

Google Cloud Platform(GCP): 2.2 Planning and configuring compute resources.



Prashanta Paudel

Oct 16, 2018 · 8 min read



One of the most important aspects in GCP is it's Compute resources. In this blog, we will see how we do the planning and configuration in compute resources.

It should be clear by now that *we don't plan a compute resource alone but we plan a system or services that will need compute resources based in google cloud platform.*

How to Use the Google Cloud Platform |
Solutions Gallery | Google Cloud

Search the GCP documentation for tutorials and
solutions.

cloud.google.com

This website lists various solutions and what infrastructure are used for implementing that system.


We can choose from a range of options to use for the particular problem in hand. It may be sometimes possible that the same problem can be

solved by using 10 different methods.


The use of the particular google computing option is ruled by your need and features available in that product.

Google has mentioned some of the needs and features and common use cases.


The screenshots have been taken from google cloud website.

Product	Your needs	Product features	Typical use cases
 <p>Google Compute Engine</p> <p>Virtual machines running in Google's global data center network</p>	<ul style="list-style-type: none"> You need complete control over your infrastructure and direct access to high-performance hardware such as GPUs and local SSDs. You need to make OS-level changes, such as providing your own network or graphic drivers, to squeeze out the last drop of performance. You want to move your application from your own colo or datacenter to the cloud without rewriting it. You need to run a software package that can't easily be containerized or you want to use existing VM images. 	<ul style="list-style-type: none"> Virtual machines with network-attached and ultra-high performance local storage options. Preemptible virtual machines for inexpensive batch jobs and fault-tolerant workloads. Customizable load-balancing and auto-scaling across homogeneous VMs. Direct access to GPUs that you can use to accelerate specific workloads. Support for the most popular flavors of Linux and Windows operating systems. 	<ul style="list-style-type: none"> Any workload requiring a specific OS or OS configuration. Currently deployed, on-premises software that you want to run in the cloud.

Compute Engine use cases

Product	Your needs	Product features	Typical use cases
 <p>Google Kubernetes Engine</p> <p>Logical infrastructure powered by Kubernetes, the open source container orchestration system.</p>	<ul style="list-style-type: none"> You want to increase velocity and improve operability dramatically by separating the app from the OS. You need a secure, scalable way to manage containers in production. You don't have dependencies on a specific operating system. 	<ul style="list-style-type: none"> Logical infrastructure - focus on your app components, not virtual machines. Easy mechanisms for building loosely-coupled distributed systems. Run the same application on your laptop, on premise and in the cloud. 	<ul style="list-style-type: none"> Containerized workloads. Cloud-native distributed systems. Hybrid applications.

Kubernetes use cases

Product	Your needs	Product features	Typical use cases
 <p>Google App Engine</p> <p>A flexible, zero ops platform for building highly available apps</p>	<ul style="list-style-type: none"> You want to focus on writing code, and never want to touch a server, cluster, or infrastructure. You want to build a highly reliable and scalable serving app or component without doing it all yourself. You value developer velocity over infrastructure control. You want to minimize operational overhead. 	<ul style="list-style-type: none"> A range of curated serving stacks with smart defaults and deep customizability. Support for Java, Python, PHP, Go, Ruby, Node.js, and ASP.NET Core (beta) ... or bring your own app runtime. Integrated SDK, managed services, and local development environment. App versioning with zero-downtime upgrades. Traffic splitting. Automatic high availability with built-in auto-scaling. 	<ul style="list-style-type: none"> Web sites. Mobile app and gaming backends. RESTful APIs. Internal Line of Business (LOB) apps. Internet of things (IoT) apps.

App Engine Use cases

Google Compute Engine(GCE)

