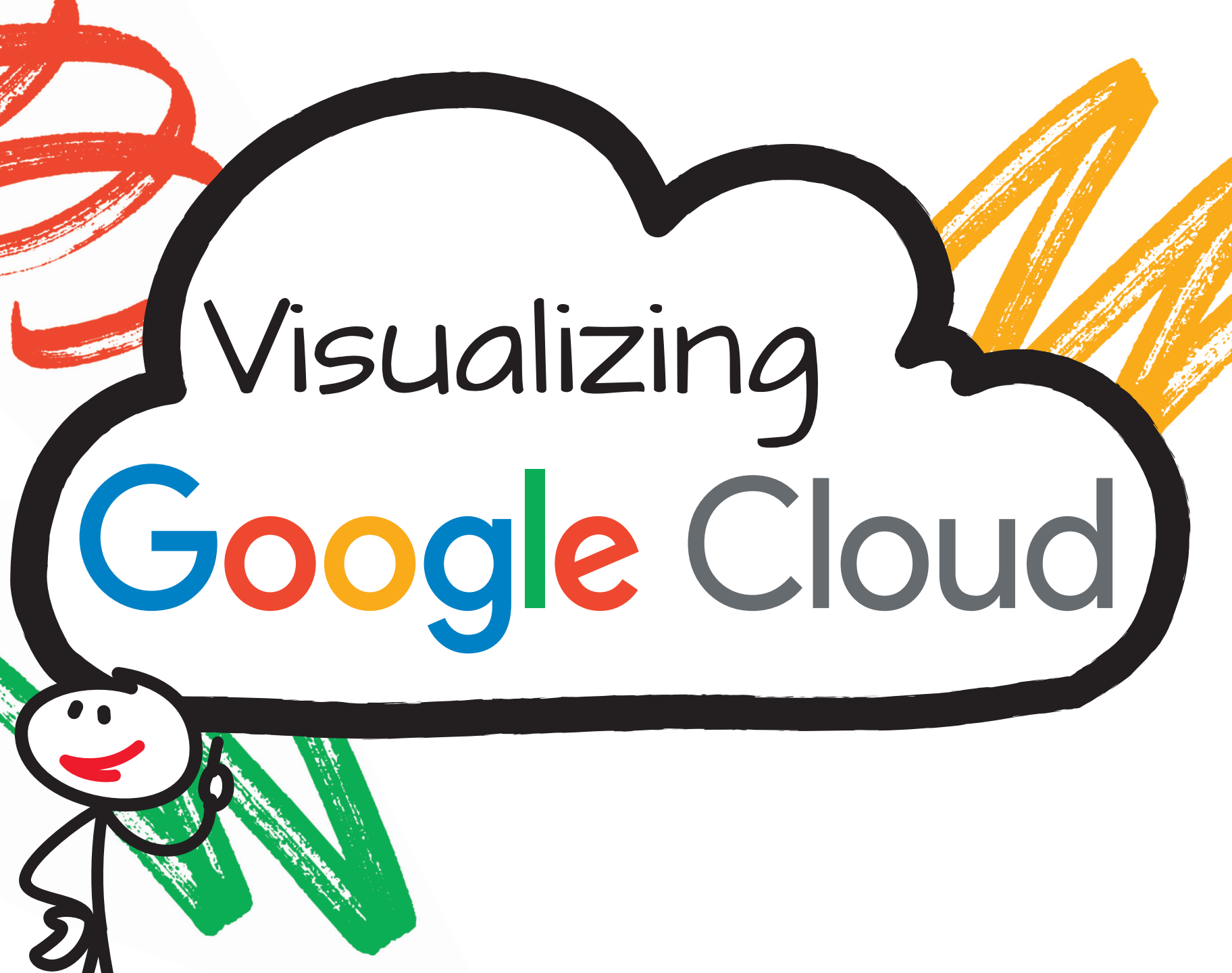
Priyanka Vergadia

Visualizing Google Cloud

101 Illustrated References for Cloud Engineers & Architects

WILEY


Visualizing
Google Cloud

Priyanka Vergadia

Visualizing Google Cloud

101 Illustrated References for Cloud Engineers & Architects

WILEY

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

ISBN: 978-1-119-81632-4
ISBN: 978-1-119-81637-9 (ebk)
ISBN: 978-1-119-81633-1 (ebk)

Library of Congress Control Number: 2022931550

Cover images: Stick Figures © Google; Cloud © Getty Images/LEOcrafts; Arrows © Adobe Stock/ vectortwins and Adobe Stock/ alex83m
Cover design: Wiley

For Shashank and Simba, the light of my life
Mom and Dad, who always believed in me
and
Google Cloud users everywhere

ACKNOWLEDGMENTS

This is my first book. I was excited to create something unique, but that goal required a lot of people to put a lot of trust in me and my wacky idea of illustrated explanations for technically complex Google Cloud concepts.

Greg Wilson, Reto Meier, Colt McAnlis, and the entire Google Cloud team believed in the idea and encouraged me to pursue it, and I am very grateful to them for that.

I also want to acknowledge Don Ulinski and Sean Carey from HD Interactive, without whom the illustrations in this book would not have been the same, as well as Jack Wilber, who tirelessly provided edits and recommendations for most of the chapters.

Further, this book would not have been possible without my technical reviewers, including Eric Brewer, who made themselves available to read early versions and those who reviewed the entire book and individual chapters.

Chapter 1: Brian Dorsey, Chelsie Czop (Peterson), Praveen Rajasekar, Steren Giannini, Matt Larkin, Vinod Ramachandran, Ken Drachnik, Kyle Meggs, Jason Polites, Jaisen Mathai, and Drew Bradstock.

Chapter 2: Geoffrey Noer, Ash Ahluwalia, Tad Hunt, Rahul Venkatraj, Lindsay Majane, Sean Derrington, Abhishek Lal, and Ajitesh Abhishek.

Chapter 3: Gabe Weiss, Vaibhav Govil, Minh Nguyen, Ron Pantofaro, Gopal Ashok, Anita Kibunguchy-Grant, and Michael Crutcher.

Chapter 4: Kir Titievsky, Zoltan Arato, Shan Kulandaivel, Susheel Kaushik, Soleil Kelley, Chaitanya (Chai) Pydimukkala, George Verghese, Filip Knapik, Etai Margolin, and Leigha Jarett.

Chapter 5: Richard Seroter, Wade Holmes, Arun Ananthampalayam, Nikhil Kaul, Kris Braun, Shikha Chetal, Lital Levy, David Feuer, and John Day.

Chapter 6: Ryan Przybyl, Adam Michelson, Kerry Takenaka, Tony Sarathchandra, Karthik Balakrishnan, Babi Seal, Tracy Jiang, Irene Abezgauz, Gautam Kulkarni, and Abhijeet Kalyan.

Chapter 7: Sara Robinson, Polong Lin, Karl Weinmeister, Sarah Weldon, Anu Srivastava, Logan Vadivelu, Marc Cohen, Shana Matthews, Shantanu Misra, Josh Porter, Calum Barnes, Lewis Liu, Zack Akil, Lee Boonstra, Arjun Rattan, and Mallika Iyer.

Chapter 8: Robert Sadowski, Max Saltonstall, Scott Ellis, and Jordanna Chord.

