Google Cloud

Professional Cloud Network Engineer

Partner Certification Academy

pls-academy-pcne-student-slides-1-2409

COURSE TITLE

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.

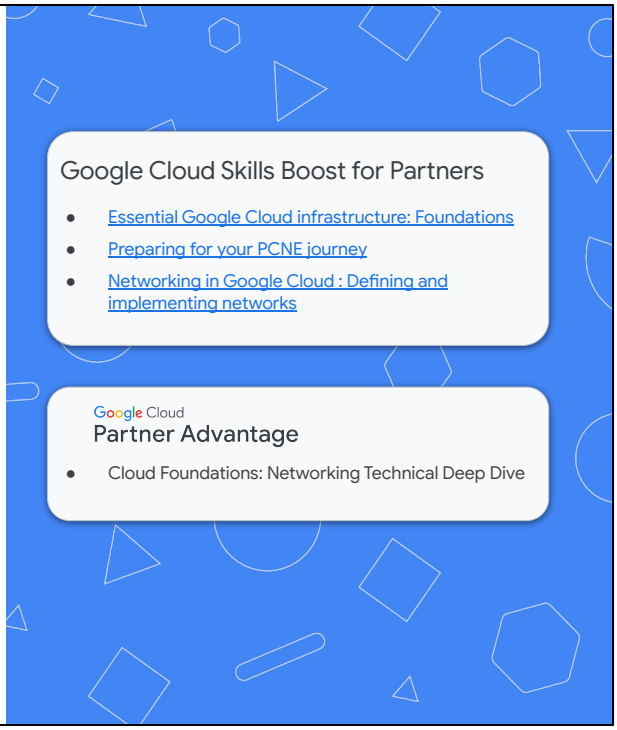
Thank you!

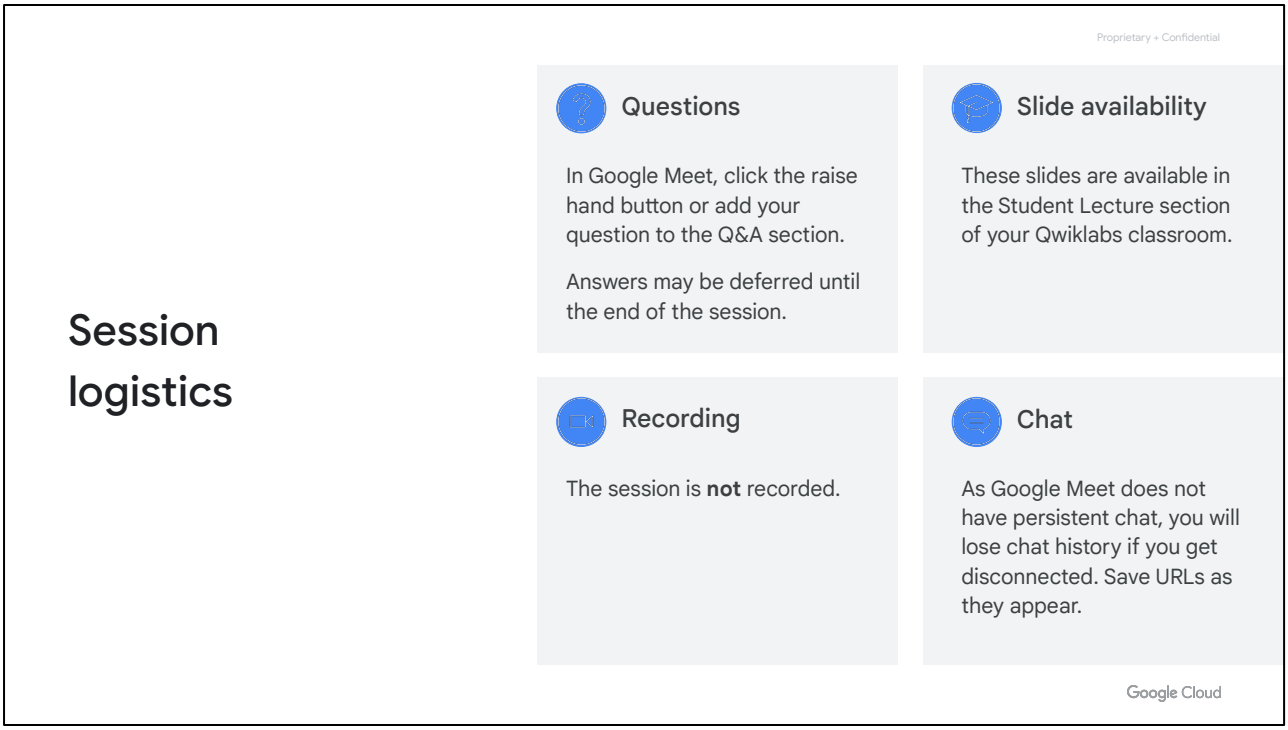
Google Cloud

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





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.

COURSE TITLE

Today's agenda

01 Partner readiness resources

02 Review of Google Cloud network fundamentals

03 Assessing your current level of Google Cloud networking knowledge

04 More aspects of Virtual Private Cloud (VPCs)

AGENDA

Objectives

01 Introduce Partner readiness resources

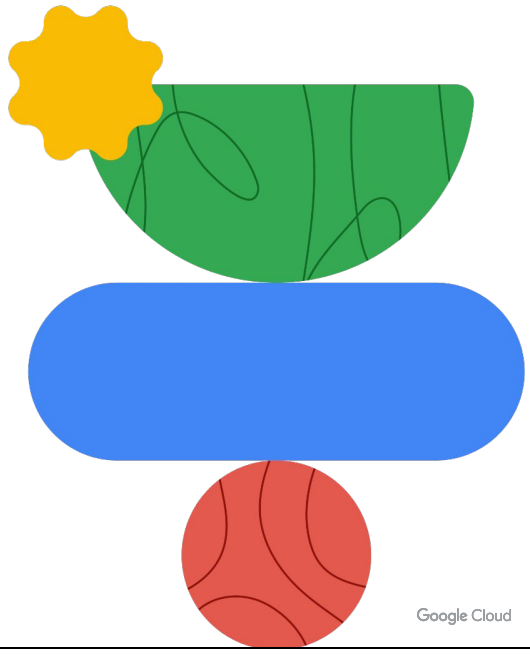
02 Review Google Cloud Network Fundamentals

03 Benchmark your existing Google Cloud network knowledge via diagnostic questions

04 Explain more advanced aspects of Virtual Private Clouds (VPCs) e.g. routes, IPv6 support, Domain Name System (DNS)

Objectives

Partner Readiness Resources

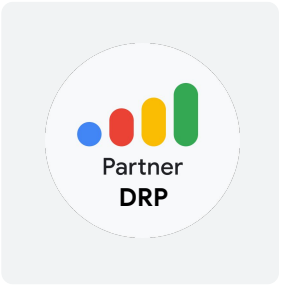


BREAK SLIDE

Path to service excellence

Certification

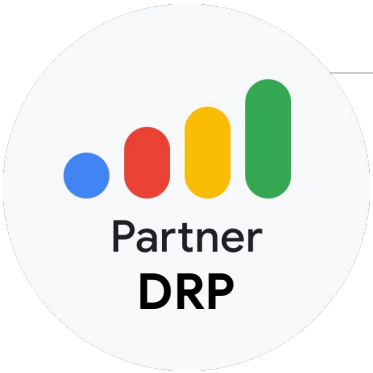
Advanced Solutions Training



Delivery Readiness Portal

Certification is just one step on your professional journey. Google Cloud also offers our partners access to advanced solutions training, and a new quality-focused program called Delivery Readiness Portal (DRP) to help you achieve service excellence with your customers.

Benchmark your skills with DRP



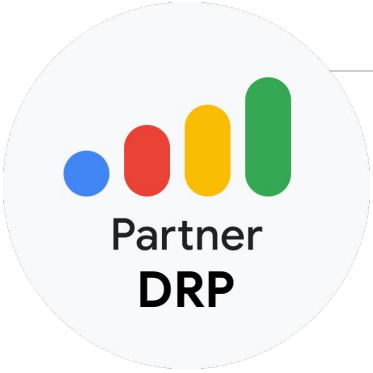
Assess

Partner proficiency and delivery capability
Benchmark partner individuals, and project teams and practices Google Cloud capabilities.

The Delivery Readiness Portal, or DRP helps to benchmark partner proficiency and capability at any point during the customer journey; however, it should be used primarily as a lead measure to predict and prepare for partner delivery success.

DRP assesses and analyzes Partner Consultant Google Cloud proficiency by creating a DRP Profile inclusive of their Google Cloud knowledge, skills, and experience.

Benchmark your skills with DRP



Analyze

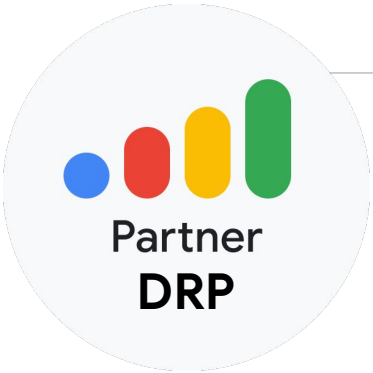
Individual partner consultants' Google Cloud readiness

Showcase partner individuals Google Cloud knowledge, skills, and experience.

With the DRP insights, you can prescriptively advise the partner project team on the ground and bridge niche capability gaps.

DRP also takes action. For partner consultants, DRP generates a tailored L&D plan that prescribes personalized learning, training, and skill development to build Google Cloud proficiency.

Benchmark your skills with DRP



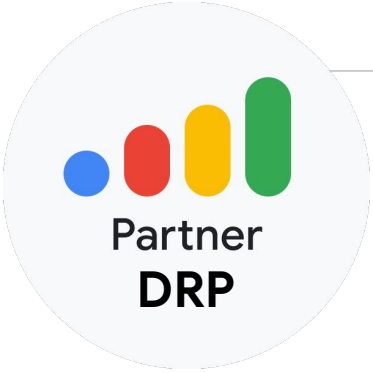
Advise

Google assurance for partner delivery

Packaged offerings to bridge specific capability gaps.

With the DRP insights, you can prescriptively advise the partner project team on the ground and bridge niche capability gaps.

Benchmark your skills with DRP



Action

Tailored L&D plan for account based enablement

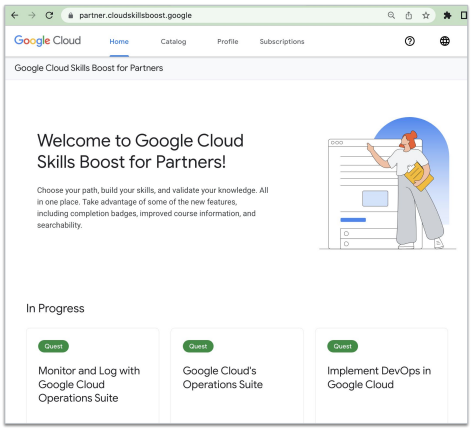
Personalized learning & development recommendations per individual consultant.

DRP also takes action. For partner consultants, DRP generates a tailored L&D plan that prescribes personalized learning, training, and skill development to build Google Cloud proficiency.

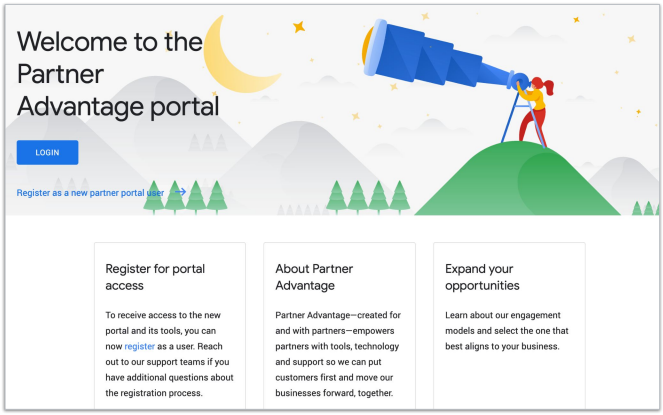
Google Cloud Skills Boost for Partners

https://partner.cloudskillsboost.google/

- On-demand course content
- Hands-on labs
- Skill Badges
- FREE to Google Cloud Partners!



Google Cloud Partner Advantage



Create login

Create a login using your company email. Your organization must verify your request prior to granting you access.

https://www.partneradvantage.goog

The getting started link:
https://support.google.com/googlecloud/topic/9198654#zippy=

Note the top section, **"Getting Started & User Guides"** and two key documents → Direct Partners to this if they need to enroll into Partner Advantage

1. Logging in to the Partner Advantage Portal - Quick Reference Guide
2. Enrolling in the Partner Advantage Program - Quick Reference Guide

Focus from this point on:

Some context on enrolling in PA:
Access to Partner Portal is given in 2 ways

- Partner Admin Led: Partner Administrator at Partner Company can set up users
- User Led: User can go through Self Registration
  - https://www.partneradvantage.goog/GCPPRM/s/partneradvantageportallogin?language=en_US
  - Or directly to the User Registration Form, https://www.partneradvantage.goog/GCPPRM/s/partnerselfregistration?language=en_US

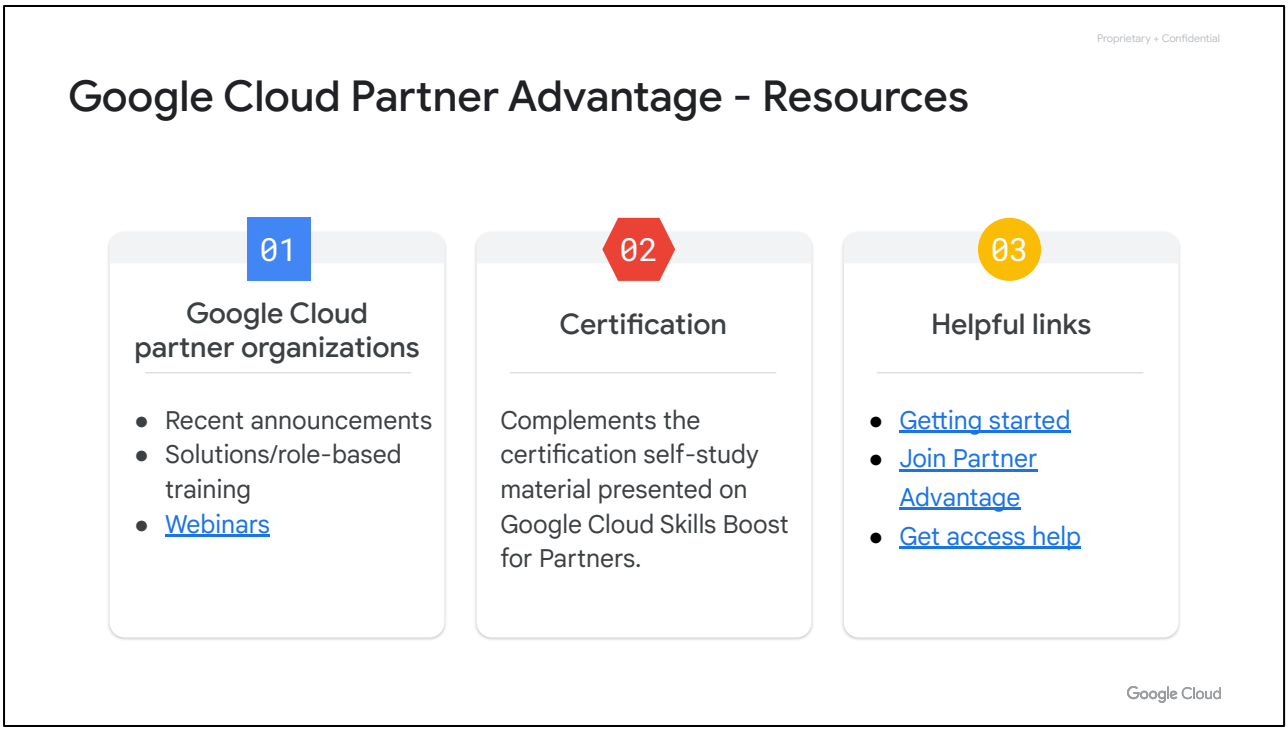
Please Note

- After a user self-registers, they receive an email that essentially states:

    - “Hi {Partner Name}, you are one step away from joining the Google Cloud Partner Advantage Community. Please click to continue with the user registration process. See you in the cloud, The Partner Advantage Team
- Once registered, they can access limited content until their **Partner Administrator approves the user**
- Their Partner Administrator also receive an email notifying them that a member of their organization has registered themselves on their organization’s Google Cloud Partner Advantage account.
    - It also states that this user has limited access to the portal
    - They are provided instructions on how to review and provision the appropriate access for the user that has registered
- Once their admin approves the user, they receive an email that states:
    - Hi {User Name}, Your Partner Administrator has updated your access to the Google Cloud Partner Advantage portal. You have been granted edit access to additional account information on the portal on behalf of your organization to help build your business. For additional access needs, please work with your Partner Administrator. See you in the cloud, The Partner Advantage Team

The net takeaway is, on the Support Page (the first link on this slide) Google Cloud Partner Advantage Support, there’s a section **“Issue accessing Partner Advantage Portal? Click here for troubleshooting steps”**

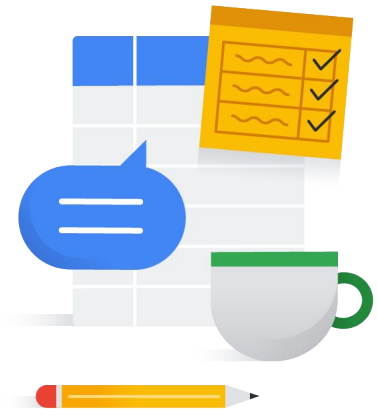
- The source of their issue can be related to the different items shown
- Additionally, there’s a Partner Administrator / Partner Adminstrator Team at their partner organization that has to approve their access.. Until that step is completed, they will have access issues/limitation. They will need to identify who this person or team is at their organization



Partner Advantage has a range of technical and marketing resources to help you develop, grow and execute your Google Cloud-related business.

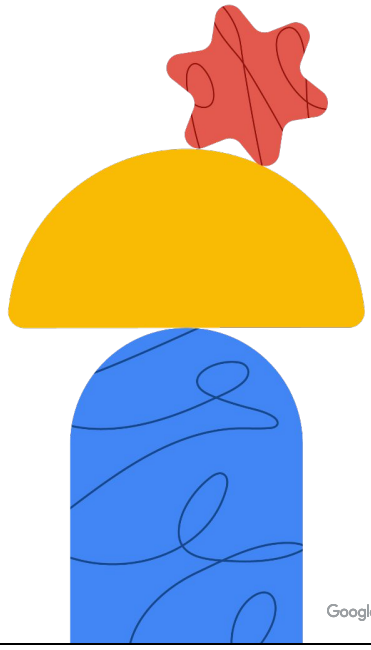
Program issues or concerns

- Problems with **accessing** Cloud Skills Boost for Partners
  - cloud-partner-training@google.com
- Problems with **a lab** (locked out, etc.)
  - support@qwiklabs.com
- Problems with accessing Partner Advantage
  - https://support.google.com/googlecloud/topic/9198654



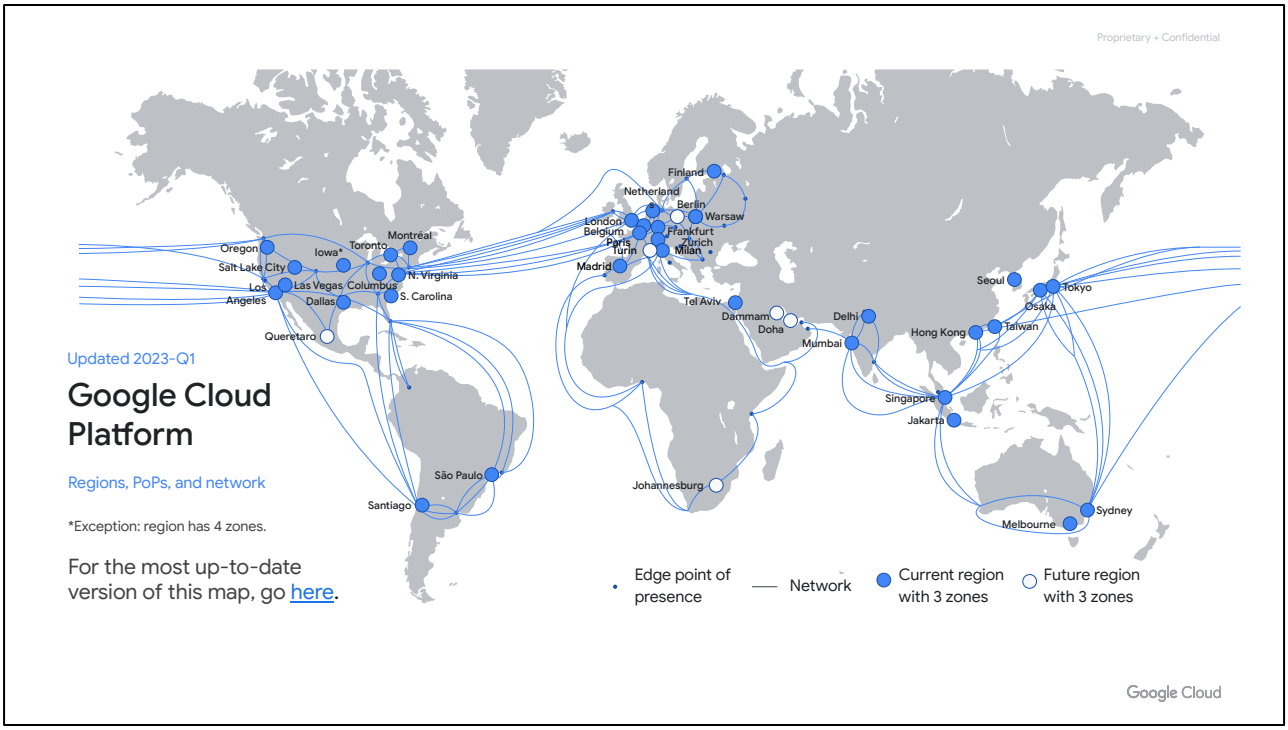
- For problems with accessing **Cloud Skills Boost for Partners**
  - partner-training@google.com
- For problems with a **lab** (locked out, etc.)
  - support@qwiklab.com
- For problems with accessing **Partner Advantage**
  - https://support.google.com/googlecloud/topic/9198654

Review of Google Cloud network fundamentals



In this section ~~module~~, you will be covering 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.



This map represents Google Cloud. On a high level, Google Cloud consists of regions, which are the icons in blue; points of presence or PoPs, which are the dots in blue; a global private network, which is represented by the blue lines; and services.

A region is a specific geographical location where you can run your resources. This map shows several regions that are currently operating, as well as future regions. Regions indicated with blue icons have three zones. Iowa is an exception, where the region called us-central1 has four zones: us-central1-a, us-central1-b, us-central1-c, and us-central1-f. For up-to-date information on regions and zones, please refer to the documentation page.

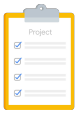
The PoPs are where Google's network is connected to the rest of the internet. Google Cloud can bring its traffic closer to its peers because it operates an extensive global network of interconnection points. This reduces costs and provides users with a better experience.

The network connects regions and PoPs and is composed of a global network of fiber optic cables with several submarine cable investments.

For more information about Google's networking infrastructure, refer to this site.

Projects and networks

A project:



- Associates objects and services with billing.
- Contains networks (up to 5) that can be shared/peered.

A network:

- Has no IP address range.
- Is global and spans all available regions.
- Contains subnetworks.
- Is available as default, auto, or custom.

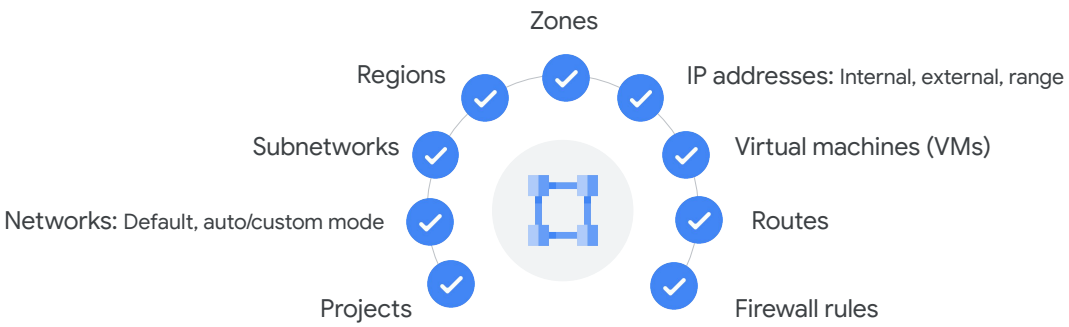
Projects are the key organizer of infrastructure resources in Google Cloud. A project associates objects and services with billing. Now, it's unique that projects actually contain entire networks. The default quota for each project is 5 networks, but you can simply request additional quota using the Google Cloud console. These networks can be shared with other projects, or they can be peered with networks in other projects, both of which we will cover later in the Architecting with Google Compute Engine course.

These networks do not have IP ranges but are simply a construct of all of the individual IP addresses and services within that network. Google Cloud's networks are global, spanning all available regions across the world that we showed earlier. So, you can have one network that literally exists anywhere in the world—Asia, Europe, Americas—all simultaneously.
Inside a network, you can segregate your resources with regional subnetworks.

We just mentioned that there are different types of networks: default, auto, and custom. Let's explore these types of networks in more detail.

[More information on setting up a VPC:
https://www.youtube.com/watch?v=cNb7xKyya5c]

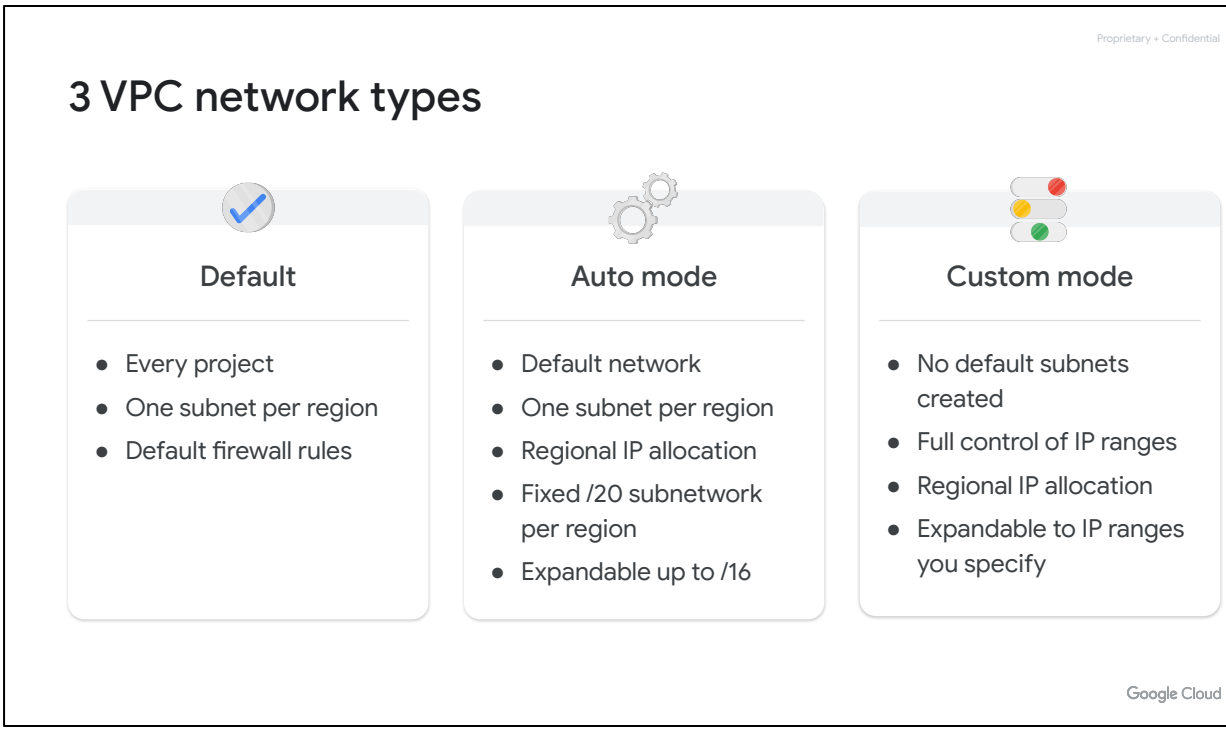
Virtual private cloud (VPC) objects



With Google Cloud, you can provision your Google Cloud resources, connect them to each other, and isolate them from each other in a Virtual Private Cloud. You can also define fine-grained networking policies within Google Cloud, and between Google Cloud and on-premises or other public clouds. Essentially, VPC is a comprehensive set of Google-managed networking objects, which we will explore in detail throughout this module.

Let me give you a high-level overview of these objects:

- Projects are going to encompass every single service that you use, including networks.
- Networks come in three different flavors: Default, auto mode, and custom mode.
- Subnetworks allow you to divide or segregate your environment.
- Regions and zones represent Google's data centers, and they provide continuous data protection and high availability.
- VPC provides IP addresses for internal and external use, along with granular IP address range selections.
  As for virtual machines, in this module we will focus on configuring VM instances from a networking perspective.
- We'll also go over routes and firewall rules.