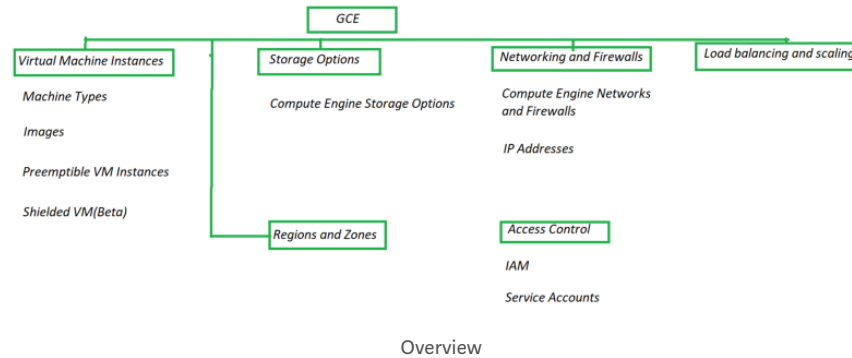


Google compute engine lets users create and run Virtual machines in Google Cloud Platform. It offers scale, performance, and value that allows us to easily launch a large compute cluster on Google's infrastructure. Using compute engine doesn't require any upfront cost. Compute Engine's tooling and workflow support enables scaling from single instances to global, load-balanced cloud computing.

Compute engine's VM come with various types of disks including persistent disk and SSD. You also have the ability to build custom VM with the amount of disk and memory you choose. On top of that, you will get a discount if you run it for a long period of time.

You can create a large cluster of compute resources and connect them to other data centers with fast and efficient google network.

Google Compute Engine Consists of Following



So, it is obvious you have to be very careful while selecting any solution based on the elements of the compute engine.

Virtual Machine Instances

An *instance* is a virtual machine (VM) hosted on Google's infrastructure. You can create an instance by using the Google Cloud Platform Console or the command-line tool.

Intro

Compute engine instances can run the public images for Linux and windows that Google explicitly provide on their platform as well as you can create or import from your existing systems. You can also deploy Docker containers, which are automatically launched on instances running the Container-Optimized OS public image. During creation, you can select no of virtual CPUs and memory by using a set of predefined machine types or by creating your own custom machine type.

Instances and projects

Instances belong to GCP console project and a project can have multiple instances. You have to define zone, OS and machine type while creating instances. When you delete the instance, it is removed from the project.

Instances and storage option

Each instance has a persistent disk where the operating system is installed. If required another disk can be attached to the instance.

Instances and networks

Each compute instance when created have one VPC(virtual private cloud). ***Up to 5 VPC can be added to the project.*** Instances in the same VPC communicate in local area network. Public IP is only used when it has to communicate outside the project.

Instances and containers

Compute Engine instances support a declarative method for launching your applications using containers. When creating a VM or an instance template, you can provide a Docker image name and launch configuration. Compute Engine will take care of the rest including supplying an up-to-date Container-Optimized OS image with Docker installed and launching your container when the VM starts up.

Google says:

Managing access to your instances

You can manage access to your instances using one of the following methods:

- Linux instances:

1. Managing Instance Access Using OS Login, which allows you to associate SSH keys with your Google account or G Suite account and manage admin or non-admin access to the instance through IAM roles. If you connect to your instances using the command-line tool or SSH from the console, Compute Engine can automatically generate SSH keys for you and apply them to your Google account or G Suite account.

2. Manage your SSH keys in project or instance metadata, which grants admin access to instances with metadata access that does not use OS Login. If you connect to your instances using the command-line tool or SSH from the console, Compute Engine can automatically generate SSH keys for you and apply them to project metadata.

- On Windows Server instances:

Create a password for a Windows Server instance

Machine Types

When implementing Virtual machines you are three options

1. Upload a custom VM from your system.
2. Use the standard VM's available in the system
3. Build a custom VM from available options

Typically defining a VM includes selecting memory size, Virtual CPUs, Disk Space etc.

So, let's see the different options

Predefined machine types

Predefined machine types have a fixed collection of resources. These are managed by Google Compute Engine and come in 4 classes

1. Standard machine types

Standard machine types are suitable for tasks that have a balance of CPU and memory needs. Standard machine types have 3.75 GB of memory per vCPU.

You can have 1 to 96 vCPUs and 3.75 to 360 GB memory.

Persistent disk usage is charged separately from machine type pricing.

2. High-memory machine types

High-memory machine types are ideal for tasks that require more memory relative to vCPUs. High-memory machine types have 6.50GB of system memory per vCPU.

You can have 2 to 96 vCPUs and 13 to 624 GB Memory.

3. High-CPU machine types

High-CPU machine types are ideal for tasks that require more vCPUs relative to memory. High-CPU machine types have 0.90 GB of memory per vCPU.