I also extend my deepest gratitude to:

- The entire Wiley team, who embraced the challenge of publishing this wacky idea as a book. I remember having my first conversation with Jim Minatel; my own excitement grew when I saw that he was immediately excited about this idea. Then while working further on the editorial process, Pete Gaughan and Kelly Talbot encouraged me constantly to stay on track with a solid deadline plan, without which the entire effort would not be possible. Pete even mentioned that this book is about to become a reference for other authors who want to share their technical thoughts in a visual format, and that really gave me a boost when I needed it!
- Shashank, my husband, for his constant support while I was working on this project over many weekends and evenings.
- Mom and Dad, for their lifelong support and for being understanding when I could not return their calls because I was immersed in the book.
- And. . .YOU for placing your trust in me and providing me with feedback, which ultimately made me pursue the idea of turning these sketches into a reference guide.

Thank you for everything!

—Priyanka Vergadia

ABOUT THE AUTHOR

Priyanka Vergadia has been working with cloud technology for a decade. She holds an M.S. in computer science from the University of Pennsylvania and a B.S. in electronics from Shri Govindram Seksaria Institute of Technology and Science, India. Now a Developer Advocate at Google Cloud, Priyanka works with companies and cloud architects to solve their most pressing business challenges using cloud computing. Her work has helped many cloud enthusiasts get started with the cloud, learn the fundamentals, and achieve cloud certifications.

Priyanka is passionate about making cloud computing approachable and easier to understand by combining her passions for art and technology. An expert technical visual storyteller, Priyanka has narrated thousands of technical stories that are fun to follow and easy to understand, and that make complex concepts a breeze to grasp. Some of her most popular work includes Build with Google Cloud, Architecting with Google Cloud, google/3g7xAC9, Deconstructing Chatbots, GCP Drawing Board, Cloud Bytes, GCP Comics, Get Cooking in Cloud. You can find all her work on the Google Cloud YouTube channel, her own YouTube channel The Cloud Girl, google/TheCloudGirl, her website thecloudgirl.dev, and her blog posts on Medium. You might also find her speaking at public and private developer events around the world.

CONTENTS

INTRODUCTION

Shortly after I started creating and sharing visual explanations of Google Cloud concepts in 2020, I began receiving overwhelmingly positive feedback. That feedback led me to think about pulling the visual explanations together into a reference guide. So here it is!

This book provides an easy-to-follow visual walkthrough of every important part of Google Cloud, from table stakes — compute, storage, database, security, and networking — to advanced concepts such as data analytics, data science, machine learning, and AI.

Most humans are visual learners; I am definitely one of them. I think it is safe to assume that you are too, since you picked up this book. So, even though it might sound cliché, I am a big believer that a picture is worth (more than) a thousand words. With that in mind, this book is my attempt at making Google Cloud technical concepts fun and interesting. This book covers the essentials of Google Cloud from end to end, with a visual explanation of each concept, how it works, and how you can apply it in your business use-case.

Who is this book for? Google Cloud enthusiasts! It is for anyone who is planning a cloud migration, new cloud deployment, preparing for cloud certification, and for anyone who is looking to make the most of Google Cloud. If you are cloud solutions architects, IT decision-makers, data and machine learning engineers you will find this book a good starting point. In short, this book is for you!

I have read thousands of pages of Google Cloud documentation and experimented with virtually every Google Cloud product and distilled that experience down to this book of accessible, bite-sized visuals. I hope this book helps you on your Google Cloud journey by making it both easier and more fun. Are you ready? Let's go!

Reader Support for This Book

How to Contact the Publisher

If you believe you've found a mistake in this book, please bring it to our attention. At John Wiley & Sons, we understand how important it is to provide our customers with accurate content, but even with our best efforts an error may occur.

In order to submit your possible errata, please email it to our Customer Service Team at wileysupport@wiley.com with the subject line "Possible Book Errata Submission."