Every project is provided with a default VPC network with preset subnets and firewall rules. Specifically, a subnet is allocated for each region with non-overlapping CIDR blocks and firewall rules that allow ingress traffic for ICMP, RDP, and SSH traffic from anywhere, as well as ingress traffic from within the default network for all protocols and ports.

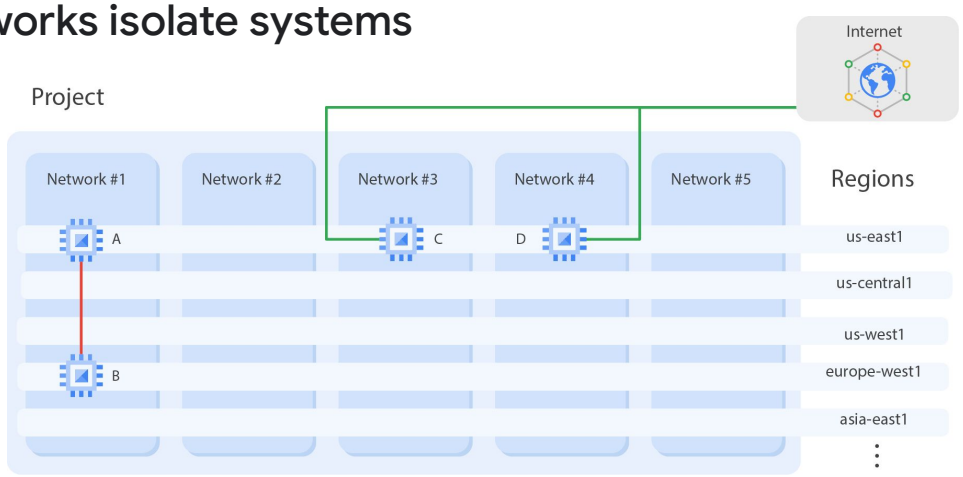
In an auto mode network, one subnet from each region is automatically created within it. The default network is actually an auto mode network. These automatically created subnets use a set of predefined IP ranges with a /20 mask that can be expanded to /16. All of these subnets fit within the 10.128.0.0/9 CIDR block. Therefore, as new Google Cloud regions become available, new subnets in those regions are automatically added to auto mode networks using an IP range from that block.

A custom mode network does not automatically create subnets. This type of network provides you with complete control over its subnets and IP ranges. You decide which subnets to create, in regions you choose, and using IP ranges you specify. These IP ranges cannot overlap between subnets of the same network.

Now, you can convert an auto mode network to a custom mode network to take advantage of the control that custom mode networks provide. However, this conversion is one way, meaning that custom mode networks cannot be changed to

auto mode networks. So, carefully review the considerations for auto mode networks to help you decide which type of network meets your needs.

Networks isolate systems



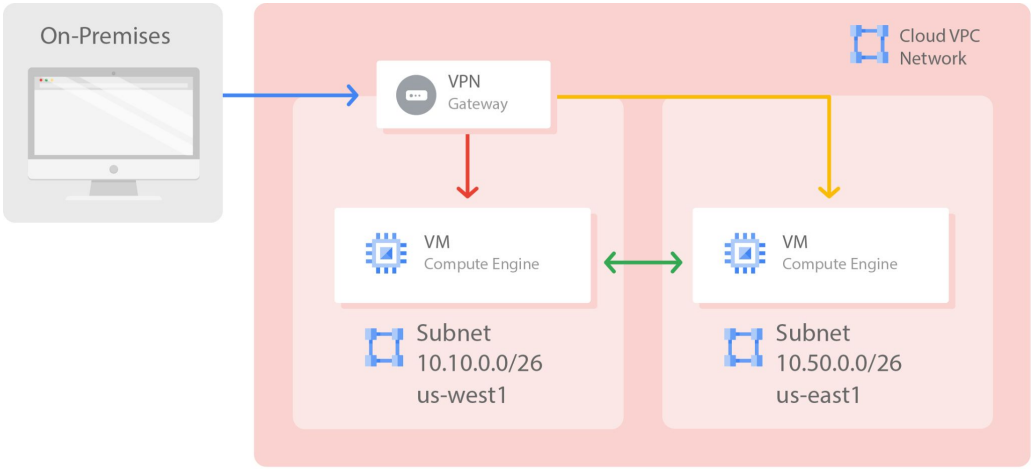
- **A** and **B** can communicate over internal IPs *even though they are in different regions.*
- **C** and **D** must communicate over external IPs *even though they are in the same region.*

On this slide, we have an example of a project that contains 5 networks. All of these networks span multiple regions across the world, as you can see on the right.

Each network contains separate virtual machines: A, B, C, and D. Because VMs A and B are in the same network, network 1, they can communicate using their internal IP addresses, even though they are in different regions. Essentially, your virtual machines, even if they exist in different locations across the world, take advantage of Google's global fiber network. Those virtual machines appear as though they're sitting in the same rack when it comes to a network configuration protocol.

VMs C and D, however, are not in the same network. Therefore, by default, these VMs must communicate using their external IP addresses, even though they are in the same region. The traffic between VMs C and D isn't actually touching the public internet, but is going through the Google Edge routers. This has different billing and security ramifications that we will explore later.

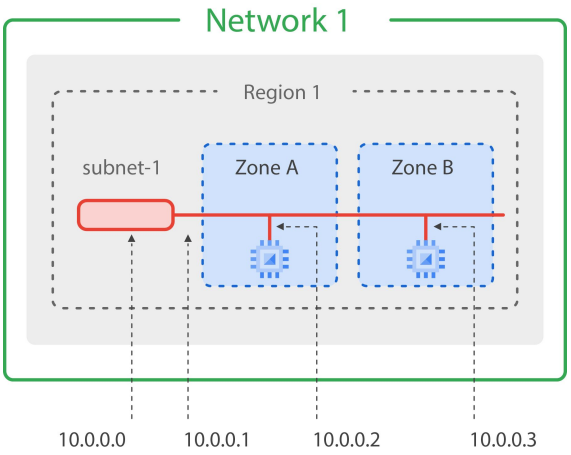
Google's VPC is global



Because VM instances within a VPC network can communicate privately on a global scale, a single VPN can securely connect your on-premises network to your Google Cloud network, as shown in this diagram. Even though the two VM instances are in separate regions (us-west1 and us-east1), they leverage Google's private network to communicate between each other and to an on-premises network through a VPN gateway.

This reduces cost and network management complexity.

Subnetworks cross zones



- VMs can be on the same subnet but in different zones.
- A single firewall rule can apply to both VMs.

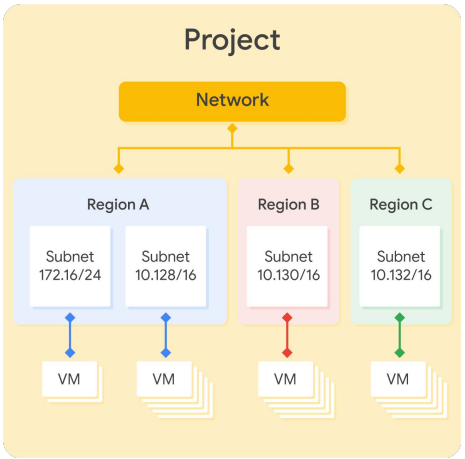
We mentioned that subnetworks work on a regional scale. Because a region contains several zones, subnetworks can cross zones.

This slide has a region, region 1, with two zones, zones A and B. Subnetworks can extend across these zones within the same region, such as, subnet-1. The subnet is simply an IP address range, and you can use IP addresses within that range. Notice that the first and second addresses in the range, .0 and .1, are reserved for the network and the subnet's gateway, respectively. This makes the first and second available addresses .2 and .3, which are assigned to the VM instances. The other reserved addresses in every subnet are the second-to-last address in the range and the last address, which is reserved as the "broadcast" address. To summarize, every subnet has four reserved IP addresses in its primary IP range.

Now, even though the two virtual machines in this example are in different zones, they still communicate with each other using the same subnet IP address. This means that a single firewall rule can be applied to both VMs, even though they are in different zones.

Expand subnets without re-creating instances

- Cannot overlap with other subnets
- IP range must be a unique valid CIDR block
- New subnet IP ranges have to fall within valid IP ranges
- Can expand but not shrink
- Auto mode can be expanded from /20 to /16
- Avoid large subnets

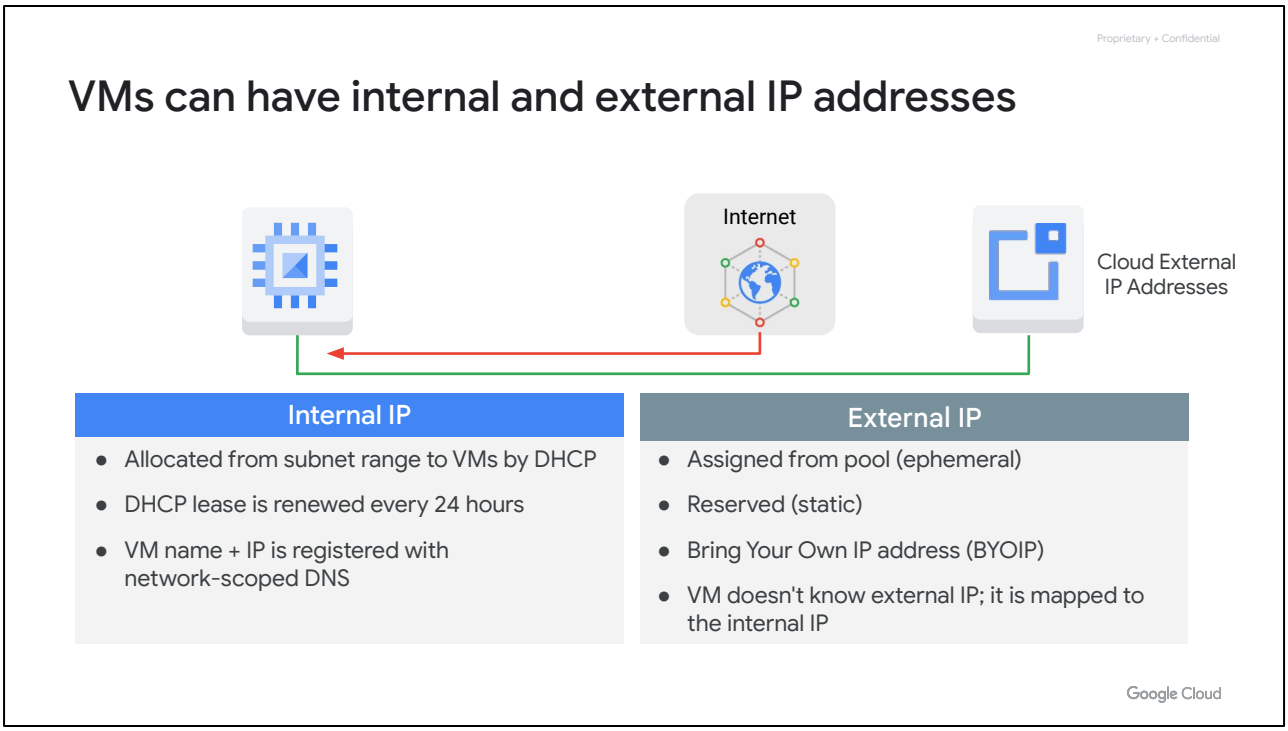


Speaking of IP addresses of a subnet, Google Cloud VPCs let you increase the IP address space of any subnets without any workload shutdown or downtime. This diagram illustrates a network with subnets that have different subnet masks, allowing for more instances in some subnets than others. This gives you flexibility and growth options to meet your needs, but there are some things to remember:

- The new subnet must not overlap with other subnets in the same VPC network in any region.
- Each IP range for all subnets in a VPC network must be a unique valid CIDR block.
- Also, the new subnet IP address ranges are regional internal IP addresses and have to fall within valid IP ranges.
    - Subnet ranges cannot match, be narrower, or be broader than a restricted range.
    - Subnet ranges cannot span a valid RFC range and a privately used public IP address range.
    - Subnet ranges cannot span multiple RFC ranges.
- The new network range must be larger than the original, which means the prefix length value must be a smaller number. In other words, you cannot undo an expansion.
- Now, auto mode subnets start with a /20 IP range. They can be expanded to a

- /16 IP range, but no larger. Alternatively, you can convert the auto mode subnetwork to a custom mode subnetwork to increase the IP range further.
- Also, avoid creating large subnets. Overly large subnets are more likely to cause CIDR range collisions when using Multiple Network Interfaces and VPC Network Peering, or when configuring a VPN or other connections to an on-premises network. Therefore, do not scale your subnet beyond what you actually need.

For a demo on how to expand a custom subnet in Google Cloud, refer to this video.



In Google Cloud, each virtual machine can have two IP addresses assigned. One of them is an internal IP address, which is going to be assigned via DHCP internally.

Every VM that starts up and any service that depends on virtual machines gets an internal IP address. Examples of such services are App Engine and Google Kubernetes Engine, which are explored in other courses.

When you create a VM in Google Cloud, its symbolic name is registered with an internal DNS service that translates the name to the internal IP address. DNS is scoped to the network, so it can translate web URLs and VM names of hosts in the same network, but it can't translate host names from VMs in a different network.

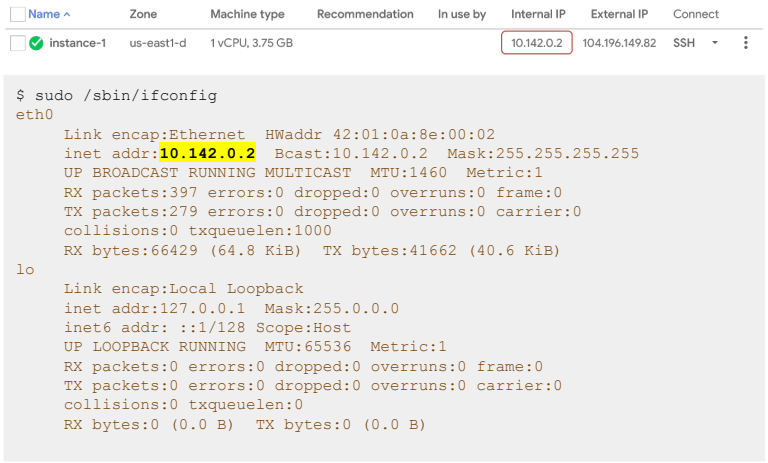
The other IP address is the external IP address but this one is optional. You can assign an external IP address, if your device or your machine is externally facing. That external IP address can be assigned from a pool, making it ephemeral, or it can be assigned a reserved external IP address, making it static. If you reserve a static external IP address and do not assign it to a resource such as a VM instance or a forwarding rule, you are charged at a higher rate than for static and ephemeral external IP addresses that are in use.

For more information about this, refer to the documentation page. You can use your

own publicly routable IP address prefixes as Google Cloud external IP addresses and advertise them on the internet. In order to be eligible, you must own and bring a /24 block or larger.

For a quick walk through of internal and external IP addresses in Google Cloud, refer to this demo.

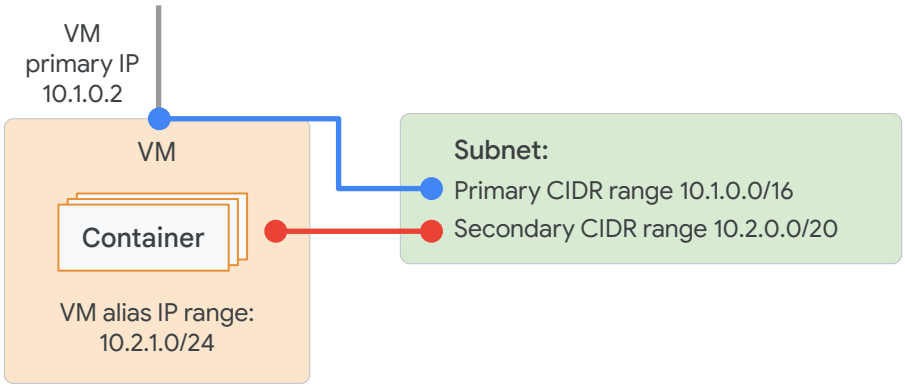
External IPs are mapped to internal IPs



Regardless of whether you use an ephemeral or static IP address, the external address is unknown to the OS of the VM. The external IP address is mapped to the VM's internal address transparently by VPC. We are illustrating this here by running ifconfig within a VM in Google Cloud, which only returns the internal IP address.

Let’s explore this further by looking at DNS resolution for both internal and external addresses.

Assign a range of IP addresses as aliases to a VM's network interface using alias IP ranges

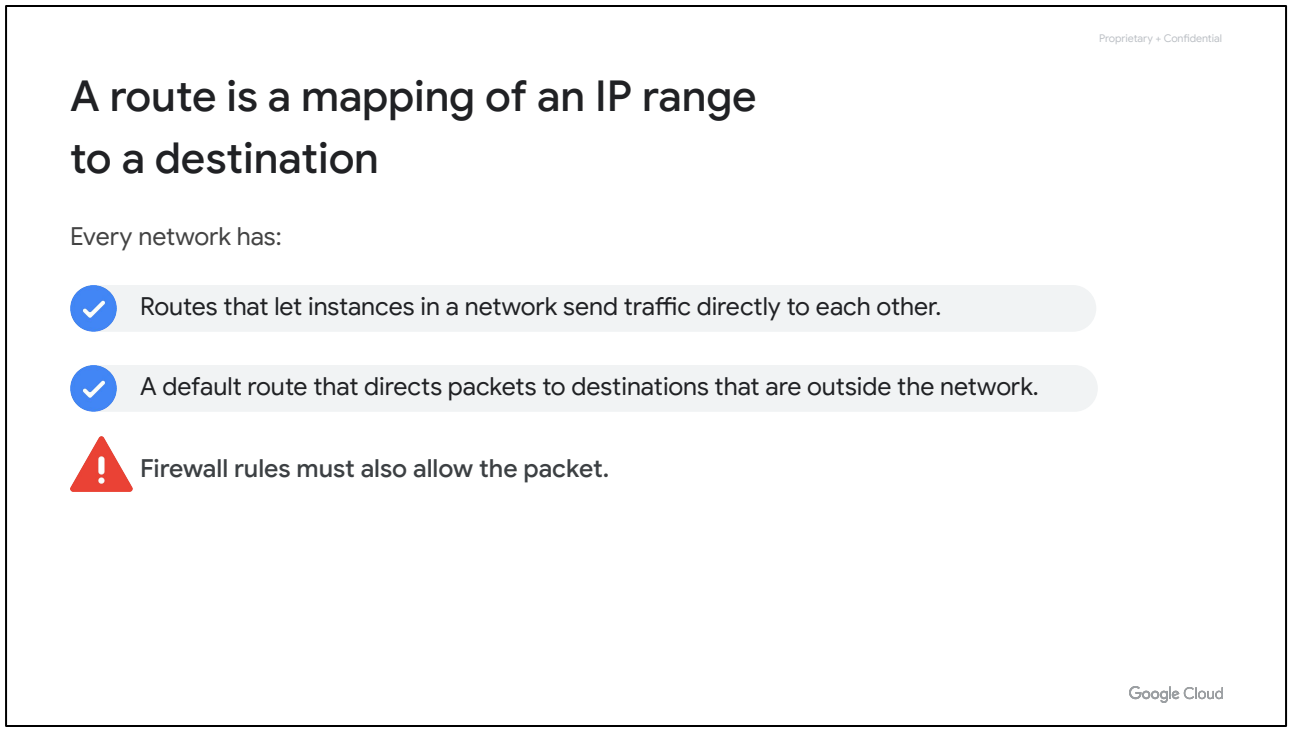


Another networking feature of Google Cloud is Alias IP Ranges.

Alias IP Ranges let you assign a range of internal IP addresses as an alias to a virtual machine's network interface. This is useful if you have multiple services running on a VM, and you want to assign a different IP address to each service.

In essence, you can configure multiple IP addresses, representing containers or applications hosted in a VM, without having to define a separate network interface. You just draw the alias IP range from the local subnet's primary or secondary CIDR ranges. This diagram provides a basic illustration of primary and secondary CIDR ranges and VM alias IP ranges.

For more information about Alias IP Ranges, refer to the documentation page.



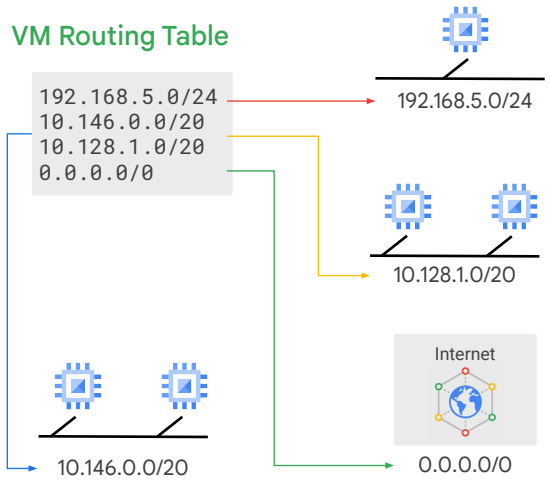
By default, every network has routes that let instances in a network send traffic directly to each other, even across subnets. In addition, every network has a default route that directs packets to destinations that are outside the network. Although these routes cover most of your normal routing needs, you can also create special routes that override these routes.

Just creating a route does not ensure that your packets will be received by the specified next hop. Firewall rules must also allow the packet.

The default network has pre-configured firewall rules that allow all instances in the network to talk with each other. Manually created networks do not have such rules, so you must create them, as you will experience in the first lab.

Routes map traffic to destination networks

- Apply to traffic egressing a VM.
- Forward traffic to most specific route.
- Are created when a subnet is created.
- Enable VMs on same network to communicate.
- Destination is in CIDR notation.
- Traffic is delivered only if it also matches a firewall rule.

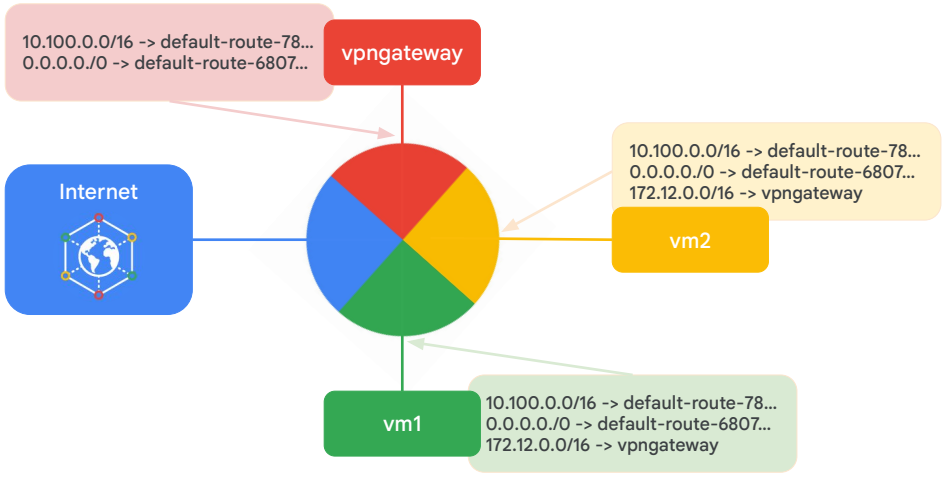


Routes match packets by destination IP address. However, no traffic will flow without also matching a firewall rule.

A route is created when a network is created, enabling traffic delivery from “anywhere.” Also, a route is created when a subnet is created. This is what enables VMs on the same network to communicate.

This slide shows a simplified routing table, but let’s look at this in more detail.

Instance routing tables



Each route in the Routes collection may apply to one or more instances. A route applies to an instance if the network and instance tags match. If the network matches and there are no instance tags specified, the route applies to all instances in that network. Compute Engine then uses the Routes collection to create individual read-only routing tables for each instance.

This diagram shows a massively scalable virtual router at the core of each network. Every virtual machine instance in the network is directly connected to this router, and all packets leaving a virtual machine instance are first handled at this layer before they are forwarded to their next hop. The virtual network router selects the next hop for a packet by consulting the routing table for that instance.

Firewall rules protect your VM instances from unapproved connections

- VPC network functions as a distributed firewall.
- Firewall rules are applied to the network as a whole.
- Connections are allowed or denied at the instance level.
- Firewall rules are stateful.
- Implied deny all ingress and allow all egress.

Cloud Firewall Rules

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. You can think of the firewall as existing not only between your instances and other networks, but between individual instances within the same network.

Google Cloud firewall rules are stateful. This means that if a connection is allowed between a source and a target or a target and a destination, all subsequent traffic in either direction will be allowed. In other words, firewall rules allow bidirectional communication once a session is established.

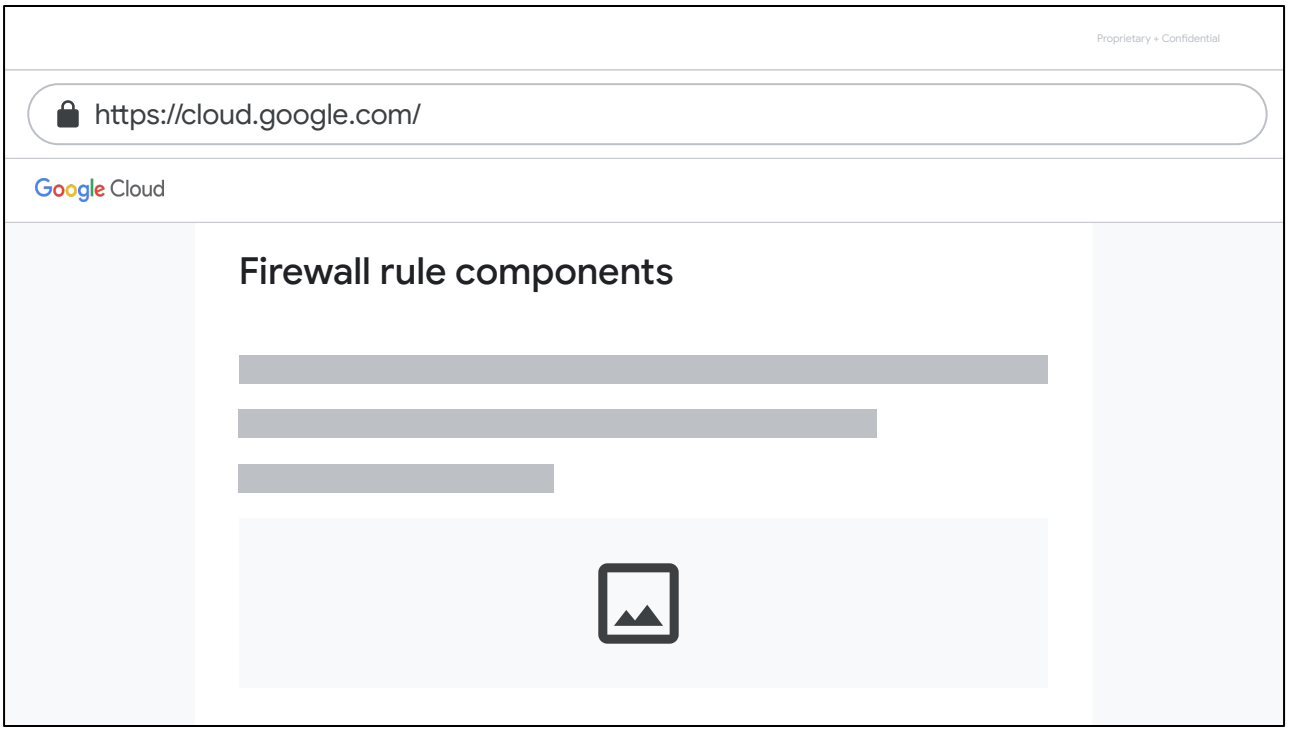
Also, if for some reason, all firewall rules in a network are deleted, there is still an implied "Deny all" ingress rule and an implied "Allow all" egress rule for the network.

A firewall rule is composed of...

Parameter	Details
`direction`	Inbound connections are matched against `ingress` rules only.
	Outbound connections are matched against `egress` rules only.
`source or destination`	For the `ingress` direction, `sources` can be specified as part of the rule with IP addresses, source tags or a source service account.
	For the `egress` direction, `destinations` can be specified as part of the rule with one or more ranges of IP addresses.
`protocol and port`	Any rule can be restricted to apply to specific protocols only or specific combinations of protocols and ports only.
`action`	To allow or deny packets that match the direction, protocol, port, and source or destination of the rule.
`priority`	Governs the order in which rules are evaluated; the first matching rule is applied.
`Rule assignment`	All rules are assigned to all instances, but you can assign certain rules to certain instances only.

You can express your desired firewall configuration as a set of firewall rules. Conceptually, a firewall rule is composed of the following parameters:

- The **direction** of the rule. Inbound connections are matched against ingress rules only, and outbound connections are matched against egress rules only.
- The **source** of the connection for ingress packets, or the **destination** of the connection for egress packets.
- The **protocol** and **port** of the connection, where any rule can be restricted to apply to specific protocols only or specific combinations of protocols and ports only.
- The **action** of the rule, which is to allow or deny packets that match the direction, protocol, port, and source or destination of the rule.
- The **priority** of the rule, which governs the order in which rules are evaluated. The first matching rule is applied.
- The **rule assignment**. By default, all rules are assigned to all instances, but you can assign certain rules to certain instances only.

