan Docker-kontiksi tai Docker-imageksi. (Leskinen 2019, 18.)

Työ LMCTFY:n parissa lopetettiin vuonna 2015, kun Google päätti siirtää sen keskeiset ideat Docker-projektin libcontainer-komponenttiin (Marquez 2023). Merkittävä syy konttien käytölle on myös niiden kustannustehokkuus. Ne käyttävät todella vähän laskentatehoa, säästää energiaa ja sitä myötä rahaa organisaatioissa.

Ilman konttiteknologiaa joutuisimme kaikille sovelluksille ottamaan käyttöön oman virtuaalikoneen. Testien perusteella virtuaalikoneella ajettu sovellus kuluttaa jopa 30 % enemmän resursseja, kuin sama sovellus kontissa ajettuna. Docker-kontti vie myös vain joitain prosentteja levytilaa virtuaalikoneeseen verrattuna. Sekä kontti, että virtuaalikone ovat kummatkin pohjimmiltaan tiedostoja. (Wallenius 2022a.) Alla näkyvässä kuvassa näkyy docker-kontin ja virtuaalikoneen ero. (Ks. kuvio 3.)



Kuvio 3. Kontin ja virtuaalikoneen ero. Vasemmalla kontti ja oikealla virtuaalikoneilla luotu ratkaisu (Use containers to Build, Share and Run your applications 2025)

4.1 Kubernetes-palvelinohjelmisto

Kubernetes on yksi monista olemassa olevista säiliöiden hallinnointiin tarkoitetuista sovelluksista. Se on Googlen kehittämä vuonna 2015 julkaistu avoimen lähdekoodin työkalu, joka on suunniteltu automatisoimaan useiden kontteihin pakattujen sovellusten ja palveluiden hallintaan liittyviä tehtäviä. Se mahdollistaa mm. palveluiden helpon löydettävyyden, liikenteen tasaisen kuormituksen jakamisen, erityyppisten tallennusratkaisujen orkestroinnin sekä sovellusten käyttöönoton ja päivitysten automatisoinnin. Lisäksi Kubernetes huolehtii automaattisesta palautumisesta virhetilanteissa ja tarjoaa tukea salaisten tietojen, kuten salausavainten, käyttäjätietojen ja muun konfiguraation turvalliseen hallintaan. (Tonteri 2022, 14.)

Google käytti konttiteknologiaa sisäisesti jo pitkään ennen kuin siitä muodostui vakiintunut standardi alalla. Yritys kehitti omien ratkaisujensa pohjalta Kubernetes-nimisen konttien hallintaohjelmiston, jonka se vuonna 2014 lahjoitti avoimen lähdekoodin projektina Cloud Native Computing Foundationille. Sittemmin Kubernetes on noussut maailman suosituimmaksi konttiorkestraattoriksi. (Tonteri 2022, 8.)

4.1.1 Kuberneteksen käyttötarkoitukset

Kun useasta kontista koostuva sovellus halutaan ottaa käyttöön tuotantoympäristössä, on ensiarvoisen tärkeää tietää, mitä kontteja sovelluksen toiminta edellyttää. Kubernetes toimii tässä yhteydessä keskeisenä hallintatyökaluna: sille määritellään tarvittavat kontit, ja se huolehtii näiden käynnistämisestä automaattisesti, kun sovellus otetaan käyttöön. (Wallenius 2022a.)

Koska tuotantoympäristöt ovat yleensä hajautettuja useille palvelimille, konttien sijoittaminen oikeille palvelimille vaatii älykästä hallintaa. Kubernetes vastaa myös tästä – se arvioi järjestelmän tilan ja päättää, mille palvelimille kontit kannattaa sijoittaa, jotta resurssit käytetään tehokkaasti ja palvelu pysyy saatavilla. (mt.)

Konttien ajon hallinta ei pääty niiden käynnistämiseen. Kun kontteja on ajossa useilla palvelimilla, on tärkeää säilyttää ajantasainen tieto siitä, mitä kontteja ajetaan missäkin. Kubernetes ylläpitää keskitetysti tietoa koko klusterin konttitilanteesta, mahdollistaen hallitun ja läpinäkyvän ympäristön valvonnan. (mt.)

Lisäksi Kubernetes tarjoaa automaattisia valvontatoimintoja, joiden avulla voidaan seurata konttien toimintaa. Mikäli jokin kontti lakkaa toimimasta, Kubernetes pystyy käynnistämään sen uudelleen automaattisesti ilman käyttäjän toimenpiteitä, mikä parantaa järjestelmän vikasietoisuutta. (Wallenius 2022a.)

Kubernetes mahdollistaa myös sovelluksen kuormituksen mukaisen skaalaamisen. Jos järjestelmään kohdistuu hetkellinen kuormitushuippu, se voi lisätä konttien määrää automaattisesti ja jakaa kuorman uusien yksiköiden kesken, varmistaen tasaisen suorituskyvyn. (mt.)

Sovellusten ylläpito ja päivitykset helpottuvat merkittävästi Kubernetesin avulla. Konttien päivitys voidaan toteuttaa hallitusti, ja mikäli uusi versio ei toimikaan odotetusti, järjestelmä voidaan palauttaa nopeasti edelliseen, toimivaan versioon. (mt.)

Lopuksi Kubernetes vastaa myös verkkoasetusten hallinnasta, kuten IP-osoitteiden jakamisesta konteille sekä sovelluksen julkaisemisesta käyttäjille. Tämä mahdollistaa sen, että käyttäjät pääsevät käsiksi sovellukseen luotettavasti ja turvallisesti. (Wallenius 2022b.) Nämä ominaisuudet tekevät Kubernetes-ympäristöstä loistavan ratkaisun palveluiden pyörittämiseen.

4.2 Kubernetes-järjestelmän arkkitehtuuri

Kuberneteksen hallintatyökaluissa on paljon ominaisuuksia ja yksi niistä on mahdollisuus seurata erilaisia Kubernetes-ympäristön osia, jotka ovat kriittinen osa klusterin toimivuutta. Kubernetesen parissa työskentelevän henkilön on elintärkeää tietää, mitä mikäkin osa Kubernetesta tekee, jotta voi täysin ymmärtää sen toimintaa ja jotta ohjelmistoa voisi operoida sujuvasti. Seuraavaksi työssä selitetään auki eri komponentteja, joita jokaisesta Kubernetes-järjestelmästä tyypillisesti löytyy.

4.2.1 Master, worker node ja controller planen rakenne

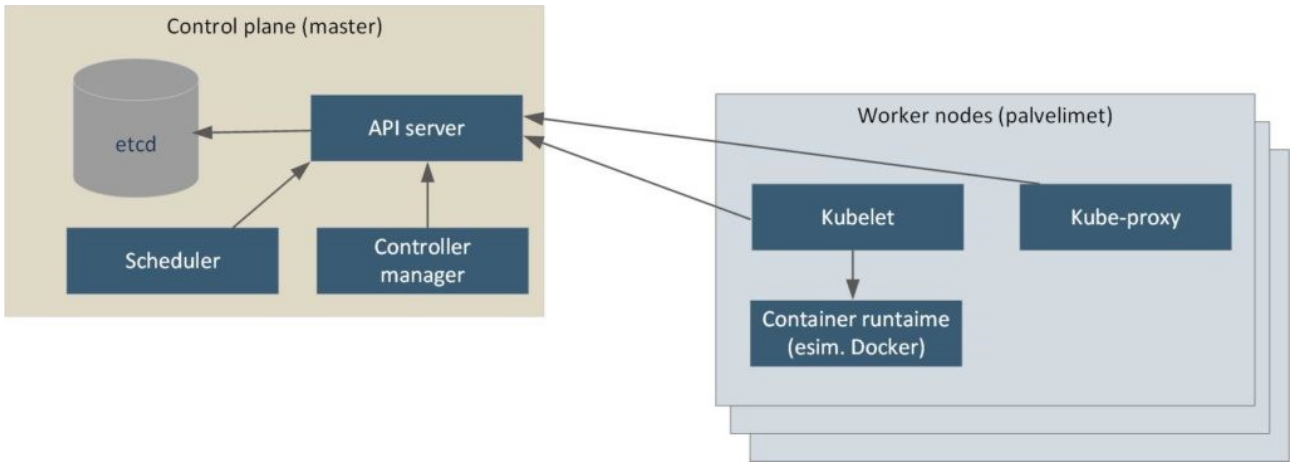
Kubernetes master toimii klusterin älynä, joka päättää, missä palvelimilla eri sovellukset ajetaan. Sovelluskehittäjä toimittaa masterille määritelmän tarvittavista konteista, minkä jälkeen master huolehtii niiden optimaalisesta sijoittelusta palvelimille. (Wallenius 2022b.) Klusterissa on myös aina vähintään yksi Worker Node, joka vastaa varsinaisten kontitettujen sovellusten suorittamisesta. Worker Nodeja kutsutaan yleisesti vain Nodeiksi (Tonteri 2022, 8).

Controller plane -nodet muodostavat joukon tärkeitä osia, joita ovat etcd, API-palvelin, Scheduler ja Controller Manager. API-palvelin, joka toimii Kubernetes-järjestelmän hallintakeskuksena. Näiden lisäksi controller planessa on ohjauskomponentteja, jotka vastaavat worker nodejen koordinoinnista sekä klusterin kokonaistilan ylläpidosta. (Mikä on Kubernetes ja miten se toimii? 2024.)

Toinen tärkeä osa-alue on etcd, hajautettu avain-arvo-tietokanta, johon tallennetaan klusterin nykyinen tila. Lisäksi Controller Manager sisältää erilaisia ohjauskomponentteja, jotka huolehtivat klusterin tilan valvonnasta ja koordinoivat worker-solmujen toimintaa. (Mikä on Kubernetes ja miten se toimii? 2024.)

Scheduler osoittaa sovelluksille paikat eri palvelimilta tarpeen ja tilanteen mukaan. Nimi onkin siis hieman harhaanjohtava, sillä kyse ei ole ajastimesta. Kubelet on komponentti, joka mahdollistaa työkuormia ajavan palvelimen ja klusterin hallintatasoa edustavan masterin välisen viestinnän. Se huolehtii siitä, että kontit ajetaan määritellyn tilan mukaisesti kyseisellä nodella. (Wallenius 2022b.)

Kube-proxy toimii väylänä, kun sovellus, joka pyörii palvelimella, haluaa kommunikoida klusterin ulkopuolisten järjestelmien kanssa. Sen tehtävänä on julkaista palvelu klusterin ulkopuolelle sekä huolehtia verkkoliikenteen reitityksestä ja kuormantasauksesta. Container Runtime, kuten esimerkiksi Docker Engine, vastaa konttien suorittamisesta palvelimella. Se on se osa, joka varsinaisesti käynnistää ja ajaa kontit. (Wallenius 2022b.) Katso kuvio 4 Kuberneteksen pääkomponenteista.



Kuvio 4. Kubernetesin pääkomponentit (Wallenius 2022b)

4.2.2 Klusteri

Klusteri on Kubernetes-ympäristössä kokoelma fyysisiä tai virtuaalisia tietokoneita eli nodeja, jotka toimivat ikään kuin ne olisivat yksi laite. Jokaisessa klusterissa on vähintään yksi Control plane sekä worker node. (Tonteri 2020, 15.)

4.2.3 Node

Kuten ylempänä määriteltiin, Nodet ovat klusterissa olevia tietokoneita. Ne voivat olla joko fyysisiä koneita tai virtuaalikoneita.

4.2.4 Pod

Pod on Kuberneteksen pienin yksikkö, jonka sisällä on yksi tai useampi kontti, joilla on sama suoritusympäristö, mutta jokaiselle on kuitenkin varattu omat klusterin resurssit, joista vastata. Jokaisella kontilla on yksi yhteinen ulkoinen IP-osoite ja mikäli konttien on keskusteltava keskenään, tapahtuu se localhost-rajapinnan kautta. Podin elinkaari koostuu niiden luomisesta, käyttöönotosta ja tuhoamisesta eli mikäli pod kaatuu, tätä samaa podia ei yritetä elvyttää, vaan se tuhotaan ja luodaan tämän tilalle uusi niin, että kaikki vanhan podin sisältämät kontit ja konfiguraatiot kuitenkin säilyvät. (Tonteri 2022, 18.)

4.2.5 Deployment/Replicasets

Kuberneteksessa deployment on korkeamman tason resurssi, joka vastaa podin ylläpidosta ja päivityksistä. Deployment käyttää Replicasettiä podien hallintaan ja hallinnoi tätä automaattisesti ja Replicaset on se, joka klusterissa luo uusia podeja vanhojen tilalle. Tätä kautta Replicaset siis tarkkailee podien tilaa. (ReplicaSet 2025.)

4.2.6 Deployment controller/Deployment Manifest/PodTemplateSpec

Deployment vastaa siitä, että podien tila on se, mikä on määritelty Deployment manifestissa. Jos määritetyssä tilassa pitäisi olla ajossa kaksi podia, mutta onkin vain yksi, Deployment controller käynnistää toisen podin. Sama pätee myös, jos podeja on liikaa. (Tuomala 2020, 20.) PodTemplateSpec keskittyy yksittäisen podin määrittelyyn. Nämä määritelmät sisältävät esimerkiksi levykuvat ja portit. Deployment Manifest taas sisältää määritellyt mallit, jonka mukaan podeja luodaan. (Deployments 2025.)

Deployment Manifest on YAML-tiedosto, joka sisältää koko Deploymentiin liittyvät määrittelyt ja se hallinnoi koko sovelluksen elinkaarta, mukaan lukien skaalaus ja päivitysstrategiat. Se sisältää PodTemplateSpecin lisäksi muita kenttiä, kuten:

- apiVersion ja kind (apiVersion kertoo Kubernetes API:n version ja kind kertoo resurssin tyypin, kuten Deployment)
- metadata (nimi, nimiavaruus, tunnisteet)
- spec (replikoiden määrä, päivitysstrategia)

- selector (määrittelee, mitkä podit kuuluvat Deploymentiin). (8 Kubernetes Deployment Strategies 2025.)

4.2.7 Service

Jokaisella Podilla on oma henkilökohtainen IP-osoite. Kun podi kaatuu ja tilalle tulee podi, tulee tilalle myös uusi IP. Tämän takia podien kanssa liikenteen vaihtaminen ei ole mielekästä, jonka johdosta Kuberneteksessa on olemassa Service, jonka IP pysyy samana. Jokaisella podilla siis on oma Service, joka mahdollistaa liikenteen lähettämisen ja vastaanottamisen tämän kautta sekä ulkoisesta verkosta, että lähiverkosta. Service on Kuberneteksessa myös toimiva kuormantasaaja, joka jakaa verkkoliikennettä podien välillä. (Janashia 2020.)

4.2.8 ConfigMap ja Secret

ConfigMap ja Secret toimivat hyvin samalla tavalla, mutta ne sisältävät hieman erilaista dataa. ConfigMap on olemassa sitä varten, että sinne on tarkoitus tallentaa dataa, jotka ei ole tietoturvan kannalta kriittistä, kuten ympäristömuuttujia ja kokonaisia konfiguraatiotiedostoja. Secret:iin taas tallennetaan sensitiivistä dataa, kuten salasanoja ja token-avaimia. (Tuomala 2020, 20-21.)

4.2.9 StatefulSet

StatefulSet on suunniteltu hallitsemaan tilallisia sovelluksia ja sen avulla pystytään sammuttamaan ja käynnistämään Podit tietyssä järjestyksessä. StatefulSet asettaa jokaiselle podille yksilöllisen vakaan, pysyvän tallennustilan ja verkkoidentiteetin vaikka podit käynnistettäisiinkin uudelleen. (StatefulSets 2025.)

4.2.10 DaemonSets

DaemonSet on taas yhdenlainen podien ohjaaja. Sen tehtävä on varmistaa, että tietty pod ajetaan jokaisessa, tai tietyssä nodessa klusterin sisällä. Se automaattisesti lisää podeja uuteen nodeen ja poistaa niitä poistetusta nodesta. DaemonSet on oiva työkalu lokien keräämiseen sekä klusterin monitorointiin. (Melnyk 2025.)

4.2.11 Ingress

Ingress mahdollistaa HTTP- ja HTTPS-liikenteen klusterin ulkopuolelta klusterin servicelle. Verkkoliikenteen reititys riippuu Ingress resurssien säännöistä. Ingressin voi konfiguroida antamaan Serviceille ulkoisen URL:n, toimia load balancerina klusterille, mutta myös reitittimelle tai frontendille, se mahdollistaa TLS-terminaation ja nimipohjaisen virtuaalisen hostauksen. (Ingress 2024.)

