TIFOSILINUX©

The Origin Community Distribution of Kubernetes

Panduan Mudah OKD - OpenShift

TUNTUNAN SEDERHANA, MUDAH, DAN PASTI BISA

with Docker, Kubernetes, Pods, etc

Panduan Mudah OKD - OpenShift

Tuntunan sederhana, mudah, dan pasti bisa

Hary Cahyono

Tifosilinux©

Panduan Mudah OKD - OpenShift

Hary Cahyono

Sejarah Revisi untuk Edisi Pertama

Tulisan ini adalah sebagian kecil dari tulisan-tulisan yang telah penulis *published* pada berbagai situs publikasi dokumen, *images*, video, atau presentasi baik gratis untuk publik maupun *private* seperti slideshare, academia, scribd, wordpress, dan lainnya.

Wording Tifosilinux memiliki hak cipta dan merupakan identitas penulis dalam menerbitkan atau mempublikasikan berbagai tulisan di berbagai bidang.

Penulis memberikan kebebasan kepada para pembaca yang telah mendapati berbagai tulisan yang telah di *upload* pada berbagai media digital untuk dapat menyebarluaskan dan memanfaatkan sebagaimana mestinya. Termasuk yang sifatnya gratis maupun yang berbayar.

Terlepas dari semua itu, penulis tidak bertanggung jawab untuk seluruh *output* atau hasil yang berbeda dari tulisan, *namun keberhasilkan seluruh program dapat berjalan dengan baik sesuai dengan tuntunan dan teori dari penulis telah terbukti dari hasil screenshot dan data yang disajikan*. Selalu menjadi sebuah tanggung jawab bagi penulis untuk memberikan fakta data dan referensi yang aktual disetiap tulisan.

Daftar Isi

Kata Pengantar

OpenShift adalah sebuah bentuk *polyglot platform*, disebut demikian karena penggunaan platform yang menggunakan lebih dari satu teknologi platform (sistem yang dibangun diatas platform). Biasanya OpenShift digunakan untuk membangun sebuah aplikasi web dan services, yang mana menggunakan sekumpulan container yang berkaitan dengan lingkungan SELinux (*Security-Enhanced Linux* – Otorisasi untuk keamanan di sistem operasi Linux). Kita dapat memasang OpenShift pada infrastruktur kita sendiri atau pada cloud yang berada di publik, atau bahkan kita bisa memanfaatkan layanan OpenShift secara *online* yang telah disediakan oleh Red Hat secara langsung di **Red Hat** OpenShift.

Versi terbaru dari OpenShift menggunakan standar platform Kubernetes dari CNCF (Cloud Native Computing Foundation) untuk mengelola dan menjalankan aplikasi di dalam sekumpulan container. Secara mendasar, OpenShift merupakan sebuah platform aplikasi dengan konsep PaaS (Platform as a Service) yang lengkap.

Kemudian muncul OKD – yang masih didukung oleh Red Hat, dengan lisensi *community*-nya agar bisa dimanfaatkan oleh komunitas meski dengan fitur terbatas. Tulisan OKD – OpenShift ini dimaksudkan untuk para developer atau system administrator yang mencoba mengevaluasi dan memutuskan menggunakan OpenShift versi *community* untuk mendukung proses *deployment* dan *development* (DevOps). Hal tersebut diharapkan dapat menjadi panduan bagi mereka yang sedang dan ingin menerapkan OKD secara cluster maupun *standalone*.

BAB 1

Mengenal OpenShift

Platform OpenShift diluncurkan pada Mei 2011. Dimana *source code* atau kode sumbernya dibuat *open source* atau tersedia secara gratis. Pada situsnya, Red Hat menawarkan sejumlah dukungan untuk OpenShift untuk bisa digunakan di skala enterprise, dan layanan hosting yang disebut OpenShift online. Versi 1.0 dari OpenShift Origin, yang berbasis Kubernetes dan Docker container runtime, dirilis pada bulan Juni 2015.