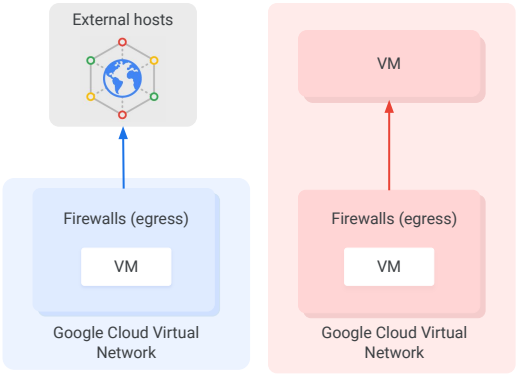


For more information on firewall rule components, refer to the official Google Cloud documentation.

Next, let's delve into some Google Cloud firewall use cases for both egress and ingress.

@trainer: please click on the words on screen to access the document. This is for all slides of this nature throughout the course.

Google Cloud firewall use case: Egress



Conditions:

- Destination CIDR ranges
- Protocols
- Ports

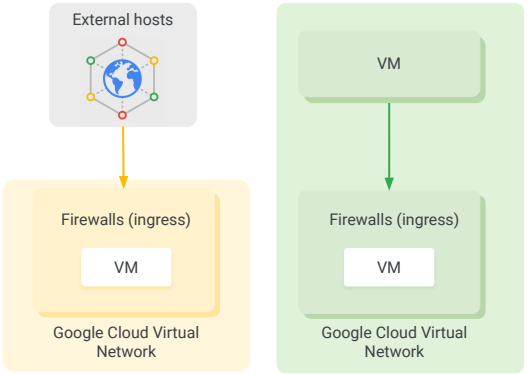
Action:

- Allow: permit the matching egress connection
- Deny: block the matching egress connection

Egress firewall rules control outgoing connections originated inside your Google Cloud network. Egress **allow** rules allow outbound connections that match specific protocol, ports, and IP addresses. Egress **deny** rules prevent instances from initiating connections that match non-permitted port, protocol, and IP range combinations.

For egress firewall rules, destinations to which a rule applies may be specified using IP CIDR ranges. Specifically, you can use destination ranges to protect from undesired connections initiated by a VM instance toward an external host, as shown on the left. You can also use destination ranges to prevent undesired connections from internal VM instances to a specific Google Cloud CIDR range. This is illustrated in the middle, where a VM in a specific subnet is shown attempting to connect inappropriately to another VM within the same network.

Google Cloud firewall use case: Ingress



Conditions:

- Source CIDR ranges
- Protocols
- Ports

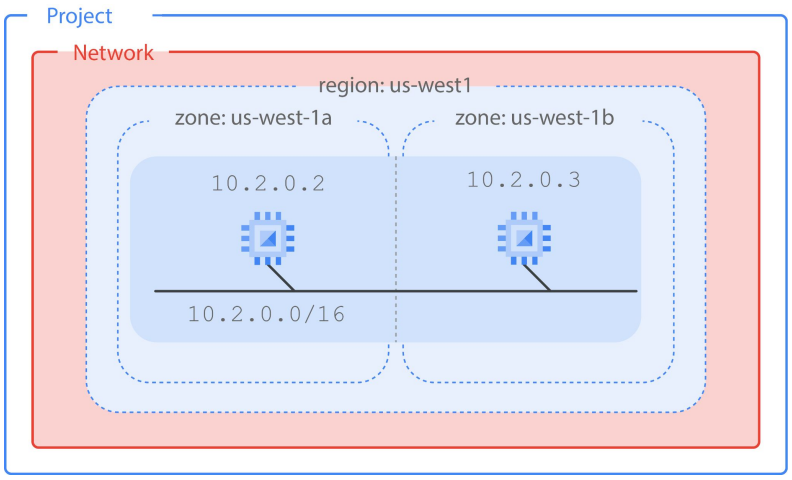
Action:

- Allow: permit the matching ingress connection
- Deny: block the matching ingress connection

Ingress firewall rules protect against incoming connections to the instance from any source. Ingress **allow** rules allow specific protocol, ports, and IP addresses to connect in. The firewall prevents instances from receiving connections on non-permitted ports or protocols. Rules can be restricted to only affect particular sources.
Source CIDR ranges can be used to protect an instance from undesired connections coming either from external networks or from Google Cloud IP ranges.

This diagram illustrates a VM receiving a connection from an external address, and another VM receiving a connection from a VM within the same network. You can control ingress connections from a VM instance by constructing inbound connection conditions using source CIDR ranges, protocols, or ports.

Increased availability with multiple zones

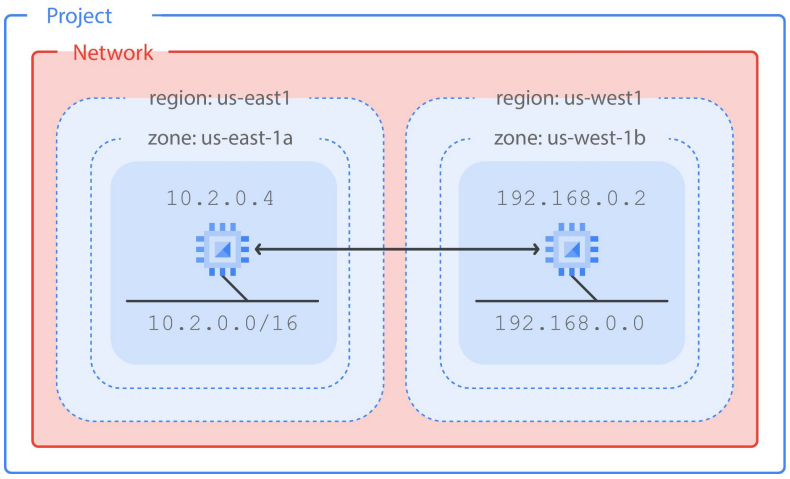


Let's start by looking at availability.

If your application needs increased availability, you can place two virtual machines into multiple zones but within the same subnetwork, as shown on this slide. Using a single subnetwork allows you to create a firewall rule against the subnetwork 10.2.0.0/16.

Therefore, by allocating VMs on a single subnet to separate zones, you get improved availability without additional security complexity. A regional managed instance group contains instances from multiple zones across the same region, which provides increased availability.

Globalization with multiple regions



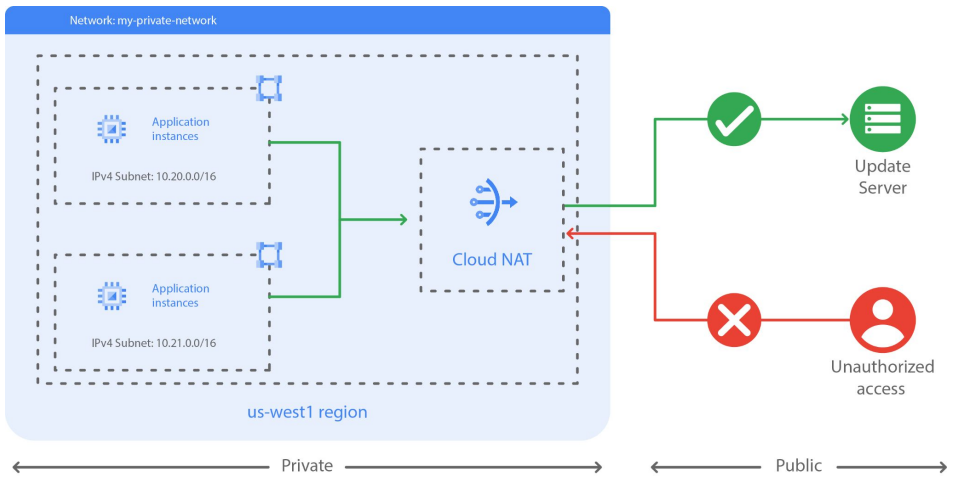
Next, let's look at globalization.

In the previous design we placed resources in different zones in a single region, which provides isolation from many types of infrastructure, hardware, and software failures. Putting resources in different regions as shown on this slide provides an even higher degree of failure independence. This allows you to design robust systems with resources spread across different failure domains.

When using a global load balancer, like the HTTP load balancer, you can route traffic to the region that is closest to the user. This can result in better latency for users and lower network traffic costs for your project.

We'll explore both managed instance groups and load balancer later in the course.

Cloud NAT provides internet access to private instances

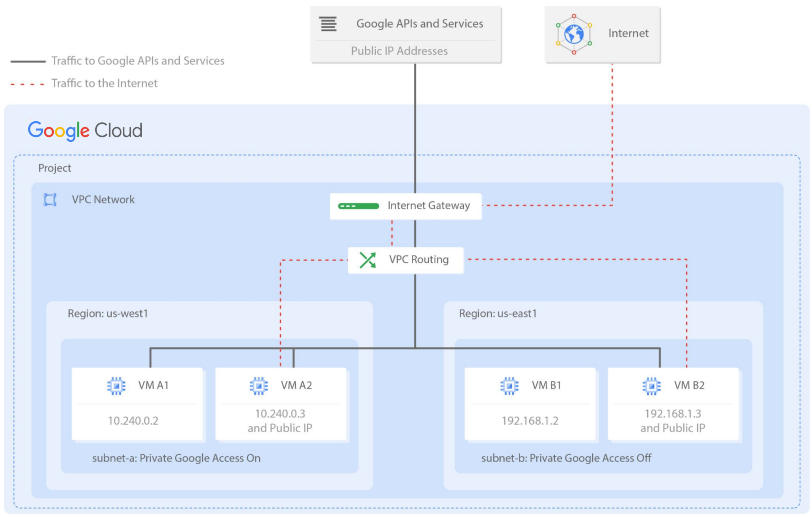


Now, as a general security best practice, we recommend using only assigning internal IP addresses to your VM instances wherever possible.

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

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 cannot directly access any of the private instances behind the Cloud NAT gateway. This helps you keep your VPC networks isolated and secure.

Private Google access to Google APIs and services

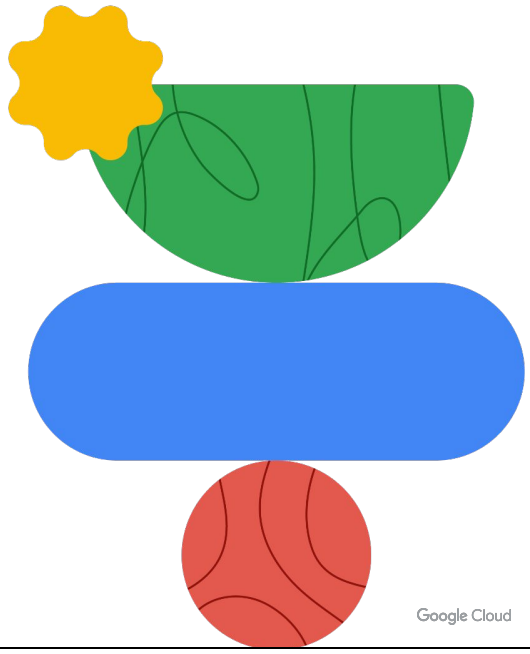


Similarly, you should enable Private Google Access to allow VM instances that only have internal IP addresses to reach the external IP addresses of Google APIs and services. For example, if your private VM instance needs to access a Cloud Storage bucket, you need to enable Private Google Access.

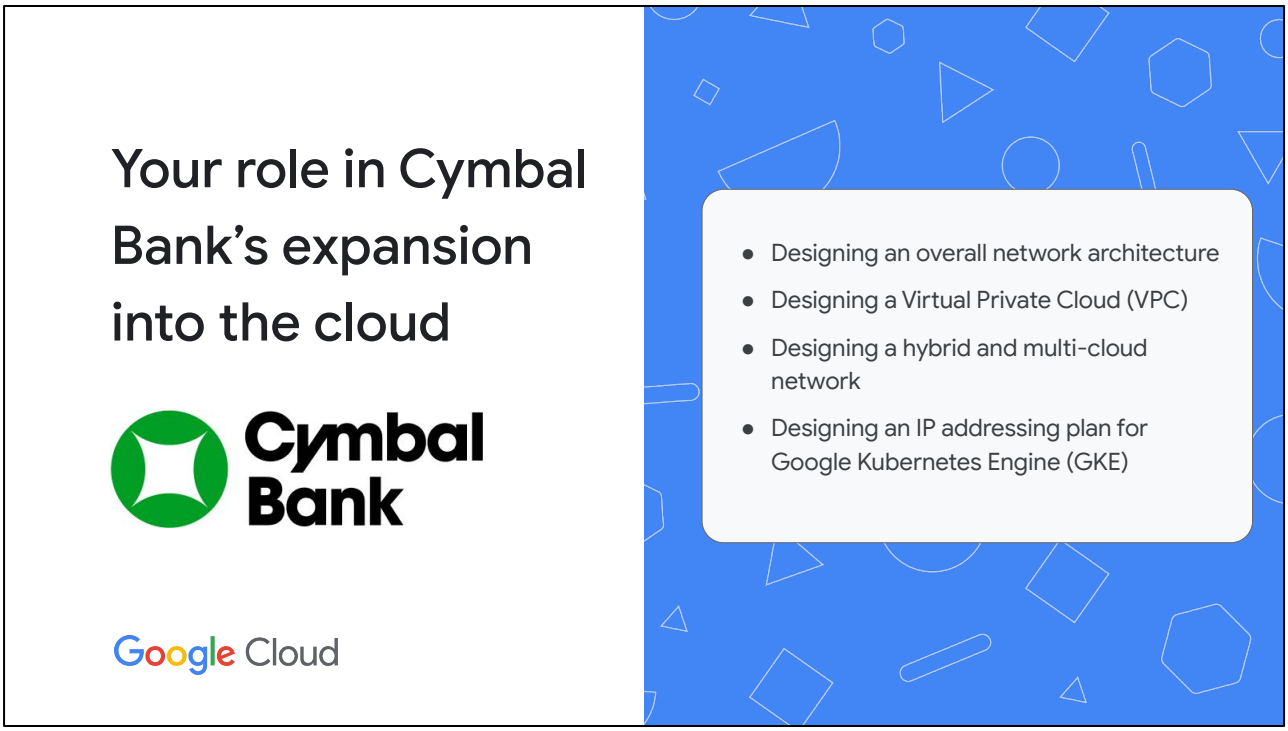
You enable Private Google Access on a subnet-by-subnet basis. As you can see in this diagram, subnet-a has Private Google Access enabled, and subnet-b has it disabled. This allows VM A1 to access Google APIs and services, even though it has no external IP address.

Private Google Access has no effect on instances that have external IP addresses. That's why VMs A2 and B2 can access Google APIs and services. The only VM that can't access those APIs and services is VM B1. This VM has no public IP address, and it is in a subnet where Google Private Access is disabled.

Assessing your current Google Cloud networking knowledge



Let's continue ~~start~~ by exploring the breadth of considerations involved in the design of a Google Cloud network and the role of the Professional Cloud Network Engineer at Cymbal Bank.



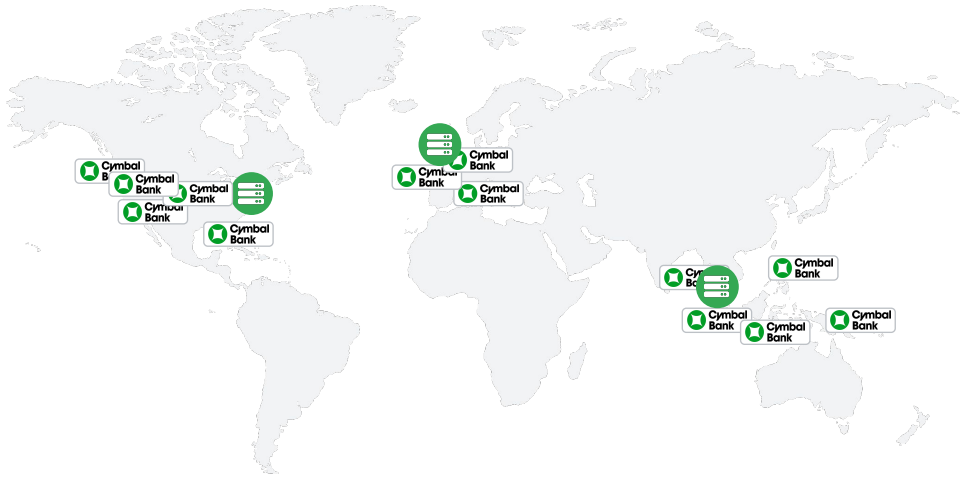
Cymbal Bank plans to extend its on-premises data center infrastructure to connect into Google Cloud to support a hybrid cloud model. As a Professional Cloud Network Engineer, you play an integral role in designing and planning the network infrastructure.

Cymbal Bank plans to continue deploying some workloads in your on-premises data center environments various locations, while moving others to Google Cloud for deployment into cloud virtual infrastructure. You need to design a hybrid cloud environment to connect on-premises branch, office, and data center environments to the new cloud environment.

These workloads running on-premises and in the cloud will communicate and exchange significant amounts of data, both for real-time transactional workloads as well as streaming or batch analytics workloads. Your network design will require secure high bandwidth, low latency communication connecting Cymbal Bank's physical and virtual data center networks with the virtual private cloud (VPC) networks in Google Cloud.

To give you a better understanding of the types of considerations involved in designing and planning the network, let's review Cymbal Bank's existing infrastructure and the changes you will make with the upcoming cloud migration.

Cymbal Bank's existing on-premises infrastructure

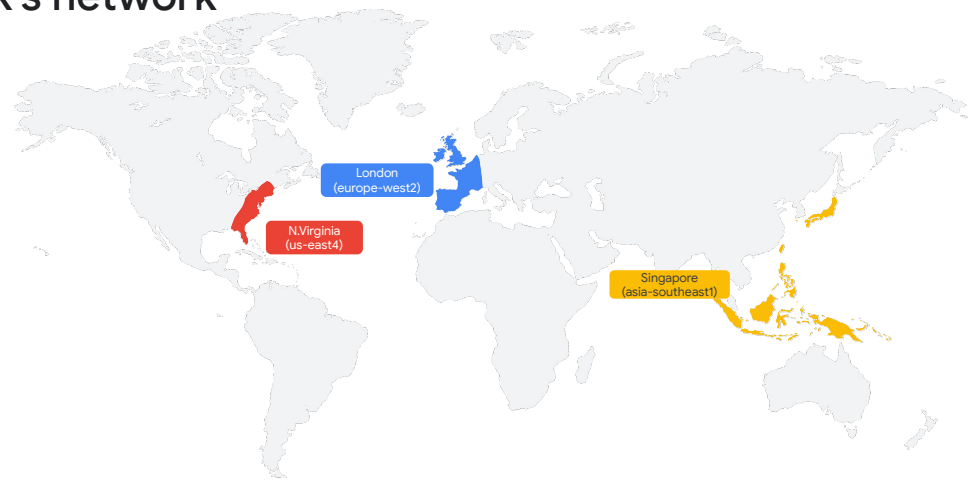


Cymbal Bank has data centers in New York (US), London (UK), and Singapore.

It also has branches and offices distributed across the US, western Europe, and southeast Asia.

Cymbal Bank deploys a mixture of monolithic and some microservices-based workloads, including both transactional and analytical workloads. Most workloads reside in data centers, but some run in branch and office environments.

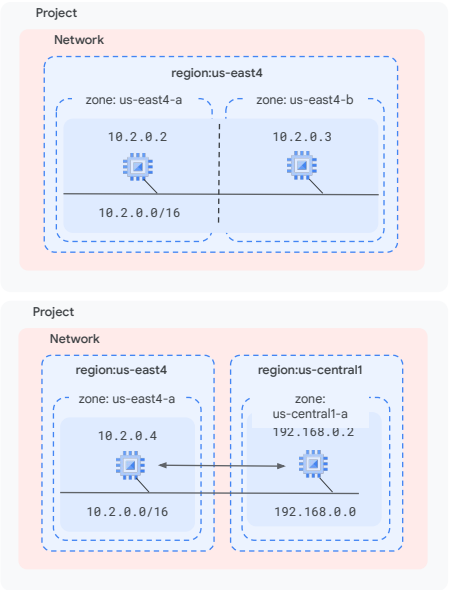
Choosing primary Google Cloud regions for Cymbal Bank's network



You want to deploy Google Cloud resources close to your data centers. Cymbal Bank plans to use Compute Engine, Google Kubernetes Engine (GKE), Cloud Storage, Dataflow, Dataproc, and BigQuery in its cloud solutions. The closest Google Cloud regions to Cymbal Bank's data centers all support these required features.

Secondary regions for higher availability

- Cymbal Bank will also deploy to secondary regions (which also support the required features)
  - Iowa (us-central1)
  - Belgium (europe-west1)
  - Jakarta (asia-southeast2)
- To increase availability they will deploy to at least 2 zones in each region

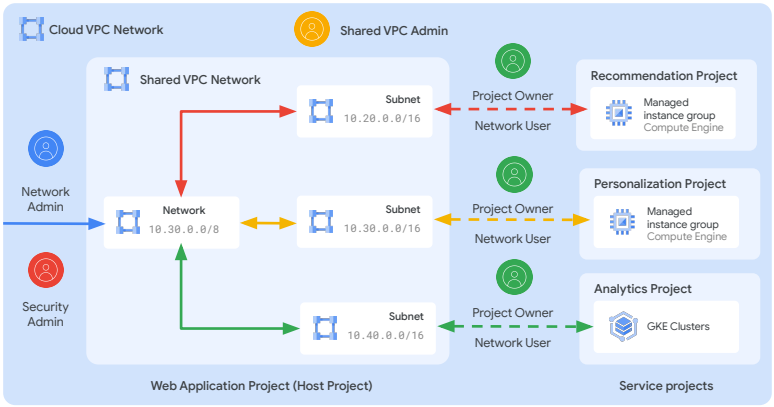


You also decide utilize other nearby regions in Iowa, Belgium, and Jakarta as secondary deployment locations. This will provide higher capacity and lower latency for users outside the primary regions. These secondary regions will also provide fallback in case of regional cloud failures in the primary regions.

The deployments will utilize at least two zones in each region to provide higher availability.

Cymbal Bank's VPC network architecture

- Deploy 4 shared VPC networks using subnet from a host project
- Service projects provided by teams will deploy resources into and use the shared VPC
- Assign access to network resources using Cloud IAM predefined roles
  - Shared VPC Admin
  - Compute Network User
  - Network Admin
  - Security Admin

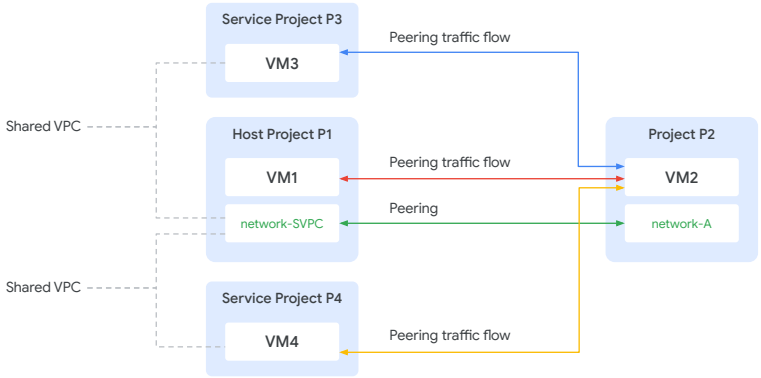


As a Professional Cloud Network Engineer your starting task is to design the network architecture for Cymbal Bank. You determine that Cymbal Bank will have four primary shared VPC networks in four different host projects corresponding to development, test, staging, and production environments. Each VPC will have subnets in the six primary and secondary regions. You will configure appropriate routes and firewall rules for the expected traffic profiles.

Cymbal Bank will have a large number of service projects using those four shared VPCs. Service projects will be deployed as quadruplets with development, testing, staging, and production projects provided per team, department, or product. Each component service project connects to its respective host project.

You will assign access to network resources using Cloud IAM predefined roles: Shared VPC Admin, Network Admin, Security Admin, and Network user roles.

Standalone VPC networks for ephemeral analytics workloads

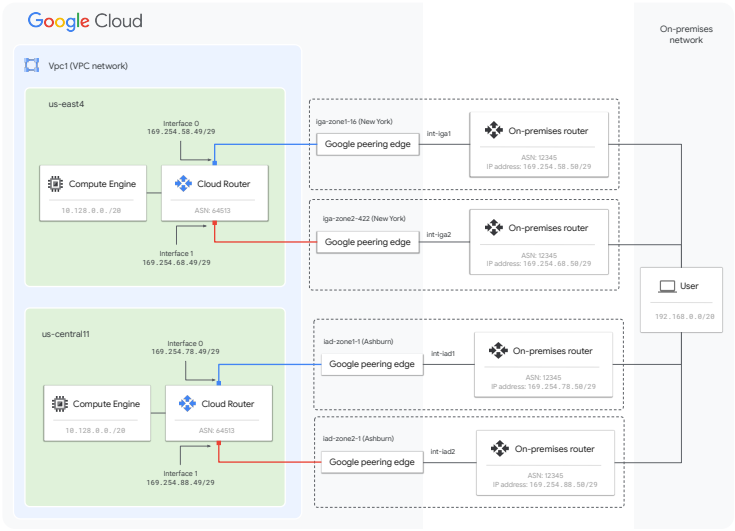


You decide that Cymbal Bank will also deploy standalone VPC networks for ephemeral analytics workloads. You will use VPC peering to connect them to the shared VPC networks as needed.

Connecting Cymbal Bank's network to Google Cloud

Dedicated Interconnect and Partner Interconnect will connect data centers to Google Cloud regions

- Mix of regular and high availability configurations
- Mix of layer 2 and layer 3 Partner Interconnect



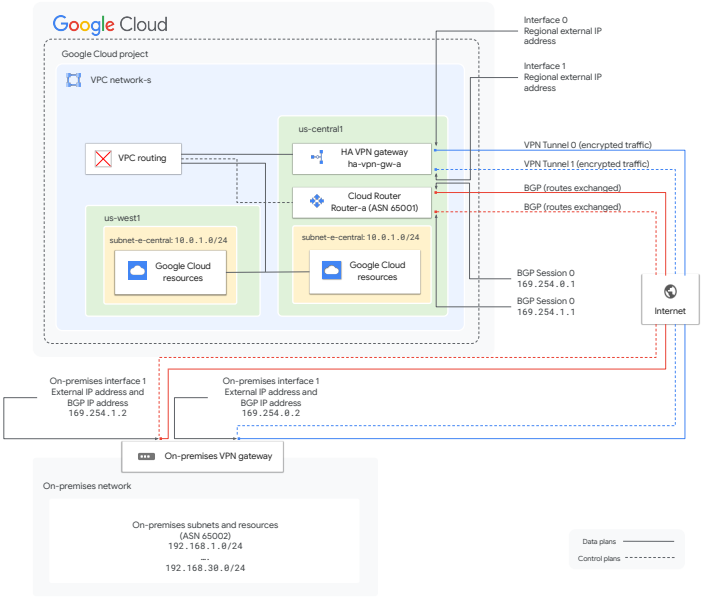
You decide to use Dedicated Interconnect connections to Google Cloud colocation facilities. You will also use Layer 2 and 3 Partner Interconnect connections. These connections will have a mix of regular and high availability configurations.

Connecting Cymbal Bank's network to Google Cloud

Cloud VPN connecting branches and offices to closest regions

- Mix of HA and Classic VPN
- Mix of static routing and dynamic routing with BGP and Cloud Routers

Note: Classic VPN partially deprecated in 2022

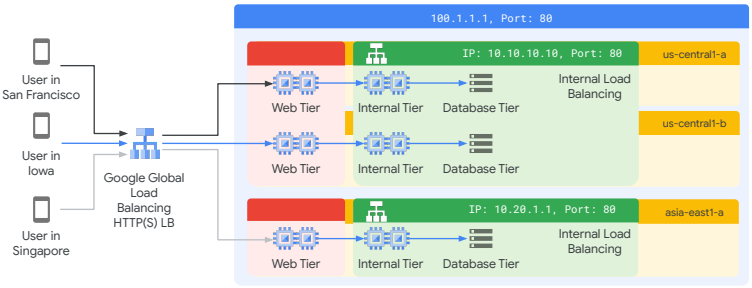


Cymbal Bank will also utilize both Classic and HA Cloud VPN with a mix of static and dynamic routing. You will use Cloud Router for private connectivity between Google Cloud and the satellite branches and offices.

Ensuring network availability and performance

Cloud load balancing

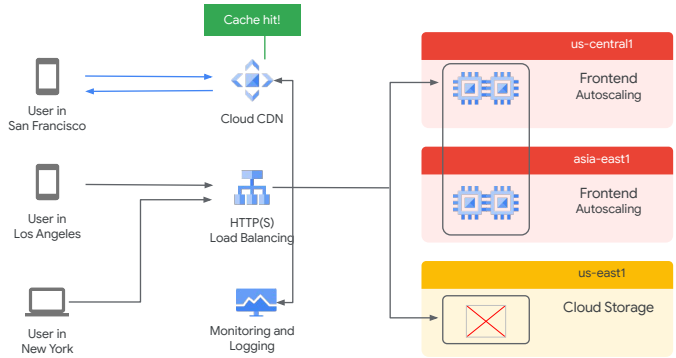
- Layer 7, utilizing global external and regional internal HTTP(S) LBs
- Provides capacity and low latency serving of static and dynamic resources



You plan to use Cloud Load Balancing with global external and regional internal HTTP(S) load balancers. This configuration will serve static and dynamic content with low latency.