4.3 Microk8s

MicroK8s on avoimen lähdekoodin järjestelmä, joka automatisoi säilöttyjen sovellusten käyttöönoton, skaalaamisen ja hallinnan. Se tarjoaa ydinkomponenttien Kubernetes-toiminnallisuudet pienessä koossa, ja sen voi laajentaa yhdestä solmusta korkean käytettävyyden tuotantoklusteriksi. Microk8s:n käyttöön vaaditaan Linux-käyttöjärjestelmän, mutta se toimii myös epäsuorasti Windowsilla sekä Macilla. (MicroK8s documentation – home 2024; anaqvi 2019.)

Microk8s tarjoaa valtavan määrän hyötyjä perinteiseen Kubernetekseen nähden. Microk8s vähentää Kubernetesin pyörittämiseen vaadittavia resursseja erinäisten ominaisuuksien avulla muuttaen Kubernetesin kevyeksi kehitystyökaluksi, mahdollistaen Kubernetesin käytön minimaalisissa ympäristöissä, kuten GitHub CI:ssä ja mukauttaen Kubernetesin pienlaitteiden IoT-sovelluksiin. (MicroK8s documentation – home 2024.)

Kehittäjät käyttävät MicroK8s:ää edullisena kokeiluympäristönä uusille ideoille. Tuotannossa ohjelmistotoimittajat hyötyvät pienemmistä ylläpitokustannuksista ja resurssivaatimuksista sekä lyhyemmistä kehityssykleistä, mikä mahdollistaa laitteiden nopeamman toimituksen kuin koskaan aiemmin. (mt.) Microk8s tarjoaa myös valtavat määrän lisäosia, joita useampaa hyödynnettiin myös Prestashop-verkkokaupan käyttöönotossa.

Jotta Microk8s toimisi, täytyy tietokoneesta löytyä snap-ohjelma, joka on Canonicalin kehittämä paketinhallintaohjelma. Ubuntu sisältää Snap-ohjelman esiasennettuna. Itse Microk8s asennetaan Ubuntuun komennolla ”sudo snap install microk8s --classic”. ”--classic”-lisäys vaaditaan, jotta käyttäjä antaa ohjelmalle laajemmat pääsyn järjestelmään, mikä mahdollistaa sen, että ohjelma toimisi oikein.

5 Ohjelmien tarkastelu ja arvio

Tässä kappaleessa vertailtiin hallinnointityökaluja keskenään. Kappaleessa vertailtiin seuraavia ohjelmia: komentorivipohjaista kubectl-hallinnointityökalua, terminaalipohjaista k9s-monitorointisovellusta, visuaalista kdash-monitorointisovellusta, Prometheusin ja Grafanan muodostamaa valvonta- ja visualisointikokonaisuutta sekä selaimessa toimivaa visuaalista hallinnointityökalua Kubernetes Dashboardia.

Työhön valittujen ohjelmien etsimisessä hyödynnettiin ChatGPT-tekoälyä sekä eri hakukoneita ja ohjelmat valikoituivat hakutulosten, suosion sekä niiden oli oltava käytettävissä joko terminaalipohjaisina tai selaimessa erillisellä tietokoneella.

Hallinnointityökalun ja monitorointityökalun erona on se, että hallinnointityökalussa on mahdollisuus muokata muun muassa YAML-tiedostoja tai esimerkiksi sammuttaa, käynnistää ja poistaa komponentteja manuaalisesti, kun taas monitorointityökalussa keskitytään ainoastaan visuaalisuuteen ja ympäristön tarkkailuun.

Aiemmin määriteltyjä ominaisuuksia etsittiin ja tarkasteltiin, kuinka helpoksi ja selkeäksi niiden monitorointi on tehty. Ensin aloitettiin kertomalla ohjelmasta yleisesti, jonka jälkeen tuli asennusohjeet Ubuntulle. Testauksessa hyödynnettiin Ubuntua, sillä se on yksi suosituimmista Linux-käyttöjärjestelmistä. Testauksessa hyödynnetiin Ubuntun versiota 24.04 ja virtuaalikoneen isäntäpalvelimenä toimi cPouta, jonka kautta virtuaalikoneet pyörivät ja josta avattiin portit sovelluksia varten.

5.1 Kubectl

Kubectl on ensisijainen Kubernetes-klusterin hallintaan tarkoitettu komentorivityökalu, joka mahdollistaa kommunikoinnin Kubernetes-klusterin ohjaustason kanssa käyttäen Kubernetes API:a (Command line tool (kubectl) 2024). Kubectl on hyvä työkalu käytännön oppimiseen, mutta ei tarjoa yhtä helppoa tapaa oppia ymmärtämään klusterin toimintaa kokonaisuudessaan. Kubectl:n on kuitenkin terminaalipohjainen, mikä tekee tästä työkalusta hyvin kevyen käyttää.

Microk8s:ää käytettäessä on otettava huomioon, että Microk8s:lla on oma sisäinen paketti kubectl-ohjelmalle, jolloin, kun halutaan käyttää kubectl:ää, pitää joka komento aloittaa kirjoittamalla ”microk8s kubectl” tai vaihtoehtoisesti voi luoda aliaksen, jolloin tätä ei joka komennossa tarvitse huomioida. Tässä testauksessa on hyödynnetty alla näkyvää aliasta.

alias kubectl=’microk8s kubectl’

5.1.1 Kubectl asentaminen

Kuten edellisessä kappaleessa oli mainittu, asentaessa Microk8s:ää kubectl tulee siinä mukana, joten erillistä asentamista ei tarvinnut tehdä. Muutoin asentaminen tapahtuu komennolla:

```
sudo snap install kubectl --classic
```

5.1.2 Kubectl toiminnot

Kaikki monitorointi ja hallinta siis tapahtuu Linuxin komentorivissä ja halutut tiedot klusterista tulostuu sellaisenaan näytölle. Kubectl:stä esimerkiksi löytyy komento, jolla näkee podien tilan, uudelleen käynnistyskerrat sekä iän. Ilman Microk8s:ää komento olisi täysin sama, mutta ”microk8s” jätettäisiin pois. (Ks. kuvio 5.)

```
ubuntu@k8s:~$ microk8s kubectl get pods
NAME                                                  READY   STATUS    RESTARTS      AGE
grafana-k8s-monitoring-alloy-logs-xj8n6               2/2     Running   4 (14h ago)   4d2h
grafana-k8s-monitoring-alloy-metrics-0                2/2     Running   4 (14h ago)   4d2h
grafana-k8s-monitoring-alloy-profiles-6mwk8           2/2     Running   4 (14h ago)   4d2h
grafana-k8s-monitoring-alloy-receiver-gbdtf           2/2     Running   4 (14h ago)   4d2h
grafana-k8s-monitoring-alloy-singleton-c8bddccc7-xk5nk   2/2     Running   4 (14h ago)   4d2h
grafana-k8s-monitoring-beyla-2zc4g                    1/1     Running   2 (14h ago)   4d2h
grafana-k8s-monitoring-kepler-fl65p                   1/1     Running   2 (14h ago)   4d2h
grafana-k8s-monitoring-kube-state-metrics-58b567d5bb-2q5lc   1/1     Running   2 (14h ago)   4d2h
grafana-k8s-monitoring-node-exporter-fxrqh            1/1     Running   2 (14h ago)   4d2h
grafana-k8s-monitoring-opencost-78754ff764-whcfs      1/1     Running   2 (14h ago)   4d2h
my-release-mariadb-0                                  1/1     Running   2 (14h ago)   4d2h
my-release-prestashop-bfc9cbf4b-7k6l5                 1/1     Running   2 (14h ago)   4d2h
```

Kuvio 5. Kubectl:n avulla tulostettu näkymä podien tilasta

Tarkempi lista Kubectl:ssä käytetyistä komennoista löytyy Liitteet-kohdasta (Ks. Liite 6.)

5.1.3 Kubectl arviointi

Kubectl tarjoaa laajan valikoiman komentoja eri toimintoihin, kuten resurssien luomiseen, muokkaamiseen, poistamiseen ja tarkasteluun. Tämä mahdollistaa kattavan hallinnan ja monitoroinnin pienessäkin ympäristössä. Komentorivityökaluna kubectl on suhteellisen yksinkertainen. Sen syntaksi on johdonmukainen eri komentojen välillä, mikä helpottaa sen omaksumista. (Command line tool (kubectl) 2024.) Ohjelma kuitenkin vaatii verrattain paljon aikaa, että sitä oppii tehokkaasti käyttämään.

Läpinäkyvyydeltä Kubectl on hyvä, sillä se tarjoaa reaaliaikaisen tilanäkymän kaikista Kuberneteksen komponenteista. Työkalu myös sopii sekä yksittäisen klusterin, että moniklusterisen ympäristön pyörittämiseen. Jatkuvaa valvontaa tukee esimerkiksi lokien osalta se, että niiden tulostumista pystyy seuraamaan reaaliajassa, joka mahdollistaa nopean reagoinnin virhetilanteessa.

Kubectl:n prosessorin ja muistin käytön seuranta tukee myös hyvin ITIL:in palvelusuunnitteluprosessia.

5.2 K9S

K9s on terminaalipohjainen Kuberneteksen hallintatyökalu, jolla seurata klustereita reaaliajassa. K9s on yksi suosituimmista kubernetes työkaluista, jossa on hyödyllisiä ja käyttäjäystävällisiä ominaisuuksia. K9s on hyvä työkalu myös sellaiselle, jolle kuberneteksen ja klusterin rakenne on vielä hieman epäselvä. Ohjelmasta siis on hyvinkin aloittelijaystävällinen. Ohjelman avulla voi myös selata läpi erilaisia ympäristön resursseja kirjoittamalla kaikista erikseen komentoja, mikä nopeuttaa Kuberneteksen hallintaa. K9s on myös varsin kevyt työkalu, tavallisella käytöllä se käyttää muistia ja prosessoria parin prosentin verran top-ohjelman avulla mitattuna.

5.2.1 K9s asentaminen Ubuntulle

Asentaminen alkaa siitä, että siirrytään K9s:n GitHub-julkaisuihin (https://github.com/derailed/k9s/releases) ja kopioidaan uusimman Linux-version ”k9s_Linux_amd64.tar.gz” -latauslinkki ja ladataan asennustiedosto komennolla:

```
wget <latauslinkki>
```

Tämän jälkeen puretaan arkisto:

```
tar -xvzf k9s_Linux_amd64.tar.gz
```

Siirretään binaaritiedosto alla näkyvään järjestelmän hakemistoon, joka on tarkoitettu käyttäjän asentamille suoritettaville ohjelmille:

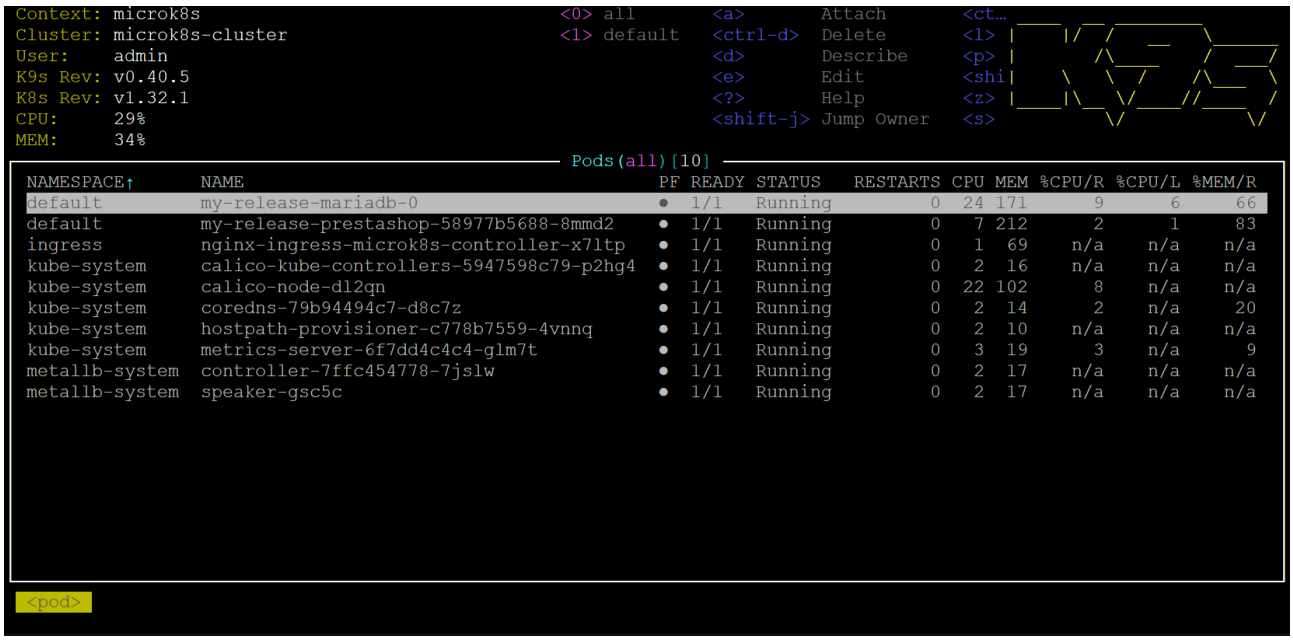
```
sudo mv k9s /usr/local/bin/
```

Jos data ei näy K9s:ssä, pitää lisätä seuraava komento:

```
microk8s config > ~/.kube/config
```

5.2.2 K9s toiminnot

K9s käynnistetään laittamalla Linux terminaaliin yksinkertaisesti “k9s”. Yleisnäkymässä näkyy myös prosessorin sekä muistin käyttö vain, jos metrics-server on asennettu. Asennus tapahtuu Kubectl:n asennusohjeista löytyvällä viimeisellä komennolla. (Ks. Liite 6.)



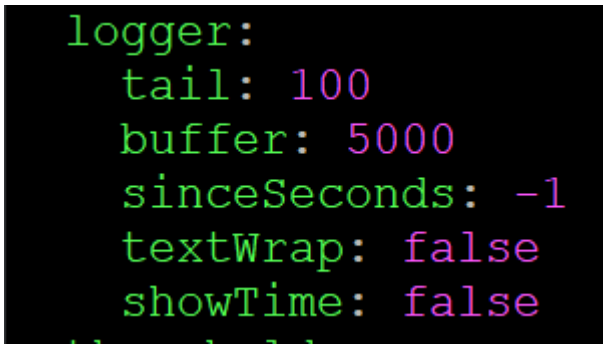
Kuvio 6. Yleisnäkymä K9s:stä Prestashop ja metrics-server asennettuna

K9s:än avulla pystytään tarkistamaan podit, servicet, nodet painamalla näppäimistöstä ctrl + : ja kirjoittamalla esimerkiksi po tai pods, niin aukeaa ikkuna, missä näytetään kaikki klusterissa olevat podit.

K9s päivittää lokit automaattisesti reaaliajassa, mikä tekee siitä loistavan työkalun jatkuvaa valvontaa varten ja työkalu myös tukee vianmääritystä lokien tarkastelun helppoudella selkeästi. Jos haluaa nähdä esimerkiksi Prestashop kontin lokit, menee ensiksi podit-sivulle, jossa avaa Prestashopin podin painamalla Enter, valitsee Prestashop kontin ja painaa enteriä, niin näkee tuon kontin lokit.

Jos haluaa nähdä vanhempia lokeja, kannattaa laittaa autoscroll pois päältä painamalla S. On mahdollista myös nähdä lokien aikaleimat painamalla T sekä laittaa näkymän koko näytölle painamalla F.

