Google Cloud

Partner Certification Academy

Professional Cloud Network Engineer

The information in this presentation is classified:

Google confidential & proprietary

⚠️ This presentation is shared with you under NDA.

- Do **not** record or take screenshots of this presentation.
- Do **not** share or otherwise distribute the information in this presentation with anyone **inside** or **outside** of your organization.

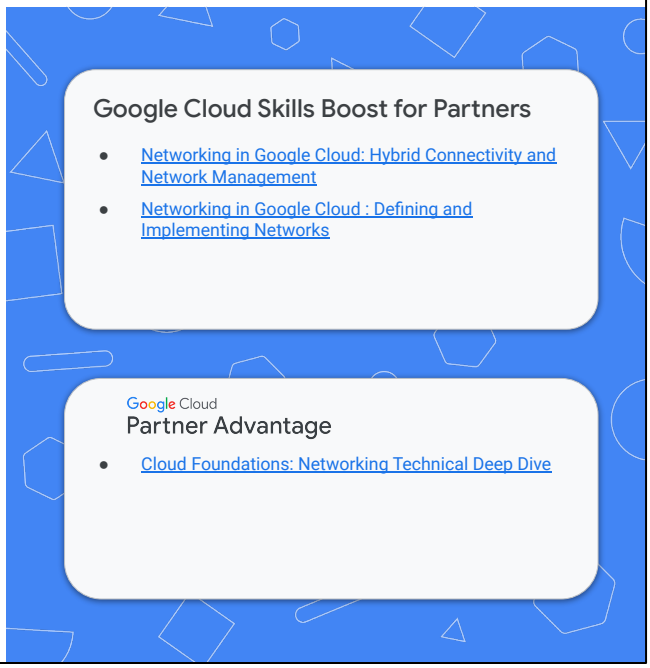
Thank you!

Google Cloud

Source Materials

Some of this program's content has been sourced from the following resources:

- Google Cloud certification site
- Google Cloud documentation
- Google Cloud console
- Google Cloud courses and workshops
- Google Cloud white papers
- Google Cloud Blog
- Google Cloud YouTube channel
- Google Cloud partner-exclusive resources



Session logistics

Questions

In Google Meet, click the raise hand button or add your question to the Q&A section.

Answers may be deferred until the end of the session.

Slide availability

These slides are available in the Student Lecture section of your Qwiklabs classroom.

Recording

The session is **not** recorded.

Chat

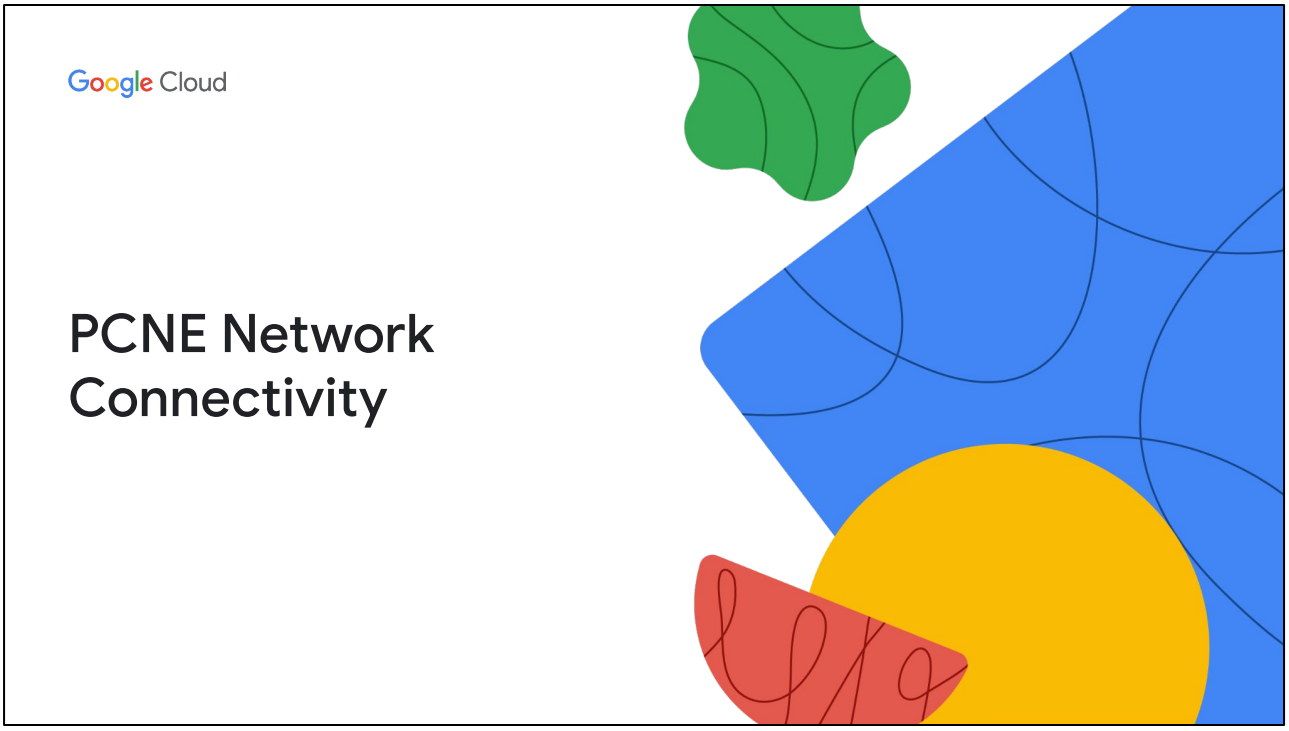
As Google Meet does not have persistent chat, you will lose chat history if you get disconnected. Save URLs as they appear.

When you have a question, please:
Click the Raise hand button in Google Meet.
Or add your question to the Q&A section of Google Meet.
Please note that answers may be deferred until the end of the session.

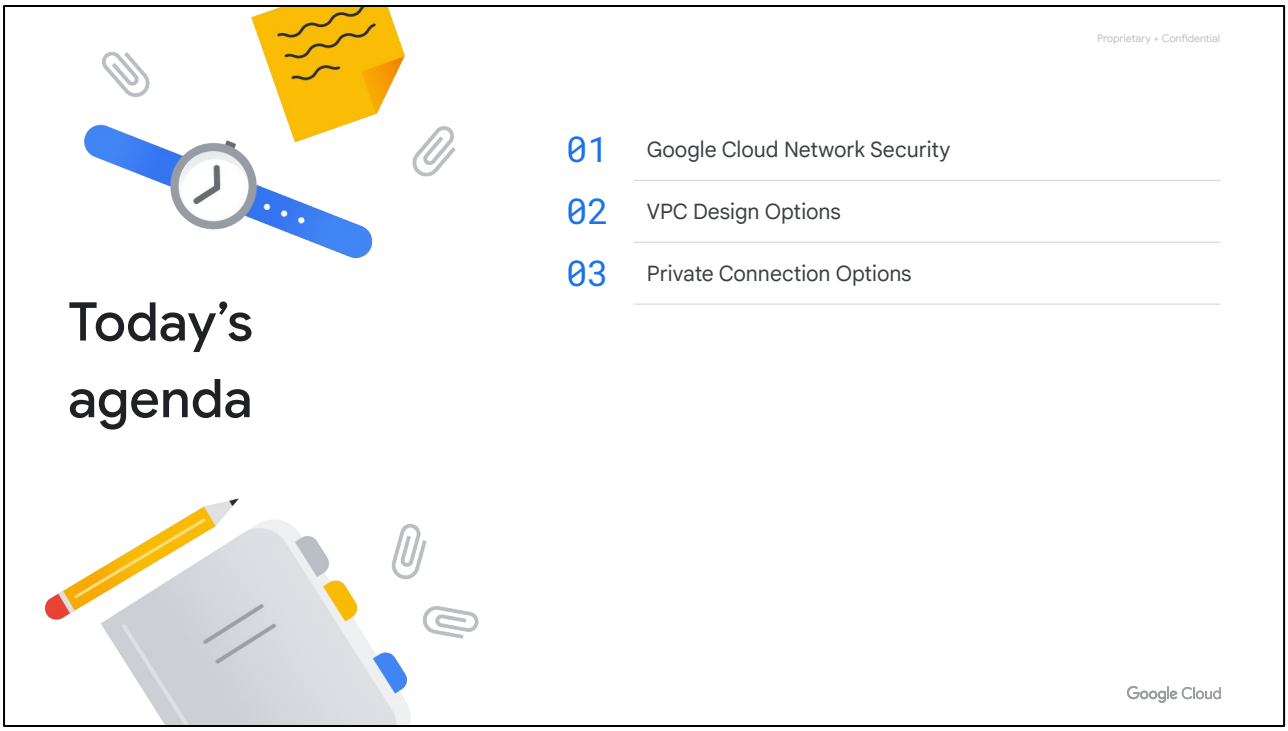
These slides are available in the Student Lecture section of your Qwiklabs classroom.

The session is not recorded.

Google Meet does not have persistent chat.
If you get disconnected, you will lose the chat history.
Please copy any important URLs to a local text file as they appear in the chat.



Welcome to this module: PCNE Network Connectivity.


Proprietary + Confidential
Today's agenda
01 Google Cloud Network Security
02 VPC Design Options
03 Private Connection Options
Google Cloud

AGENDA

Objectives

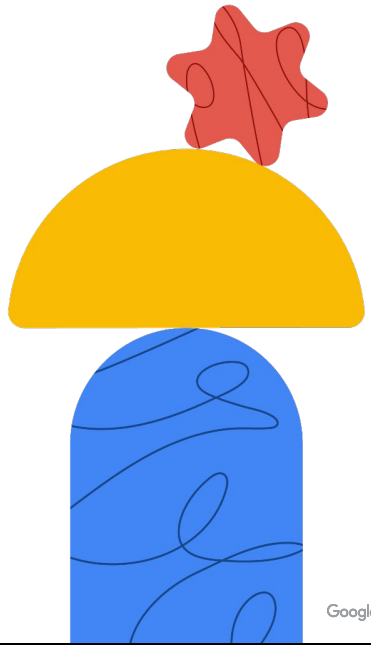
01 Explain Google Cloud security mechanisms, as they relate to networking.

02 Describe VPC Design options such as VPC Peering and Shared VPC.

03 Describe the different private connection options available in Google Cloud networks.

AGENDA

Google Cloud network security



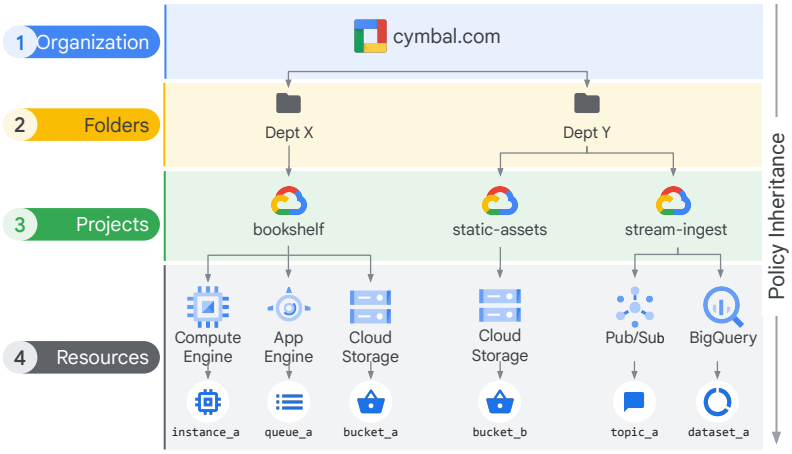
~~In this module, we will be covering virtual networks.~~
In this module, you will learn about virtual networks.

Google Cloud uses a software-defined network that is built on a global fiber infrastructure. This infrastructure makes Google Cloud one of the world's largest and fastest networks. ~~Thinking about resources as services instead of as hardware will help you understand the options that are available, and their behavior.~~

Rethinking resources as services rather than hardware can offer a deeper understanding of the available options and their behaviors.

Cloud IAM resource hierarchy

- A policy is set on a resource, and each policy contains a set of roles and members.
- Resources inherit policies from the parent.
- If the parent policy is less restrictive, it overrides a more restrictive resource policy.



The diagram that is displayed now is a sample Cloud IAM resource hierarchy. Let's use the diagram to review how Cloud IAM works.

Google Cloud resources are organized hierarchically as shown in this tree structure. The Organization node is the root node in this hierarchy, folders are the children of the organization, projects are the children of the folders, and the individual resources are the children of projects. Each resource has exactly one parent.

Cloud IAM allows you to set policies at all of these levels, where a policy contains a set of roles and members. Let's go through each of the levels from top to bottom, as resources inherit policies from their parent.

The organization resource represents your company. Cloud IAM roles granted at this level are inherited by all resources under the organization.

The folder resource could represent your department. Cloud IAM roles granted at this level are inherited by all resources that the folder contains.

Projects represent a trust boundary within your company. Services within the same project have a default level of trust.

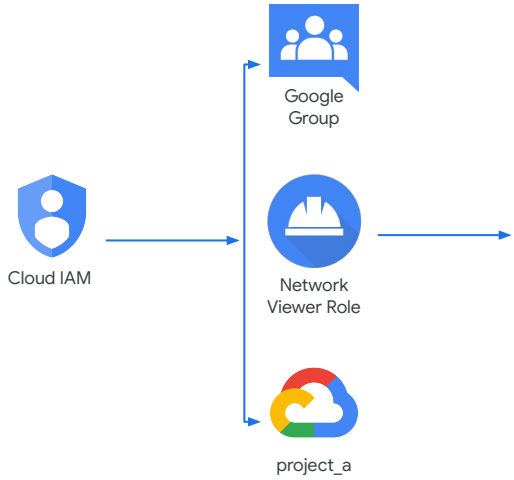
Cloud IAM resource hierarchy

- Cloud IAM policy hierarchy follows the Google Cloud resource hierarchy.
- Changing resource hierarchy changes the policy hierarchy.
- Moving a project to a different organization updates its policy to inherit from the new organization's policy.
- Child policies can't restrict access granted at the parent level.
- Follow the principle of least privilege for identities, roles, and resources.
- Select the smallest scope to reduce risk exposure.

The Cloud IAM policy hierarchy always follows the same path as the Google Cloud resource hierarchy, meaning, if you change the resource hierarchy, the policy hierarchy also changes. For example, moving a project into a different organization will update the project's Cloud IAM policy to inherit from the new organization's Cloud IAM policy.

Another thing to point out, is that child policies cannot restrict access granted at the parent level. For example, if someone grants you the Editor role for Department X and someone grants you the Viewer role at the bookshelf project level, then you still have the Editor role for that project. ~~Therefore, it is a best practice is~~ Therefore, it is best practice to follow the principle of least privilege. The principle applies to identities, roles, and resources. Always select the smallest scope that's necessary to reduce your exposure to risk.

Predefined roles



List of Permissions

- ✓ compute.addresses.get
- ✓ compute.addresses.list
- ✓ compute.globalAddresses.get
- ✓ compute.globalAddresses.list
- ✓ commute.backendBuckets.get
- ✓ compute.backendBuckets.list
- ✓ compute.networks.list

...

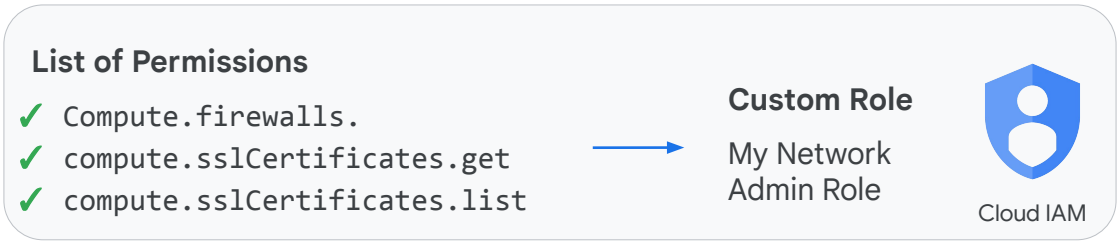
In addition to the basic roles, Cloud IAM provides predefined roles that give granular access to specific Google Cloud resources and prevents unwanted access to other resources. These roles are collections of permissions.

Most of the time, to do any meaningful operations, you need more than one permission.
~~For example, in this slide, a group of users is granted the Network Viewer role on project_a. This provides the users of that group a lot of permissions, of which some are illustrated on the right-hand side.~~
As illustrated in the example, a group of users is currently granted the Network Viewer role within project_a. This grants the members of this group an extensive range of permissions, a few of which are listed here.

The permissions are classes and methods in the APIs. For example, compute.networks.list can be broken into the service, resource, and verb, meaning that this permission is used to list all of the VPC networks that project_a contains.

Grouping these permissions into roles, and having ~~those~~ these roles represent abstract functions, makes them easier to manage. Also, users can have multiple roles, providing flexibility.

Custom roles

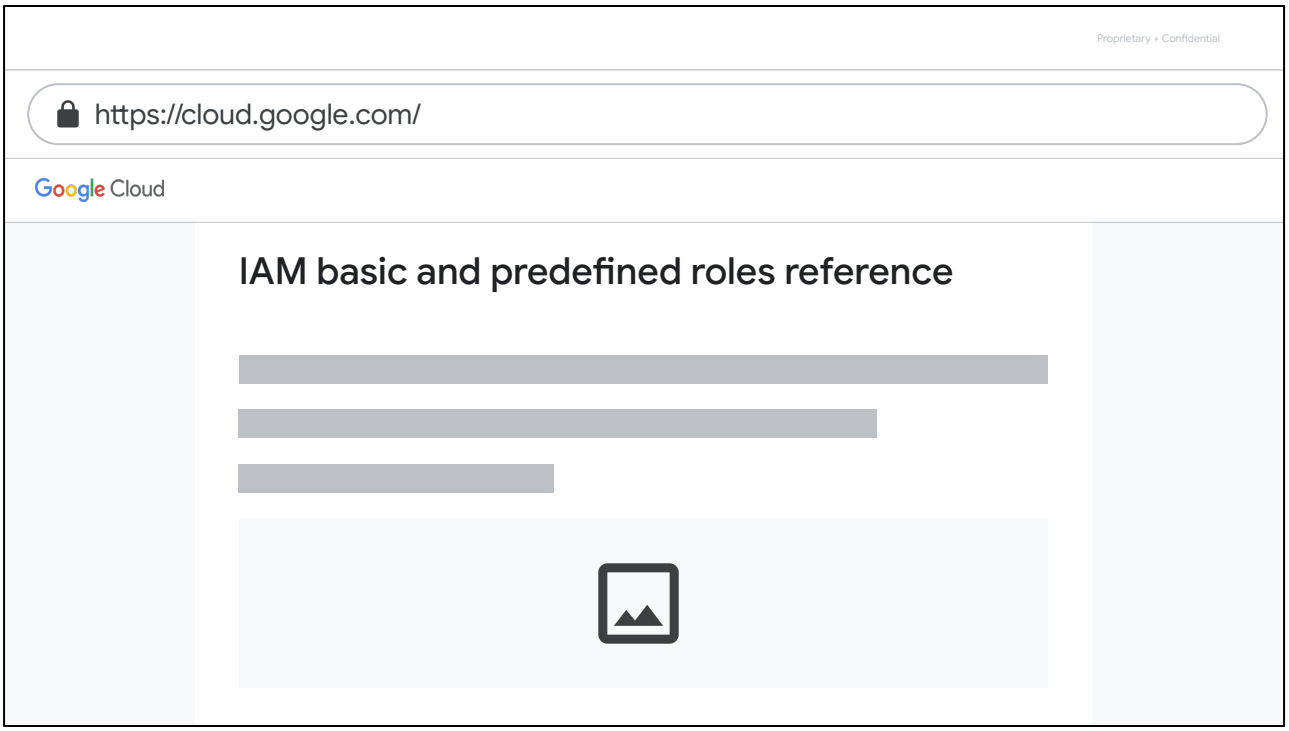


In addition to the predefined roles, Cloud IAM also provides the ability to create customized Cloud IAM roles. You can create a custom Cloud IAM role with one or more permissions, and then grant that custom role to users who are part of your organization.

In essence, custom roles enable you to enforce the principle of least privilege, ensuring that the user and service accounts in your organization have only the permissions essential to performing their intended functions.

For example, you might want a user to create, modify, and delete firewall rules but have read-only permissions to SSL certificates. In this case, the Security Admin role provides too many permissions and the Network Admin role does not provide enough. So, you can select the corresponding permissions for firewall rules and the SSL certificate, as displayed here, ~~as shown on the left-hand side~~ along with any other permissions to create a new custom network admin role.

Cloud IAM provides a UI and API for creating and managing custom roles. For more information on custom roles, refer to Custom roles in the Google Cloud documentation.



For more information on custom roles, refer to Custom roles in the Google Cloud documentation.

