Google Cloud Fundamentals: Core Infrastructure

Module 1: Introducing Google Cloud

Overview of Cloud Computing

Cloud computing is an IT paradigm defined by five essential characteristics:

1. **On-demand self-service**: Customers can provision computing resources (processing power, storage, networking) automatically without human intervention.
2. **Broad network access**: Resources are available over the internet from anywhere.
3. **Resource pooling**: Providers serve multiple customers from shared physical resources.
4. **Rapid elasticity**: Resources can scale up or down quickly based on demand.
5. **Measured service**: Customers pay only for what they use (pay-as-you-go model).

History of Cloud Computing

1. **First wave - Colocation**: Companies rented physical space in data centers instead of building their own.
2. **Second wave - Virtualized data centers**: Virtual versions of physical infrastructure components (servers, CPUs, disks).
3. **Third wave - Container-based architecture**: Fully automated, elastic cloud with automated services and scalable data.

Cloud Service Models

1. **Infrastructure as a Service (IaaS)**: Provides raw compute, storage, and network capabilities (e.g., Google Compute Engine).
    - Customers pay for allocated resources.
2. **Platform as a Service (PaaS)**: Manages hardware and software, providing libraries to bind application code to infrastructure (e.g., Google App Engine).
    - Customers pay for actual usage.
3. **Software as a Service (SaaS)**: Complete cloud-based applications (e.g., Gmail, Google Docs).

Google Cloud Network

- **Global infrastructure**: Largest network of its kind with billions in investment.
- **High throughput design**: 100+ content caching nodes worldwide for low latency.

- **Geographic organization**:
    - **Locations**: 7 major geographic areas (North America, Europe, Asia, etc.)
    - **Regions**: Independent geographic areas (e.g., London/europe-west2).
    - **Zones**: Deployment areas within regions (e.g., europe-west2-a).
- **Multi-region support**: Some services can replicate across multiple regions for redundancy.

Environmental Impact

- Data centers consume ~2% of world's electricity.
- Google initiatives:
    - First major company to be carbon neutral.
    - First to achieve 100% renewable energy.
    - Goal to operate carbon-free by 2030.
    - Innovative cooling systems (e.g., seawater cooling in Finland).

Security

Google implements security at multiple levels:

1. **Hardware infrastructure**:
    - Custom-designed server boards and chips.
    - Secure boot stack with cryptographic signatures.
    - Multi-layer physical security.
2. **Service deployment**:
    - Encrypted inter-service communication.
3. **User identity**:
    - Risk-based authentication challenges.
    - Support for Universal 2nd Factor (U2F) security keys.
4. **Storage services**:
    - Encryption at rest with centrally managed keys.
5. **Internet communication**:
    - Google Front End (GFE) for TLS termination.
    - Multi-tier DDoS protection.
6. **Operational security**:
    - Intrusion detection systems.
    - Strict employee access controls.
    - Mandatory U2F for employees.

Open APIs and Open Source

- Google avoids vendor lock-in by supporting open standards.
- Examples:
    - TensorFlow (open-source ML library).

    - Kubernetes (container orchestration).
    - Interoperability across cloud providers.

Pricing and Billing

- **Per-second billing**: For Compute Engine, Kubernetes Engine, etc.
- **Discounts**:
    - Sustained-use discounts for long-running instances.
    - Custom machine types for optimal resource allocation.
- **Cost control tools**:
    - Pricing calculator.
    - Budgets and alerts.
    - Quotas (rate and allocation).

Module 2: Resources and Access in the Cloud

Google Cloud Resource Hierarchy

1. **Organization node**: Top-level container (requires Google Workspace or Cloud Identity).
2. **Folders**: Can contain projects or other folders (for departmental organization).
3. **Projects**: Fundamental organizing entity that contains resources.
4. **Resources**: Individual cloud components (VMs, storage buckets, etc.).

- **Policy inheritance**: Policies applied at higher levels flow downward.