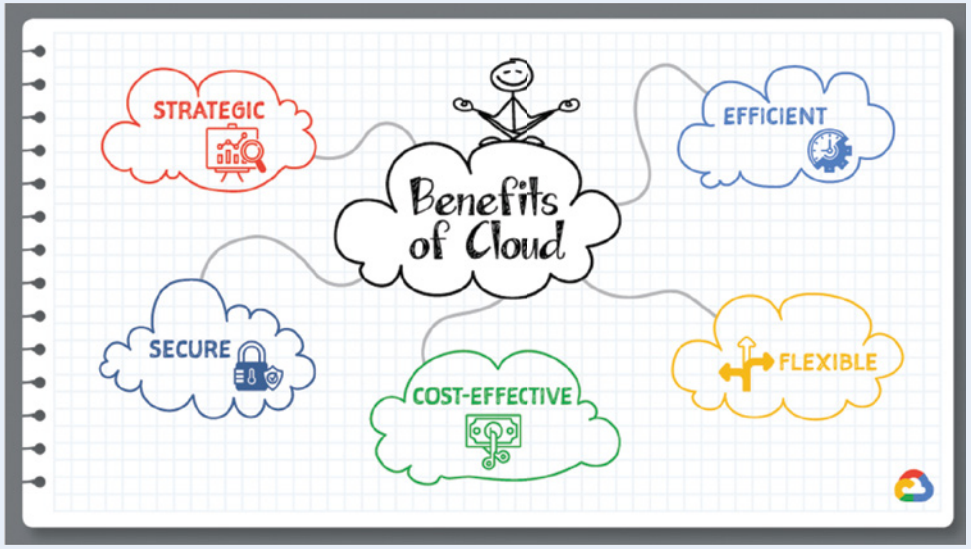
CHAPTER ONE

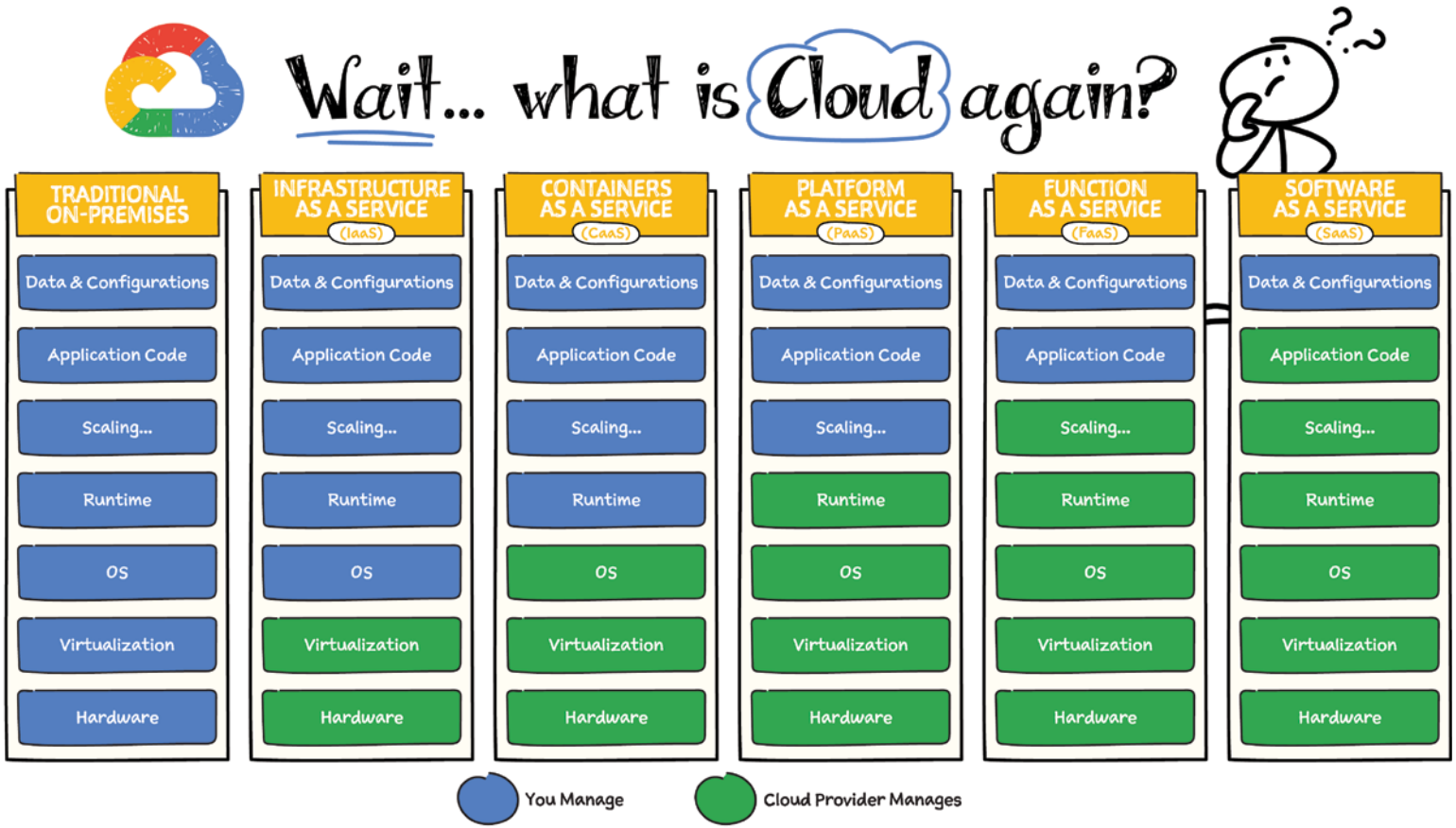
Infrastructure

Cloud computing is the on-demand availability of computing resources—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet. It eliminates the need for enterprises to procure, configure, or manage these resources themselves, while enabling them to pay only for what they use. The benefits of cloud computing include:

- Flexibility: You can access cloud resources from anywhere and scale services up or down as needed.
- Efficiency: You can develop new applications and rapidly get them into production, without worrying about the underlying infrastructure.
- Strategic value: When you choose a cloud provider that stays on top of the latest innovations and offers them as services, it opens opportunities for you to seize competitive advantages and higher returns on investment.
- Security: The depth and breadth of the security mechanisms provided by cloud providers offer stronger security than many enterprise data centers. Plus, cloud providers also have top security experts working on their offerings.
- Cost-effectiveness: You only pay for the computing resources you use. Because you don't need to overbuild data center capacity to handle unexpected spikes in demand or sudden surges in business growth, you can deploy resources and IT staff on more strategic initiatives.

In this first chapter, you will learn about cloud computing models and dive into the various compute options that Google Cloud offers. The following chapters provide a closer look at specific cloud resources and topics, including storage, databases, data analytics, networking, and more.

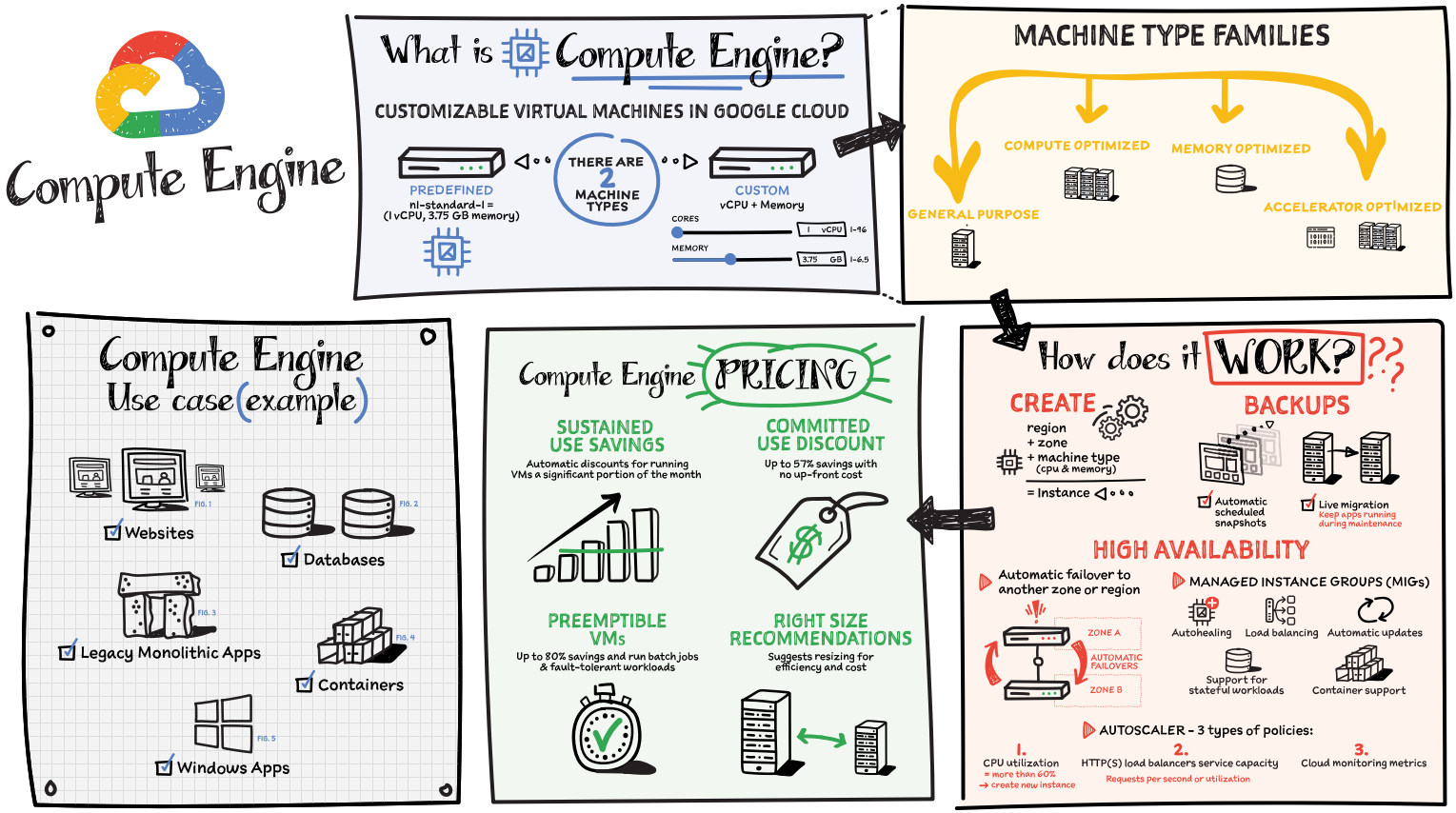



Wait... what is Cloud again?
TRADITIONAL ON-PREMISES
INFRASTRUCTURE AS A SERVICE (IaaS)
CONTAINERS AS A SERVICE (CaaS)
PLATFORM AS A SERVICE (PaaS)
FUNCTION AS A SERVICE (FaaS)
SOFTWARE AS A SERVICE (SaaS)
Data & Configurations
Application Code
Scaling...
Runtime
OS
Virtualization
Hardware
You Manage
Cloud Provider Manages