Network-related IAM roles

Role Title	Description
Network Viewer	Read-only access to all networking resources
Network Admin	Permissions to create, modify, and delete networking resources, except for firewall rules and SSL certificates
Security Admin	Permissions to create, modify, and delete firewall rules and SSL certificates

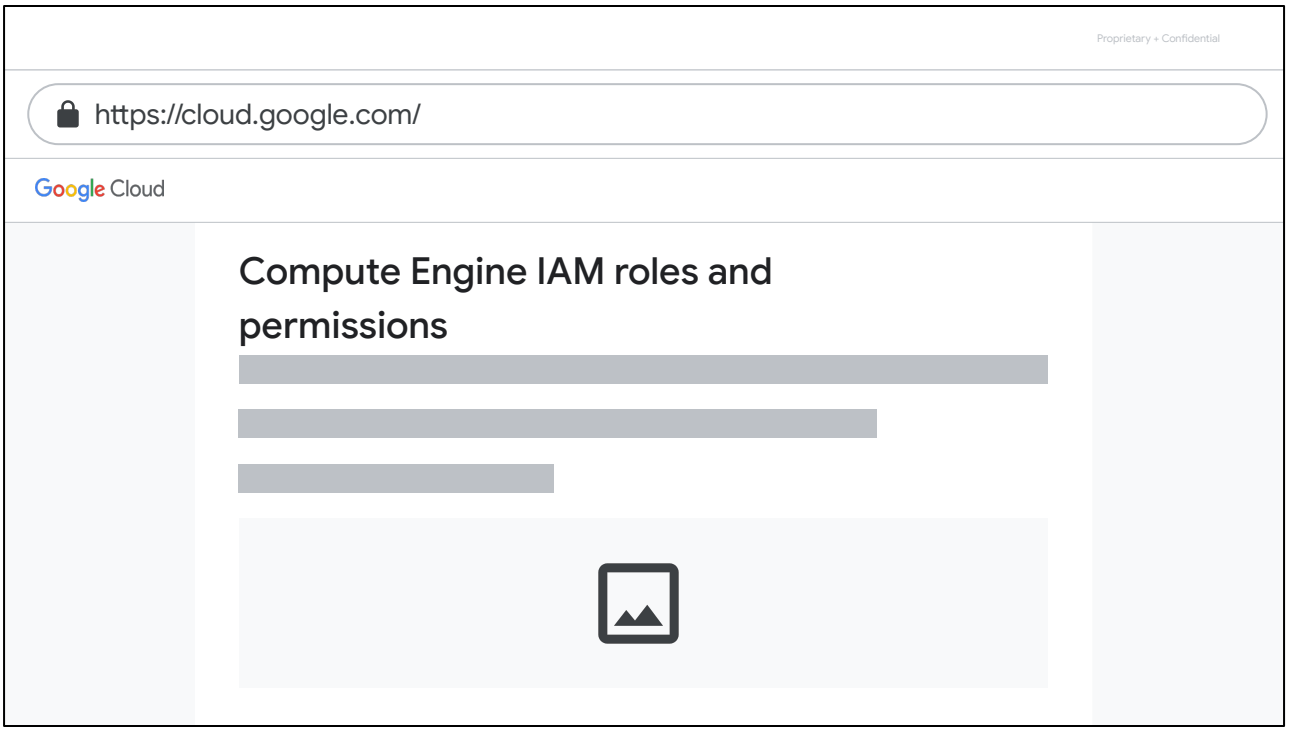
Let's focus on predefined roles that provide granular access to VPC networking resources.

There is the Network Viewer role that provides read-only access to all networking resources. For example, if you have software that inspects your network configuration, you could grant that software's service account the Network Viewer role.

Next, the Network Admin role contains permissions to create, modify, and delete networking resources, except for firewall rules and SSL certificates. In other words, the network admin role allows read-only access to firewall rules, SSL certificates, and instances to view their ephemeral IP addresses.

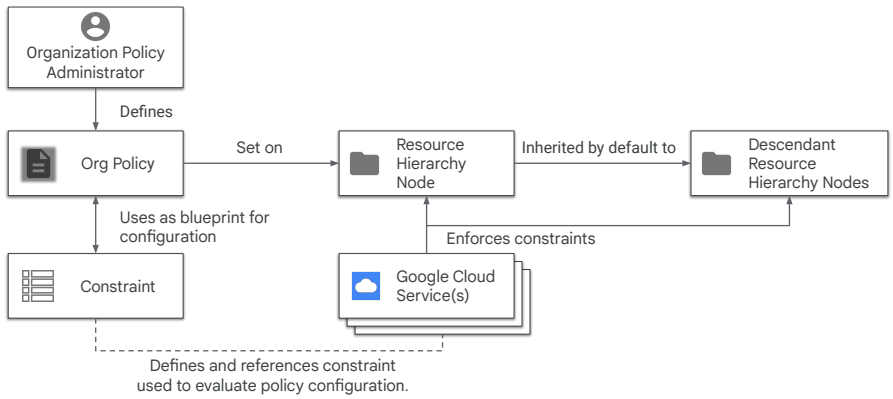
The Security Admin role contains permissions to create, modify, and delete firewall rules and SSL certificates.

Now, there are other predefined roles for networking resources that relate to Shared VPC, which allows an organization to connect resources from multiple projects to a common VPC network. We will cover Shared VPC along with those other predefined roles in a later module of this course.



For more information on these roles, see Compute Engine IAM roles and permissions in the Google Cloud documentation.

The Organization Policy Service gives you centralized and programmatic control over your cloud resources



Identity and Access Management focuses on the who, and lets the administrator authorize who can take action on specific resources based on permissions. Organization Policy focuses on the what, and lets the administrator set restrictions on specific resources to determine how they can be configured. The Organization Policy Service gives you centralized and programmatic control over your organization's cloud resources. An organization policy administrator, configures constraints across the entire resource hierarchy.

~~An organization policy is a configuration of restrictions. You, as the organization policy administrator, define an organization policy, and you set apply that organization policy on organizations, folders, and projects in order to enforce the restrictions on that those resources and its descendants. In order to define an organization policy, you choose a constraint, which is a particular type of restriction against either a Google Cloud service or a group of Google Cloud services. You configure that constraint with your desired restrictions.~~

An organization policy is a configuration of restrictions. The organization policy administrator, defines an organization policy, and then applies it to organizations, folders, and projects in order to enforce the restrictions on those resources and descendants.

To establish an organizational policy, select a constraint, which is a specific type of restriction applicable to either a Google Cloud service or a group of Google Cloud services. Afterward, configure that constraint with your preferred restrictions.

Descendants of the targeted resource hierarchy node inherit the organization policy. By applying an organization policy to the root organization node, you are able to effectively drive enforcement of that organization policy and configuration of restrictions across your organization.

Organization policies

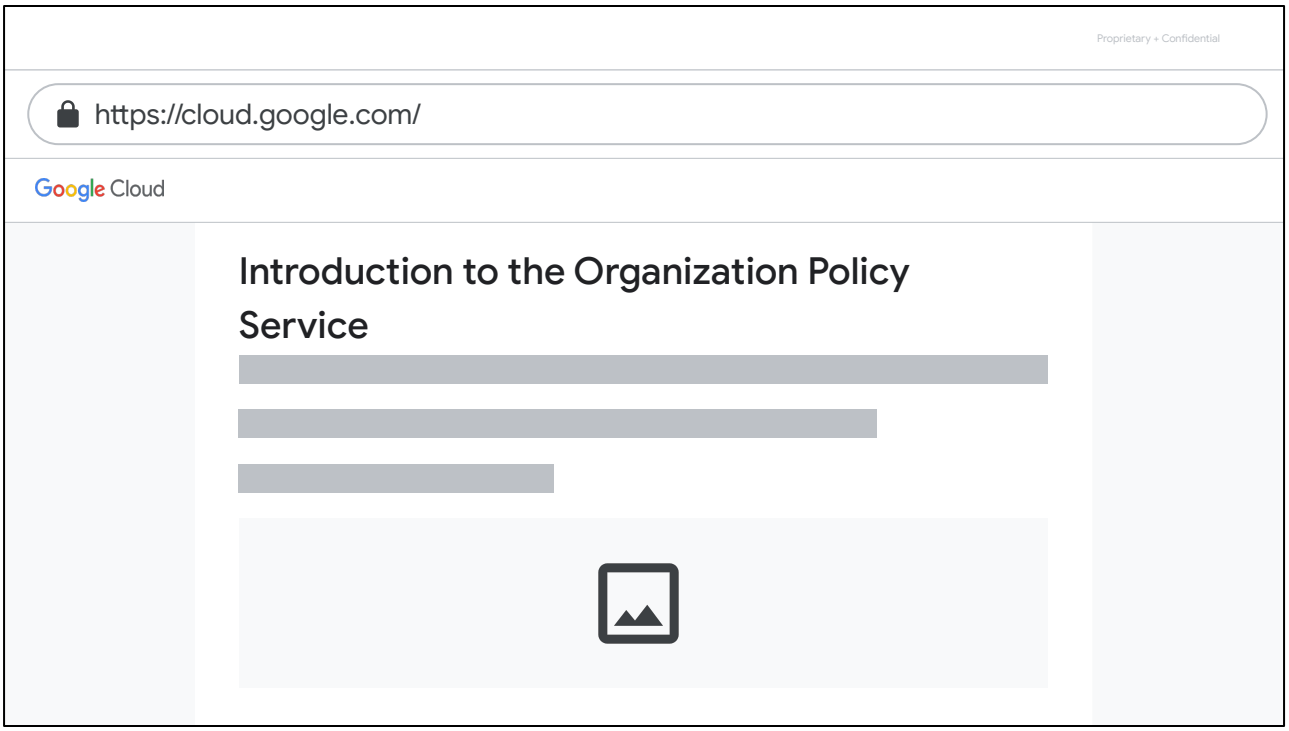
- The Organization Policy Service constrains the allowed resource configurations.
- Policies are managed centrally on the organization level, and can be applied to the **organization**, **folders**, and **projects**.

From Cloud Foundations: Security

The Organization Policy Service gives you centralized and programmatic control over your organization's cloud resources. As the organization policy administrator, you will be able to configure constraints across your entire resource hierarchy.

Identity and Access Management focuses on who, and lets the administrator authorize who can take action on specific resources based on permissions.

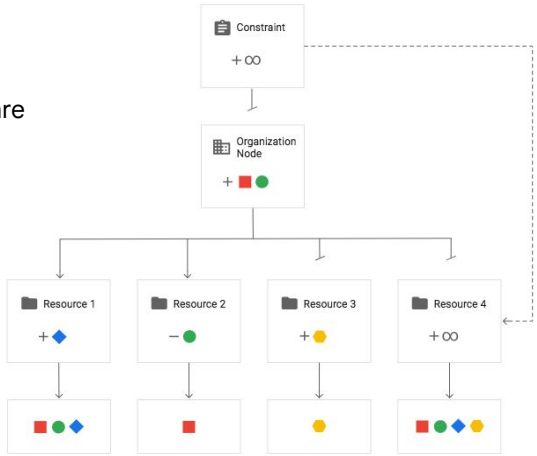
Organization Policy focuses on what, and lets the administrator set restrictions on specific resources to determine how they can be configured.



For more information on the Organization Policy Service, visit the Introduction to the Organization Policy Service resource in the official Google Cloud documentation.

Organization Policy hierarchy evaluation

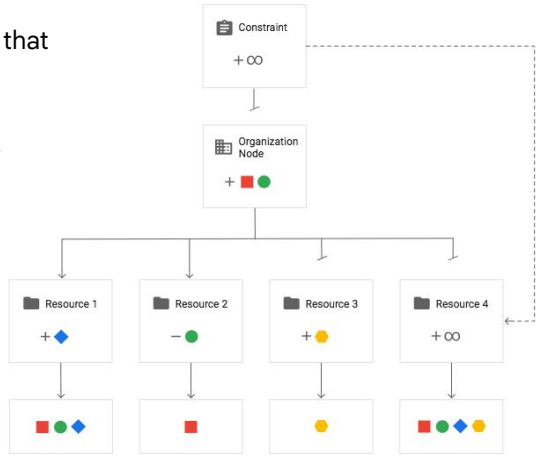
In this example, **Organization Node** defines a policy that allows red square and green circle.



In the example displayed now, the **Organization Node** defines a policy that allows the red square and green circle.

Organization Policy hierarchy evaluation

Resource 1 defines a custom policy that sets inheritFromParent to TRUE and allows blue diamond. The effective policy evaluates to allow red square, green circle, and blue diamond.

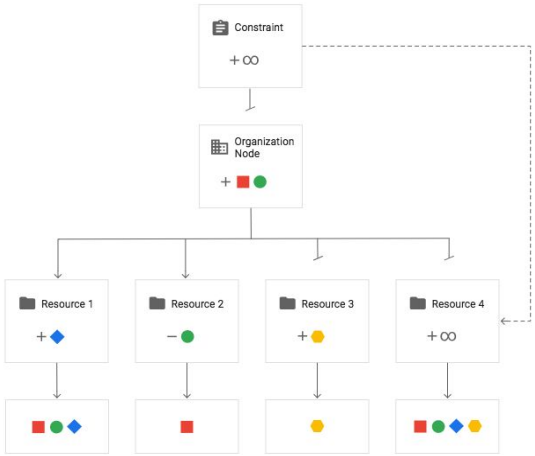


Resource 1 defines a custom policy that sets inheritFromParent to TRUE and allows the blue diamond. The effective policy evaluates to allow the red square, green circle, and blue diamond.

Organization Policy hierarchy evaluation

Resource 2 defines a custom policy that sets inherit FromParent to TRUE and denies green circle. Deny takes precedence. The effective policy evaluates to allow only red square.

.



Resource 2 defines a custom policy that sets inheritFromParent to TRUE and denies the green circle. Deny takes precedence. The effective policy evaluates to allow only the red square.

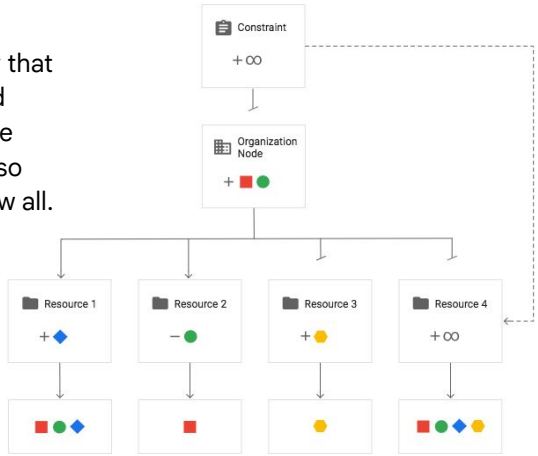
Organization Policy hierarchy evaluation

Resource 3 defines a custom policy that sets inheritFromParent to FALSE and allows yellow hexagon. The effective policy evaluates to only allow yellow hexagon.

Resource 3 defines a custom policy that sets inheritFromParent to FALSE and allows the yellow hexagon. The effective policy evaluates to only allow the yellow hexagon.

Organization Policy hierarchy evaluation

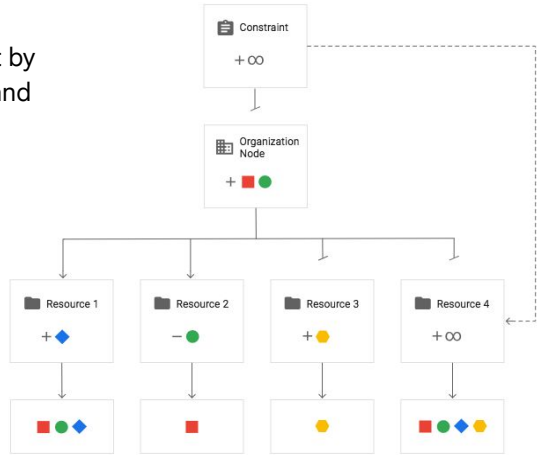
Resource 4 defines a custom policy that sets inheritFromParent to FALSE and includes the restoreDefaultvalue. The default constraint behavior is used, so the effective policy evaluates to allow all.



Resource 4 defines a custom policy that sets inheritFromParent to FALSE and includes the restoreDefaultvalue. The default constraint behavior is used, so the effective policy evaluates to allow all.

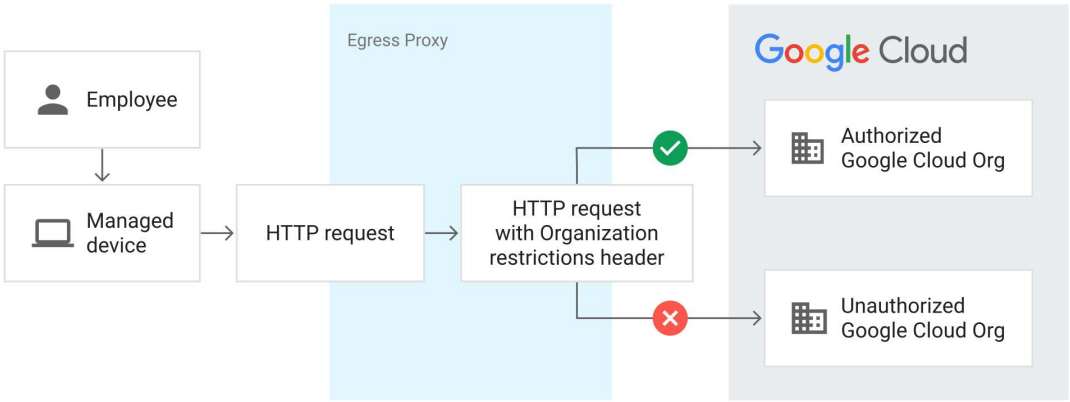
Organization Policy hierarchy evaluation

Note: The exceptions are always set by org-level organization policy roles, and not by project-level roles.

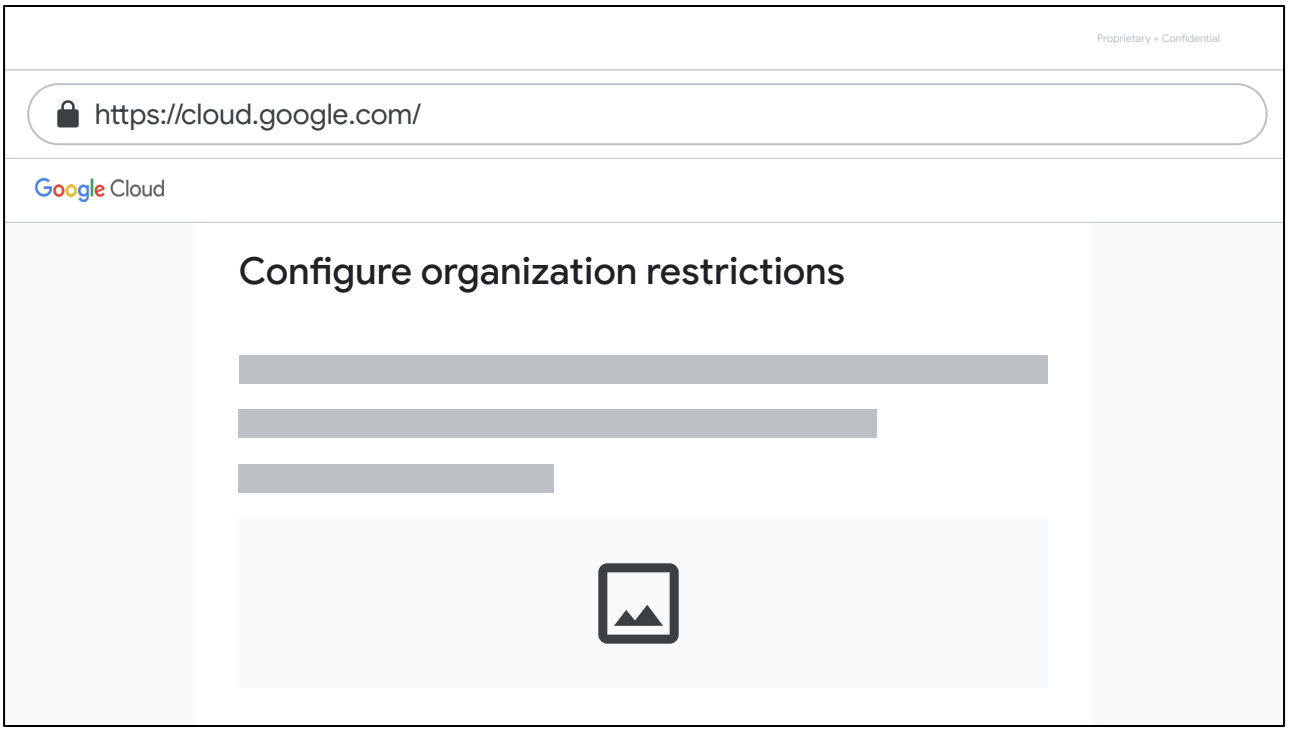


It is important to note that the exceptions are always set by org-level organization policy roles, and not by project-level roles.

Organization Restrictions



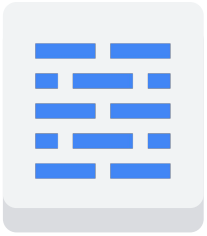
- Restrict your employees' access to authorized organizations only
- Requires [configuration of Egress Proxy](#).



To learn more about configuring the egress proxy, visit the official Google Cloud documentation titled ‘Configure organization restrictions.

