JUNIF: Jurnal Nasional Informatika

OPEN ACCES p-ISSN: 2774-3497 e-ISSN: 2774-3500

Vol. 2, No. 2, Oktober 2021, pp. 72-91

AUTOMATION PROVISIONING DEV-OPS WEBSITE SERVER MENGGUNAKAN ANSIBLE DAN VAGRANT

PENULIS **Natalia Evianti, Asep Mulyana Wihandar, Ari Kurniawan**

ABSTRAK

Otomatisasi merupakan satu terobosan baru di dunia IT yaitu dengan konsep untuk mengubah proses yang semula masih manual menjadi otomatis. Pembaharuan konsep ini dikarenakan timbulnya permasalahan bahwa masih sulit dipahami oleh admin jaringan untuk mengkonfigurasi web server dan load balancer. Dari pemanfaatan konsep baru ini diharapkan mengoptimalkan beberapa aspek, dan menjadi solusi baru di bidang IT khususnya pada otomatisasi konfigurasi server. Dari proses-proses dalam langkah konfigurasi diatas akan dibuat menjadi satu proses yang disebut otomatisasi. Konfigurasi yang akan di implementasikan pada penelitian kali ini adalah membuat virtual machine dan konfigurasi web server. Bila dibandingkan dengan cara manual yang dilakukan dengan banyak proses, cara otomatis ini lebih efisien karena dilakukan oleh software atau tools ansible dan vagrant dengan sekali proses. Ansible dan Vagrant merupakan salah satu software kembangan dari DevOps (Development Operations) untuk melakukan proses otomatisasi, selain itu ansible dan vagrant juga dapat melakukan proses instalasi, deployment, dan update server. Secara garis besar, sistem yang dirancang ini akan memanfaatkan ansible untuk konfigurasi load balancer dan web server, sedangkan vagrant dimanfaatkan untuk instalasi server load balancer dan server website.

Kata Kunci Otomatisasi, Provisioning, Web Server, Ansible, Vagrant

AFILIASI

Prodi, Fakultas Teknik Informatika, Fakultas Ilmu Komputer
Nama Institusi Institut Bisnis dan Informatika (IBI) Kosgoro 1957
Alamat Institusi Jl. M. Kahfi II No. 33, Jagakarsa, Jakarta Selatan, DKI Jakarta

KORESPONDENSI

Penulis Natalia Evianti
Email natalia.evianti@gmail.com

diterbitkan oleh:
Program Studi Teknik Informatika
Institut Bisnis dan Informatika (IBI) Kosgoro 1957

I. PENDAHULUAN

Teknologi Informasi dan Komunikasi saat ini sangat berkembang dengan baik dan pesat serta memasuki berbagai bidang dalam kehidupan. Baik dalam bidang teknologi, perusahaan, kesehatan, pendidikan, perbankan, dan media hiburan lainnya.

Dari semua bidang yang ada diatas, salah satunya dibidang teknologi. Tidak jarang satu orang atau administrator server memiliki lebih dari satu server di zaman teknologi seperti sekarang ini, tentunya akan sulit jika administrator tersebut tidak memiliki pengetahuan tentang otomatisasi server. DevOps adalah suatu system pendekatan dari Dev (pengembang) dan Ops (sistem) agar lebih mudah, tepat, dan terkontrol dalam memproduksi atau mengembangkan system dalam skala besar.

Banyak alat DevOps yang mampu mengelola sejumlah besar server saat digunakan. Alat DevOps diperlukan untuk mendukung teknologi ini. Fungsi dan penggunaan DevOps harus menjadi semacam jembatan yang menghubungkan pembuat kode, server, dan penguji. DevOps sangat penting bagi perusahaan pengembangan perangkat lunak yang produknya perlu dirilis secara berkala. Di lingkungan DevOps, ada banyak hal yang perlu diaktifkan secara otomatis sehingga administrator server tidak perlu melakukan tugas berulang.Dengan DevOps, banyak hal yang dapat diaktifkan secara otomatis yang tentunya dapat menghemat waktu.

Berdasarkan uraian di atas, perlu adanya suatu sistem yang dapat mengatasi kendala tersebut, terutama dengan Using Ansible akan banyak membantu admin server biasa dan orang-orang yang memiliki beralih ke DevOps. mengelola server dalam jumlah yang cukup besar, Ansible menyediakan cara untuk membuat manajemen server (bahkan dalam jumlah besar) menjadi efisien. Dengan alat Ansible ini, kami akan menyederhanakan manajemen server secara otomatis, tanpa upaya manual. Jadi secara singkat Ansible adalah suatu jenis dari Configuration Management Tools dan dapat digunakan dalam mengubah proses infrastruktur dari satu program manual menjadi otomatis.

II. PENELITIAN YANG TERKAIT

Penelitian yang terkait menguraikan ulasan penelitan yang pernah dilakukan sebelumnya oleh peneliti lain yg pertama adalah "Otomatisasi Keamanan Pada Router Mikrotik Menggunakan Ansible" oleh Ahmad Givari Adi Prasetyo dan I Putu Hariyadi dari Universitas Bumigora, Indonesia, 2020 [1]. Penelitian ini membahas tentang keamanan pada router yang dilakukan secara manual yang tidak akan efektif jika jumlah router yang membutuhkan keamanan cukup banyak. Untuk itu diperlukan solusi untuk mempercepat proses keamanan dan meminimalisir kesalahan. Mengadopsi sistem otomasi menggunakan Ansible dapat merampingkan proses yang dilakukan secara manual. Ansible adalah mesin otomatisasi TI sederhana yang dapat mengotomatiskan proses manajemen konfigurasi berulang. Fungsi keamanan otomatis pada router Mikrotik adalah membuat, menonaktifkan pengguna demi pengguna, filter firewall, menonaktifkan server mac ping, menonaktifkan server btest, menonaktifkan layanan udara yang diperlukan, menyimpan dan memulai ulang dengan test case pada setiap perangkat yang relevan berdasarkan desain pengujian dan analisis pengujian hasil selesai. Perbedaan dengan penelitian ini adalah ditambahkannya penggunaan Vagrant disamping Ansible untuk dapat memaksimalkan hasil dan juga diarahkan kepada Provisioning Website Server.

Penelitian sejenis yang kedua adalah penelitian yang berjudul "A Review Paper on DevOps: Beginning and More To Know" oleh Pratibha Jha dan Rizwan Khan yang terbit pada jurnal International Journal of Computer Applications, 2018 [2]. Penelitian ini bercerita tentang

ungkapan "DevOps" secara teratur mengacu pada peningkatan pengembangan mahir yang mendukung hubungan kerja berorientasi komunitas antara peningkatan dan aktivitas Teknolgi Informasi, menghasilkan aliran cepat pekerjaan yang diatur (yaitu, tingkat pengiriman yang tinggi) sementara pada saat yang sama memperluas ketergantungan, kemantapan, fleksibilitas, dan keamanan lingkungan kreatif. Mungkin diterima bahwa "DevOps" bergabung dengan tugas kemajuan dan operasional. Namun, DevOps tidak terikat dengan menggabungkan tugas-tugas tradisional ke dalam kelompok tersendiri. Atau mungkin, DevOps adalah pengaturan standar dan teknik pengelolaan yang memajukan kesempurnaan pemrograman sepanjang siklus hidup SDLC-nya. Batas antara disiplin penyampaian pemrograman dipisahkan untuk memberikan perubahan yang konsisten mempercepat waktu untuk beriklan sambil meningkatkan kualitas. Dari penelitian ini diambil kesimpulan bahwa DevOps berhubungan dengan peningkatan Teknologi Informasi sehingga DevOps ini diangkat kedalam penelitian.