Introduction

To understand the cloud and the different models you can choose from, let's map it with an everyday analogy of housing:

- **On-Premises** — If you decide to make your house from scratch, you do everything yourself: source the raw materials, tools, put them together, and run to the store every time you need anything. That is very close to running your application on-premises, where you own everything from the hardware to your applications and scaling.
- **Infrastructure as a Service** — Now, if you are busy you consider hiring a contractor to build a custom house. You tell them how you want the house to look and how many rooms you want. They take the instructions and make you a house. IaaS is the same for your applications; you rent the hardware to run your application on, but you are responsible for managing the OS, runtime, scale, and the data. Example: GCE.
- **Containers as a Service** — If you know that buying is just too much work due to the maintenance it comes with, then you decide to rent a house. The basic utilities are included, but you bring your own furniture and make the space yours. Containers are the same where you bring a containerized application so that you don't have to worry about the underlying operating system but you still have control over scale and runtime. Example: GKE.
- **Platform as a Service** — If you just want to enjoy the space without having to even worry about furnishing it, then you rent a furnished house. That is what PaaS is for; you can bring your own code and deploy it and leave the scale to the cloud provider. Example: App Engine & Cloud Run.
- **Function as a Service** — If you just need a small dedicated space in which to work that is away from your home, you rent a desk in a workspace. That is close to what FaaS offers; you deploy a piece of code or a function that performs a specific task, and every time a function executes, the cloud provider adds scale if needed. Example: Cloud Functions.
- **Software as a Service** — Now, you move into the house (rented or purchased), but you pay for upkeep such as cleaning or lawn care. SaaS is the same; you pay for the service, you are responsible for your data, but everything else is taken care of by the provider. Example: Google Drive.


Compute Engine
What is Compute Engine?
CUSTOMIZABLE VIRTUAL MACHINES IN GOOGLE CLOUD
THERE ARE 2 MACHINE TYPES
PREDEFINED
n1-standard-1 =
(1 vCPU, 3.75 GB memory)
CUSTOM
vCPU + Memory
CORES
1 vCPU
1-96
MEMORY
3.75 GB
1-6.5
MACHINE TYPE FAMILIES
COMPUTE OPTIMIZED
MEMORY OPTIMIZED
GENERAL PURPOSE
ACCELERATOR OPTIMIZED
Compute Engine
Use case (example)
Websites
Databases
Legacy Monolithic Apps
Containers
Windows Apps
Compute Engine PRICING
SUSTAINED USE SAVINGS
Automatic discounts for running VMs a significant portion of the month
COMMITTED USE DISCOUNT
Up to 57% savings with no up-front cost
PREEMPTIBLE VMs
Up to 80% savings and run batch jobs & fault-tolerant workloads
RIGHT SIZE RECOMMENDATIONS
Suggests resizing for efficiency and cost
How does it WORK???
CREATE
region
+ zone
+ machine type
(cpu & memory)
= Instance
BACKUPS
Automatic scheduled snapshots
Live migration
Keep apps running during maintenance
HIGH AVAILABILITY
Automatic failover to another zone or region
ZONE A
AUTOMATIC FAILOVERS
ZONE B
MANAGED INSTANCE GROUPS (MIGs)
Autohealing
Load balancing
Automatic updates
Support for stateful workloads
Container support
AUTOSCALER - 3 types of policies:
1. CPU utilization = more than 60% → create new instance
2. HTTP(S) load balancers service capacity Requests per second or utilization
3. Cloud monitoring metrics

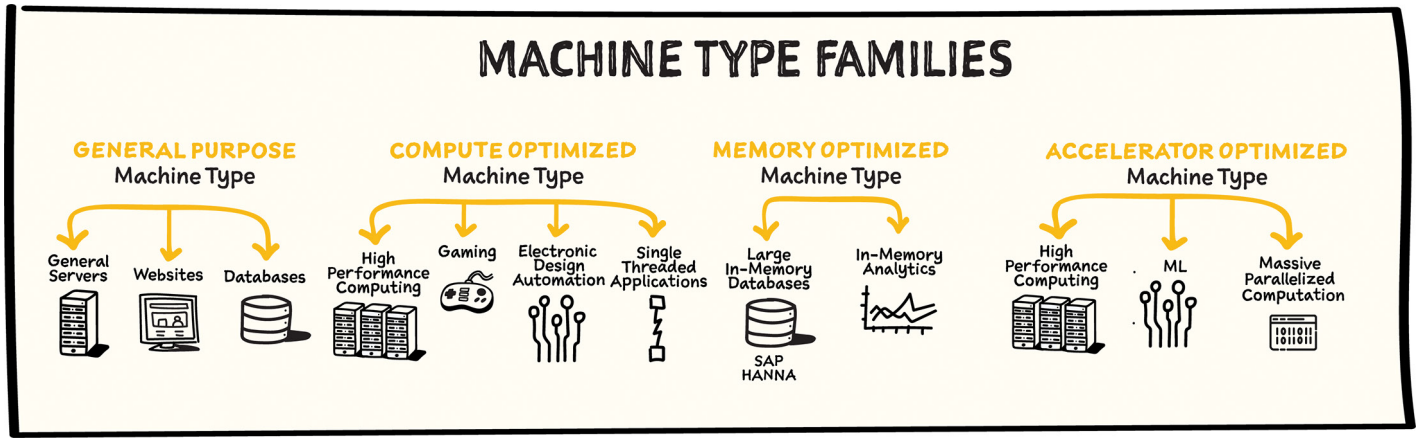
What Is Compute Engine?

Compute Engine is a customizable compute service that lets you create and run virtual machines on Google's infrastructure. You can create a virtual machine (VM) that fits your needs. Predefined machine types are prebuilt and ready-to-go configurations of VMs with specific amounts of vCPU and memory to start running apps quickly. With Custom Machine Types, you can create virtual machines with the optimal amount of CPU and memory for your workloads. This allows you to tailor your infrastructure to your workload. If requirements change, using the stop/start feature you can move your workload to a smaller or larger Custom Machine Type instance, or to a predefined configuration.

Machine Types

In Compute Engine, machine types are grouped and curated by families for different workloads. You can choose from general-purpose, memory-optimized, compute-optimized, and accelerator-optimized families.

- **General-purpose** machines are used for day-to-day computing at a lower cost and for balanced price/performance across a wide range of VM shapes. The use cases that best fit here are web serving, app serving, back office applications, databases, cache, media-streaming, microservices, virtual desktops, and development environments.
- **Memory-optimized** machines are recommended for ultra high-memory workloads such as in-memory analytics and large in-memory databases such as SAP HANA.
- **Compute-optimized** machines are recommended for ultra high-performance workloads such as High Performance Computing (HPC), Electronic Design Automation (EDA), gaming, video transcoding, and single-threaded applications.
- **Accelerator-optimized** machines are optimized for high-performance computing workloads such as machine learning (ML), massive parallelized computations, and High Performance Computing (HPC).



How Does It Work?

You can create a VM instance using a boot disk image, a boot disk snapshot, or a container image. The image can be a public operating system (OS) image or a custom one. Depending on where your users are, you can define the zone you want the virtual machine to be created in. By default all traffic from the Internet is blocked by the firewall, and you can enable the HTTP(s) traffic if needed.