Ensuring network availability and performance

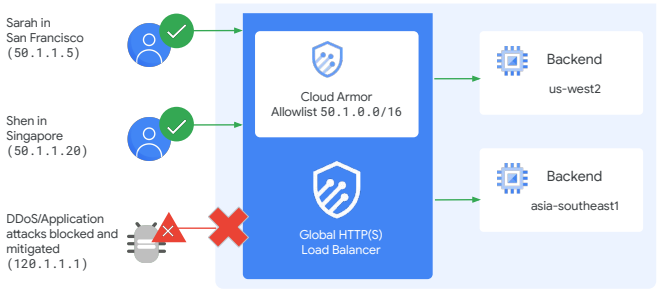
Cloud Content Delivery Network (CDN)
Caches static resources to increase capacity and reduce latency for static resources.



You will use Cloud CDN to provide caching of static resources to increase capacity and reduce latency. Cloud CDN improves serving capacity because most users are served resource requests from edge locations rather than having the requests enter the VPC and be served from the origin.

Protecting the network

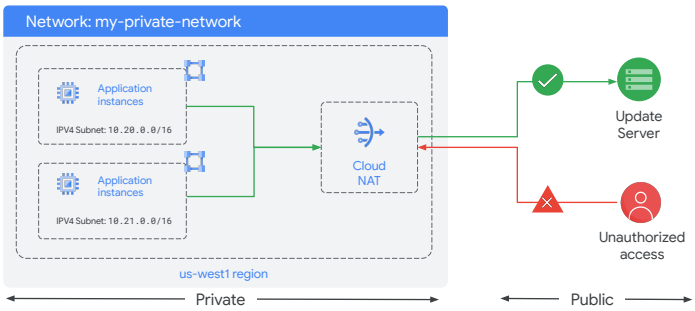
Cloud armor
Provides web-application firewall attack and DDoS protection.



Cymbal Bank will use Cloud Armor to provide DDoS and other attack protections to their public endpoints (which will all be exposed via Cloud Load Balancing)

Protecting the network

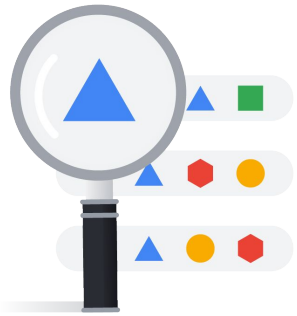
Cloud NAT
Provides ability to make requests to internet for resources with only private/internal IP connectivity.



They will use Cloud NAT to provide internet access to resources with no public/external IP address.

Network monitoring and debugging with operations suite

- VPC Flow logs
- Firewall logs
- Cloud Monitoring metrics for network activity
- Cloud trace for latency across microservices
- Logs may be exported to Pubsub, Cloud storage, or BigQuery

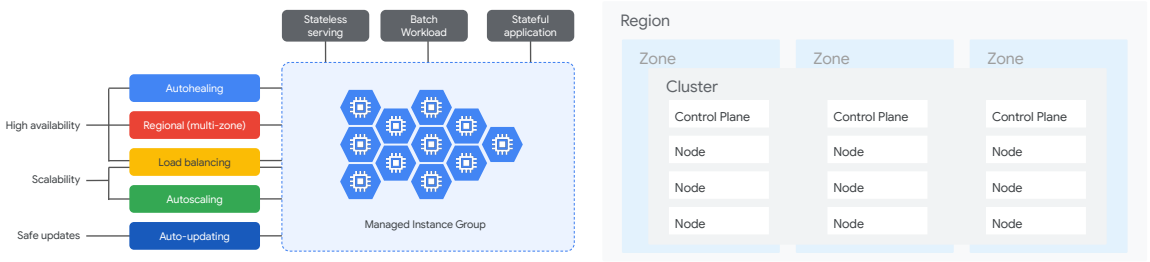


Google Cloud's operations suite lets you perform general network monitoring and debugging using VPC Flow logs and firewall logs. You'll use it for incident detection and response, latency tracking, network traffic flow or firewall debugging, and forensics.

Logs may also be exported to Pubsub, Cloud storage, or BigQuery for integration with external systems, long-term storage, and/or SQL analysis

Cymbal Bank networking in Google Cloud

- Deploy legacy monoliths as managed instance groups of VMs
- Deploy new microservices to GKE
- Convert some monoliths to microservices and deploy to GKE



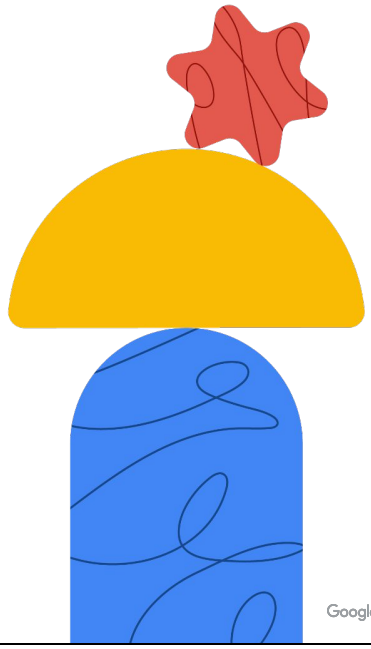
Some of Cymbal Bank's legacy monoliths cannot or will not be converted to microservices. These will be deployed into regional managed instance groups of VMs.

Cymbal Bank will deploy all its microservices to GKE. Some existing monoliths will be converted to microservices that will also be deployed into GKE.

You decide to utilize private VPC-native GKE clusters in standard mode to ensure security and maximum infrastructure flexibility.

Learning about Cymbal Bank's network design should give you a sense of the scope of considerations involved in planning a Google Cloud network. This is a vital part of your role as a Professional Cloud Network Engineer, so let's explore further with diagnostic questions focusing on this area.

Diagnostic questions



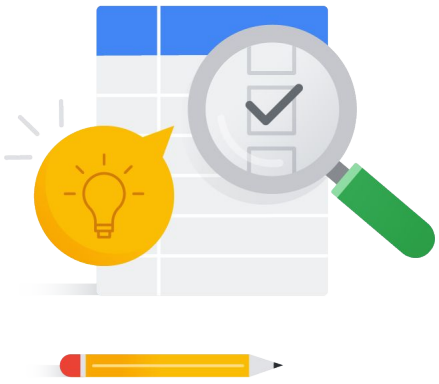
BREAK SLIDE

Exercise

20 mins | Individual

Diagnostic questions

- These questions are a subset of the questions available in the "Preparing for your PCNE Journey" course
- The questions are organised by the exam objectives listed in the PCNE Exam Guide
- Students should complete **all** the diagnostic questions in the "Preparing for your PCNE Journey" course as part of the self-study requirement



It's your turn to assess your experience and skills related to this section with some diagnostic questions. Remember, these questions are intended to help you understand, or diagnose, which areas you'll want to focus on in your study plan, so we don't expect you to know all the answers yet.

1.1 Diagnostic question 01 discussion

You are a network engineer designing a network IP plan and need to select an IP address range to use for a subnet. The subnet will need to host up to 2000 virtual machines, each to be assigned one IP address from the subnet range. It will also need to fit in the network IP range 10.1.0.0/16 and be as small as possible.

A. 10.1.1.0/24
B. 10.1.240.0/21
C. 10.1.1.0/21
D. 10.1.240.0/20

What subnet range should you use?

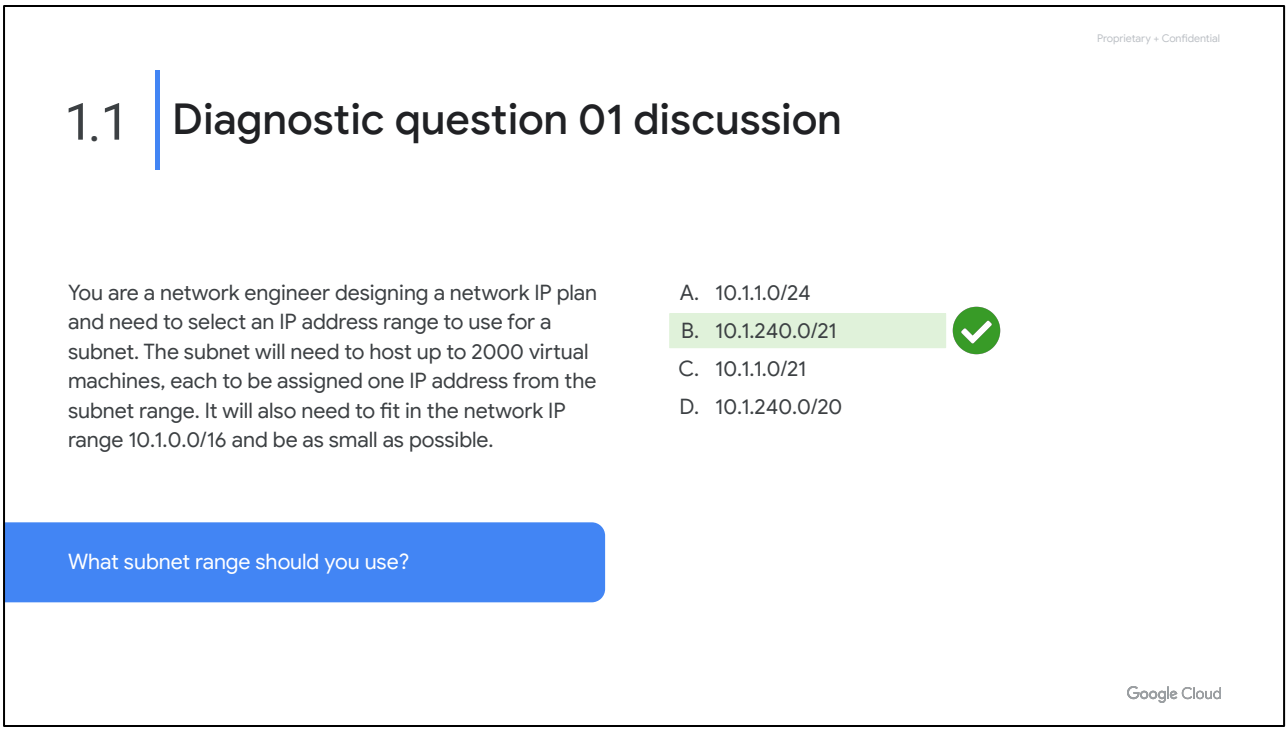
~~Feedback:~~
~~A: Incorrect. This range would have a maximum of 255 IP addresses and could not support 2000 virtual machines each having one IP address~~
~~*B: Correct! This range will satisfy the requirements. It has 2040 IP addresses and can therefore host 2000 virtual machines with one IP address per machine. It is the smallest range that could host this number of VMs, and it fits within the network range of 10.1.0.0/16~~
~~C: Incorrect. This range is invalid; the 3rd byte of the range mask occupies the range.~~
~~D: Incorrect. This range has 4080 IP addresses. However, 10.1.240.0/21 can host up to 2040 IP addresses and is therefore a better fit.~~

~~Where to look:~~ ~~Public learning resources such as https://www.coursera.org/learn/computer-networking~~

~~Content mapping:~~ ~~Google option outside of the PCNE learning path: The Bits and Bytes of Computer Networking~~

~~Summary:~~
~~There are networking fundamentals that are common to all environments, cloud or otherwise. A network engineer should have familiarity with such networking fundamentals. For example, a network engineer should be able to select subnet~~

~~ranges that satisfy requirements such as available IP addresses for assignment and prevent overlap.~~



Feedback:
A: Incorrect. This range would have a maximum of 255 IP addresses and could not support 2000 virtual machines each having one IP address
*B: Correct! This range will satisfy the requirements. It has 2040 IP addresses and can therefore host 2000 virtual machines with one IP address per machine. It is the smallest range that could host this number of VMs, and it fits within the network range of 10.1.0.0/16
C: Incorrect. This range is invalid; the 3rd byte of the range mask occupies the range.
D: Incorrect. This range has 4080 IP addresses. However, 10.1.240.0/21 can host up to 2040 IP addresses and is therefore a better fit.

Where to look: Public learning resources such as https://www.coursera.org/learn/computer-networking

Content mapping: Google option outside of the PCNE learning path: The Bits and Bytes of Computer Networking

Summary:
There are networking fundamentals that are common to all environments, cloud or otherwise. A network engineer should have familiarity with such networking fundamentals. For example, a network engineer should be able to select subnet

ranges that satisfy requirements such as available IP addresses for assignment and prevent overlap.

1.1 Diagnostic question 03 discussion

You are a network engineer designing a solution for hosting a Cymbal Bank web application in Google Cloud. The application will serve a collection of static and dynamic web resources served over HTTPS to users worldwide. You need to design a solution that maximizes availability while minimizing average user latency.

Which of the following features of Google Cloud networking can you utilize? (Select 2)

A. Cloud CDN could be used to cache static content resources at edge locations close to end-users, increasing their availability and minimizing their latency.

B. Cloud NAT could be used to provide outbound connectivity to the internet for resources with only internal IP addresses, thereby increasing their availability.

C. Cloud Armor could be used to provide protection against DDoS and injection attacks and thereby minimize solution latency.

D. An HTTPS load balancer with a backend service connected to a set of regional MIGs, distributed over the regions closest to the users, to improve availability and minimize latency.

E. Network Intelligence Center could be used to provide network insights, enabling the web application to be deployed in a configuration with maximum availability and minimal latency.

~~**Feedback:**~~

~~*A: Correct! Cloud CDN can be used to cache static content at edge locations. This would help maximize the availability and minimize the average latency for end users accessing those resources.~~

~~B: Incorrect. Cloud NAT can be used to provide outbound connectivity to the internet for resources with only internal IP addresses, but this does nothing to increase availability or minimize latency for end users interacting with those resources.~~

~~C: Incorrect. Cloud Armor can be used to provide protection against DDoS and injection attacks. This can improve solution availability in the presence of such attacks but does nothing to minimize average user latency.~~

~~*D: Correct! Using an HTTPS LB with a backend service connected to a set of regional MIGs distributed over the regions closest to the users would ensure high availability and minimal average user latency for serving dynamic web resources.~~

~~E: Incorrect. The Network Intelligence Center has features that can be used to troubleshoot network connectivity and performance issues, but this would be of limited applicability in designing a networking solution for high availability and low latency.~~

~~**Where to look**:~~

~~https://cloud.google.com/cdn/docs/overview~~

~~https://cloud.google.com/nat/docs/overview~~

https://cloud.google.com/load-balancing/docs/load-balancing-overview
https://cloud.google.com/armor/docs/cloud-armor-overview
https://cloud.google.com/network-intelligence-center/docs

Content mapping: Networking in Google Cloud

- Instructor-led training
  - M4 Load balancing
  - M5 Hybrid connectivity
  - M7 Network Design and Deployment
- OnDemand
  - M4 Load balancing
  - M1 Hybrid connectivity
  - M3 Network Design and Deployment

Summary:

Google Cloud provides many networking features. Cloud CDN is a content delivery network that caches static resources at Google edge locations around the world. It serves these resources to a large user base with maximal availability and minimal latency. Cloud NAT provides network address translation that enables private/internal IP-only resources to make secure requests to the internet. Cloud Armor is a web-application firewall (WAF) providing traffic scanning and filtering.It provides configurable protection against attack traffic including distributed denial of service (DDoS). Google Cloud Load Balancing provides managed software load balancing in many varieties including full global anycast load balancing to enable delivery traffic to the nearest regional backends using a single IP address. Google Cloud Network Intelligence Center provides several features such as Network Topology, Connectivity Testing, Performance Dashboard, and Firewall Insights. These features aid in network troubleshooting and performance debugging.

1.1 Diagnostic question 03 discussion

You are a network engineer designing a solution for hosting a Cymbal Bank web application in Google Cloud. The application will serve a collection of static and dynamic web resources served over HTTPS to users worldwide. You need to design a solution that maximizes availability while minimizing average user latency.

Which of the following features of Google Cloud networking can you utilize? (Select 2)

A. Cloud CDN could be used to cache static content resources at edge locations close to end-users, increasing their availability and minimizing their latency.

B. Cloud NAT could be used to provide outbound connectivity to the internet for resources with only internal IP addresses, thereby increasing their availability.

C. Cloud Armor could be used to provide protection against DDoS and injection attacks and thereby minimize solution latency.

D. An HTTPS load balancer with a backend service connected to a set of regional MIGs, distributed over the regions closest to the users, to improve availability and minimize latency.

E. Network Intelligence Center could be used to provide network insights, enabling the web application to be deployed in a configuration with maximum availability and minimal latency.

Feedback:
*A: Correct! Cloud CDN can be used to cache static content at edge locations. This would help maximize the availability and minimize the average latency for end users accessing those resources.
B: Incorrect. Cloud NAT can be used to provide outbound connectivity to the internet for resources with only internal IP addresses, but this does nothing to increase availability or minimize latency for end users interacting with those resources.
C: Incorrect. Cloud Armor can be used to provide protection against DDoS and injection attacks. This can improve solution availability in the presence of such attacks but does nothing to minimize average user latency.
*D: Correct! Using an HTTPS LB with a backend service connected to a set of regional MIGs distributed over the regions closest to the users would ensure high availability and minimal average user latency for serving dynamic web resources.
E: Incorrect. The Network Intelligence Center has features that can be used to troubleshoot network connectivity and performance issues, but this would be of limited applicability in designing a networking solution for high availability and low latency.

Where to look:
https://cloud.google.com/cdn/docs/overview
https://cloud.google.com/nat/docs/overview

https://cloud.google.com/load-balancing/docs/load-balancing-overview
https://cloud.google.com/armor/docs/cloud-armor-overview
https://cloud.google.com/network-intelligence-center/docs

Content mapping: Networking in Google Cloud

- Instructor-led training
  - M4 Load balancing
  - M5 Hybrid connectivity
  - M7 Network Design and Deployment
- OnDemand
  - M4 Load balancing
  - M1 Hybrid connectivity
  - M3 Network Design and Deployment

Summary:
Google Cloud provides many networking features. Cloud CDN is a content delivery network that caches static resources at Google edge locations around the world. It serves these resources to a large user base with maximal availability and minimal latency. Cloud NAT provides network address translation that enables private/internal IP-only resources to make secure requests to the internet. Cloud Armor is a web-application firewall (WAF) providing traffic scanning and filtering.It provides configurable protection against attack traffic including distributed denial of service (DDoS). Google Cloud Load Balancing provides managed software load balancing in many varieties including full global anycast load balancing to enable delivery traffic to the nearest regional backends using a single IP address. Google Cloud Network Intelligence Center provides several features such as Network Topology, Connectivity Testing, Performance Dashboard, and Firewall Insights. These features aid in network troubleshooting and performance debugging.

1.2 Diagnostic question 04 discussion

Cymbal Bank needs to create one or more VPC networks to host their cloud services in 3 regions: Northeastern US, Western Europe, and Southeast Asia. The services require bi-directional inter-regional communication on port 8443. The services receive external internet traffic on port 443.

What is the minimal network topology in Google Cloud that would satisfy these requirements?

A. 3 custom VPC networks, one in each region with one subnet each. The VPC networks all connected with VPC peering with default firewall rules, and custom routes added to support the traffic requirements.

B. 3 custom VPC networks, one in each region with one subnet each. The VPC networks all connected with VPC peering with default routes, and firewall rules added to support the traffic requirements.

C. 1 custom VPC network, with a subnet in each region. The VPC network has the default routes, and the appropriate firewall rules added to support the traffic requirements.

D. 1 custom VPC network, with a subnet in each region. The VPC network has default firewall rules and custom routes added to support the traffic requirements.

~~**Feedback:**~~

~~A: Incorrect. A single VPC network with 3 subnets is the minimal topology to satisfy these requirements.~~

~~B: Incorrect. A single VPC network with 3 subnets is the minimal topology to satisfy these requirements.~~

~~*C: Correct! This is the correct minimal topology satisfying the requirements.~~

~~D: Incorrect. The traffic requirements can be satisfied with the default routes but would require additional firewall rules.~~

~~**Where to look**:~~

~~https://cloud.google.com/vpc/docs/vpc~~

~~https://cloud.google.com/vpc/docs/firewalls~~

~~https://cloud.google.com/vpc/docs/routes~~

~~**Content Mapping**:~~

- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M1 Google Cloud VPC networking fundamentals~~
- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~
    - ~~M1 Google Cloud VPC networking fundamentals~~

Summary:

VPC networks are global resources in Google Cloud and contain regional subnetworks. Auto VPC networks will have at least one subnet in each Google Cloud region. Custom VPC networks can have subnetworks in only desired regions. Each subnetwork will have at least one primary subnet range and none of the primary or secondary subnet ranges can overlap in a single VPC. Each VPC network has a set of default routes routing the subnet IP ranges of each subnet to that subnet and a route for all other IP addresses to the internet. Other custom routes can also be added. Firewall rules are the primary mechanism for traffic control and can be used to allow or deny traffic matching the configured rule parameters. Firewall rules are evaluated in priority order. The two implicit firewall rules in every VPC - implicit deny all ingress and allow all egress rules - have the lowest priority.

1.2 Diagnostic question 04 discussion

Cymbal Bank needs to create one or more VPC networks to host their cloud services in 3 regions: Northeastern US, Western Europe, and Southeast Asia. The services require bi-directional inter-regional communication on port 8443. The services receive external internet traffic on port 443.

What is the minimal network topology in Google Cloud that would satisfy these requirements?

A. 3 custom VPC networks, one in each region with one subnet each. The VPC networks all connected with VPC peering with default firewall rules, and custom routes added to support the traffic requirements.

B. 3 custom VPC networks, one in each region with one subnet each. The VPC networks all connected with VPC peering with default routes, and firewall rules added to support the traffic requirements.

C. 1 custom VPC network, with a subnet in each region. The VPC network has the default routes, and the appropriate firewall rules added to support the traffic requirements.

D. 1 custom VPC network, with a subnet in each region. The VPC network has default firewall rules and custom routes added to support the traffic requirements.

Feedback:
A: Incorrect. A single VPC network with 3 subnets is the minimal topology to satisfy these requirements.
B: Incorrect. A single VPC network with 3 subnets is the minimal topology to satisfy these requirements.
*C: Correct! This is the correct minimal topology satisfying the requirements.
D: Incorrect. The traffic requirements can be satisfied with the default routes but would require additional firewall rules.

Where to look:
https://cloud.google.com/vpc/docs/vpc
https://cloud.google.com/vpc/docs/firewalls
https://cloud.google.com/vpc/docs/routes

Content Mapping:
- Instructor-led training
  - Networking in Google Cloud
    - M1 Google Cloud VPC networking fundamentals
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M1 Google Cloud VPC networking fundamentals

Summary:

VPC networks are global resources in Google Cloud and contain regional subnetworks. Auto VPC networks will have at least one subnet in each Google Cloud region. Custom VPC networks can have subnetworks in only desired regions. Each subnetwork will have at least one primary subnet range and none of the primary or secondary subnet ranges can overlap in a single VPC. Each VPC network has a set of default routes routing the subnet IP ranges of each subnet to that subnet and a route for all other IP addresses to the internet. Other custom routes can also be added. Firewall rules are the primary mechanism for traffic control and can be used to allow or deny traffic matching the configured rule parameters. Firewall rules are evaluated in priority order. The two implicit firewall rules in every VPC - implicit deny all ingress and allow all egress rules- have the lowest priority.

1.2 Diagnostic question 05 discussion

Sarah is a network architect responsible for the network design between Cymbal Bank's on-premises network and Google Cloud resources, and also between Cymbal Bank's Google Cloud resources and a partner company's Google Cloud resources. These connections must provide private IP connectivity and support up to 100 Gbps of data exchange with minimum possible latency.

Which options satisfy these requirements? (Select 2)

A. Shared VPC network connecting Google Cloud resources for Cymbal Bank and the partner company
B. VPC peering between VPC networks for Cymbal Bank and the partner company
C. A Dedicated Interconnect connection between Cymbal Bank's on-premises network and their Google Cloud VPC network
D. A Cloud VPN tunnel between Cymbal Bank's on-premises network and their Google Cloud VPC network
E. 50 Cloud VPN tunnels between Cymbal Bank's on-premises network and their Google Cloud VPC network

~~**Feedback:**~~
~~A: Incorrect. A Shared VPC network cannot be used to connect resources across separate organizations.~~
~~*B: Correct!. VPC peering allows for private IP connectivity between Google Cloud resources across organizations and is the lowest latency and highest bandwidth option for such connectivity~~
~~*C: Correct! Dedicated Interconnect provides private IP connectivity with bandwidths ranging from 10-200 Gbps per interconnect link and has the lowest possible latency.~~
~~D: Incorrect. Cloud VPN maximum bandwidth is 3 Gps per tunnel, which is considerably less than the 100 Gbps that is required. Also Cloud VPN has significantly more latency than Cloud Interconnect and Dedicated Interconnect.~~
~~E. Incorrect. Cloud VPN maximum bandwidth is 3 Gps per tunnel, 50 tunnels would provide more than the 100 Gbps that is required. However Cloud VPN has significantly more latency than Cloud Interconnect and Dedicated Interconnect.~~

~~**Where to look**:~~
~~https://cloud.google.com/vpc/docs/shared-vpc~~
~~https://cloud.google.com/vpc/docs/vpc-peering~~
~~https://cloud.google.com/network-connectivity/docs/how-to/choose-product~~

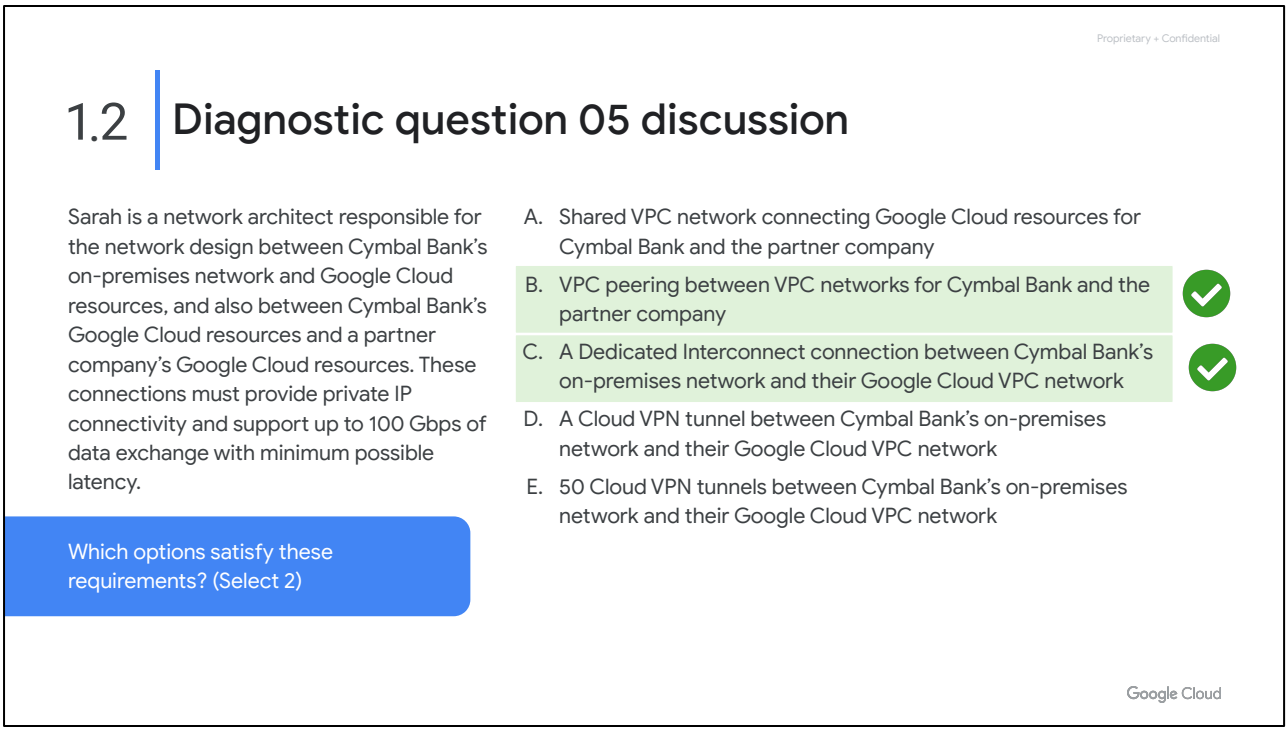
~~**Content Mapping**:~~

- Instructor-led training
  - Networking in Google Cloud
    - M3 Sharing networks across projects
    - M5 Hybrid connectivity
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M3 Sharing networks across projects
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M1 Hybrid connectivity

Summary:

There are 2 main approaches to connecting resources using private/internal IP communication across projects or VPC networks in Google Cloud. Shared VPC provides a centralized networking model. VPC peering provides a decentralized model. Shared VPC allows resources from multiple projects to be placed in a common VPC network owned by a host project. VPC peering allows VPC networks to be connected within or across projects, or even across organizations.