III. METODE PENELITIAN

Untuk mempermudah dan memperlancar dalam penyusunan penelitian ini, digunakan beberapa metode diantaranya :

1. Wawancara
   Untuk mendapatkan informasi secara lengkap maka penulis melakukan metode tanya jawab dengan bagian Direktur untuk mendapatkan data.
2. Tahapan-tahapan Metode Studi Literatur
   Dalam studi kepustakaan ini dilakukan proses pemilihan suatu masalah yang akan dijadikan penelitian, kemudian dilanjutkan dengan pencarian referensi untuk mendasari dan mendukung pekerjaan dan pemecahan masalah. Dari kajian bibliografi ini agar rumusan dan batasan permasalahan yang dihadapi menjadi lebih jelas.
3. Disain Sistem
   Pada tahap desain, merupakan tahap perancangan sistem yang akan dirancang nantinya. Kemudian tujuan dari pendisain sistem tersebut akan dirancang sehingga nantinya dapat di implementasikan secara baik dan benar.
4. Implementasi
   Setelah tahap disain selesai dilanjutkan ke tahap implementasi, tahap ini dibagi menjadi 3, yaitu :
   a. Perancangan
      Pada proses ini merupakan bentuk persiapan perancangan dengan mulai melakukan kegiatan pengguna perangkat keras dan perangkat lunak yang digunakan.
   b. Instalasi
      Pada tahap ini dilakukan penginstalan software pada server. Dalam hal ini software yang digunakan pada server adalah Ansible.
   c. Pengujian
      Proses ini bertujuan untuk mengetahui apakah sistem yang telah dirancang mampu berjalan dengan baik dan benar atau tidak.

IV. LANDASAN TEORI

A. Ansible

Ansible adalah perangkat lunak atau perangkat lunak komputer yang dapat membantu DevOps atau administrator sistem dengan otomatisasi server. Ansible dapat membantu menginstal, menyebarkan, dan bahkan memperbarui server. Ansible juga dapat terhubung ke server seperti LDAP dan Kerberos dan mengelola apa pun yang ada di dalamnya, jadi Ansible biasanya berfungsi seperti alat lain seperti Chef, Puppet, hanya saja Ansible tidak memerlukan agen yang hanya berfungsi dengan koneksi SSH [3].
Istilah Dasar yang digunakan dalam Ansible [4]:

1. Control Node
   Semua mesin yang digunakan untuk menjalankan Ansible. Kamu bisa menggunakan laptop / komputer ataupun server sebagai control node, selama ada Python yang terinstall di mesin tersebut. Tetapi yang perlu diingat bahwa kamu tidak bisa menggunakan Windows sebagai control node.
2. Managed Node
   Perangkat jaringan atau server yang dikelola menggunakan Ansible, bisa juga disebut sebagai “hosts”.
3. Inventory
   List hosts / managed nodes. Semua hosts dimasukkan ke dalam sebuah file yang biasa disebut sebagai inventory file / hostfile. Isinya bisa berupa IP atau hostname server. Di dalam inventory kamu juga bisa mengelompokkan beberapa hosts sesuai dengan fungsi ataupun lokasi.
4. Task
   Semua action yang dijalankan Ansible di dalam hosts. Kamu dapat menjalankan satu tasks saja dengan menggunakan ad-hoc command atau menjalankan beberapa tasks langsung dengan menggunakan playbook.
5. Playbooks
   List tasks yang dijalankan berurutan. Dengan menggunakan playbook kamu bisa menjalankan task berulang kali. Playbooks dibuat dengan menggunakan YAML sehingga mudah untuk dibuat dan dipelajari.
6. Modules
   Ketika Ansible menjalankan tasks, Ansible sebenarnya mengeksekusi sebuah script sesuai dengan task yang dijalankan. Contoh ketika Ansible menjalankan task untuk copy file, maka Ansible akan mengeksekusi script yang akan melakukan copy file dari control node ke dalam hosts. Script inilah yang disebut sebagai module.

Ansible bekerja melalui koneksi SSH jarak jauh ke klien yang ingin Anda terapkan atau otomatisasi. Ansible juga membutuhkan data inventaris atau data server tujuan. Pada tingkat yang lebih tinggi, ansible juga dapat memainkan peran seperti buku dan peran. Konfigurasi ditulis dalam format markup YAML , mungkin karena YAML sangat mudah dibaca manusia. Jadi nanti bisa jadi dokumen tersendiri (Infra as code).

B. Provisioning

Provisioning adalah menyediakan layanan atau konfigurasi kepada pengguna, termasuk semua yang diperlukan untuk mengatur layanan, seperti paket software, peralatan, perkabelan dan transmisi.

C. Configuration Management

Manajemen konfigurasi jaringan adalah disiplin TI yang berhubungan dengan pencadangan konfigurasi jaringan, otomatisasi jaringan, perubahan konfigurasi, pemulihan, dan pemeriksaan kesalahan. Blog ini adalah tentang manajemen konfigurasi jaringan, bagaimana membantu jaringan perusahaan dari semua ukuran. administrator kapan pun mereka mengonfigurasi perangkat lunak pemantauan jaringan (NMS) untuk digunakan di lingkungan produksi mereka Menurut laporan gartner terbaru tentang Manajemen Konfigurasi Jaringan untuk infrastruktur virtual dan cloud cloud Sekitar 80% pemadaman jaringan perusahaan disebabkan oleh kesalahan manusia manual, sementara hanya sebagian kecil Persentase pemadaman disebabkan oleh kegagalan peralatan, seperti kabel yang rusak.

Manajemen konfigurasi Bentuk jaringan sering dipertimbangkan oleh jaringan.Ada alasan praktis untuk mendukungnya. Seluruh jaringan seringkali membutuhkan banyak perencanaan dan tidak terukur. Menyebarkan manajemen kesalahan dan sistem manajemen kinerja, seperti NMS, memberi administrator TI wawasan langsung tentang kinerja dan ketersediaan jaringan. , pengguna juga dapat mengukur dampak positif, mengelola konfigurasi jaringan yang dibuat setelah digunakan.

D. Aplication Deployement

Aplication Deployment adalah semua aktfitas yang membuat sebuah sistem perangkat lunak dapat digunakan. Proses penyebaran tipikal terdiri dari beberapa aktivitas terkait dengan kemungkinan modifikasi di antaranya. Kegiatan ini bisa di sisi produsen, di sisi konsumen, atau keduanya. Karena setiap sistem perangkat lunak adalah unik, proses atau prosedur yang tepat antara setiap operasi sulit untuk ditentukan. Oleh karena itu, "penyebaran" harus dipahami sebagai proses umum yang harus berkembang untuk kebutuhan khusus atau karakteristik khusus.