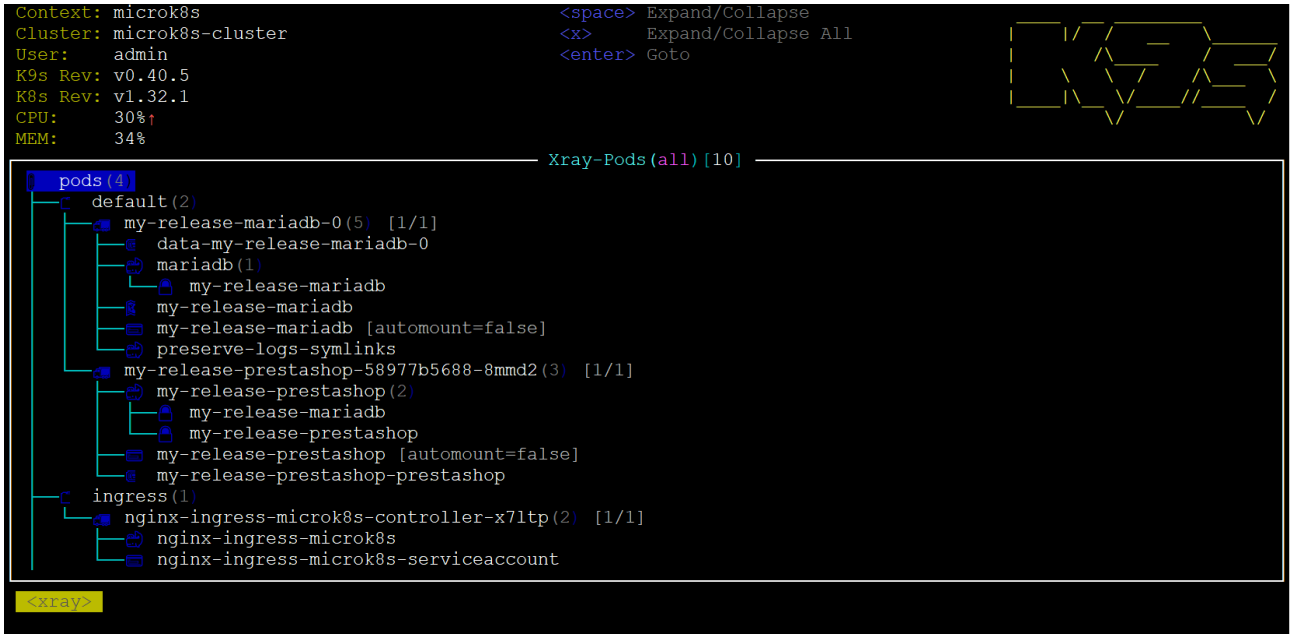
Perusasetuksilla k9s näyttää 100 viimeisintä lokia. Tämän voi muuttaa menemällä k9s konfigurointikansioon ~/.config/k9s/config.yaml ja muuttamalla loggerin alla olevaa tail kohtaan haluamansa numeron. Esimerkiksi laittamalla tähän 500, näytetään käyttäjälle viimeisimmät 500 lokia. Myös muita alla olevassa kuvassa näkyä ominaisuuksia, kuten textWrap (W) ja showTime (T) voi muokata mielensä mukaan (Ks. kuvio 7.)



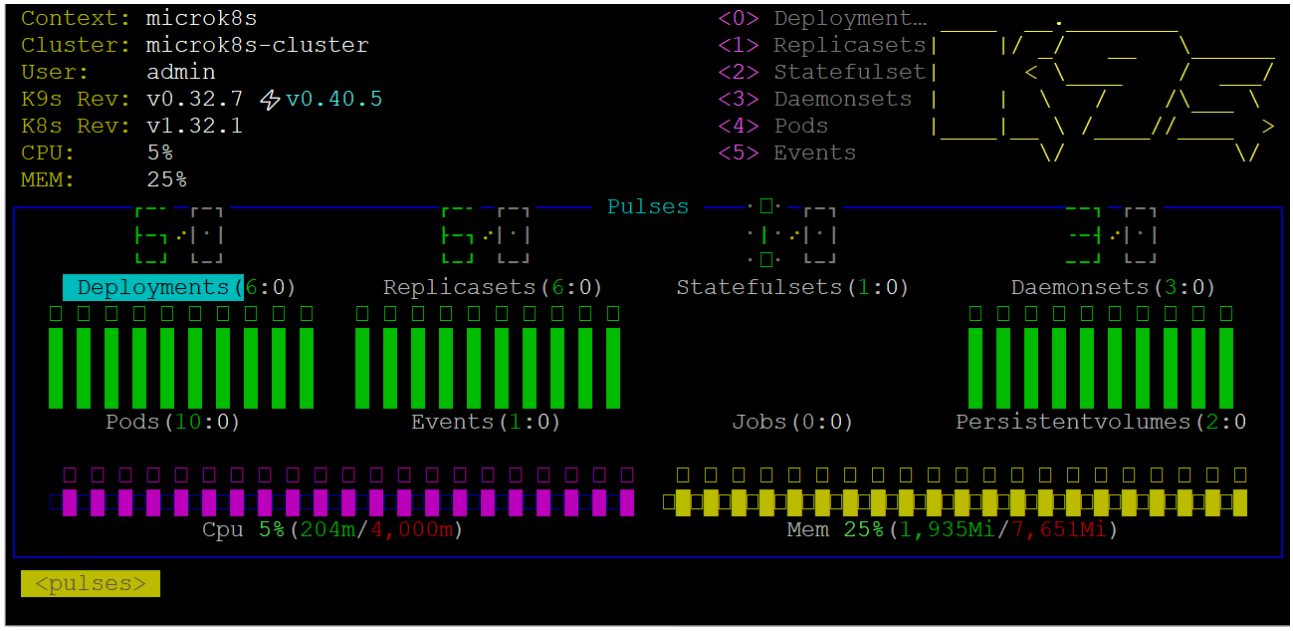
Kuvio 7. Näkymä ~/.config/k9s/config.yaml -tiedostossa

Kaikkien resurssien aliakset saa ohjelmasta näkyviin painamalla ctrl + a. Pulse-näkymän saa laittamalla ”:pulse” ja X-ray-näkymän saa laittamalla ”:xray” + resurssi eli esimerkiksi podien resurssit

saa näkyviin ”:xray po”, jolloin samassa näkymässä näkee kaikki podit ja niiden koko polun. (Katso kuviot 8 ja 9, joissa on esimerkkinäkymät molemmista.)



Kuvio 8. "xray pods" -näkymä K9s:ssä



Kuvio 9. pulse-näkymä K9s:ssä

Podeja voi myös poistaa menemällä pod näkymään, valitsemalla poistettava podi ja painamalla Ctrl + D ja OK.

5.2.3 K9s:n arviointi

K9s on tehokas Kubernetes-hallintatyökalu, joka tukee sekä ITIL-pohjaista palvelutuotantoa että DevOps-käytäntöjä tarjoamalla reaaliaikaisen näkyvyyden ja hallinnan klusteriresursseihin (K9s 2024). Sen avulla voidaan toteuttaa ITIL:in mukaisia operatiivisia prosesseja, kuten tapahtumien hallintaa (Incident Management), kapasiteetin hallintaa (Capacity Management) ja jatkuvaa palvelun parantamista (Continual Service Improvement).

Reaaliaikainen seuranta ja suorituskyvyn monitorointi ovat keskeisiä ominaisuuksia (K9s 2024), jotka tukevat palvelutuotannon jatkuvuutta ja häiriöiden nopeaa hallintaa. K9s tarjoaa tarkan näkymän Kubernetes-klusterin resursseihin, kuten podeihin, kontteihin ja nodeihin (Hoffman 2023), mikä helpottaa kapasiteetin optimointia ja proaktiivista vianhallintaa. ITIL:in näkökulmasta tämä mahdollistaa nopeamman häiriönhallinnan (Incident Management) ja muutostenhallinnan (Change Management), kun voidaan reagoida nopeasti resurssien tarpeisiin ja skaalata palveluita sujuvasti.

DevOps-käytäntöjen mukaisesti K9s tukee jatkuvaa toimitusta (Continuous Delivery) ja infrastruktuurin hallintaa koodina (Infrastructure as Code, IaC). Sen avulla operaatiotiimit voivat automatisoida ja mukauttaa hallintatoimenpiteitä, kuten lokien tarkastelua, uudelleenkäynnistyksiä ja palveluiden skaalautuvuutta. (Hoffman 2023.) "Error Zoom" -ominaisuus nopeuttaa häiriöiden juurisyiden analysointia (K9s 2024), mikä on olennaista DevOpsin mukaisessa jatkuvassa kehityksessä ja palvelun laadun parantamisessa.

Työkalu tukee myös kapasiteetin ja suorituskyvyn hallintaa (Capacity and Performance Management), sillä sen sisäänrakennettu benchmarking-toiminto mahdollistaa HTTP-palveluiden ja podien suorituskyvyn arvioinnin. Lisäksi K9s:n visualisointiominaisuudet auttavat palveluarkkitehtuurin ymmärtämisessä (K9s 2024), mikä tukee sekä ITIL:in mukaista palveluportfoliohallintaa (Service Portfolio Management) että DevOpsin läpinäkyvyyttä ja yhteistyötä.

K9s:n mukautettavuus, kattavat hallintaominaisuudet ja ilmainen saatavuus tekevät siitä erinomaisen työkalun sekä ammattilaisille että vasta-alkajalle. Se auttaa kehittäjiä ja operaatiotiimejä hallitsemaan Kubernetes-ympäristöjä tehokkaasti, tukien sekä ITIL:in palvelutuotantoprosesseja esimerkiksi muistin ja prosessorin käytön seuraamisella. DevOpsia K9s tukee jatkuvaan parantamiseen perustuvalla toimintamallilla automaation kautta.

5.3 KDash

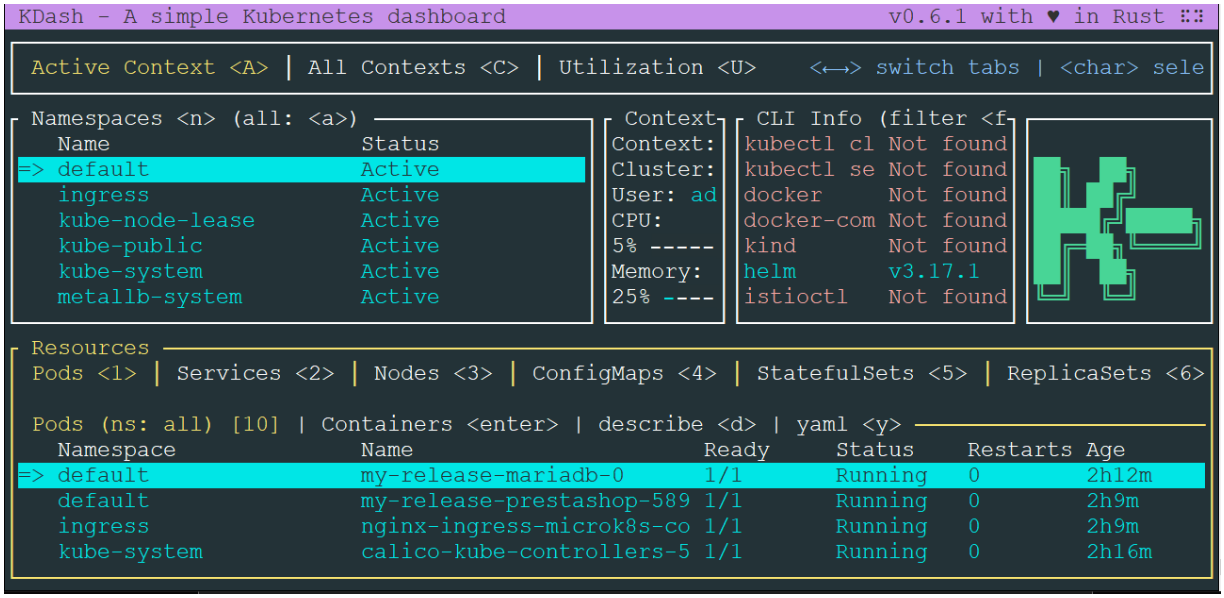
KDash on yksinkertainen mutta tehokas K9s:n tapainen terminaalipohjainen Kuberneteksen monitorointityökalu. Se on avoimen lähdekoodin ohjelma, joka on kehitetty Rust-ohjelmointikielellä. KDash on erittäin kevyt käyttää ja se vie muistia vain noin 0,2 % ja prosessoria 1,5 %, mikä tekee siitä hyvin kevyen ohjelman.

5.3.1 KDash:in asentaminen Ubuntulle

Asentaminen on varsin simppeli toimenpide. Pitää vain ajaa seuraavalla sivulla näkyvä komento, jossa tiedosto ladataan github:ista ja putkitetaan ladattu skripti bash-komentotulkille:

```
curl https://raw.githubusercontent.com/kdash-rs/kdash/main/deployment/getLatest.sh | bash
```

Ohjelma on linux skripti, joka suoritetaan eli käynnistetään laittamalla Linux terminaaliin ”./kdash” tai lisätään järjestelmän ohjelmia sisältävään hakemistoon ”/usr/local/bin”, jonka jälkeen ohjelma käynnistyy kirjoittamalla ”kdash”. Yleinen näkymä Kdashista seuraavalla sivulla (ks. kuvio 10)



Kuvio 10. Yleisnäkymä KDash:ista Prestashop ja metrics-server asennettuna.

5.3.2 KDash toiminnot palvelutuotannon, ITIL:in ja DevOpsin näkökulmasta

KDash sisältää komentorivirajapinnan (CLI) ja yksi KDashin keskeisistä ominaisuuksista on nodejen metriikoiden seuranta, mikä mahdollistaa klusterin suorituskyvyn tarkan valvonnan. Tämä on erityisen tärkeää ITILin mukaisessa kapasiteetin hallinnassa ja suorituskyvyn seurannassa.

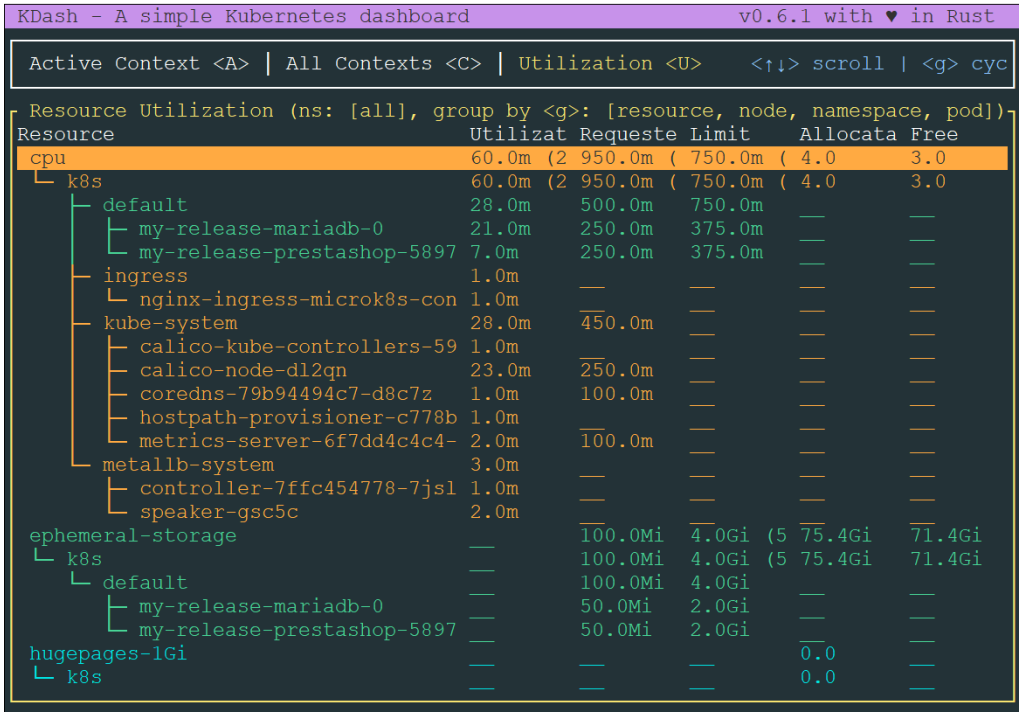
KDash tukee myös mukautettuja resurssimäärityksiä (Custom Resource Definitions, CRD), mikä mahdollistaa Kubernetesin laajentamisen organisaation erityistarpeisiin. Käyttäjät voivat kuvata resursseja ja kopioida tulosteen, mikä helpottaa resurssien analysointia ja dokumentointia. Lisäksi KDash mahdollistaa resurssien YAML-määritysten noutamisen ja kopioinnin, mikä on hyödyllistä konfiguraatioiden hallinnassa ja vianmäärityksessä.

Konttien lokitietojen suoratoiston avulla sovelluskehittäjät ja ylläpitäjät voi seurata ja diagnosoida ongelmia reaaliajassa. Tämä tukee DevOpsin jatkuvaa integrointia ja toimitusta (CI/CD) mahdollistamalla nopean palautteen ja nopeammat korjaukset.

KDash tarjoaa myös kattavat kontekstinhallintaominaisuudet, kuten kontekstitietojen näyttämisen, tarkkailun, nimiavaruuden vaihtamisen ja kontekstin vaihtamisen. Nämä ominaisuudet helpottavat palveluiden hallintaa ja valvontaa, mikä on tärkeää ITIL:in mukaisessa palvelutason hallinnassa.

KDash sisältää resurssien käyttöasteiden seurannan (Utilization) nodeille, nimiavaruuksille ja podeille. Tämä ominaisuus perustuu metrics-serveriin, joka täytyy olla asennettuna järjestelmään. Tämä mahdollistaa resurssien tehokkaan käytön seurannan ja optimoinnin, mikä on olennaista sekä ITILin kapasiteetin hallinnassa, että DevOpsin suorituskyvyn optimoinnissa. Utilization-ikkunan perusnäkymässä näkyy auki kaikki järjestelmän resurssit, nodet, nimiavaruudet ja podit ja painamalla G, saa minimoitua ikkunassa näkyviä elementtejä.

