

Create Autoscaling VM's – on GCP

Use case :

Seorang system admin ingin aplikasinya selalu available dan juga bisa ter-scale saat tingkat request pada server bertambah, dalam arti akan dibuatkan sebuah cluster VM's yang mampu memberikan layanan dimana saat banyak yang mengakses server/aplikasi maka otomatis jumlah server(VM) yang menjalankan aplikasi tersebut bisa bertambah dengan sendirinya, serta jika jumlah request berkurang maka server(VM) akan berkurang juga. Cluster ini akan dihubungkan dengan sebuah load balancer yang berfungsi sebagai forwarding serta menyediakan expose IP untuk digunakan untuk domain.

Prerequisite:

Memiliki google container Registry, yang berfungsi sebagai private registry

Flow:



Step by Step:

- Buka <https://console.cloud.google.com/>
- Sekarang kita akan membuat Custom Image yang telah terinstall Docker + Google SDK, yang berfungsi sebagai default image sehingga tidak diperlukan lagi start script untuk menginstall Docker dan Google SDK. Hal ini juga bertujuan untuk mempercepat time to UP aplikasi nantinya.
- Klik Compute Engine → VM Instances → Create Instance → buat sesuai dengan requirement Server anda → SSH dan install Docker services + Google SDK → setelah terinstall klik vm dan Edit → pada Boot Disk , centang Deletion rule, hal ini untuk mencegah image dari VM terhapus saat VM di destroy → Delete VM
- Klik Compute Engine → Images → Create an image → isi nama dan pada bagian Source pilih Disk → Source Disk, cari nama VM yang telah diDelete tadi → Pilih Location sesuai kebutuhan anda → Family, Desc, Label adalah opsional sesuai kebutuhan → Create
- Sekarang kita akan membuat instance templates, hal ini bertujuan agar instance group nantinya memiliki instance template yang sama.
- Klik Compute Engine → Instance templates → Create Instance template → isi nama, tipe machine → pada bagian Boot disk change pilih custom images dan pilih images yang telah dicustom tadi → tambahkan firewall rules, serta jika ada tambahan di [Management, security, disks, networking, sole tenancy](#) → Create

Boot disk

Select an image to create a boot disk. The image determines the operating system installed solutions in [Marketplace](#).

Public images Custom images

Show images from cloud-image-ubuntu18

☐ Show deprecated images

Image cloud-image-ubuntu18

Created on Feb 6, 2020, 3:25:49 PM

Boot disk type ? Size (GB) ?

Standard persistent disk 10

- Setelah kita memiliki instance template, sekarang kita akan membuat instance group.
- Klik Compute engine → Instance groups → Create an instance group → isi nama, deskripsi, location → pada bagian Instance template, pilih instance template yang telah dibuat sebelumnya → Autoscaling mode pilih Autoscale → Autoscaling metrics pilih default (CPU utilization: 60%) untuk mengetahui lebih lanjut mengenai policy autoscaling dapat dibaca di <https://cloud.google.com/compute/docs/autoscaler> → cool down period = 60 s → kemudian isi number minimum instances 1 dan max 3 (sesuai kebutuhan anda), bagian ini bertujuan untuk membatasi jumlah VM maksimal yang dapat di create saat trafik telah mencukupi berdasarkan metrik → Autohealing, create dengan protocol HTTP:80 → Create

← Instance groups ROLLING UPDATE ROLLING RESTART/REPLACE DELETE GROUP REMOVE FROM GROUP

cloud-grp-vm01

Members Details Monitoring Errors

Zone: asia-southeast1 (3/3 zones) Template: backend-vm-template Autoscaling: On Target: CPU utilization 60% In use by: backend-lb

Filter group members Columns

<input type="checkbox"/>	Name	Creation time	Zone	Health check status	External IP	Connect
<input checked="" type="checkbox"/>	cloud-grp-vm01-c1c6	Feb 6, 2020, 3:35:33 PM	asia-southeast1-a	HEALTHY	35.234.12.12	SSH

- Setelah instance group telah selesai di setup, kita butuh loadbalancer sebagai wadah untuk menghubungkan user kepada vm-vm yang ada di instance group.
- Klik network services → Load balancing → create load balancer → HTTP(S) Load Balancing → Start Configuration → From internet to my Vms → Isi nama → Backend Configuration, Backend Services, create a backend services → isi nama → new backend, pilih instance group yang telah disetup sebelumnya → setup balancing mode, done → pilih/create Health check → Create

Host and Path rules, pada bagian ini anda dapat mengarahkan traffic berdasarkan host dan backend yang tersedia.

Pada bagian Frontend Configuration, isi nama → protocol HTTP → IP Version, create IP address → Done

Review and finalize → Create

The screenshot shows the 'Load balancer details' page for a load balancer named 'frontend-lb'. The page has tabs for 'Details', 'Monitoring', and 'Caching'. Under the 'Frontend' section, the configuration is: Protocol: HTTP, IP:Port: 34.107.1.61:80, Network Tier: Premium. The 'Host and path rules' section shows a single rule with Hosts: All unmatched (default), Paths: All unmatched (default), and Backend: frontend-grp-01. The 'Backend' section shows 'Backend services' with one service: '1. frontend-grp-01'. Its configuration includes: Endpoint protocol: HTTP, Named port: http, Timeout: 30 seconds, Cloud CDN: enabled, Traffic policy: disabled, and Health check: hc-01. An 'Advanced configurations' link is also present. Below this is a table of backend services:

Name	Type	Zone	Healthy	Autoscaling	Balancing mode	Capacity
frontend-grp-vm01	Instance group	asia-southeast1	1 / 1	On: Target CPU utilization 60%	Max backend utilization: 80%	100%

Test, untuk bagian ini saya menggunakan loader test untuk melakukan hit ke server untuk melihat autoscaling pada instance group.

