Google Cloud

Professional Cloud Architect

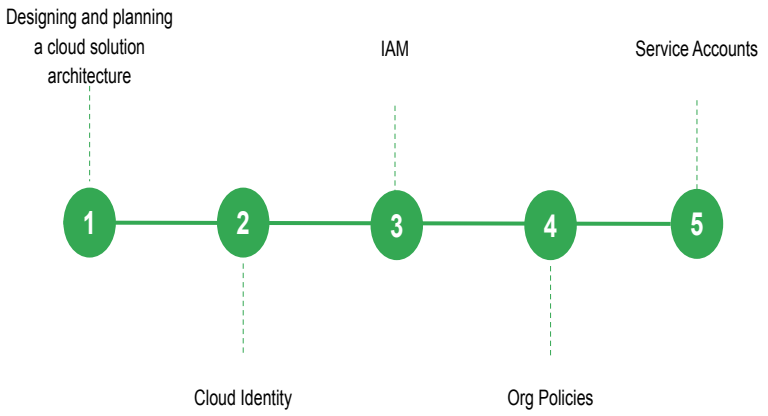
Preparing for Professional Cloud Architect Journey for AWS Professionals

Plan: show demo of each of 4 topics covered:

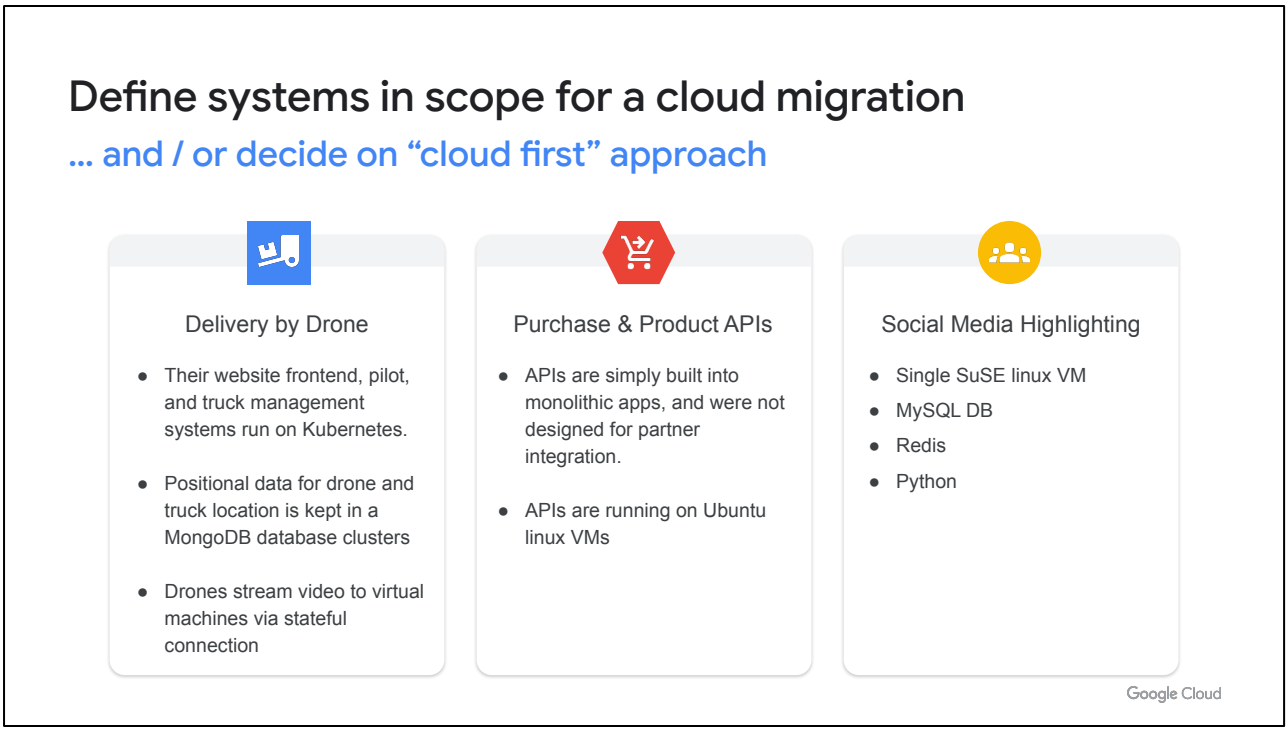
- Cloud Identity + GCDS + ADFS + 2 FA + …
- IAM (show the same UI on different levels: org, folder, project
- Org Policies
- Service Accounts (including "story" about SA keys.

60 mins should be just enough time. Skip most (all?) slides, except for diag questions

Session 2 topics



Designing and planning a cloud solution architecture



one of the first steps when planning for cloud migration would be to identify systems in scope for cloud migration, or decide on a "cloud first" approach.

The ones you see on the slide (with their components) come from PCA Workbook: https://docs.google.com/presentation/d/1pdREC42L-8T4QbbgqCfMDPCbIPgHvHwe
… but we’ll just treat those as examples. As we see, one of them is running Kubernetes and MongoDB with stateful connections for drone delivery; another is about APIs built into monoliths running on Ubuntu VMs; and third: a Social Media Highlighting system running on a single SuSE Linux VM using MySQL, Redis, and Python

Define business and technical requirements

Business requirements

- Easily scale to handle additional demand when needed and expand to more test markets.
- Streamline development for application modernization and new features/products
- Ensure that developers spend as much time on core business functionality as possible, and not have to worry about scalability wherever possible
- Let partners order directly via API
- Deploy a production version of the social media highlighting service and ensure no inappropriate content

Technical requirements

- Move to serverless services wherever possible
- Ensure that developers can deploy container-based workloads to testing and production environments in a highly scalable environment.
- Standardize on containers where possible, but also allow for existing virtualization infrastructure to run as-is without a re-write, so it can be slowly refactored over time
- Securely allow partner integration
- Stream IoT data from drones

Google Cloud

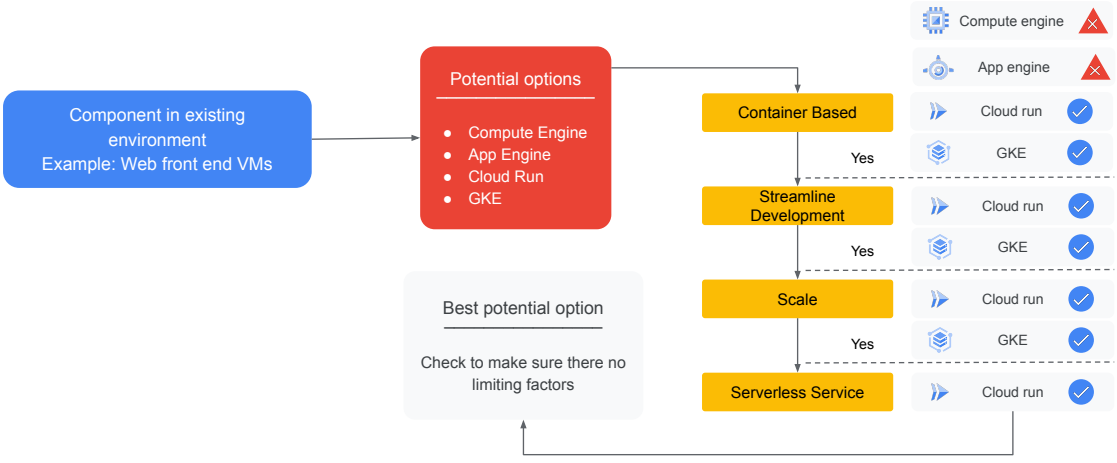
Once we identified those, we now need to also define both business and technical requirements so that we can choose optimal migration paths.

– again, based on Cymbal Direct (from PCA Workbook):
https://docs.google.com/presentation/d/1pdREC42L-8T4QbbgqCfMDPCbIPgHvHwe

For example, Business requirements focus on achieving scalability, expanding to new markets, streamlining development for modernization, maximizing developer focus on core business, enabling partner ordering via API, and ensuring production readiness/content safety. Technical requirements focus on moving to serverless services wherever possible, enabling scalable deployment of container-based workloads, standardizing on containers while allowing existing virtualization to run as-is for slow refactoring, securely allowing partner integration, and streaming IoT data

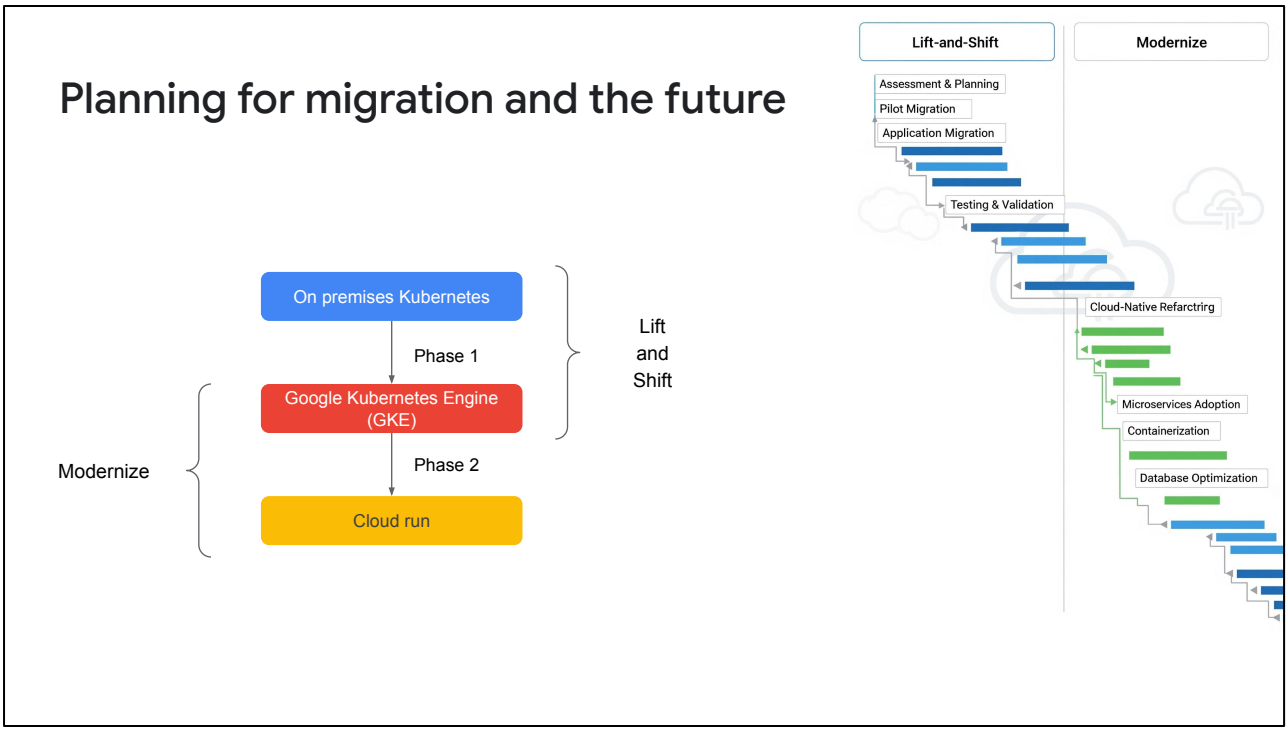
Mapping requirements to GCP resources

... through the lens of business & technical requirements



Google Cloud

Once we have chosen systems in scope, and we have a good understanding of our company's requirements, we can try mapping existing components (like Web front end VMs) to potential GCP options (Compute Engine, App Engine, Cloud Run, GKE)



we should now understand where we're starting from and where we're going, we can start planning.
It may be the case that we decide to migrate and modernize in one step, but also - just like here - it may be that we split those into two steps, where we might perform a relatively straightforward lift-and-shift migration from on-premises Kubernetes cluster to GKE, and only then - perform modernization steps to finally land on Cloud Run.