Use snapshot schedules (hourly, daily, or weekly) as a best practice to back up your Compute Engine workloads. Compute Engine offers live migration by default to keep your virtual machine instances running even when software or hardware update occurs. Your running instances are migrated to another host in the same zone instead of requiring your VMs to be rebooted.

Availability

For High Availability (HA), Compute Engine offers automatic failover to other regions or zones in event of a failure. Managed instance groups (MIGs) help keep the instances running by automatically replicating instances from a predefined image. They also provide application-based auto-healing health checks. If an application is not responding on a VM, the auto-healer automatically re-creates that VM for you. Regional MIGs let you spread app load across multiple zones. This replication protects against zonal failures. MIGs work with load-balancing services to distribute traffic across all of the instances in the group.

Compute Engine offers autoscaling to automatically add or remove VM instances from a managed instance group based on increases or decreases in load. Autoscaling lets your apps gracefully handle increases in traffic, and it reduces cost when the need for resources is lower. You define the autoscaling policy for automatic scaling based on the measured load, CPU utilization, requests per second, or other metrics.

Active Assist's new feature, predictive autoscaling, helps improve response times for your applications. When you enable predictive autoscaling, Compute Engine forecasts future load based on your MIG's history and scales it out in advance of predicted load so that new instances are ready to serve when the load arrives. Without predictive autoscaling, an autoscaler can only scale a group reactively, based on observed changes in load in real time. With predictive autoscaling enabled, the autoscaler works with real-time data as well as with historical data to cover both the current and forecasted load. That makes predictive autoscaling ideal for those apps with long initialization times and whose workloads vary predictably with daily or weekly cycles. For more information, see How predictive autoscaling works or check if predictive autoscaling is suitable for your workload, and to learn more about other intelligent features, check out Active Assist.

Pricing

You pay for what you use. But you can save cost by taking advantage of some discounts! Sustained use savings are automatic discounts applied for running instances for a significant portion of the month. If you know your usage upfront, you can take advantage of committed use discounts, which can lead to significant savings without any upfront cost. And by using short-lived preemptive instances, you can save up to 80%; they are great for batch jobs and fault-tolerant workloads. You can also optimize resource utilization with automatic recommendations. For example, if you are using a bigger instance for a workload that can run on a smaller instance, you can save costs by applying these recommendations.

Security

Compute Engine provides you default hardware security. Using Identity and Access Management (IAM) you just have to ensure that proper permissions

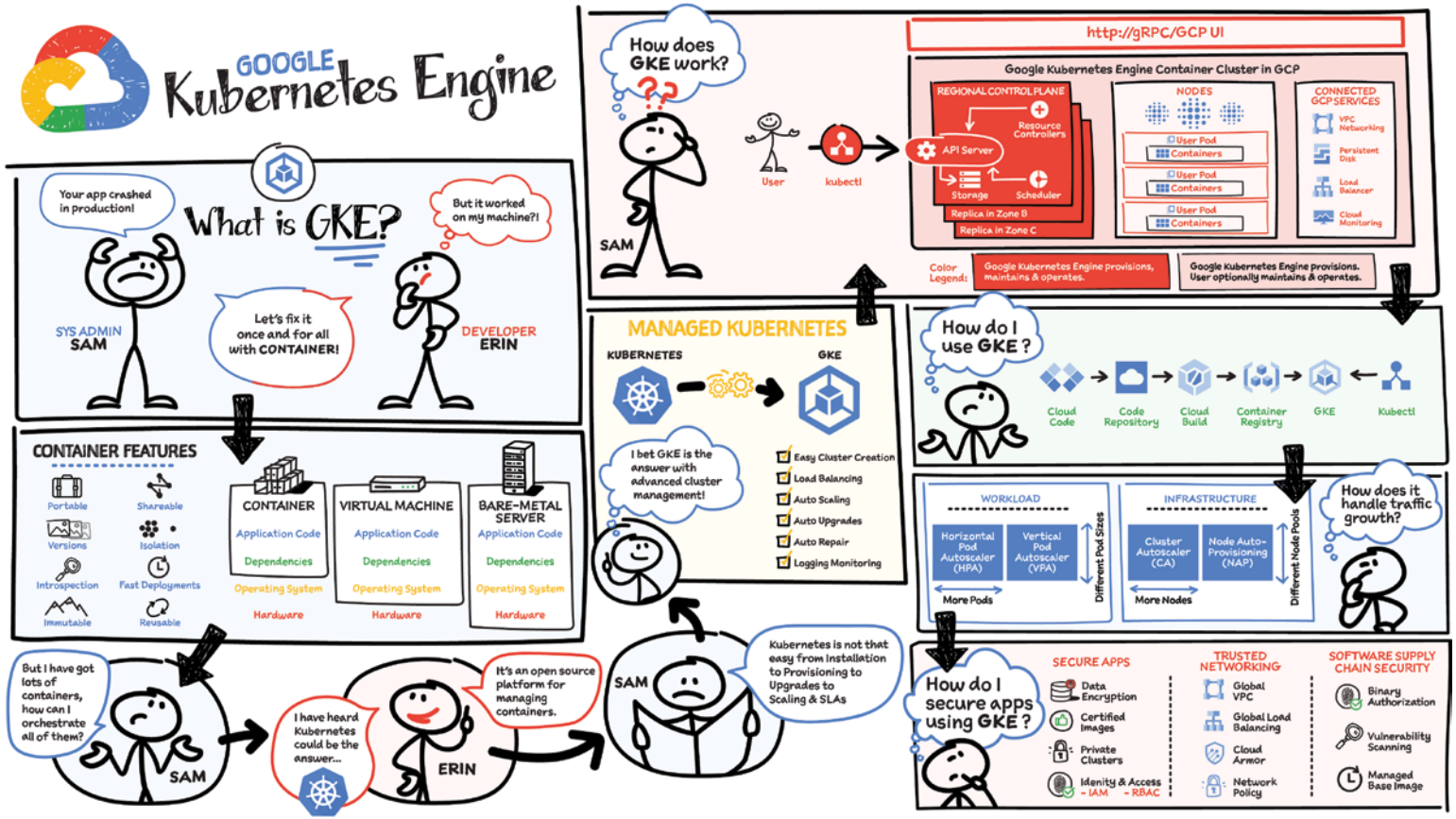
are given to control access to your VM resources. All the other basic security principles apply; if the resources are not related and don't require network communication among themselves, consider hosting them on different VPC networks. By default, users in a project can create persistent disks or copy images using any of the public images or any images that project members can access through IAM roles. You may want to restrict your project members so that they can create boot disks only from images that contain approved software that meet your policy or security requirements. You can define an organization policy that only allows Compute Engine VMs to be created from approved images. This can be done by using the Trusted Images Policy to enforce images that can be used in your organization.

By default all VM families are Shielded VMs. Shielded VMs are virtual machine instances that are hardened with a set of easily configurable security features to ensure that when your VM boots, it's running a verified bootloader and kernel — it's the default for everyone using Compute Engine, at no additional charge. For more details on Shielded VMs, refer to the documentation at https://cloud.google.com/compute/shielded-vm/docs/shielded-vm.