E. Manajemen Keamanan

Penting untuk menerapkan manajemen keamanan agar informasi yang mengalir melalui bisnis dapat dikelola dengan baik sehingga bisnis dapat mengambil keputusan yang tepat berdasarkan informasi yang benar untuk memberikan layanan terbaik kepada pelanggan. Pelanggan terdiri dari empat tahap :

1. Identifikasi threats (ancaman) yang dapat menyerang sumber daya informasi perusahaan.
2. Mendefinisikan resiko dari ancaman yang dapat memaksakan.
3. Penetapan kebijakan keamanan informasi.
4. Menerapkan controls yang tertuju pada resiko.

Istilah manajemen risiko diciptakan untuk menggambarkan pendekatan ini, di mana keamanan sumber daya informasi organisasi dibandingkan dengan risiko yang dihadapinya.

Benchmark adalah tingkat kinerja yang direkomendasikan. Standar keamanan informasi adalah tingkat keamanan yang direkomendasikan. yang, dalam keadaan normal, harus

memberikan perlindungan yang memadai terhadap entri yang tidak sah. Standar atau tolok ukur ini ditetapkan oleh pemerintah dan asosiasi industri dan mencerminkan komponen program keamanan informasi yang kuat menurut lembaga ini.

Ketika perusahaan mengambil pendekatan ini, yang dikenal sebagai standar kepatuhan, dapat diasumsikan bahwa pemerintah dan otoritas industri telah mempertimbangkan ancaman dan risiko dengan baik.bervariasi dan standar ini memberikan ukuran perlindungan yang baik.

F. Network Orchestration

Network Orchestration adalah lapisan manajemen dan kontrol yang memusatkan tugas manajemen khusus virtualisasi yang diperlukan di seluruh siklus hidup Fungsi Jaringan Virtual (VNF).

G. HAProxy

HAProxy adalah proyek sumber terbuka yang dilisensikan di bawah GPLv2. Haproxy atau High Av available Proxy adalah penyeimbang beban TCP/IP dan server proxy yang dapat memuat-berbagi permintaan masuk dengan server multi-simpul. Dengan demikian, beban server akan dibagi di antara node server yang ada. Ada cara yang berbeda, ada yang sama, ada yang berdasarkan jumlah trafik dan ada juga yang berbeda. Kemudian kita akan mempelajari algoritma pembagian beban request.

Jadi haproxy ini merupakan perantara antara client/user dan server (web/database/dll) sehingga disebut juga haproxy reverse proxy.

H. Vagant

Vagrant adalah sebuah software yang menggunakan teknologi virtual machine dimana kita dapat membuat lingkungan development secara portable, konsisten dan lebih fleksible [6].

I. Automation

Automation menggambarkan berbagai macam teknologi yang mengurangi campur tangan manusia dalam proses. Intervensi manusia dikurangi dengan menentukan kriteria keputusan, hubungan subproses, dan tindakan terkait - dan mewujudkan predeterminasi tersebut dalam mesin.

Automation, mencakup penggunaan berbagai sistem kontrol untuk peralatan operasi seperti mesin, proses di pabrik, boiler, dan oven pengolah panas, penyalaan jaringan telepon, pengemudian, dan stabilisasi kapal, pesawat, serta aplikasi dan kendaraan lain. dengan intervensi manusia berkurang.

J. Web Server

Web Server adalah sebuah software yang berfungsi untuk menerima dan melayani permintaan yang dikirim user melalui browser kemudian ditampilkan kepada user sesuai dengan permintaan yang dikirim ke server.

V. ANALISIS DAN PERANCANGAN

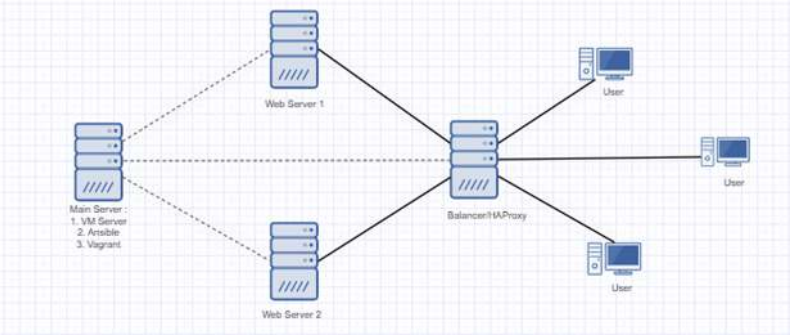
A. Analisis Kebutuhan Sistem

Analisis yang dilakukan berdasarkan pengumpulan data yang diperoleh, terdapat kebutuhan otomasi yaitu disimpulkan sebagai berikut:

1. Otomatisasi untuk konfigurasi pada beberapa server agar waktu yang digunakan lebih efisien.
2. Otomatisasi update patch program agar mengurangi kesalahan pada saat proses update dan lebih hemat waktu.
3. Otomatisasi untuk membuat Virtual Machine baru agar tidak terjadi kesalahan persyaratan program yang server gunakan.