Käyttöliittymän osalta KDash tarjoaa sekä tumman että vaalean teeman, mikä mahdollistaa miellyttävän käyttökokemuksen erilaisissa työympäristöissä. (Katso kuvio 11, jossa näkyy kuva Utilization-ikkunassa tummalla teemalla.)



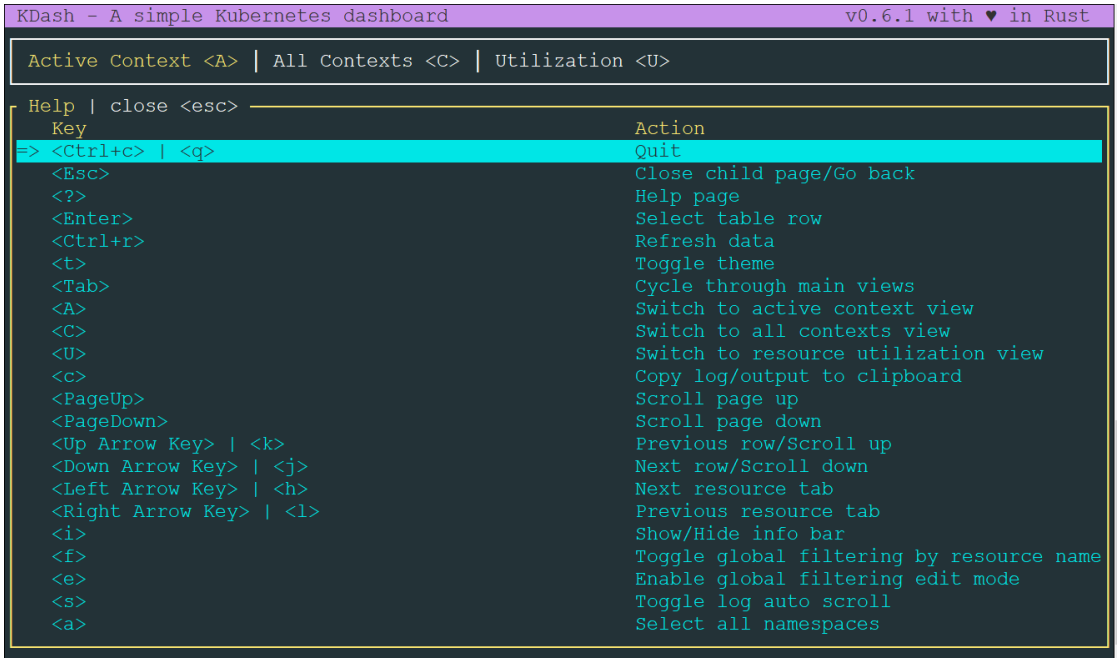
Kuvio 11. KDash:in näkymä Utilization-ikkunasta

Käytettävyyttä parantavat järkevät pikanäppäimet, jotka nopeuttavat yleisimpiä toimintoja. Lisäksi KDash tarjoaa globaalin glob-suodatuksen, joka tarkoittaa jokerimerkkien käyttöä resurssien näkymän rajaamiseen. Suodatus helpottaa suurten resurssimäärien hallintaa ja resurssien nopeaa löytämistä.

Ohjelman alalaidasta voi nopeasti selata seuraavia resursseja: Pods, Services, Nodes, ConfigMaps, StatefulSets, ReplicaSets, Deployments, Jobs, Daemonsets, More ja Dynamic. Määrän takia kaikkia näitä ei voi nähdä, ellei tee Linux järjestelmän ikkunasta tosi pientä tai skaalaa ikkunaa kahden näytön levyiseksi. More ja Dynamic näkymistä voi nähdä halutun lisäresurssin näkymän.

KDash toimii pods näkymässä aivan samalla tavalla, kuin K9s. Kun Pods-näkymässä painaa enter, niin saa näkyviin tuon podin kontit ja sen jälkeen valitseman kontin lokit. Kaikki lokit eivät suoraan tulostu näytölle, vaan ne tulee yksitellen, kun näkymään pääsee. Lisäksi jos lokeja on paljon, eivät ne kaikki mahdu näkymään eikä lokeja pysty selaamaan nuolinäppäimillä, joka tekee ohjelmasta vastaavaan K9s:ään verrattuna selkeästi huonomman käyttää.

Kaikki ominaisuudet ja niiden näppäimet löytyvät Help-ikkunasta, johon pääsee painamalla Shift + ?. Listaa pystyy selaamaan nuolinäppäimillä. (Ks. kuvio 12.)



Kuvio 12. Help-näkymä KDash:issa

5.3.3 KDash vs K9S

K9S:ssä ja KDashissa on omat hyvät puolensa ja käyttötarkoitus ratkaisee sen, mikä on kullekin sopivin vaihtoehto.

K9S hyvät puolet ovat, että sen avulla pystyy monitoroinnin lisäksi hallinnoimaan klusteria monin tavoin, kuten, että ohjelman kautta pystyy poistamaan podeja. Se tukee suodattimia, pikanäppäimiä ja mukautettuja näkymiä. Ohjelma myös tarjoaa kattavan tilannekuvan klusterin terveydestä ja resursseista.

Haittapuolina ovat, että ohjelma on raskas ja monimutkainen yksinkertaiseen seurantaan ja se vaatii hieman totuttelua, koska ohjelmasta löytyy paljon toimintoja.

KDashin hyvät puolet ovat, että sen käyttö on selkeää ja se keskittyy klusterin tilan ja statistiikan nopeaan visualisointiin näyttämällä keskeiset metriikat. Ohjelma on myös kevyt, sillä se kuormittaa paljon vähemmän muistia ja prosessoria, kuin K9s**.**

5.3.4 KDash:in arviointi

KDash on siitä potentiaalisesti hyvä valinta, että KDashin kyky seurata resurssien käyttöä nodeilla, podeilla ja nimiavaruuksilla auttaa ymmärtämään kapasiteettitarpeita, joka parantaa palveluiden kapasiteetin hallintaa. Yllä mainitut ominaisuudet yhdessä tekevät KDashista tehokkaan työkalun Kubernetes-klusterien hallintaan, tarjoten kattavan näkymän klusterin tilaan ja mahdollistaen nopean reagoinnin mahdollisiin ongelmiin.

KDash tarjoaa ilmaisen ja tehokkaan ratkaisun pienten Kubernetes-ympäristöjen hallintaan ja ohjelmassa muistin ja prosessorin käyttö tukee ITIL:in palvelusuunnittelua. Sen kevyt rakenne ja helppokäyttöisyys tekevät siitä ihanteellisen työkalun palvelutuotannon periaatteiden oppimiseen ja soveltamiseen käytännössä.

5.4 Prometheus/Grafana

Prometheus ja Grafana ovat suosittuja avoimen lähdekoodin työkaluja, jotka täydentävät toisiaan monitoroinnin ja datan visualisoinnin alueilla. Prometheus on suunniteltu keräämään ja tallentamaan dataa, kuten sovellusten ja infrastruktuurin suorituskykyä. Sen vahvuuksia ovat tehokas aikasarjatietokanta (time series database), joustava kyselykieli (PromQL) ja kyky tuottaa hälytyksiä perustuen määriteltyihin sääntöihin. Prometheus toimii erityisen hyvin Kubernetes-ympäristöissä, joissa se voi automaattisesti löytää ja kerätä mittareita eri palveluista. (Leppänen 2021.) Prometheus osaa kerätä dataa Kubernetes-ympäristön lisäksi Docker-ympäristöstä.

Grafana puolestaan on visualisointiin keskittynyt työkalu, joka käyttää Prometheuksen keräämää dataa luodakseen selkeitä ja informatiivisia hallintapaneeleja. Grafanan avulla käyttäjät voivat esittää monimutkaisia tietoja helposti ymmärrettävissä visuaalisissa muodoissa, kuten kaavioina ja graafeina. Se tukee myös useiden datalähteiden integrointia, mikä tekee siitä erittäin monipuolisen ratkaisun. Grafana saa datan Prometheukselta tämän HTTP API:n kautta. Tämän API:n avulla Grafana voi lähettää PromQL-kyselyitä ja vastaanottaa aikasarjatietoja vastauksena. (Anand 2024.)

Yhdessä nämä työkalut tarjoavat seuraavat edut: reaaliaikainen näkyvyys järjestelmän tilaan, kyky havaita ongelmat nopeasti, skaalautuvuus suurten ympäristöjen monitorointiin sekä mahdollisuus luoda räätälöityjä hallintapaneeleja eri kohderyhmille. Näiden ominaisuuksien ansiosta Prometheus ja Grafana ovat suosittuja valintoja IT-ympäristöjen monitorointiin ja ylläpitoon. (Anand 2024.)

Grafanan käyttö ei välttämättä ihan ilmaista ole. On hyvin paljon käyttäjän tarpeista kiinni, pitääkö Grafanan käyttöön satsata rahallisesti vai ei. Ilmaisessa versiossa on tarjolla kattavasti dataa käyttäjää varten, mutta isommassa Kubernetes-ympäristössä voi ilmaisversion rajoitukset muodostua ennen pitkään ongelmaksi. (Katso liite 1, jossa on taulukko Grafanan sivulta saaduista tilaustiedoista ja niiden hinnoista.)

5.4.1 Grafanan asentaminen

Tässä testauksessa Grafanaa on käytetty grafana.net -sivuston avulla, jossa asentaminen on tehty kaikista helpoimmaksi. Tämä tapa tukee hyvin vasta-alkajia, joilla vielä puuttuu vankka kokemus

Microk8s:n konfiguroinnista, sillä asennuksen osalta ei juuri mitään muuta tarvitse osata tehdä, kuin seurata annettuja ohjeita. Ohjelma toimii myös ilmaisversiona ja se on pienemmälle Kubernetes installaatiolle sopiva vaihtoehto. Skaalautuessa on otettava huomioon rajalliset resurssit ilmaisversiossa.

Tässä testauksessa Grafanan asennus lähti grafana.netin, Grafana Cloud Stackin Configuration ohjeiden mukaisesti. Ensiksi Grafanan sivulle piti luoda käyttäjä. Tämän jälkeen on mentävä Grafana cloudiin, josta pitää mennä vasemmalta valikosta kohtaan Kubernetes -> Configuration ja seurata Cluster configuration ohjeita. Ensimmäinen kohta ohjeesta sanoo, että Kubectl ja Helm pitää olla asennettuna, mutta tämä kohta voidaan ohittaa, sillä ne tulevat Microk8s:n mukana omana paketoituna versiona.

Seuraavaksi pitää tehdä Backend-asennus, jossa ei tarvitse tehdä muuta kuin painaa ”install”. Kolmannessa kohdassa klusterille ja nimiavaruuteen voi asettaa vapaavalintaisen nimen sekä on valittava, että käyttää Kubernetesta.

Tämän jälkeen valitaan Kubernetes klusterin sekä kontitettujen sovellusten monitorointiin halutut mittarit. Testauksessa on laitettu kaikki vaihtoehdot päälle. Sen jälkeen on luotava ns. käyttöoikeustunnus (access policy token), jotta Grafana Alloy voi lähettää mittareita ja lokeja Grafana Cloudiin. Tokenille pitää antaa nimi ja halutessaan vanhenemispäivä. Voi myös valita, että Token ei vanhene koskaan. Tunnus kannattaa ottaa talteen, mikäli sitä haluaa käyttää vielä uudestaan tulevissa konfiguraatioissa.

Lopuksi viidentenä vaiheena on varmistaa, että on valinnut ”Helm” vaihtoehdon ja kopio Grafanan luoman skriptin, jolla otetaan monitorointi resurssit käyttöön klusterissa, jonka Grafana on luonut annettujen tietojen perusteella. Prestashop on asennettu Microk8s:a käyttäen ja halutaan käyttää sen mukana tulevaa kubectl:ää sekä helmiä. Kirjoittamisvaivan vähentämistä varten on luotava alias, joka lisää aina helm-komennon eteen microk8s:n, jonka jälkeen voi vain ajaa skriptin. Aliaksen voi halutessaan luoda seuraavalla komennolla:

```
alias helm='microk8s helm'
```

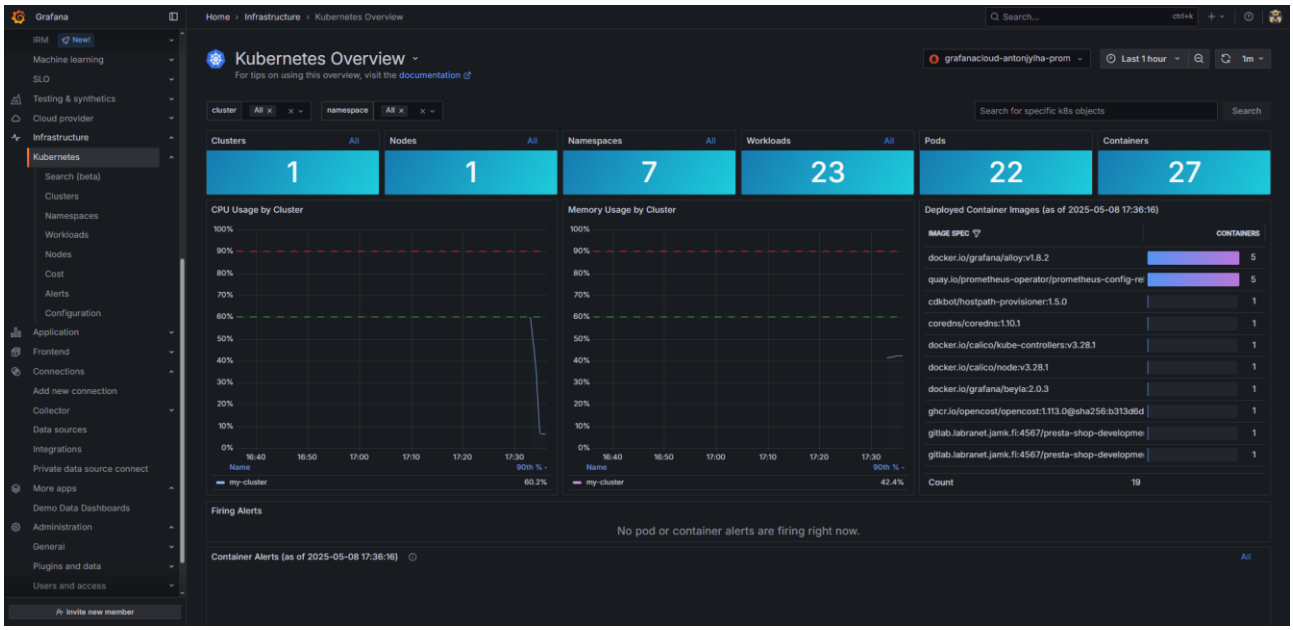
Kun skriptin on ajanut, se saattaa antaa jonkin virhetekstin. Yleensä terminaaliin tulee ohjeet, miten ne korjataan ja korjaus yleensä tapahtuu skriptiä hiukan muokkaamalla mahdollisten ohjeiden mukaan, jonka jälkeen skriptin ajo onnistuu ja monitoroinnin resurssit on saatu käyttöön.

Kun kaikki on valmista, voi alhaalta mennä katsomaan mittareiden tilan ”See cluster status” -kohdasta. Mittareiden käyttöönottoon voi mennä hetki, eli ne ei välttämättä kaikki heti näy Online-tilassa. Jos kaikki on mennyt kuten pitää, niin ainoastaan Windows Exporter pitäisi näkyä Offline-tilassa. Jos mittari toimii, sen kohdalle tulee näkyviin vihreä valintamerkki.

5.4.2 Yleisilme

Kun käsittelyssä on Kuberneteksen mittareiden ja muun datan tarkastelu, niin keskitytään ainoastaan siihen, mitä tuon Infrastructure -> Kubernetes alta löytyy.

Aloitetaan tarkastelu Grafanassa Kuberneteksen yleisnäkymästä. Yleisnäkymä näyttää äkkiseltään tosi sekavalta. (Ks. kuvio 13.) Ensimmäinen kuitenkin osuu silmään siniset neliöt, joissa lukee, montako Klusteria, Nodea, nimiavaruutta, työmäärää (workloadia), podia ja konttia Grafanaan on ajettu. Tämän alta löytyy testauksessa ajetun klusterin prosessorin ja keskusmuistin käyttö prosentteina. Vakiona tieto käytöstä piirtyy kaavioon kymmenen minuutin välein.