Cloud Next Generation Firewall (NGFW)

- Distributed firewall service
- Supports network and firewall policies
- Supports VPC firewall rules
- Firewall rule logging
- IAM-governed tags for micro-segmentation
- Address groups allow combining multiple addresses/ranges into a single logical unit
- Three tiers: Essentials, Standard, and Enterprise

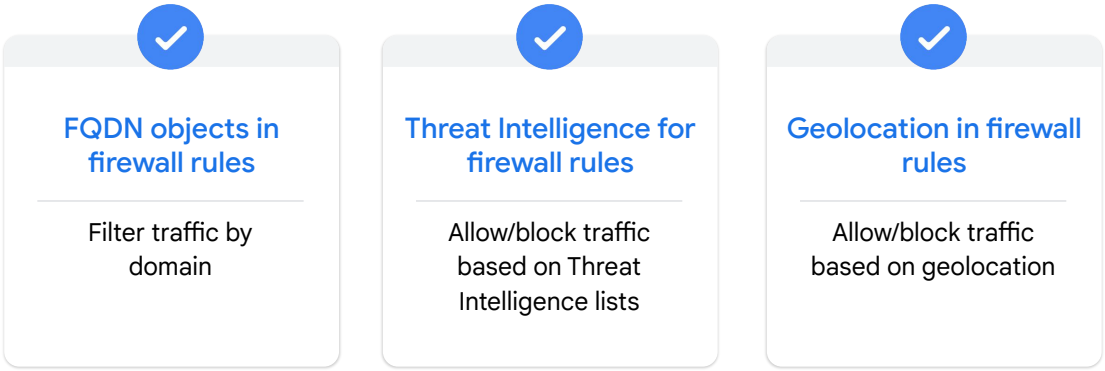


All features are available in all tiers

https://cloud.google.com/firewall/docs/about-firewalls

Cloud NGFW Standard

All of the features of NGFW Essentials, plus:



https://cloud.google.com/firewall/docs/about-firewalls#firewall-standard

Cloud NGFW Enterprise

All of the features of NGFW Standard, plus:

- Advanced layer 7 capabilities
- Intrusion Prevention Service (in Preview as of 03/24)
- TLS interception and decryption

https://cloud.google.com/firewall/docs/about-firewalls#firewall-plus

Firewall rules

Protect your VM instances from unapproved connections

Google Cloud firewall rules

- Firewall rules are applied to the network as a whole.
- Connections are allowed or denied at the instance level.
- Firewall rules are stateful.
- Deny all ingress and allow all egress rules are implied.

Google Cloud firewall rules protect your virtual machine instances from unapproved connections, both inbound and outbound, known as ingress and egress, respectively. Essentially, every VPC network functions as a distributed firewall.

Although firewall rules are applied to the network as a whole, connections are allowed or denied at the instance level.

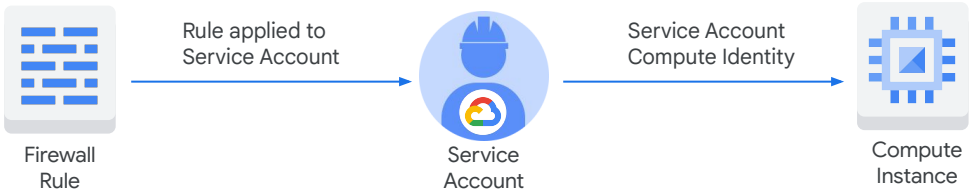
Let's explore this concept more deeply by controlling access to specific target instances and from specific source instances, not by their IP range but by using network tags and service accounts.

Firewall rule parameters

Target and source

Target:
- All instances in the network
- Specified target tags
- Specified service accounts

Source:
- IP ranges
- Subnets
- Source tags
- Service account



A firewall rule is composed of several parameters.

The target parameter of a firewall rule defines the instances to which the firewall rule is intended to apply. This parameter has three choices, as shown on this slide. “All instances in the network” specifies that the firewall rule applies to all instances in the network, which we explored in the previous module.

To provide more granularity, firewall rules can also be applied to only those VM instances that match specific service accounts or network tags, which are user-defined strings that you can apply to your VM instances.

The source parameter of a firewall rule is intended for ingress traffic and defines the allowed sources of the traffic. Similar to the target parameter, firewall rules can be applied to traffic coming from VM instances that match a specific network tag or use a specific service account.

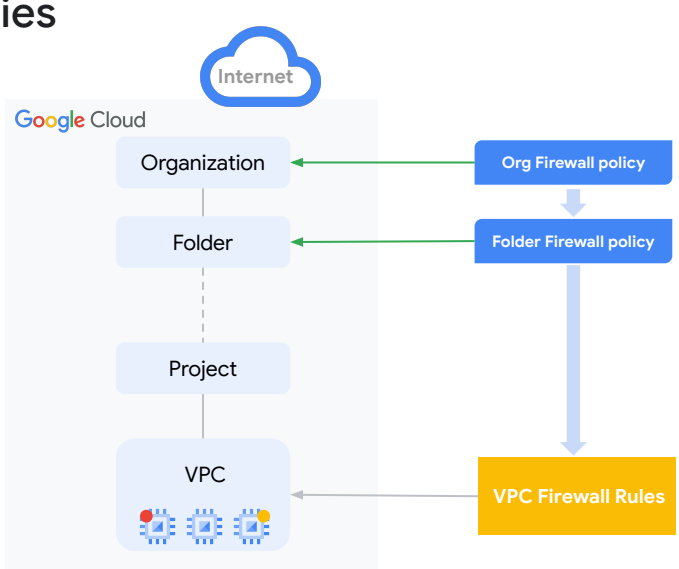
Because service accounts are controlled through Cloud IAM, they are considered more secure than tags.

For more information on filtering by service account versus network tag, refer to Filtering by service account versus network tag in the Google Cloud documentation.

Hierarchical firewall policies

Secure and flexible protection for your dynamic cloud environment

1. **Hierarchical** policy enforcement enables safe delegation and automatic protection of new projects and networks
2. **Flexible** firewall target configuration supports the need for complex cloud deployment protection
3. **Firewall Policies** containing multiple firewall rules as a single object attachable to Organization and Folders



Situation: dynamic environment, project owner setting rules, central team cannot control consistency, hard to catch or prevent leak → resort to full control via central team with no delegation and lengthy approval process

Solution: hierarchical firewall policy with exceptions, policy IAM control for safe delegation and manage @scale

- Traditional Firewall Rules -- VPC level → Many projects and VPCs → separate Rule Sets → operational complexity,
- Project owner → independence, flexibility, dynamic cloud environment
- Challenge for corporate security team:
  - common rule mgmt and update
  - project owner config FW → knowledge
  - Balance -- dynamic & safety, control & delegation

- Hierarchical firewall policies are introduced precisely to address this pain point

- User can define and attach firewall policies to higher level in the hierarchy -- org and folder
- Firewall Policy → away from individual rules → define then attach, multiple node, batch update → separate IAM between definition and attachment
- Hierarchical attachment → strict top-down enforcement sequence → IAM control → Project owner cannot overwrite or bypass
- New project and New network → Auto protected from Day 1

- Policy covers Org or Folder → Not to all Instances → Target Service Accounts across the entire Org → eg. Egress traffic allowed only for Proxy servers

- Effective Firewall rules → GUI and API

Hierarchical firewall rules key points

Effective Rules	Rules are enforced in a “Top-down” fashion using first match. Org - Folder -VPC
Network Administration	Hierarchical firewall rules are managed at an the organization level with Org level permissions
Quotas and Limits	2000 Rule Attributes in a HFW Policy. 50 unused policies per Org, but no max number for associated policies
Design Complexity	Typically results in fewer firewall rules for an organization
Flexibility	Associated at the Org or Folder level, but can target VPCs or service accounts

Example use cases:

- Blocking Internet Ingress or Egress traffic in all VPCs at org or folder level
- Blocking a specific protocol (for example, SMTP) for Ingress or Egress in all VPCs at org or folder level
- Defining protocols or ranges reachable from any VPC (for example, Google APIs, Google Load Balancers Health checks, shared services in Google Cloud or on-premises.)

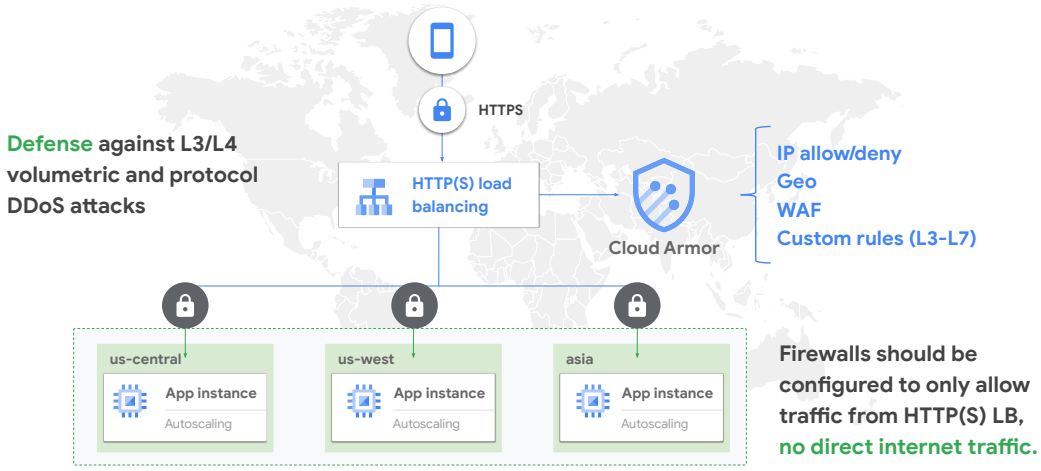
Tags for firewalls

Tags

- Used to define sources and targets in network firewall policies
- Not supported for VPC Firewall rules or hierarchical network policies
- Key:value format
- Sometimes referred to as secure tags
- NOT the same as network tags
- Subject to IAM controls (unlike network tags)

https://cloud.google.com/firewall/docs/tags-firewalls-overview

Cloud Armor: DDoS protection and WAF



TL;DR / Purpose of the slide:

- Explain approach for **DDoS Protection & WAF** in Google Cloud

Key points:

- As discussed above, when using **External HTTP(s) LB,** you **benefit** from an unrivaled level of **L3/L4 DDoS protection** from a wide variety of attack types.
- **Cloud Armor**
  - **Managed WAF solution**
  - Runs **distributed over Google's edge**
  - Integrates with **External HTTP(s) Load Balancer**
  - **Provides**
    - **Layer 7 and Application layer** protection
    - **Access controls** to backend services based on **IP and Geo**
    - **Security policies** based on **L3 to L7 request and client attributes**
    - **Request throttling**
    - **Pre-configured** rules (XSS and SQLi based on OWASP Modsecurity)
    - **Real time telemetry** in the form of **Cloud Operations logs** containing Cloud Armor's decisions on a per-request basis, as well as a **monitoring dashboard** that gives granular views of

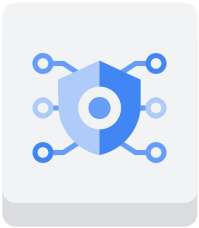
        - allow, denied, or previewed traffic.
    - **Security policies and IP deny list** are **not supported with Cloud CDN**

Probing questions (optional):

- Are you using any WAF product today?

Google Cloud IDS Service

- **Cloud-native managed NG-IDS** for advanced network security and compliance needs
- **Ease of setup and management**, high performance, and availability
- **Advanced Threat Detection powered by Palo Alto Networks** – detect exploit attempts, evasion techniques, port scans, buffer overflows, protocol fragmentation, and obfuscation attempts
- **Alerts in UI and Cloud Logging**, integrations with SIEM/SOAR partners



Google Cloud IDS Service

Google Cloud

Simple, cloud-scale native service with best-in-class infrastructure

+

paloalto NETWORKS

Best-in-class security for advanced threats

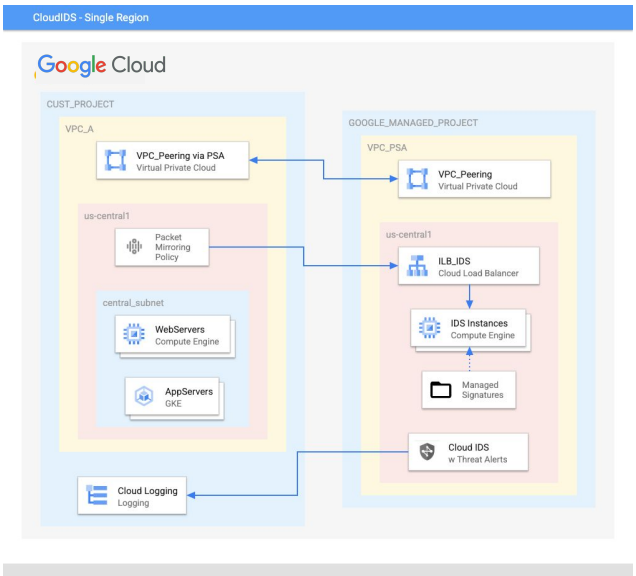
=

Simple, cloud-scale, native, best-in-class security for customers

Cloud IDS

Cloud-native Intrusion Detection System

- Cloud-native network threat detection
- Monitor intra- and inter-VPC communication
- Managed service with high performance and simple deployment



Google Cloud Internet egress options

VMs with public IP

- Static NAT (1:1) solution
- Also allows for direct ingress from Internet

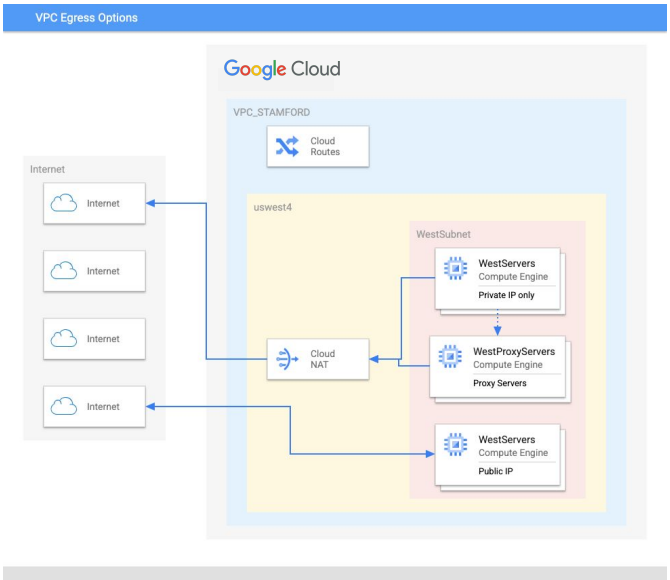
Cloud NAT

- Managed NAT Solution (many to one)

Customer Managed Proxy

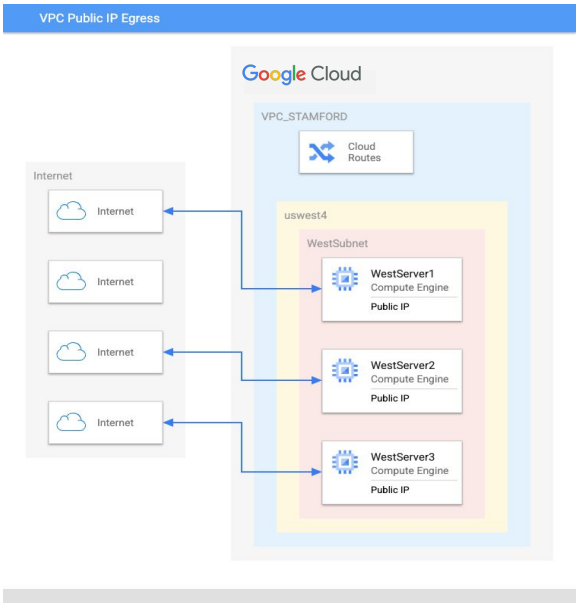
- Allows for advanced filtering and additional egress control

Combination of any/all of the above



Public IPs

- Static 1:1 NAT solution
- Bidirectional: Allows for both egress and ingress traffic
- Public IP can be ephemeral or reserved (static)



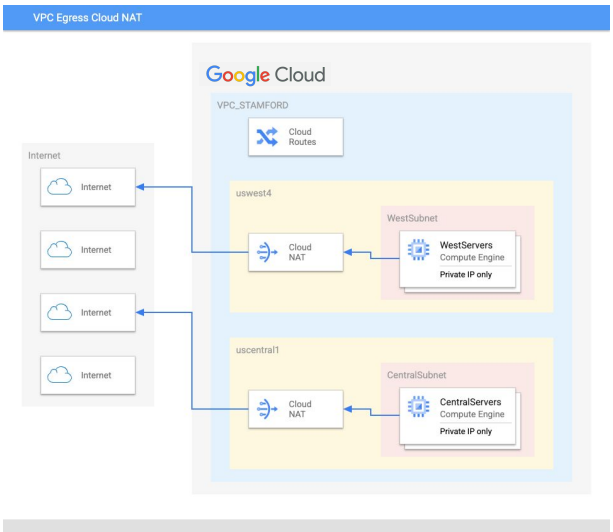
TL;DR / Purpose of the slide:

Key points:

- Allows **outbound and inbound connections to the Internet.**
- **Regional**
- Ephemeral or reserved static IPs
- **Relies on default internet route** in the VPC
- Cases **when NAT is not performed** even when configured:
  - VPN or Interconnect
  - Default internet route has changed
- Still supports ingress and egress Cloud Firewall filtering

Cloud NAT

- Managed (Source) NAT solution.
- Improved security, only outbound connections to the Internet for those resources using “Default Internet Gateway”
- Scales seamlessly
  - Static IPs
  - Auto-allocated IPs
  - Regional
  - Single NAT gateway scales to thousands of VMs



TL;DR / Purpose of the slide:

- Use **Cloud NAT** to **allow Google Cloud VMs without external IP addresses** and private Google Kubernetes Engine (GKE) clusters to **connect to the Internet**

Key points:

- Allows **outbound connections only to the Internet.** Inbound traffic is allowed only if it is in response to a connection initiated by an instance.
- **Regional**
- **Managed service** that provides **high availability** and **seamless scalability**
- **Software defined and fully distributed**
  - **No intermediate NAT proxy** in the data path
  - **NAT configuration** is stored in the control plane and is **pushed to the hosts**. This means NAT keeps working regardless of the control plane state.
  - **No choke points** or performance penalty.
- Support **alias IPs** on VMs
- **Relies on default internet route** in the VPC
- Cases **when NAT is not performed** even when configured:
  - VMs has an external IP
  - Default internet route has changed

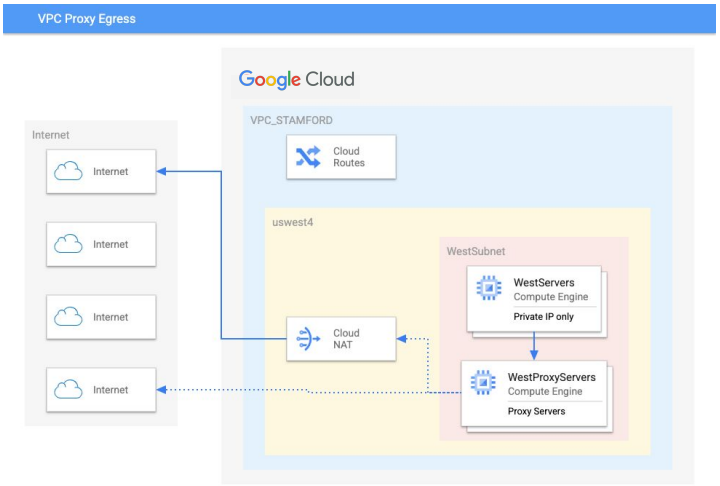
    - Communication between backend VMs and the LB proxy
    - Communication with Google APIs (Private Google Access is used instead)
- Doesn’t support hostname or URL based filtering, use customer managed proxy instead

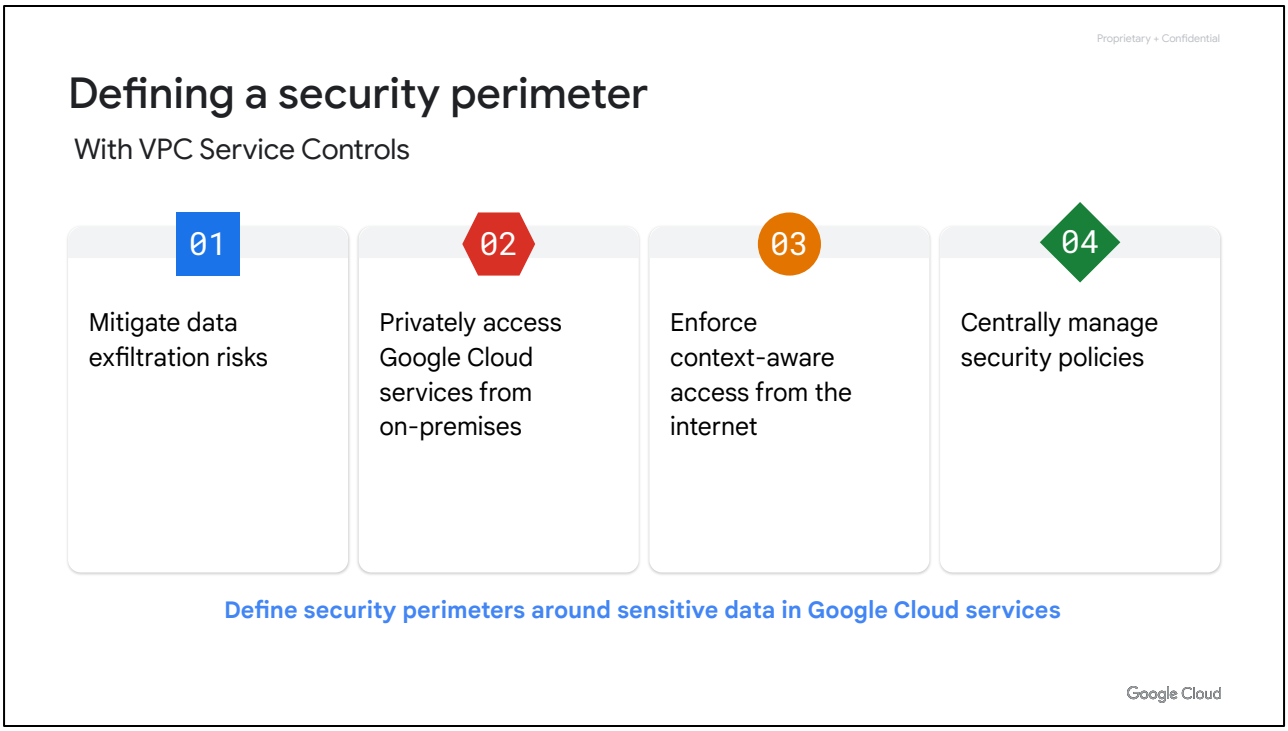
Probing questions (optional):

- Do you have any use cases for using NAT? If so, what are they?