For additional security, you also have the option to use Confidential VM to encrypt your data in use while it's being processed in Compute Engine. For more details on Confidential VM, refer to the documentation at https://cloud.google.com/compute/confidential-vm/docs/about-cvm.

Use Cases

There are many use cases Compute Engine can serve in addition to running websites and databases. You can also migrate your existing systems onto Google Cloud, with Migrate for Compute Engine, enabling you to run stateful workloads in the cloud within minutes rather than days or weeks. Windows, Oracle, and VMware applications have solution sets, enabling a smooth transition to Google Cloud. To run Windows applications, either bring your own license leveraging sole-tenant nodes or use the included licensed images.


GOOGLE
Kubernetes Engine
What is GKE?
Your app crashed in production!
But it worked on my machine?!
SYS ADMIN SAM
Let's fix it once and for all with CONTAINER!
DEVELOPER ERIN
CONTAINER FEATURES
Portable
Shareable
Versions
Isolation
Introspection
Fast Deployments
Immutable
Reusable
CONTAINER
Application Code
Dependencies
Operating System
Hardware
VIRTUAL MACHINE
Application Code
Dependencies
Operating System
Hardware
BARE-METAL SERVER
Application Code
Dependencies
Operating System
Hardware
But I have got lots of containers, how can I orchestrate all of them?
SAM
I have heard Kubernetes could be the answer...
ERIN
It's an open source platform for managing containers.
How does GKE work?
SAM
User
kubectl
http://gRPC/GCP UI
Google Kubernetes Engine Container Cluster in GCP
REGIONAL CONTROL PLANE
Resource Controllers
API Server
Storage
Scheduler
Replica in Zone B
Replica in Zone C
NODES
User Pod
Containers
User Pod
Containers
User Pod
Containers
CONNECTED GCP SERVICES
VPC Networking
Persistent Disk
Load Balancer
Cloud Monitoring
Color Legend:
Google Kubernetes Engine provisions, maintains & operates.
Google Kubernetes Engine provisions. User optionally maintains & operates.
MANAGED KUBERNETES
KUBERNETES
GKE
I bet GKE is the answer with advanced cluster management!
Easy Cluster Creation
Load Balancing
Auto Scaling
Auto Upgrades
Auto Repair
Logging Monitoring
SAM
Kubernetes is not that easy from Installation to Provisioning to Upgrades to Scaling & SLAs
How do I use GKE?
Cloud Code
Code Repository
Cloud Build
Container Registry
GKE
Kubectl
WORKLOAD
Horizontal Pod Autoscaler (HPA)
Vertical Pod Autoscaler (VPA)
More Pods
Different Pod Sizes
INFRASTRUCTURE
Cluster Autoscaler (CA)
Node Auto-Provisioning (NAP)
More Nodes
Different Node Pools
How does it handle traffic growth?
How do I secure apps using GKE?
SECURE APPS
Data Encryption
Certified Images
Private Clusters
Identity & Access - IAM - RBAC
TRUSTED NETWORKING
Global VPC
Global Load Balancing
Cloud Armor
Network Policy
SOFTWARE SUPPLY CHAIN SECURITY
Binary Authorization
Vulnerability Scanning
Managed Base Image

Why Containers?

Containers are often compared with virtual machines (VMs). You might already be familiar with VMs: a guest operating system such as Linux or Windows runs on top of a host operating system with virtualized access to the underlying hardware. Like virtual machines, containers enable you to package your application together with libraries and other dependencies, providing isolated environments for running your software services. As you'll see, however, the similarities end here as containers offer a far more lightweight unit for developers and IT Ops teams to work with, bringing a myriad of benefits.

Instead of virtualizing the hardware stack as with the virtual machines approach, containers virtualize at the operating system level, with multiple containers running atop the OS kernel directly. This means that containers are far more lightweight: They share the OS kernel, start much faster, and use a fraction of the memory compared to booting an entire OS.

Containers help improve portability, shareability, deployment speed, reusability, and more. More importantly to the team, containers made it possible to solve the "*it worked on my machine*" problem.

Why Kubernetes?

The system administrator is usually responsible for more than just one developer. They have several considerations when rolling out software:

- Will it work on all the machines?
- If it doesn't work, then what?
- What happens if traffic spikes? (System admin decides to over-provision just in case. . .)

With lots of developers containerizing their apps, the system administrator needs a better way to orchestrate all the containers that developers ship. The solution: Kubernetes!

What Is So Cool about Kubernetes?

The Mindful Container team had a bunch of servers and used to make decisions on what ran on each manually based on what they knew would conflict if it were to run on the same machine. If they were lucky, they might have some sort of scripted system for rolling out software, but it usually involved SSHing into each machine. Now with containers — and the isolation they provide — they can trust that in most cases, any two applications can fairly share the resources of the same machine.

With Kubernetes, the team can now introduce a control plane that makes decisions for them on where to run applications. And even better, it doesn't just statically place them; it can continually monitor the state of each machine and make adjustments to the state to ensure that what is happening is what they've actually specified. Kubernetes runs with a control plane, and on a number of nodes. We install a piece of software called the kubelet on each node, which reports the state back to the primary.

Here is how it works:

- The primary controls the cluster.
- The worker nodes run pods.
- A pod holds a set of containers.
- Pods are bin-packed as efficiently as configuration and hardware allows.
- Controllers provide safeguards so that pods run according to specification (reconciliation loops).
- All components can be deployed in high-availability mode and spread across zones or data centers.

Kubernetes orchestrates containers across a fleet of machines, with support for:

- Automated deployment and replication of containers
- Online scale — in and scale — out of container clusters

- Load balancing over groups of containers
- Rolling upgrades of application containers
- Resiliency, with automated rescheduling of failed containers (i.e., self-healing of container instances)
- Controlled exposure of network ports to systems outside the cluster

A few more things to know about Kubernetes:

- Instead of flying a plane, you program an autopilot: declare a desired state, and Kubernetes will make it true — and continue to keep it true.
- It was inspired by Google's tools for running data centers efficiently.
- It has seen unprecedented community activity and is today one of the largest projects on GitHub. Google remains the top contributor.

The magic of Kubernetes starts happening when we don't require a sysadmin to make the decisions. Instead, we enable a build and deployment pipeline. When a build succeeds, passes all tests, and is signed off, it can automatically be deployed to the cluster gradually, blue/green, or immediately.

Kubernetes the Hard Way

By far, the single biggest obstacle to using Kubernetes (k8s) is learning how to install and manage your own cluster. Check out k8s the Hard Way, a step-by-step guide to install a k8s cluster. You have to think about tasks like:

- Choosing a cloud provider or bare metal
- Provisioning machines
- Picking an OS and container runtime
- Configuring networking (e.g., P ranges for pods, SDNs, LBs)
- Setting up security (e.g., generating certs and configuring encryption)
- Starting up cluster services such as DNS, logging, and monitoring

Once you have all these pieces together, you can finally start to use k8s and deploy your first application. And you're feeling great and happy and k8s is awesome! But then, you have to roll out an update...

Wouldn't it be great if *Mindful Containers* could start clusters with a single click, view all their clusters and workloads in a single pane of glass, and have Google continually manage their cluster to scale it and keep it healthy?

What Is GKE?

GKE is a secured and fully managed Kubernetes service. It provides an easy-to-use environment for deploying, managing, and scaling your containerized applications using Google infrastructure.

Mindful Containers decided to use GKE to enable development self-service by delegating release power to developers and software.