Lalu apa itu OpenShift ?

Pada dokumentasi resminya, Red Hat mengklaim bahwa OpenShift 100% comply dengan konsep Kubernetes. Walaupun masih muncul pertanyaan dari para *Red Hatters* mengenai "Apakah Red Hat OpenShift benar-benar mengimplementasikan konsep Kubernetes ?". Pertanyaan-pertanyaan lain muncul perihal; apakah Red Hat OpenShift mendukung *command* kubectl ?, apakah Red Hat OpenShift mendukung proses deployment dengan Kubernetes ?, apakah Red Hat OpenShift mendukung Pod Security Policies (PSPs) ?, dan berbagai macam pertanyaan lainnya.

Namun pada intinya, OpenShift merupakan sebuah platform yang membantu kita untuk mengembangkan dan men-deploy aplikasi web, baik frontend maupun backend meliputi layanan microservices atau database. Aplikasi tersebut merupakan bentuk implementasi dari berbagai bahasa pemrograman. OpenShift (berikut juga OKD – kita akan bahas juga nanti), dapat berjalan pada berbagai jenis Sistem Operasi Red Hat Enterprise Linux (RHEL), CentOS, atau Fedora. Walaupun pada suatu kasus kita bisa menggunakan container yang berbasis image Debian atau Ubuntu.

Pada bab awal ini kita akan mengenal teknologi yang OpenShift tawarkan.

The National Institute of Standards and Technology (NIST) mendefinisikan cloud computing menjadi tiga model standar layanan dari cloud computing:

Software as a Service (SaaS)

Kemampuan untuk menyediakan layanan kepada konsumer berupa software aplikasi yang berjalan di atas infrastruktur yang berbasis cloud. Aplikasi dapat diakses dari berbagai perangkat dengan interface client, baik berupa web browser maupun program GUI. Konsumer tidak perlu repot dengan pengetahuan tentang cloud infrastruktur meliputi network, server, dan sistem operasi.

Platform as a Service (PaaS)

Kemampuan untuk menyediakan layanan kepada konsumer untuk men-deploy aplikasi yang telah dibuat – menggunakan berbagai macam bahasa pemrograman, library/ pustaka, services, dan tools pendukung – ke dalam infrastruktur cloud. Konsumer tidak perlu repot dengan pengetahuan tentang infrastruktur dari cloud meliputi network, server, dan sistem operasi namun memiliki kontrol terhadap aplikasi yang di-deploy dan konfigurasi untuk environment aplikasi.

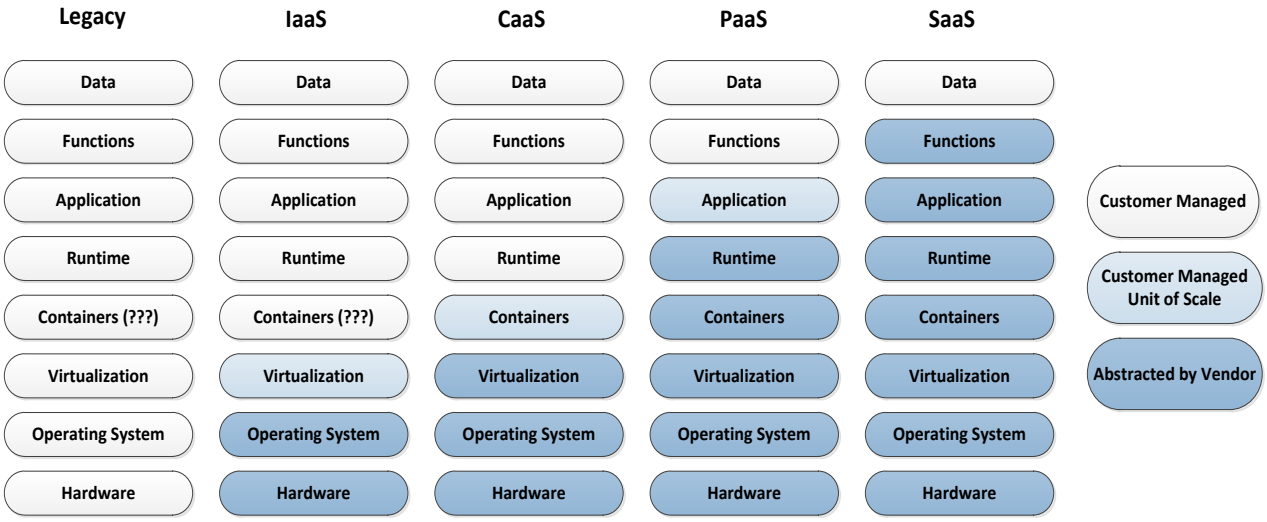
Infrastructure as a Service (IaaS)