Customer - managed Internet egress proxy

- Customer Controlled (Source) NAT solution using third-party instances such as Squid or PaloAlto FWs
- Custom increased security, for higher layer inspection such as URL and domain name access control
- Proxy servers can use Public IPs or utilize Cloud NAT for egress





TL;DR / Purpose of the slide:

- **VPC Service Controls** helps **preventing data exfiltration** and **controlling access to Google APIs**

Key points:

- So far we have discussed protecting our **VPC network**. It is not less important to **protect data** stored in **Google services**, such as **Cloud Storage** and **BigQuery**.
- While this is **not purely a networking topic**, decisions regarding use of VPC SC **might influence the network design**, which is why we discuss it here.
- **Overview**
  - Provides an **additional layer of security** for Google Cloud services, **on top of IAM.** This is a **broader context-based layer**, compared to IAMs **granular identity-based layer.**
  - This is done by **defining a security perimeter** around **Google services** and controlling the **movement of data** across the perimeter.
  - **Org-level management**
  - Note that **most services are supported**, but not all. See full list.
  - Perimeters can be **extended** to include **authorized VPN or interconnected networks.**
- **Benefits**

- **Protect against stolen credentials**: In case of stolen credentials (for example. service account key), access from outside of the perimeter and it's authorized networks will not be allowed.
- **Preventing data exfiltration**: Prevent accessing or egressing data outside of the perimeter and authorized networks. For example, a malice actor cannot copy Cloud Storage objects to a remote network or Google resource such as a bucket outside of the perimeter.
- **Additional protection in case of misconfigured IAM**: Denying access from unauthorized networks in case of an IAM misconfiguration.

Probing questions (optional):

- None

VPC Service Controls versus VPC Firewall

	VPC Firewall	VPC Service Control
Control path	VM ⟷ VM VM ⟷ Internet VM ⟷ On-premises	Google Cloud Service ⟷ VM Google Cloud Service ⟷ Internet Google Cloud Service ⟷ On-premises Google Cloud Service ⟷ Google Cloud Service
Conditions	5-tuple VM tags VM service accounts	VPC network Google Cloud project Service accounts Internet IPs
Policies apply to	VMs in a VPC network VMs grouped by tag VMs grouped by service account	API based resources grouped by project

TL;DR / Purpose of the slide:

- Comparing VPC Firewalls and VPC Service Controls

Key points:

- It might be **easy to confuse VPC SC and Firewalls**. Before discussing VPC SC more closely, I'd like to **quickly compare the two.**
- **VPC Firewall** is an infrastructure mechanism, and accordingly controls access to/from **VMs,**
- **VPC SC**
  - Essentially controls the path to and from **Google Cloud services**, including between Google Cloud services.
  - **Perimeters** are defined on **projects and Google services.**
  - **Authorization** can be given to **service accounts, public IP ranges, VPC networks** and **bridged with projects.**

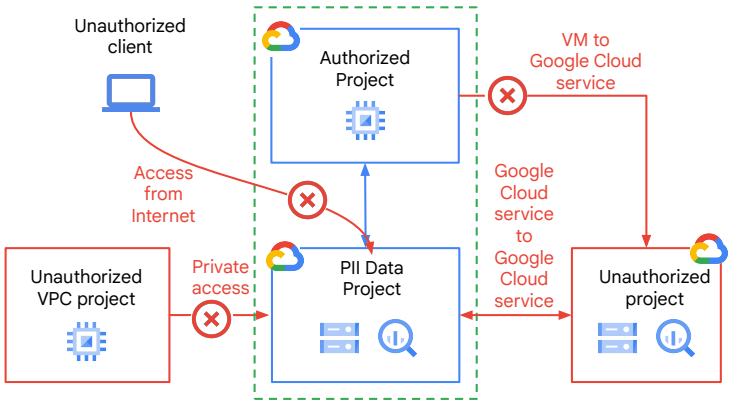
Probing questions (optional):

- None

VPC Service Controls

Service perimeter
Extend perimeter security to managed Google Cloud services

- Control VM-to-service and service-to-service paths
- Ingress: prevent access from the unauthorized networks
- Egress: prevent copying of data to unauthorized Google Cloud Projects
- Project-level granularity



TL;DR / Purpose of the slide:

- Discuss **service perimeters** with an example

Key points:

- **Service perimeters**
  - Defined by **adding projects and services** to include within the perimeter.
  - A **project** can belong to **one perimeter only.**
- In this example, we have a **service perimeter** around **two projects**, including a **VPC network, Cloud Storage and BigQuery.**
- This means that
  - **VMs in the VPC network** can only reach **services within the perimeter.**
  - Accordingly, the **services** within the perimeter **can only communicate with each other**.
- Let's see which **risks** this helps protecting against
  - **Stolen identity / misconfigured IAM**
  - **Data exfiltration through VM**:
    - A malice actor exploiting a **vulnerability in a VM** within the perimeter, attempts to **copy data from the VM** to their own external bucket.
    - The same malice actor can try to copy data **using Cloud**

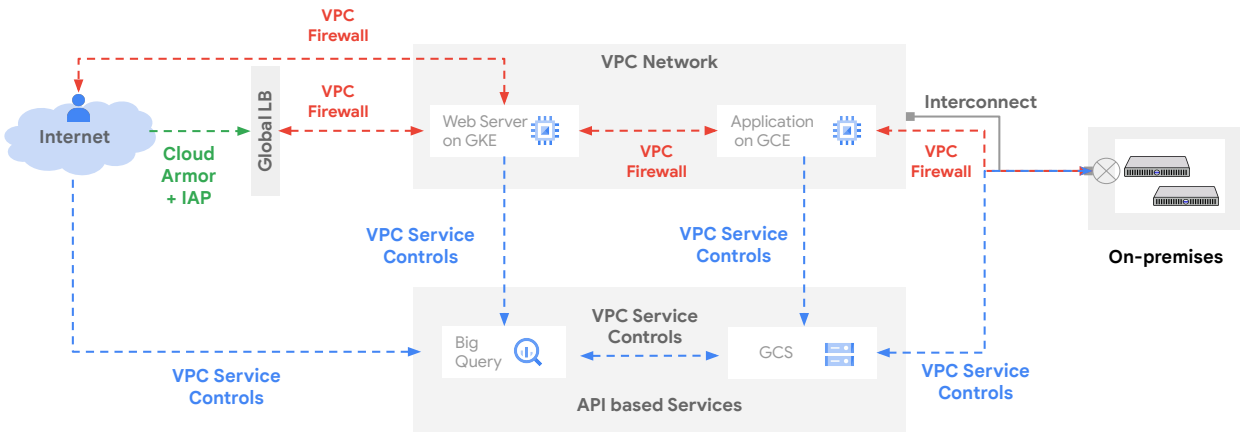
        - **Storage or BigQuery built-in commands.**
        - Both attempts would be **blocked by VPC SC**, but **wouldn't be blocked by a VPC FW**.
    - Access to BigQuery / Cloud Storage from **VM outside of the perimeter**
- Additionally, you could **allow access** to the perimeter **based on context,** for example:
    - Access from on-premises networks during working hours
    - Access from specific geo-locations

Probing questions (optional):

- None

VPC Service Controls

Example architecture



TL;DR / Purpose of the slide:

- Discuss an example including **VPC SC** and **VPC Firewalls**

Key points:

- None

Probing questions (optional):

- None

Identity-aware proxy

Central enforcement

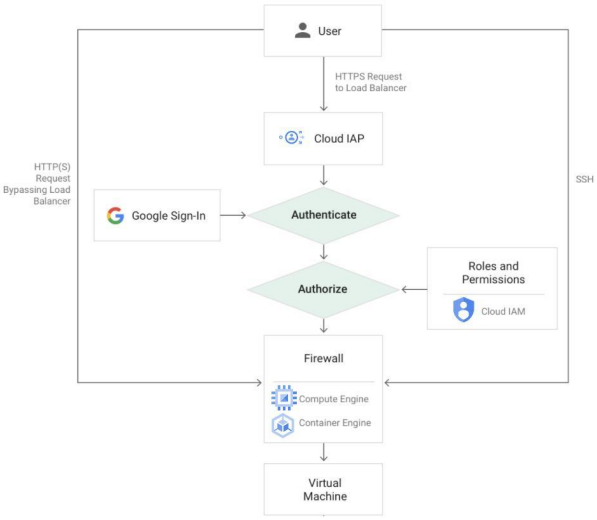
- Single point of control for managing user acces
- Security team can define and enforce policy

Access control

- Control access by user identity
- Apply policy by group membership
- Supports 2FA Security Keys

Deployment

- Little to no change to applications
- No need to implement own authentication for e application
- Integrated with HTTP(s) Load Balancer



TL;DR / Purpose of the slide:

- Discuss **Identity-Aware Proxy**

Key points:

- Cloud IAP provides a **much simpler administration process** and with **reduced operational overhead** than more **traditional VPN** solutions.
  - There is **no VPN to implement** or VPN clients to install and maintain.
  - It also makes the end **user experience more streamlined** as the user no longer has to launch the VPN client and sign into the VPN.
- **How it works**
  - Enables an **app-level access control model** instead of a **network-level control model**
  - Access only possible through **IAP** by users with the **right IAM role**
  - **Authentication** with **Google**, including **2FA**
  - **Authorization** with **Cloud IAM** - users need *IAP-secured Web App User* on the resource project
  - Simple **integration** with **HTTPS Load Balancer** for **Compute Engine/GKE**
    - **Little** to no app/code **change required**
- **Best practice:** apply **VPC firewalls** to **ensure traffic** goes through the **LB**
  - **IAP** only **protects traffic** going **through the LB**

    - **App-serving ports** are still **accessible**

Probing questions (optional):

- None

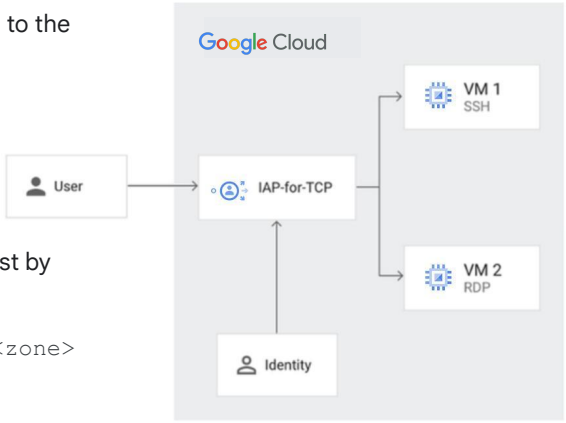
Identity-aware proxy for TCP

- Tunnel TCP traffic to instances without exposing them to the public internet
- Traffic between client and IAP is wrapped in HTTPS
- Access controlled by user identity and IAM

SSH Common pattern

- IAP for TCP can be easily used instead of a bastion host by using Cloud SDK:

```
gcloud compute ssh user@instance --zone <zone>
```



TL;DR / Purpose of the slide:

- Discuss **Identity-Aware Proxy for TCP**

Key points:

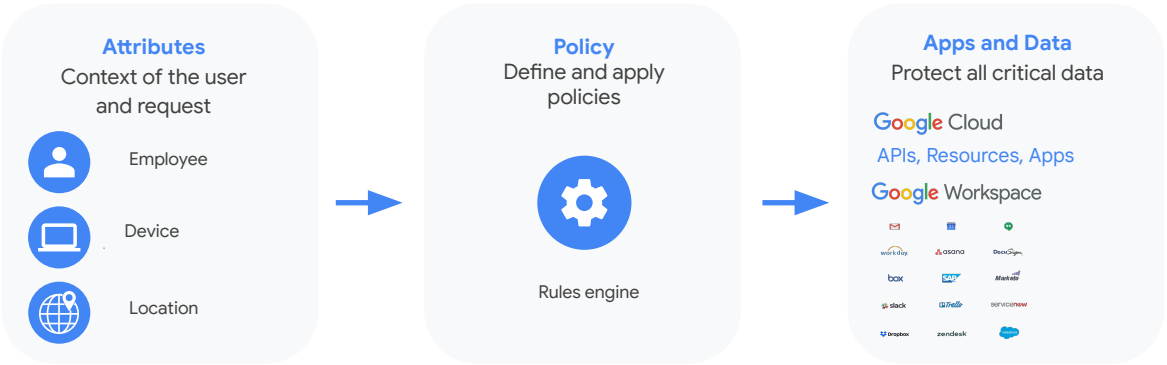
- **IAP** can be used to **tunnel TCP traffic** to instances **without assigning an external IP**
- Also in this case access is authenticated with Google and authorized with IAM
- **SSH (or RDP)** is a very **common use case**
  - Bastion host replacement
  - **Simple integration** with **Cloud SDK**. SSH to a VM with no external IP by running *gcloud compute ssh user@instance --zone <zone>*

Probing questions (optional):

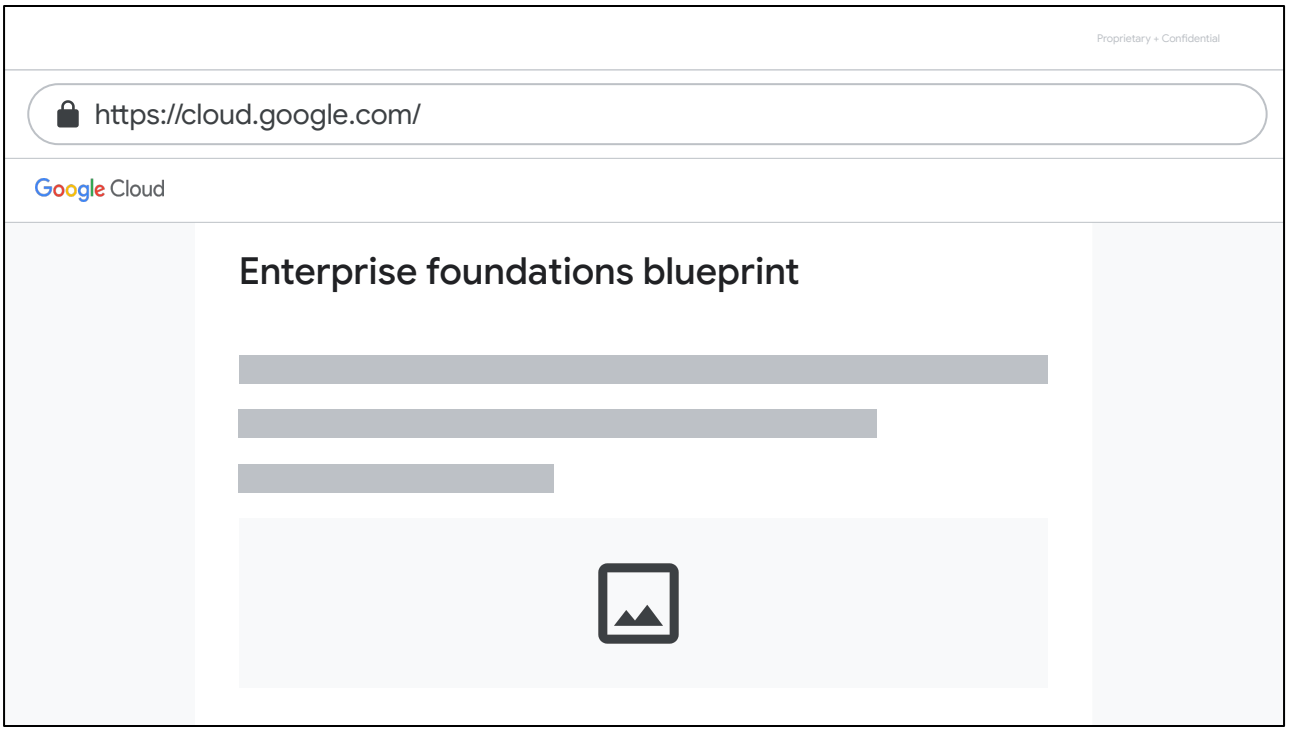
- Do you need to allow SSH access to VMs from the public internet?

- Did you intend on having a bastion host for this purpose?

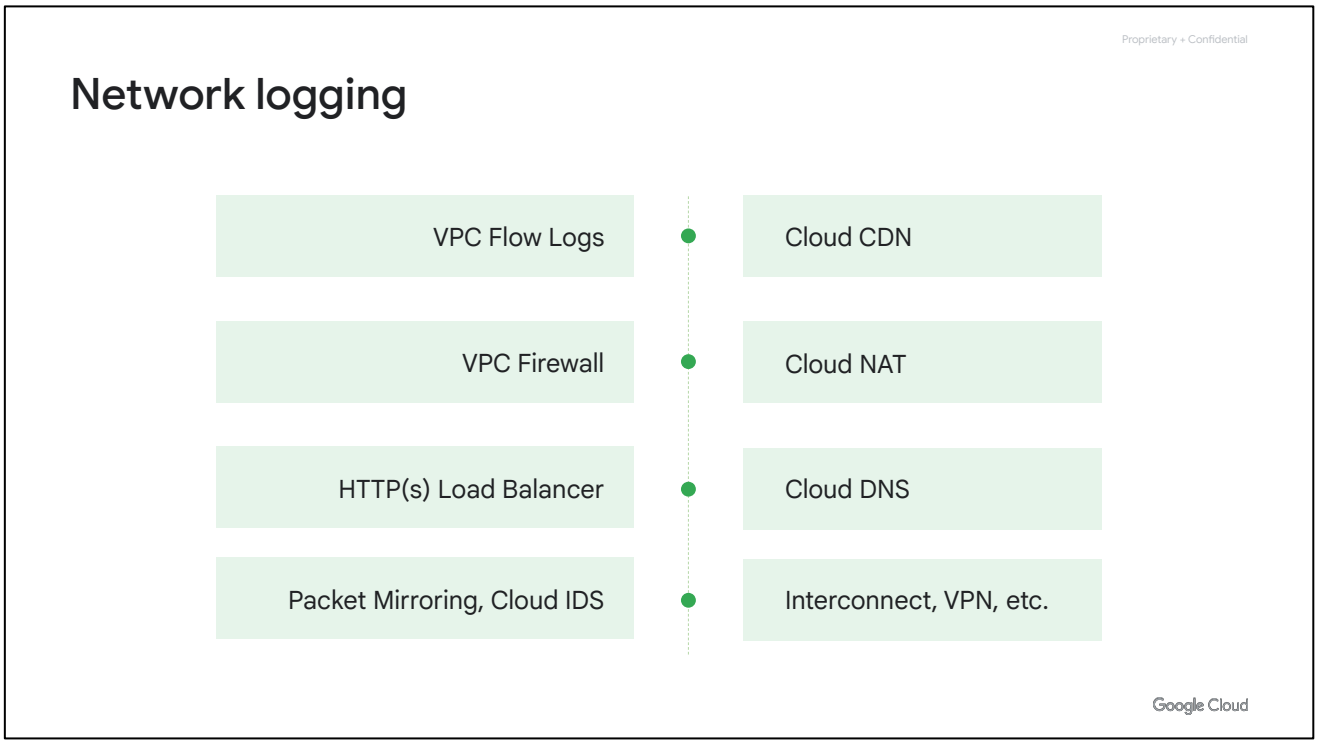
Google Cloud IDS Service



User + Device + Context is the new security perimeter



For more information about security architectures, visit the document titled: Enterprise foundations blueprint.