ASK: why would someone want to split it this way?
-> redure risks, gain confidence

Migration guide & best practices

- Types of migrations and their use-cases
  - For example, “If the current app isn't meeting your goals—for example, you don't want to maintain it, it's too costly to migrate using one of the previously mentioned approaches, or it's not supported on Google Cloud—you can do a rebuild migration.”
- Building inventory of workloads in scope
  - … along with their dependencies!
- Best practices for validating a migration plan.

Google Cloud

I’m aware that was a bit of oversimplification, so let’s have a look at a bit more detailed view and Google’s best practices around this whole process https://cloud.google.com/architecture/migration-to-gcp-getting-started

Diagnostic Question Discussion

You work for a large financial institution that is planning a multi-phase migration of its on-premises workloads to Google Cloud. The migration involves sensitive customer data and requires high availability and disaster recovery capabilities. You want to design a cloud solution architecture that meets these requirements while minimizing costs and ensuring compliance with industry regulations.

What should you do?

A. Migrate all workloads to a single region in Google Cloud to simplify management and reduce latency. Implement basic security measures, such as firewalls and intrusion detection systems, to protect against common threats. Use a combination of managed services and self-managed services to balance cost and flexibility.

B. Design a multi-zone architecture within a single region to reduce costs while maintaining high availability. Implement security measures based on industry best practices, such as using strong passwords and multi-factor authentication. Use primarily self-managed services to have greater control over the environment.

C. Design a multi-region architecture with active-passive failover for high availability and disaster recovery. Implement appropriate security measures, such as data encryption at rest and in transit, to protect sensitive customer data. Use managed services whenever possible to reduce operational overhead and costs.

D. Defer considerations for high availability and disaster recovery until a later phase to minimize initial costs and complexity. Implement security measures as needed based on the sensitivity of the data. Use primarily open-source tools and technologies to minimize licensing costs.

Quickly eliminate A, B and D

C. Easy

Diagnostic Question Discussion

You work for a large financial institution that is planning a multi-phase migration of its on-premises workloads to Google Cloud. The migration involves sensitive customer data and requires high availability and disaster recovery capabilities. You want to design a cloud solution architecture that meets these requirements while minimizing costs and ensuring compliance with industry regulations.

What should you do?

A. Migrate all workloads to a single region in Google Cloud to simplify management and reduce latency. Implement basic security measures, such as firewalls and intrusion detection systems, to protect against common threats. Use a combination of managed services and self-managed services to balance cost and flexibility.

B. Design a multi-zone architecture within a single region to reduce costs while maintaining high availability. Implement security measures based on industry best practices, such as using strong passwords and multi-factor authentication. Use primarily self-managed services to have greater control over the environment.

C. Design a multi-region architecture with active-passive failover for high availability and disaster recovery. Implement appropriate security measures, such as data encryption at rest and in transit, to protect sensitive customer data. Use managed services whenever possible to reduce operational overhead and costs.

D. Defer considerations for high availability and disaster recovery until a later phase to minimize initial costs and complexity. Implement security measures as needed based on the sensitivity of the data. Use primarily open-source tools and technologies to minimize licensing costs.

Google Cloud

Quickly eliminate A, B and D

C. Easy

Diagnostic Question Discussion

You are a Professional Cloud Architect working with a large retail customer that has a monolithic e-commerce application hosted on-premises. They are experiencing challenges with scalability and performance, especially during peak shopping seasons. They want to migrate this application to Google Cloud and modernize it to be more resilient, scalable, and cost-effective.

Your goal is to design a solution that meets their requirements.

What should you do? (choose two)

A. Migrate the application to Google Cloud using a phased approach, starting with a lift-and-shift migration and gradually modernizing components.

B. Deploy the entire application to a single, large Compute Engine instance to ensure resource availability and minimize management overhead.

C. Continue running the application on-premises and use Cloud CDN and Cloud Load Balancing to enhance performance and scalability.

D. Decompose the monolithic application into microservices and leverage managed services like Google Kubernetes Engine (GKE) and Cloud SQL.

E. Refactor the entire application to serverless architecture using Cloud Functions and Cloud Run to minimize operational overhead and maximize cost savings.

Eliminate:
B - obvious
E - it’s a monolith, so such an approach might be impossible or extremely difficult + not cost-effective.
C - it might help, but it will not be more scalable or cost-effective

Diagnostic Question Discussion

You are a Professional Cloud Architect working with a large retail customer that has a monolithic e-commerce application hosted on-premises. They are experiencing challenges with scalability and performance, especially during peak shopping seasons. They want to migrate this application to Google Cloud and modernize it to be more resilient, scalable, and cost-effective.

Your goal is to design a solution that meets their requirements.

What should you do? (choose two)

A. **Migrate the application to Google Cloud using a phased approach, starting with a lift-and-shift migration and gradually modernizing components.**

B. Deploy the entire application to a single, large Compute Engine instance to ensure resource availability and minimize management overhead.

C. Continue running the application on-premises and use Cloud CDN and Cloud Load Balancing to enhance performance and scalability.

D. **Decompose the monolithic application into microservices and leverage managed services like Google Kubernetes Engine (GKE) and Cloud SQL.**

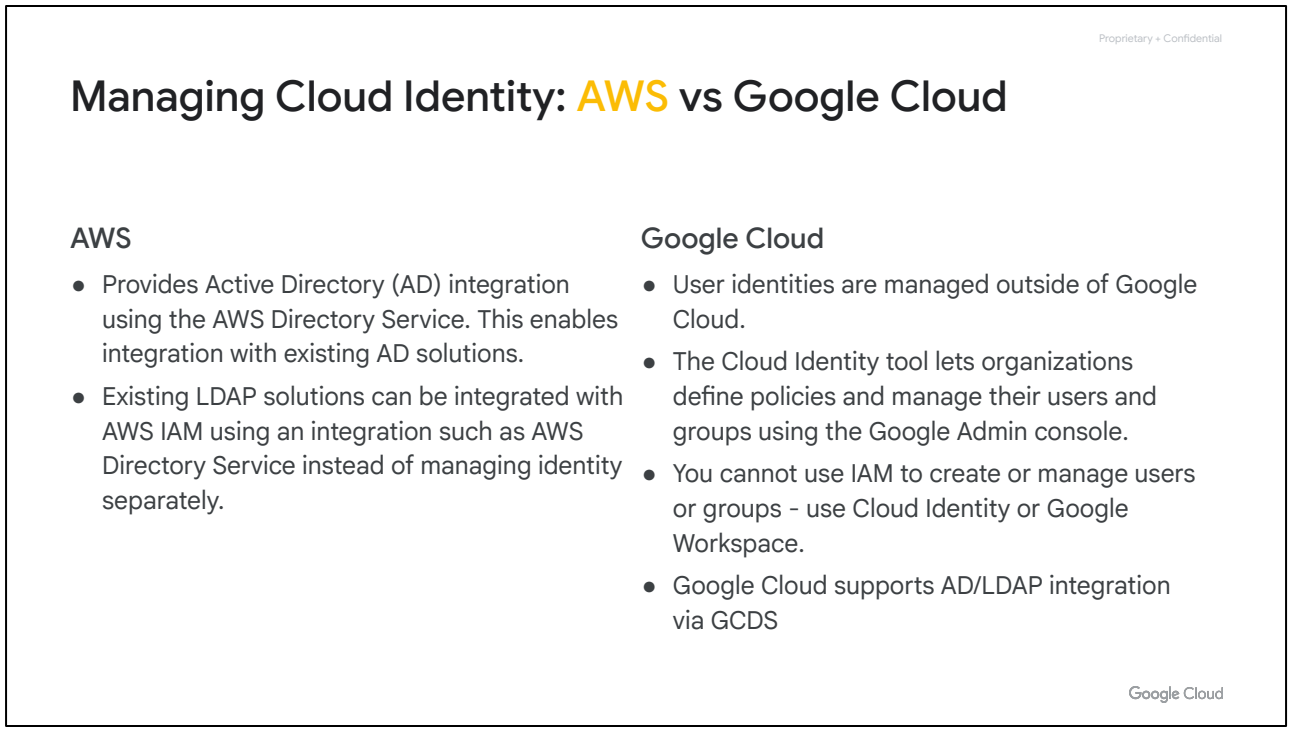
E. Refactor the entire application to serverless architecture using Cloud Functions and Cloud Run to minimize operational overhead and maximize cost savings.

Google Cloud

Eliminate:
B - obvious
E - it’s a monolith, so such an approach might be impossible or extremely difficult + not cost-effective.
C - it might help, but it will not be more scalable or cost-effective

* **Option C is incorrect** because deploying a large monolithic application to a single instance does not address scalability or resilience and can lead to resource contention and management challenges.
* **Option D is incorrect** because while CDNs and Load Balancers can improve performance, they don't address the underlying issues of a monolithic architecture and don't leverage the full benefits of cloud migration.
* **Option E is incorrect** because while serverless can be beneficial, refactoring the entire application to serverless might not be the most practical or efficient approach for an initial migration and modernization effort.

Cloud Identity



the differences in how identity is managed between the two platforms. Let's explore the difference.

AWS

- AWS provides Active Directory (AD) integration using the AWS Directory Service. This enables integration with existing AD solutions.
- Typically organizations will have an existing LDAP solution, and will choose to integrate the LDAP solution with AWS IAM using an integration such as AWS Directory Service instead of managing identity separately.

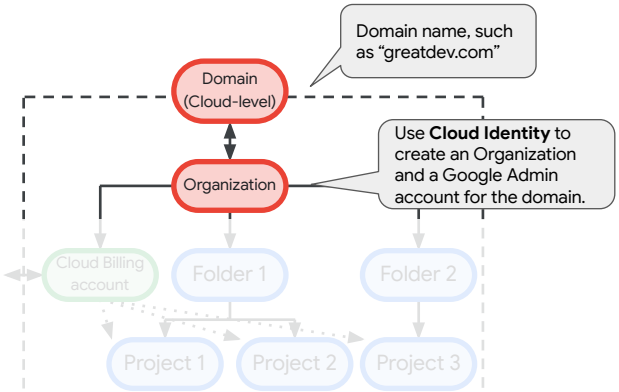
Google Cloud

- For Google Cloud, user identities are managed outside of Google Cloud. For example, Google Workspace or Gmail accounts can be used to manage identities.
- With a tool called Cloud Identity, organizations can define policies and manage their users and groups using the Google Admin console.

It’s important to note that you cannot use IAM to create or manage your users or groups in Google Cloud. Instead, you can use Cloud Identity or Google Workspace to create and manage users.