Kemampuan menyediakan layanan kepada konsumer untuk persediaan processor, storage, network, dan sumber daya lain sehingga dapat men-deploy dan menjalankan aplikasi di dalam Sistem Operasi. Konsumer tidak perlu repot mengelola cloud infrastruktur namun memiliki kontrol terhadap sistem operasi, storage, dan aplikasi yang telah di-deploy, serta kontrol terbatas terhadap komponen network (semisal firewall)

Berdasarkan definisi tersebut untuk model layanan cloud computing, maka OpenShift diklasifikasikan sebagai sebuah layanan PaaS. Baik model PaaS dan SaaS, kontainerisasi sering digunakan untuk memisahkan aplikasi satu dengan yang lainnya dan dari user yang berbeda. Sebagai sebuah teknologi, kontainerisasi memiliki sejarah panjang, dengan freeBSD jails dan Solaris zones sebagai pelopornya. Di Linux, dukungan untuk kontainerisasi berporos pada proyek Linux Containers (LXC).

Menggunakan kemampuan CaaS (Container as a Service) dari Kubernetes, OpenShift mampu men-deploy aplikasi dari container yang disimpan di dalam images dengan banyak image registry. Kubernetes sendiri tidak menyediakan banyak dukungan untuk membangun image container. Kita perlu menjalankan sebuah *build tool* untuk membuat image aplikasi ke sistem dan memasukkannya ke dalam image registry. Hal ini dikarenakan CaaS hanya fokus

menjalankan container dan memiliki kekurangan di sisi PaaS, dimana kita dapat mengakses source code dan mengetahui cara kerja platform dalam menjalankan container.



Gambar 1. Layanan Cloud

Untuk memenuhi kemampuan di sisi PaaS yang mampu menghandle source code dan men-deploy-nya, OpenShift menambahkan otomasi untuk performansi build di atas Kubernetes. OpenShift mendukung dua cara untuk membangun source code.

Yang pertama, seperti pada model tradisional PaaS, OpenShift dapat mem-build source code menggunakan builder bahasa pemrograman yang kita pilih. Sebagai contoh, jika kita ingin membangun aplikasi ruby on rails maka OpenShift menyediakan template untuk membangun source code aplikasi tersebut, menciptakan image aplikasi hingga container berjalan. Sebagai seorang developer, kita tidak perlu repot menuliskan berbagai macam perintah untuk membuat atau menjalankan sebuah image container, OpenShift yang akan melakukan hal tersebut.

Sebagai tambahan, jika kita terbiasa mengambil source code hingga menciptakan image aplikasi menggunakan satu set instruksi pada berkas Dockerfile, kali ini kita tidak perlu repot, OpenShift akan melakukannya. Meskipun pada kenyataannya kita tetap bisa menggunakan cara manual tersebut dan meregisternya kedalam platform OpenShift.

Yang kedua, OpenShift akan mencatat setiap image aplikasi dalam bentuk cache ke dalam image registry. Setiap image aplikasi yang akan di deploy berasal dari internal image registry ini.