TL;DR / Purpose of the slide:

- Quick review of possible networking logging options (not complete list, just examples)

Key points:

- This will be discussed more in details under **Monitoring and Logging topic**

Probing questions (optional):

- Which of the following logging options are you interested in?

Mitigating DDoS attacks

Mitigating DDoS attacks is a shared responsibility between Google and you. You should consider:

Attack surface	Reduce the attack surface on Google Cloud by reducing externally facing resources
Internal traffic	Isolate internal traffic from the outside world
Load balancing	Use proxy-based load balancing to distribute load across resources
Scaling	Ensure that your apps scale well to handle the increased load
CDN offloading	Offload static content to a CDN (such as Cloud CDN) to minimize impact
DDoS protection	Deploy DDoS protection (such as Cloud Armor) if necessary
Rate limits and quotas	Be aware of the role API rate limits and resource quotas play in protection against DDoS

TL;DR / Purpose of the slide:

- Discuss the high-level approach for mitigating DDoS attacks in Google Cloud

Key points:

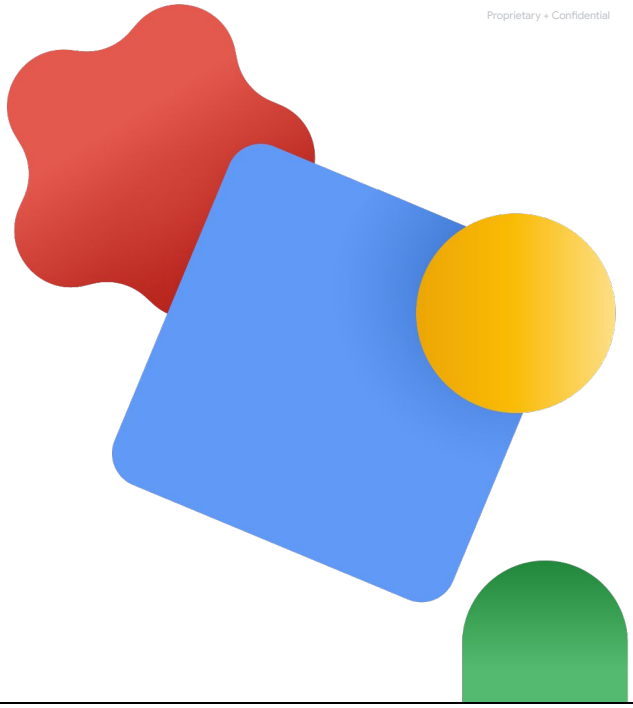
- DDoS mitigation is done in **multiple layers**, and is a **shared responsibility**.
  - **Attack surface and internal traffic**
    - We want to make sure the **external attack surface is minimal**, and is narrowed down to **mainly load balancers** exposing services externally.
    - Minimise the attack surface by making sure **an external IP** is only assigned to VMs that actually need it.
    - Apply **org policies** to restrict usage of **external IPs**, and make use of **NAT** where needed.
    - Isolate traffic using **FW rules**.
  - **Load balancing**
    - When the attack surface is narrowed down, most resources should be accessible only through load balancers.
    - Where possible, make use of **proxy-based external load balancers** (HTTP(s), TCP and SSL).

        - That offers **L3/L4 DDoS protection** at the edge, **before** it reaches **your backends**, similar to how Google protects its own services.
    - **Scaling**
        - Load balancer will **split the load** across all **instances globally,** which will help **absorbing DDoS attacks** that make it to the backends.
        - Consider **configuring auto scaling limitations** to balance between **billing** due to absorption of DDoS attacks, and **service availability.**
    - **CDN offloading**
        - Make use of CDN to absorb DDoS on static content at the edge.
        - Some CDN providers offer DDoS protection as wells.
    - **DDoS protection**
        - **Cloud Armor** is a great compliment to load balancers, being deployed at the edge.
            - **Blacklist** access from **DDoS sources**.
            - Some **CDN** providers offer **DDoS protection**. When services are exposed through them, **only whitelist CDN IPs** to enforce DDoS mitigation through CDN providers. Make sure to **whitelist newly added IPs** by the CDN provider.
        - Consider deploying **third-party tools**: WAFs, next-gen FWs, DDoS protection. See list of security partners
        - Some of these partners offer **integration with Cloud Armor**, by creating **deny rules in real time** to **stop traffic at the edge.**
    - **Rate limits and quotas**
        - There is a **rate limit** for Compute Engine **API calls,** and **quotas on resources**, both on a **project level.**
        - **Both** play a role in **protecting against DDoS** attacks by **preventing unforeseen spikes** in usage.
- **Publicly available article**: Best Practices for DDoS Protection and Mitigation on Google Cloud

Probing questions (optional):

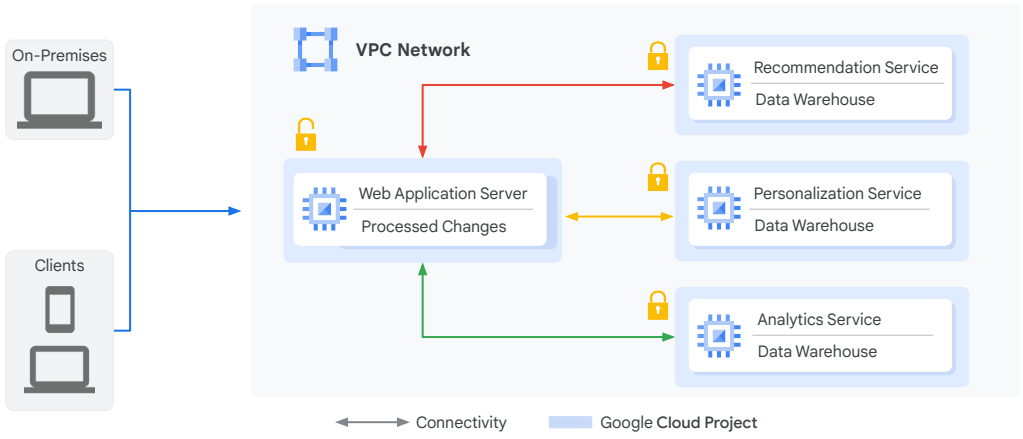
- None

VPC Design Options



BREAK SLIDE

Shared VPC

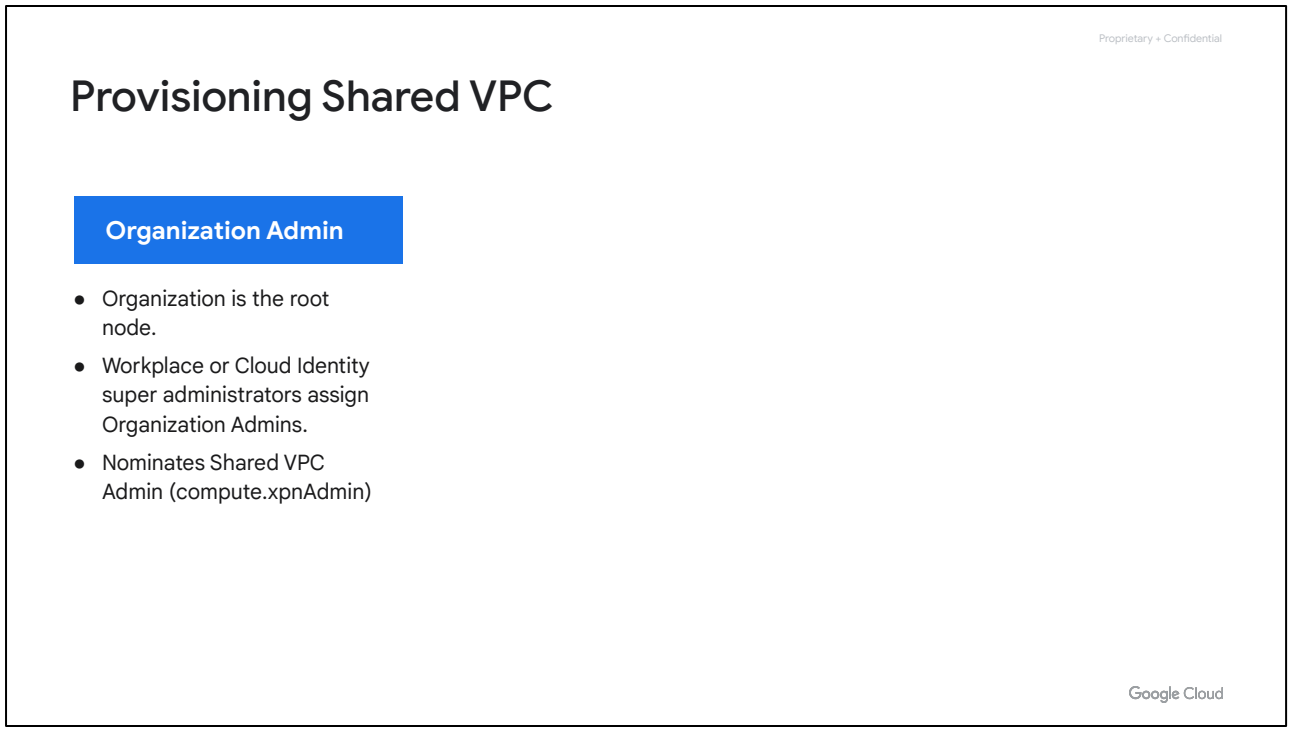


Shared VPC Networks allows an organization to connect resources from multiple projects to a common VPC network. This allows the resources to communicate with each other securely and efficiently using internal IPs from that network.

For example, in this diagram there is one network that belongs to the Web Application Server's project. This network is shared with three other projects, namely the Recommendation Service, the Personalization Service, and the Analytics Service. Each of those service projects has instances that are in the same network as the Web Application Server, allowing for private communication to that server, using internal IP addresses. The Web Application Server communicates with clients and on-premises using the server's external IP address. The backend services, on the other hand, cannot be reached externally because they only communicate using internal IP addresses.

When you use shared VPC, you designate a project as a host project and attach one or more other service projects to it. In this case, the Web Application Server's project is the host project, and the three other projects are the service projects.

The overall VPC network is called the shared VPC network.



Shared VPC makes use of Cloud IAM roles for delegated administration.

Let me walk through how to provision shared VPC by focusing on the required administrative roles.

The first required role is the organization admin. The Organization resource represents an organization, for example, a company, and is the root node in the Google Cloud resource hierarchy.

The Google Workplace or Cloud Identity super administrators are the first users who can access the organization, and they assign the organization admin role to users.

The organization admin's role in provisioning shared VPC is to nominate Shared VPC Admins by granting them appropriate project creation and deletion roles, and the compute.xpnAdmin role for the organization.

Note that shared VPC is also referred to as "XPN" in the API and command-line interface.

Provisioning Shared VPC

Organization Admin

- Organization is the root node.
- Workplace or Cloud Identity super administrators assign Organization Admins.
- Nominates Shared VPC Admin (compute.xpnAdmin)

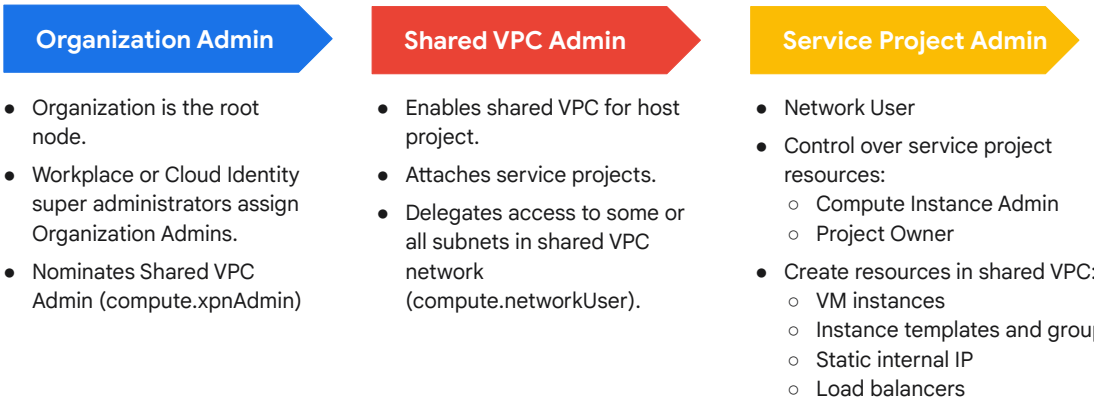
Shared VPC Admin

- Enables shared VPC for host project.
- Attaches service projects.
- Delegates access to some or all subnets in shared VPC network (compute.networkUser).

Next, the Shared VPC Admin performs various tasks necessary to set up shared VPC, such as enabling shared VPC on the host project, attaching service projects to the host project, and delegating access to some or all of the subnets in shared VPC networks to Service Project Admins by granting the compute.networkUser role.

Typically, a Shared VPC Admin is also the project owner for a given host project.

Provisioning Shared VPC



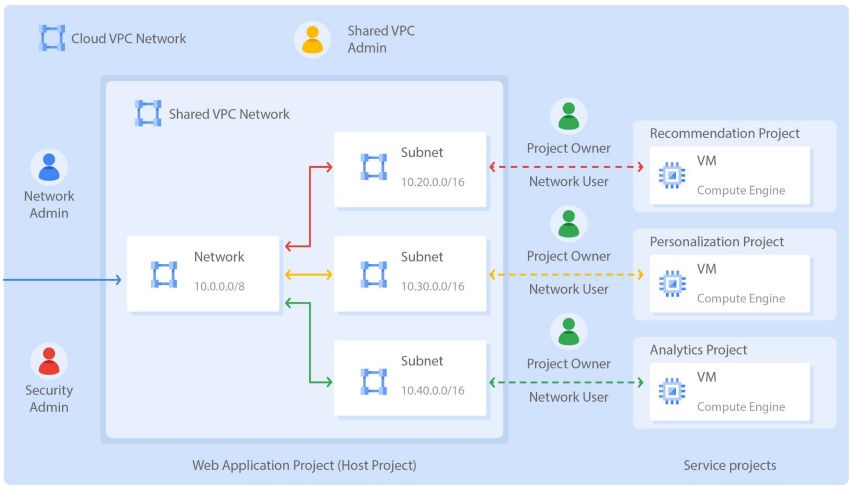
In addition to being a Network User, Service Project Admins also maintain ownership and control over resources defined in their service projects.

The Service Project Admins must at least have the compute.instanceAdmin role to the corresponding service project. However, typically, the Service Project Admins are project owners of their service projects. This allows them to create and manage resources in the shared VPC.

These resources could be:

- VM instances
- Instance templates and groups
- static internal IP addresses
- and load balancers.

Shared VPC



Let's come back to our original example that had one host project and 3 service projects:

In this diagram, the Shared VPC Admin, which was nominated by an organization admin, configured the Web Application Project to be a host project with subnet-level permissions. Doing so allowed the Shared VPC Admin to selectively share subnets from the VPC network.

Next, the Shared VPC Admin attached the three service projects to the host project and gave each project owner the Network User role for the corresponding subnets. Each project owner then created VM instances from their service projects in the shared subnets. By the way, billing for those VM instances is attributed to the project where the resources are created, which are the service projects.

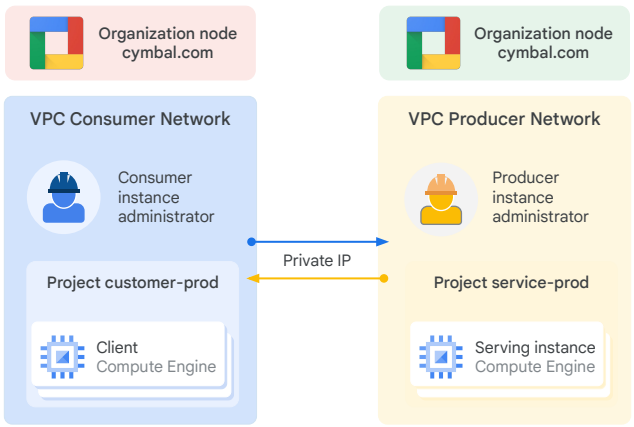
Shared VPC Admins have full control over the resources in the host project, including administration of the shared VPC network. They can optionally delegate the Network Admin and Security Admin roles for the host project. Overall, shared VPC is a centralized approach to multi-project networking because security and network policy occurs in a single designated VPC network.

For a demo on how to create VM instances in a Shared VPC network, please refer

here:
https://storage.googleapis.com/cloud-training/gcpnet/student/M3_Demo_SharedVPC.mp4

VPC Network Peering

- VPC Network Peering allows private RFC 1918 connectivity across two VPC networks.
- Peering works:
  - Across organizations.
  - Within the same project.



VPC Network Peering allows private RFC 1918 connectivity across two VPC networks. Even if both VPC networks belong to the same project or the same organization, you can still peer them.

Remember that each VPC network will have firewall rules that define what traffic is allowed or denied between the peered networks.

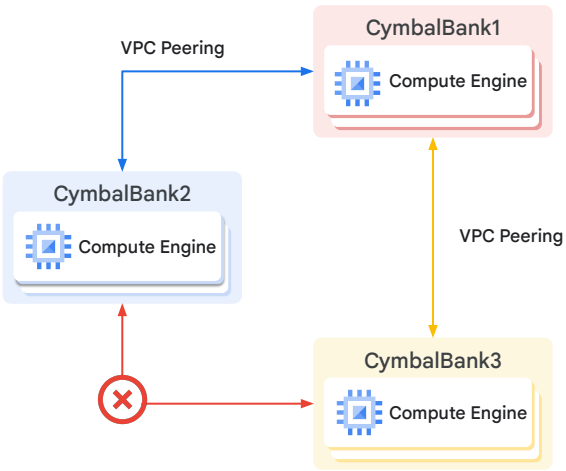
For example, in this diagram, two organizations represent a consumer and a producer, respectively. Each organization has its own organization node, VPC network, VM instances, Network Admin, and Instance Admin. To successfully establish VPC Network Peering, two peering relationships must be created. The Producer Network Administrator must peer the producer network with the consumer network. The Consumer Network Admin must peer the consumer network with the producer network. When both peering connections are created, the VPC Network Peering session becomes active and routes are exchanged. The peering relationships let the VM instance use their internal IP addresses to communicate privately.

VPC Network Peering is a decentralized or distributed approach to multi-project networking. Each VPC network may remain in the control of separate administrator groups and maintains its own global firewall and routing tables. Historically, such projects would consider external IP addresses or VPNs to facilitate private communication between VPC networks. However, VPC Network Peering does not

incur the network latency, security, and cost drawbacks that are present when you use external IP addresses or VPNs.

Using VPC Network Peering

- You can peer Compute Engine, Kubernetes Engine, and App Engine flexible environments.
- Peered VPC networks remain administratively separate.
- Each side of a peering association is set up independently.
- No subnet IP ranges overlap across peered VPC networks.
- Transitive peering is not supported.



Let's talk about a few points to remember when using VPC Network Peering. First of all, VPC Network Peering works with Compute Engine, Google Kubernetes Engine, and App Engine flexible environments.

Peered VPC networks remain administratively separate. In other words, routes, firewalls, VPNs, and other traffic management tools are administered and applied separately in each of the VPC networks. These tools are not managed centrally for all network peers.

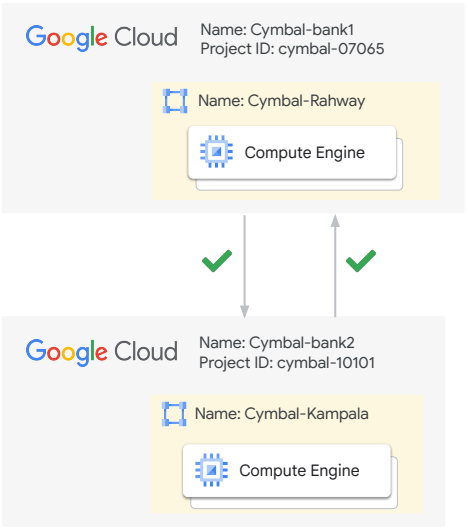
Each side of a VPC Network Peering association is set up independently. Peering will be active only when the configuration from both sides matches. This arrangement allows either side to delete the peering association at any time.

A subnet CIDR prefix in one peered VPC network cannot overlap with a subnet CIDR prefix in another peered network. For example, two auto mode VPC networks that only have the default subnets cannot peer.

Only directly peered networks can communicate, which means that transitive peering is not supported. For example, suppose VPC network CymbalBank1 is peered with CymbalBank2 and CymbalBank3. In that scenario, CymbalBank2 and CymbalBank3 are not directly connected. VPC network CymbalBank2 cannot communicate with VPC network CymbalBank3 over a peered connection. If CymbalBank1 offered SaaS services to CymbalBank2 and CymbalBank3, this situation could be critical.

Initiating VPC Network Peering

- To initiate VPC Network Peering with another VPC network, you need the name of the other VPC network.
- If the VPC network is located in another project, you also need the project ID.
- The peering connection is not active until it's initiated from both VPC networks.



Before you begin, you must have the name of the VPC network to which you will peer with. If that VPC network is located in another project, you must also have the project ID.