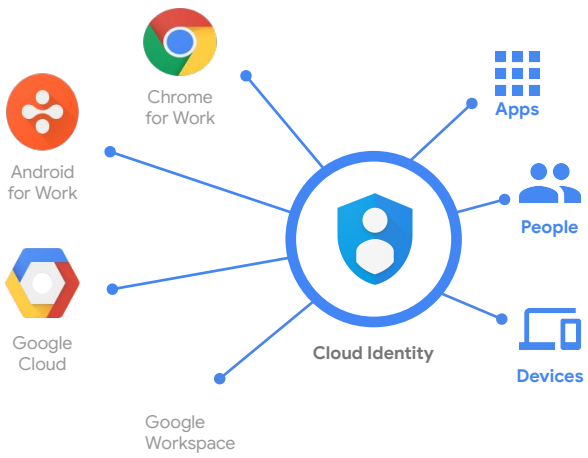
How is an Organization Created?

- **Cloud Identity** manages the users and groups that have access to Google Cloud
  - Federated identities from Google Workspace and other identity providers, such as Active Directory and Azure Active Directory
    - Bring existing users/groups into Cloud Identity
    - Use Identity and Access Management (IAM) to manage access to Google Cloud resources



GCP vs AWS: users in GCP are NOT created in IAM, or anywhere within the cloud environment (except for SAs)

What is Cloud Identity?



- Cloud Identity is an Identity as a Service (IDaaS) solution that **allows you to centrally manage users and groups** who can access GCP and Google Workspace cloud resources
- It is the same identity service that powers Google Workspace and can also be used as IdP for 3rd party applications (supports SAML and LDAP applications)

Main difference around identities::

Google separates the **identity provider (the "who")** from the **authorization service (the "can do")**, while AWS bundles them into a single service.

- **Google's Approach:** Uses **Google Cloud Identity** (as the identity provider) in combination with **Google Cloud IAM** (as the authorization service).
- **AWS's Approach:** Uses **AWS Identity and Access Management (IAM)**, which handles *both* identity and authorization.

TL;DR / Purpose of the slide:

- Overview of Cloud Identity

Key points:

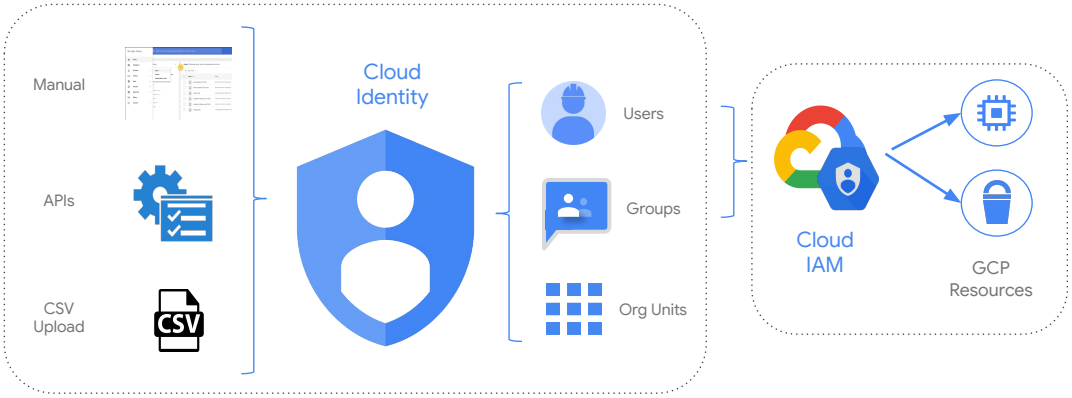
- Cloud Identity is the **same IDaaS** solution that **powers Google Workspace.** It provides free identity services for users.
- When you **migrate to Cloud Identity,** you **create a free account** for each of your users and you can manage all users from the Google Admin console.
- It enables and accelerates the use of cloud-centric application models, but offers capabilities to meet organizations where they are with their on-premise IAM systems and apps.
- Setting up Cloud Identity is a **prerequisite** to **onboarding your organization**

- **onto GCP.**
    - Upon setting up your Cloud Identity instance you will be asked to **add a domain that you own** (typically the company's main domain)
    - You will be asked to **validate that you own this domain** - most typically done through adding an automatically generated TXT record in your DNS records.
    - Once this is validated, a **GCP organization will be created** with the **same name as your domain** and you can begin using GCP.

Probing questions (optional):

- None

Members in GCP do not come from GCP!



Users and groups created in Cloud Identity are the **Google Identities** that *can* be assigned **IAM roles** in the GCP console

The **Cloud Identity roles** only manage aspects of Cloud Identity such as user/group management, **and are different from GCP roles** which manage permissions to cloud resources

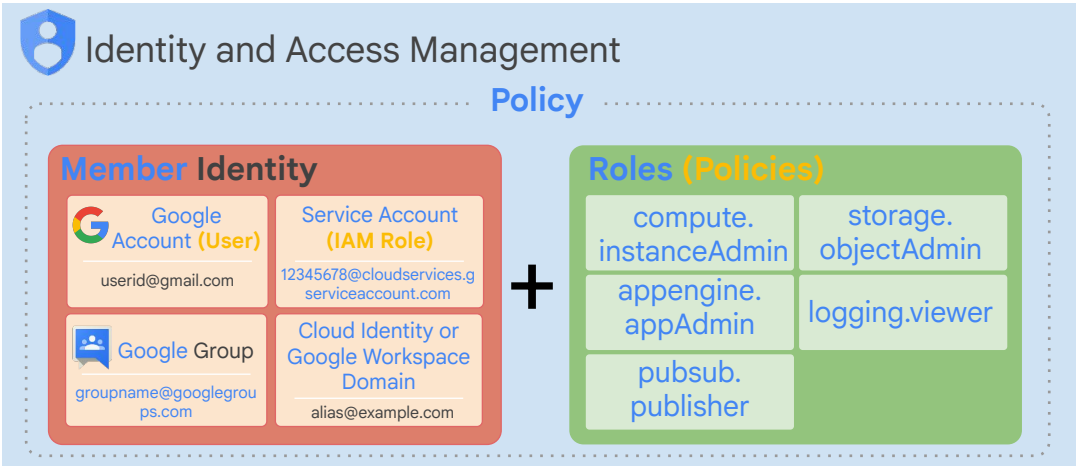
TL;DR / Purpose of the slide:

- Show how Cloud Identity connects with GCP

Key points:

- **Variety of ways** in which **identities** can be **imported** (will be detaileds later)
- **Users and groups** are the identities that will be **used by Cloud IAM** to grant them **access** to particular **GCP resources** as you see in this diagram.
- The **Cloud Identity roles** only manage aspects of Cloud Identity such as user/group management, and **are different from GCP roles** which manage permissions to cloud resources

Members in GCP do not come from GCP!



Note: You *cannot* use IAM to create or manage your users or groups.
You can use IAM or IAM Identity Center to manage users and groups.

*** with one exception: Service Accounts
Source: Architecting with Compute Engine slides.

Speaker Notes:

Difference with AWS: You can put a service account in a Google Group - while you cannot put an IAM role in a group

There are five different types of members: Google Accounts, Service Accounts, Google Groups, Google Workspace domains, and Cloud Identity domains.

A Google account represents a developer, an administrator, or any other person who interacts with Google Cloud. Any email address that is associated with a Google account can be an identity, including gmail.com or other domains. New users can sign up for a Google account by going to the Google account signup page, without receiving mail through Gmail.

A service account is an account that belongs to your application instead of to an individual end user. When you run code that is hosted on Google Cloud, you specify the account that the code should run as. You can create as many service accounts as needed to represent the different logical components of your application.

A Google group is a named collection of Google accounts and service accounts. Every group has a unique email address that is associated with the group. Google groups are a convenient way to apply an access policy to a collection of users. You can grant and change access controls for a whole group at once instead of granting or changing access controls one-at-a-time for individual users or service accounts.

A Workspace domain represents a virtual group of all the Google accounts that have been created in an organization's Workspace account. Workspace domains represent your organization's internet domain name, such as example.com, and when you add a user to your Workspace domain, a new Google account is created for the user inside this virtual group, such as username@example.com.

Google Cloud customers who are not Workspace customers can get these same capabilities through Cloud Identity. Cloud Identity lets you manage users and groups using the Google Admin Console, but you do not pay for or receive Workspace's collaboration products such as Gmail, Docs, Drive, and Calendar. Refer to the documentation for more information on Cloud Identity Editions.

Now it's important to note that you cannot use IAM to create or manage your users or groups. Instead, you can use Cloud Identity or Workspace to create and manage users.

Cloud Identity: central user and group management

Google's Identity as a Service (IDaaS) solution

- Users and groups that are to be added to Google Cloud need accounts in Cloud Identity

Manually creating user accounts

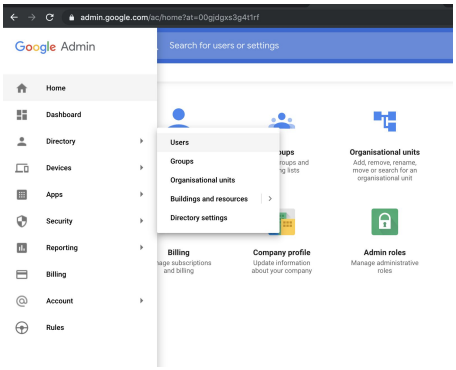
- Add users individually using the Google Admin console
- Add several users at once by uploading their names in a CSV file

Options for large organizations

- Use Google Cloud Directory Sync to synchronize user data in your existing LDAP directory with your Google account
- Use the Admin SDK Directory API to provision a large number of users with data from your existing LDAP directory, such as Microsoft® Active Directory®
  - Requires programming

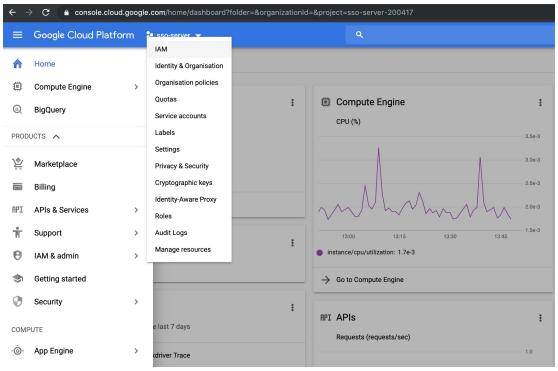
*Cloud Identity has advanced management features not covered in this module, e.g., mobile app management, 2-Step verification, etc.