Why GKE?

- Production-ready with autopilot mode of operation for hands-off experience
- Best-in-class developer tooling with consistent support for first- and third-party tools
- Offers container-native networking with a unique BeyondProd security approach
- Most scalable Kubernetes service; only GKE can run 15,000 node clusters, outscaling competition up to 15X
- Industry-first to provide fully managed Kubernetes service that implements full Kubernetes API, 4-way autoscaling, release channels, and multicluster support

How Does GKE Work?

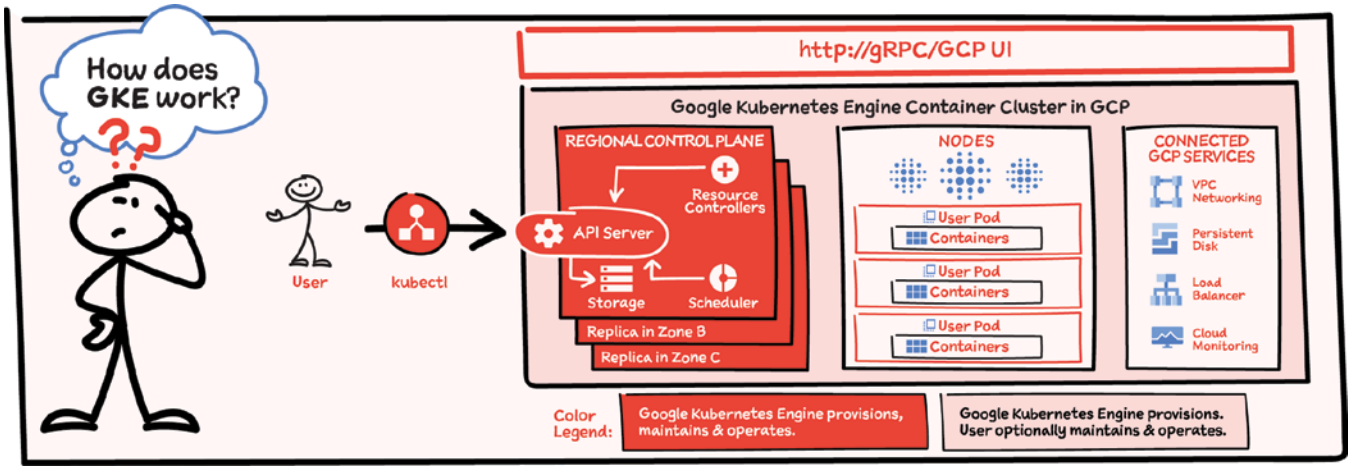
The GKE control plane is fully operated by the Google SRE (Site Reliability Engineering) team with managed availability, security patching, and upgrades. The Google SRE team not only has deep operational knowledge of k8s, but is also uniquely positioned to get early insights on any potential issues by managing a fleet of tens of thousands of clusters. That's something that is simply not possible to achieve with self-managed k8s. GKE also provides comprehensive management for nodes, including autoprovisioning, security patching, opt-in auto-upgrade, repair, and scaling. On top of that, GKE provides end-to-end container security, including private and hybrid networking.

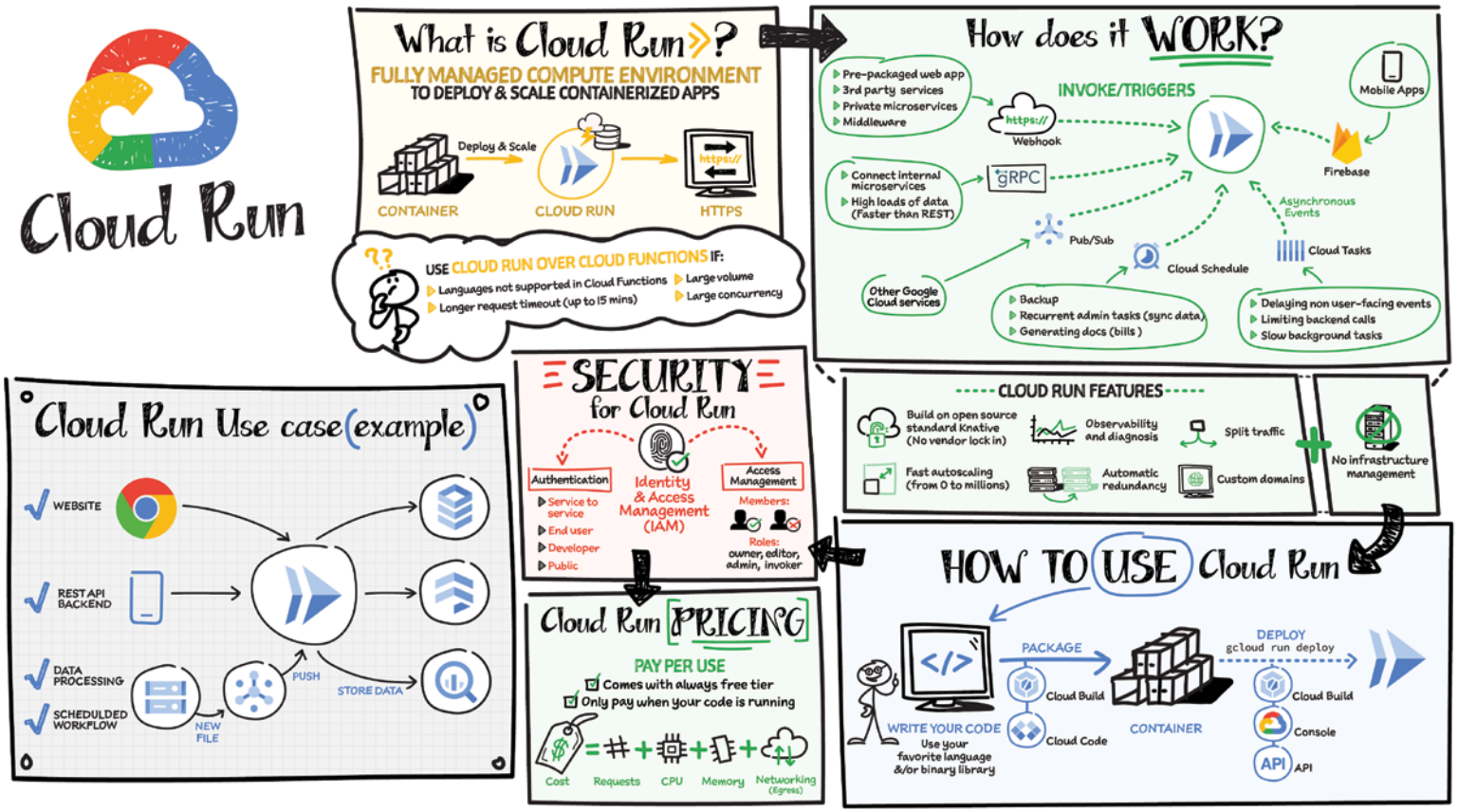
How Does GKE Make Scaling Easy?

As the demand for *Mindful Containers* grows, they now need to scale their services. Manually scaling a Kubernetes cluster for availability and reliability can be complex and time consuming. GKE automatically scales the number of pods and nodes based on the resource consumption of services.

- Vertical Pod Autoscaler (VPA) watches resource utilization of your deployments and adjusts requested CPU and RAM to stabilize the workloads.
- Node Auto Provisioning optimizes cluster resources with an enhanced version of Cluster Autoscaling.