You can have 2 to 96 vCPUs and 1.80 to 86.4 GB memory

4. Shared-core machine types

Shared-core machine types provide one vCPU that is allowed to run for a portion of the time on a single hardware hyper-thread on the host CPU running your instance. Shared-core instances can be more cost-effective for running small, non-resource intensive applications than standard, high-memory or high-CPU machine types.

f1-micro Bursting

f1-micro machine types offer bursting capabilities that allow instances to use additional physical CPU for short periods of time. Bursting happens automatically when your instance requires more physical CPU than originally allocated. During these spikes, your instance will opportunistically take advantage of available physical CPU in bursts. Note that bursts are not permanent and are only possible periodically.

Machine name	Description	vCPUs	Memory (GB)	Max number of persistent disks (PDs) ¹	Max total PD size (TB)
f1-micro	Micro machine type with 0.2 vCPU, 0.60 GB of memory, backed by a shared physical core.	0.2	0.60	4 (16 in Beta)	3
g1-small	Shared-core machine type with 0.5 vCPU, 1.70 GB of memory, backed by a shared physical core.	0.5	1.70	4 (16 in Beta)	3

Persistent disk usage is charged separately from machine type pricing.

5. Memory-optimized machine types

Memory-optimized machine types are ideal for tasks that require intensive use of memory with higher memory to vCPU ratios than high-memory machine types. These machines types are perfectly suited for in-memory databases and in-memory analytics, such as SAP Hana and business warehousing (BW) workloads, genomics analysis, SQL

analysis services, and more. Memory-optimized machine types have greater than 14 GB of memory per vCPU.

See Regions and Zones to find where memory-optimized machine types are available.

Machine name	Description	vCPUs	Memory (GB)	Max number of persistent disks (PDs) ¹	Max total PD size (TB)	Local SSD
n1-ultramem-40	Memory-optimized machine type with 40 vCPUs and 961GB of memory.	40	961	16 (128 in Beta)	64	No
n1-ultramem-80	Memory-optimized machine type with 80 vCPUs and 1.87 TB of memory.	80	1922	16 (128 in Beta)	64	No
n1-megamem-96	Memory-optimized machine type with 96 vCPUs and 1.4 TB of memory.	96	1433.6	16 (128 in Beta)	64	Yes
n1-ultramem-160	Memory-optimized machine type with 160 vCPUs and 3.75 TB of memory.	160	3844	16 (128 in Beta)	64	No

Custom machine types

If your requirement is not matching any of the predefined machine types then you can go for custom machine type.

Custom machine types are ideal for the following scenarios:

- Workloads that are not a good fit for the predefined machine types that are available to you.
- Workloads that require more processing power or more memory, but don't need all of the upgrades that are provided by the next larger predefined machine type.

It costs slightly more to use a custom machine type than an equivalent predefined machine type, and there are still some limitations in the amount of memory and vCPUs you can select.

GPUs and machine types

You can attach GPUs only to instances with a predefined machine type or custom machine type that you are able to create in a zone. GPUs are not supported on shared-core machine types or memory-optimized machine types.

Preemptible VM Instances

Those instances that can be terminated anytime by compute engine are called preemptive VM instances. These VM's can be implemented at a lower price than normal instances since compute engine has the right to kill it anytime.

These kinds of instances are best for fault tolerant application such as batch processing that doesn't affect the system completely even if it is down or may continue later.

Preemptible instances function like normal instances, but have the following limitations:

- Compute Engine might terminate preemptible instances at any time due to system events. The probability that Compute Engine will terminate a preemptible instance for a system event is generally low but might vary from day to day and from zone to zone depending on current conditions.
- Compute Engine always terminates preemptible instances after they run for 24 hours.
- Preemptible instances are finite Compute Engine resources, so they might not always be available.
- Preemptible instances cannot live to migrate to a regular VM instance or be set to automatically restart when there is a maintenance event.
- Due to the above limitations, preemptible instances are not covered by any Service Level Agreement (and, for clarity, are excluded from the Google Compute Engine SLA).

Preemption process

Compute Engine performs the following steps to preempt an instance:

1. Compute Engine sends a preemption notice to the instance in the form of an ACPI G2 Soft Off signal. You can use a shutdown script to handle the preemption notice and complete cleanup actions before the instance stops.
2. If the instance does not stop after 30 seconds, Compute Engine sends an ACPI G3 Mechanical Off signal to the operating system.
3. Compute Engine transitions the instance to a `TERMINATED` state.

You can simulate an instance preemption by stopping the instance.