Melalui dua pendekatan dalam hal metode build tersebut, OpenShift dapat mengunduh source code dari repositori Git untuk kemudian di build. Jika kita menggunakan layanan seperti GitHub, GitLab, atau BitBucket, kita dapat melakukan konfigurasi repositori untuk kemudian mengirimkan notifikasi ke OpenShift setiap kali kita melakukan *push* perubahan pada Git repositori. Notifikasi ini akan menjadi trigger untuk development dan proses build yang dilakukan pada aplikasi.

Karena dibangun di atas Kubernetes, membuat OpenShift memiliki peran baik sebagai sebuah CaaS maupun PaaS. Kita dapat melakukan deployment aplikasi berdasarkan request atau mengimpor aplikasi pihak ketiga atau *third-party* seperti database, sistem messaging, atau aplikasi lain yang mendukung proses bisnis.

Mengenal OKD (OpenShift Kubernetes Distribution)

Dikutip langsung dari situsnya, OKD adalah distribusi dari Kubernetes yang telah di optimasi untuk pengembangan aplikasi yang berkesinambungan dan multi-tenant. OKD menambahkan tools untuk para developer dan sistem operations untuk membangun aplikasi dengan cepat di atas Kubernetes, bahkan untuk maintenance jangka panjang.

Apa yang bisa kita jalankan di OKD – OpenShift ?

OKD di desain untuk menjalankan Kubernetes dalam jumlah yang besar, dibangun dan dikembangkan dengan konsep kontainerisasi.

Pada web dashboard OKD – OpenShift, kita dapat melihat beberapa default service catalog yang bisa digunakan, beberapa diantaranya adalah:

- .NET Core
- .NET Core + PostgreSQL (Persistent)
- 3scale-gateway
- Apache HTTP Server (httpd)
- CakePHP + MySQL
- CakePHP + MySQL (Ephemeral)
- Dancer + MySQL (Perl Web Framework)
- Dancer + MySQL (Ephemeral) (Perl Web Framework)
- Django + PostgreSQL
- Django + PostgreSQL (Ephemeral)
- Jenkins
- Jenkins (Ephemeral)
- MariaDB
- MariaDB (Ephemeral)

- MongoDB
- MongoDB (Ephemeral)
- MySQL
- MySQL (Ephemeral)
- Nginx HTTP server dan reverse proxy (nginx)
- Node.js
- Node.js + MongoDB
- Node.js + MongoDB (Ephemeral)
- Perl
- PHP
- PostgreSQL
- PostgreSQL (Ephemeral)
- Python
- Rails + PostgreSQL
- Rails + PostgreSQL (Ephemeral)
- Redis
- Redis (Ephemeral)
- Ruby
- Registry-console
- System
- WildFly

Kita juga dapat membuat sendiri catalog yang telah dikustomisasi atau memperolehnya dari GitHub atau Git repositori yang tersedia secara publik, misalnya saja catalog untuk service github-ce, Red Hat OpenJDK 8, JBOSS Web Server, Apache Tomcat 8 + PostgreSQL (with https), dan masih banyak lagi.

Kontributor OKD – OpenShift

OKD dibangun dari berbagai macam proyek Open Source, mulai dari Fedora CoreOS, CentOS dan UBI RPM ecosystem, cri-o, Kubernetes, dan masih banyak lagi. Organisasi atau forum dari OpenShift yang tergabung di dalam GitHub secara aktif memegang peranan dalam

pengembangan komponen-komponen yang ada di atas Kubernetes dan membuat dokumentasi-dokumentasi serta referensi proyek untuk dibagikan secara luas.

Pada ebook ini kita akan belajar bagaimana mengintegrasikan aplikasi ke dalam platform OKD – OpenShift, bagaimana melakukan konfigurasi melalui OKD – OpenShift, bagaimana membuat persistent volume, bagaimana agar platform OKD – OpenShift dan aplikasi yang di build di dalamnya dapat diakses secara publik, serta bagaimana memonitor dan melakukan debugging.