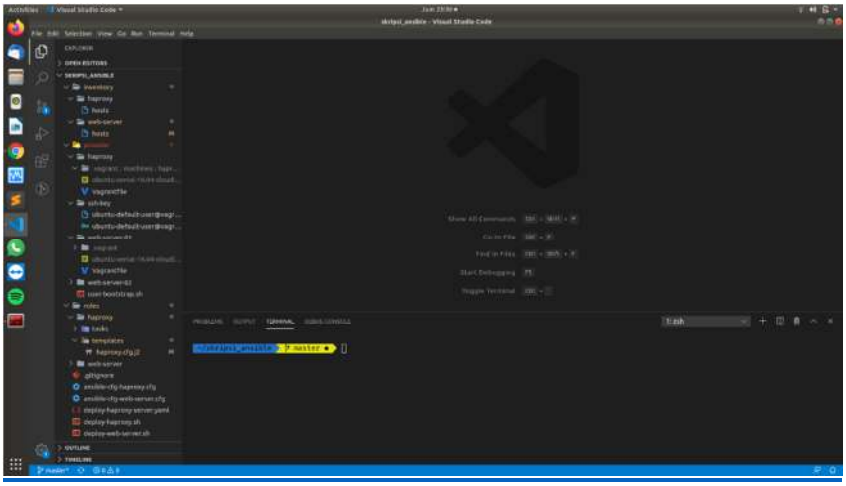
B. Perancangan

1. Topologi Jaringan



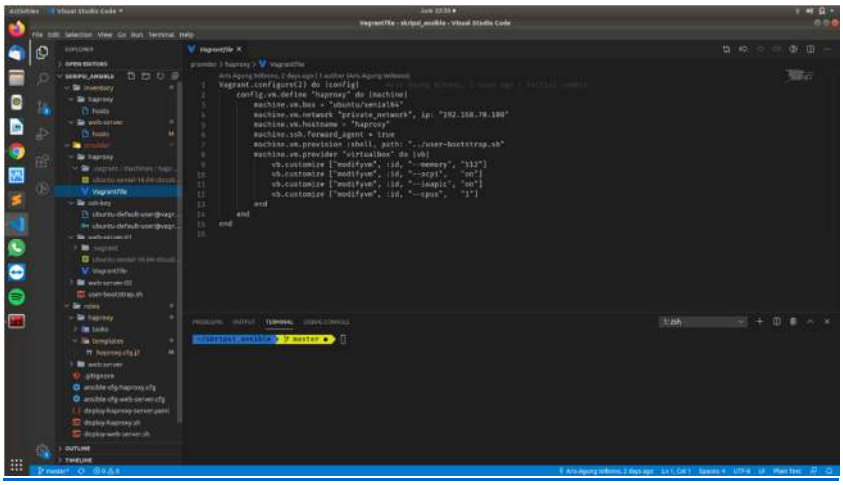
Gambar 1. Topologi Jaringan

2. Perancangan Alur Direktori File



Gambar 2. Perancangan Alur Direktori File

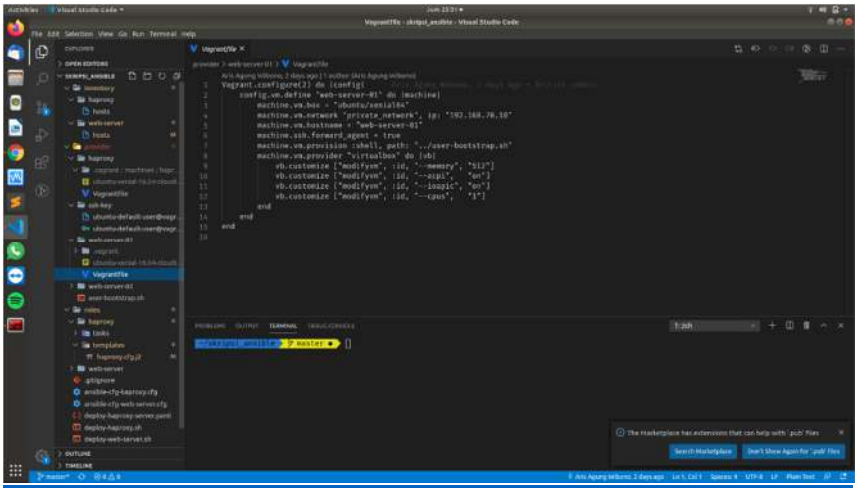
3. Script Vagrant HAProxy



Gambar 3. Script Vagrant HAProxy

Script ini berfungsi untuk membuat serta konfigurasi secara otomatis Virtual Machine Load Balancer HAProxy [5].

4. Script Vagrant Webserver



Gambar 4. Script Vagrant Web Server

Script ini berfungsi untuk membuat konfigurasi secara otomatis Virtual Machine web server1 dan web server2.

VI. HASIL DAN PEMBAHASAN

A. Implementasi

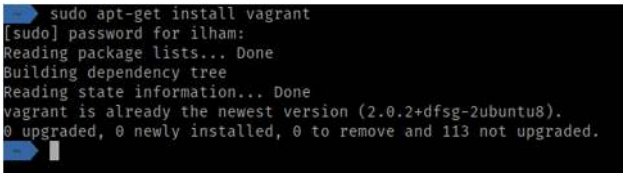
Automation Ansible ini sangat cocok di implementasikan bagi sysadmin yang ingin membuat server yang sama di beda tempat. Untuk membuat atau menginstall server, hanya perlu menjalankan script vagrant. Dan untuk konfigurasi sysadmin hanya perlu menjalankan

script deploy-haproxy.sh dan deploy-web-server.sh maka server tersebut akan sama persis, ini sangat membantu sysadmin dalam menghemat waktu, dan jika terjadi satu kesalahan sysadmin hanya perlu memperbaikinya di satu server saja dan untuk server lainnya hanya perlu menjalankan script script deploy-haproxy.sh dan deploy-web-server.sh kembali.

B. Implentasi Tools

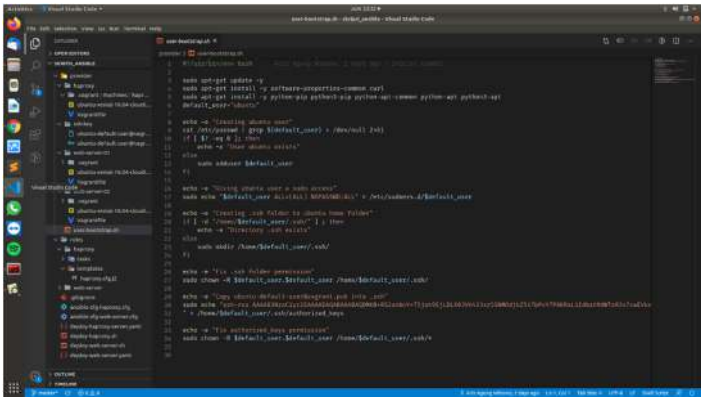
1. Instalasi Vagrant

Cara Instalasi vagrant adalah sebagai berikut [7]:
sudo apt-get install vagrant



Gambar 5. Instalasi Vagrant

Setelah Vagrant terinstall, lalu membuat bash script untuk konfigurasi dan instalasi paket yang dibutuhkan server.
Setelah membuat script konfigurasi, penulis akan membuat folder project vagrant pada directory provider.
mkdir web-server-01



Gambar 6. Script Konfigurasi Server

Setelah folder terbuat, langkah selanjutnya adalah masuk kefolder tersebut.
cd web-server-01

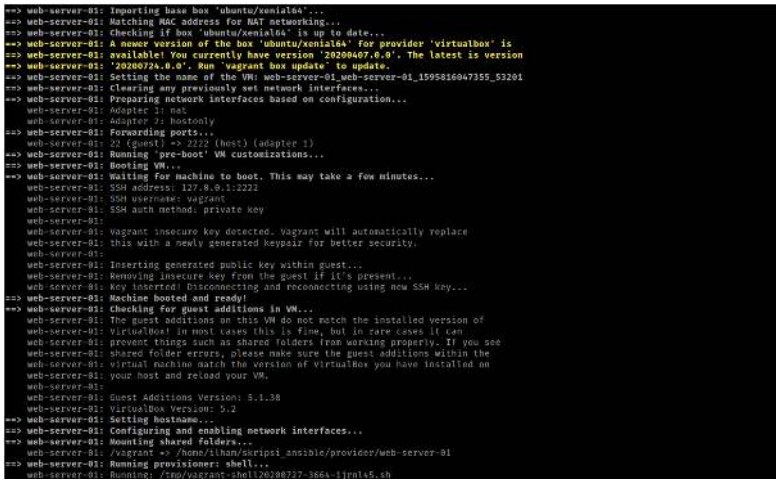
Gambar 7. folder web-server-01

Buka dan edit file vagrant, pada konfigurasi ini penulis menggunakan virtual box sebagai backend provider untuk membuat virtual machine

```
Vagrant.configure(2) do |config|
    config.vm.define "web-server-01" do |machine|
        machine.vm.box = "ubuntu/xenial64"
        machine.vm.network "private_network", ip: "192.168.70.10"
        machine.vm.hostname = "web-server-01"
        machine.ssh.forward_agent = true
        machine.vm.provision :shell, path: "../user-bootstrap.sh"
        machine.vm.provider "virtualbox" do |vb|
            vb.customize ["modifyvm", :id, "--memory", "512"]
            vb.customize ["modifyvm", :id, "--acpi",   "on"]
            vb.customize ["modifyvm", :id, "--ioapic", "on"]
            vb.customize ["modifyvm", :id, "--cpus",   "1"]
        end
    end
end