Two consoles for administration



Cloud Identity (`admin.google.com`)

Managing Users, Groups, and Authentication settings



GCP (`console.cloud.google.com`)

Roles & Authorization for GCP

TL;DR / Purpose of the slide:

- 2 Consoles for administration

Key points:

- Left: Google Admin to manage users, groups, authentication methods, security settings
- Right: Cloud Console to manage roles and authorization for GCP specifically

Probing questions (optional):

- None

Two key admin functions

	(CI) Super Admin	(Cloud IAM) Org. Admin
Role	**GCP Org. Admin by default**	It can add/assume any other IAM roles
Manages	**User/group account lifecycle** and Org's security settings	**IAM policies** and Resource Manager hierarchy
Delegates	**GCP Org. Admin role** and CI admin roles	**GCP IAM roles** to users and groups
Managed in	**Admin console**	**GCP console**
Visibility	**Cloud Identity and GCP environments**	**GCP environment**

TL;DR / Purpose of the slide:

- Clarify Super Admin vs Org Admin roles

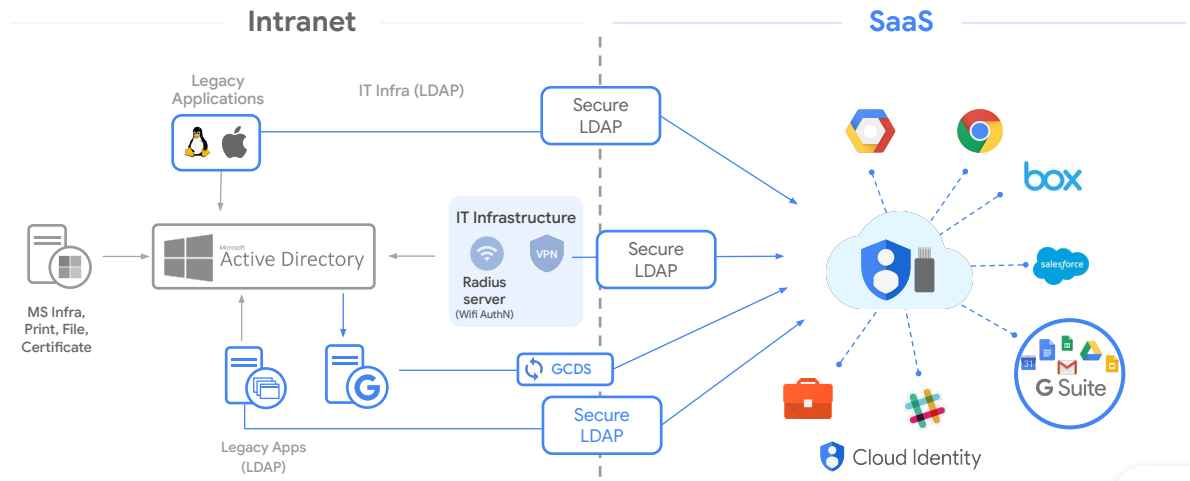
Key points:

- Super Admin is also an Org Admin by default.
- Super Admin has visibility on both platforms

Probing questions (optional):

- None

Cloud Identity as an identity provider: Typical architecture



TL;DR / Purpose of the slide:

- Show a typical architecture where Cloud Identity is the Identity Provider

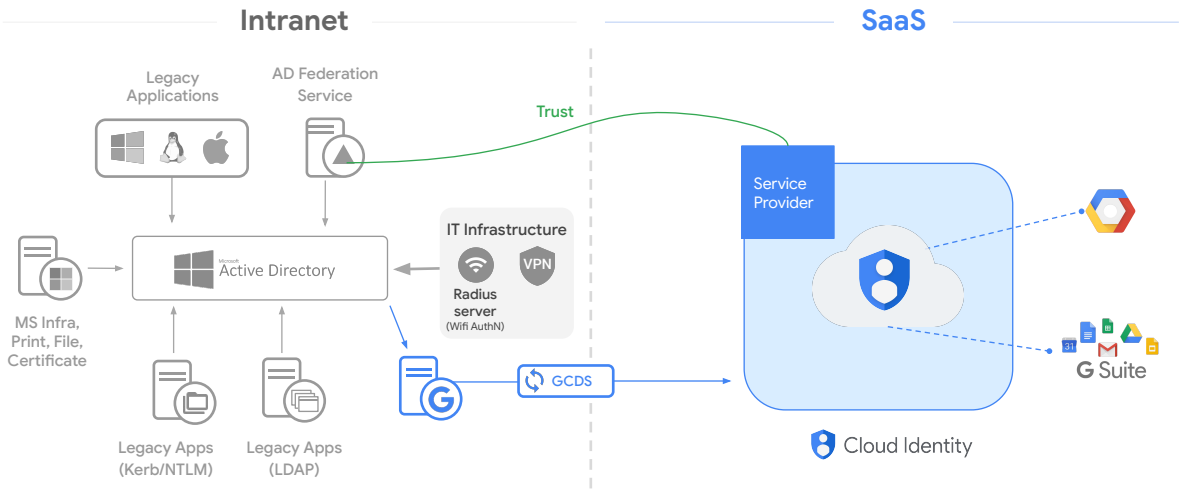
Key points:

- We will address GCDS later on
- User will authenticate with Google and leverage Google authentication and security
- 3rd party apps (Workday, Salesforce, etc..) will be connected via SAML or Secure LDAP
- Secure LDAP requires CI Premium

Probing questions (optional):

- None

3rd party as an identity provider: Typical architecture



TL;DR / Purpose of the slide:

- Show a typical architecture where a 3rd party is the Identity Provider

Key points:

- We will address GCDS later on
- User will authenticate with the 3rd party IdP (AD **Federation** Service in this diagram, but it can also be Azure AD, Ping, OKTA, etc..)
- **Not leverage Google authentication and advanced security**
- Leverage existing Identity Management and authentication systems
- gcloud, gsutil, and bq, use SAML 2.0-based SSO for browser-based authentication as well

Probing questions (optional):

- None

TL;DR / Purpose of the slide:

- Help customer decide on best provisioning options for them. We recommend GCDS.

Key points:

- **GCDS is recommended** for user provisioning, but other options are available.
- In case customer is using **Azure AD**, there is also the possible for them to configure auto-provisioning from Azure AD to Cloud Identity
- You can mention also **Google Workspace Terraform provider** in case customer has a very simple setup and is really interested in Terraform (https://github.com/DeviaVir/terraform-provider-gsuite)

Probing questions (optional):

- Which method do you choose?

Diagnostic Question Discussion

Your company wants to start using Google Cloud resources but wants to retain their on-premises Active Directory domain controller for identity management.

What should you do?

A. Use the Admin Directory API to authenticate against the Active Directory domain controller.

B. Use Google Cloud Directory Sync to synchronize Active Directory usernames with cloud identities and configure SAML SSO.

C. Use Cloud Identity-Aware Proxy configured to use the on-premises Active Directory domain controller as an identity provider.

D. Use Compute Engine to create an Active Directory (AD) domain controller that is a replica of the on-premises AD domain controller using Google Cloud Directory Sync.

B

https://cloud.google.com/architecture/identity/federating-gcp-with-active-directory-introduction

Diagnostic Question Discussion

Your company wants to start using Google Cloud resources but wants to retain their on-premises Active Directory domain controller for identity management.

What should you do?

A. Use the Admin Directory API to authenticate against the Active Directory domain controller.

B. Use Google Cloud Directory Sync to synchronize Active Directory usernames with cloud identities and configure SAML SSO.

C. Use Cloud Identity-Aware Proxy configured to use the on-premises Active Directory domain controller as an identity provider.

D. Use Compute Engine to create an Active Directory (AD) domain controller that is a replica of the on-premises AD domain controller using Google Cloud Directory Sync.

https://cloud.google.com/architecture/identity/federating-gcp-with-active-directory-introduction

B

https://cloud.google.com/architecture/identity/federating-gcp-with-active-directory-introduction

IAM

IAM: GCP vs AWS

Feature	Google Cloud IAM	AWS IAM
Permission Model	**Role-Based.** Bind a *Member* to a ***Role*** **on a Resource**.	**Policy-Based.** Attach a ***Policy*** **to a Principal** (User/Role).
Hierarchy	**Full Inheritance.** Permissions flow down from Org > Folder > Project.	**No Inheritance.** Permissions are scoped to an Account. (OUs provide guardrails, not grants).
"Role" means...	A **set of permissions** (like compute.viewer).	An **identity** that can be assumed (like EC2-S3-Access-Role).
"Policy" means...	The *entire collection of bindings* on one resource.	A JSON document listing Allow/Deny statements.
Day-to-Day Task	"I will grant user@a.com the Storage Viewer role **on Project B**."	"I will attach the S3ReadOnly policy **to the 'dev-group'**."

Google Cloud

The main difference is this:

- **GCP IAM** is about: "Who can do what **on which resource**?" The policy is part of the resource.
- **AWS IAM** is about: "What can **this identity** do?" The policy is attached to the identity.

In GCP, You grant permissions by **binding** a "member" (like a user or group) to a "role" (like "Compute Viewer") *directly onto a resource* (like a Project or a Storage Bucket). **Full Inheritance** GCP is built based on **resource hierarchy;** Permissions are **inherited** down this tree.

Also, The two platforms use the word "Role" to mean completely different things.

- **Google Cloud IAM: A "Role" is just a set of permissions.** It is **not an identity**.It's conceptually similar to an **AWS Managed Policy**.

AWS IAM: A "Role" is an identity to be assumed. An AWS Role is a **principal** (an identity), just like a User.

Comparing IAM: AWS vs GCP

AWS

- Offers built-in and custom **policies**
- The collection of permissions and resources those permissions apply is a policy.
- The account root can fully administer roles and policies unless otherwise restricted by a Service Control Policy.
- Policies can be attached to users, groups of users, or roles.

Google Cloud

- Offers built-in and custom **roles**
- It has sets of predefined roles, and with definitions of where those roles can be applied.
- It grants granular access to specific Google Cloud resources and prevents unwanted access to other resources.
- Roles are collections of permissions.

Let's take a moment to explore the differences between the Resource Manager role in the AWS and Google Cloud resource hierarchies.

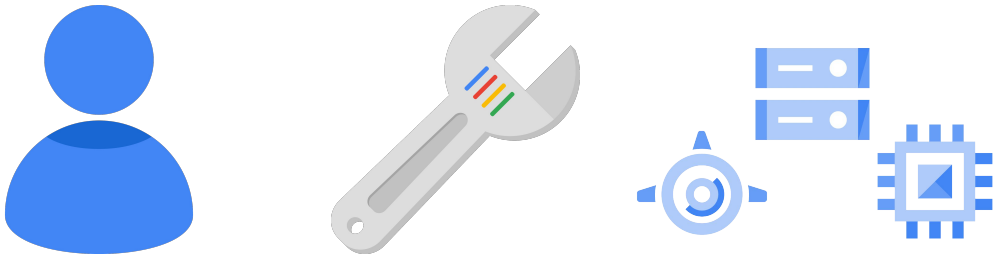
AWS

- AWS offers built-in and custom policies; Google Cloud offers built-in and custom roles.
- AWS refers to the collection of permissions and resources those permissions apply to as a policy.
- AWS lets the account root fully administer roles and policies unless otherwise restricted by a Service Control Policy.
- These policies can be attached to users, groups of users, or roles.

Google Cloud

- Google Cloud offers sets of predefined roles, and they define where those roles can be applied. This provides granular access to specific Google Cloud resources and prevents unwanted access to other resources. These roles are collections of permissions because, to do any meaningful operations, you usually need more than one permission.