Untuk berinteraksi dengan OKD – OpenShift, kita bisa menggunakan mode web console atau command-line di sisi client. Pada tulisan ini kita akan mengetahui dan mempelajari sedikit banyak perihal dua cara tersebut.

Salah satu bentuk aplikasi yang kita jadikan contoh adalah aplikasi dengan framework CakePHP + MySQL yang menggunakan persistent volume atau menggunakan volume sementara yang dikenal dengan istilah Ephemeral.

Source code atau kode sumber dari platform OKD – Openshift lengkap dengan file konfigurasi inventory-nya dapat diperoleh di GitHub melalui link yang saya sisipkan di dalam bab ebook ini.

Requirement OKD - OpenShift

Kebutuhan minimal sistem untuk platform OKD – OpenShift ini sebenarnya tergantung seberapa besar kebutuhan kita kedepannya. Jika terjadi peningkatan bisnis di perusahaan, sebagai contoh pendapatan yang eksponensial menuntut sistem yang bisa di scale up, lebih handal, serta mampu mengakomodir kebutuhan para stockholders /shareholders atau stakeholders. Jika berbicara teknis, environment development atau staging, serta production harus memadai dan robust.

Salah satu requirement yang penulis rekomendasikan untuk uji coba adalah cukup dengan pont-point berikut, meskipun pada kenyataannya dan implementasinya nanti kita perlu melakukan scale up. Berikutnya, karena kita menggunakan konsep dan teknologi cluster, maka sedikitnya kita membutuhkan 3 node yang memiliki peran sebagai master, worker, dan infra.

Requirement minimal untuk node-master:

- GNU/ Linux: CentOS Linux Release 7.9.2009 (Core)
- CPU: 2
- Memory: 8 GB
- Storage1 (/dev/sda): 40 GB
- Storage2 (/dev/sdx): 10 GB
- Ansible version 2.6.10
- SSH version: OpenSSH_7.4p1, OpenSSL
- OKD - OpenShift version:
    - OpenShift Master : v3.11.0+1cd89d4-542
    - Kubernetes Master : v1.11.0+d4cacc0
    - OpenShift Web Console : v3.11.0+ea42280

Requirement minimal untuk node-worker:

- GNU/ Linux: CentOS Linux Release 7.9.2009 (Core)
- CPU: 2
- Memory: 3 GB
- Storage1 (/dev/sda): 40 GB
- Storage2 (/dev/sdx): 10 GB
- SSH version: OpenSSH_7.4p1, OpenSSL

Requirement minimal untuk node-infra:

- GNU/ Linux: CentOS Linux Release 7.9.2009 (Core)
- CPU: 2
- Memory: 3 GB
- Storage1 (/dev/sda): 40 GB
- Storage2 (/dev/sdx): 10 GB
- SSH version: OpenSSH_7.4p1, OpenSSL

Package OKD – OpenShift memiliki *bundling* versi Kubernetes dan Docker (yang akan dipasang pada ketiga node tersebut) serta Web Console tersendiri.

Salah satu hal yang perlu diperhatikan adalah kesesuaian versi dari package yang digunakan atau dependensi dari package. Karena satu package dengan package yang lain saling berkaitan dan mempengaruhi parameter di dalam konfigurasi untuk bisa dijalankan. Sebagai contoh, parameter yang digunakan untuk me-*load* SSL berikut ini terlihat benar, namun pada suatu kasus akan terjadi masalah inkompatibilitas versi ansible-playbook yang digunakan.

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

Potongan konfigurasi tersebut merupakan bagian dari template OKD – OpenShift yang akan kita lihat lebih jauh pada bab 5.