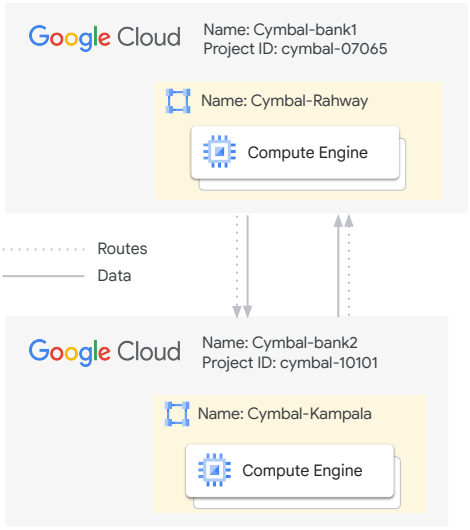
A peering configuration establishes the intent to connect to another VPC network. The VPC networks are not connected until each one has a peering configuration for the other. After the other network has a corresponding configuration to peer with your network, the peering state changes to active in both networks. At that point, the VPC networks are connected through VPC Network Peering.

Until both VPC networks create peering configurations for each other, no peering connection exists between them. The peering state remains inactive. An inactive peering state indicates that there is not yet a full VPC Network Peering connection.

Suppose you want to peer the two VPC networks shown on the slide. Let’s begin with the Cymbal-Rahway VPC network. You must know the name of the other VPC network, which is Cymbal-Kampala. Because Cymbal-Kampala is not in the same project as Cymbal-Rahway, you must also have the project ID. Cymbal-Kampala is in the Cymbal-bank2 project, whose project ID is cymbal-10101.

Sharing custom routes

- Sharing custom routes with peered VPC networks lets networks learn routes directly from their peered networks.
- If a custom route in a peered network is updated, your VPC network automatically learns and uses the updated custom route.
- You can import and export routes.



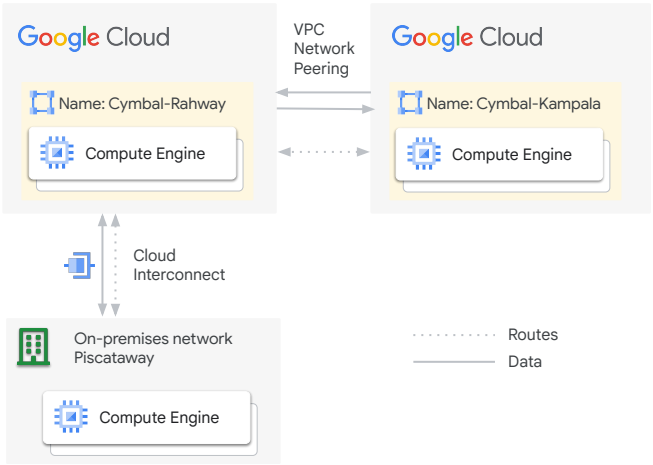
Sharing custom routes with peered VPC networks let networks learn routes directly from their peered networks.

For example, if a custom route in a peered network is updated, your VPC network automatically learns and uses the updated custom route. You are not required to configure any additional settings in your VPC network.

When you create or modify a peering configuration, you can choose to import routes, export routes, or both. The peer network administrator must similarly configure their peering configuration before routes are exchanged. This process ensures that both network administrators explicitly agree to exchange custom routes before they are exchanged.

A sample scenario

Share custom routes so that peered networks can reach your on-premises network.



Let's consider a sample scenario.

If you have a VPN tunnel or a Cloud Interconnect connection, you can share custom routes. Sharing these routes enables peered networks to reach your on-premises network, as shown in the graphic. For dynamic routes, you must add Cloud Router custom route advertisements in your VPC network. These advertisements announce peered network subnets to your on-premises network.

Local routes are always preferred over dynamic routes that are learned by using VPC Network Peering.

Shared VPC versus VPC Network Peering

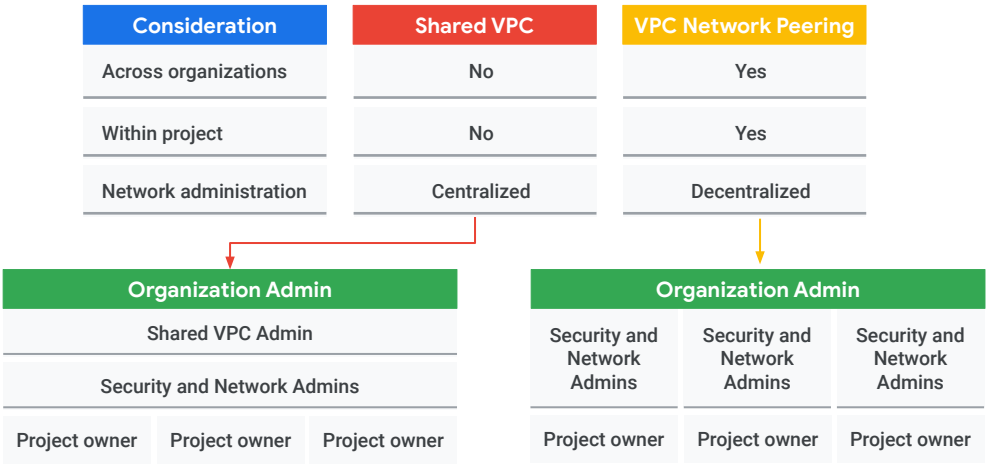
Consideration	Shared VPC	VPC Network Peering
Across organizations	No	Yes
Within project	No	Yes
Network administration	Centralized	Decentralized

Let's compare both of these configurations to help you decide which is appropriate for a given situation.

If you want to configure private communication between VPC networks in different organizations, you must use VPC Network Peering. Shared VPC only works within the same organization.

If you want to configure private communication between VPC networks in the same project, you must use VPC Network Peering. The VPC networks can be in the same project, but it's not required. Shared VPC only works across projects.

Shared VPC versus VPC Network Peering



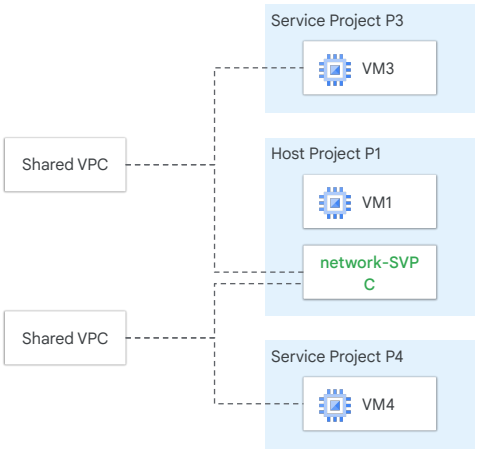
In our opinion, the biggest difference between the two configurations is the network administration models. Shared VPC is a centralized approach to multi-project networking, because security and network policy occurs in a single designated VPC network.

In contrast, VPC Network Peering is a decentralized approach. Each VPC network is controlled by administrator groups in that VPC network's organization. Each VPC network can maintain its own global firewall and routing tables.

For more information about the limits of VM instances per VPC network, see Per Network on the Quotas and Limits page of the Google Cloud documentation.

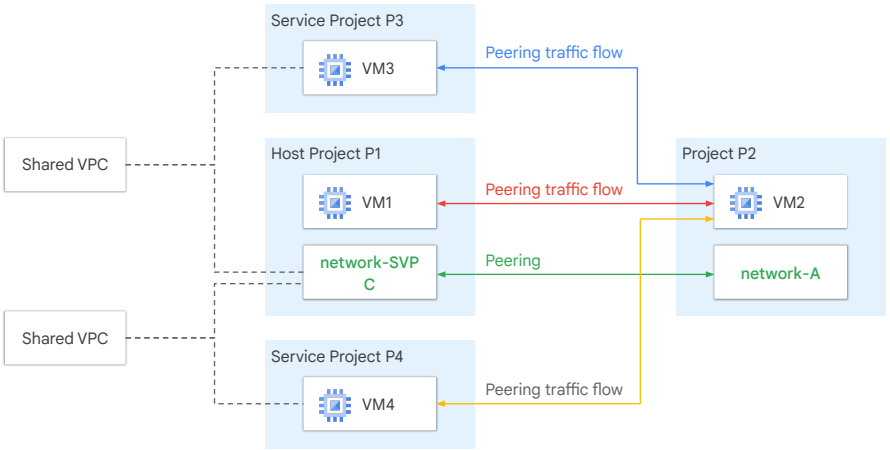
Now that we've compared both of these configurations for sharing VPC networks across Google Cloud projects, let's look at one last use case.

Peering with a Shared VPC network



In Google Cloud, you can peer with a Shared VPC network. Here you can see an example of a Shared VPC network called network-SVPC. network-SVPC is in host project P1. Service projects P3 and P4 can attach VM instances to network-SVPC, which enables private communication between VMs 1, 2, and 4.

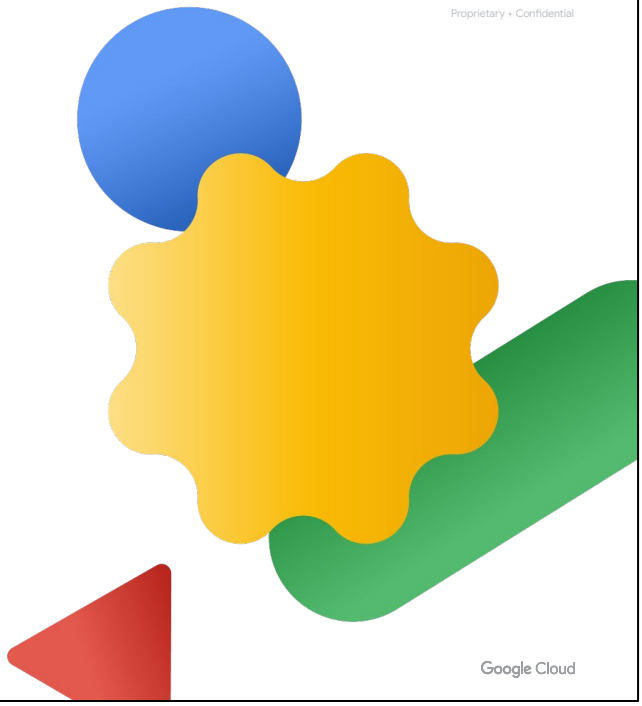
Peering with a Shared VPC network



If we establish a peering session between network-A and network-SVPC, all VM instances will have private, internal IP connectivity. Each VPC network has firewall rules that define which traffic is allowed or denied between the networks.

You can also set up VPC Network Peering between two Shared VPC networks.

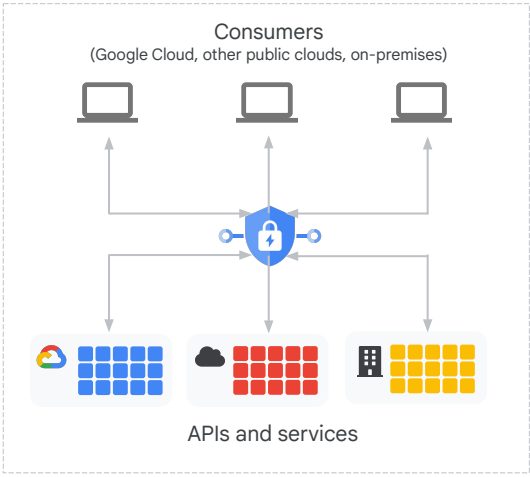
Private Connection Options



BREAK SLIDE

Private access options for Google APIs and services

- Private access refers to the ability to connect to APIs and services locally.
- Private access uses internal IP addresses.
- Access is quicker and more secure.
- Choose a private access option based on your needs.
- All Google Cloud APIs and services support private access.



However, all Google Cloud APIs and services support private access. Private access uses internal IP addresses. Therefore, consumers connect to supported APIs and services with an internal connection. Unless a consumer connects to Google Cloud by using an external connection, private access communication does not go through the public internet.

Access is quicker and more secure.

Choose a private access option based on your needs.

You can also set up private access to APIs and services that you publish. You can access these API and services from Google Cloud, other public clouds, or on-premises.

Google APIs and services have public URLs and are accessible on the public internet.

Private access options

Option	Connection	Usage
Private Google Access	Connect to the public IP addresses of Google APIs and services through the default internet gateway of the VPC network.	Lets you use Google APIs and services without giving your Google Cloud resources external IP addresses.
Private Service Connect	Connect to Google, third-party, or your own services by using internal IP addresses.	Lets you use internal IP addresses to consume, produce, and make services available.
Serverless VPC Access	Connect serverless products to your VPC network to access Google, third-party, or your own services with internal IP addresses.	Lets Cloud Run, App Engine standard, and Cloud Functions connect to the internal IPv4 addresses in a VPC network.
Private services access	Connect Google and third-party services privately and directly to your VPC network with VPC Network Peering.	Lets you use internal IP addresses to connect to specific Google and third-party services by using VPC Network Peering.

Google provides several private access options. Each option allows VM instances with internal IP addresses to reach certain APIs and services. You can configure one or all of these options, because they operate independently of each other.

Private Google Access for on-premises hosts lets your on-premises hosts connect Google APIs and services through the default internet gateway of the VPC network. Your on-premises hosts don't need external IP addresses; instead, they use internal IP addresses.

Private Service Connect lets you connect to a Google or third-party managed VPC network through a service attachment. As with Private Google Access, the connection is internal.

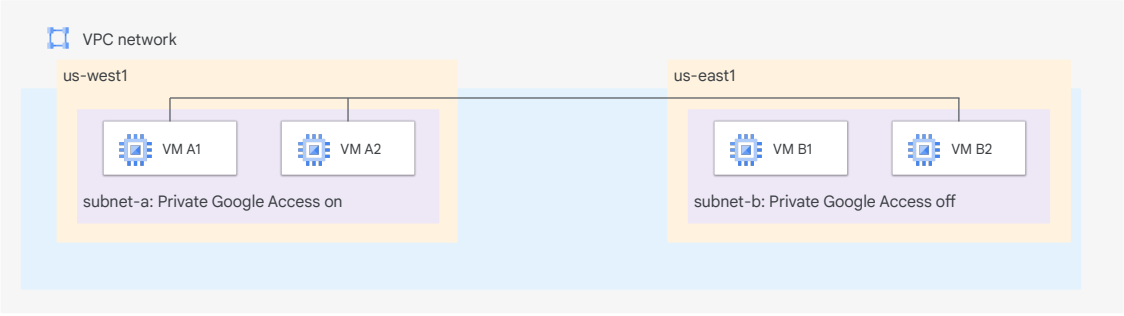
Serverless VPC Access connects serverless products to your VPC network to access Google, third-party, or your own services with internal IP addresses. For example, Cloud Run, App Engine standard, and Cloud Functions environments send packets to the internal IPv4 address of the resource. Serverless VPC Access is not covered in this module. For more information, refer to Connect from serverless Google services to VPC networks in the Google Cloud documentation.

Private services access is a private connection between your VPC network and a service producer VPC network. This connection is implemented as a VPC Network Peering connection. The service producer network is created exclusively for you, and it’s not shared with other customers.

For more information, see Private access options for services in the Google Cloud documentation.

Private Google Access

- Private Google Access is enabled on a subnet-by-subnet basis.
- If you disable Private Google Access for a subnet, VMs with internal IP addresses can only send traffic within the VPC network.
- Private Google Access has no effect on VMs with external IP addresses.

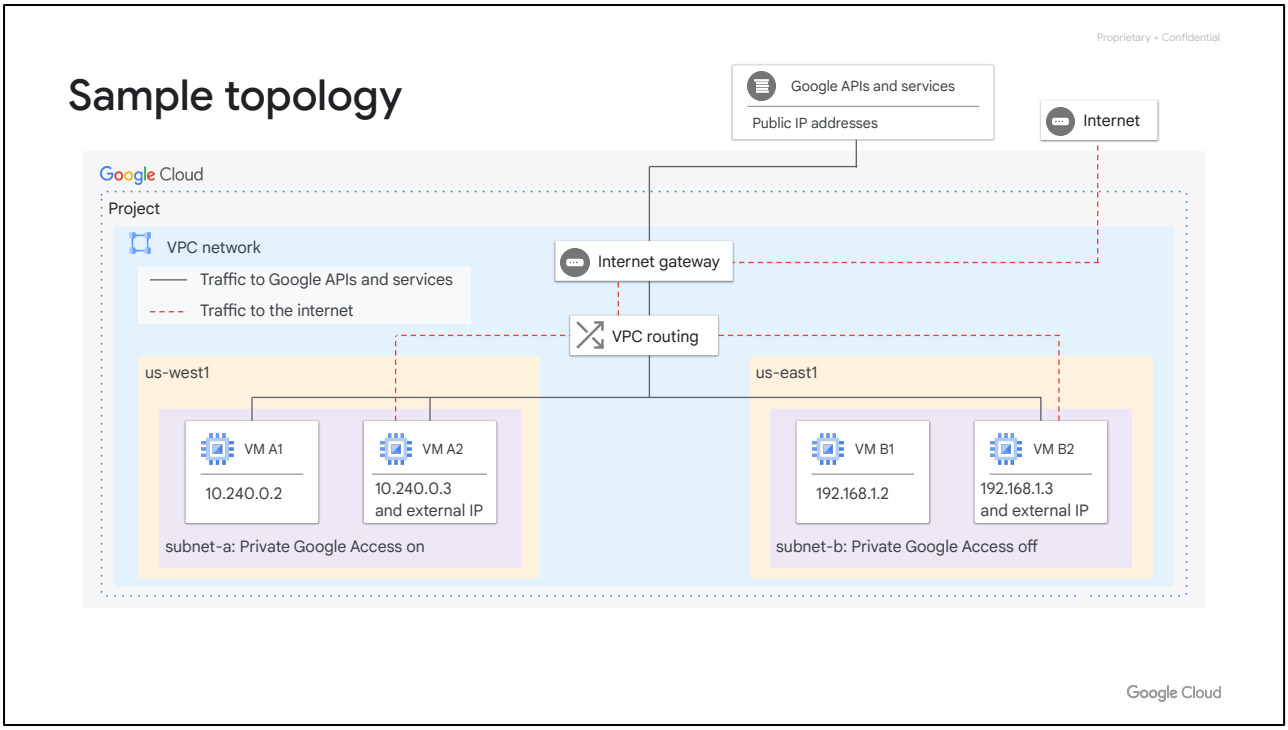


You enable the Private Google Access feature on a subnet-by-subnet basis by editing the subnet in Google Cloud console or the Google Cloud CLI.

If you disable Private Google Access for a subnet, VMs with internal IP addresses can only send traffic within the VPC network. Later in this module, you will learn about Cloud NAT, which can allow these VMs to send traffic outside of the VPC network.

Private Google Access has no effect on instances that have external IP addresses.

For a list of the services that are supported by Private Google Access, see Private access options for services in the Google Cloud documentation.



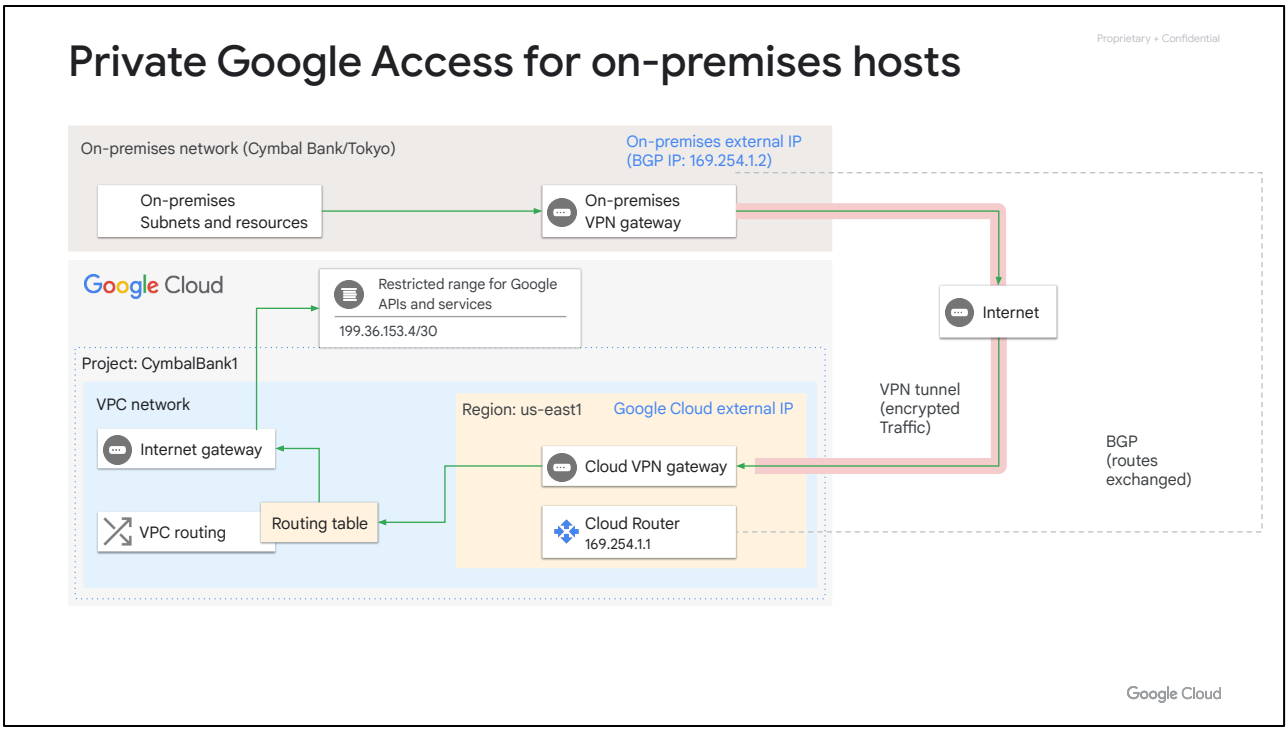
In the sample topology, the VPC network has two subnets: subnet-a and subnet-b. The network has been configured to meet the Domain Name System (DNS), routing, and firewall network requirements for Google APIs and services.