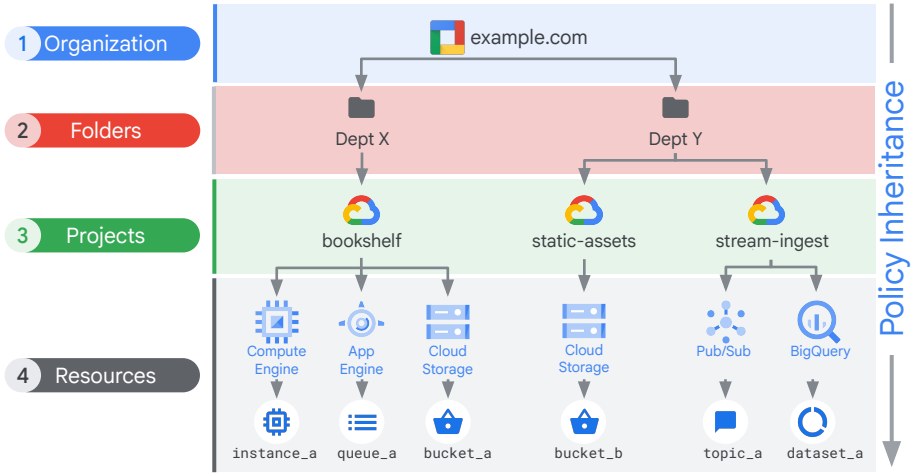
Identity and Access Management



Who | can do what | on which resource

IAM GCP vs AWS: in AWS, a policy is a set of actions for identified resources, and then attached to IAM identities.
in GCP, principals (users, groups) are assigned roles on specific resources, so IAM policy in GCP associates roles and principals together
– SHOW in GCP

IAM inheritance



Assign permissions to multiple accounts with IAM Identity Center and Permission Sets.

AWS does not support policy inheritance, but you can use Service Control Policies (SCPs) to limit granted permissions and AWS supports SCP inheritance within an organization.

Unlike AWS, in GCP IAM roles granted at high level (org / folder) are inherited by all resources underneath.

IAM objects

Organization

AWS Organizations

Folders

OUs in Organizations

Projects

Accounts

Resources

Roles

Policies (intra-account)
Policy Sets (across accounts)

Members

Users, Groups, Roles

Source: Architecting with Compute Engine slides.

IAM is composed of different objects as shown on this slide. We are going to cover each of these in this module. To get a better understanding of where these fit in, let's look at IAM policies, and the IAM resource hierarchy.

IAM policies

- A policy consists of a list of bindings.
- A binding binds a list of members to a role.

AWS does not have an equivalent, but you can use Service or Resource Summary views to see what permissions have been assigned

In AWS a policy is a JSON document that describes permissions and a role is a wrapper for policies that can be assumed by users or services.
In GCP a policy also grants permissions but includes a resource, the permissions and principles.

Source: Architecting with Compute Engine slides.

A policy consists of a list of bindings.

A binding binds a list of members to a role, where the members can be user accounts, Google groups, Google domains, and service accounts. A role is a named list of permissions defined by IAM. Let's revisit the IAM resource hierarchy.

Proprietary + Confidential

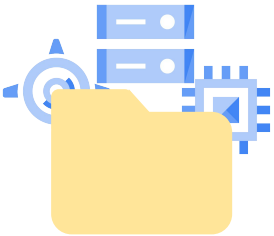
Three types of IAM roles (IAM policies)

Basic

Simply do NOT use those!

Predefined (AWS managed policies)

Custom (customer managed policies)



Google Cloud

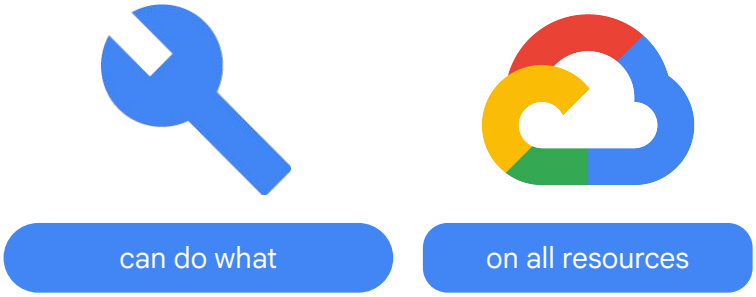
Unlike AWS, GCP (and Azure) employ a role-based access control (RBAC) model,

GCP separates the permissions from the resources in the document named “role.”. In essence, a role is just a group of single permissions, which you then grant to a principal at a specific level of org hierarchy (SHOW)

Source: Architecting with Compute Engine slides.

There are three types of roles in IAM: basic roles, predefined roles, and custom roles.

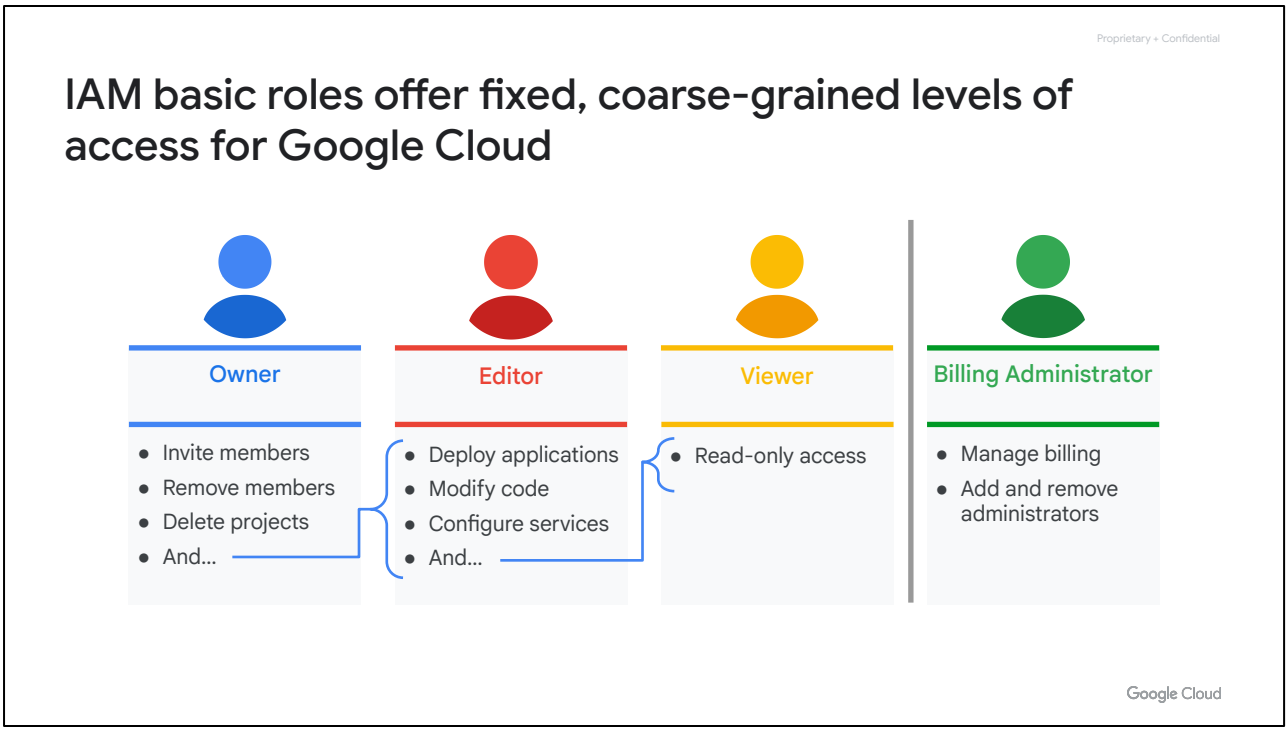
IAM basic roles apply across all Google Cloud services in a project



Source: Architecting with Compute Engine slides.

Basic roles are the original roles that were available in the Cloud Console, but they are broad.

You apply them to a Google Cloud project, and they affect all resources in that project.



Source: Architecting with Compute Engine slides.

In other words, IAM basic roles offer fixed, coarse-grained levels of access.

The basic roles are the Owner, Editor, and Viewer roles.

- The owner has full administrative access. This includes the ability to add and remove members and delete projects.
- The editor role has modify and delete access. This allows a developer to deploy applications and modify or configure its resources.
- The viewer role has read-only access.

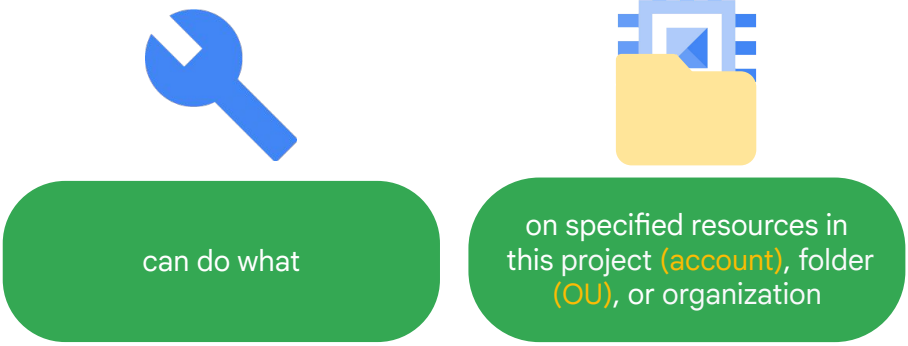
All of these roles are concentric; that is, the Owner role includes the permissions of the Editor role, and the Editor role includes the permissions of the Viewer role.

There is also a billing administrator role to manage billing and add or remove administrators without the right to change the resources in the project.

Each project can have multiple owners, editors, viewers, and billing administrators.

IAM predefined roles apply to a particular Google Cloud service in a project

Google IAM predefined roles = AWS managed policies

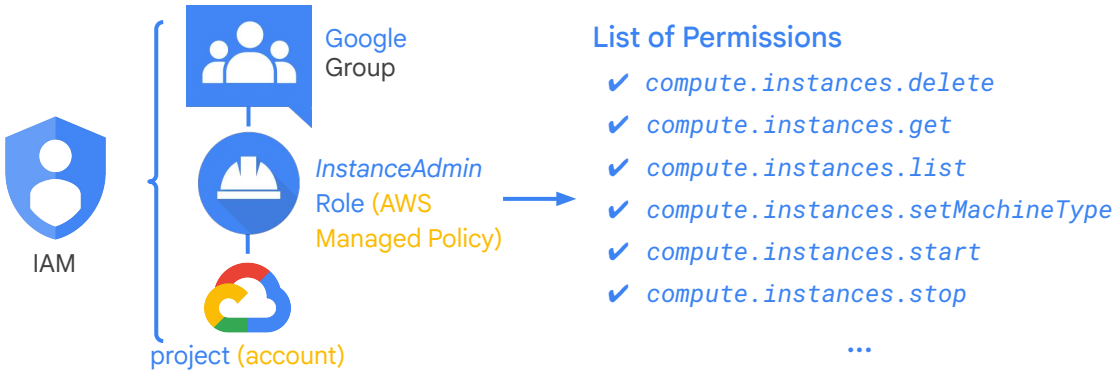


Source: Architecting with Compute Engine slides.

Google Cloud services offers their own sets of predefined roles, and they define where those roles can be applied. This provides members with granular access to specific Google Cloud resources and prevents unwanted access to other resources.

These roles are *collections* of permissions because, to do any meaningful operations, you usually need more than one permission.