```

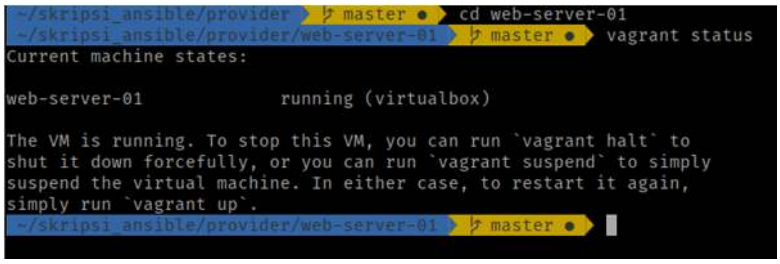
Gambar 8. Vagrant file web server 1

Setelah konfigurasi file vagrant web server 1
Lalu jalankan vagrant dengan cara menulis perintah.
vagrant up



Gambar 9. Proses Pembuatan VM Web Server1 oleh Vargant

Pada proses ini vagrant akan membuat satu virtual machine web server 1 dengan operasi system Ubuntu 16.04 dan menjelaskan seluruh perintah yang ada pada script konfigurasi awal virtual machine. Kemudian penulis akan memastikan bahwa virtual machine sudah berjalan dengan menggunakan perintah sebagi berikut :
vagrant status



Gambar 10. Melihat Status VM Web Server 1

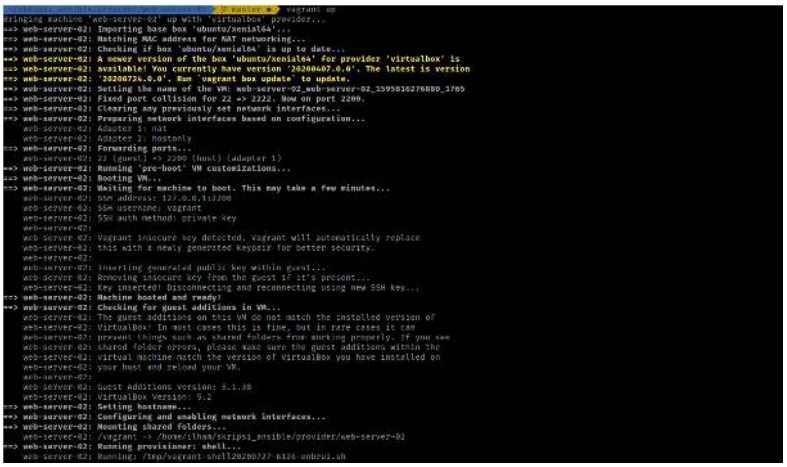
Setelah VM web server 1 terbuat, selanjutnya membuat folder project vagrant untuk web server 2 pada directory provider.
mkdir web-server-02
Setelah folder terbuat, langkah selanjutnya adalah masuk ke folder tersebut.
cd web-server-02

Gambar 11. Folder web-server-02

Buka dan edit file vagrant, pada konfigurasi ini penulis menggunakan virtual box sebagi backend provider untuk membuat virtual machine.

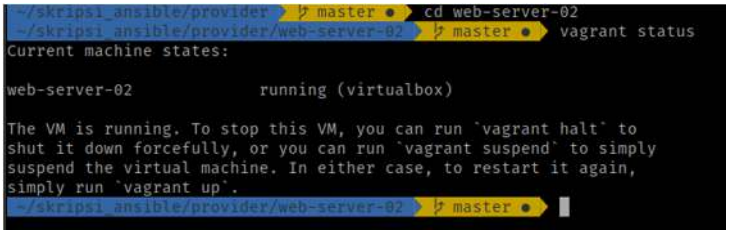
```
Vagrant.configure(2) do |config|
    config.vm.define "web-server-02" do |machine|
        machine.vm.box = "ubuntu/xenial64"
        machine.vm.network "private_network", ip: "192.168.70.20"
        machine.vm.hostname = "web-server-02"
        machine.ssh.forward_agent = true
        machine.vm.provision :shell, path: "../user-bootstrap.sh"
        machine.vm.provider "virtualbox" do |vb|
            vb.customize ["modifyvm", :id, "--memory", "512"]
            vb.customize ["modifyvm", :id, "--acpi",   "on"]
            vb.customize ["modifyvm", :id, "--ioapic", "on"]
            vb.customize ["modifyvm", :id, "--cpus",   "1"]
        end
    end
end

```
Gambar 12. Vagrant file web server 2

Setelah konfigurasi file vagrant web server 2, lalu jalankan vagrant dengan cara menulis perintah.
vagrant up


Gambar 13. Proses Pembuatan VM Web Server 2 Oleh Vagrant

Pada proses ini vagrant akan membuat satu virtual machine web server 1 dengan operasi system Ubuntu 16.04 dan menjalankan seluruh perintah yang ada pada script konfigurasi

awal virtual machine. Kemudian penulis akan memastikan bahwa virtual machine sudah berjalan dengan menggunakan perintah sebagai berikut :
vagrant status



Gambar 14. Melihat Status VM Web Server 2

Setelah VM web server 1 dan web server 2 terbuat, selanjutanya membuat folder project vagrant untuk haproxy pada directory provider.
mkdir haproxy
Setalah folder terbuat, langkah selanjutnya adalah masuk ke folder terebut
cd haproxy

Gambar 15. Folder Haproxy

Buka dan edit file vagrant, pada konfigurasi ini penulis menggunakan virtual box sebagai backend provider untuk membuat virtual machine.