There are 3 main approaches to connecting resources using private/internal IP communication between on-premises and cloud environments in Google Cloud. Dedicated Interconnect provides the highest bandwidth and lowest latencies, but requires connectivity to and installing a router in a Google Cloud colocation facility. Partner Interconnect provides a variety of sub-10gbps bandwidth options for customers who do not need the full 10 or 100gbps that Dedicated Interconnect provides, and allows meeting Google Cloud Partners in many more locations around the world. Cloud VPN typically provides for lower bandwidths and higher latency, but is relatively inexpensive and quick to set up.



Feedback:
A: Incorrect. A Shared VPC network cannot be used to connect resources across separate organizations.
*B: Correct!. VPC peering allows for private IP connectivity between Google Cloud resources across organizations and is the lowest latency and highest bandwidth option for such connectivity
*C: Correct! Dedicated Interconnect provides private IP connectivity with bandwidths ranging from 10-200 Gbps per interconnect link and has the lowest possible latency.
D: Incorrect. Cloud VPN maximum bandwidth is 3 Gps per tunnel, which is considerably less than the 100 Gbps that is required. Also Cloud VPN has significantly more latency than Cloud Interconnect and Dedicated Interconnect.
E. Incorrect. Cloud VPN maximum bandwidth is 3 Gps per tunnel, 50 tunnels would provide more than the 100 Gbps that is required. However Cloud VPN has significantly more latency than Cloud Interconnect and Dedicated Interconnect.

Where to look:
https://cloud.google.com/vpc/docs/shared-vpc
https://cloud.google.com/vpc/docs/vpc-peering
https://cloud.google.com/network-connectivity/docs/how-to/choose-product

Content Mapping:

- Instructor-led training
    - Networking in Google Cloud
        - M3 Sharing networks across projects
        - M5 Hybrid connectivity
- OnDemand
    - Networking in Google Cloud: Defining and implementing networks
        - M3 Sharing networks across projects
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M1 Hybrid connectivity

Summary:
There are 2 main approaches to connecting resources using private/internal IP communication across projects or VPC networks in Google Cloud. Shared VPC provides a centralized networking model. VPC peering provides a decentralized model. Shared VPC allows resources from multiple projects to be placed in a common VPC network owned by a host project. VPC peering allows VPC networks to be connected within or across projects, or even across organizations.

There are 3 main approaches to connecting resources using private/internal IP communication between on-premises and cloud environments in Google Cloud. Dedicated Interconnect provides the highest bandwidth and lowest latencies, but requires connectivity to and installing a router in a Google Cloud colocation facility. Partner Interconnect provides a variety of sub-10gbps bandwidth options for customers who do not need the full 10 or 100gbps that Dedicated Interconnect provides, and allows meeting Google Cloud Partners in many more locations around the world. Cloud VPN typically provides for lower bandwidths and higher latency, but is relatively inexpensive and quick to set up.

1.2 Diagnostic question 06 discussion

You are selecting Google Cloud locations to deploy Google Cloud VMs. You have general requirements to maximize availability and reduce average user latency with a lower priority goal of reducing networking costs. The users served by these VMs will be in Toronto and Montreal. You must deploy workloads requiring instances at 99.5% availability in Toronto and 99.99% availability in Montreal. These instances all exchange a large amount of traffic among themselves.

Which deployment option satisfies these requirements?

A. Deploy instances in multiple zones in the northamerica-northeast1 (Montreal) and northamerica-northeast2 (Toronto) regions.

B. Deploy instances in a single zone in the northamerica-northeast1 (Montreal) and northamerica-northeast2 (Toronto) regions.

C. Deploy instances in a single zone in the northamerica-northeast1 (Montreal) region and multiple zones in the northamerica-northeast2 (Toronto) region.

D. Deploy instances in multiple zones in the northamerica-northeast1 (Montreal) region and a single zone in the northamerica-northeast2 (Toronto).

~~**Feedback:**~~

~~A: Incorrect. This would provide higher than necessary availability in Toronto and increase the networking costs in that region by incurring inter-zone traffic.~~

~~B: Incorrect. This would not provide the required availability in Montreal as single-zone deployments would not provide 99.99% availability.~~

~~C: Incorrect.This would provide higher than necessary availability in Toronto and increase the networking costs in that region by incurring inter-zone traffic. It would also not provide the required availability in Montreal as single-zone deployments would not provide 99.99% availability.~~

~~*D: Correct! This satisfies the availability, latency and cost requirements. It ensures the lowest possible latency for users in Toronto and Montreal. It provides the desired availability (99.5% in Toronto and 99.99% in Montreal). By minimizing inter-zone network traffic, this solution minimizes networking costs.~~

~~**Where to look**:~~

~~https://cloud.google.com/about/locations~~
~~https://cloud.google.com/compute/docs/regions-zones~~
~~https://cloud.google.com/compute/docs/regions-zones/global-regional-zonal-resources~~
~~https://cloud.google.com/vpc/network-pricing~~
~~https://cloud.google.com/compute/sla~~

Content mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M1 Google Cloud VPC networking fundamentals
    - M4 Load balancing
    - M7 Network Design and Deployment
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M1 Google Cloud VPC networking fundamentals
    - M4 Load balancing
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M3 Network Design and Deployment

Summary:

Google Cloud offers 3 zones across approximately 30 regions (4 zones in us-central1 in Iowa). Though many capabilities are available across all regions, there are some differences in supported features and capabilities by region. Resources may be zonal, regional, multi-regional, or global with implications for availability, latency, and data residency. Placing replica resources across multiple zones increases availability (protecting against resource and zone failure). Placing replica resources across multiple regions can further increase availability, protect against regional outage, and reduce average latency for users close to those regions.

1.2 Diagnostic question 06 discussion

You are selecting Google Cloud locations to deploy Google Cloud VMs. You have general requirements to maximize availability and reduce average user latency with a lower priority goal of reducing networking costs. The users served by these VMs will be in Toronto and Montreal. You must deploy workloads requiring instances at 99.5% availability in Toronto and 99.99% availability in Montreal. These instances all exchange a large amount of traffic among themselves.

Which deployment option satisfies these requirements?

A. Deploy instances in multiple zones in the northamerica-northeast1 (Montreal) and northamerica-northeast2 (Toronto) regions.

B. Deploy instances in a single zone in the northamerica-northeast1 (Montreal) and northamerica-northeast2 (Toronto) regions.

C. Deploy instances in a single zone in the northamerica-northeast1 (Montreal) region and multiple zones in the northamerica-northeast2 (Toronto) region.

D. Deploy instances in multiple zones in the northamerica-northeast1 (Montreal) region and a single zone in the northamerica-northeast2 (Toronto).

Feedback:
A: Incorrect. This would provide higher than necessary availability in Toronto and increase the networking costs in that region by incurring inter-zone traffic.
B: Incorrect. This would not provide the required availability in Montreal as single-zone deployments would not provide 99.99% availability.
C: Incorrect.This would provide higher than necessary availability in Toronto and increase the networking costs in that region by incurring inter-zone traffic. It would also not provide the required availability in Montreal as single-zone deployments would not provide 99.99% availability.
*D: Correct! This satisfies the availability, latency and cost requirements. It ensures the lowest possible latency for users in Toronto and Montreal. It provides the desired availability (99.5% in Toronto and 99.99% in Montreal). By minimizing inter-zone network traffic, this solution minimizes networking costs.

Where to look:
https://cloud.google.com/about/locations
https://cloud.google.com/compute/docs/regions-zones
https://cloud.google.com/compute/docs/regions-zones/global-regional-zonal-resources
https://cloud.google.com/vpc/network-pricing
https://cloud.google.com/compute/sla

Content mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M1 Google Cloud VPC networking fundamentals
    - M4 Load balancing
    - M7 Network Design and Deployment
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M1 Google Cloud VPC networking fundamentals
    - M4 Load balancing
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M3 Network Design and Deployment

Summary:

Google Cloud offers 3 zones across approximately 30 regions (4 zones in us-central1 in Iowa). Though many capabilities are available across all regions, there are some differences in supported features and capabilities by region. Resources may be zonal, regional, multi-regional, or global with implications for availability, latency, and data residency. Placing replica resources across multiple zones increases availability (protecting against resource and zone failure). Placing replica resources across multiple regions can further increase availability, protect against regional outage, and reduce average latency for users close to those regions.

1.3 Diagnostic question 08 discussion

To reduce latency, you will be replacing an existing Cloud VPN Classic VPN connection. You will connect your organization's on-premises data center to Google Cloud resources in a VPC network with all resources in a single subnet and region using private/internal IP connectivity. The connection will need to support 1.5 Gbps of traffic. Due to cost considerations, you would like to order the option that provides just enough bandwidth and not more but must have significantly lower latency than the existing Cloud VPN connection.

What should you use?

A. A 10 Gbps dedicated interconnect connection with one 10 Gbps VLAN attachments
B. A 2 Gbps dedicated interconnect connection with one 2 Gbps VLAN attachments
C. A Partner interconnect connection with 1 or 2 VLAN attachments
D. A Cloud VPN HA VPN connection with cloud router

~~**Feedback:**~~

~~A: This option will not be the lowest cost as it involves purchasing the 10 Gbps connection. Only 1.5 Gbps is required and can be purchased at lower cost through Partner Interconnect.~~

~~B: This option is not possible. Dedicated Interconnect connections start at 10 Gbps.~~

~~*C: This option will be the most cost effective among the options that would satisfy the requirement to reduce the latency significantly compared to the previous Cloud VPN connection.~~

~~D: This option will be inexpensive but would not reduce the latency significantly relative to the Cloud VPN Classic VPN connection.~~

~~**Where to look**:~~

~~https://cloud.google.com/network-connectivity/docs/interconnect/concepts/overview~~
~~https://cloud.google.com/network-connectivity/docs/interconnect/concepts/dedicated-overview~~
~~https://cloud.google.com/network-connectivity/docs/interconnect/concepts/partner-overview~~
~~https://cloud.google.com/network-connectivity/docs/interconnect/concepts/terminology~~
~~https://cloud.google.com/network-connectivity/docs/interconnect/pricing~~
~~https://cloud.google.com/network-connectivity/docs/how-to/choose-product#cloud-interconnect~~

Content Mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M5 Hybrid Connectivity
- OnDemand
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M1 Hybrid Connectivity

Summary:

At a high level there are two options for Interconnect: Dedicated Interconnect and Partner Interconnect. Dedicated and Partner Interconnect can provide higher bandwidths and lower latencies. Dedicated Interconnect requires installing a router in a Google co-location facility and requires purchasing at least a 10 Gbps connection. Partner Interconnect can provide lower bandwidths and will be available at more locations from various providers. It can provide connections as low as 50 Mbps, which means it would typically be more cost effective at bandwidths lower than 10 Gbps.

1.3 Diagnostic question 08 discussion

To reduce latency, you will be replacing an existing Cloud VPN Classic VPN connection. You will connect your organization's on-premises data center to Google Cloud resources in a VPC network with all resources in a single subnet and region using private/internal IP connectivity. The connection will need to support 1.5 Gbps of traffic. Due to cost considerations, you would like to order the option that provides just enough bandwidth and not more but must have significantly lower latency than the existing Cloud VPN connection.

What should you use?

A. A 10 Gbps dedicated interconnect connection with one 10 Gbps VLAN attachments
B. A 2 Gbps dedicated interconnect connection with one 2 Gbps VLAN attachments
C. A Partner interconnect connection with 1 or 2 VLAN attachments
D. A Cloud VPN HA VPN connection with cloud router

Feedback:
A: This option will not be the lowest cost as it involves purchasing the 10 Gbps connection. Only 1.5 Gbps is required and can be purchased at lower cost through Partner Interconnect.
B: This option is not possible. Dedicated Interconnect connections start at 10 Gbps.
*C: This option will be the most cost effective among the options that would satisfy the requirement to reduce the latency significantly compared to the previous Cloud VPN connection.
D: This option will be inexpensive but would not reduce the latency significantly relative to the Cloud VPN Classic VPN connection.

Where to look:
https://cloud.google.com/network-connectivity/docs/interconnect/concepts/overview
https://cloud.google.com/network-connectivity/docs/interconnect/concepts/dedicated-overview
https://cloud.google.com/network-connectivity/docs/interconnect/concepts/partner-overview
https://cloud.google.com/network-connectivity/docs/interconnect/concepts/terminology
https://cloud.google.com/network-connectivity/docs/interconnect/pricing
https://cloud.google.com/network-connectivity/docs/how-to/choose-product#cloud-interconnect

Content Mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M5 Hybrid Connectivity
- OnDemand
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M1 Hybrid Connectivity

Summary:

At a high level there are two options for Interconnect: Dedicated Interconnect and Partner Interconnect. Dedicated and Partner Interconnect can provide higher bandwidths and lower latencies. Dedicated Interconnect requires installing a router in a Google co-location facility and requires purchasing at least a 10 Gbps connection. Partner Interconnect can provide lower bandwidths and will be available at more locations from various providers. It can provide connections as low as 50 Mbps, which means it would typically be more cost effective at bandwidths lower than 10 Gbps.

1.4 Diagnostic question 09 discussion

You need to create a GKE cluster, be able to connect to pod IP addresses from your on-premises environment, and control access to pods directly using firewall rules. You will need to support 300 nodes, 30000 pods, and 2000 services.

Which configuration satisfies these requirements?

A. A GKE route-based cluster in a subnet with primary IP range 10.0.240.0/20 and pod IP range of 10.1.0.0/16

B. A GKE route-based cluster in a subnet with primary IP range 10.0.240.0/20 and pod IP range of 10.252.0.0/14

C. A GKE VPC-native cluster in a subnet with primary IP range 10.0.240.0/20, pod IP range of 10.252.0.0/15, and service IP range of 10.0.224.0/20

D. A GKE VPC-native cluster in a subnet with primary IP range 10.0.240.0/20, pod IP range of 10.252.0.0/16, and service IP range of 10.0.224.0/20

~~**Feedback:**~~

~~A: Incorrect. A route-based GKE cluster will not satisfy the first 2 requirements. Additionally, the pod IP range would not be large enough to support the required number of nodes and pods.~~

~~B: Incorrect. A route-based GKE cluster will not satisfy the first 2 requirements~~

~~*C: Correct! This option will satisfy all requirements. A VPC-native cluster will satisfy the first 2 requirements and the provided ranges will support the required number of nodes, pods, and services.~~

~~D: Incorrect. This option will satisfy the first 2 requirements (being a VPC-native cluster), but the pod IP range will not be sufficient to hold the required number of nodes and pods.~~

~~**Where to look:**~~

~~https://cloud.google.com/kubernetes-engine/docs/concepts/types-of-clusters~~
~~https://cloud.google.com/kubernetes-engine/docs/concepts/alias-ips~~
~~https://cloud.google.com/kubernetes-engine/docs/how-to/alias-ips~~
~~https://cloud.google.com/kubernetes-engine/docs/how-to/routes-based-cluster~~

~~**Content Mapping:** Not covered in learning path~~

~~**Summary:**~~

With respect to network routing and IP planning, there are two main approaches to deploying GKE: routes-based or VPC-native. VPC-native is the newer and recommended approach that provides several benefits. It is important to be aware of the subtle differences in how to select the correct IP ranges to use with each type, as well as the supported numbers of resources based on size of the ranges.

1.4 Diagnostic question 09 discussion

You need to create a GKE cluster, be able to connect to pod IP addresses from your on-premises environment, and control access to pods directly using firewall rules. You will need to support 300 nodes, 30000 pods, and 2000 services.

Which configuration satisfies these requirements?

A. A GKE route-based cluster in a subnet with primary IP range 10.0.240.0/20 and pod IP range of 10.1.0.0/16

B. A GKE route-based cluster in a subnet with primary IP range 10.0.240.0/20 and pod IP range of 10.252.0.0/14

C. A GKE VPC-native cluster in a subnet with primary IP range 10.0.240.0/20, pod IP range of 10.252.0.0/15, and service IP range of 10.0.224.0/20

D. A GKE VPC-native cluster in a subnet with primary IP range 10.0.240.0/20, pod IP range of 10.252.0.0/16, and service IP range of 10.0.224.0/20

Feedback:
A: Incorrect. A route-based GKE cluster will not satisfy the first 2 requirements. Additionally, the pod IP range would not be large enough to support the required number of nodes and pods.
B: Incorrect. A route-based GKE cluster will not satisfy the first 2 requirements
*C: Correct! This option will satisfy all requirements. A VPC-native cluster will satisfy the first 2 requirements and the provided ranges will support the required number of nodes, pods, and services.
D: Incorrect. This option will satisfy the first 2 requirements (being a VPC-native cluster), but the pod IP range will not be sufficient to hold the required number of nodes and pods.

Where to look:
https://cloud.google.com/kubernetes-engine/docs/concepts/types-of-clusters
https://cloud.google.com/kubernetes-engine/docs/concepts/alias-ips
https://cloud.google.com/kubernetes-engine/docs/how-to/alias-ips
https://cloud.google.com/kubernetes-engine/docs/how-to/routes-based-cluster

Content Mapping: Not covered in learning path

Summary:

With respect to network routing and IP planning, there are two main approaches to deploying GKE: routes-based or VPC-native. VPC-native is the newer and recommended approach that provides several benefits. It is important to be aware of the subtle differences in how to select the correct IP ranges to use with each type, as well as the supported numbers of resources based on size of the ranges.

1.4 Diagnostic question 10 discussion

Cymbal Bank wants to ensure communication from their on-premises data centers to the GKE control plane stays private using internal IP communication and their Dedicated Interconnect links. However, they will need to allow administrators to periodically connect to the cluster control plane from remote internet-accessible locations that don't have access to the on-premises private network. You want to select a configuration and connection approach that will enable these requirements while providing the highest security.

What should you do?

A. Deploy a private GKE cluster with public endpoint access enabled and authorized networks disabled.

B. Deploy a private GKE cluster with public endpoint access enabled and authorized networks enabled. Configure authorized networks for the cluster to include all remote source IP ranges that administrators may connect from.

C. Deploy a private GKE cluster with public endpoint access disabled. Create a VM in the same subnet with only an internal IP address and provide IAP tunnel based SSH access to remote administrators for this VM. Have remote administrators connect via IAP tunnel SSH to this VM when requiring access to the GKE cluster control plane.

D. Deploy a private GKE cluster with public endpoint access disabled. Provide remote administrators IAP tunnel based SSH access to a node in the cluster. Have remote administrators connect via an IAP tunnel SSH to this node when requiring access to the GKE cluster control plane.

~~**Feedback:**~~

~~A: Incorrect. This option satisfies the requirements; however, it is the least secure option because it provides access to the control plane from any public IP address.~~

~~B: Incorrect. This option satisfies the requirements; however, it is not as secure as options C and D because it provides access to the control plane from a set of public IP addresses/ranges.~~

~~*C: Correct! This options satisfies the requirements in the most secure way by not providing any public access to the control plane and no private access to the cluster nodes.~~

~~D: Incorrect.Though satisfying requirements, this approach is slightly less secure than approach C as it provides direct node access to the remote administrators when only control plane access is required. (Option C only provides access to the control plane without access to the nodes.)~~

~~**Where to look:**~~

~~https://cloud.google.com/kubernetes-engine/docs/concepts/private-cluster-concept~~
~~https://cloud.google.com/kubernetes-engine/docs/how-to/private-clusters~~

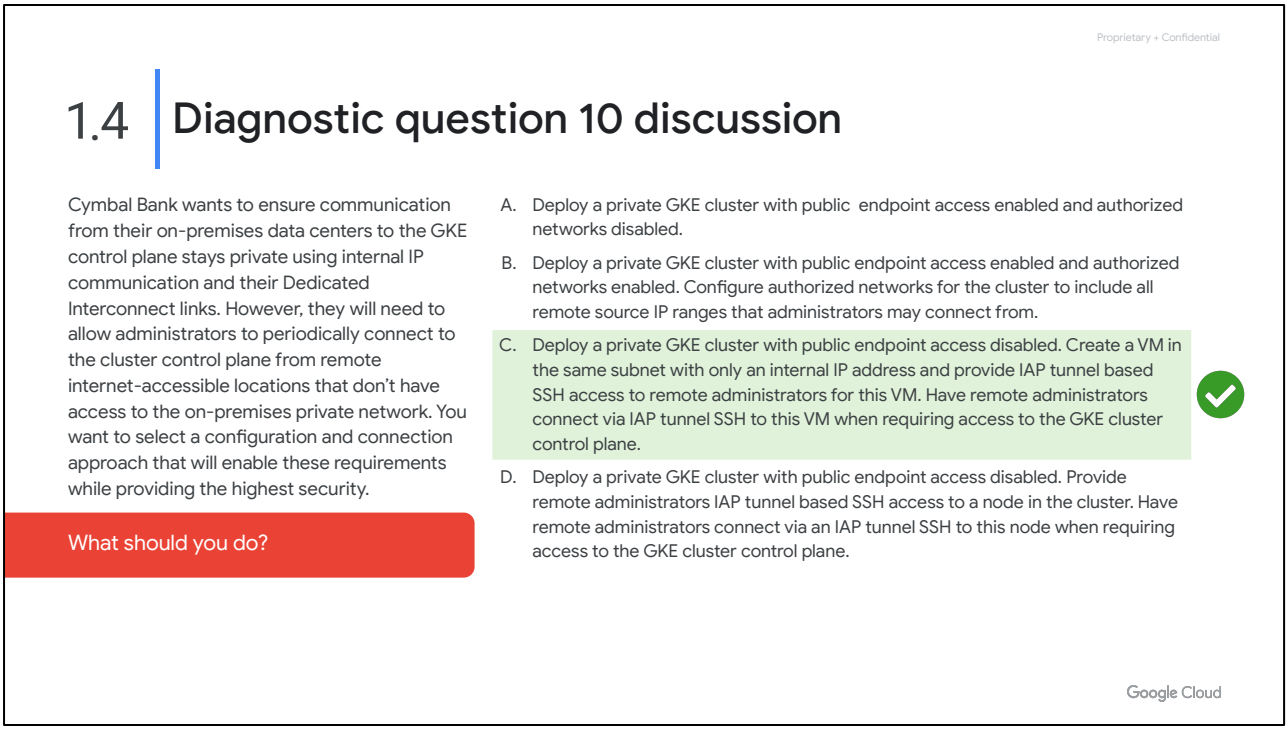
~~**Content mapping:**~~

- ~~Quests~~
  - ~~Partial coverage in Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)~~

Summary:

Creating GKE private clusters improves security. There are 3 high level configurations available for private clusters with varying levels of security and ease of access for each:

- Public endpoint access disabled is the most secure. Connectivity to the control plane is not allowed from any client outside the cluster subnet.
- Public endpoint access enabled, authorized networks enabled is the next most secure and provides connectivity to the control plane from only specified public or private IP ranges.
- Public endpoint access enabled, authorized networks disabled provides the least secure and provides the most permissive connectivity allowing access to the public endpoint from any IP address.



Feedback:
A: Incorrect. This option satisfies the requirements; however, it is the least secure option because it provides access to the control plane from any public IP address.
B: Incorrect. This option satisfies the requirements; however, it is not as secure as options C and D because it provides access to the control plane from a set of public IP addresses/ranges.
*C: Correct! This options satisfies the requirements in the most secure way by not providing any public access to the control plane and no private access to the cluster nodes.
D: Incorrect.Though satisfying requirements, this approach is slightly less secure than approach C as it provides direct node access to the remote administrators when only control plane access is required. (Option C only provides access to the control plane without access to the nodes.)

Where to look:
https://cloud.google.com/kubernetes-engine/docs/concepts/private-cluster-concept
https://cloud.google.com/kubernetes-engine/docs/how-to/private-clusters

Content mapping:
- Quests
  - Partial coverage in Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)

Summary:

Creating GKE private clusters improves security. There are 3 high level configurations available for private clusters with varying levels of security and ease of access for each:

- Public endpoint access disabled is the most secure. Connectivity to the control plane is not allowed from any client outside the cluster subnet.
- Public endpoint access enabled, authorized networks enabled is the next most secure and provides connectivity to the control plane from only specified public or private IP ranges.
- Public endpoint access enabled, authorized networks disabled provides the least secure and provides the most permissive connectivity allowing access to the public endpoint from any IP address.

2.1 Diagnostic question 02 discussion

You are designing a networking scheme for Cymbal Bank with the requirement to use internal IP addresses for communication, with the lowest possible latency. Cymbal Bank has several teams, each with its own projects: P1, P2, and P3. Cymbal Bank would like consolidated network billing, administration, and access control for the cloud environment. VMs in these projects need to connect to VMs in a partner organization in projects P4 and P5.

Select the networking option that best satisfies these requirements.

A. Connect the VMs across the Cymbal projects and partner organization using VPCs in each project (V1, V2, V3, V4, V5) and VPC peering (peering V1 to V2, V2 to V3, V3 to V4, and V4 to V5).

B. Connect the VMs across the Cymbal projects (P1-P3) using a Shared VPC network (Shared VPC host project P6 with VPC V6, and P1-P3 are the service projects) and then peer that Shared VPC network to the partner organization VPCs (V6 peered to V4 and V4 to V5).

C. Connect the VMs across Cymbal and partner organization projects (P1-P5) using a Shared VPC network (Shared VPC host project P6 with VPC V6, and P1-P5 are the service projects).

D. Connect the VMs across the Cymbal projects (P1-P3) using a Shared VPC network (Shared VPC host project P6 with VPC V6, and P1-P3 are the service projects) and then peer that Shared VPC network to the partner organization VPCs (V6 peered to V4 and V6 to V5).

~~**Feedback:**~~

~~A. Incorrect. VPC peering is not transitive ("V1 to V2 and V2 to V3" does not provide connectivity between V1 and V3). Also, this option doesn't satisfy the requirement to have centralized/consolidated network billing, administration, and access control for Cymbal project networking.~~

~~B. Incorrect. VPC peering is not transitive ("V6 peered to V4 and V4 peered to V5" does not provide connectivity between V6 and V5).~~

~~C. Incorrect. Shared VPC cannot work across organizations (only for projects within the same organization).~~

~~*D. Correct! This option satisfies the requirements and provides connectivity between VMs of all the projects.~~

~~**Where to look**:~~

~~https://cloud.google.com/vpc/docs/vpc-peering~~
~~https://cloud.google.com/vpc/docs/using-vpc-peering~~
~~https://cloud.google.com/vpc/docs/shared-vpc~~
~~https://cloud.google.com/vpc/docs/provisioning-shared-vpc~~

~~**Content mapping:**~~

- ~~Instructor-led training~~
  - ~~Partial coverage in Networking in Google Cloud~~
    - ~~M3 Sharing networks across projects~~

- ~~M7 Network Design and Deployment~~
- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~
    - ~~M3 Sharing networks across projects~~
  - ~~Networking in Google Cloud: Hybrid connectivity and network management~~
    - ~~M3 Network Design and Deployment~~
- ~~Skill badges~~
  - ~~Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)~~

~~Summary:~~

~~Shared VPC is a centralized networking model (for billing, administration, and access control) whereas VPC peering is a decentralized approach. VPC peering can work across organizations whereas Shared VPC can only work within organizations. VPC peering does not support transitive peering and requires any two connected VPCs to be directly peered.~~

2.1 Diagnostic question 02 discussion

You are designing a networking scheme for Cymbal Bank with the requirement to use internal IP addresses for communication, with the lowest possible latency. Cymbal Bank has several teams, each with its own projects: P1, P2, and P3. Cymbal Bank would like consolidated network billing, administration, and access control for the cloud environment. VMs in these projects need to connect to VMs in a partner organization in projects P4 and P5.

Select the networking option that best satisfies these requirements.

A. Connect the VMs across the Cymbal projects and partner organization using VPCs in each project (V1, V2, V3, V4, V5) and VPC peering (peering V1 to V2, V2 to V3, V3 to V4, and V4 to V5).

B. Connect the VMs across the Cymbal projects (P1-P3) using a Shared VPC network (Shared VPC host project P6 with VPC V6, and P1-P3 are the service projects) and then peer that Shared VPC network to the partner organization VPCs (V6 peered to V4 and V4 to V5).

C. Connect the VMs across Cymbal and partner organization projects (P1-P5) using a Shared VPC network (Shared VPC host project P6 with VPC V6, and P1-P5 are the service projects).

D. Connect the VMs across the Cymbal projects (P1-P3) using a Shared VPC network (Shared VPC host project P6 with VPC V6, and P1-P3 are the service projects) and then peer that Shared VPC network to the partner organization VPCs (V6 peered to V4 and V6 to V5).

Google Cloud

Feedback:
A. Incorrect. VPC peering is not transitive (“V1 to V2 and V2 to V3” does not provide connectivity between V1 and V3). Also, this option doesn’t satisfy the requirement to have centralized/consolidated network billing, administration, and access control for Cymbal project networking.
B. Incorrect. VPC peering is not transitive (“V6 peered to V4 and V4 peered to V5” does not provide connectivity between V6 and V5).
C. Incorrect. Shared VPC cannot work across organizations (only for projects within the same organization).
*D. Correct! This option satisfies the requirements and provides connectivity between VMs of all the projects.