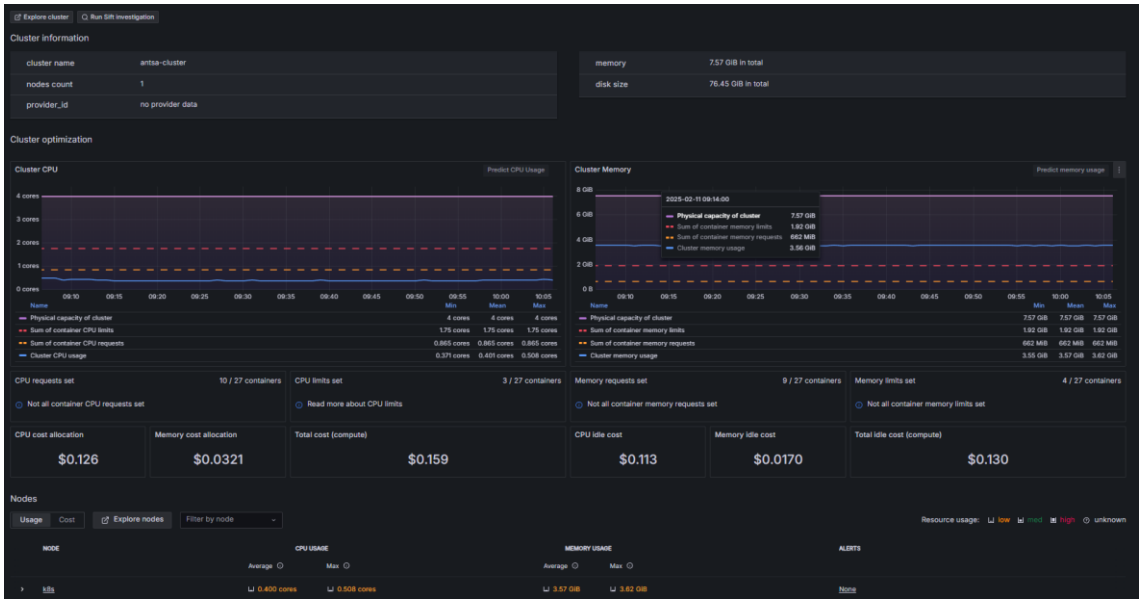


Kuvio 13. Yleisnäkymä Grafanasta

Tämän alapuolelta löytyy tieto Firing-hälytyksistä. Firing-hälytykset Grafanassa tarkoittavat hälytyksiä, jotka ovat ylittäneet ennalta määritetyn kynnyksen ja ovat aktiivisesti käynnissä eli lähettävät ilmoituksia. Tämän alta sitten löytyvät vielä Konttien sekä podien hälytyksille omat kohdat. Hälytykset ovat siitä hyviä, että ne mahdollisesti poistavat tarpeen jatkuvalle klusterin monitoroinnille, koska Grafana osaa itse poikkeavassa tilanteessa luoda hälytyksiä käyttäjälle.

Tästä kun siirtyy ”Search” -kohtaan, niin avautuu kattava yleisnäkymä kaikista Grafanaan tuoduista klustereista, nodeista, nimiavaruuksita, työkuormista, podeista sekä konteista tässä järjestyksessä. Näkymän määrä on rajoitettu sataan ja testauksessa käytetyn pienen ympäristöni kanssa siitä ei muodostunut ongelmaa

”Clusters” -näkymässä näkyy kaikki nykyiset klusterit, klustereiden nodejen määrä, niiden prosessorin sekä keskusmuistin keskimääräinen käyttö, mutta myös prosessorin ja keskusmuistin maksimi käyttö prosentteinä, että tavuina. Numeroiden väri myös ilmaisee, onko käyttö vähäistä, keskitasoista vai suurta. (Katso alla oleva kuvio 14, jossa näkyy yleiskatsaus klusterista saadusta datasta.)



Kuvio 14. "Clusters" -näkymässä klusterin data

Tämän jälkeen tulee ”Network” -näkymä, jossa on dataa verkon kaistanleveyksistä ja saturaatioista. (Ks. kuvio 15.)

Kuvio 15. Klusterin "Network" -näkymä

Tämän jälkeen ”Storage” -näkymässä on tietoa Persistent Volume Claimista (PVC) ja ”Energy” -näkymässä energiankäyttö wattitunteina. Viimeinen näkymä on myös varsin hyödyllinen, sillä siellä näkee klusterin lokitiedot ja tapahtumat. Seuraavaksi kun siirtyy Clusters näkymän jälkeen Namespaces, Workloads tai Nodes-näkymään, näkee nimen mukaiset kaikki samat tiedot, kuin mitä Clusters-näkymästä näkyi ylempänä.

”Cost” -näkymässä nimensä mukaisesti näkee paljonko Kubernetes-ympäristö kustantaa Grafana Cloudille. (Ks. Liite 2.) Tämä tietenkin vaihtelee Kubernetes klusterin koosta ja käyttöasteesta riippuen. Sitten lopuksi on Alerts näkymä, johon tulee tietoa hälytyksistä.

5.4.3 Prometheus/Grafanan arviointi

Grafana/Prometheus on osoittautunut ehdottomasti laajimmaksi ominaisuuksiltaan, kun vertaa Kubectl:ään, K9s:n ja KDashiin. Grafana on erinomainen työkalu käyttäjälle, joka työskentelee Kubernetes-ympäristössä ja haluaa syventää ymmärrystään palvelutuotannon, ITIL:in ja DevOpsin näkökulmista.

Grafana tarjoaa käyttäjälle mahdollisuuden visualisoida Kubernetes-klusterin tuottamia metriikoita helposti ymmärrettävässä muodossa (Rabinovich 2024). Tämä auttaa käyttäjää kehittämään käytännön taitoja monitoroinnissa ja vianmäärityksessä, jotka ovat keskeisiä palvelutuotannon ja DevOps-käytäntöjen kannalta

Palvelutuotannon näkökulmasta Grafana mahdollistaa reaaliaikaisen näkyvyyden Kubernetes-ympäristön suorituskykyyn ja terveyteen (Rabinovich 2024). Käyttäjä voi luoda mukautettuja kojelautoja, jotka näyttävät kriittisiä metriikoita kuten CPU:n käyttö, muistin kulutus ja verkon suorituskyky. Tämä auttaa ymmärtämään palveluiden toimintaa ja tunnistamaan mahdollisia pullonkauloja. Grafanan jatkuva seuranta lokien avulla ja hälytys asetusten asettamisen mahdollisuus tukevat sen käyttäjää hyödyntämään ITIL-prosesseja palvelutuotannossa. Grafanassa on myös mahdollista laittaa julkiseen jakoon hallintapaneelin, vaikka kohde ei olisikaan osa organisaatiota, joka lisää läpinäkyvyyttä ja tiedonjakoa.

ITIL:inkin näkökulmasta käyttäjä voisi muun muassa seurata ja visualisoida poikkeamia normaalitilasta sekä resurssien käytön seuranta mahdollistaa tehokkaan kapasiteettisuunnittelun. Sitten taas DevOpsin kannalta Grafana edistää jatkuvaa monitorointia ja nopeaa palautetta sekä sen avulla on helppo seurata sovellusten suorituskykyä ja luotettavuutta tehokkaasti. (Kuutti 2024.) Cost-näkymä on hyödyllinen ITIL:n palvelustrategian näkökulmasta, kun pitää arvioida IT-palveluiden kustannuksia ja budjetointia.

Grafanaa, kun monitoroidaan selaimen kautta, on hyvä varmistaa, että sivustolle pääsee vain sinne pääsyn vaativat tahot, eikä kukaan ulkopuolinen, sillä näin laajan datan monitorointi tunkeilijan toimesta on merkittävä tietoturvariski käyttäjälle tai yritykselle.

5.5 Kubernetes Dashboard

Dashboard on verkkopohjainen Kubernetesin käyttöliittymä. Dashboardia voi käyttää kontitettujen sovellusten käyttöönottoon Kubernetes-klusteriin, kontitettujen sovellusten vianmääritykseen sekä klusterin resurssien hallintaan. Dashboardin avulla voi saada yleiskatsauksen klusterissa käynnissä olevista sovelluksista sekä luoda tai muokata yksittäisiä Kubernetes-resursseja (Deployments, Jobs, DaemonSets jne.). Dashboardissa voi esimerkiksi skaalata Deploymentin, aloittaa rolling update, käynnistää podi uudelleen tai ottaa käyttöön uusia sovelluksia deploy wizardin avulla.

Dashboard tarjoaa myös tietoa Kubernetes-resurssien tilasta klusterissasi sekä mahdollisista virheistä, joita on voinut tapahtua. (Deploy and Access the Kubernetes Dashboard 2024.)

5.5.1 Asentaminen

Asennus tehtiin Kubernetes Dashboard (Deploy and Access the Kubernetes Dashboard 2024) sivun ohjeiden avulla. Asennus oli hieman pidempi prosessi, kuin muiden ohjelmien kohdalla, joten sen löytää Liitteet-kohdasta. (Ks. Liite 7.)

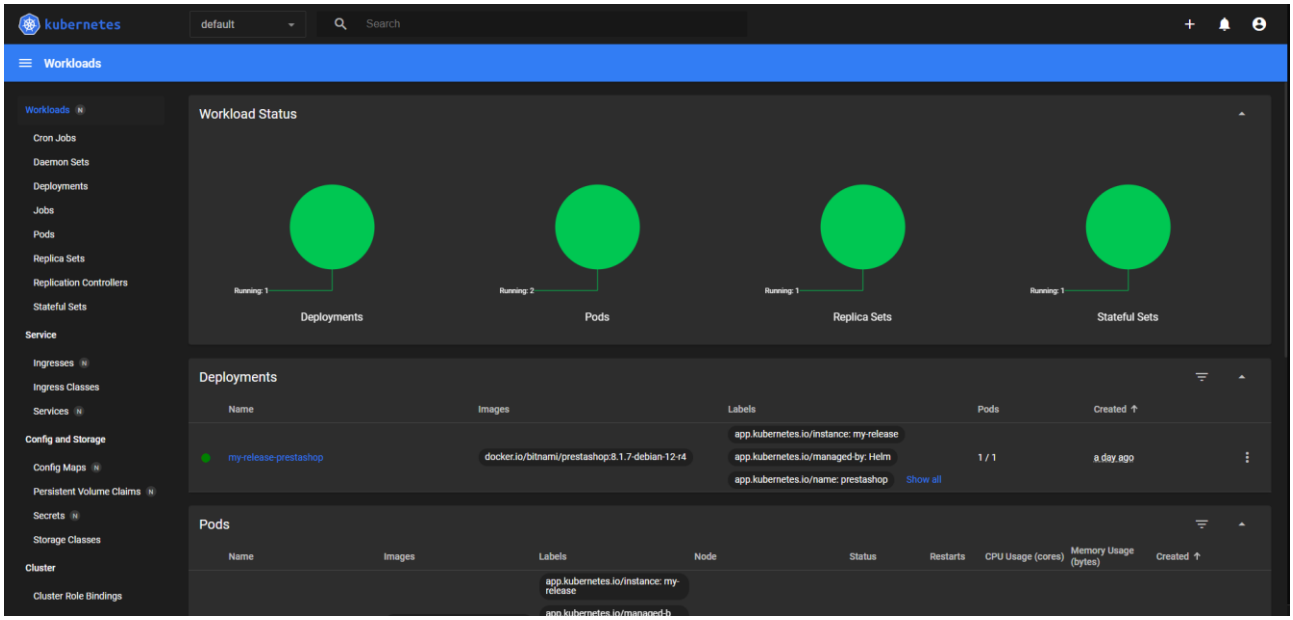
Ohjeiden lopusta saadun tokenin saamisen jälkeen mennään seuraavaksi selaimeen, laitetaan Linux koneen IP ja Nodeportin numero, niin dashboard kirjautumissivustolle pääsee käsiksi, eli https://<koneenIP>:<NodePort>. cPoudasta piti myös muistaa avata kyseinen portti, jota tässä tapauksessa käytetään.

Token mikä tulostuu edellisellä komennolla, laitetaan sivuston ”Bearer Token”-kohtaan ja kirjaudutaan, jonka jälkeen sivuston pitäisi toimia.

Bearer Token on OAuth 2.0:n autentikointikehys, joka myöntää pääsyn suojattuihin resursseihin. Bearer Tokenit myöntää valtuutuspalvelin, ja ne sisältyvät HTTP-pyyntöihin, jotka todentavat pyynnön tekijän. Bearer Tokeneita käytetään web-sovelluksissa ja API:ssa. (Mottammal 2024.)

5.5.2 Yleisnäkymä

Kubernetes Dashboardin “Workloads” -sivulla näkyy ympäristön kaikki Deploymentit, Podit, Replica Setit ja Stateful Setit. (Ks. kuvio 16.) Sivulta näkee näiden nimet, imaget ja tilan. Podien kohdalla näkee uudelleen käynnistykset, prosessorin ja muistin käytön sekä niiden luontiajankohta.



Kuvio 16. Yleisnäkymä Kubernetes Dashboardista

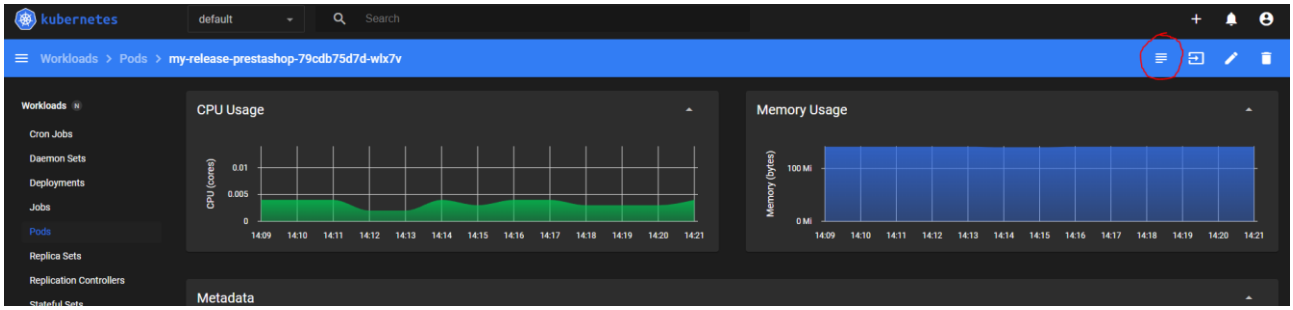
Sivulta näkee dataa seuraavista Workload kategorian komponenteista: Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets. (Katso liite 3 pods-näkymästä.)

“Service”-Kategoriasta taas löytyy Ingressit, Ingress Classes ja Servicet. “Config and Storage”-kategorian alta löytyy Config Maps, Persistent Volume Claims, Secrets ja Storage Classes.

“Cluster”-kategorian alla on Cluster Role Bindings, Cluster Roles, Events, Namespaces, Network Policies, Nodes, Persistent Volumes, Role Bindings, Roles ja Service Accounts.

”Custom Resource Definitions” kautta pystyy myös tarkistamaan lisää tietoja klusterista kuten sertifikaatit ja niiden YAML-tiedoston.

Sivuilla eri resurssit näyttävät kattavasti tietoa nimistä, luontiajankohdista, mahdolliset IP:t, Type, Imaget ja muuta lisätietoa riippuen, mitä resurssia tarkastellaan. Esimerkiksi podin lokeja pystyy tarkastelemaan, kun valitsee ”Pods”-kohdasta tarkasteltavan podin ja oikeasta yläkulmasta valitsee kolmea viivaa esittävän iconin. (Ks. kuvio 17.)