IAM predefined roles offer more fine-grained permissions on particular services



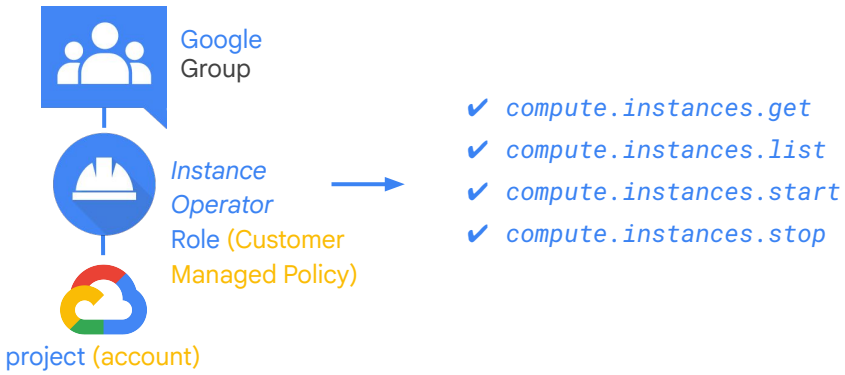
Source: Architecting with Compute Engine slides.

For example, as shown here, a group of users is granted the InstanceAdmin role on project_a. This provides the users of that group with all the Compute Engine permissions listed on the right and more. Grouping these permissions into a role makes them easier to manage. The permissions themselves are classes and methods in the APIs.

For example, compute.instances.start can be broken down into the service, resource, and verb that mean that this permission is used to start a stopped Compute Engine instance.

These permissions usually align with the action's corresponding REST API.

IAM custom roles let you define a precise set of permissions



Source: Architecting with Compute Engine slides.

Speaker Notes:

Custom Roles can be added at the Org level and Project Level but not at the folder level.

That's what custom roles permit. A lot of companies use the "least-privilege" model, in which each person in your organization is given the minimal amount of privilege needed to do their job.

Let's say you want to define an "Instance Operator" role to allow some users to start and stop Compute Engine virtual machines, but not reconfigure them. Custom roles allow you to do that.

For a demo on how to create a custom role in Google Cloud, refer to this video.

IAM conditions

- IAM Conditions are used to enforce conditional ABAC for cloud resources
- With IAM Conditions, you can choose to grant resource access to identities (members) only if configured conditions are met
- Conditions are specified in the role bindings of a resource's IAM policy. When a condition exists, the access request is only granted if the condition expression evaluates to true.

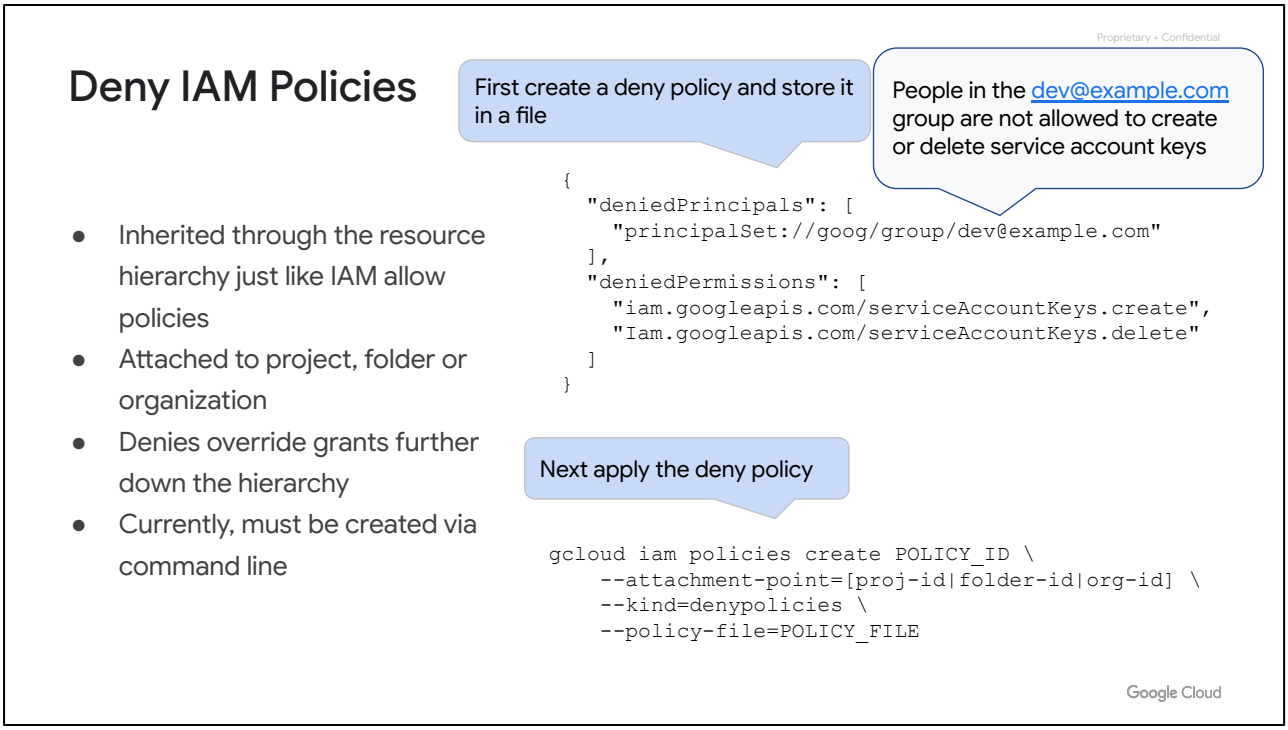
Finally, let's look at IAM conditions.

In AWS and in Google Cloud, IAM Conditions are used to enforce conditional attribute-based access control (ABAC) for cloud resources.

With IAM Conditions, you can choose to grant resource access to identities (members) only if configured conditions are met.

For example, this could be done to configure temporary access for users in the event of a production issue or to limit access to resources only for employees making requests from your corporate office.

Conditions are specified in the role bindings of a resource's IAM policy. When a condition exists, the access request is only granted if the condition expression evaluates to true. Each condition expression is defined as a set of logic statements allowing you to specify one or more attributes to check.



Cloud providers prevent breaches by putting different guardrails in place. One of those in GCP (with no AWS equivalent) is a Deny IAM policy, which is a set of rules that determines what a principal is **denied** access to. You can use a Deny policy to, for example, prevent making resources public.

Deny policies
https://cloud.google.com/iam/docs/deny-overview

IAM policy pattern example

inheritance

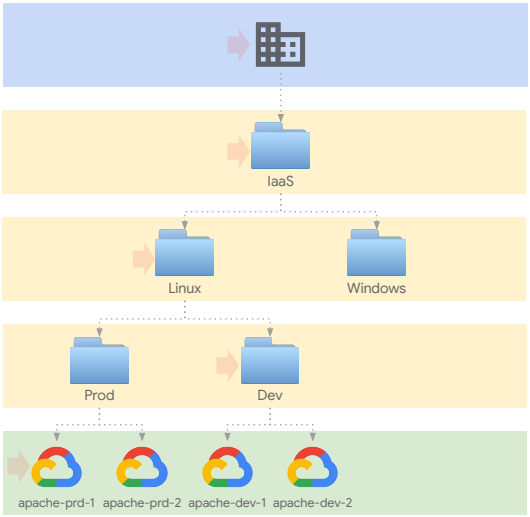
`roles/browser` → `domain:example.org`
all domain users should be able to see the hierarchy

`roles/viewer` → `group:first-lvl-support@`
support users should be able to view logs and VMs

`roles/compute.admin` → `group:linux-os@`
instance admins should manage resources

`roles/logging.logViewer` → `group:app-team-1@`
app admins should view logs in dev

`roles/storage.admin` → `serviceAccount:app1@`
ad-hoc permissions on project or single resource



This starts introducing patterns used for the actual org design that will come later on. Don't read too much into the individual roles, they are just a reasonably sensible illustration on why having different touch points for IAM is useful.

Diagnostic Question Discussion

You are responsible for the Google Cloud environment in your company. Multiple departments need access to their own projects, and the members within each department will have the same project responsibilities. You want to structure your environment for minimal maintenance and follow Google-recommended practices.

What should you do?

A. Create a Google Group per department and add all department members to their respective groups. Create a folder per department and grant the respective group the required IAM permissions at the folder level. Add the projects under the respective folders.

B. Grant all department members the required IAM permissions for their respective projects.

C. Create the folder per department and grant the respective members of the department the required IAM permissions at the folder level. Structure all projects for each department under the respective folders.

D. Create a Google Group per department and add all department members to their respective groups. Grant each group the required IAM permissions for their respective projects.

A

Diagnostic Question Discussion

You are responsible for the Google Cloud environment in your company. Multiple departments need access to their own projects, and the members within each department will have the same project responsibilities. You want to structure your environment for minimal maintenance and follow Google-recommended practices.

What should you do?

A. **Create a Google Group per department and add all department members to their respective groups. Create a folder per department and grant the respective group the required IAM permissions at the folder level. Add the projects under the respective folders.**

B. Grant all department members the required IAM permissions for their respective projects.

C. Create the folder per department and grant the respective members of the department the required IAM permissions at the folder level. Structure all projects for each department under the respective folders.

D. Create a Google Group per department and add all department members to their respective groups. Grant each group the required IAM permissions for their respective projects.

A

Diagnostic Question Discussion

You have a user in your environment that has read-only access to four Cloud Storage buckets through the use of IAM. An Organization Administrator mistakenly adds that user to the Project Admin group. The Project Admin group has the Project Editor role associated with it for the project where the four Cloud Storage buckets reside

What are the consequences of adding the user to the Project Admin group?

A. The user will now have full access to every item in the bucket except for the four buckets, which the user will have read-only access to.

B. The user will now have full access to every item in the project.

C. Groups have no association with privileges, therefore, nothing will change..

D. The user will have read-only access to the entire project.

B

Diagnostic Question Discussion

You have a user in your environment that has read-only access to four Cloud Storage buckets through the use of IAM. An Organization Administrator mistakenly adds that user to the Project Admin group. The Project Admin group has the Project Editor role associated with it for the project where the four Cloud Storage buckets reside

What are the consequences of adding the user to the Project Admin group?

A. The user will now have full access to every item in the bucket except for the four buckets, which the user will have read-only access to.

B. The user will now have full access to every item in the project.

C. Groups have no association with privileges, therefore, nothing will change..

D. The user will have read-only access to the entire project.

B

Organization Policy

Organization policies (Service Control Policies)
(≠ IAM policies)

An organization policy is:

- A configuration of restrictions
- Defined by configuring a constraint with desired restrictions.
- Applied to the organization node, folders or projects.

In AWS Organizations, Service Control Policies (SCPs) can do some of the same things, but are mostly restrictions on granted permissions that apply to everyone

GCP's Organization Policies are the direct equivalent of AWS's Service Control Policies (SCPs).
Conceptually, they are almost identical. They are the top-level governance tool used by central administrators to enforce security and compliance across their entire cloud estate

_____

AWS offers Service Control Policies as a means of defining limitations within an AWS Account or OU
GCP provides the Organization Policy service which allows for setting constraints on GCP services within any resource in the Resource Manager (Organization, Folder, Project).