Where to look:
https://cloud.google.com/vpc/docs/vpc-peering
https://cloud.google.com/vpc/docs/using-vpc-peering
https://cloud.google.com/vpc/docs/shared-vpc
https://cloud.google.com/vpc/docs/provisioning-shared-vpc

Content mapping:
- Instructor-led training
  - Partial coverage in Networking in Google Cloud
    - M3 Sharing networks across projects

        - M7 Network Design and Deployment
- OnDemand
    - Networking in Google Cloud: Defining and implementing networks
        - M3 Sharing networks across projects
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M3 Network Design and Deployment
- Skill badges
    - Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)

Summary:
Shared VPC is a centralized networking model (for billing, administration, and access control) whereas VPC peering is a decentralized approach. VPC peering can work across organizations whereas Shared VPC can only work within organizations. VPC peering does not support transitive peering and requires any two connected VPCs to be directly peered.

2.2 Diagnostic question 04 discussion

You are designing a VPC network with the requirement that all external traffic destined for the internet is passed through a proxy VM. The proxy will have software installed to scan, detect, and drop invalid egress traffic and to help prevent data exfiltration, outbound attacks, or access to blocked websites.

Select the configuration that can most easily accomplish this.

A. Create a custom route to the destination 0.0.0.0/0, and specify the next hop as the proxy VM.

B. Delete the system-generated default route, create a custom route to destination 0.0.0.0/0, and specify the next hop as the proxy VM.

C. Create a custom route to the destination 0.0.0.0/0, specify the next hop as the proxy VM, and configure the scanning VM to enable IP forwarding.

D. Delete the system-generated default route, and then create a custom route to destination 0.0.0.0/0. Specify the next hop as the proxy VM, and configure the proxy VM to enable IP forwarding.

~~**Feedback:**~~

~~A. Incorrect. You can't create a new custom route to the 0.0.0.0/0 destination until you first delete the system-generated default route (which goes to that same destination).~~

~~B. Incorrect. You must enable IP forwarding in a VM to allow it to proxy egress traffic.~~

~~C. Incorrect. You can't create a new custom route to the 0.0.0.0/0 destination until you first delete the system-generated default route (which goes to that same destination).~~

~~*D. Correct! This is the minimal set of steps to configure this routing scenario.~~

~~**Where to look:**~~

~~https://cloud.google.com/vpc/docs/routes~~

~~https://cloud.google.com/vpc/docs/using-routes~~

~~**Content Mapping**:~~

- ~~Instructor-led training~~
  - ~~Partial coverage in Networking in Google Cloud~~
    - ~~M1 Google Cloud VPC networking fundamentals~~
    - ~~M3 Sharing networks across projects~~
    - ~~M5 Hybrid connectivity~~
    - ~~M7 Network Design and Deployment~~
- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~

- ~~M1 Google Cloud VPC networking fundamentals~~
- ~~M3 Sharing networks across projects~~

- ~~Networking in Google Cloud: Hybrid connectivity and network management~~
  - ~~M1 Hybrid connectivity~~
  - ~~M3 Network Design and Deployment~~

- ~~Skill badges~~
  - ~~Build and Secure Networks in Google Cloud Quest (https://www.qwiklabs.com/quests/128?locale=en)~~
  - ~~Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)~~
  - ~~Network Performance and Optimization Quest (https://www.qwiklabs.com/quests/46?locale=en)~~

~~Summary:~~

~~Most standard network routing can be accomplished using the default created routes. In some scenarios, traffic must be forwarded through a NAT or Proxy instance, or routed through an intermediary different than the destination IP address. This can be accomplished by creating custom routes. When creating a custom route, you can specify a destination IP address or range and the next hop instance, load balancer, IP address, internet gateway, or VPN gateway. You can limit which VMs would use a custom route by adding a tag to the custom route that matches a tag on the appropriate VMs.~~

2.2 Diagnostic question 04 discussion

You are designing a VPC network with the requirement that all external traffic destined for the internet is passed through a proxy VM. The proxy will have software installed to scan, detect, and drop invalid egress traffic and to help prevent data exfiltration, outbound attacks, or access to blocked websites.

Select the configuration that can most easily accomplish this.

A. Create a custom route to the destination 0.0.0.0/0, and specify the next hop as the proxy VM.

B. Delete the system-generated default route, create a custom route to destination 0.0.0.0/0, and specify the next hop as the proxy VM.

C. Create a custom route to the destination 0.0.0.0/0, specify the next hop as the proxy VM, and configure the scanning VM to enable IP forwarding.

D. Delete the system-generated default route, and then create a custom route to destination 0.0.0.0/0. Specify the next hop as the proxy VM, and configure the proxy VM to enable IP forwarding.

Feedback:

A. Incorrect. You can't create a new custom route to the 0.0.0.0/0 destination until you first delete the system-generated default route (which goes to that same destination).

B. Incorrect. You must enable IP forwarding in a VM to allow it to proxy egress traffic.

C. Incorrect. You can't create a new custom route to the 0.0.0.0/0 destination until you first delete the system-generated default route (which goes to that same destination).

*D. Correct! This is the minimal set of steps to configure this routing scenario.

Where to look:

https://cloud.google.com/vpc/docs/routes

https://cloud.google.com/vpc/docs/using-routes

Content Mapping:

- Instructor-led training
  - Partial coverage in Networking in Google Cloud
    - M1 Google Cloud VPC networking fundamentals
    - M3 Sharing networks across projects
    - M5 Hybrid connectivity
    - M7 Network Design and Deployment
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks

        - M1 Google Cloud VPC networking fundamentals
        - M3 Sharing networks across projects
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M1 Hybrid connectivity
        - M3 Network Design and Deployment
- Skill badges
    - Build and Secure Networks in Google Cloud Quest (https://www.qwiklabs.com/quests/128?locale=en)
    - Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)
    - Network Performance and Optimization Quest (https://www.qwiklabs.com/quests/46?locale=en)

Summary:

Most standard network routing can be accomplished using the default created routes. In some scenarios, traffic must be forwarded through a NAT or Proxy instance, or routed through an intermediary different than the destination IP address. This can be accomplished by creating custom routes. When creating a custom route, you can specify a destination IP address or range and the next hop instance, load balancer, IP address, internet gateway, or VPN gateway. You can limit which VMs would use a custom route by adding a tag to the custom route that matches a tag on the appropriate VMs.

2.3 Diagnostic question 05 discussion

Cymbal Bank has an existing subnet that you want to use for a new VPC-native GKE cluster. The subnet primary IP address range is 10.128.128.0/20. Currently the 1000 other VMs using that subnet have taken 1000 of the available IP addresses. The new GKE cluster should support 200,000 pods and 30,000 services.

Select the minimal set of configuration steps and the smallest possible IP ranges to enable this.

A. Expand the subnet primary IP address range to 10.128.0.0/16; create a secondary range in the subnet of size /14 for pods and another of size /17 for services; and create the GKE VPC-native cluster in the subnet using these secondary ranges.

B. Create a secondary range in the subnet of size /13 for pods and another of size /16 for services, and create the GKE VPC-native cluster in the subnet using these secondary ranges.

C. Create a GKE VPC-native cluster in the subnet, specifying the size of the pod range as /14 and the size of the services range as /17.

D. Create a GKE VPC-native cluster in the subnet, specifying the size of the pod range as /13 and the size of the services range as /17.

~~**Feedback:**~~

~~A. Incorrect. Expansion of the subnet is unnecessary because there are enough IP addresses for sufficient nodes to support 200,000 pods. The secondary ranges can be specified when creating the cluster. Also, the pod range is not large enough for 200,000 pods (requires /13 size) because each pod requires one IP address.~~

~~B. Incorrect. Secondary ranges can be created automatically when the GKE cluster is created and don't require manual creation beforehand. Also, the service range is larger than necessary to support 30,000 service (/17 would suffice) because each pod requires one IP address.~~

~~C. Incorrect. The pod range is not large enough for 200,000 pods (requires /13 size) because each pod requires one IP address.~~

~~*D. Correct! This is the minimal configuration with the smallest possible ranges because each pod and service requires one IP address..~~

~~**Where to look**:~~

~~https://cloud.google.com/kubernetes-engine/docs/concepts/types-of-clusters~~
~~https://cloud.google.com/kubernetes-engine/docs/concepts/alias-ips~~
~~https://cloud.google.com/kubernetes-engine/docs/how-to/alias-ips~~
~~https://cloud.google.com/kubernetes-engine/docs/how-to/flexible-pod-cidr~~

~~**Content mapping**: Highly recommend reviewing documentation~~

- ~~Skill badges~~
    - ~~Minor coverage in Security & Identity Fundamentals Quest~~

~~Summary:~~

~~For VPC-native clusters, secondary subnet ranges are used for pod and service IP ranges. You can either pre-create the secondary ranges or simply specify them when creating the cluster. Each node will support up to 110 pods. The primary IP range needs to be large enough that when the number of nodes is multiplied by 110, the supported number of pods will be sufficient. The size of the pod range also needs to be large enough to provide one IP address per pod for the maximum number of pods. The service range should be large enough to provide one IP address for the maximum number of services.~~

2.3 Diagnostic question 05 discussion

Cymbal Bank has an existing subnet that you want to use for a new VPC-native GKE cluster. The subnet primary IP address range is 10.128.128.0/20. Currently the 1000 other VMs using that subnet have taken 1000 of the available IP addresses. The new GKE cluster should support 200,000 pods and 30,000 services.

Select the minimal set of configuration steps and the smallest possible IP ranges to enable this.

A. Expand the subnet primary IP address range to 10.128.0.0/16; create a secondary range in the subnet of size /14 for pods and another of size /17 for services; and create the GKE VPC-native cluster in the subnet using these secondary ranges.

B. Create a secondary range in the subnet of size /13 for pods and another of size /16 for services, and create the GKE VPC-native cluster in the subnet using these secondary ranges.

C. Create a GKE VPC-native cluster in the subnet, specifying the size of the pod range as /14 and the size of the services range as /17.

D. Create a GKE VPC-native cluster in the subnet, specifying the size of the pod range as /13 and the size of the services range as /17.

Feedback:
A. Incorrect. Expansion of the subnet is unnecessary because there are enough IP addresses for sufficient nodes to support 200,000 pods. The secondary ranges can be specified when creating the cluster. Also, the pod range is not large enough for 200,000 pods (requires /13 size) because each pod requires one IP address.
B. Incorrect. Secondary ranges can be created automatically when the GKE cluster is created and don't require manual creation beforehand. Also, the service range is larger than necessary to support 30,000 service (/17 would suffice) because each pod requires one IP address.
C. Incorrect. The pod range is not large enough for 200,000 pods (requires /13 size) because each pod requires one IP address.
*D. Correct! This is the minimal configuration with the smallest possible ranges because each pod and service requires one IP address..

Where to look:
https://cloud.google.com/kubernetes-engine/docs/concepts/types-of-clusters
https://cloud.google.com/kubernetes-engine/docs/concepts/alias-ips
https://cloud.google.com/kubernetes-engine/docs/how-to/alias-ips
https://cloud.google.com/kubernetes-engine/docs/how-to/flexible-pod-cidr

Content mapping: Highly recommend reviewing documentation

- Skill badges
    - Minor coverage in Security & Identity Fundamentals Quest

Summary:
For VPC-native clusters, secondary subnet ranges are used for pod and service IP ranges. You can either pre-create the secondary ranges or simply specify them when creating the cluster. Each node will support up to 110 pods. The primary IP range needs to be large enough that when the number of nodes is multiplied by 110, the supported number of pods will be sufficient. The size of the pod range also needs to be large enough to provide one IP address per pod for the maximum number of pods. The service range should be large enough to provide one IP address for the maximum number of services.

2.4 Diagnostic question 07 Discussion

You are configuring firewall rules for securing a set of microservices (MS1, MS2, MS3) running in separate managed instance groups (MIGs) of VMs in a single subnet of a VPC network. The primary range of the VPC network is 10.128.128.0/20. MS1 will send requests to MS2 on TCP port 8443; MS2 will send requests to MS3 on TCP port 8663; and MS3 will send requests to MS1 on TCP port 8883. There will be no other communication to or between these microservices.

Select a simple and secure firewall configuration to support this traffic requirement.

A. Create service accounts (S1, S2, S3) for the microservices, and assign those service accounts to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source S1 to target S2; for TCP 8663 from source S2 to target S3, and for TCP 8883 from source S3 to target S1.

B. Create network tags (T1, T2. T3) for the microservices, and assign those network tags to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source T1 to target T2; for TCP 8663 from source T2 to target T3; and for TCP 8883 from source T3 to target T4.

C. Create service accounts (S1, S2, S3) for the microservices, and assign those service accounts to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source 10.128.128.0/20 to target S2; for TCP 8663 from source 10.128.128.0/20 to target S3; for TCP and 8883 from source 10.128.128.0/20 to target S1.

D. Create network tags (T1, T2, T3) for the microservices, and assign those network tags to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source 10.128.128.0/20 to target T2; for TCP 8663 from source 10.128.128.0/20 to target T3; and for TCP 8883 from source 10.128.128.0/20 to target T1.

~~**Feedback:**~~

~~*A. Correct! This option is as simple as the others but provides better security: service accounts have tighter access control than network tags.~~

~~B. Incorrect. This option is slightly less secure than option A: service accounts have tighter access control than network tags.~~

~~C. Incorrect. This option is significantly less secure than option A. It would allow requests into the microservices from any other workloads deployed in the same subnet, using the same ports as the intended microservices.~~

~~D. Incorrect. This is significantly less secure than option A because it would allow requests into the microservices from any other workloads deployed in the same subnet using the same ports as the intended microservices.~~

~~**Where to look:**~~

~~https://cloud.google.com/vpc/docs/firewalls~~

~~https://cloud.google.com/vpc/docs/using-firewalls~~

~~**Content mapping:**~~

- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M1 Google Cloud VPC networking fundamentals~~
    - ~~M2 Controlling access to VPC networks~~
    - ~~M3 Sharing networks across projects~~
    - ~~M4 Load balancing~~

- ~~M7 Network Design and Deployment~~
- ~~M8 Network Monitoring and Troubleshooting~~

- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~
    - ~~M1 Google Cloud VPC networking fundamentals~~
    - ~~M2 Controlling access to VPC networks~~
    - ~~M3 Sharing networks across projects~~
    - ~~M4 Load balancing~~
  - ~~Networking in Google Cloud: Hybrid connectivity and network management~~
    - ~~M3 Network Design and Deployment~~
    - ~~M4 Network Monitoring and Troubleshooting~~
- ~~Skill badges~~
  - ~~Build and Secure Networks in Google Cloud Quest (https://www.qwiklabs.com/quests/128?locale=en)~~
  - ~~Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)~~
  - ~~Network Performance and Optimization Quest (https://www.qwiklabs.com/quests/46?locale=en)~~

~~Summary:~~

~~Firewall rules can be assigned to operate on all instances in a VPC. You can also assign them to specific VMs using source or destination IP addresses or IP ranges. You can also assign firewall rules using source and target service accounts or tags. In general using source and target service accounts is the recommended approach and provides the best security.~~

2.4 Diagnostic question 07 Discussion

You are configuring firewall rules for securing a set of microservices (MS1, MS2, MS3) running in separate managed instance groups (MIGs) of VMs in a single subnet of a VPC network. The primary range of the VPC network is 10.128.128.0/20. MS1 will send requests to MS2 on TCP port 8443; MS2 will send requests to MS3 on TCP port 8663; and MS3 will send requests to MS1 on TCP port 8883. There will be no other communication to or between these microservices.

Select a simple and secure firewall configuration to support this traffic requirement.

A. Create service accounts (S1, S2, S3) for the microservices, and assign those service accounts to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source S1 to target S2; for TCP 8663 from source S2 to target S3, and for TCP 8883 from source S3 to target S1.

B. Create network tags (T1, T2. T3) for the microservices, and assign those network tags to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source T1 to target T2; for TCP 8663 from source T2 to target T3; and for TCP 8883 from source T3 to target T4.

C. Create service accounts (S1, S2, S3) for the microservices, and assign those service accounts to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source 10.128.128.0/20 to target S2; for TCP 8663 from source 10.128.128.0/20 to target S3; for TCP and 8883 from source 10.128.128.0/20 to target S1.

D. Create network tags (T1, T2, T3) for the microservices, and assign those network tags to the instance template for the MIG used by each microservice. Create three ingress allow firewall rules: for TCP 8443 from source 10.128.128.0/20 to target T2; for TCP 8663 from source 10.128.128.0/20 to target T3; and for TCP 8883 from source 10.128.128.0/20 to target T1.

Google Cloud

Feedback:
*A. Correct! This option is as simple as the others but provides better security: service accounts have tighter access control than network tags.
B. Incorrect. This option is slightly less secure than option A: service accounts have tighter access control than network tags.
C. Incorrect. This option is significantly less secure than option A. It would allow requests into the microservices from any other workloads deployed in the same subnet, using the same ports as the intended microservices.
D. Incorrect. This is significantly less secure than option A because it would allow requests into the microservices from any other workloads deployed in the same subnet using the same ports as the intended microservices.

Where to look:
https://cloud.google.com/vpc/docs/firewalls
https://cloud.google.com/vpc/docs/using-firewalls

Content mapping:
- Instructor-led training
  - Networking in Google Cloud
    - M1 Google Cloud VPC networking fundamentals
    - M2 Controlling access to VPC networks
    - M3 Sharing networks across projects
    - M4 Load balancing

        - M7 Network Design and Deployment
        - M8 Network Monitoring and Troubleshooting
- OnDemand
    - Networking in Google Cloud: Defining and implementing networks
        - M1 Google Cloud VPC networking fundamentals
        - M2 Controlling access to VPC networks
        - M3 Sharing networks across projects
        - M4 Load balancing
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M3 Network Design and Deployment
        - M4 Network Monitoring and Troubleshooting
- Skill badges
    - Build and Secure Networks in Google Cloud Quest (https://www.qwiklabs.com/quests/128?locale=en)
    - Security & Identity Fundamentals Quest (https://www.qwiklabs.com/quests/40?locale=en)
    - Network Performance and Optimization Quest (https://www.qwiklabs.com/quests/46?locale=en)

Summary:
Firewall rules can be assigned to operate on all instances in a VPC. You can also assign them to specific VMs using source or destination IP addresses or IP ranges. You can also assign firewall rules using source and target service accounts or tags. In general using source and target service accounts is the recommended approach and provides the best security.

2.5 Diagnostic question 09 discussion

Cymbal Bank requires that access to the Cloud Storage buckets in a project is restricted to ensure that the only way the buckets or objects within can be accessed is via users (who also have the necessary IAM role or ACL access to the bucket or object) first connecting to a VM running in a VPC in the project via SSH. You also want to ensure that users and service accounts are blocked from access to other Google Cloud APIs in the same project from VMs in the project VPCs, regardless of whether they have access via Identity and Access Management (IAM) roles.

Select the approach that can accomplish this with minimal configuration effort and complexity.

A. Create a VPC service controls service perimeter that includes the project and restricts access to Cloud Storage APIs, and enable VPC accessible services configuring Cloud Storage APIs as accessible.

B. Create a VPC service controls service perimeter that includes the project and restricts access to Cloud Storage APIs.

C. Create a VPC service controls service perimeter that includes an ingress rule for all users ingressFrom.identityType: ANY_USER_ACCOUNT; ingressFrom.sources.resource set to the project full path; ingressTo.operations.serviceName set to storage.googleapis.com; ingressTo.operations.methodSelectors.permission set to google.storage.buckets.get; and ingressTo.resources set to \"*\"

D. Update the IAM role bindings for all users with access to the buckets to add an IAM condition of the access level attribute type.

~~Feedback:~~

~~*A. Correct! This is the minimal configuration that can satisfy the requirements~~

~~B. Incorrect. This would not be sufficient to ensure that access to other Google Cloud APIs is blocked~~

~~C. Incorrect. This would require more configuration, would not provide access for all operations to Cloud Storage bucket and objects, and would not ensure that access was restricted for service accounts or other Google Cloud APIs~~

~~D. Incorrect. This would require much more configuration and would not ensure that access to other Google Cloud APIs was blocked~~

~~Where to look:~~

~~https://cloud.google.com/vpc-service-controls/docs/overview~~
~~https://cloud.google.com/vpc-service-controls/docs/service-perimeters~~
~~https://cloud.google.com/vpc-service-controls/docs/ingress-egress-rules~~
~~https://cloud.google.com/vpc-service-controls/docs/share-across-perimeters~~
~~https://cloud.google.com/vpc-service-controls/docs/create-service-perimeters~~
~~https://cloud.google.com/vpc-service-controls/docs/dry-run-mode~~

~~Content Mapping~~~~: Not in current course, please refer to documentation~~

~~Summary:~~

~~VPC Service controls provide another option for access control and restriction along with Cloud IAM and Firewall rules. They allow for the creation of service perimeters around one or more projects that can allow or restrict access to Google Cloud APIs in those projects from inside (VPCs within the project) and outside those projects (VPCs in other projects or the internet). Service perimeter access control can be tailored with access levels, ingress and egress rules, and service perimeter bridges allowing for flexibility in how access is enabled or disabled.~~

2.5 Diagnostic question 09 discussion

Cymbal Bank requires that access to the Cloud Storage buckets in a project is restricted to ensure that the only way the buckets or objects within can be accessed is via users (who also have the necessary IAM role or ACL access to the bucket or object) first connecting to a VM running in a VPC in the project via SSH. You also want to ensure that users and service accounts are blocked from access to other Google Cloud APIs in the same project from VMs in the project VPCs, regardless of whether they have access via Identity and Access Management (IAM) roles.

Select the approach that can accomplish this with minimal configuration effort and complexity.

A. Create a VPC service controls service perimeter that includes the project and restricts access to Cloud Storage APIs, and enable VPC accessible services configuring Cloud Storage APIs as accessible.

B. Create a VPC service controls service perimeter that includes the project and restricts access to Cloud Storage APIs.

C. Create a VPC service controls service perimeter that includes an ingress rule for all users ingressFrom.identityType: ANY_USER_ACCOUNT; ingressFrom.sources.resource set to the project full path; ingressTo.operations.serviceName set to storage.googleapis.com; ingressTo.operations.methodSelectors.permission set to google.storage.buckets.get; and ingressTo.resources set to \"*\"

D. Update the IAM role bindings for all users with access to the buckets to add an IAM condition of the access level attribute type.

Feedback:
*A. Correct! This is the minimal configuration that can satisfy the requirements
B. Incorrect. This would not be sufficient to ensure that access to other Google Cloud APIs is blocked
C. Incorrect. This would require more configuration, would not provide access for all operations to Cloud Storage bucket and objects, and would not ensure that access was restricted for service accounts or other Google Cloud APIs
D. Incorrect. This would require much more configuration and would not ensure that access to other Google Cloud APIs was blocked

Where to look:
https://cloud.google.com/vpc-service-controls/docs/overview
https://cloud.google.com/vpc-service-controls/docs/service-perimeters
https://cloud.google.com/vpc-service-controls/docs/ingress-egress-rules
https://cloud.google.com/vpc-service-controls/docs/share-across-perimeters
https://cloud.google.com/vpc-service-controls/docs/create-service-perimeters
https://cloud.google.com/vpc-service-controls/docs/dry-run-mode

Content Mapping: Not in current course, please refer to documentation

Summary:

VPC Service controls provide another option for access control and restriction along with Cloud IAM and Firewall rules. They allow for the creation of service perimeters around one or more projects that can allow or restrict access to Google Cloud APIs in those projects from inside (VPCs within the project) and outside those projects (VPCs in other projects or the internet). Service perimeter access control can be tailored with access levels, ingress and egress rules, and service perimeter bridges allowing for flexibility in how access is enabled or disabled.

2.5 Diagnostic question 10 discussion

Cymbal Bank has a set of VPC service control service perimeters around several projects with BigQuery datasets, and each project is in a separate service perimeter. You want to restrict access to these projects' BigQuery datasets to VMs in the VPCs of one of these projects (project P1,) and a small set of users should have external access from a combination of a specific IP range, geo-location, and device type.

Select the configuration that satisfies these requirements with minimal configuration.

A. Create a service perimeter bridge connecting the service perimeters of all the projects.

B. Create a service perimeter bridge connecting the service perimeters of all the projects, and update all the service perimeters to add an access level providing the external access for the specified users.

C. Update the service perimeter configurations for all the projects to add an ingress rule with an access level to provide the external access for the specified users.

D. Update the service perimeter configurations for all the projects to add an ingress rule to provide external access for the specified users and add another ingress rule to provide access from the VPCs of the specified project P1.

~~**Feedback:**~~

~~A. Incorrect. This would not satisfy the requirement or provide access to external users, and it would not limit access to VMs in VPCs of only project P1.~~

~~B. Incorrect. This would not satisfy the requirement of providing access only to VMs in VPCs of project P1.~~

~~C. Incorrect. This would not satisfy the requirement of providing access only to VMs in VPCs of project P1.~~

~~*D. Correct! This is the correct minimal configuration to satisfy the requirements.~~

~~**Where to look:**~~

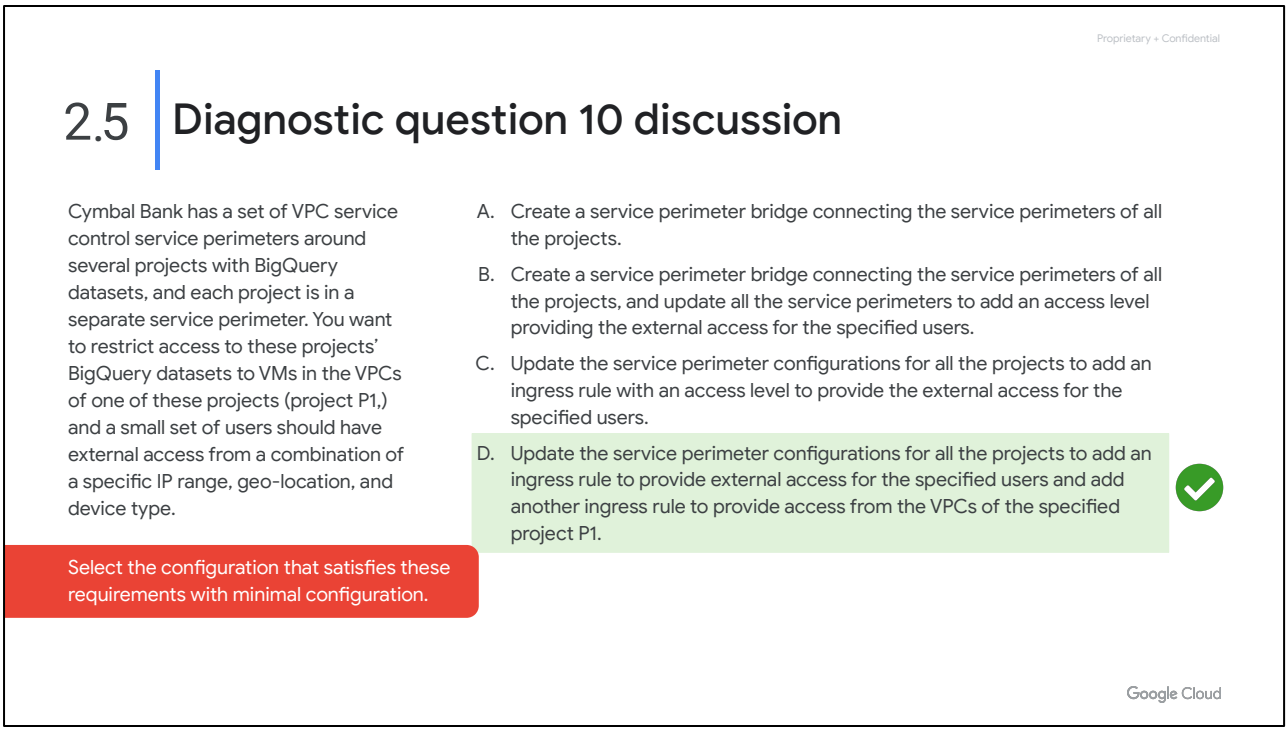
~~https://cloud.google.com/vpc-service-controls/docs/ingress-egress-rules~~
~~https://cloud.google.com/vpc-service-controls/docs/share-across-perimeters~~
~~https://cloud.google.com/vpc-service-controls/docs/create-perimeter-bridges~~
~~https://cloud.google.com/vpc-service-controls/docs/secure-data-exchange~~
~~https://cloud.google.com/vpc-service-controls/docs/context-aware-access~~
~~https://cloud.google.com/access-context-manager/docs/access-level-attributes~~
~~https://cloud.google.com/access-context-manager/docs/custom-access-level-spec~~

~~**Content mapping**: Not in current course, please refer to documentation~~

~~**Summary:**~~

~~Access Levels, Service perimeter bridges, and Ingress and Egress rules can all be~~

~~used to adjust the access control provided by VPC service control service perimeters. Access levels allow external access across service perimeters based on the attributes of the request (such as IP address, identity, geolocation, and device type). Service perimeter bridges provide bi-directional access to all the projects of both service perimeters on either side of the service perimeter bridge. Ingress and Egress rules allow for more granular control of access across service perimeter boundaries as compared to service perimeter bridges. Ingress and Egress rules can also include and adjust the access provided by access levels.~~



Feedback:
A. Incorrect. This would not satisfy the requirement or provide access to external users, and it would not limit access to VMs in VPCs of only project P1.
B. Incorrect. This would not satisfy the requirement of providing access only to VMs in VPCs of project P1.
C. Incorrect. This would not satisfy the requirement of providing access only to VMs in VPCs of project P1.
*D. Correct! This is the correct minimal configuration to satisfy the requirements.