Kuvio 17. Podin toiminnot

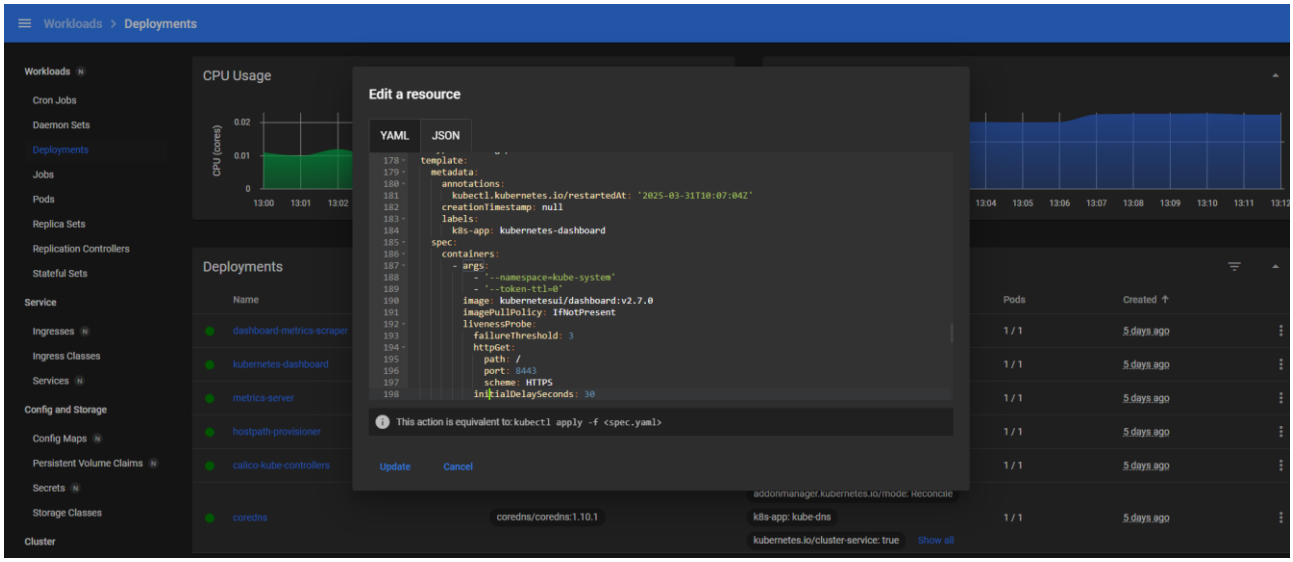
Lokinäkymä painikkeen viereisestä vaihtoehdosta, kun painaa, aukeaa shell ikkuna kyseisen podin sisälle, jossa voi ajaa erinäisiä komentoja halutessaan. Tämä toiminto löytyy myös kubectl:ssä ja siellä se ajetaan seuraavalla komennolla:

```
kubectl exec -it <pod nimi> -- /bin/bash
```

Tämän jälkeen tulee myös YAML-tiedoston muokkaus ja resurssin poisto.

Resurssin muokkaaminen Kubernetes Dashboardissa

Kaikkien resurssien konfiguraatiota pystyy myös muokkaamaan Kubernetes Dashboardin kautta. Testissä haluttiin ottaa automaattinen uloskirjautuminen pois, jonka pystyi toteuttamaan perinteisellä kubectl:n avulla tai sitten Dashboardin kautta valitsemalla namespace ylivalikosta ”kube-system”, etsimällä Deployment kohdasta ”kubernetes dashboard” ja kolmesta pisteestä kun painaa, voi valita ”Edit” ja muokata tiedostoon ”--token-ttl=0”. Lopuksi pitää painaa ”Update”, jotta muokkaukset tallentuvat tiedostoon. (Ks. kuvio 17.)



Kuvio 18. YAML-tiedoston muokkaaminen Kubernetes Dashboardissa

Yllä oleva kuva myös näyttää, että mikä kubectl:n komento tekisi saman toimenpiteen, joten ohjelma on myös sikäli hyvä, että se samalla opettaa käyttämään kubectl:ää, joka voi joissain konfiguraatiotilanteissa olla nopeampi vaihtoehto, varsinkin jos tykkää ja on tottunut käyttämään terminaalia.

Virhetilanteet

Virhetilanteita voi selvittää sillä, että jos esimerkiksi Deploymentin kanssa tulee ongelmia ja se ei käynnisty, voi ongelman virhetekstin löytää Deploymentin ”Events”-kohdasta. Jos vika onkin deploymentin podista kiinni, voi podin sivulta nähdä tarkempia tietoja ongelmasta kattavasti. Keltainen pallo podin vieressä indikoi varoitusta tai odottamista ja punainen virhettä. (Ks. liite 4 ja 5.)

5.5.3 Kubernetes Dashboard arviointi

Kubernetes Dashboard on hyvin kattava tapa monitoroida sekä muokata eri resursseja ja niiden YAML-tiedostoja. Käyttöliittymä on selkeä ja ohjelma auttaa käyttäjää ymmärtämään Kubernetes-ympäristöä hieman paremmin tekemällä rakenteesta selkeän.

Tällainen klusterin hallintaympäristö, joka on mielekästä ohjata toimimaan selaimen kautta, on myös sellaisenaan tietoturvariski ja on varmistettava, että pääsy Dashboardiin käsiksi tapahtuu

vain sallituilta tahoilta, sillä kuten on huomattu, ohjelman avulla pystyy muokkaamaan Kubernetes-ympäristöä hyvin laajasti ja näin mahdollista aiheuttaa harmia tai kerätä dataa sivulta. Sivusto pääsy on kuitenkin hyvin turvattu Service käyttäjien bearer avainten avulla. ITIL:in tietoturvaprosessien hallintaa tukee Kubernetes Dashboardissa olevien lokien reaaliaikainen seuranta, miinusta tulee kuitenkin siitä, että Dashboard ei tarjoa itsessään hälytysominaisuutta, kuten Grafana tekee.

Muistin ja prosessorin käytön seuranta edistää sitä, että klusterin ylläpitäjä pystyy seuraamaan tarkasti palvelun kuormittavuutta ja sen skaalautumisen kuormittavia vaikutuksia ja tämä taas tukee hyvin ITIL:in palvelusuunnitteluprosessia.

6 Työkalujen soveltuminen palvelutuotantoon

6.1 Kubectl

Tukee ITIL:n periaatteita, kuten palvelunhallinnan standardointia ja tehokkuutta, tarjoamalla suoran pääsyn klusterin hallintaan sekä mahdollistaa tarkat muutostenhallintaprosessit YAML-konfiguraatioiden avulla. Ohjelma myös mahdollistaa Infrastruktuurin koodina (IaC)-periaatteiden käytön. Toisaalta haittana voi pitää sitä, että ohjelma on vain komentorivipohjainen, joka tuo hitautta tiimeille, joilla ei ole syvällistä Kubernetes osaamista sekä visuaalisuuden puute voi aiheuttaa kattavaa tilannekuvan hahmottamista.

6.2 K9s

ITIL:in näkökulmasta ohjelma helpottaa operatiivista hallintaa tarjoamalla reaaliaikaisen näkymän klusterin tilaan ja vähentää virheiden riskiä yksinkertaistamalla resurssien hallintaa. DevOpsin kannalta tukee ketterää toimintaa nopeilla navigointiominaisuuksilla, mutta sitten ei taas tue pitkän aikavälin monitorointia tai analytiikkaa, ja ohjelmassa on rajallinen skaalautuvuus suurissa ympäristöissä.

6.3 KDash

ITIL:in näkökulmasta ohjelma parantaa käyttäjäkokemusta selkeillä visuaalisilla näkymillä palveluiden tilasta sekä tukee palveluiden suunnittelua ja käyttöönottovaihetta yksinkertaisella käyttöliit-

tymällä. DevOpsin näkökulmasta ohjelma sopii tiimeille, jotka haluavat välttää komentorivin käyttöä kehityksen aikana sekä auttaa nopeuttamaan päätöksentekoa visuaalisten tietojen avulla ja on myös hyvin kevyt ohjelma käyttää, jos esimerkiksi ympäristön kapasiteetti on hyvin rajallinen. Haittapuolina taas voisi pitää hälytysominaisuuksien puutetta eikä ohjelma ole myöskään yhtä laajasti tuettu, kuin muut.

6.4 Prometheus/Grafana

ITIL:in näkökulmasta ohjelma mahdollistaa palveluiden suorituskyvyn seurannan ja jatkuvan parantamisen (continual improvement) sekä tukee riskienhallintaa hälytyksillä ja pitkän aikavälin datan analysoinnilla. DevOpsin näkökulmasta taas ohjelma tukee jatkuvaa integrointia ja toimitusta (CI/CD) tarjoamalla selkeät mittarit sovellusten tilasta. Haittana taas on, että joissain tapauksissa asennus ja konfigurointi voivat olla monimutkaisia, mikä voi hidastaa käyttöönottoa.

6.5 Kubernetes Dashboard

ITIL:in näkökulmasta ohjelma tarjoaa yleiskatsauksen klusterin tilaan, mikä tukee operatiivista tehokkuutta ja asiakastyytyväisyyttä sekä helpottaa palveluiden hallintaa visuaalisuuden avulla. DevOpsin näkökulmasta ohjelman visuaalinen hallinnointi helpottaa manuaalista työtä. Ohjelmassa on kuitenkin hieman rajoitetummat toiminnallisuudet, kuin Prometheus/Grafanassa. Myös turvallisuus voi olla ongelma tuotantoympäristöissä ilman asianmukaista konfigurointia sen saralla.

6.6 Yhteenveto taulukoina

Sivuilla 49–51 vielä yhteenveto ohjelmien ominaisuuksista ja siitä, kuinka ne tukevat ITIL v4 käytäntöjä sekä DevOpsia. Vaikka käytäntöjä on yhteensä 34, taulukkoon on valikoitunut näistä ainoastaan kahdeksan, sillä nämä käytännöt ITIL:istä ovat käytännönläheisesti relevantteja Kubernetes-ympäristöjen hallinnan kannalta ja juuri tässä työssä käsiteltyjen ohjelmien osalta. Visualisoinnin helpottamiseksi tukevat kohdat on maalattu vihreällä, osittain tukevat keltaisella ja ei tukevat punaisella.

Taulukko 1. Yhteenveto ohjelmien ominaisuuksista ITILin ja DevOpsin kannalta

ITIL-käytäntö	Kubectl	K9s	Kdash	Prometheus/Grafana	Kubernetes Dashboard
Incident Management	**Osittain:** Komentorivityökalu mahdollistaa nopean reagoinnin ja resurssien hallinnan, mutta ei tarjoa automaattisia hälytyksiä.	**Osittain:** Tarjoaa reaaliaikaisen näkymän klusterin tilaan ja lokeihin, jolloin vikatilanteet voidaan havaita nopeammin.	**Osittain:** Visuaalinen käyttöliittymä antaa nopean kuvan klusterin tilasta, mikä voi auttaa tunnistamaan häiriötilanteita.	**Tukee:** Reaaliaikaiset mittarit, hälytykset ja lokit mahdollistavat proaktiivisen vikatilanteiden seurannan ja hallinnan.	**Osittain:** Näyttää klusterin tilatiedot, mutta ei sisällä automaattista hälytysjärjestelmää.
Problem Management	**Osittain:** Tarjoaa lähinnä manuaalista diagnostiikkaa, eli käyttäjän on itse tarkastettava lokit ja eventit.	**Osittain:** Lokien ja tapahtumien seuranta auttaa havaitsemaan toistuvia ongelmia, mutta analyysi vaatii lisätyökaluja.	**Osittain:** Selkeä näkymä klusterin trendeihin voi tukea ongelmien juurisyiden etsimistä, kuitenkaan ei tarjoa kokonaisanalyysejä.	**Tukee:** Historiallinen data ja kehittyneet analytiikkatoiminnot auttavat tunnistamaan ja seuraamaan toistuvia ongelmia.	**Osittain:** Perustason tilanäkyvyys mahdollistaa ongelmien varhaisen huomion, mutta edistyneet analyysityökalut puuttuvat.
Change Management	**Tukee:** Käytetään suoraan konfiguraatioiden päivittämiseen, deployeihin ja muutosten toteutukseen.	**Osittain:** Näyttää muutosten vaikutukset reaaliajassa, mutta ei hallinnoi muutosten hyväksymisprosessia.	**Osittain:** Samat kuin K9s:ssä	**Ei tue:** Keskittyy ensisijaisesti monitorointiin eikä hallinnoi muutosten hyväksymistä tai dokumentointia.	**Tukee:** Mahdollistaa resurssien päivityksen visuaalisesti sekä tarjoaa halutessaan myös Kubectl:n komennon samalle toiminnolle.

Configuration Management	**Tukee:** Mahdollistaa suoran hallinnan konfiguraatioitemeistä (esim. deployments, configmaps) komentoriviltä.	**Tukee:** Näyttää konfiguraatiot reaaliaikaisesti, mikä auttaa tilanvalvonnassa.	**Tukee:** Visuaalinen näkymä klusterin konfiguraatioista antaa hyvän yleiskuvan asetuksista ja muutoksista.	**Ei tue:** Vaikka se ei hallinnoi konfiguraatioita, se voi auttaa tunnistamaan poikkeavuuksia mittaamalla klusterin tilaa.	**Tukee:** Näyttää resurssien konfiguraatiot selkeästi, vaikka se on rajoitetumpi muokkausmahdollisuuksien osalta.
Service Level Management	**Ei tue:** Ei tarjoa SLA-metriikoita tai palvelutason seurantaa ilman lisäintegraatioita.	**Osittain:** Reaaliaikaiset tilaraportit antavat käsityksen palveluiden tilasta, mutta ei erikseen mittaa SLA-tasoja.	**Osittain:** Näyttää perusnäkymän klusterin tilasta, mikä voi tukea SLA-seurantaa, mutta ilman syvällisiä metriikoita.	**Tukee:** Voidaan rakentaa yksityiskohtaiset SLA-dashboardit ja asettaa hälytyksiä palvelutasojen alittuessa, mahdollistaen aktiivisen SLA-seurannan.	**Osittain:** Perustason tilatiedot löytyvät, mutta ei tukea yksityiskohtaiseen SLA-mittaukseen tai -analyysiin.
Capacity Management	**Osittain:** Mahdollistaa resurssien manuaalisen tarkastuksen, mutta ei tarjoa analysoivia raportteja kapasiteetista.	**Osittain:** Näyttää reaaliaikaiset resurssikäyttötilastot, jotka voivat tukea kapasiteetin suunnittelua, mutta ei analyysiä syvällisesti.	**Osittain:** Esittää klusterin resurssien tilastoja, mutta edistyneempiä kapasiteettianalyysejä ei sisällä.	**Tukee:** Mittaa kuormitusta ja resursseja laajasti, mikä mahdollistaa tarkat kapasiteetti- ja kuormitusanalyysit sekä tulevaisuuden suunnittelun.	**Osittain:** Näyttää resurssien peruskäyttötilat, mutta edistyneet kapasiteettiraportit vaativat lisäintegraatioita.
Availability Management	**Ei tue:** Ei tarjoa suoria toimintoja palvelun saatavuuden seurantaan.	**Osittain:** Näyttää podien ja palveluiden tilan, mikä auttaa määrittämään käytettävyyttä reaaliajassa.	**Osittain:** Antaa yleiskuvan klusterin tilasta, josta on mahdollista arvioida saatavuutta, mutta	**Tukee:** Monitoroi palveluiden uptimea ja käytettävyyttä, mahdollistaen jatkuvan saatavuuden seurannan ja raportoinnin.	**Osittain:** Näyttää perustason tilatiedot, joista voidaan päätellä saatavuus, mutta yksityiskohtaiset analyysit puuttuvat.

			ilman syvällistä analyysiä.		
DevOps-yhteensopivuus	**Osittain:** Vahva operointityökalu, mutta ei automaatiota tai palautesykliä itsessään	**Osittain:** Erinomainen nopeaan CLI-navigointiin, mutta ei automatisoi prosesseja	**Osittain:** Kevyt visuaalinen näkymä, mutta ei tue työvirtoja tai automaatiota.	**Osittain:** Hyödyllinen GUI, mutta manuaalinen käyttöraja; ei DevOps-pipeline-tuki.	**Tukee:** Tukee jatkuvaa mittarointia, palautesykliä ja integraatiota alertingiin.

Taulukon perusteella vielä pisteytin ohjelmat ja toteutin sen niin, että jos ohjelma ei tue käytäntöä saisi tämä 0 pistettä, osittaisesta tuennasta 0,5 ja täydestä tuesta 1, tässä vertailussa tasapelin saisi Prometheus/Grafana sekä Kubernetes Dashboard 5,5 pisteellä, muut pisteet ovat Kubectl:lle 4, K9s:lle 4,5 ja KDash 4,5.

nämä ovat siis arviot yksinomaan ITIL:in ja DevOpsin näkökulmasta. Tutkimuksessa tutkittiin myös ohjelmia vianmäärityksen, käytettävyyden, läpinäkyvyyden, sekä kustannusten kannalta, josta on koostettu taulukko sivuille 51 ja 52.