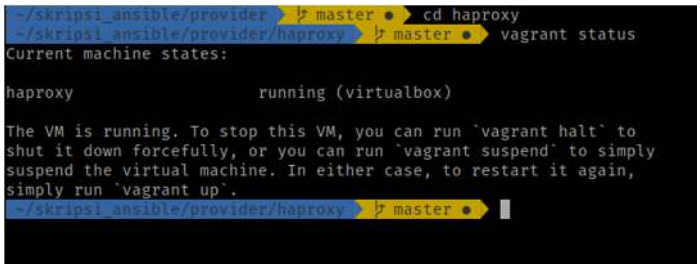
```
Vagrant.configure(2) do |config|
    config.vm.define "haproxy" do |machine|
        machine.vm.box = "ubuntu/xenial64"
        machine.vm.network "private_network", ip: "192.168.70.100"
        machine.vm.hostname = "haproxy"
        machine.ssh.forward_agent = true
        machine.vm.provision :shell, path: "../user-bootstrap.sh"
        machine.vm.provider "virtualbox" do |vb|
            vb.customize ["modifyvm", :id, "--memory", "512"]
            vb.customize ["modifyvm", :id, "--acpi",   "on"]
            vb.customize ["modifyvm", :id, "--ioapic", "on"]
            vb.customize ["modifyvm", :id, "--cpus",   "1"]
        end
    end
end

```

Gambar 16. Vagrant File Haproxy

Setelah konfigurasi file vagrant, lalu jalankan vagrant dengacara menulis perintah
vagrant up

Gambar 17. Proses Pembuatan VM Haproxy Oleh Vagrant

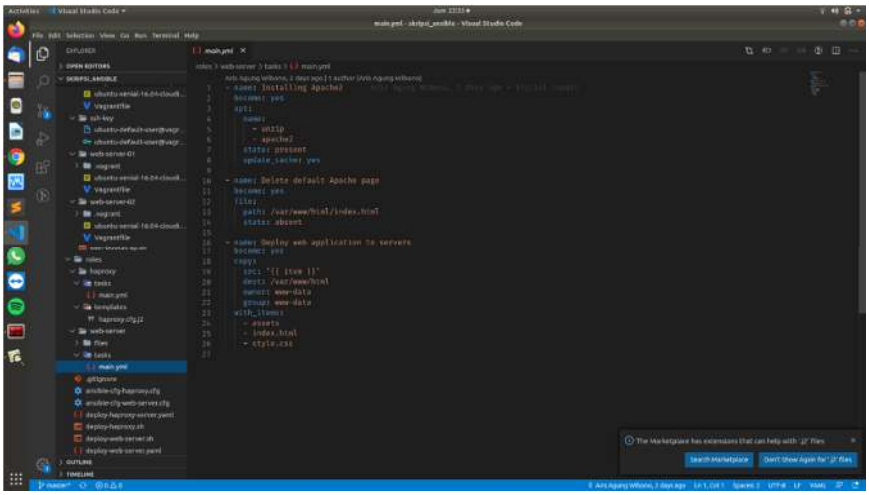
Pada proses ini vagrant akan membuat satu virtual machine web server 1 dengan operasi system ubuntu 16.04 dan menjalankan seluruh perintah yang ada pada script konfigurasi awal virtual machine. Kemudian penulis akan memastika bahwa virtual machine sudah berjalan dengan menggunakan perintah sebagai berikut :
vagrant status



Gambar 18. Melihat Status VM haproxy

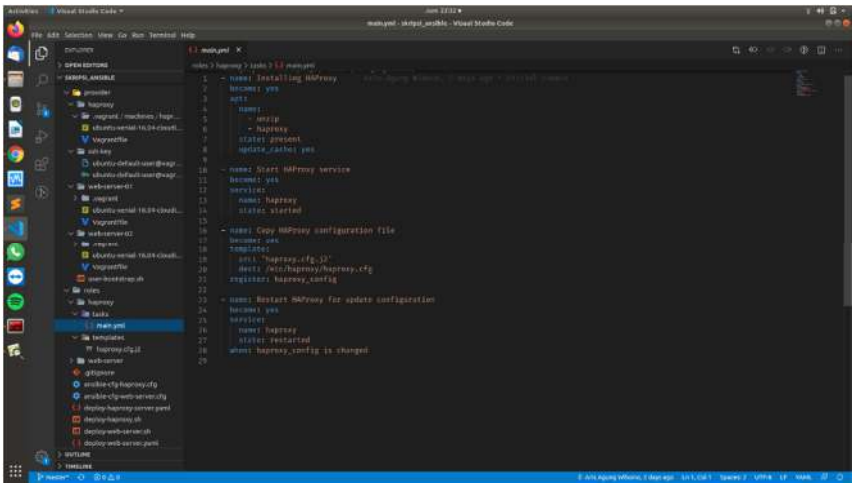
2. Instalasi Ansible

Cara instalasi ansible adalah sebagai berikut :
sudo apt-add-repository ppa:ansible/ansible
sudo apt-get update
sudo apt-get install ansible
Setelah ansible terinstall, penulis akan membuat roles untuk melakukan konfigurasi web server.
Berikut yaml script pada roles untuk installasi dan penyiapan web server.



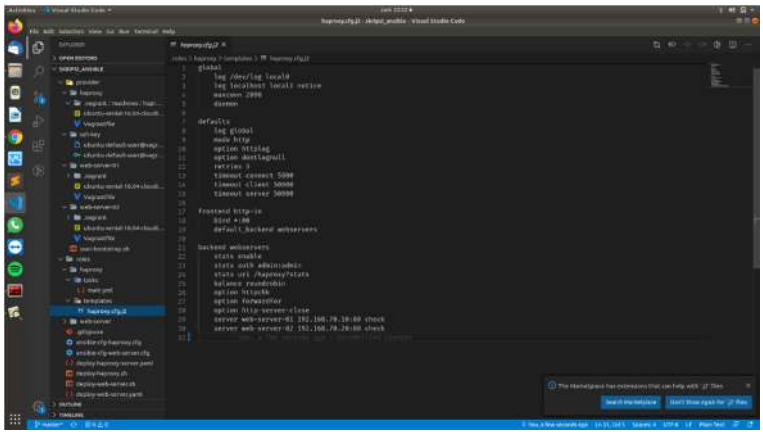
Gambar 19. Script Roles Untuk Web Server

Berikut yaml script pada roles untuk installasi dan penyiapan HAProxy.



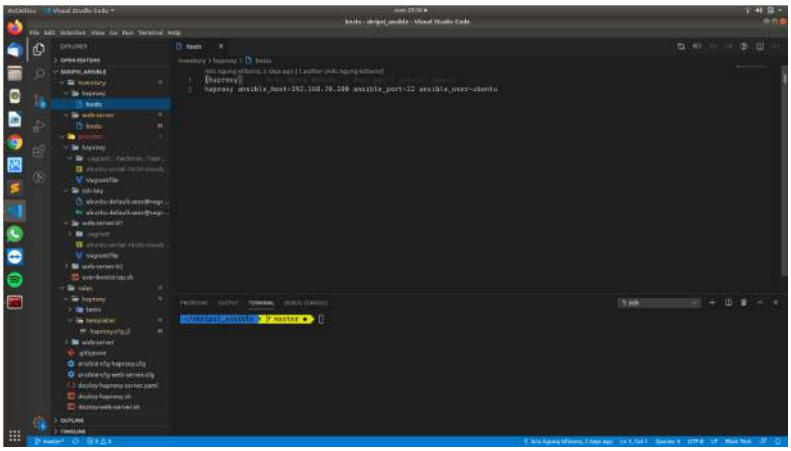
Gambar 20. Script Roles Untuk Haproxy

Pada roles haproxy tersedia template konfigurasi haproxy yang akan di deploy pada server haproxy.
Berikut konfigurasinya :