Projects

- Fundamental unit for enabling services, managing APIs, and billing.
- Attributes:
    - **Project ID**: Globally unique, immutable after creation.
    - **Project name**: User-defined, mutable.
    - **Project number**: Google-assigned, immutable.
- Managed via **Resource Manager** API.

Identity and Access Management (IAM)

Defines "who can do what on which resources":

- **Who (principal)**: Google account, Google group, service account, or Cloud Identity domain.
- **What (role)**: Collection of permissions.
- **Which resource**: Hierarchy element (organization, folder, project, or resource).

IAM Roles

1. **Basic roles**:
   - Owner: Full control + role management.
   - Editor: Modify resources.
   - Viewer: Read-only access.
   - Billing Administrator: Manage billing only.
2. **Predefined roles**: Service-specific roles (e.g., Compute Engine instance admin).
3. **Custom roles**: Tailored permissions for least-privilege access.

Policy Types

- **Allow policies**: Grant permissions (inherited downward).
- **Deny policies**: Override allow policies (also inherited).

Service Accounts

- Special accounts for applications/VMs (not humans).
- Identified by email addresses.
- Use cryptographic keys instead of passwords.
- Can have IAM policies applied to them.

Cloud Identity

- Centralized user/group management via Google Admin console.
- Integrates with existing Active Directory/LDAP systems.
- Available in free and premium editions.

Interacting with Google Cloud

1. **Google Cloud Console**: Web-based GUI.
2. **Google Cloud SDK & Cloud Shell**:
   - Command-line tools (gcloud, bq).
   - Browser-based shell with persistent storage.
3. **APIs**:
   - Programmatic control of services.
   - Client libraries for multiple languages.
4. **Google Cloud Mobile App**:
   - Manage resources on-the-go.
   - View metrics and billing information.

Module 3: Virtual Machines and Networks in the Cloud

Virtual Private Cloud (VPC) Networking

What is a VPC?

- A secure, private cloud computing model hosted within Google's public cloud
- Combines public cloud scalability with private cloud data isolation
- Global in scope with subnets that can span multiple zones within a region

Key VPC Features:

- Connects Google Cloud resources to each other and the internet
- Built-in routing tables (no need to provision routers)
- Global distributed firewall (no need to provision separate firewalls)
    - Rules can be defined using network tags
- Subnet IP ranges can be expanded without affecting existing VMs

VPC Connectivity Options:

1. **VPC Peering**: Direct connection between two VPCs
2. **Shared VPC**: Central VPC shared across multiple projects with IAM controls

Compute Engine

Overview:

- Google's Infrastructure-as-a-Service (IaaS) offering
- Allows creation and management of virtual machines (VMs) on Google's infrastructure
- No upfront investments, scales to thousands of vCPUs

Key Features:

- Supports Linux and Windows Server images (both Google-provided and custom)
- Flexible configurations:
    - Predefined machine types
    - Custom machine types (select vCPUs and memory)
- Multiple creation methods:
    - Google Cloud Console
    - gcloud CLI
    - Compute Engine API

Pricing Models:

1. **On-demand VMs**: Per-second billing (1-minute minimum)
2. **Sustained-use discounts**: Automatic discounts for long-running instances (>25% of month)
3. **Committed-use discounts**: Up to 57% discount for 1-3 year commitments
4. **Preemptible/Spot VMs**:
    - Up to 90% cost savings
    - Can be terminated if resources are needed elsewhere
    - Ideal for batch jobs and fault-tolerant workloads

Storage Options:

- Zonal persistent disk (standard block storage)
- Regional persistent disk (replicated across zones)
- Local SSD (high performance, transient)
- Cloud Storage buckets (object storage)
- Filestore (high performance file storage)

Scaling and Load Balancing

Autoscaling:

- Automatically adds/removes VMs based on load metrics
- Works in conjunction with load balancing

Cloud Load Balancing:

- Fully distributed, software-defined managed service
- Types:
  - **Application Load Balancers (Layer 7)**
    - HTTP/HTTPS traffic
    - Advanced features like content-based routing
  - **Network Load Balancers (Layer 4)**
    - TCP/UDP traffic
    - Available in proxy and passthrough variants
- Features:
  - Cross-region load balancing
  - Automatic multi-region failover
  - No "pre-warming" needed for traffic spikes