Taulukko 2. Yhteenveto ohjelmien ominaisuuksista läpinäkyvyyden, käytettävyyden, vianmäärityksen ja kustannusten kannalta

Ohjelma	**Läpinäkyvyys**	**Käytettävyys**	**Vianmääritys**	**Kustannukset**
kubectl	Täysi näkyvyys kaikkiin resursseihin; yksityiskohtainen mutta vaatii osaamista	Vaatii komentorivin hallintaa ja komentojen muistamista; ei graafista käyttöliittymää	Täydet vianmääritysmahdollisuudet: lokit, tilat, tapahtumat, suorat komennot	Ilmainen, avoin lähdekoodi; palvelimen osalta ylläpitokustannukset vähäisiä
K9s	Reaaliaikainen yleisnäkymä klusteriin terminaalissa; hyvä näkyvyys mutta	Helpompi kuin kubectl; komentopohjainen mutta intuitiivinen TUI	Hyvä reaaliaikainen vianmääritys: tilat, logit,	Ilmainen, avoin lähdekoodi; kevyt ylläpitää

	suppeampi kuin kubectl		uudelleenkäynnistys käyttöliittymästä	
KDash	Visuaalinen perustason näkymä podien ja resurssien tilaan	Yksinkertainen ja kevyt visuaalinen käyttöliittymä; rajoitetut ominaisuudet	Perustason vianmääritys: tilat ja statukset, ei syviä lokitietoja	Ilmainen, avoin lähdekoodi; hyvin kevyt
Prometheus/Grafana	Erinomainen suorituskyvyn ja metriikoiden näkyvyys; pitkän aikavälin trendien seuranta ja monia eri visuaalisia vaihtoehtoja	Grafana helppokäyttöinen ja visuaalinen; Prometheus vaatii enemmän konfigurointiosaamista	Vianmääritys metriikoiden perusteella: kuormitus, lokit, data tietokannasta	Ilmaiset ja avoimen lähdekoodin; skaalautuessa resurssi- ja ylläpitokustannukset kasvavat
Kubernetes Dashboard	Graafinen näkymä perusresursseihin; rajallinen näkyvyys, ei syvää teknistä tietoa	Helppokäyttöinen graafinen käyttöliittymä; aloittelijaystävällinen	Kevyt vianmääritys: tilat, logit; ei syvällisiä tapahtumia tai säätöjä	Ilmainen, avoin lähdekoodi; palvelimen osalta pienet ylläpitokustannukset

7 Työkaluille sopivat ympäristöt

Yllä mainituille viidelle työkalulle löytyy varmasti kaikille sopiva ympäristö riippuen sen koosta, käyttötarkoituksesta, käytetyistä prosesseista ja standardeista. Tässä työssä tarkasteltiin ohjelmia palvelutuotannon, ITIL:in ja DevOpsin näkökulmasta ja että millaiseen ympäristöön mikäkin ohjelma olisi hyvä. Hyvin pieneen ja vaatimattomaan ympäristöön, johon tarvitaan kevyt ja yksinkertainen monitorointi ja hallintatyökalu, sekä käyttäjälle, jolle Kuberneteksen rakenne on vielä tuntematon, olisi K9s sopiva valinta.

Hieman isompaan, keskikokoisiin IT-palvelutuontantoympäristöihin sopiva vaihtoehto olisi Kubernetes Dashboard, sillä se tarjoaa helpon visuaalisen ratkaisun klusterin seurantaan ja sen hallinnointiin laajasti. Sen asentaminen ei ole vaikeaa ja se helpottaa sellaisen ylläpitäjän elämää, jolle kubectl:n syntaksit eivät ole vielä täysin hallussa.

Suuret yritykset usein hyödyntävät tiedon keräämiseen Prometheusta ja Grafanaa. Monimuotoisuuden vuoksi tämä olisi näistä vaihtoehdoista paras ratkaisu suurten yritysten tarpeisiin. Grafana kun ei tarjoa suoraan resurssien muokkaukseen työkaluja, vaan toimii lähinnä monitorointia varten, niin muokkaukseen olisi hyvä valita Kubernetes Dashboard yksinkertaisuuden ja selkeyden näkökulmasta

Toisaalta kokenut Kubernetes käyttäjä voi valita itselleen käyttöön perinteisen kubectl:n tehokkaan hallinnoinnin osalta sekä siksi, että Grafanan hoitaessa kattavaa datan luentaa, ei ympäristöön välttämättä kahta datan visualisointityökalua kaivata.

8 Loppupohdinta

Tässä työssä käsiteltiin muutamia Kubernetes-ympäristön hallintaan ja monitorointiin soveltuvia ohjelmia. Kuten tuloksista voi päätellä, eri ohjelmat tuki eri ITIL-käytäntöjä ja DevOpsin tuennassakin oli vaihtelua. Kaikki ohjelmat ovat kuitenkin ympäristöstä riippuen varteen otettavia vaihtoehtoja ja näitä on hyvä punnita keskenään. Samankaltaisia ohjelmia, kuin mitä tähän työhön oli valittu, on valtavasti. Kaikkea ei voi kuitenkaan mukaan ottaa, jotta pituus säilyisi maltillisena.

Muita soveltuvia työkaluja, joista olisi hyvä tehdä palvelutuotannon näkökulmasta jatkotutkimusta voisi olla esimerkiksi Openshift, Kustomize, Portainer, Rancher. Ohjelmia päivitetään jatkuvasti ja tämän mukana saattaa tulla uusia ominaisuuksia ja parannuksia, jolloin se, mikä ohjelma sopii mihinkin tilanteeseen, voi muuttua ajan saatossa.

Organisaatioita tämä tutkimus auttaa valitsemaan sopivimman ohjelman Kubernetes-ympäristön hallinnointiin sekä helpottaa tutustumista valitun ohjelman ominaisuuksiin. Oikean ohjelman valit-

seminen, kun vaikuttaa palvelun jatkuvuuteen, tehokkuuteen ja reaktionopeuteen. Myös ohjelmien yhdistäminen ei ole poissuljettua, vaan voi parhaassa tapauksessa tukea toinen toista sekä ITIL:in järjestelmällisyyttä, että DevOpsin ketteryyttä.

Näihin tuloksiin kannattaa suhtautua pienellä varauksella, sillä tämä testi toteutettiin yksinkertaisessa ja pienessä testiympäristössä. Mielikuva hallintatyökalun käytöstä voi jossain määrin erota näistä lopputuloksista, kun kyseessä on esimerkiksi suuri tuotantoympäristö.

Jatkoa ajatellen kannattaa tehdä lisätutkimusta myös muista ohjelmista, joista osasta löytyy varmasti ominaisuuksia, joita ei näissä valikoiduissa ohjelmissa ollut tai toteutus on tehty osittain jollain paremmalla tavalla. Jatkotutkimuksissa on myös hyvä satsata enemmän käyttäjien kokemuksiin sekä turvallisuusnäkökulmaan, sillä turvallisuus on kuitenkin koko palvelutuotannon pohja.

Jatkotutkimuksissa kannattaa myös kiinnittää huomiota siihen, olisiko ohjelmien avulla helppo skaalata tuotantoympäristöä suuremmaksi ja vaikuttaako skaalautuminen niiden käyttöön merkittävästi. Jos vaikutus olisi suuri, on pohdittava, että onko skaalautumisen yhteydessä syytä vaihtaa ohjelmasta toiseen. Kaikissa ohjelmissa on kuitenkin omat puolensa.

Tänä päivänä myös tekoäly on ottanut paljon tilaa erityisesti IT-alalta ja se varmasti alkaa näkyä myös palveluiden hallinnassa. Organisaatioiden on syytä pohtia, olisiko esimerkiksi AIOpsin tai automaatiotyökalujen kuten GitOpsin käyttöön syytä tutustua tarkemmin. Nämä ohjelmat voivat auttaa organisaatioita pyrkimään parempaan kustannustehokkuuteen ja vähentämään manuaalista työtä. Konttiteknologia ja Kubernetes ovat vakiinnuttaneet paikkansa sekä ovat edelleen kasvavassa asemassa IT-palvelutuotannossa. Tästä syystä on aiheellista tehdä lisää jatkotutkimuksia aiheen saralla.

Tämän työn miellyttävin vaihe oli itse havainnoida työkalujen visuaalista ilmettä ja yleisiä ominaisuuksia ja rankimmat olivat teorian kirjoittaminen ja kaikkien ominaisuuksien löytäminen sekä niiden reflektointi ITILiin ja DevOpsiin. Tällaisesta näkökulmasta, kun ei löytynyt tietoa edes englanniksi, oli lähtökohdat tosi hankalat. Ympäristö soveltui melko hyvin kaikkien valittujen työkalujen testaamiseen, vaikkakin asennustavoissa oli jossain määrin merkittäviä eroja.

Tämä työ syvensi ymmärrystä Kuberneteksen arkkitehtuurista sekä palvelutuotannon vaatimuksista. Ymmärrys syveni myös ITILin prosessien ja DevOpsin käytäntöjen yhdistämisen saralla teknisissä ratkaisuissa. Työn pohjalta saa vankkaa tietämystä eri työkaluista järjestelmän ylläpitäjän työelämää varten.

Lähteet

8 Kubernetes Deployment Strategies. 2025. NetApp:in Spot-verkkosivut. Viitattu 26.2.2025. https://spot.io/resources/kubernetes-autoscaling/8-kubernetes-deployment-strategies/.

Anand, A. 2024. How Grafana Retrieves and Visualizes Prometheus Data. SigNoz-verrosivuston opas. Viitattu 24.3.2025. https://signoz.io/guides/how-does-grafana-get-data-from-prometheus/.

CNCF SURVEY. 2020. Cloud Native Computing Foundation (CNCF) tuottama raportti. Viitattu 14.3.2025. https://www.cncf.io/wp-content/uploads/2020/12/CNCF_Survey_Report_2020.pdf.

Command line tool (kubectl). 2024. Kubernetes-verkkosivun dokumentaatio. Viitattu 10.3.2025. https://kubernetes.io/docs/reference/kubectl/.

Deploy and Access the Kubernetes Dashboard. 2024. Kubernetes-sivuston dokumentaatio. Viitattu 25.3.2025. https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard/.

Deployments. 2025. Kubernetes dokumentaatio. Viitattu 25.2.2025. https://kubernetes.io/docs/concepts/workloads/controllers/deployment/.

DevOps vs. ITIL -- Which matters for your team? 2024. atlassian.com-verkkosivut. Viitattu 18.2.2025. https://www.atlassian.com/itsm/itil/devops-vs-itil.

DevOps. N.d. DevOps. itewikin verkkosivustot. Viitattu 19.2.2025. https://www.itewiki.fi/opas/devops/.

Ezinne, A. 2023. Containers: what is containerization and container orchestration? dev.to-verkkosivuston artikkeli. Viitattu 21.4.2025. https://dev.to/ezinne_anne/containers-what-is-containerization-and-container-orchestration-38pe.

Haara, L. 2016. IT-palvelunhallinnan kehittäminen kohdeorganisaatiossa. Diplomityö. Lappeenrannan teknillinen yliopisto. Tietotekniikan koulutusohjelma. Viitattu 18.2.2025. https://urn.fi/URN:NBN:fi-fe201701051069.

Hoffman, K. 2023. Maximizing productivity with Kubernetes: the benefits of using K9s. Civo-verkkosivun opas. Viitattu 6.3.2025. https://www.civo.com/learn/k9s-the-tool-that-can-increase-your-productivity-with-kubernetes.

Hoogenraad, W. 2022. ITIL ja DevOps: ystäviä vai vihollisia? ITpedia. Viitattu 19.2.2025. https://fi.itpedia.nl/2022/10/23/itil-devops-vrienden-of-vijanden/.

Ingress. 2024. Kubernetes-sivuston dokumentaatio. Viitattu 27.3.2025. https://kubernetes.io/docs/concepts/services-networking/ingress/.

ITIL. 2023. wakaru.fi-verkkosivut. Viitattu 14.2.2025 https://www.wakaru.fi/palvelujohtaminen/itil/.

Janashia, N. 2020. Kubernetes Services explained | ClusterIP vs NodePort vs LoadBalancer vs Headless Service. Youtube-video. Lataaja TechWorld with Nana. Viitattu 19.3.2025. https://www.youtube.com/watch?v=T4Z7visMM4E

K9s. 2024. K9s:n virallinen verkkosivu. Viitattu 27.2.2025. https://k9scli.io/.

Kaikkonen, A. 2021. ITIL-VIITEKEHYKSEN TOTEUTUMINEN L2-TASON IT-TUESSA. Opinnäytetyö, AMK. Hämeen ammattikorkeakoulu. Tieto- ja viestintätekniikan tutkinto-ohjelma. Viitattu 18.2.2025. https://urn.fi/URN:NBN:fi:amk-202104235578.

Kdash. 2025. KDash:in Github-sivu. Viitattu 28.2.2025 https://github.com/kdash-rs/kdash

Kesanto, A. 2021. Teknisen työnkuvan tiketöintijärjestelmän kehitys ITIL:n standardien mukaiseksi eri sidosryhmien välillä Yrityksessä X. Opinnäytetyö, AMK. Haaga-Helia ammattikorkeakoulu. Tietojenkäsittelyn koulutusohjelma. Viitattu 25.3.2025. https://urn.fi/URN:NBN:fi:amk-2021100818459.

Kotala, M. 2022. IT-palvelutuotannon suorituskyvyn mittarit. Opinnäytetyö, Ylempi AMK. Hämeen ammattikorkeakoulu. Tietojohtaminen ja älykkäät palvelut. Viitattu 18.2.2025. https://urn.fi/URN:NBN:fi:amk-2022112824622.

Kuutti, P. 2024. Erilaiset työkalut voivat tehostaa DevOps-kehitysprosessia – erityisesti Azure DevOps -ympäristössä-blogi. Viitattu 10.3.2025. https://www.nico.fi/blogi/erilaiset-tyokalut-voivat-tehostaa-devops-kehitysprosessia-erityisesti-azure-devops-ymparistossa.

Leppänen, T. 2021. Data visualization and monitoring with Grafana and Prometheus. Opinnäytetyö, AMK. Turun ammattikorkeakoulu. Tieto- ja viestintätekniikan tutkinto-ohjelma. Viitattu 24.3.2025. https://urn.fi/URN:NBN:fi:amk-2021122090248.

Leskinen, A. 2019. MIKROPALVELUARKKITEHTUURIN KÄYTTÄMINEN PILVIPALVELUSSA. Opinnäytetyö, AMK. Turun ammattikorkeakoulu. Tietojenkäsittelyn tutkinto-ohjelma. Viitattu 25.2.2025. https://urn.fi/URN:NBN:fi:amk-2019052111007.

Marquez, E. 2023. pluralsight.com-verkkosivuston blogi. Viitattu 14.4.2025. https://www.pluralsight.com/resources/blog/cloud/history-of-container-technology.

Melnyk, R. 2025. Kubernetes DaemonSet: Practical Guide to Monitoring in Kubernetes-blogi. Viitattu 26.2.2025. https://cast.ai/blog/kubernetes-daemonset-practical-guide-to-monitoring-in-kubernetes/.

MicroK8s documentation – home. 2024. Canonicalin luomat Microk8s:n omat verkkosivun dokumentti. Viitattu 5.5.2025. https://microk8s.io/docs.

Mikä on Kubernetes ja miten se toimii? 2024. Contrasec:in artikkeli. Viitattu 19.2.2025. https://contrasec.fi/mika-on-kubernetes-ja-miten-se-toimii/.

Mitä on DevOps - mallin periaatteet ja hyödyt. 2023. haltu.fi-verkkosivun blogi. Viitattu 7.5.2025. https://www.haltu.fi/blogi/mita-on-devops.

Mitä tarkoittaa DevOps? 2024. Hurja verkkosivuston blogi. Viitattu 19.2.2025. https://www.hurja.fi/blogi/mita-tarkoittaa-devops/.

Mottammal, A. 2024. Understanding Bearer Tokens: Usage, Examples, and Differences from API Keys. medium.com-verkkosivut. Viitattu 1.4.2025. https://medium.com/@akhil-mottammal/understanding-bearer-tokens-usage-examples-and-differences-from-api-keys-496f9bfb6038.

Palveluliiketoiminnan sanasto. 2010. Businessfinland.fi-verkkosivuilta löytyvä pdf. Viitattu 18.2.2025. https://www.businessfinland.fi/globalassets/julkaisut/palveluliiketoiminnan_sanasto.pdf.

Pricing. 2025. Grafanan verkkosivut. Viitattu 6.3.2025. https://grafana.com/pricing/.

Rabinovich, Y. 2024. Master Grafana Dashboards for Kubernetes Monitoring. Groundcover-sivuston blogiteksti 15.12.2024. Viitattu 10.3.2025. https://www.groundcover.com/blog/grafana-kubernetes.

ReplicaSet. 2025. Kubernetes dokumentaatio. Viitattu 26.2.2025. https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/.

Salo, S. 2018. ITIL-prosessikehys suuren IT-alan yrityksen Service Deskissä. Insinöörityö, AMK. Metropolia Ammattikorkeakoulu. Tieto- ja viestintätekniikan tutkinto-ohjelma. Viitattu 14.2.2025. https://urn.fi/URN:NBN:fi:amk-201805087081.

Saviranta, S. 2019. Palvelutuotannon kehittäminen : Miten parantaa suorituskyvyn edellytyksiä? Opinnäytetyö, Ylempi AMK. Metropolia Ammattikorkeakoulu. Tradenomi, Liiketoiminnan kehittämisen tutkinto-ohjelma. Viitattu 18.2.2025. https://urn.fi/URN:NBN:fi:amk-201905139391.