Private Google Access has been enabled on subnet-a, but not on subnet-b.

VM A1 can access Google APIs and services, including Cloud Storage, because its network interface is located in subnet-a, which has Private Google Access enabled. Private Google Access applies to the instance because it only has an internal IP address.

VM B1 can't access Google APIs and services because it only has an internal IP address and Private Google Access is disabled for subnet-b.

VM A2 and VM B2 can both access Google APIs and services, including Cloud Storage, because each of them has its own IP address. Private Google Access has no effect on whether these instances can access Google APIs and services because both have external IP addresses.



Cymbal Bank wants to use internal IP addresses to access Cloud SQL and Cloud TPU from their Tokyo on-premises network. This example shows how this access can be achieved. In the example, the on-premises network is connected to a VPC network through a Cloud VPN tunnel. Traffic from on-premises hosts to Google APIs travels through the tunnel to the VPC network. After traffic reaches the VPC network, it's sent through a route that uses the default internet gateway as its next hop. This next hop allows traffic to leave the VPC network and be delivered to restricted.googleapis.com (199.36.153.4/30).

The on-premises DNS configuration maps *.googleapis.com requests to restricted.googleapis.com, which resolves to the 199.36.153.4/30 address range.

In this example, Cloud Router uses a custom route advertisement for this IP address range. This route sends traffic through the Cloud VPN tunnel. The traffic that goes to Google APIs is routed through the tunnel to the VPC network.

A custom static route was added to the VPC network. This route directs traffic with the destination 199.36.153.4/30 to the default internet gateway as the next hop. Google then directs traffic to the appropriate API or service.

In this example, network administrators at Cymbal Bank created a Cloud DNS managed private zone for *.googleapis.com that maps to the 199.36.153.4/30 address range. The network administrators authorized the VPC network to use that zone. Requests to the googleapis.com domain are sent to the IP addresses that are

used by restricted.googleapis.com. Only the supported APIs are accessible with this configuration, which might cause other services to be unreachable. Cloud DNS doesn't support partial overrides. If you require partial overrides, use BIND (a software that interacts with DNS).

Caveats

Private Google Access that uses IPv6

If you want to use IPv6 to connect to Google APIs and services:

- Your VM must be configured with a /96 IPv6 address range.
- The software running on the VM must send packets whose sources match one of those IPv6 addresses from that range.
- You must send the packets to the IPv6 addresses for the default domains.

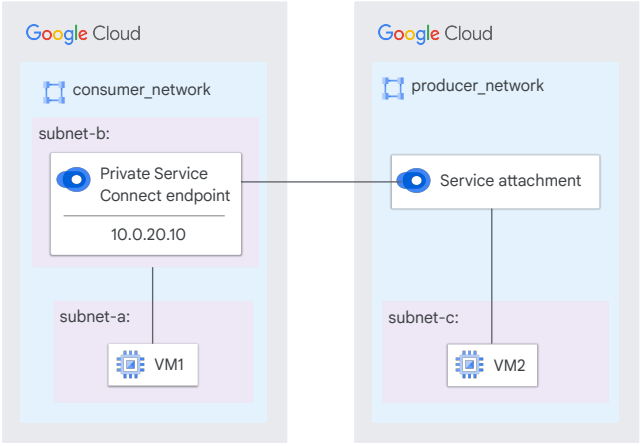
If you want to use IPv6 to connect to Google APIs and services:

Your VM must be configured with a /96 IPv6 address range. The software running on the VM must send packets whose sources match one of those IPv6 addresses from that range.

You must send the packets to the IPv6 addresses for the default domains.

Private Service Connect

- With Private Google Access, Google APIs and services can be accessed with internal IP addresses.
- With Private Service Connect, third-party resources are also accessed with internal IP addresses.
- You can access resources through a Private Service Connect endpoint.
- Private Service Connect is fast and scalable.



As with Private Google Access, you can use Private Service Connect to access Google APIs and services with a global internal IP address. Private Service Connect also lets you access third-party services.

To access resources with Private Service Connect, use a Private Service Connect endpoint. Organizations can choose the internal IP address to associate with each endpoint.

Private Service Connect has line rate performance and scales to enterprise-size networks. In other words, Private Service Connect is fast and grows with your organization.

Look at the example shown on the right. VM1 in the consumer_network uses a Private Service Connect endpoint to connect to services that run in the producer_network on VM2. The Private Service Connect endpoint has an internal IP address, 10.0.20.10. VM1 uses the internal address to access services on VM2.

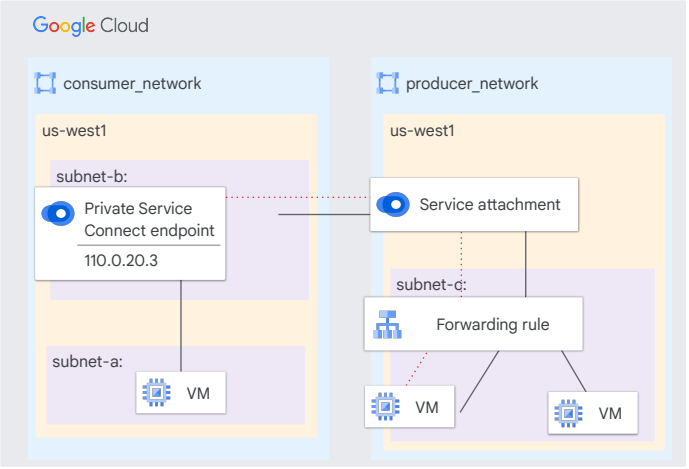
Next, let's talk about other things you can do with Private Service Connect.

Components of Private Services Connect

Consumers	Producers	Service attachments	Endpoint
Access managed services via private IP from within own VPC	• Can expose services to consumers via service attachments	These link to producer load balancers. Security can be applied with a consumer accept list	Private IP addresses in a consumer VPC that are mapped to a service attachment and forward request to the attached service

Using a forwarding rule

- The service attachment:
  - Receives requests redirected from the Private Service Connect endpoint.
  - Sends them to a forwarding rule.
- The forwarding rule sends the traffic to the correct VM or service.

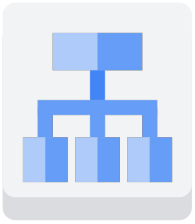


The consumer contacts a producer service or VM by using the Private Service Connect endpoint in their VPC network. This endpoint has an internal IP address and maps to the service attachment in the producer VPC network.

A service attachment refers to services from a producer. In this example, the service attachment receives requests redirected from the Private Service Connect endpoint and sends it to a forwarding rule. The forwarding rule sends the request to the appropriate VM or service. You can see this flow in the example, with the red, dotted line.

What you can do with Cloud Load Balancing

- Rename services and map them to URLs of your choice.
- Configure Cloud Load Balancing to log all requests to Cloud Logging.



Cloud Load Balancing

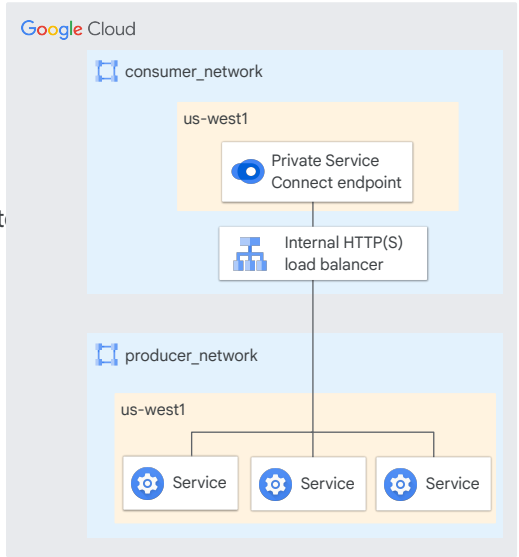
Using a load balancer provides some additional features. You can assign DNS names to these internal IP addresses–or even Google APIs and services–with meaningful names for your organization. For example, if you have a service with the name spanner.example.com, you can map it spanner.cymbal.com or some other name that makes sense for your organization. These names and IP addresses are internal to your VPC network. On-premises networks use Cloud VPN tunnels or VLAN attachments to connect to it

Also, you can control which traffic goes to which endpoint and demonstrate that the traffic stays within Google Cloud.

Using an internal HTTP(S) load balancer

With Private Service Connect and an HTTP(S) load balancer, you can:

- Use a URL map to evaluate requests and route them t the correct VM or service.
- Use customer-managed TLS certificates.
- Enable data residency in-transit by connecting to regional endpoints for Google APIs from workloads in that same region.

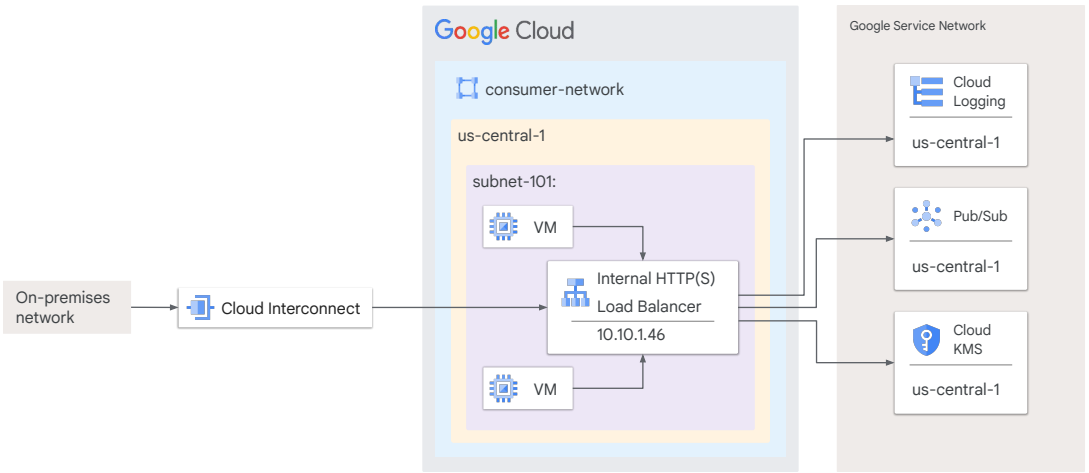


With Private Service Connect and an HTTP(S) load balancer, you can use a URL map to choose which services are available to consumers. You can also use it to evaluate the request and route it to the correct VM or service.

For added security between clients and the load balancer, you can use customer-managed Transport Layer Security (TLS) certificates.

You can also enable data residency in-transit by connecting to regional endpoints for Google APIs from workloads in that same region. In other words, you can be certain that data at rest is stored in the region you configure.

Sample topology

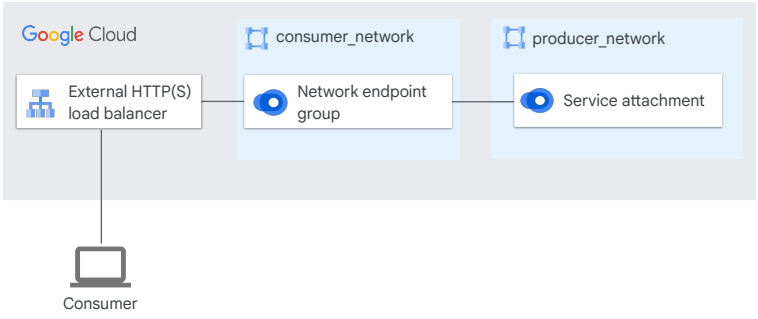


This example shows an on-premises network that is connected to a subnet of a Google Cloud VPC network in the us-central-1 region. The requests that a VM or service in the on-premises network make to Cloud Logging, Pub/Sub, or Cloud KMS are not routed on the public internet. They remain within the Google Cloud backbone network. Requests to other Google Cloud services are routed over the public internet.

The Internal HTTP(S) load balancer in the consumer-network and the desired Google services are both located in the us-central-1 zone. You can see that after the request from the on-premises network is sent through the Cloud Interconnect connection to the load balancer, it remains in the us-central-1 zone. If desired, access through the load balancer can be sent to Cloud Logging.

Using a global external HTTP(S) load balancer

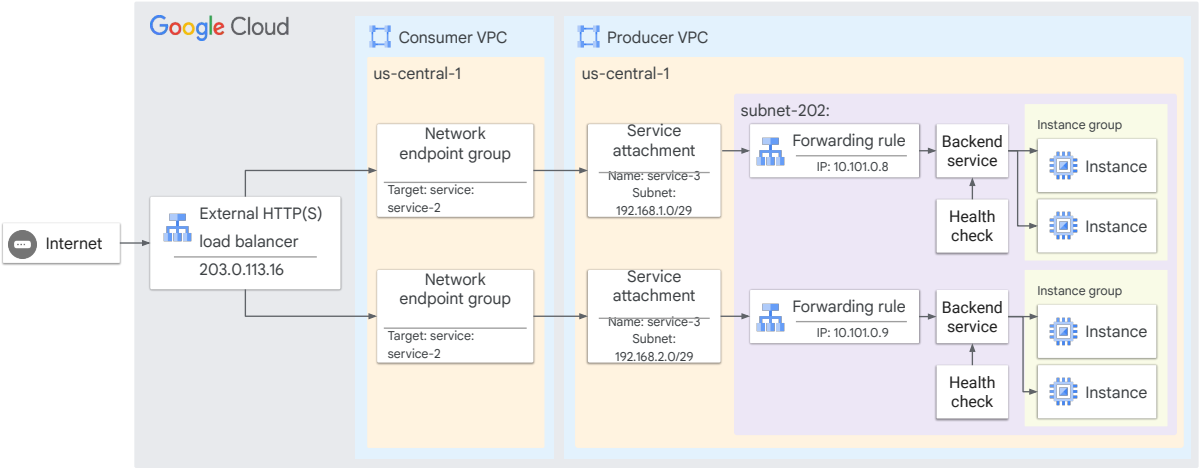
- Consumers connect to an external IP address.
- Private Service Connect uses a network endpoint group to route the request to the service producer.



With Private Service Connect and consumer HTTP(S) service controls that use a global external HTTP(S) load balancer, consumers connect to an external IP address. Private Service Connect uses a network endpoint group to route the request to the service producer.

Let's look at a more detailed example of Private Service Connect that uses a global external HTTP(S) load balancer.

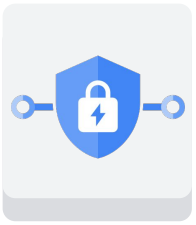
Sample topology



This topology shows a Private Service Connect endpoint based on a global external HTTP(S) load balancer. This topology lets service consumers with internet access connect to the load balancer. The load balancer then directs the requests to the appropriate network endpoint group in a consumer network. Each of these endpoints is associated with a service attachment. A forwarding rule then routes the request to the appropriate VM instance or service.

General benefits

- Except for the global external HTTP(S) load balancer use case, connections use internal IP addresses.
- Traffic stays on the Google backbone network.
- Configuration is simple.



Private Service Connect

Private Service Connect lets you configure access to specific Google and third-party services using internal IP addresses.

Except for the global HTTP(S) load balancer use case, connections use internal IP addresses. Traffic stays on the Google backbone network. Connections are thus more secure and much faster than over the public internet. Services that use Private Service Connect interact like services on a private network,

Configuration is simple. Private Service Connect works with the internal IP address range that you provide and sets up the routing tables.

Benefits for consumers

Consumers:

- Can control the internal IP address that is used to connect to a managed service.
- Do not need to reserve internal IP address ranges for backend services that are consumed in their VPC network.
- Must initiate traffic to the service provider, which improves security.

Private Service Connect is highly scalable, and supports thousands of consumers. Consumers use consumer VPC networks to access VMs and services from producer VPC networks.

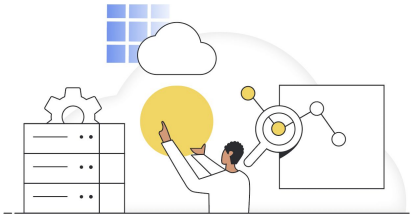
Consumers can control the internal IP address that is used to connect to a managed service. They don't need to reserve internal IP address ranges for backend services that are consumed in their VPC network. Instead, consumers choose an IP address from their own subnet to connect to the producer services.

For security purposes, all communications between the consumer VPC network and service producer VPC network must be initiated by the consumer. Service producers can't initiate this communication. This unidirectional connectivity drastically simplifies firewall configuration, but also reduces risk from rogue traffic originating from the service producer.

Benefits for producers

Producers:

- Can choose to deploy a multi-tenant model; serving multiple consumer VPC networks.
- Can scale services to as many VM instances as required, without asking consumers for more IP addresses.
- Don't need to change firewall rules based on the subnet ranges in the consumer VPC networks.



Service producers make VMs and services available to consumers. Producers can choose to deploy a multi-tenant model, where your VPC network contains services that are used by multiple consumer VPCs. The consumer networks can have overlapping subnet ranges.

Service producers can scale services to as many VM instances as required, without asking consumers for more IP addresses.

Service producers don't need to change firewall rules based on the subnet ranges in the consumer VPC networks. You can simply create firewall rules for the network address translation (NAT) IP address range configured for your service.

Caveats

Private Service Connect

01 You can't create a Private Service Connect endpoint in the same VPC network as the published service that you are accessing.

02 The IP address that you use for the Private Service Connect endpoint counts toward the project quota for global internal IP addresses.

03 Private Service Connect endpoints are not accessible from peered VPC networks.

Private Service Connect has a few caveats.

You can't create a Private Service Connect endpoint in the same VPC network as the published service that you are accessing. The endpoint can only be used to access a published service in another VPC network.

The address counts toward the project quota for global internal IP addresses.

Private Service Connect endpoints are not accessible from peered VPC networks. Instead, create a Private Service Connect endpoint in the peered VPC network. You can then configure workloads to refer to that endpoint.

Connections from on-premises environments to non-Google services must use Cloud VPN tunnels. These on-premises environments must be in the same region as the Private Service Connect endpoint.

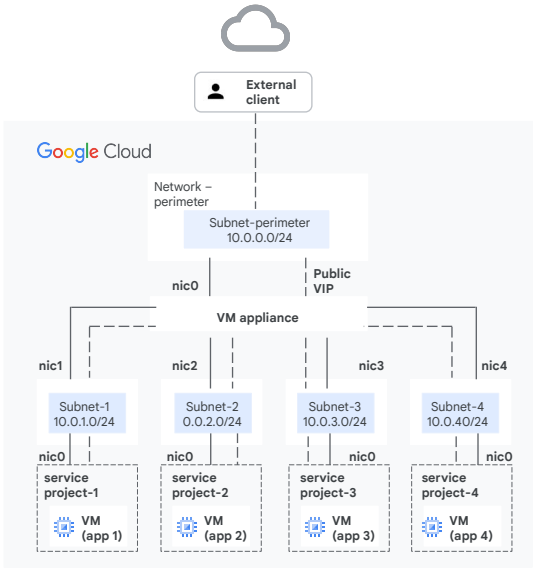
For information about accessing Private Service Connect endpoints from on-premises environments that are connected using Cloud VPN, see Access the endpoint from on-premises hosts in the Google Cloud documentation.