Networking Services

Cloud DNS:

- Managed DNS service running on Google's infrastructure
- Low latency, high availability
- Programmable via console, CLI, or API

Cloud CDN:

- Content Delivery Network using Google's edge caching
- Benefits:
  - Lower latency for end users
  - Reduced load on origin servers
  - Cost savings

- Simple activation (single checkbox when using Application Load Balancer)

Network Connectivity Options

1. **Cloud VPN**:
   - Encrypted tunnel over the internet
   - Uses Cloud Router for dynamic routing (BGP)
2. **Direct Peering**:
   - Connect at Google's Points of Presence (100+ worldwide)
   - Not covered by SLA
3. **Carrier Peering**:
   - Connect through service provider partners
   - Access Google Workspace and Cloud services
4. **Dedicated Interconnect**:
   - Direct, private connection to Google
   - Up to 99.99% SLA
   - Can be backed by VPN for redundancy
5. **Partner Interconnect**:
   - Connection through supported service providers
   - Up to 99.99% SLA (Google portion only)
6. **Cross-Cloud Interconnect**:
   - High-bandwidth connection to other cloud providers
   - Supports multicloud strategies
   - Available in 10Gbps or 100Gbps

Module 4: Storage in the Cloud

Cloud Storage

Overview:

- Fully managed object storage service
- Ideal for binary large objects (BLOBs) like videos, images, backups
- Organized into buckets (globally unique names, regional/multi-regional)

Key Features:

- **Object Versioning**: Optional tracking of object history
- **Lifecycle Management**: Automatically transition/delete objects
- **Storage Classes**:
  - Standard (frequently accessed "hot" data)
  - Nearline (accessed ≤1/month)
  - Coldline (accessed ≤1/90 days)
  - Archive (accessed ≤1/year)

- **Autoclass**: Automatically optimizes storage class based on access patterns
- **Security**:
    - Server-side encryption by default
    - IAM and ACLs for access control
    - HTTPS/TLS for data in transit

Data Transfer Options:

1. Online transfer (gcloud, Console)
2. Storage Transfer Service (large batch transfers)
3. Transfer Appliance (petabyte-scale physical transfer)

Relational Database Options

Cloud SQL:

- Fully managed relational databases:
    - MySQL, PostgreSQL, SQL Server
- Features:
    - Automatic patches/updates
    - Managed backups (7 included)
    - Encryption at rest and in transit
    - Scales up to 128 vCPUs, 864GB RAM, 64TB storage
- Ideal for:
    - Traditional web applications
    - Existing applications using relational databases

Cloud Spanner:

- Horizontally scalable relational database
- Strong consistency at global scale
- SQL support with joins and secondary indexes
- Petabyte-scale capacity
- Powers Google's $80B business
- Ideal for:
    - Globally distributed applications
    - High-throughput transactional systems

NoSQL Database Options

Firestore:

- Flexible, scalable NoSQL database
- Document-based data model (collections > documents)
- Features:

    - Real-time updates
    - Offline support
    - Automatic multi-region replication
- Ideal for:
    - Mobile and web applications
    - Real-time collaborative apps

Bigtable:

- Petabyte-scale NoSQL database
- Powers core Google services (Search, Analytics, Maps)
- Features:
    - Low latency, high throughput
    - Ideal for time-series data
    - No SQL support or multi-row transactions
- Use cases:
    - IoT data
    - Financial analytics
    - AdTech platforms

Storage Option Comparison

Service	Type	Best For	Capacity	Key Features
Cloud Storage	Object	Large immutable blobs (>10MB)	PB (5TB/object)	Versioning, lifecycle management
Cloud SQL	Relational	Traditional web apps, OLTP	Up to 64TB	Full SQL, managed service
Spanner	Relational	Global-scale OLTP	Petabytes	Horizontal scaling, strong consistency
Firestore	NoSQL	Mobile/web apps, real-time data	Terabytes	Offline support, synchronization

Bigtable	NoSQL	Analytical workloads, time-series	Petabytes	High throughput, low latency

Module 5: Containers in the Cloud

Introduction to Containers

What are Containers?