StatefulSets. 2025. Kubernetes dokumentaatio. Viitattu 26.2.2025. https://kubernetes.io/docs/concepts/workloads/controllers/statefulset/.

The Evolution of ITIL: From ITIL v3 to ITIL 4 – What's Changed and Why It Matters. 2024. imcinstitute.ae-verkkosivut. Viitattu 2.4.2025. https://imcinstitute.ae/the-evolution-of-itil-from-itil-v3-to-itil-4-whats-changed-and-why-it-matters.

Tonteri, M. 2022. Kubernetes-klusterin tilan dokumentointi ohjelmallisesti. Opinnäytetyö, AMK. Jyväskylän ammattikorkeakoulu. Tradenomi, tietojenkäsittelyn koulutusohjelma. Viitattu 19.2.2025. https://urn.fi/URN:NBN:fi:amk-202205128746.

Tuomala, T. 2020. Kubernetes ajoympäristön automatisoitu käyttöönotto modernissa pilviympäristössä. Maisteritutkielma. Helsingin yliopisto. Tietojenkäsittely. Viitattu 26.2.2025. http://hdl.handle.net/10138/315163.

Use containers to Build, Share and Run your applications 2025. Dockerin verkkosivustot. Viitattu 25.2.2025. https://www.docker.com/resources/what-container/.

Wallenius, N. 2022a. Konttiteknologia – mitä kontit ovat ja mitä hyötyä niistä on? Niklas Walleniuksen nettisivun artikkeli. Viitattu 19.2.2025. https://niklaswallenius.fi/konttiteknologia-mita-hyotya/

Wallenius, N. 2022b. Mikä on Kubernetes ja mitä hyötyä siitä on? Niklas Walleniuksen nettisivun artikkeli. Viitattu 19.2.2025. https://niklaswallenius.fi/kubernetes/.

What is ITIL? 2023. milldesk.com-verkkosivun blogi. Viitattu 7.5.2025. https://www.milldesk.com/what-is-itil-2/.

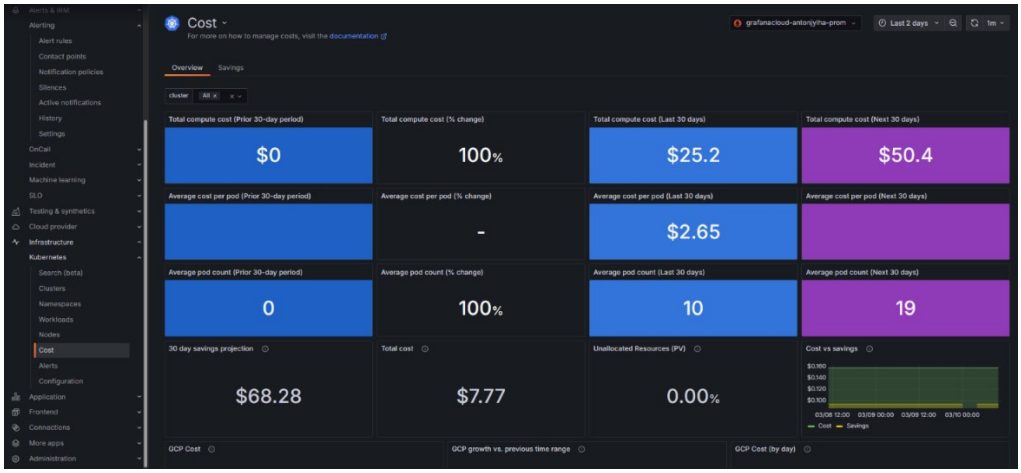
Winter, A. N.d. Mikä on DevOps? itewikin verkkosivustot. Viitattu 19.2.2025. https://www.itewiki.fi/p/mika-on-devops.

Liitteet

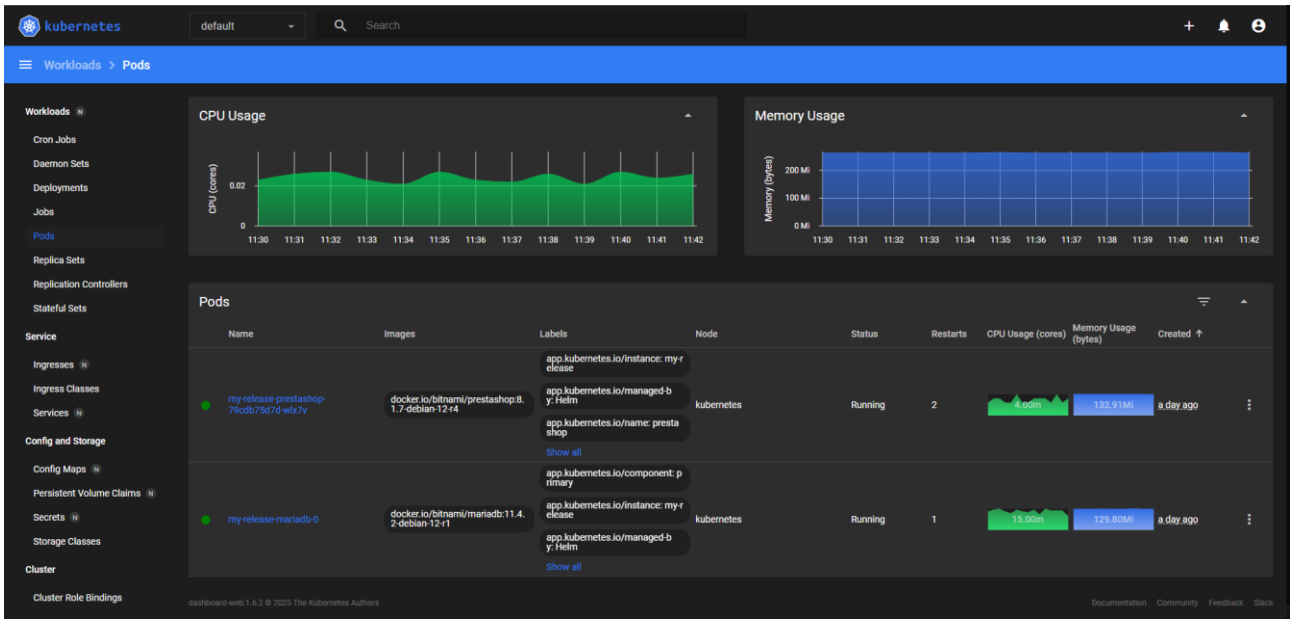
Liite 1. Grafanan eri tilauksien ominaisuudet ja hinnat (Pricing 2025)

Ominaisuus	Free Forever ($0)	Pro ($19/kk + käyttöperusteinen hinnoittelu)	Advanced ($299/kk + käyttöperusteinen hinnoittelu)
Hinta	Ilmainen	$19/kk peruskäytöllä + lisämaksut ylityksistä	$299/kk peruskäytöllä + lisämaksut ylityksistä
Metriikat	10000 metriikkasarjaa, 14 päivää säilytys	10000 metriikkasarjaa, 13 kk säilytys	20000 metriikkasarjaa, 13 kk säilytys
Lokitiedot, jäljitystiedot, profiilitiedot	50 GB jokainen, 14 päivää säilytys	50 GB jokainen, 30 päivää säilytys	100 GB jokainen, 30 päivää säilytys
Tapahtuma- ja riskienhallinta, IRM (Incident & Risk Management)	3 aktiivista käyttäjää	3 käyttäjää sisältyy, lisäkäyttäjät $20/kpl	5 käyttäjää sisältyy, lisäkäyttäjät $20/kpl
Sovellusten havainnointi	2232 host-tuntia	2,232 host-tuntia sisältyy, lisäkäyttö $0.04 per host hour	3,720 host-tuntia sisältyy, lisäkäyttö $0.04 per host hour
Kubernetes Monitorointi	2200 host / 37000 kontti tuntia	Sama kuin ilmainen, lisäkäyttö $0.015 per host hour	Sama kuin Pro
Frontend Observability	50000 istuntoa	50000 istuntoa sisältyy, lisäkäyttö $0.90 per 1000 istuntoa	100000 istuntoa sisältyy, lisäkäyttö $0.90 per 1000 istuntoa
Synteettiset testisuoritukset	100000 testejä	100000 testejä sisältyy, lisäkäyttö $5 per 10000 testiä	200k testejä sisältyy, lisäkäyttö $5 per 10000 testiä
k6 Suorituskykytestaus	500 VUh, 14 päivää säilytys	500 VUh, 30 päivää säilytys, lisäkäyttö $0.015 per VUh	1000 VUh, 180 päivää säilytys, lisäkäyttö $0.015 per VUh
Tuki	Yhteisön tuki	8/5 tukipalvelu	24/7 tukipalvelu

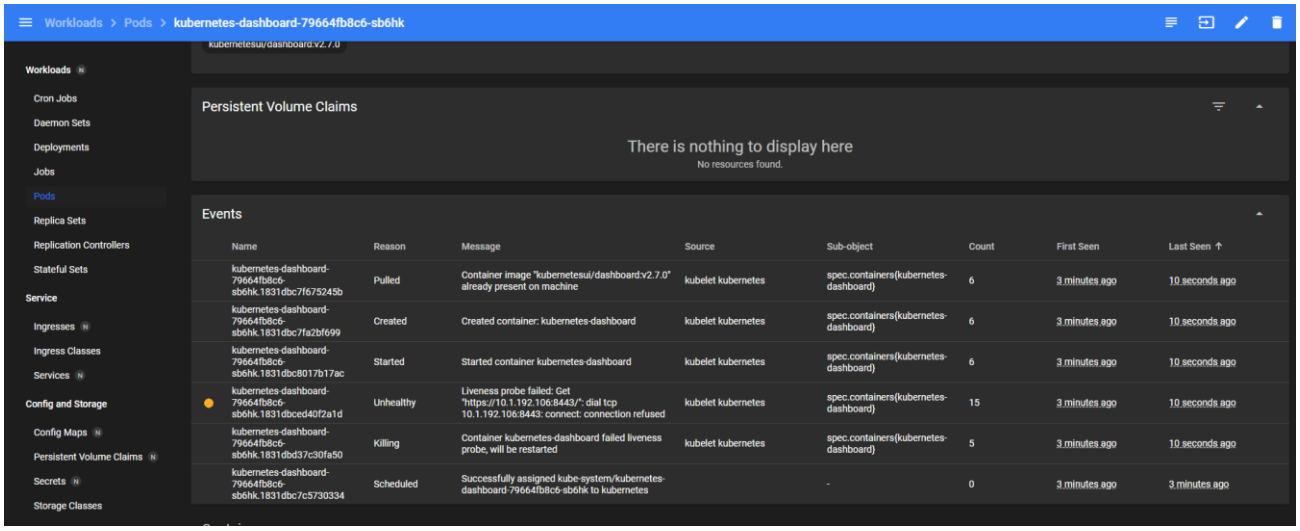
Liite 2. ”Cost” -näkymä Grafanassa viimeisen kahden päivän ajalta



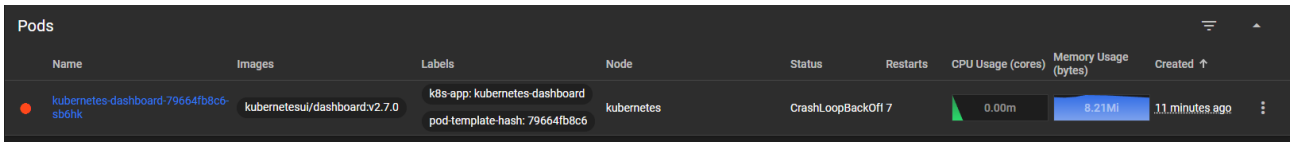
Liite 3. "Pods"-näkymä Kubernetes Dashboardissa



Liite 4. Yksittäisen podin tapahtumat Kubernetes Dashboardissa



Liite 5. Virheellinen podin indikaattori Kubernetes Dashboardissa



Liite 6. Kubectl olennaisimmat komennot, joilla monitoroida klusteria

Seuraava komento listaa kaikki klusterin nodet:

```
Kubectl get node
```

Seuraava komento listaa kaikista klusterin serviceista:

```
Kubectl get svc
```

tulostaa näytölle halutun podin lokitiedot:

```
kubectl logs <podin nimi>
```

tulostaa näytölle halutun podin lokitiedot jatkuvana reaaliajassa, joka edistää jatkuvaa valvontaa:

```
kubectl logs -f <podin nimi>
```

Seuraavalla komennolla voi luoda uusia resursseja ympäristöön, alla esimerkki loadbalancer servicesta:

```
kubectl create service loadbalancer <nimi> <optio>
```

Seuraava komento ottaa käyttöön konfiguraatiotiedoston asetukset Kubernetes-ympäristössä:

```
Kubectl apply -f <tiedoston nimi>
```

Seuraavan komennon avulla voi editoida konfiguraatiotiedostoja ja deploymentteja (editointityökaluna toimii Vim).

```
kubectl edit -f <tiedoston nimi>

kubectl edit deployment <deploymentin nimi>
```

Kubectl delete -komennolla voi poistaa muun muassa yksittäisiä podeja, nodeja, klusterin, resursseja, statefulsetteja, replicasetteja, serviceja tai vaikka kaikki deploymentit. Alla pari esimerkkiä:

```
kubectl delete pod <podin nimi>

kubectl delete deployment -all
```

Kubectl tukee myös varmuuskopiointia sen verran, että kaikki resurssit (Deployments, Services, ConfigMaps jne.) voi tallentaa versionhallintaa varmuuskopiona seuraavanlaisella komennolla:

```
Kubectl get all --all-namespaces -o yaml > backup.yaml
```

Ja palauttaa komennolla:

```
kubectl apply -f backup.yaml
```

Seuraavan komennon avulla tehdään useita verkkoon liittyviä muutoksia, kuten antaa serviceille käyttöön Klusterin ulkoinen IP, antaa kontille tai servicelle haluttu portti, johon ohjata liikenne. Alla pari optiota:

```
kubectl expose
```

On myös mahdollista listata kaikkien podien tai nodejen resurssien käyttö (cpu, muisti) laittamalla seuraava komento, jossa käytetään esimerkkinä podia:

```
kubectl top pod
```

Jos käyttäjä haluaa saada lisää tietoa, mitä mikäkin komponentti (kuten podit, nodet, servicet jne.) Kubernetes-ympäristössä tekee, tähän löytyy myös ratkaisu komennolla

```
Kubectl explain <type>
```

Kubectl (sekä K9s) näyttää prosessorin, että muistin käytön vain, jos metrics-server on asennettu. Asennus tapahtuu komennolla:

```
Microk8s kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

Lisää komentoja ja optioita voi löytää muun muassa kubectl:stä käsittelevältä Kuberneteksen sivulta: https://kubernetes.io/docs/reference/kubectl/.

Liite 7. Kubernetes Dashboard asennusohjeet

Tässäkin microk8s:lla on helm omana pakettina, joten tässä tapauksessa voi käyttää komentoa ”microk8s helm” tai tehdä aliaksen ”alias helm=’microk8s helm’”. Tässä tapauksessa on tehty alias.

Ensiksi piti ottaa dashboard käyttöön

```
microk8s enable dashboard
```

jonka jälkeen lisättiin kubernetes dashboard repository:

```
microk8s helm repo add kubernetes-dashboard https://kubernetes.github.io/dashboard/
```

Asennettiin Helm-release nimeltä "kubernetes-dashboard" käyttämällä kubernetes-dashboard-charttia:

```
microk8s helm upgrade --install kubernetes-dashboard kubernetes-dashboard/kubernetes-dashboard --create-namespace --namespace kubernetes-dashboard
```

Dashboardiin saa muodostettua yhteyden seuraavan komennon avulla:

```
microk8s kubectl -n kubernetes-dashboard port-forward svc/kubernetes-dashboard-kong-proxy 8443:443
```

Jotta saa dashboardin näkymään Windows-koneella, pitää muokata alla olevaa servicen konfiguraatiotiedostoa seuraavalla komennolla:

```
microk8s kubectl edit service kubernetes-dashboard-kong-proxy -n kubernetes-dashboard
```

Tiedostossa scrollataan alas ja muuttaa ”type:” kohtaan ”NodePort” ja ottaa vielä ylös nodePortin portin numeron, joka testissä on 30241.

```
  - name: kong-proxy-tls
    nodePort: 30241
    port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    app.kubernetes.io/component: app
    app.kubernetes.io/instance: kubernetes-dashboard
    app.kubernetes.io/name: kong
  sessionAffinity: None
  type: NodePort
```

Viimeiseksi luodaan serviceaccount:

```
microk8s kubectl create serviceaccount <käyttäjän nimi>
```

Luodaan tälle token:

```
microk8s kubectl -n kubernetes-dashboard create token <käyt-
täjän nimi>
```