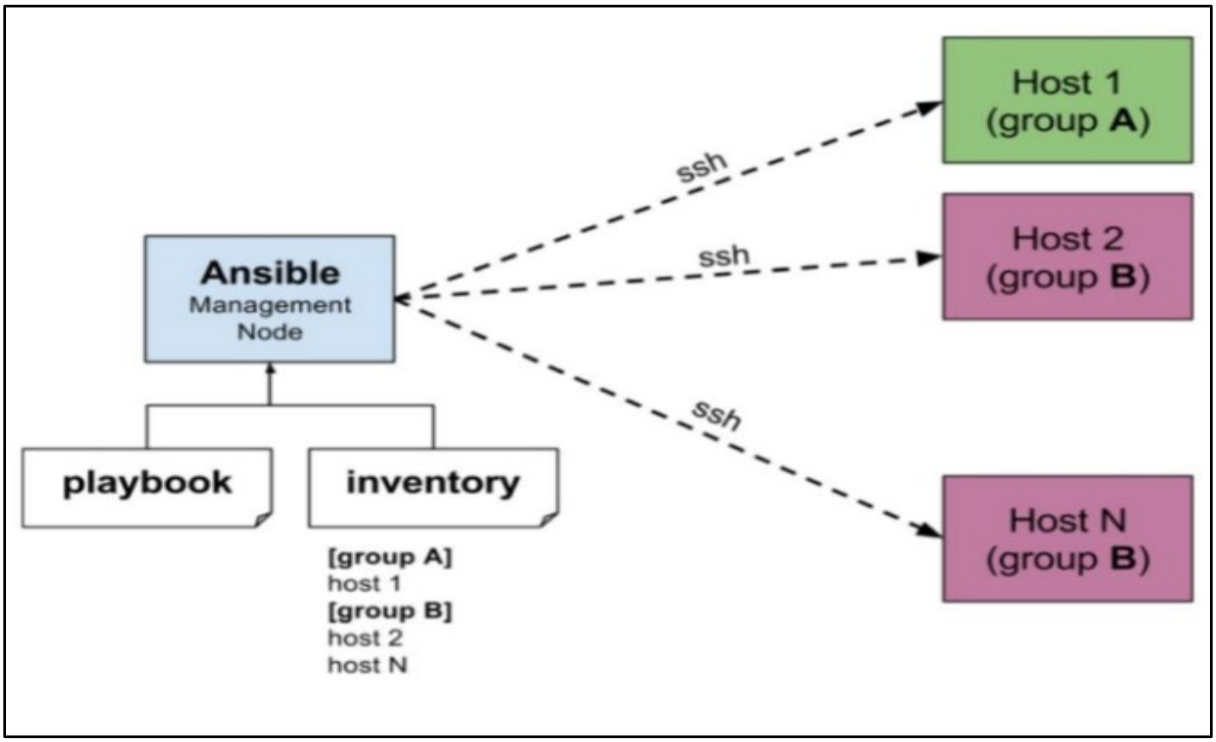
Requirement minimal di atas berjalan baik dan cukup untuk dijadikan bahan uji coba yang penulis implementasikan untuk staging di perusahaan. Kemudian, syarat requirement diatas juga menjadi subrequirements untuk packages yang otomatis terpasang nantinya ketika kita mengaplikasikan template dan inventory OKD – OpenShift. Jadi, pastikan syarat minimal tersebut terpenuhi.

Topologi OKD - OpenShift

Topologi disini mengacu kepada konsep availability dari OKD – OpenShift yang kita implementasikan dengan konsep cluster. Dimana terdiri dari 3 node yang terdiri dari:

- Node Master
- Node Worker atau Node Compute, dan
- Node Infra

Node Master, merupakan core dari cluster OKD – OpenShift yang akan kita bangun dimana seluruh packages yang dibutuhkan atau yang dikenal sebagai dependencies packages akan di *spread* dari node ini. Metode yang digunakan adalah dengan memanfaatkan Command Control Tools, yakni ansible. Secara sederhana, metode *spread* dengan ansible bisa di ilustrasikan sebagai berikut.



Gambar 2. Ansible Flow