_____

Source: Architecting with Compute Engine slides.

An organization policy is a configuration of restrictions, defined by configuring a constraint with the desired restrictions for that organization.

Examples:

- Limit the usage of Identity and Access Management service accounts.

- Restrict the physical location of newly created resources.
- Restricting public IP / resources

An organization policy can be applied to the organization node, and all of its folders or projects within that node. Descendants of the targeted resource hierarchy node inherit the organization policy that has been applied to their parents.

Exceptions to these policies can be made, but only by a user who has the organization policy admin role.

Most common Organization Policies

Policy Constraint	Description
compute.vmExternalIpAccess	A list of project/zone/instance names that are allowed to have external IP addresses and deny all others. Attempts to create any other VMs with an external IP address will fail.
compute.trustedImageProjects	A list of projects that contain trusted images that can be used as the basis for a VM and deny all others. Attempting to instantiate a VM with an image from another project is denied.
compute.skipDefaultNetworkCreation	Disables the creation of default VPC when creating a project. The default VPC uses auto mode subnetworks and includes default firewall rules which are often incompatible with production deployments.
iam.disableServiceAccountKeyCreation	This boolean constraint disables the creation of service account external keys where this constraint is set to `True`.
gcp.resourceLocations	This list constraint defines the set of locations where location-based GCP resources can be created. Policies for this constraint can specify multi-regions such as asia and europe, regions such as us-east1 or europe-west1, or individual zones such as europe-west1-b as allowed or denied locations.
sql.restrictPublicIp	This boolean constraint restricts configuring Public IP on Cloud SQL instances where this constraint is set to True. This constraint is not retroactive, Cloud SQL instances with existing Public IP access will still work even after this constraint is enforced. By default, Public IP access is allowed to Cloud SQL instances.
sql.disableDefaultEncryptionCreation	Restrict default Google-managed encryption on Cloud SQL instances
compute.requireShieldedVm	This boolean constraint, when set to True, requires that all new Compute Engine VM instances use Shielded disk images with Secure Boot, vTPM, and Integrity Monitoring options enabled. Secure Boot can be disabled after creation, if desired. Shielded VM features add verifiable integrity and exfiltration resistance to your VMs.

[But there are many more of these and you should be familiar with the most common ones. It's both because you can get some questions on those on the exam, but also because they're really best practices when working with GCP.]

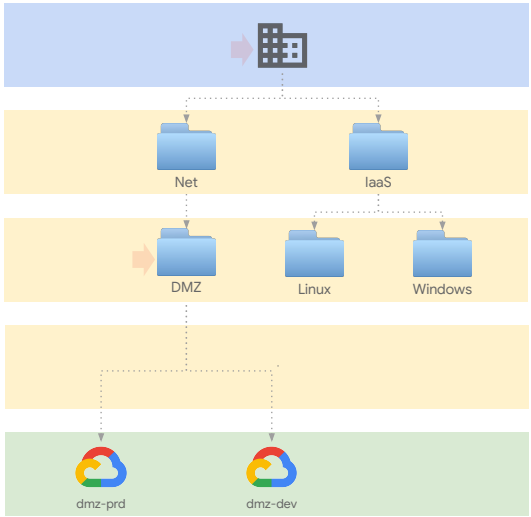
…. EXPLAIN 2-3 of the examples from the slide

Organization policy pattern example

inheritance

`constraints/compute.vmExternalIpAccess` → `false`
no VMs should be able to use external IPs

`constraints/compute.vmExternalIpAccess` → `true`
DMZ appliances should be able to use external IPs



Organization Policies vs IAM Policies

Organization Policies	IAM Policies
Constraints that allow you to: ● Limit resource sharing based on domain. ● Limit the usage of Identity and Access Management service accounts. ● Restrict the physical location of newly created resources.	Effectively they're bindings which specify what access should be granted to principal on resources.
Focuses on "what". Allows to set restrictions on specific resources to determine how they can be configured	Focuses on "who". Let's you authorize who can take action on specific resources based on permissions
Can be set on different levels (org, folder, project), propagate down but lower-level policy overwrites a higher-level one.	Effective IAM Policy on each level is a SUM of all privileges (* with an exception of "deny policies", which are not covered on the exam as of Q1 '23)
Both should be used as part of a security posture! It's NOT one or the other.	

Service Accounts

Differences in service-to-service authentication (1/3)

AWS

- Applications use IAM roles and **instance profiles** for permissions management on applications.
- Instance profiles are a container for an IAM role that can be attached to an application running in an AWS EC2 container, providing the permissions granted by the named role.

Google Cloud

- Service accounts are a type of member in Google Cloud IAM that let you give access permissions to virtual machines and other services
- Google Cloud uses **service accounts** to control service-to-service authentication using Google Cloud IAM.

A GCP **Service Account** is a **non-human identity** for your applications, virtual machines (VMs), or other services.

The closest equivalent in AWS is an **AWS IAM Role**. However, there's a key difference in how they're used for *external* applications. GCP uses a single object (the **Service Account**) for both internal and external workloads, while in AWS you might use IAM Roles or IAM Users, depending on the use-case

A GCP Service Account can be used in two primary ways:

1) Attached Service Accounts (The Best Practice)
2) Service Account Keys / Short-lived tokens

In AWS, applications use IAM roles and instance profiles for permissions management on applications.

Service accounts are a type of member in Google Cloud IAM that let you give access permissions to virtual machines and other services.

Instance profiles are a container for an IAM role that can be attached to an application running in an AWS EC2 container, providing the permissions granted by the named role. Cross-account role access can also be granted to provide fine-grained access to

resources in a separate account. Instance profiles can be managed in the AWS Console, CLI, or API.

Google Cloud uses service accounts to control service-to-service authentication using Google Cloud IAM.

Differences in service-to-service authentication (2/3)

Topic	AWS	Google Cloud
Access management	AWS IAM roles	Google Cloud IAM roles
Principle type	A role with the appropriate permissions is created and configured in the instance profile associated with a service.	Service accounts are assigned to Google Cloud instances and can be used to access Google Cloud services, resources, and application code.

Let's explore how using instance profiles in AWS compares to using service accounts in Google Cloud.

In both AWS and Google Cloud, access management is done using roles.

Principle types differ. In that in AWS, a role with the appropriate permissions is created and configured in the instance profile associated with a service. In Google Cloud, IAM service accounts are assigned to Google Cloud instances and can be used to access Google Cloud services, resources, and application code.

Differences in service-to-service authentication (3/3)

Topic	AWS	Google Cloud
Authentication	Instance profiles contain a specified role that can be used to access specified AWS resources without the need for credential management. The instance profile handles temporary credentialing and rotation of those credentials.	Service accounts are named with an email address and use cryptographic keys to access resources.

Regarding authentication, in AWS, instance profiles contain a specified role that can be used to access specified AWS resources without the need for any credential management. The instance profile handles temporary credentialing and rotation of those credentials. In Google Cloud, service accounts are named with an email address, but instead of passwords they use cryptographic keys to access resources.

Service accounts (IAM Roles)

- Provide an identity for carrying out server-to-server interactions
- Programs running within Compute Engine (EC2) instances can automatically acquire access tokens with credentials.
- Tokens are used to access to any service API in your project (account) and any other services that granted access to that service account (role).
- Service accounts (roles) are convenient when you're not accessing user data.

- **AWS IAM Role:** An identity with specific permissions that can be assumed by AWS services, IAM users, or external entities to gain temporary access to AWS resources.

AWS uses roles to grant permissions for inter-service communication. For example, a Lambda in AWS may access resources by assuming an IAM role.

In GCP the equivalent would be a service account. As in AWS, Google's cloud functions (the GCP equivalent of Lambda) would have a service account attached to them and the cloud function can then use the permissions granted to that service account. (SHOW)

AWS IAM Roles are identities that you can create in your AWS account that have specific permissions. Unlike IAM Users, roles are not associated with a single person or a long-term credential like a password or access key. Instead, they are designed to be *assumed* by trusted entities.

Source: Architecting with Compute Engine slides.

A service account is an account that belongs to your application instead of to an individual end user. This provides an identity for carrying out server-to-server interactions in a project without supplying user credentials.

For example, if you write an application that interacts with Google Cloud Storage, it must first authenticate to either the Google Cloud Storage XML API or JSON API.

You can enable service accounts and grant read-write access to the account on the instance where you plan to run your application.

Then, program the application to obtain credentials from the service account. Your application authenticates seamlessly to the API without embedding any secret keys or credentials in your instance, image, or application code.

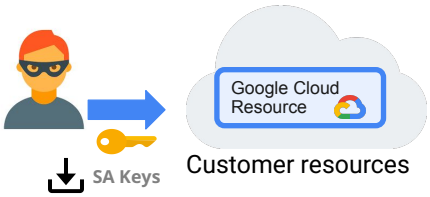
Service account use cases

- Typically, service accounts are used in scenarios such as:
  - Running workloads on virtual machines (VMs)
    - E.g., Create a service account with permissions to query BigQuery
      - Attach it to a VM
      - Deploy an application onto the VM that submits SQL commands to BigQuery
  - Running workloads on on-premises workstations or data centers that call Google APIs.
    - E.g., Same example as above, but now the application is running on-premise
  - Running workloads which are not tied to the lifecycle of a human user.
    - E.g., Batch jobs that are scheduled to run periodically

Service account keys pose a security risk to your resources

Proprietary + Confidential

- SA keys are similar to a **password without an expiration date**.
- SA Keys can be leaked accidently and attackers can use it to access your (GCP project or org admin) sensitive GCP resources.
- Usage cannot be audited → compounding the risk



Customers have downloaded > 48 Million Service Account Keys!!

So what's the solution? Ditch the keys and use Workload Identity for GKE & Workload Identity Federation!

SA keys are long lived.
Difficult to safeguard. Can be leaked accidentally in public repo/debug/system logs
Usage cannot be audited
Risk increases as multi-cloud deployment increased

Workload Identity Federation: Keyless Access

User Story: As an App Developer, I want to **securely connect my services (= NOT users, like in Workforce Identity Federation!)** to GCP resources without downloading access keys.