Where to look:
https://cloud.google.com/vpc-service-controls/docs/ingress-egress-rules
https://cloud.google.com/vpc-service-controls/docs/share-across-perimeters
https://cloud.google.com/vpc-service-controls/docs/create-perimeter-bridges
https://cloud.google.com/vpc-service-controls/docs/secure-data-exchange
https://cloud.google.com/vpc-service-controls/docs/context-aware-access
https://cloud.google.com/access-context-manager/docs/access-level-attributes
https://cloud.google.com/access-context-manager/docs/custom-access-level-spec

Content mapping: Not in current course, please refer to documentation

Summary:

Access Levels, Service perimeter bridges, and Ingress and Egress rules can all be

used to adjust the access control provided by VPC service control service perimeters. Access levels allow external access across service perimeters based on the attributes of the request (such as IP address, identity, geolocation, and device type). Service perimeter bridges provide bi-directional access to all the projects of both service perimeters on either side of the service perimeter bridge. Ingress and Egress rules allow for more granular control of access across service perimeter boundaries as compared to service perimeter bridges. Ingress and Egress rules can also include and adjust the access provided by access levels.

3.1 Diagnostic question 01 discussion

Cymbal Bank wants a web application to have global anycast load balancing across multiple regions. The web application will serve static asset files and will also use REST APIs that serve dynamic responses. The load balancer should support HTTP and HTTPS requests and redirect HTTP to HTTPS. The load balancer should also serve all the requests from the same domain name, with different paths indicating static versus dynamic resources.

Select the load balancer configuration that would most effectively enable this scenario.

A. A global external HTTP(S) load balancer with one global forwarding rule, forwarding to one target proxy with one URL map connected to two backend services

B. A global external HTTP(S) load balancer with two global forwarding rules, forwarding to two target proxies: one with URL map and no backend service, and the other with URL map and two backend services

C. Two global external HTTP(S) load balancers, each with one global forwarding rule forwarding to one target proxy with one URL map connected to one backend service

D. A global external HTTP(S) load balancer with two global forwarding rules forwarding to two target proxies: one with URL map and no backend service, and the other with URL map, one backend service, and one backend bucket

~~**Feedback:**~~

~~A. Incorrect. A single target proxy could not listen for both HTTP and HTTPS traffic.~~

~~B. Incorrect. Backend bucket is a more effective way (simpler, lower cost, higher availability) of serving static resources than a backend service, which must be connected to compute resources such as managed instance groups or network endpoint groups.~~

~~C. Incorrect. This solution would use two IP addresses, which would require either two domain names or DNS based load-balancing. Also, a backend bucket is a more effective way (simpler, lower cost, higher availability) of serving static resources than a backend service, which must be connected to compute resources like managed instance groups or network endpoint groups~~

~~*D. Correct! This is the most effective solution. A single global external HTTP(S) load balancer (with a single IP address and domain name) is connected via two global forwarding rules to two target proxies. The first global forwarding rule forwarding to the first target proxy (with URL map issuing a redirect, and no connected backend service) serves HTTP requests and simply redirects to HTTPS. The second global forwarding rule forwards to the second target proxy. This serves HTTPS and uses the URL map to forward static resource requests to either a backend bucket or a backend service, based on the path of the request. The backend bucket serves the web application files and connects to a Google Cloud Storage bucket. The backend service serves dynamic requests, and is connected to a managed instance group or~~

network endpoint group to serve those dynamic requests via a REST API.

Where to look:
https://cloud.google.com/load-balancing/docs/load-balancing-overview
https://cloud.google.com/load-balancing/docs/choosing-load-balancer
https://cloud.google.com/load-balancing/docs/features
https://cloud.google.com/load-balancing/docs/https
https://cloud.google.com/load-balancing/docs/l7-internal
https://cloud.google.com/load-balancing/docs/network
https://cloud.google.com/load-balancing/docs/internal
https://cloud.google.com/load-balancing/docs/ssl
https://cloud.google.com/load-balancing/docs/tcp
https://cloud.google.com/load-balancing/docs/backend-service
https://cloud.google.com/load-balancing/docs/forwarding-rule-concepts

Content mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M4 Load balancing
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M4 Load balancing
- Skill badges
  - Create and Manage Cloud Resources

Summary:
There are many types of load balancer available in Google Cloud. Some operate at layer 4 (L4) and others at layer 7 (L7). For HTTP and HTTPS traffic the L7 internal or external regional or global HTTP(S) load balancer is typically the best choice.

When supporting both HTTP and HTTPS for the same domains, the recommended approach is to use 2 separate target proxies. Each target proxy requires a forwarding rule. One of the target proxies is used to redirect HTTP requests to HTTPS and requires no backend bucket or service. Each target proxy would also have a URL map that it uses to either redirect traffic, or direct traffic to backend services or backend buckets based on the domain or path of the request. This is referred to as content-based load balancing.

For global load balancers, each backend bucket or backend service can operate over many regions and will intelligently route to the nearest region to where the user traffic originates. Backend buckets are connected to Cloud Storage buckets. Backend

~~buckets are the most effective way to serve static resources because those resources can be served directly from the GCS bucket without requiring any compute resources. Static resources can also easily be populated into Cloud CDN to be served with low latency worldwide.~~

3.1 Diagnostic question 01 discussion

Cymbal Bank wants a web application to have global anycast load balancing across multiple regions. The web application will serve static asset files and will also use REST APIs that serve dynamic responses. The load balancer should support HTTP and HTTPS requests and redirect HTTP to HTTPS. The load balancer should also serve all the requests from the same domain name, with different paths indicating static versus dynamic resources.

Select the load balancer configuration that would most effectively enable this scenario.

A. A global external HTTP(S) load balancer with one global forwarding rule, forwarding to one target proxy with one URL map connected to two backend services

B. A global external HTTP(S) load balancer with two global forwarding rules, forwarding to two target proxies: one with URL map and no backend service, and the other with URL map and two backend services

C. Two global external HTTP(S) load balancers, each with one global forwarding rule forwarding to one target proxy with one URL map connected to one backend service

D. A global external HTTP(S) load balancer with two global forwarding rules forwarding to two target proxies: one with URL map and no backend service, and the other with URL map, one backend service, and one backend bucket

Feedback:

A. Incorrect. A single target proxy could not listen for both HTTP and HTTPS traffic.

B. Incorrect. Backend bucket is a more effective way (simpler, lower cost, higher availability) of serving static resources than a backend service, which must be connected to compute resources such as managed instance groups or network endpoint groups.

C. Incorrect. This solution would use two IP addresses, which would require either two domain names or DNS based load-balancing. Also, a backend bucket is a more effective way (simpler, lower cost, higher availability) of serving static resources than a backend service, which must be connected to compute resources like managed instance groups or network endpoint groups

*D. Correct! This is the most effective solution. A single global external HTTP(S) load balancer (with a single IP address and domain name) is connected via two global forwarding rules to two target proxies. The first global forwarding rule forwarding to the first target proxy (with URL map issuing a redirect, and no connected backend service) serves HTTP requests and simply redirects to HTTPS. The second global forwarding rule forwards to the second target proxy. This serves HTTPS and uses the URL map to forward static resource requests to either a backend bucket or a backend service, based on the path of the request. The backend bucket serves the web-application files and connects to a Google Cloud Storage bucket. The backend service serves dynamic requests, and is connected to a managed instance group or

network endpoint group to serve those dynamic requests via a REST API.

Where to look:
https://cloud.google.com/load-balancing/docs/load-balancing-overview
https://cloud.google.com/load-balancing/docs/choosing-load-balancer
https://cloud.google.com/load-balancing/docs/features
https://cloud.google.com/load-balancing/docs/https
https://cloud.google.com/load-balancing/docs/l7-internal
https://cloud.google.com/load-balancing/docs/network
https://cloud.google.com/load-balancing/docs/internal
https://cloud.google.com/load-balancing/docs/ssl
https://cloud.google.com/load-balancing/docs/tcp
https://cloud.google.com/load-balancing/docs/backend-service
https://cloud.google.com/load-balancing/docs/forwarding-rule-concepts

Content mapping:

- Instructor-led training
    - Networking in Google Cloud
        - M4 Load balancing
- OnDemand
    - Networking in Google Cloud: Defining and implementing networks
        - M4 Load balancing
- Skill badges
    - Create and Manage Cloud Resources

Summary:
There are many types of load balancer available in Google Cloud. Some operate at layer 4 (L4) and others at layer 7 (L7). For HTTP and HTTPS traffic the L7 internal or external regional or global HTTP(S) load balancer is typically the best choice.

When supporting both HTTP and HTTPS for the same domains, the recommended approach is to use 2 separate target proxies. Each target proxy requires a forwarding rule. One of the target proxies is used to redirect HTTP requests to HTTPS and requires no backend bucket or service. Each target proxy would also have a URL map that it uses to either redirect traffic, or direct traffic to backend services or backend buckets based on the domain or path of the request. This is referred to as content-based load balancing.

For global load balancers, each backend bucket or backend service can operate over many regions and will intelligently route to the nearest region to where the user traffic originates. Backend buckets are connected to Cloud Storage buckets. Backend

buckets are the most effective way to serve static resources because those resources can be served directly from the GCS bucket without requiring any compute resources. Static resources can also easily be populated into Cloud CDN to be served with low latency worldwide.

3.1 Diagnostic question 02 discussion

You are designing a load balanced autoscaling frontend for Cymbal Bank. It is intended to be deployed into Google Kubernetes Engine (GKE). You want to use container-native load balancing and autoscale based on the amount of traffic to the service.

Select the type of backend and autoscaling that would accomplish this.

A. A managed instance group of GKE nodes that autoscale using cluster autoscaling

B. A zonal network endpoint group of Kubernetes pods that autoscale using a Horizontal Pod Autoscaler

C. A managed instance group of GKE nodes that contain pods that autoscale using a Horizontal Pod Autoscaler

D. A serverless network endpoint group of GKE pods that autoscale using a Horizontal Pod Autoscaler

~~**Feedback:**~~

~~A. Incorrect. Managed instance groups of VMs are not used for container-native load balancing. They are used for non container-native load balancing in GKE. Also, cluster autoscaling is based on the resource demands of all workloads in the cluster. Cluster autoscaling is not tied to specific workload, nor does it directly affect capacity or number of pod replicas for a particular workload. It has an indirect impact by allowing more or less pods to be scheduled and run.~~

~~*B. Correct! Zonal network endpoint groups connect the load balancer directly to the pods for container-native load balancing in GKE. The ReplicaSet of pod replicas can autoscale based on the amount of traffic arriving at the associated Service via the HorizontalPodAutoscaler.~~

~~C. Incorrect. Managed instance groups of the Kubernetes Engine nodes would be used for non container-native load balancing in GKE.~~

~~D. Incorrect. Serverless network endpoint groups connect load balancers to App Engine, Cloud Run, Cloud Functions, or API Gateway services.~~

~~**Where to look**:~~

~~https://cloud.google.com/compute/docs/instance-groups~~
~~https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-managed-instances~~
~~https://cloud.google.com/load-balancing/docs/negs~~
~~https://cloud.google.com/load-balancing/docs/negs/zonal-neg-concepts~~

https://cloud.google.com/load-balancing/docs/negs/internet-neg-concepts
https://cloud.google.com/load-balancing/docs/negs/serverless-neg-concepts

Content mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M4 Load balancing
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M4 Load balancing
- Skill badges
  - Create and Manage Cloud Resources Quest
  - Build and Secure Networks in Google Cloud Quest

Summary:

Managed instance groups and network endpoint groups can both autoscale. They create or delete replicas to serve traffic automatically, based on resource target values that are measured by metrics collected by Google Cloud Monitoring. Managed instance groups are always groups of identical VMs. Network endpoint groups can be more general, such as FQDN/port or IP/port combinations, or managed services. In GKE you can use either managed instance groups or network endpoint groups, but container-native load balancing only uses network endpoint groups.

In GKE workloads, autoscaling is typically accomplished using a HorizontalPodAutoscaler, though VerticalPodAutoscaler and MultidimPodAutoscaler are possible alternatives. GKE cluster autoscaling occurs based on the resource demands and scheduling of pods across all workloads using a given node pool.

3.1 Diagnostic question 02 discussion

You are designing a load balanced autoscaling frontend for Cymbal Bank. It is intended to be deployed into Google Kubernetes Engine (GKE). You want to use container-native load balancing and autoscale based on the amount of traffic to the service.

Select the type of backend and autoscaling that would accomplish this.

A. A managed instance group of GKE nodes that autoscale using cluster autoscaling

B. A zonal network endpoint group of Kubernetes pods that autoscale using a Horizontal Pod Autoscaler

C. A managed instance group of GKE nodes that contain pods that autoscale using a Horizontal Pod Autoscaler

D. A serverless network endpoint group of GKE pods that autoscale using a Horizontal Pod Autoscaler

Feedback:
A. Incorrect. Managed instance groups of VMs are not used for container-native load balancing. They are used for non container-native load balancing in GKE. Also, cluster autoscaling is based on the resource demands of all workloads in the cluster. Cluster autoscaling is not tied to specific workload, nor does it directly affect capacity or number of pod replicas for a particular workload. It has an indirect impact by allowing more or less pods to be scheduled and run.
*B. Correct! Zonal network endpoint groups connect the load balancer directly to the pods for container-native load balancing in GKE. The ReplicaSet of pod replicas can autoscale based on the amount of traffic arriving at the associated Service via the HorizontalPodAutoscaler.
C. Incorrect. Managed instance groups of the Kubernetes Engine nodes would be used for non container-native load balancing in GKE.
D. Incorrect. Serverless network endpoint groups connect load balancers to App Engine, Cloud Run, Cloud Functions, or API Gateway services.

Where to look:
https://cloud.google.com/compute/docs/instance-groups
https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-managed-instances
https://cloud.google.com/load-balancing/docs/negs
https://cloud.google.com/load-balancing/docs/negs/zonal-neg-concepts

https://cloud.google.com/load-balancing/docs/negs/internet-neg-concepts
https://cloud.google.com/load-balancing/docs/negs/serverless-neg-concepts

Content mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M4 Load balancing
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M4 Load balancing
- Skill badges
  - Create and Manage Cloud Resources Quest
  - Build and Secure Networks in Google Cloud Quest

Summary:

Managed instance groups and network endpoint groups can both autoscale. They create or delete replicas to serve traffic automatically, based on resource target values that are measured by metrics collected by Google Cloud Monitoring. Managed instance groups are always groups of identical VMs. Network endpoint groups can be more general, such as FQDN/port or IP/port combinations, or managed services. In GKE you can use either managed instance groups or network endpoint groups, but container-native load balancing only uses network endpoint groups.

In GKE workloads, autoscaling is typically accomplished using a HorizontalPodAutoscaler, though VerticalPodAutoscaler and MultidimPodAutoscaler are possible alternatives. GKE cluster autoscaling occurs based on the resource demands and scheduling of pods across all workloads using a given node pool.

3.2 Diagnostic question 03 discussion

Cymbal Bank wants to protect their services, which are deployed behind an HTTP(S) load balancer, from L7 distributed denial of service (DDoS), SQL injection (SQLi), and cross-site scripting (XSS) attacks.

A. Configure Google Cloud Armor with the appropriate rules.
B. Configure a VM with appropriate scanning and filtering software in front of the HTTP(S) load balancer.
C. Configure Google Cloud WAF with the appropriate rules.
D. Configure Cloud NAT with the appropriate rules.

Select the simplest approach to accomplish this.

~~Feedback:~~

~~*A. Correct! Google Cloud Armor can be easily configured to protect against these types of attacks.~~
~~B. Incorrect. This can accomplish the desired effect but would not be as simple, scalable or cost effective as using Google Cloud Armor for that purpose~~
~~C. Incorrect. Google Cloud Armor is the managed WAF service for Google Cloud, and there is no product named Google Cloud WAF~~
~~D. Incorrect. Cloud NAT would not be able to satisfy the specified requirements~~

~~Where to look:~~

~~https://cloud.google.com/armor/docs/security-policy-overview~~
~~https://cloud.google.com/armor/docs/configure-security-policies~~
~~https://cloud.google.com/armor/docs/rules-language-reference~~
~~https://cloud.google.com/armor/docs/rule-tuning~~

~~Content Mapping:~~

- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M4 Load balancing~~
    - ~~M7 Network design and deployment~~
- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~

- M4 Load balancing
- Networking in Google Cloud: Hybrid connectivity and network management
  - M3 Network design and deployment
- Skill badges
  - Build and Secure Networks in Google Cloud Quest

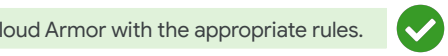
Summary:

Cloud Armor can provide WAF capabilities via an expression language that allows logical combinations of various clauses in allow or deny rules. Allow rules allow matching traffic and block all other traffic, while deny rules block matching traffic and allow all other traffic. Allow and deny rules can be combined in a priority order. Clauses can incorporate attack patterns or attributes of the request, such as source IP address, headers being present, or their values.

3.2 Diagnostic question 03 discussion

Cymbal Bank wants to protect their services, which are deployed behind an HTTP(S) load balancer, from L7 distributed denial of service (DDoS), SQL injection (SQLi), and cross-site scripting (XSS) attacks.

Select the simplest approach to accomplish this.

A. Configure Google Cloud Armor with the appropriate rules. 
B. Configure a VM with appropriate scanning and filtering software in front of the HTTP(S) load balancer.
C. Configure Google Cloud WAF with the appropriate rules.
D. Configure Cloud NAT with the appropriate rules.

Feedback:
*A. Correct! Google Cloud Armor can be easily configured to protect against these types of attacks.
B. Incorrect. This can accomplish the desired effect but would not be as simple, scaleable or cost effective as using Google Cloud Armor for that purpose
C. Incorrect. Google Cloud Armor is the managed WAF service for Google Cloud, and there is no product named Google Cloud WAF
D. Incorrect. Cloud NAT would not be able to satisfy the specified requirements

Where to look:
https://cloud.google.com/armor/docs/security-policy-overview
https://cloud.google.com/armor/docs/configure-security-policies
https://cloud.google.com/armor/docs/rules-language-reference
https://cloud.google.com/armor/docs/rule-tuning

Content Mapping:
- Instructor-led training
  - Networking in Google Cloud
    - M4 Load balancing
    - M7 Network design and deployment
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks

        - M4 Load balancing
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M3 Network design and deployment
- Skill badges
    - Build and Secure Networks in Google Cloud Quest

Summary:
Cloud Armor can provide WAF capabilities via an expression language that allows logical combinations of various clauses in allow or deny rules. Allow rules allow matching traffic and block all other traffic, while deny rules block matching traffic and allow all other traffic. Allow and deny rules can be combined in a priority order. Clauses can incorporate attack patterns or attributes of the request, such as source IP address, headers being present, or their values.

3.3 Diagnostic question 04 discussion

Cymbal Bank uses Cloud CDN to cache a web application served from a backend bucket connected to a Cloud Storage bucket. You need to cache all the web-app files with appropriate time to live (TTL) except for the index.html file. The index.html file contains links to versioned files and should always be fetched or re-validated from the origin.

Select the configuration option to satisfy these requirements with minimal effort.

A. Set the Cloud CDN cache mode for the backend bucket to CACHE_ALL_STATIC.

B. Set the Cloud CDN cache mode for the backend bucket to FORCE_CACHE_ALL, and ensure that the Cache-Control metadata for index.html is set to private.

C. Set the Cloud CDN cache mode for the backend bucket to CACHE_ALL_STATIC, and ensure that the Cache-Control metadata for index.html is not set or is set to no-store, no-cache, or private.

D. Set the Cloud CDN cache mode to USE_ORIGIN_HEADERS, set the Cache-Control metadata for index.html to no-store, and set the Cache-Control headers for all the other files with appropriate TTL values.

~~**Feedback:**~~

~~A. Incorrect. This option does not guarantee that the requirements are met. It would satisfy the requirements only if Cache-Control metadata on index.html is set appropriately.~~

~~B. Incorrect. This option would violate requirements and trigger caching for index.html~~

~~*C. Correct! This is the correct minimal-effort configuration to ensure that requirements are met.~~

~~D. Incorrect. This option would ensure that the caching requirements are met, but would require setting or verifying Cache-Control metadata across all the files of the web-app.~~

~~**Where to look:**~~

~~https://cloud.google.com/cdn/docs/overview~~
~~https://cloud.google.com/cdn/docs/features~~
~~https://cloud.google.com/cdn/docs/best-practices~~
~~https://cloud.google.com/cdn/docs/caching~~

~~**Content Mapping:**~~

- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M4 Load balancing~~
    - ~~M7 Network design and deployment~~

- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M4 Load balancing
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M3 Network design and deployment
- Skill badges
  - Build and Secure Networks in Google Cloud Quest

Summary:

Cloud CDN has a simple configuration model that utilizes a cache mode parameter. The parameter can be one of three values.

- CACHE_ALL_STATIC will cache static content based on Content-Type/MIME matching standard static types such as Javascript, CSS, photos, video, and audio, unless Cache-Control metadata for the associated object in Cloud Storage has a private or no-store directive. When the mode is CACHE_ALL_STATIC, a configuration parameter called Default TTL, which defaults to 1h, sets the lifetime for static file types that don't have Cache-Control specified expiry time. If they do have Cache-Control specified expiry time, they use the specified value. Other file types that are not in the set identified as static would be cached or not based on Cache-Control metadata normally.
- FORCE_CACHE_ALL will enforce caching of all objects regardless of Cache-Control metadata and will also use the Default TTL value for expiry time.
- USE_ORIGIN_HEADERS strictly uses the Cache-Control headers for controlling the caching. In addition to the Default TTL parameter there are also Max TTL and Client TTL configuration parameters that can adjust the behavior in the CACHE_ALL_STATIC and FORCE_CACHE_ALL modes.

3.3 Diagnostic question 04 discussion

Cymbal Bank uses Cloud CDN to cache a web application served from a backend bucket connected to a Cloud Storage bucket. You need to cache all the web-app files with appropriate time to live (TTL) except for the index.html file. The index.html file contains links to versioned files and should always be fetched or re-validated from the origin.

Select the configuration option to satisfy these requirements with minimal effort.

A. Set the Cloud CDN cache mode for the backend bucket to CACHE_ALL_STATIC.

B. Set the Cloud CDN cache mode for the backend bucket to FORCE_CACHE_ALL, and ensure that the Cache-Control metadata for index.html is set to private.

C. Set the Cloud CDN cache mode for the backend bucket to CACHE_ALL_STATIC, and ensure that the Cache-Control metadata for index.html is not set or is set to no-store, no-cache, or private.

D. Set the Cloud CDN cache mode to USE_ORIGIN_HEADERS, set the Cache-Control metadata for index.html to no-store, and set the Cache-Control headers for all the other files with appropriate TTL values.

Feedback:
A. Incorrect. This option does not guarantee that the requirements are met. It would satisfy the requirements only if Cache-Control metadata on index.html is set appropriately.
B. Incorrect. This option would violate requirements and trigger caching for index.html
*C. Correct! This is the correct minimal effort configuration to ensure that requirements are met.
D. Incorrect. This option would ensure that the caching requirements are met, but would require setting or verifying Cache-Control metadata across all the files of the web-app.

Where to look:
https://cloud.google.com/cdn/docs/overview
https://cloud.google.com/cdn/docs/features
https://cloud.google.com/cdn/docs/best-practices
https://cloud.google.com/cdn/docs/caching

Content Mapping:
- Instructor-led training
  - Networking in Google Cloud
    - M4 Load balancing
    - M7 Network design and deployment

- OnDemand
    - Networking in Google Cloud: Defining and implementing networks
        - M4 Load balancing
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M3 Network design and deployment
- Skill badges
    - Build and Secure Networks in Google Cloud Quest

Summary:
Cloud CDN has a simple configuration model that utilizes a cache mode parameter. The parameter can be one of three values.

- CACHE_ALL_STATIC will cache static content based on Content-Type/MIME matching standard static types such as Javascript, CSS, photos, video, and audio, unless Cache-Control metadata for the associated object in Cloud Storage has a private or no-store directive. When the mode is CACHE_ALL_STATIC, a configuration parameter called Default TTL, which defaults to 1h, sets the lifetime for static file types that don't have Cache-Control specified expiry time. If they do have Cache-Control specified expiry time, they use the specified value. Other file types that are not in the set identified as static would be cached or not based on Cache-Control metadata normally.
- FORCE_CACHE_ALL will enforce caching of all objects regardless of Cache-Control metadata and will also use the Default TTL value for expiry time.
- USE_ORIGIN_HEADERS strictly uses the Cache-Control headers for controlling the caching. In addition to the Default TTL parameter there are also Max TTL and Client TTL configuration parameters that can adjust the behavior in the CACHE_ALL_STATIC and FORCE_CACHE_ALL modes.

3.3 Diagnostic question 05 discussion

Cymbal Bank is serving files from a backend bucket and wants to ensure time-limited read access without authentication. The backend bucket uses signed URLs to access those files. The files are also being cached in Cloud CDN. There is a problem with one of the files, and you want to delete it. You also want to immediately ensure no read access via the signed URL to the cached file copy in Cloud CDN, although the expiry time is currently set to sometime in the future.

Select the option that accomplishes this with lowest cost and effort.

A. Perform cache invalidation for the file using the full path.
B. Perform cache invalidation for the file using the path and excluding the query parameters used for the signed URL.
C. Update the expiry time for the signed URL to be the current time.
D. Delete the key used to create the signed URL.

~~**Feedback:**~~
~~A. Incorrect. Cache invalidation requests would not consider query parameters and are not the most cost-effective approach.~~
~~B. Incorrect. Cache invalidation is not the lowest-cost approach to satisfy the requirements.~~
~~C. Incorrect. This action is not possible.~~
~~*D. Correct! Deleting the key will ensure that Cloud CDN rejects requests to the associated signed URLs and doesn't serve the cached values. This approach~~ ~~requires minimal cost and effort.~~

~~**Where to look:**~~
~~https://cloud.google.com/cdn/docs/private-content~~
~~https://cloud.google.com/cdn/docs/using-signed-urls~~
~~https://cloud.google.com/cdn/docs/using-signed-cookies~~
~~https://cloud.google.com/cdn/docs/cache-invalidation-overview~~
~~https://cloud.google.com/cdn/docs/invalidating-cached-content~~

~~**Content mapping:**~~
- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M4 Load balancing~~
    - ~~M7 Network design and deployment~~

- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~
    - ~~M4 Load balancing~~
  - ~~Networking in Google Cloud: Hybrid connectivity and network management~~
    - ~~M3 Network design and deployment~~

~~Summary:~~

~~Signed URLs and Cookies provide a mechanism to give controlled access to objects that are stored in Cloud Storage and cached in Cloud CDN.~~

3.3 Diagnostic question 05 discussion

Cymbal Bank is serving files from a backend bucket and wants to ensure time-limited read access without authentication. The backend bucket uses signed URLs to access those files. The files are also being cached in Cloud CDN. There is a problem with one of the files, and you want to delete it. You also want to immediately ensure no read access via the signed URL to the cached file copy in Cloud CDN, although the expiry time is currently set to sometime in the future.

Select the option that accomplishes this with lowest cost and effort.

A. Perform cache invalidation for the file using the full path.

B. Perform cache invalidation for the file using the path and excluding the query parameters used for the signed URL.

C. Update the expiry time for the signed URL to be the current time.

D. Delete the key used to create the signed URL. 

Feedback:
A. Incorrect. Cache invalidation requests would not consider query parameters and are not the most cost-effective approach.
B. Incorrect. Cache invalidation is not the lowest-cost approach to satisfy the requirements.
C. Incorrect. This action is not possible.
*D. Correct! Deleting the key will ensure that Cloud CDN rejects requests to the associated signed URLs and doesn't serve the cached values. This approach requires minimal cost and effort.

Where to look:
https://cloud.google.com/cdn/docs/private-content
https://cloud.google.com/cdn/docs/using-signed-urls
https://cloud.google.com/cdn/docs/using-signed-cookies
https://cloud.google.com/cdn/docs/cache-invalidation-overview
https://cloud.google.com/cdn/docs/invalidating-cached-content

Content mapping:
- Instructor-led training
  - Networking in Google Cloud
    - M4 Load balancing
    - M7 Network design and deployment

- OnDemand
    - Networking in Google Cloud: Defining and implementing networks
        - M4 Load balancing
    - Networking in Google Cloud: Hybrid connectivity and network management
        - M3 Network design and deployment

Summary:
Signed URLs and Cookies provide a mechanism to give controlled access to objects that are stored in Cloud Storage and cached in Cloud CDN.

3.4 Diagnostic question 06 discussion

Cymbal Bank will use a hybrid DNS approach. Cymbal has a VPC in Google Cloud that connects to their on-premises networks via Interconnect. You will use Cloud DNS for Cymbal's public DNS zone at cymbalbank.com and also for private DNS for resources at gcp.cymbalbank.com. You will use Cymbal's on-premises DNS, which is configured as authoritative for on-premises private resources at corp.cymbalbank.com.