In addition to the fully managed control plane that GKE offers, using the Autopilot mode of operation automatically applies industry best practices and can eliminate all node management operations, maximizing your cluster efficiency and helping to provide a stronger security posture.




Cloud Run
What is Cloud Run?
FULLY MANAGED COMPUTE ENVIRONMENT
TO DEPLOY & SCALE CONTAINERIZED APPS
Deploy & Scale
CONTAINER
CLOUD RUN
HTTPS
USE CLOUD RUN OVER CLOUD FUNCTIONS IF:
Languages not supported in Cloud Functions
Longer request timeout (up to 15 mins)
Large volume
Large concurrency
How does it WORK?
Pre-packaged web app
3rd party services
Private microservices
Middleware
INVOKE/TRIGGERS
Webhook
Mobile Apps
Firebase
Connect internal microservices
High loads of data (Faster than REST)
gRPC
Asynchronous Events
Pub/Sub
Cloud Schedule
Cloud Tasks
Other Google Cloud services
Backup
Recurrent admin tasks (sync data)
Generating docs (bills)
Delaying non user-facing events
Limiting backend calls
Slow background tasks
Cloud Run Use case (example)
WEBSITE
REST API BACKEND
DATA PROCESSING
SCHEDULDED WORKFLOW
NEW FILE
PUSH
STORE DATA
SECURITY for Cloud Run
Authentication
Service to service
End user
Developer
Public
Identity & Access Management (IAM)
Access Management
Members:
Roles: owner, editor, admin, invoker
CLOUD RUN FEATURES
Build on open source standard Knative (No vendor lock in)
Observability and diagnosis
Split traffic
Fast autoscaling (from 0 to millions)
Automatic redundancy
Custom domains
No infrastructure management
Cloud Run PRICING
PAY PER USE
Comes with always free tier
Only pay when your code is running
Cost
Requests
CPU
Memory
Networking (Egress)
HOW TO USE Cloud Run
WRITE YOUR CODE
Use your favorite language &/or binary library
PACKAGE
Cloud Build
Cloud Code
CONTAINER
DEPLOY
gcloud run deploy
Cloud Build
Console
API
API

What Is Cloud Run?

Cloud Run is a fully managed compute environment for deploying and scaling serverless HTTP containers without worrying about provisioning machines, configuring clusters, or autoscaling.

- **No vendor lock-in** — Because Cloud Run takes standard OCI containers and implements the standard Knative Serving API, you can easily port over your applications to on-premises or any other cloud environment.
- **Fast autoscaling** — Microservices deployed in Cloud Run scale automatically based on the number of incoming requests, without you having to configure or manage a full-fledged Kubernetes cluster. Cloud Run scales to zero — that is, uses no resources — if there are no requests.
- **Split traffic** — Cloud Run enables you to split traffic between multiple revisions, so you can perform gradual rollouts such as canary deployments or blue/green deployments.
- **Custom domains** — You can set up custom domain mapping in Cloud Run, and it will provision a TLS certificate for your domain.
- **Automatic redundancy** — Cloud Run offers automatic redundancy so you don't have to worry about creating multiple instances for high availability.

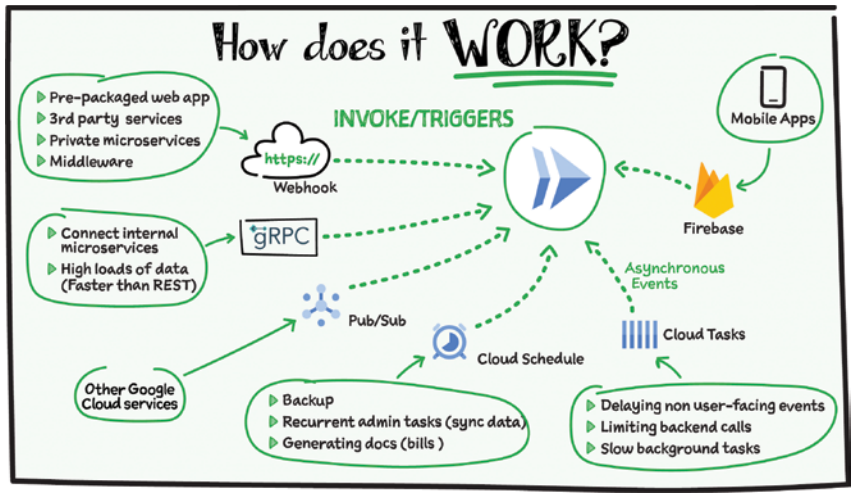
How to Use Cloud Run

With Cloud Run, you write your code in your favorite language and/or use a binary library of your choice. Then push it to Cloud Build to create a container build. With a single command — `gcloud run deploy` — you go from a container image to a fully managed web application that runs on a domain with a TLS certificate and autoscales with requests.

How Does Cloud Run Work?

Cloud Run service can be invoked in the following ways:

- **HTTPS:** You can send HTTPS requests to trigger a Cloud Run-hosted service. Note that all Cloud Run services have a stable HTTPS URL. Some use cases include:
  - Custom RESTful web API
  - Private microservice
  - HTTP middleware or reverse proxy for your web applications
  - Prepackaged web application
- **gRPC:** You can use gRPC to connect Cloud Run services with other services — for example, to provide simple, high-performance communication between internal microservices. gRPC is a good option when you:
  - Want to communicate between internal microservices
  - Support high data loads (gRPC uses protocol buffers, which are up to seven times faster than REST calls)
  - Need only a simple service definition and you don't want to write a full client library
  - Use streaming gRPCs in your gRPC server to build more responsive applications and APIs
- **WebSockets:** WebSockets applications are supported on Cloud Run with no additional configuration required. Potential use cases include any application that requires a streaming service, such as a chat application.
- **Trigger from Pub/Sub:** You can use Pub/Sub to push messages to the endpoint of your Cloud Run service, where the messages are subsequently delivered to containers as HTTP requests. Possible use cases include:
  - Transforming data after receiving an event upon a file upload to a Cloud Storage bucket



  - Processing your Google Cloud operations suite logs with Cloud Run by exporting them to Pub/Sub
  - Publishing and processing your own custom events from your Cloud Run services
- **Running services on a schedule:** You can use Cloud Scheduler to securely trigger a Cloud Run service on a schedule. This is similar to using cron jobs. Possible use cases include:
  - Performing backups on a regular basis
  - Performing recurrent administration tasks, such as regenerating a sitemap or deleting old data, content, configurations, synchronizations, or revisions
  - Generating bills or other documents
- **Executing asynchronous tasks:** You can use Cloud Tasks to securely enqueue a task to be asynchronously processed by a Cloud Run service. Typical use cases include:
  - Handling requests through unexpected production incidents
  - Smoothing traffic spikes by delaying work that is not user-facing
  - Reducing user response time by delegating slow background operations, such as database updates or batch processing, to be handled by another service
  - Limiting the call rate to backend services like databases and third-party APIs
- **Events from Eventrac:** You can trigger Cloud Run with events from more than 60 Google Cloud sources. For example:
  - Use a Cloud Storage event (via Cloud Audit Logs) to trigger a data processing pipeline
  - Use a BigQuery event (via Cloud Audit Logs) to initiate downstream processing in Cloud Run each time a job is completed