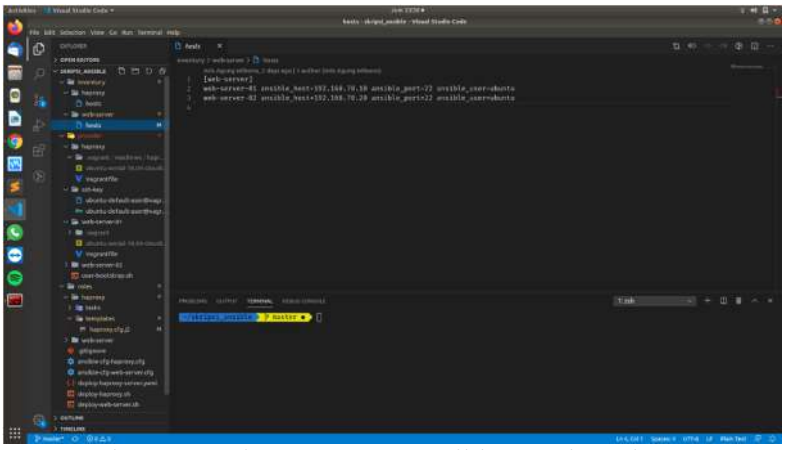
Gambar 21. Script Konfigurasi Haproxy

Setelah membuat roles penulis akan membuat inventory ansible untuk mendaftarkan server agar bisa di index oleh ansible playbook, ketika playbook dijalankan.
Berikut konfigurasi inventory haproxy



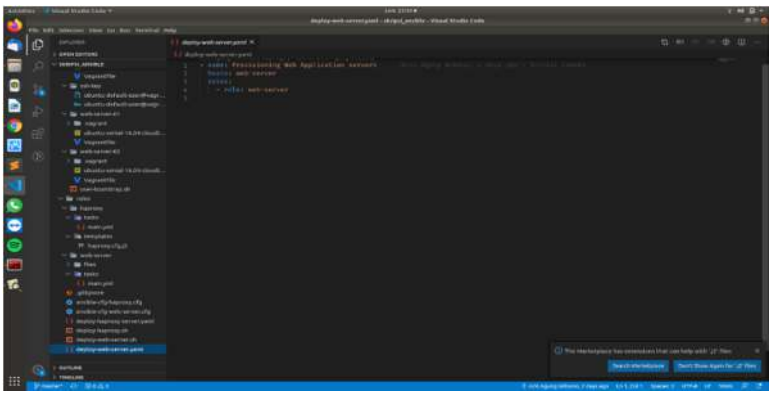
Gambar 22. Script Inventory Ansible Untuk Haprox

Berikut konfigurasi inventory web server



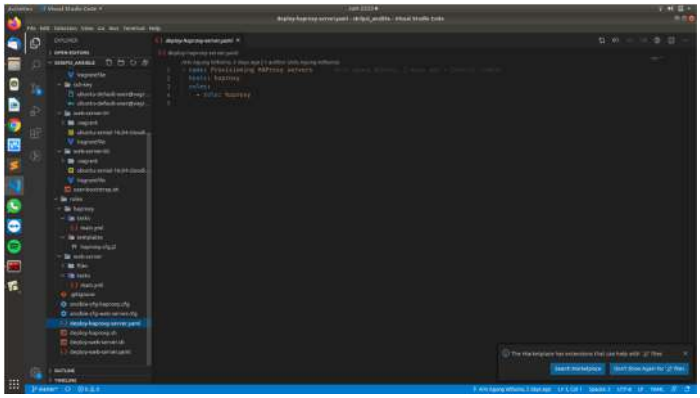
Gambar 23. Script Inventory Ansible Untuk Web Server

Setelah membut inventory, penulis akan membuat playbook yang akan digunakan untuk deployment pada server-server yang sudah disiapkan oleh vagrant.
Berikut playbook untuk melakukan deployment pada web server.



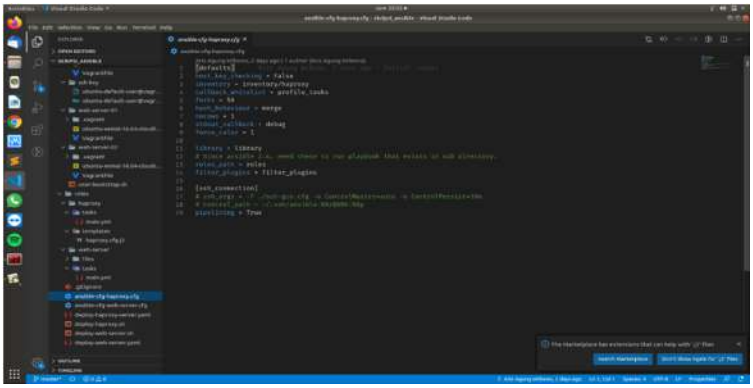
Gambar 24. Playbook Ansible Web Server

Berikut playbook untuk melakukan deployment pada haproxy.



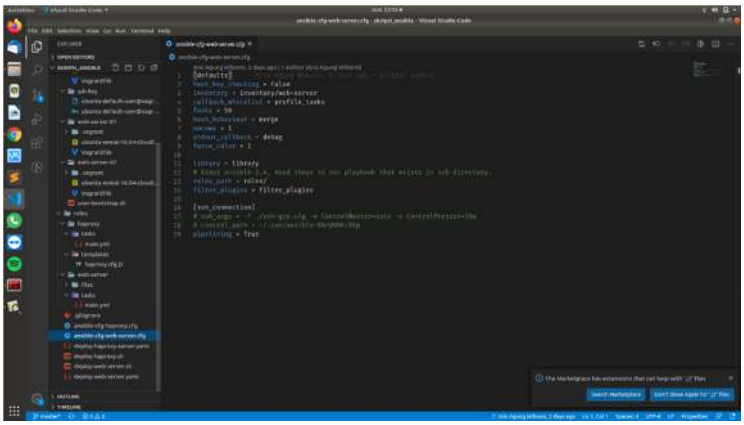
Gambar 25. Playbook Ansible Haproxy

Setelah membuat playbook penulis akan menyiapkan konfigurasi yang digunakan untuk melakukan indexing pada inventory yang sudah disiapkan.
Berikut konfigurasi ansible untuk haproxy.



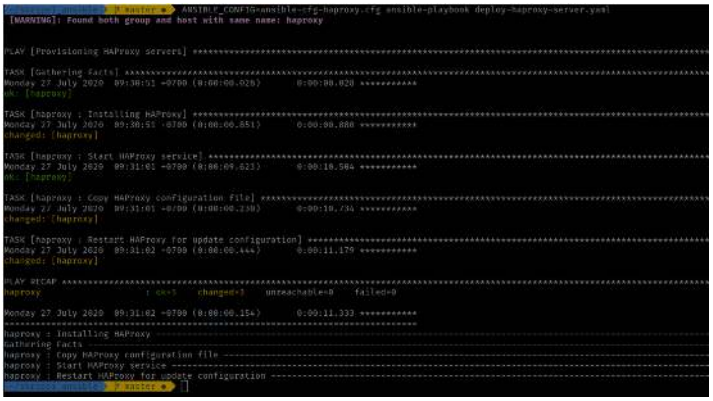
Gambar 26. Konfigurasi Ansible Haproxy

Berikut konfigurasi ansible untuk web server.