- Lightweight, portable units that package application code with all dependencies
- Provide isolated environments with limited access to host system resources
- Start quickly (like processes) and scale efficiently
- Key benefits:
  - Portability: "Code once, run anywhere"
  - Efficiency: Higher density than VMs
  - Consistency: Same environment from development to production

Container Characteristics:

- Virtualize OS and dependencies (not hardware like VMs)
- Require container runtime and OS kernel support
- Combine benefits of PaaS (scaling) and IaaS (flexibility)

Scaling with Containers:

1. **Single Container Scaling**: Duplicate identical containers on a host
2. **Microservices Architecture**:
   - Decompose apps into specialized containers
   - Scale components independently
   - Connect via network interfaces

Kubernetes and Google Kubernetes Engine (GKE)

Kubernetes Fundamentals:

- Open-source container orchestration platform
- Manages containerized workloads across clusters
- Key concepts:
  - **Pods**: Smallest deployable units (1+ containers)
    - Share network namespace and storage volumes
  - **Deployments**: Manage replicated pods
  - **Services**: Stable endpoints for pod groups

    - **Nodes**: Compute instances running pods

Kubernetes Workflow:

1. Define desired state in configuration files (declarative approach)
2. Kubernetes control plane implements and maintains state
3. Features:
    - Automatic scaling
    - Rolling updates/rollbacks
    - Self-healing (restarts failed containers)
    - Load balancing

Google Kubernetes Engine (GKE):

- Managed Kubernetes service with two modes:
    1. **Autopilot (Recommended)**:
        - Fully managed infrastructure
        - Automatic scaling, security, and upgrades
        - Optimized for production workloads
    2. **Standard**:
        - User-managed node configuration
        - Greater control but more operational overhead

GKE Benefits:

- Integrated with Google Cloud services:
    - Cloud Load Balancing
    - IAM for access control
    - Cloud Monitoring/Logging
- Automated features:
    - Node auto-provisioning
    - Cluster upgrades
    - Node auto-repair

Module 6: Applications in the Cloud

Cloud Run (Serverless Containers)

Overview:

- Fully managed platform for stateless containers
- Built on Knative (Kubernetes-based open-source project)
- Key features:
    - Scales to zero when not in use
    - 100ms billing granularity

    - Supports any language/runtime (Linux x64 binaries)
    - Automatic HTTPS endpoints

Workflow Options:

1. **Container-Based**:
    - Build container image → Push to Artifact Registry → Deploy
2. **Source-Based**:
    - Deploy source code directly (uses Buildpacks)
    - Google manages containerization

Use Cases:

- Web applications/APIs
- Microservices
- Event processors (with Pub/Sub integration)
- Batch jobs (with HTTP triggers)

Pricing Model:

- Pay only while handling requests
- Charges for:
    - Execution time (per 100ms)
    - CPU/memory allocation
    - Network egress
- Free tier available

Cloud Run Functions (Event-Driven Functions)

Overview:

- Lightweight serverless functions service
- Event-driven execution model
- Key features:
    - Automatic scaling
    - 100ms billing granularity
    - Tight integration with Google Cloud services
    - Supports multiple languages (Node.js, Python, Go, etc.)

Trigger Types:

1. **HTTP**: Synchronous invocation via web requests
2. **Event-Based**: Asynchronous triggers from:
    - Cloud Storage (object changes)
    - Pub/Sub (messages)
    - Audit Logs

Use Cases:

- File processing (e.g., image thumbnailing)
- Data transformation pipelines
- Notification systems
- Lightweight API endpoints

Comparison: Cloud Run vs. Cloud Run Functions

Feature	Cloud Run	Cloud Run Functions
Unit of Deployment	Containers	Functions
Scaling	Request-based	Event-based
Cold Starts	Possible (scales to 0)	Possible (scales to 0)
Max Execution Time	60 minutes	60 minutes
Best For	Web apps, APIs	Event processing, triggers