Secure intra-cloud access

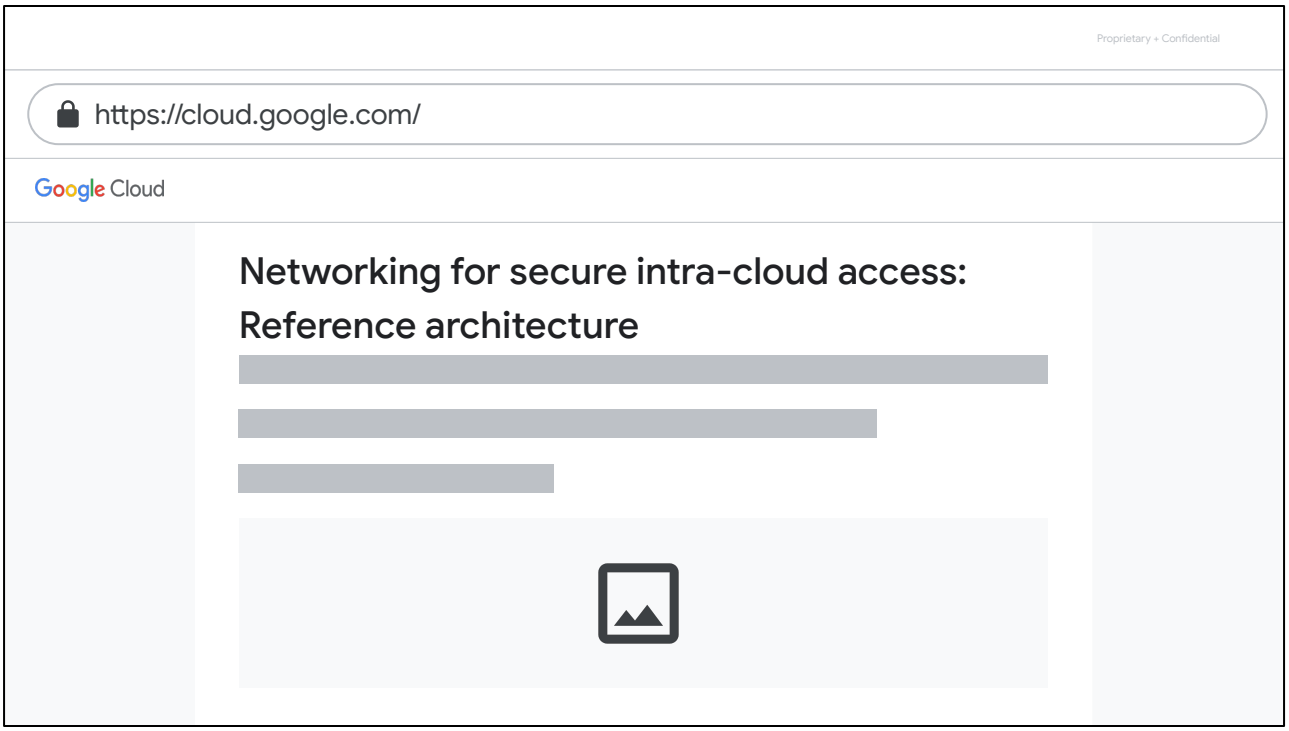
Network Virtual Appliance

Alternative to Private Service Connect for secure intra-cloud access in the Network Reference Architecture

Use when you need to route traffic through a centralized VM appliance



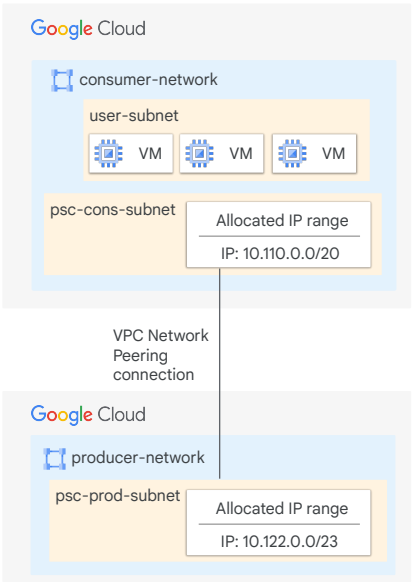
https://cloud.google.com/blog/topics/developers-practitioners/two-networking-patterns-secure-intra-cloud-access



For more information about the Network reference Architecture, visit the ‘Networking for secure intra-cloud access: Reference Architecture’ documentation.

Private services access

- Uses internal IPv4 addresses.
- Uses VPC Network Peering to connect consumer and producer VPC networks.
- Automates much of the VPC Network Peering configuration.
- Doesn't require explicitly importing and exporting routes.
- Is only available for some producer services, like Apigee, Cloud SQL, and Cloud TPU.



Like with Private Service Connect, private services access lets consumers use internal IPv4 addresses to consume producer services.

The connection between consumer and producer uses VPC Network Peering. Private services access automates much of the VPC Network Peering configuration.

With Private Service Connect, you had to import and export routes between the consumer and producer VPC networks. Because the connection between the consumer and the producer is made using VPC Network Peering, you don't need to import and export routes. Subnet routes that don't use privately used public IP addresses are always exchanged between peered VPC networks.

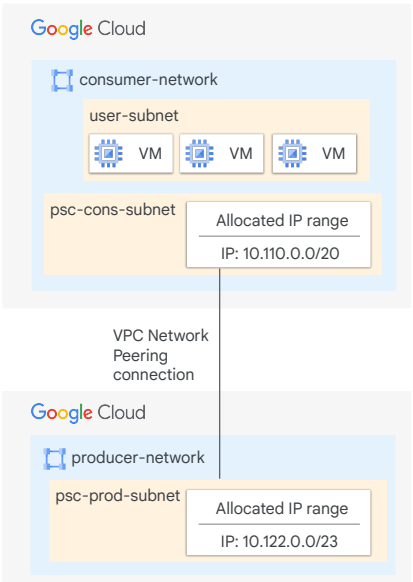
Private services access is available only for supported producer services, like Apigee, Cloud SQL, and Cloud TPU. For a complete list of supported producer services, see Private services access Supported services in the Google Cloud documentation.

To offer private connectivity, the service producer must complete a one-time onboarding process. To complete the onboarding process, contact your Google representative. For more information, see Onboarding process on the Enabling private services access page of the Google Cloud documentation.

After the onboarding process is complete, you can configure private services access.

Configuring private services access

- The service producer and consumer must activate the Service Networking API in their projects.
- Service producers must allocate an IPv4 address range in the VPC network that contains the service.
- Consumers must:
  - Allocate an IPv4 address range in their VPC network.
  - Create a private connection to a service producer.



To use private services access, both service consumers and producers must activate the Service Networking API in their projects.

The consumer and producer VPC networks require some configuration as well.

Service producers must allocate an IPv4 address range in the VPC network that contains the service. This address range is used for each connection from a service consumer.

Service consumers must also allocate an IPv4 address range in their VPC network for each service producer. For example, to use services from three different producer VPC networks, the consumer must allocate three IPv4 address ranges, one for each producer VPC network.

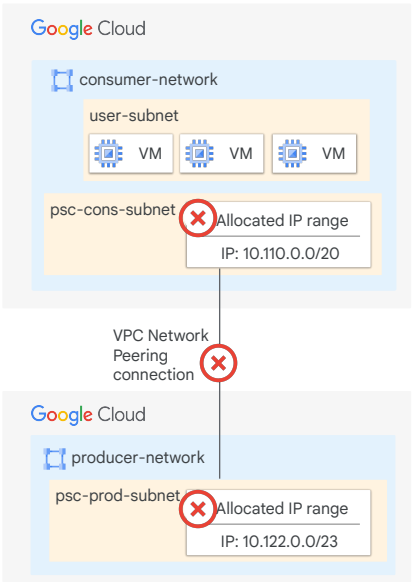
After the service producer has completed the initial configuration, consumers can create a private services access connection to the producer VPC network. You can use the Google Cloud console or the Google Cloud CLI to create this connection. Consumers and producers can use the Google Cloud console to edit their VPC network to configure private services access.

If a service producer offers multiple services, you only need one private connection. For example, if a consumer uses Cloud SQL and Cloud TPU, only one private connection is created.

Google Cloud uses VPC Network Peering to implement the connection between the consumer and producer VPC networks.

Deleting the connection

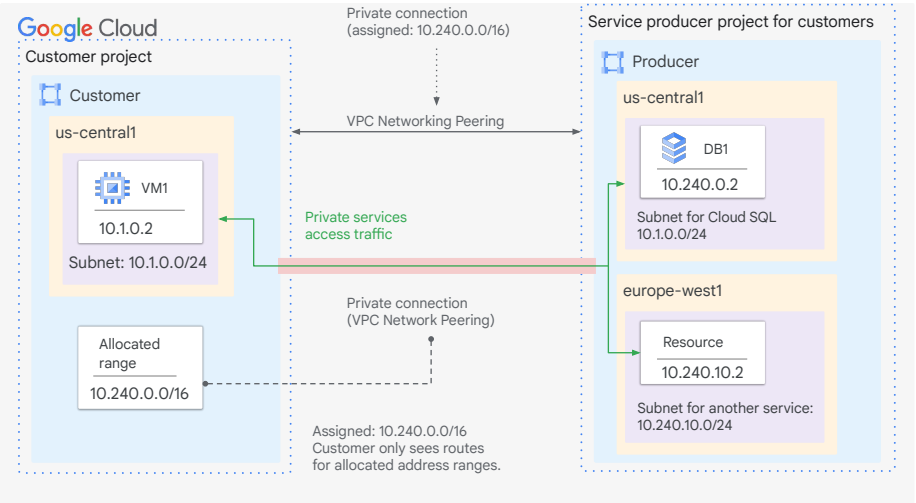
- Consumers can disable the private services access connection between their VPC network and the producer VPC network.
- Disabling the connection does not:
  - Delete the VPC Network Peering connection.
  - Release the IPv4 address range.



Consumers can disable the private services access connection between their VPC network and the producer VPC network. Consumers can also edit their VPC network settings to disable access. Disabling the private services access connection does not delete the VPC Network Peering to the producer VPC network. You can delete the VPC Network Peering connection by editing the VPC network.

Likewise, disabling the private services access connection does not release the IPv4 address range. Consumers must edit the VPC network to release the IPv4 address range.

Sample topology – private services access



This example shows a sample private services access topology. The customer VPC network allocated the 10.240.0.0/16 address range for Google services and established a private connection that uses the allocated range. Google then creates a project for the customer. With that project, each Google service creates a subnet from the allocated block to provision new resources in a given region. In the example, you can see a Cloud SQL instance. The Cloud SQL instance is assigned an IP of 10.240.0.2, which is within the 10.240.0.0/16 range.

In the customer VPC network, requests with a destination of 10.240.0.2 are routed over the private connection to the producer VPC network. The request is then sent to the correct resource in the producer VPC network.

If the service supports cross-region communication, VM instances in the customer network can access service resources in any region. Some services might not support cross-region communication. For more information, see the documentation of the relevant service.

Caveats: Private services access

01 If you connect on-premises networks, you must export the routes to the VPC producer network.

02 Not all Google services are supported.

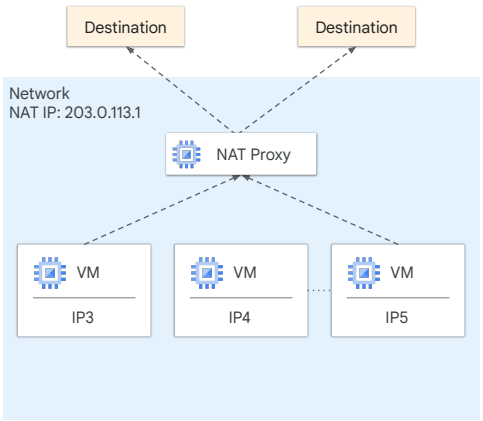
03 The same quota and limits that apply to VPC Network Peering also apply to private services access.

Private services access has a few caveats. For private services access to an on-premises network to work, you must export custom routes from the on-premises network to the producer VPC network.

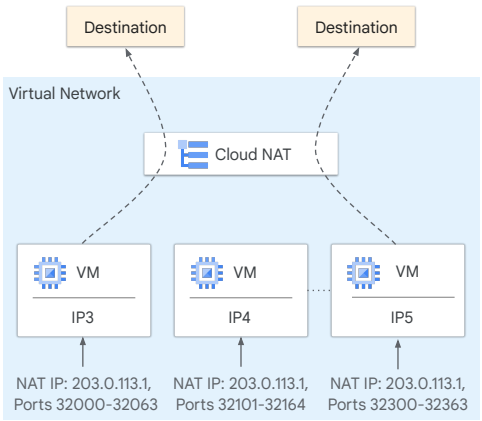
Not all Google services are supported. For a complete list of supported Google services, see Private services access supported services in the Google Cloud documentation.

The same quota and limits that apply to VPC Network Peering also apply to private services access. Private services access uses VPC Network Peering to implement connections and thus has the same restrictions as VPC Network Peering.

Cloud NAT is a fully managed, software-defined service



1. Typical NAT proxies



2. Cloud NAT

Cloud NAT is the Google-managed network address translation service. It lets you provision your application instances without public IP addresses, and it also lets them access the internet in a controlled and efficient manner. With Cloud NAT, your private instances can access the internet for updates, patching, configuration management, and more.

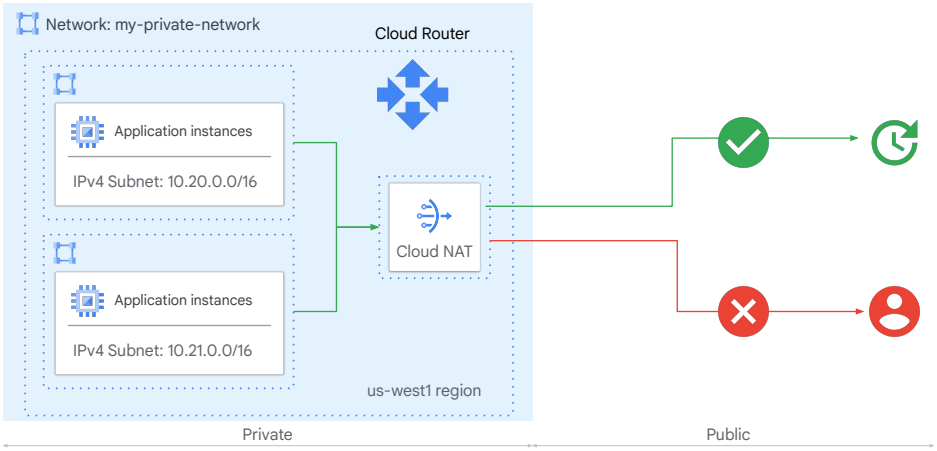
Cloud NAT as shown on the right offers several advantages when compared to other NAT offerings as shown on the left.

As a fully managed, software-defined service, Cloud NAT differs from traditional NAT proxy solutions. There are no NAT middle proxies in the path from the instance to the destination. Instead, each instance is allocated a NAT IP address along with a slice of the associated port range. This allocated IP address and port range are used by the instance to perform NAT. This design is free of chokepoints and is highly reliable, performant, and scalable.

Cloud NAT lets you configure multiple NAT IP addresses per NAT gateway. You can scale based on the size of your network without having to add or manage another NAT gateway. NAT IP allocation has two modes: manual and auto. The manual mode provides full control when specifying IP addresses. If you want to allow NAT addresses on the receiving side, use the manual mode. The auto mode enables the NAT IP addresses to be allocated and scaled automatically based on the number of instances.

For a full overview of Cloud NAT features, see Cloud NAT overview in the Google Cloud documentation.

Cloud NAT provides internet access to private instances



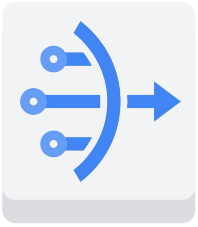
Note: IPv6 is supported

In this diagram, Cloud NAT enables two private instances to access an update server on the internet, which is referred to as outbound NAT. However, Cloud NAT does not implement inbound NAT.

In other words, hosts outside your VPC network can't directly access any of the private instances behind the Cloud NAT gateway. Your VPC networks remain isolated and secure.

Benefits of Cloud NAT

- Reduces the need for individual VMs to each have external IP addresses.
- Automatically scales the number of NAT IP addresses that it uses.
- Is not dependent on a single physical gateway device.

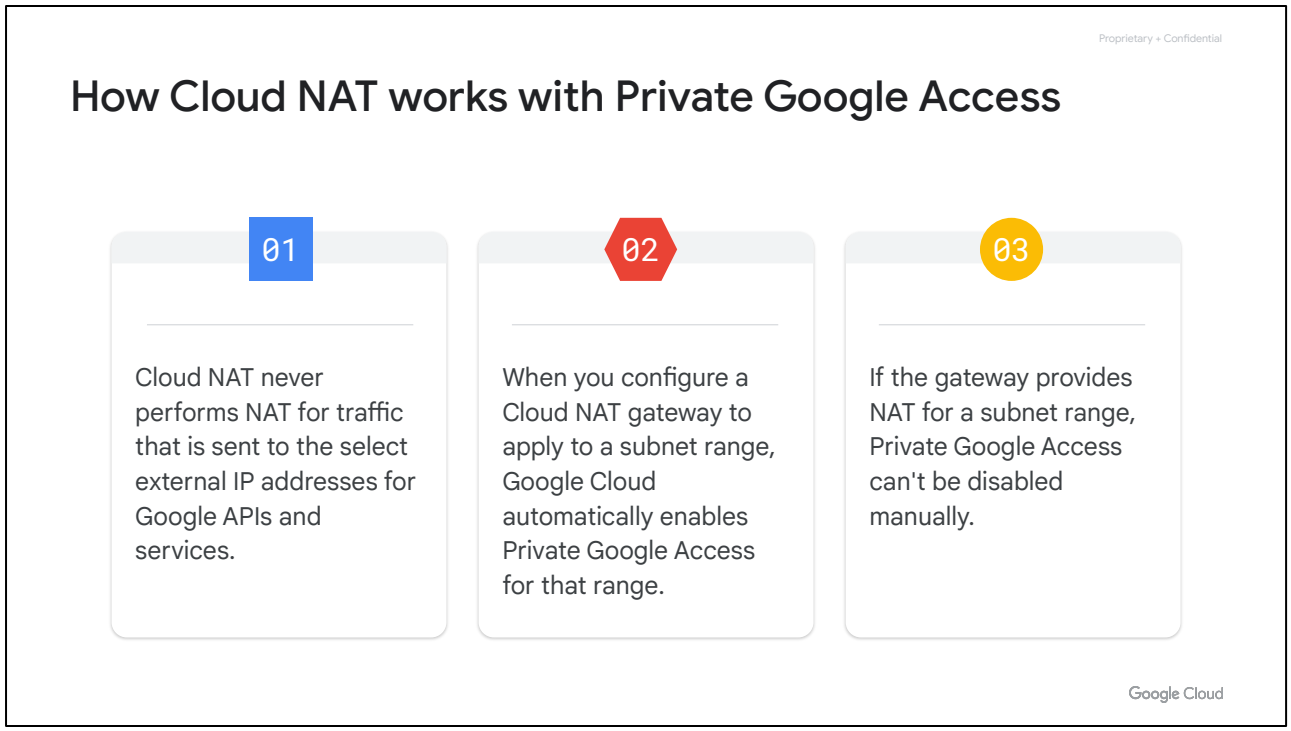


Cloud Spanner

With Cloud NAT, VMs without external IP addresses can access destinations on the internet. For example, you might have VMs that only need internet access to download updates or complete provisioning. Cloud NAT allows you to configure these VMs with an internal IP address. Thus, your organization needs fewer external IP addresses.

Cloud NAT can be configured to automatically scale the number of NAT IP addresses that it uses. Cloud NAT supports VMs that belong to managed instance groups, including those with auto scaling enabled.

Cloud NAT is not dependent on a single, physical gateway device. Cloud NAT is a distributed, software-defined managed service. You configure a NAT gateway on a Cloud Router, which provides the control plane for NAT. Cloud Router contains the NAT configuration parameters. Google Cloud runs and maintains processes on the physical machines that run your Google Cloud VMs.



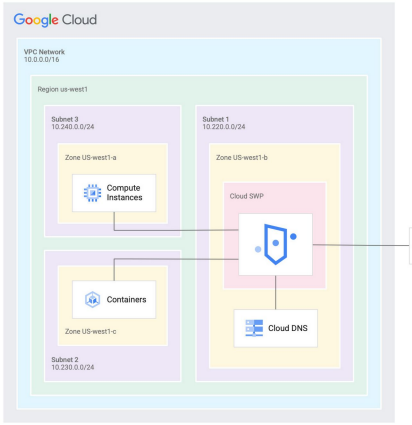
With Private Google Access, Cloud NAT never performs NAT (network address translation) for traffic sent to the select external IP addresses of Google APIs and services. Google Cloud routes this traffic internally.

When you configure a Cloud NAT gateway to apply to a subnet range, Google Cloud automatically enables Private Google Access for that range. Thus, any VMs in that subnet range use Private Google Access to connect to Google APIs and services.

If the gateway provides NAT for a subnet range, Private Google Access is in effect for that range and can't be disabled manually.

Cloud Secure Web Proxy (SWP)

- web (HTTP/S) egress traffic inspection, protection, and control
- Managed service (no VMs to set up)
- Explicit proxy (configure clients to use)
- Integrated with Cloud Monitoring/Logging
- Requests can come from:
  - VMs
  - Containers
  - Serverless environments
  - Hybrid workloads connected via VPN/Interconnect



https://cloud.google.com/secure-web-proxy/docs/overview

Compute Engine VMs

Optimising Network Performance

Google Virtual NIC	Machine type	Compact placement policies	TCP Network Performance
Use the Google Virtual NIC – it offers optimal network performance for all machine types	Choose your machine type carefully – network bandwidth can vary significantly between machine types	Reduce latency through compact placement policies – keep communicating VMs physically close to each other	Optimize TCP Network Performance – with TCP Window Scaling

https://cloud.google.com/compute/docs/networking/using-gvnic

https://cloud.google.com/compute/docs/network-bandwidth#summary-table

https://cloud.google.com/compute/docs/instances/use-compact-placement-policies

https://cloud.google.com/compute/docs/networking/tcp-optimization-for-network-performance-in-gcp-and-hybrid

Congratulations. You have completed the the PCNE Network Connectivity module.