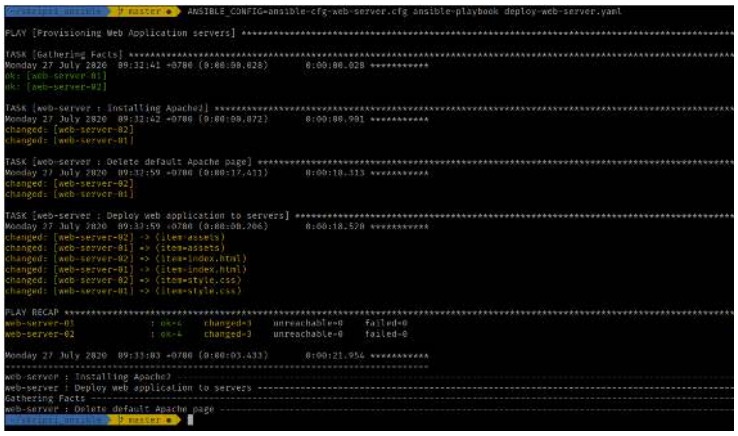
Gambar 27. Konfigurasi Ansible Web Server

Setelah itu penulis akan menjalankan playbook untuk melakukan deployment haproxy dengan perintah
ANSIBLE_CONFIG=ansible-cfg-haproxy.cfg ansible-playbook deploy-haproxy-server.yaml



Gambar 28. Deployment HAProxy Dengan Ansible

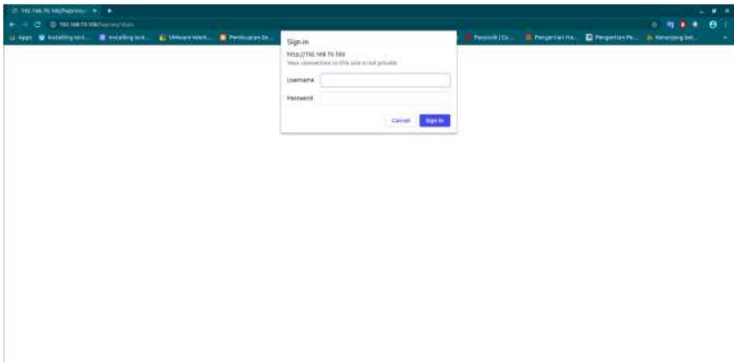
Setelah menjalankan playbook untuk melakukan deployment haproxy, penulis akan menjalankan Kembali playbook untuk deployment web server dengan perintah sebagai berikut.
#ANSIBLE_CONFIG=ansible-cfg-web-server.cfg ansible-playbook deploy-web-server.yaml



Gambar 29. Deployment Web Server 1 Dan 2 Dengan Ansible

C. Implementasi Pengujian

Pengujian sistem akan dilaksanakan adalah tampilan login ke dashboard haproxy yang bisa berfungsi sebagai monitoring server sekaligus menjadi load balancer.



Gambar 30. Login Haproxy

Ketika selesai login, akan beralih ke tampilan dashboard monitoring haproxy.

Gambar 31. Dashboard Haproxy

Di dalam dashboard haproxy terlihat list server yang sudah di koneksikan dengan haproxy, seperti pada Gambar 4.26 Dashboard Haproxy nama web server 1 dan web server 2 masih berwarna merah, yaitu berarti web server sedang off. Ketika ke dua web server di hidupkan warna kolom web server 1 dan web server 2 akan berubah menjadi warna hijau. Seperti gambar di bawah ini.

Gambar 32. Menghidupkan Web Server 1 Dan Web Server 2

Setelah ke dua web server hidup, web site akan bisa di akses melalu IP server load balancer atau haproxy. Seperti gambar berikut.

Automation Ansible ini sangat cocok di implementasikan bagi sysadmin yang ingin membuat server yang sama di beda tempat. Untuk membuat atau menginstall server, hanya perlu menjalankan script vagrant. Dan untuk konfigurasi sysadmin hanya perlu menjalankan script deploy-haproxy.sh dan deploy-web-server.sh maka server tersebut akan sama persis, ini sangat membantu sysadmin dalam menghemat waktu, dan jika terjadi satu kesalahan sysadmin hanya perlu memperbaikinya di satu server saja dan untuk server lainnya hanya perlu menjalankan script script deploy-haproxy.sh dan deploy-web-server.sh kembali.

VII. KESIMPULAN

Kesimpulan yang dapat diambil dari penelitian ini adalah sebagai berikut :

1. Otomatisasi yang dilakukan dengan memanfaatkan software tool ansible dan vagrant dapat digunakan untuk menginstal virtual machine sekaligus mengkonfigurasi load balancer dan web server agar lebih efektif dan efisien dalam waktu pengerjaannya.
2. Dengan menggunakan otomatisasi dari ansible dan vagrant system administrator dapat memanfaatkan waktu kerjanya secara maksimal dan lebih hemat waktu.
3. Dengan menggunakan ansible system administrator tidak perlu melakukan penabahan server virtual secara manual. Karena hanya dengan menjalankan playbook ansible secara otomatis server akan terinstall dan terkonfigurasi sesuai dengan apa yang system administrator perintahkan pada playbook ansible.

DAFTAR PUSTAKA

Journal Article

[1] (Givari et al., 2020)Givari, A., Prasetyo, A., & Hariyadi, I. P. (2020). *Otomatisasi KeamananPada Router Mikrotik Menggunakan Ansible*.

[2] Jha, P., & Khan, R. (2018). A Review Paper on DevOps: Beginning and More To Know. *International Journal of Computer Applications*, *180*(48), 16–20. https://doi.org/10.5120/ijca2018917253

Electronic Publication, Information from the internet

[3] Nostra Tech (2018). Implementasi Ansible Playbook, 02(01). http://blog.nostratech.com/2018/06/implementasi-ansible-playbook.html

[4] Ridwan Fazar (2017). Mengunakan Ansible Untuk Mengelola Server yang Lebih Baik. https://www.codepolitan.com/menggunakan-ansible-untuk-mengelola-server-yang-lebih-mudah-58e8e909d3861 .

[5] Ridwan Fazar (2017). Instalasi dan Konfigurasi Dasar VagrantBox. https://www.codepolitan.com/instalasi-dan-konfigurasi-dasar-vagrantbox-58e8dea0433be.
[6] Rizki Munfrizal (2016). Belajar Vagrant. https://rizkimufrizal.github.io/belajar-vagrant/
[7] Taufik Mulyana (2019). Cara Instal Ansible. https://nothinux.id/cara-install-ansible/.