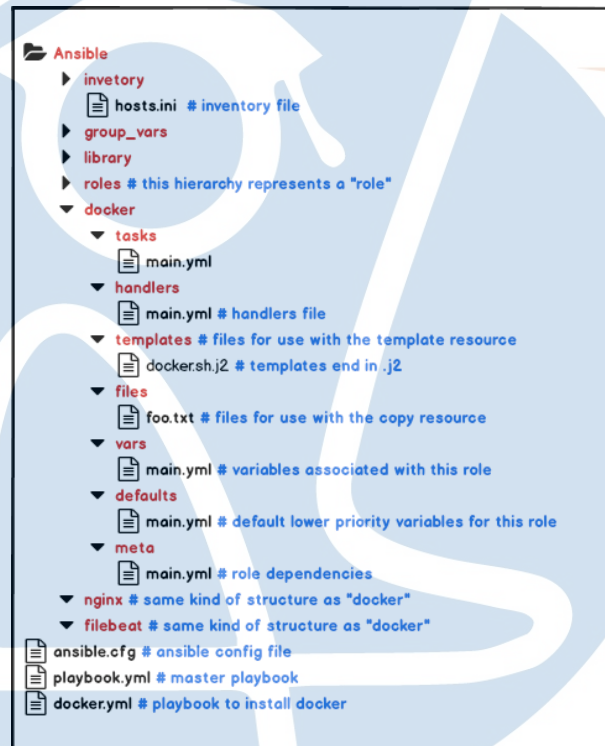


Dengan kata lain *software* yang dapat membantu seorang DevOps untuk melakukan automasi di *server*nya. Ansible ditulis dalam satu bahasa sederhana yang disebut YAML.

Berbeda dengan Puppet dan Chef, Ansible bersifat *agent-less*, yang artinya kita tidak perlu memiliki agent tambahan yang terpasang sebelumnya karena Ansible bekerja cukup pada koneksi ssh. Struktur dasar pada Ansible dapat dilihat pada gambar 2.2



Gambar 2.2: Struktur Ansible

2.3.1. Terminologi pada Ansible

a. Controller Machine

Mesin dimana Ansible diinstalasi dan bertanggung jawab untuk menjalankan *provisioning* pada *server* yang dikelola.

b. Ansible *Playbook*

Titik masuk untuk Ansible provisioning, dimana automasi didefinisikan melalui tugas (*tasks*) menggunakan format YAML.

c. Task

Blok yang mendefinisikan satu prosedur untuk dieksekusi, sebagai contoh instalasi *package* tertentu.

d. Module

Module merupakan abstraksi dari tugas sistem, seperti berkaitan dengan *package* atau membuat dan mengubah *file*. Ansible memiliki banyak modul *built-in*, namun dapat juga dibuat modul *custom*.

e. Role

Cara yang telah ditentukan sebelumnya untuk mengatur *Playbook* dan *file* lainnya untuk memfasilitasi berbagi pakai dan menggunakan kembali bagian dari *provisioning*.

f. Play

Provisioning yang dieksekusi mulai dari awal sampai akhir disebut dengan *play*. Dengan kata lain, eksekusi dari *Playbook* disebut dengan *play*.

g. Facts

Variable global yang memuat informasi tentang sistem, seperti *interface* jaringan atau sistem operasi.

h. Handlers

Handlers digunakan untuk memicu perubahan status dari *service*, seperti me-restart atau menghentikan *service*.

i. Inventory

Pada Ansible dikenal satu istilah dengan nama “*inventory*”, *inventory* yang dimaksudkan di sini adalah *inventory server* kita. Yaitu sebuah file berisikan daftar *server* yang kita akan konfigurasi menggunakan Ansible. *Server* ini merupakan target *deployment* atau apapun yang kita lakukan dengan Ansible yang ada di komputer.

j. Ad-Hoc Commands

Perintah yang dapat diketik pada terminal untuk melakukan sesuatu yang sangat cepat. *Ad-hoc commands* adalah fitur untuk mulai memahami dasar-dasar tentang apa yang dapat dilakukan oleh Ansible sebelum mempelajari sebuah *Playbook*. Seorang dapat menjalankan satu baris *command* cepat di Ansible tanpa harus menulis *Playbook*.

k. Bahasa pemrograman YAML

Bahasa pemrograman yang digunakan untuk membuat sebuah *Playbook*. YAML (dibaca yamel) adalah sebuah format serialisasi data yang mudah dibaca manusia secara umum yang mengambil konsep bahasa pemrograman C, Perl, Python dan ide lain dari format XML dan format email. Ekstensi file YAML adalah **.yaml** atau **.yml**. Berikut pedoman dalam membuat file yaml:

- YAML tidak menggunakan tab, tetapi menggunakan spasi (indentasi)
- YAML sensitif terhadap huruf besar dan kecil
- Komentar dimulai dengan tanda pagar (#), dapat dimulai di mana saja pada suatu baris dan berlanjut hingga akhir baris.
- Daftar anggota dilambangkan dengan tanda hubung (-) dengan satu anggota per baris.
- Batasan blok dibatasi dengan tanda (|)
- Beberapa dokumen dalam satu aliran dipisahkan oleh tiga tanda hubung (---)

2.3.2. Perbandingan *Tools Automation*

Selain Ansible, Chef, Puppet, dan SaltStack adalah alat untuk Manajemen Konfigurasi, masing-masing menyajikan jalur yang berbeda untuk mencapai tujuan bersama mengelola infrastruktur *server* berskala besar secara efisien. Keempat alat Manajemen Konfigurasi tersebut dirancang untuk mengurangi kerumitan mengkonfigurasi sumber daya infrastruktur terdistribusi. Berikut ini merupakan tabel perbandingan *tools automation* tersebut:

Metrics	Chef	Puppet	Ansible	SaltStack
Availability	✓	✓	✓	✓
Easy of Setup	Not very easy	Not very easy	Easy	Not very easy
Management	Not very easy	Not very easy	Easy	Easy
Scalability	Highly Scalable	Highly Scalable	Highly Scalable	Highly Scalable
Configuration Language	DSL(Ruby)	DSL(Puppet)	YAML(Python)	YAML(Python)
Interoperability	High	High	High	High
Pricing (up to 100 nodes)	\$13700	\$11200-\$19900	\$10,000	\$15,000(approx.)

Tabel 2.1: Perbandingan Tools Automation

Sebagai pendatang baru dibandingkan dengan Puppet, Chef dan SaltStack, Ansible dikembangkan untuk menyederhanakan tugas orkestrasi dan Manajemen Konfigurasi yang rumit. Platform ini ditulis dengan Python dan memungkinkan pengguna untuk melakukan skrip perintah di YAML sebagai paradigma pemrograman *imperatif*. Ansible menawarkan beberapa model *push* untuk mengirim modul perintah ke *node* melalui *ssh* yang dieksekusi secara berurutan. Kelebihan Ansible adalah tidak memerlukan agen tambahan di setiap sistem, dan modul dapat berada di *server* apa pun.

Setiap *tools* ditujukan untuk segmen pengguna yang berbeda dalam target pasar yang sama. Tim DevOps yang berinvestasi dalam solusi Manajemen Konfigurasi harus mempertimbangkan persyaratan khusus di sekitar alur kerja mereka agar tepat sasaran. Untuk memilih solusi Manajemen Konfigurasi yang tepat yang sesuai dengan organisasi, maka perlu pertimbangan *architecture*, *operational model*, *features*, dan *usability and support*, di antara aspek teknis dan bisnis lainnya.

2.4. SSH (*Secure Shell*)

Protokol SSH (*Secure Shell*) adalah metode untuk melakukan login jarak jauh dari satu komputer ke komputer lainnya [14]. SSH memanfaatkan kriptografi untuk melakukan komunikasi data pada perangkat jaringan agar lebih aman. Dalam konsepnya penggunaan SSH ini harus didukung oleh *server* maupun perangkat atau komputer klien yang melakukan pertukaran data. Keduanya harus memiliki SSH *Server* dari sisi komputer *server* dan SSH Client untuk komputer penerima.

Banyak digunakan pada sistem operasi berbasis Linux dan Unix untuk mengakses akun Shell, SSH dirancang sebagai pengganti Telnet dan shell remote tidak aman lainnya, yang mengirim informasi, terutama kata sandi, dalam bentuk teks sederhana yang membuatnya mudah untuk dicegat. Enkripsi yang digunakan oleh SSH menyediakan kerahasiaan dan integritas data melalui jaringan yang tidak aman seperti internet. Ansible sendiri bekerja di koneksi SSH remote ke *host* yang ingin di deploy atau dilakukan automasi tersebut.

2.5. *Server*

Server adalah komputer yang menyediakan jenis layanan tertentu dalam sebuah jaringan komputer [15]. Aplikasi yang disimpan di komputer ini dan terminal komputer lain terhubung, dapat mengaksesnya. Komputer *server* ini memberikan pelayanan bagi sejumlah komputer yang saling berhubungan, dalam melakukan akses data misalnya untuk pembatasan akses, melakukan kontrol data, dan aliran data yang terjadi. *Server* mempunyai banyak fungsi, beberapa diantaranya adalah, menyimpan *database* aplikasi, memberikan keamanan dengan firewall. Kesimpulannya *server* adalah sebuah komputer yang berperan untuk melayani, mengelola, serta mengatur semua keperluan komputer client dalam sebuah jaringan komputer.

Tugas yang dilakukan oleh komputer *server* tergantung pada penggunaan komputer *server*. Fungsi *server* secara umum dilakukan oleh sebuah *server* komputer adalah:

- Menyimpan aplikasi dan database yang dibutuhkan oleh komputer yang terhubung.