Benefits

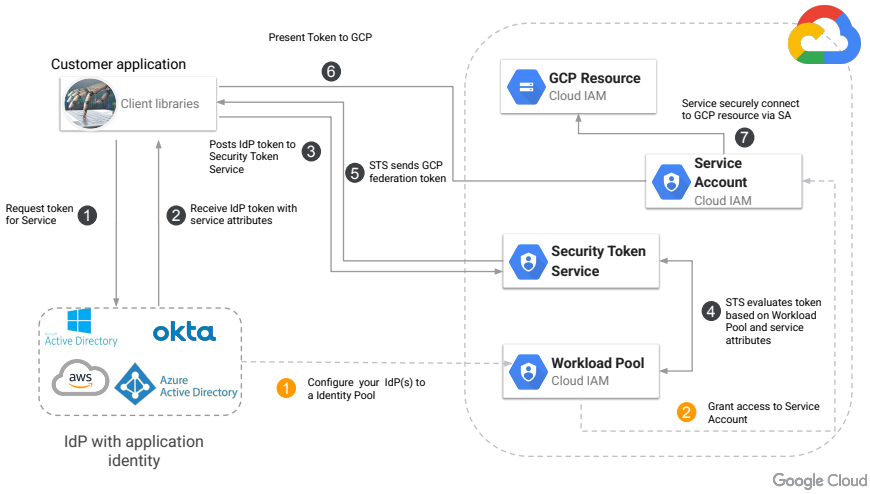
Keyless access to GCP APIs

Auditability through Cloud logs

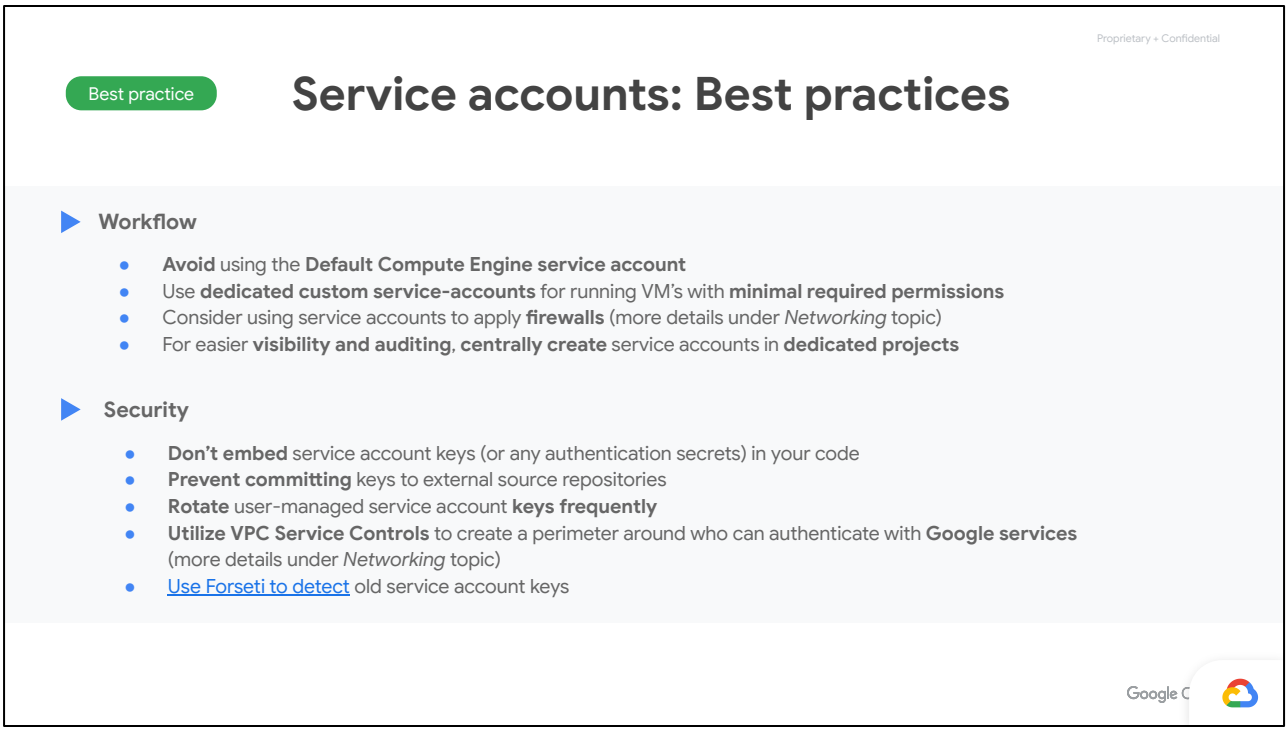
Attribute-based access control

Exam Tips:

- Have a look at a great explanation of WIF
- Have a look at Workflow Identity Federation documentation



– show “Workload Identity Federation” service in GCP console, encourage to try it out.



TL;DR / Purpose of the slide:

- Summarize **best practices** for **managing service accounts and their keys**

Key points:

- **Workflow**
  - **Default svc account** is too generous with permissions (Project Editor). Create **app specific** accts, and **only grant needed permissions**.
  - **Svc accts** can be used for **selective application** of **apply firewalls**. E.g: Open 443 (HTTPS) for VM's when svc account 'webapp-fe'
  - **Create svc accts** on **dedicated projects** for **central mgmt**
- **Security risks** related to **user managed keys**
  - A major risk is **keys** being compromised, either **maliciously** or mistakingly **publishing keys** by **embedding in code**
  - To help with that, **rotate keys frequently**
  - **VPC Service Controls** helps limiting who can access GCP services (which is what svc accts are ultimately for). E.g: Access only permitted from on-prem IP ranges (when interconnecting). This **greatly minimizes attack surface.** More under *Networking* topic.
  - Combine this with a **proactive approach** by using **Forseti** to alert on old keys that need to be rotated

Probing questions (optional):

- None

Diagnostic Question Discussion

Cymbal Direct's social media app must run in a **separate project** from its APIs and web store. You want to use **Identity and Access Management (IAM)** to ensure a **secure environment.**

How should you set up IAM?

A. Use **separate** service accounts for each component (social media app, APIs, and web store) with **basic** roles to grant access.

B. Use **one** service account for all components (social media app, APIs, and web store) with **basic** roles to grant access.

C. Use separate service accounts for each component (social media app, APIs, and web store) with **predefined or custom** roles to grant access.

D. Use **one** service account for all components (social media app, APIs, and web store) with **predefined or custom** roles to grant access.

Google Cloud

Feedback:
A. Incorrect. Basic roles include thousands of permissions across all Google Cloud services. In production environments, do not grant basic roles unless there is no alternative.
B. Incorrect. Treat each component of your application as a separate trust boundary. If multiple services require different permissions, create a separate service account for each service, and then grant only the required permissions to each service account. Basic roles include thousands of permissions across all Google Cloud services. In production environments, do not grant basic roles unless there is no alternative.
C. Correct! Using separate service accounts for each component allows you to grant only the access needed to each service account with either a predefined or custom role.
D. Incorrect. Treat each component of your application as a separate trust boundary. If multiple services require different permissions, create a separate service account for each service, and then grant only the required permissions to each service account.

Where to look:
https://cloud.google.com/blog/products/identity-security/iam-best-practice-guides-available-now

Content mapping:
Architecting with Google Cloud: Design and Process, M8

Summary:
Using IAM and designing the best environmental approach largely relies on abiding by the “principle of least privilege.” This question included some of the recommended practices. Treat each component of your application as a separate trust boundary. If multiple services require different permissions, create a separate service account for each service, and then grant each the least permissions possible. Basic roles include thousands of permissions across all Google Cloud services, so using them probably provides more abilities than necessary. In production environments, do not grant basic roles unless there is no alternative. A checklist you can use to ensure best practices is available here: https://cloud.google.com/iam/docs/using-iam-securely

Diagnostic Question Discussion

Cymbal Direct's social media app must run in a **separate project** from its APIs and web store. You want to use **Identity and Access Management (IAM)** to ensure a **secure environment.**

How should you set up IAM?

A. Use **separate** service accounts for each component (social media app, APIs, and web store) with **basic** roles to grant access.

B. Use **one** service account for all components (social media app, APIs, and web store) with **basic** roles to grant access.

C. **Use separate service accounts for each component (social media app, APIs, and web store) with predefined or custom roles to grant access.**

D. Use **one** service account for all components (social media app, APIs, and web store) with **predefined or custom** roles to grant access.

Google Cloud

Feedback:
A. Incorrect. Basic roles include thousands of permissions across all Google Cloud services. In production environments, do not grant basic roles unless there is no alternative.
B. Incorrect. Treat each component of your application as a separate trust boundary. If multiple services require different permissions, create a separate service account for each service, and then grant only the required permissions to each service account. Basic roles include thousands of permissions across all Google Cloud services. In production environments, do not grant basic roles unless there is no alternative.
C. Correct! Using separate service accounts for each component allows you to grant only the access needed to each service account with either a predefined or custom role.
D. Incorrect. Treat each component of your application as a separate trust boundary. If multiple services require different permissions, create a separate service account for each service, and then grant only the required permissions to each service account.

Where to look:
https://cloud.google.com/blog/products/identity-security/iam-best-practice-guides-available-now

Content mapping:
Architecting with Google Cloud: Design and Process, M8

Summary:
Using IAM and designing the best environmental approach largely relies on abiding by the “principle of least privilege.” This question included some of the recommended practices. Treat each component of your application as a separate trust boundary. If multiple services require different permissions, create a separate service account for each service, and then grant each the least permissions possible. Basic roles include thousands of permissions across all Google Cloud services, so using them probably provides more abilities than necessary. In production environments, do not grant basic roles unless there is no alternative. A checklist you can use to ensure best practices is available here: https://cloud.google.com/iam/docs/using-iam-securely

Diagnostic Question Discussion

You are writing an application that will be required to read and write from Cloud Storage. You want to ensure that you do not cause any security issues in your configuration.

What would be the best option for granting the application access to Cloud Storage?

A. Create a user account and apply a curated role that gives the user read/write permissions to the Cloud Storage bucket. Then, in the application configuration file, enter the username and password of the account in the file.

B. Open all access to the Cloud Storage bucket so it is accessible to the public, then use the bucket for your application.

C. Create a service account that only has permissions to read and write from the Cloud Storage bucket you are using for the application. Then, apply the service account to the virtual machine that is running your application.

D. Ensure the scope of the instance has full privileges to Cloud Storage.

C

Diagnostic Question Discussion

You are writing an application that will be required to read and write from Cloud Storage. You want to ensure that you do not cause any security issues in your configuration.

What would be the best option for granting the application access to Cloud Storage?

A. Create a user account and apply a curated role that gives the user read/write permissions to the Cloud Storage bucket. Then, in the application configuration file, enter the username and password of the account in the file.

B. Open all access to the Cloud Storage bucket so it is accessible to the public, then use the bucket for your application.

C. Create a service account that only has permissions to read and write from the Cloud Storage bucket you are using for the application. Then, apply the service account to the virtual machine that is running your application.

D. Ensure the scope of the instance has full privileges to Cloud Storage.

C

Diagnostic Question Discussion

Which of the following are the best practices recommended by Google Cloud when dealing with Service Accounts? (choose 3):

A. Grant the Service Accounts full set of permissions

B. Do not delete service accounts that are in use by running instances on App Engine or Compute Engine

C. Grant “Service Account User” role to all users in the organization

D. When you create a service account, give it a meaningful name to indicate the purpose of the service account

E. Create separate service accounts for different purposes with minimal set of privileges required for that service.

B D E

Diagnostic Question Discussion

Which of the following are the best practices recommended by Google Cloud when dealing with Service Accounts? (choose 3):

A. Grant the Service Accounts full set of permissions

B. Do not delete service accounts that are in use by running instances on App Engine or Compute Engine

C. Grant “Service Account User” role to all users in the organization

D. When you create a service account, give it a meaningful name to indicate the purpose of the service account

E. Create separate service accounts for different purposes with minimal set of privileges required for that service.

B D E

Make sure to...

Enjoy the journey as much as the destination!

Now that you know about the overall setup of this course and how to use the workbook, let's get started by exploring section 1 of the exam guide.