Select the Cloud DNS managed zone configuration that will satisfy the requirements.

A. Create a single Cloud DNS managed zone in Google Cloud that is configured for private DNS for gcp.cymbalbank.com and public DNS for cymbalbank.com and that also acts as a forwarding zone to the on-premises DNS for corp.cymbalbank.com DNS requests.

B. Create a Cloud DNS private managed zone for gcp.cymbalbank.com, a public managed zone for cymbalbank.com, and a third forwarding zone for corp.cymbalbank.com that forwards DNS requests to the on-premises DNS.

C. Create a public managed zone for cymbalbank.com and a Cloud DNS private managed zone for gcp.cymbalbank.com that also forwards DNS requests for corp.cymbalbank.com to the on-premises DNS.

D. Create a Cloud DNS private managed zone for gcp.cymbalbank.com and a public managed zone for cymbalbank.com that also forwards DNS requests for corp.cymbalbank.com to the on-premises DNS.

~~**Feedback:**~~

~~A. Incorrect. This configuration is invalid. A single managed zone cannot be both public and private. Also forwarding zones must also be a separate managed zone.~~
~~*B. Correct! This is the valid configuration option.~~
~~C. Incorrect. This configuration is invalid. A separate forwarding zone is necessary to send DNS requests to the on-premises DNS.~~
~~D. Incorrect. This configuration is invalid. A separate forwarding zone is necessary to send DNS requests to the on-premises DNS.~~

~~**Where to look:**~~

~~https://cloud.google.com/dns/docs/overview~~
~~https://cloud.google.com/dns/docs/dns-overview~~
~~https://cloud.google.com/dns/docs/best-practices~~
~~https://cloud.google.com/dns/docs/key-terms~~
~~https://cloud.google.com/dns/docs/zones~~
~~https://cloud.google.com/dns/docs/records~~
~~https://cloud.google.com/dns/docs/dnssec~~

~~**Content mapping:** No coverage in existing training~~

~~**Summary:**~~

~~Cloud DNS allows for creation of managed zones that model DNS zones. These zones can contain DNS record sets for authoritative DNS serving for a specified public or private domain.~~

~~Cloud DNS also supports many other capabilities including DNS forwarding and peering, but requires valid configuration to support these DNS scenarios. Separate managed zones must be created for different domains, public vs private DNS, DNS peering, or DNS forwarding.~~

~~Cloud DNS also allows for DNS server policies to be created for VPCs that allow inbound or outbound forwarding of DNS requests. For outbound DNS forwarding, both outbound DNS server policies and forwarding zones can be used. When forwarding DNS requests to on-premises DNS servers, forwarding zones are the recommended approach.~~

3.4 Diagnostic question 06 discussion

Cymbal Bank will use a hybrid DNS approach. Cymbal has a VPC in Google Cloud that connects to their on-premises networks via Interconnect. You will use Cloud DNS for Cymbal's public DNS zone at cymbalbank.com and also for private DNS for resources at gcp.cymbalbank.com. You will use Cymbal's on-premises DNS, which is configured as authoritative for on-premises private resources at corp.cymbalbank.com.

Select the Cloud DNS managed zone configuration that will satisfy the requirements.

A. Create a single Cloud DNS managed zone in Google Cloud that is configured for private DNS for gcp.cymbalbank.com and public DNS for cymbalbank.com and that also acts as a forwarding zone to the on-premises DNS for corp.cymbalbank.com DNS requests.

B. Create a Cloud DNS private managed zone for gcp.cymbalbank.com, a public managed zone for cymbalbank.com, and a third forwarding zone for corp.cymbalbank.com that forwards DNS requests to the on-premises DNS.

C. Create a public managed zone for cymbalbank.com and a Cloud DNS private managed zone for gcp.cymbalbank.com that also forwards DNS requests for corp.cymbalbank.com to the on-premises DNS.

D. Create a Cloud DNS private managed zone for gcp.cymbalbank.com and a public managed zone for cymbalbank.com that also forwards DNS requests for corp.cymbalbank.com to the on-premises DNS.

Feedback:
A. Incorrect. This configuration is invalid. A single managed zone cannot be both public and private. Also forwarding zones must also be a separate managed zone.
*B. Correct! This is the valid configuration option.
C. Incorrect. This configuration is invalid. A separate forwarding zone is necessary to send DNS requests to the on-premises DNS.
D. Incorrect. This configuration is invalid. A separate forwarding zone is necessary to send DNS requests to the on-premises DNS.

Where to look:
https://cloud.google.com/dns/docs/overview
https://cloud.google.com/dns/docs/dns-overview
https://cloud.google.com/dns/docs/best-practices
https://cloud.google.com/dns/docs/key-terms
https://cloud.google.com/dns/docs/zones
https://cloud.google.com/dns/docs/records
https://cloud.google.com/dns/docs/dnssec

Content mapping: No coverage in existing training

Summary:

Cloud DNS allows for creation of managed zones that model DNS zones. These zones can contain DNS record sets for authoritative DNS serving for a specified public or private domain.

Cloud DNS also supports many other capabilities including DNS forwarding and peering, but requires valid configuration to support these DNS scenarios. Separate managed zones must be created for different domains, public vs private DNS, DNS peering, or DNS forwarding.

Cloud DNS also allows for DNS server policies to be created for VPCs that allow inbound or outbound forwarding of DNS requests. For outbound DNS forwarding, both outbound DNS server policies and forwarding zones can be used. When forwarding DNS requests to on-premises DNS servers, forwarding zones are the recommended approach.

3.4 Diagnostic question 07 discussion

You are configuring hybrid DNS for Google Cloud using Cloud DNS and your on-premises DNS. You have three VPC networks in Google Cloud in three different projects that should forward DNS requests for a particular private domain to the on-premises DNS. All three projects have Cloud VPN connections to the on-premises network.

Select the Google recommended approach for enabling this requirement.

A. For the VPC in one of the projects, create a Cloud DNS forwarding zone for its VPC. For the VPC in each of the other projects, create a Cloud DNS peering zone that targets the VPC with the forwarding zone.

B. Create a forwarding zone in one of the projects that is visible to the VPCs in all of the projects.

C. Create a forwarding zone in each of the projects that is visible to the VPC in that project.

D. Create a forwarding zone and a peering zone in each project. Make the forwarding zone visible to the VPC in the same project and the peering managed zones associated with the VPCs in the other projects.

~~**Feedback:**~~

~~*A. Correct! .This approach ensures that all DNS requests and responses are routed correctly to and from the on-premises network. It also satisfies the visibility requirements: private managed zones and forwarding zones are only visible to VPCs in the same project.~~

~~B. Incorrect. This approach is not possible. A forwarding zone can only be visible to VPCs in the same project.~~

~~C. Incorrect. This approach is not recommended. It would not function properly if all the VPCs were connected to the same on-premises network for routing reasons. The Cloud Routers would all advertise different routes for the same destination IP range in Google Cloud for the responses to Cloud DNS requests.~~

~~D. Incorrect. This approach is not possible. A peering managed zone can only be peered with one other project and VPC and also has visibility to one or more VPCs in the local project.~~

~~**Where to look:**~~

~~https://cloud.google.com/dns/docs/vpc-name-res-order~~
~~https://cloud.google.com/dns/docs/policies-overview~~
~~https://cloud.google.com/dns/docs/zones/cross-project-binding~~
~~https://cloud.google.com/dns/docs/server-policies-overview~~
~~https://cloud.google.com/dns/docs/zones/manage-response-policies~~
~~https://cloud.google.com/dns/docs/zones/manage-routing-policies~~

~~**Content mapping:** No coverage in existing training~~

~~**Summary:**~~
~~DNS Peering is recommended to avoid outbound DNS forwarding from multiple VPCs which can cause problems with return traffic. DNS peering allows a single forwarding zone to be associated with a single VPC and then other VPCs to have their requests forwarded by DNS peering with the forwarding zone.~~

3.4 Diagnostic question 07 discussion

You are configuring hybrid DNS for Google Cloud using Cloud DNS and your on-premises DNS. You have three VPC networks in Google Cloud in three different projects that should forward DNS requests for a particular private domain to the on-premises DNS. All three projects have Cloud VPN connections to the on-premises network.

Select the Google recommended approach for enabling this requirement.

A. For the VPC in one of the projects, create a Cloud DNS forwarding zone for its VPC. For the VPC in each of the other projects, create a Cloud DNS peering zone that targets the VPC with the forwarding zone.

B. Create a forwarding zone in one of the projects that is visible to the VPCs in all of the projects.

C. Create a forwarding zone in each of the projects that is visible to the VPC in that project.

D. Create a forwarding zone and a peering zone in each project. Make the forwarding zone visible to the VPC in the same project and the peering managed zones associated with the VPCs in the other projects.

Feedback:

*A. Correct! .This approach ensures that all DNS requests and responses are routed correctly to and from the on-premises network. It also satisfies the visibility requirements: private managed zones and forwarding zones are only visible to VPCs in the same project.

B. Incorrect. This approach is not possible. A forwarding zone can only be visible to VPCs in the same project.

C. Incorrect. This approach is not recommended. It would not function properly if all the VPCs were connected to the same on-premises network for routing reasons. The Cloud Routers would all advertise different routes for the same destination IP range in Google Cloud for the responses to Cloud DNS requests.

D. Incorrect. This approach is not possible. A peering managed zone can only be peered with one other project and VPC and also has visibility to one or more VPCs in the local project.

Where to look:

https://cloud.google.com/dns/docs/vpc-name-res-order
https://cloud.google.com/dns/docs/policies-overview
https://cloud.google.com/dns/docs/zones/cross-project-binding
https://cloud.google.com/dns/docs/server-policies-overview
https://cloud.google.com/dns/docs/zones/manage-response-policies
https://cloud.google.com/dns/docs/zones/manage-routing-policies

Content mapping: No coverage in existing training

Summary:
DNS Peering is recommended to avoid outbound DNS forwarding from multiple VPCs which can cause problems with return traffic. DNS peering allows a single forwarding zone to be associated with a single VPC and then other VPCs to have their requests forwarded by DNS peering with the forwarding zone.

3.5 Diagnostic question 08 discussion

Cymbal is using Cloud NAT to provide internet connectivity to a group of VMs in a subnet. There are 500 VMs in the subnet, and each VM may have up to 1000 internet bound connections simultaneously.

What Cloud NAT configuration will support this requirement?

A. Set the minimum ports per VM to 1000, and set the number of IP addresses used by the Cloud NAT Gateway to 8.
B. Set the minimum ports per VM to 2000, and set the number of IP addresses used by the Cloud NAT Gateway to 8.
C. Set the minimum ports per VM to 2000, and set the number of IP addresses used by the Cloud NAT Gateway to 10.
D. Set the minimum ports per VM to 1000, and set the number of IP addresses used by the Cloud NAT Gateway to 6.

~~**Feedback:**~~

~~*A. Correct! 1000 ports per VM will be sufficient to provide up to 1000 internet-bound UDP and TCP connections to any destination 3-tuple (combination of destination IP, destination port, and protocol) per VM. With a minimum 1000 ports allocated per VM, each NAT IP address can support up to 64.5 VMs (64,512 ports per NAT IP address/1000 ports per VM), and 8 NAT IP addresses would be sufficient for 500 VMs.~~

~~B. Incorrect. 2000 ports per VM would be more than you need. With that setting, a single NAT IP address could only support 32.26 VMs, and 8 NAT IP addresses would only support 258 VMs.~~

~~C. Incorrect. 2000 ports per VM would be more than you need. With that setting, a single NAT IP address could only support 32.26 VMs, and 10 NAT IP addresses would only support 322 VMs.~~

~~D. Incorrect. 6 NAT IP addresses with 1000 ports per VM would only support 387 VMs.~~

~~**Where to look:**~~

~~https://cloud.google.com/nat/docs/overview~~
~~https://cloud.google.com/nat/docs/ports-and-addresses~~
~~https://cloud.google.com/nat/docs/using-nat~~
~~https://cloud.google.com/nat/docs/gce-example~~

https://cloud.google.com/nat/docs/using-nat-rules

Content Mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M7 Network design and deployment
- OnDemand
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M3 Network design and deployment

Summary:

Cloud NAT allows outbound internet requests for VMs without external IP addresses and will allow responses to those outbound requests to be forwarded back to the initiating VM. Cloud NAT can either automatically allocate the set of external IP addresses it uses, or they can be manually specified. Each NAT IP address can support 64512 TCP and 64512 UDP connections. The connections can be distributed among all the VMs using that NAT gateway based on the minimum ports per VM configuration. This minimum ports per VM value can be either statically or dynamically configured, but will create a limit on how many VMs can be served by each NAT IP address.

3.5 Diagnostic question 08 discussion

Cymbal is using Cloud NAT to provide internet connectivity to a group of VMs in a subnet. There are 500 VMs in the subnet, and each VM may have up to 1000 internet bound connections simultaneously.

What Cloud NAT configuration will support this requirement?

A. Set the minimum ports per VM to 1000, and set the number of IP addresses used by the Cloud NAT Gateway to 8.
B. Set the minimum ports per VM to 2000, and set the number of IP addresses used by the Cloud NAT Gateway to 8.
C. Set the minimum ports per VM to 2000, and set the number of IP addresses used by the Cloud NAT Gateway to 10.
D. Set the minimum ports per VM to 1000, and set the number of IP addresses used by the Cloud NAT Gateway to 6.

Feedback:
*A. Correct! 1000 ports per VM will be sufficient to provide up to 1000 internet-bound UDP and TCP connections to any destination 3-tuple (combination of destination IP, destination port, and protocol) per VM. With a minimum 1000 ports allocated per VM, each NAT IP address can support up to 64.5 VMs (64,512 ports per NAT IP address/1000 ports per VM), and 8 NAT IP addresses would be sufficient for 500 VMs.
B. Incorrect. 2000 ports per VM would be more than you need. With that setting, a single NAT IP address could only support 32.26 VMs, and 8 NAT IP addresses would only support 258 VMs.
C. Incorrect. 2000 ports per VM would be more than you need. With that setting, a single NAT IP address could only support 32.26 VMs, and 10 NAT IP addresses would only support 322 VMs.
D. Incorrect. 6 NAT IP addresses with 1000 ports per VM would only support 387 VMs.

Where to look:
https://cloud.google.com/nat/docs/overview
https://cloud.google.com/nat/docs/ports-and-addresses
https://cloud.google.com/nat/docs/using-nat
https://cloud.google.com/nat/docs/gce-example

https://cloud.google.com/nat/docs/using-nat-rules

Content Mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M7 Network design and deployment
- OnDemand
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M3 Network design and deployment

Summary:

Cloud NAT allows outbound internet requests for VMs without external IP addresses and will allow responses to those outbound requests to be forwarded back to the initiating VM. Cloud NAT can either automatically allocate the set of external IP addresses it uses, or they can be manually specified. Each NAT IP address can support 64512 TCP and 64512 UDP connections. The connections can be distributed among all the VMs using that NAT gateway based on the minimum ports per VM configuration. This minimum ports per VM value can be either statically or dynamically configured, but will create a limit on how many VMs can be served by each NAT IP address.

3.6 Diagnostic question 09 discussion

You are designing a system in Google Cloud to ensure that all traffic being sent between two subnets is passed through a security gateway VM. The VM runs third-party software that scans traffic for known attack signatures and then forwards or drops traffic based on the scan results.

Select a configuration that satisfies these requirements.

A. Create the two subnets in the same VPC. Create a VM running the third-party scanning software in one of the subnets. Create custom routes in the VPC to send traffic for each subnet from the opposite subnet through that VM.

B. Create the two subnets in the same VPC. Create a VM running the third-party scanning software in each of the subnets. Create custom routes in the VPC to send traffic destined for each subnet originating in the opposite subnet through the VM in its subnet.

C. Create the two subnets in two separate VPCs. Create a VM with two network interfaces (NICs), with each NIC connected to the subnet in each VPC. Create custom routes in each VPC to send traffic destined for each subnet originating in the opposite subnet through the VM.

D. Create the two subnets in the same VPC. Create two VMs running the third-party scanning software, with one in each of the subnets. Create custom routes in the VPC to send traffic destined for each subnet originating in the opposite subnet through the VM in the opposite subnet.

~~**Feedback:**~~

~~A. Incorrect. This configuration is not possible. Custom routes cannot match or be contained in subnet routes, so there would be no way to enforce subnet bound traffic to go through the VM.~~

~~B. Incorrect. This configuration is not possible. Custom routes cannot match or be contained in subnet routes, so there would be no way to enforce subnet bound traffic to go through the VM.~~

~~*C. Correct! This configuration is the only option that can accomplish the requirements. Custom routes can be created in each VPC that match the subnet range of the opposite subnet with a next hop of the gateway VM. The gateway VM can then forward or drop the traffic based on scan results.~~

~~D. Incorrect. This configuration is not possible. Custom routes cannot match or be contained in subnet routes, so there would be no way to enforce subnet bound traffic to go through the VM.~~

~~**Where to look:**~~

~~https://cloud.google.com/vpc/docs/multiple-interfaces-concepts~~
~~https://cloud.google.com/vpc/docs/create-use-multiple-interfaces~~
~~https://cloud.google.com/load-balancing/docs/internal/ilb-next-hop-overview~~
~~https://cloud.google.com/load-balancing/docs/internal/setting-up-ilb-next-hop~~

~~Content Mapping:~~

- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M1 Google Cloud VPC networking fundamentals~~
    - ~~M3 Sharing networks across projects~~
    - ~~M7 Network Design and Deployment~~
- ~~OnDemand~~
  - ~~Networking in Google Cloud: Defining and implementing networks~~
    - ~~M1 Google Cloud VPC networking fundamentals~~
    - ~~M3 Sharing networks across projects~~
  - ~~Networking in Google Cloud: Hybrid connectivity and network management~~
    - ~~M3 Network Design and Deployment~~

~~Summary:~~

~~VMs with multiple NICs can be used to connect multiple VPCs. Each NIC in a VM with multiple NICs must be in a separate VPC and each VPC can have its own independent set of custom routes (in addition to the default routes) and custom routes can have next hops on a VM with multiple NICs that act as a gateway between the VPCs. This pattern can be useful in scenarios where VPCs with overlapping subnet ranges need to be connected for private IP communication, or in cases where traffic needs to be scanned between trusted and untrusted networks without using public IP addresses.~~

3.6 Diagnostic question 09 discussion

You are designing a system in Google Cloud to ensure that all traffic being sent between two subnets is passed through a security gateway VM. The VM runs third-party software that scans traffic for known attack signatures and then forwards or drops traffic based on the scan results.

Select a configuration that satisfies these requirements.

A. Create the two subnets in the same VPC. Create a VM running the third-party scanning software in one of the subnets. Create custom routes in the VPC to send traffic for each subnet from the opposite subnet through that VM.

B. Create the two subnets in the same VPC. Create a VM running the third-party scanning software in each of the subnets. Create custom routes in the VPC to send traffic destined for each subnet originating in the opposite subnet through the VM in its subnet.

C. Create the two subnets in two separate VPCs. Create a VM with two network interfaces (NICs), with each NIC connected to the subnet in each VPC. Create custom routes in each VPC to send traffic destined for each subnet originating in the opposite subnet through the VM.

D. Create the two subnets in the same VPC. Create two VMs running the third-party scanning software, with one in each of the subnets. Create custom routes in the VPC to send traffic destined for each subnet originating in the opposite subnet through the VM in the opposite subnet.

Feedback:

A. Incorrect. This configuration is not possible. Custom routes cannot match or be contained in subnet routes, so there would be no way to enforce subnet bound traffic to go through the VM.

B. Incorrect. This configuration is not possible. Custom routes cannot match or be contained in subnet routes, so there would be no way to enforce subnet bound traffic to go through the VM.

*C. Correct! This configuration is the only option that can accomplish the requirements. Custom routes can be created in each VPC that match the subnet range of the opposite subnet with a next hop of the gateway VM. The gateway VM can then forward or drop the traffic based on scan results.

D. Incorrect. This configuration is not possible. Custom routes cannot match or be contained in subnet routes, so there would be no way to enforce subnet bound traffic to go through the VM.

Where to look:

https://cloud.google.com/vpc/docs/multiple-interfaces-concepts
https://cloud.google.com/vpc/docs/create-use-multiple-interfaces
https://cloud.google.com/load-balancing/docs/internal/ilb-next-hop-overview
https://cloud.google.com/load-balancing/docs/internal/setting-up-ilb-next-hop

Content Mapping:

- Instructor-led training
  - Networking in Google Cloud
    - M1 Google Cloud VPC networking fundamentals
    - M3 Sharing networks across projects
    - M7 Network Design and Deployment
- OnDemand
  - Networking in Google Cloud: Defining and implementing networks
    - M1 Google Cloud VPC networking fundamentals
    - M3 Sharing networks across projects
  - Networking in Google Cloud: Hybrid connectivity and network management
    - M3 Network Design and Deployment

Summary:

VMs with multiple NICs can be used to connect multiple VPCs. Each NIC in a VM with multiple NICs must be in a separate VPC and each VPC can have its own independent set of custom routes (in addition to the default routes) and custom routes can have next hops on a VM with multiple NICs that act as a gateway between the VPCs. This pattern can be useful in scenarios where VPCs with overlapping subnet ranges need to be connected for private IP communication, or in cases where traffic needs to be scanned between trusted and untrusted networks without using public IP addresses.

3.6 Diagnostic question 10 discussion

Select the list of the resources that must be created or configured to enable packet mirroring.

A. A packet mirroring policy and a collector instance
B. A packet mirroring policy, an internal TCP/UDP load balancer configured for packet mirroring, an instance group of collector instances, and firewall rules
C. A packet mirroring policy, a collector instance, and firewall rules
D. A packet mirroring policy, an instance group of collector instances, and firewall rules

~~**Feedback:**~~

~~A. Incorrect. This list is missing necessary components: an internal TCP/UDP load balancer configured for packet mirroring, an instance group of collector instances, and firewall rules.~~

~~*B. Correct! This list contains the necessary component resources.~~

~~C. Incorrect. This list is missing necessary components: an internal TCP/UDP load balancer configured for packet mirroring and an instance group of collector instances.~~

~~D. Incorrect. This list is missing a necessary component: an internal TCP/UDP load balancer configured for packet mirroring.~~

~~**Where to look:**~~

~~https://cloud.google.com/vpc/docs/packet-mirroring~~
~~https://cloud.google.com/vpc/docs/using-packet-mirroring~~
~~https://cloud.google.com/vpc/docs/monitoring-packet-mirroring~~

~~**Content mapping:**~~

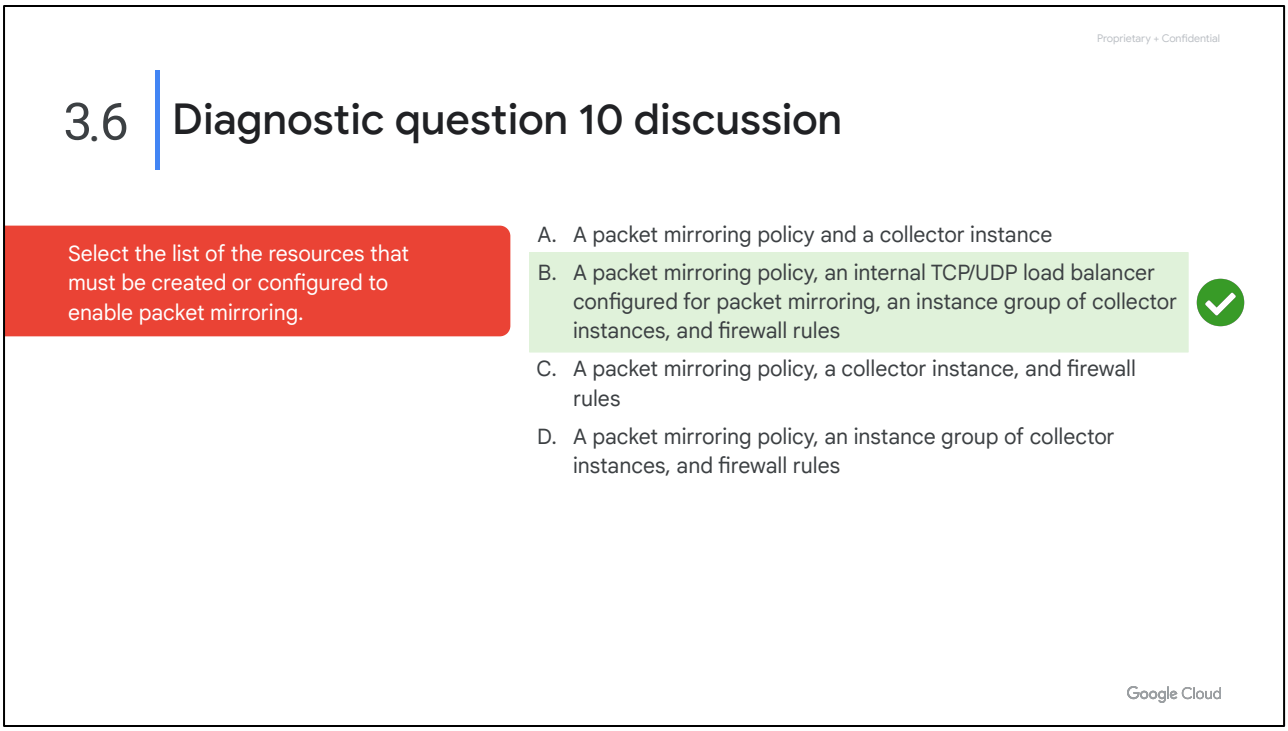
- ~~Instructor-led training~~
  - ~~Networking in Google Cloud~~
    - ~~M8 Network monitoring and troubleshooting~~
- ~~OnDemand~~
  - ~~Networking in Google Cloud: Hybrid connectivity and network management~~

- M4 Network monitoring and troubleshooting

- Skill badge
  - Security and Identity Fundamentals Quest

Summary:

Each packet mirroring policy sends traffic from a collection of source VMs, which must all be in the same project, VPC, and region, to a destination internal TCP/UDP load balancer configured for packet mirroring. The source VMs can be specified in the packet mirroring policy by name, tag, or subnet, with different limitations depending on the approach taken. The destination load balancer can be connected to one or more VMs in an unmanaged or managed instance group. It must also be in the same region, but can be in either the same VPC or a peered VPC. Packet mirroring policies can be configured to forward only ingress or only egress traffic, or to limit the traffic to only include certain source or destination IP ranges and/or protocols.



Feedback:
A. Incorrect. This list is missing necessary components: an internal TCP/UDP load balancer configured for packet mirroring, an instance group of collector instances, and firewall rules.
*B. Correct! This list contains the necessary component resources.
C. Incorrect. This list is missing necessary components: an internal TCP/UDP load balancer configured for packet mirroring and an instance group of collector instances.
D. Incorrect. This list is missing a necessary component: an internal TCP/UDP load balancer configured for packet mirroring.

Where to look:
https://cloud.google.com/vpc/docs/packet-mirroring
https://cloud.google.com/vpc/docs/using-packet-mirroring
https://cloud.google.com/vpc/docs/monitoring-packet-mirroring

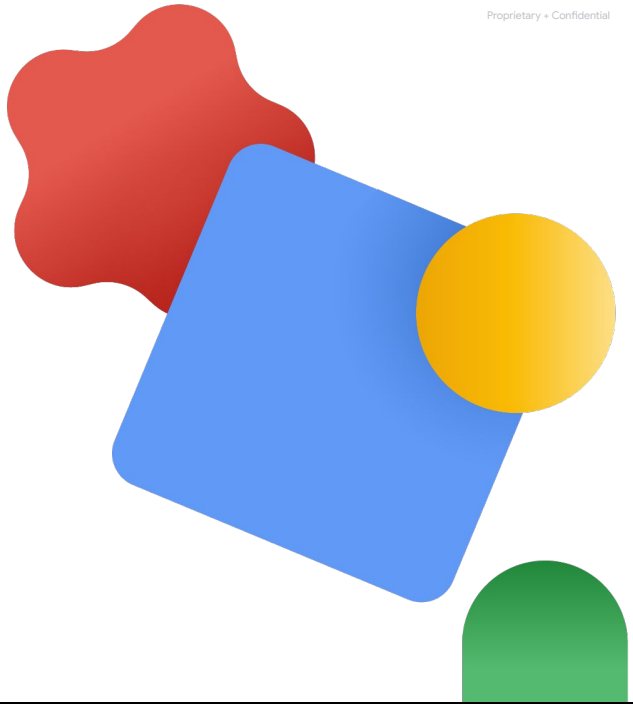
Content mapping:
- Instructor-led training
  - Networking in Google Cloud
    - M8 Network monitoring and troubleshooting
- OnDemand
  - Networking in Google Cloud: Hybrid connectivity and network management

        - M4 Network monitoring and troubleshooting
- Skill badge
    - Security and Identity Fundamentals Quest

Summary:

Each packet mirroring policy sends traffic from a collection of source VMs, which must all be in the same project, VPC, and region, to a destination internal TCP/UDP load balancer configured for packet mirroring. The source VMs can be specified in the packet mirroring policy by name, tag, or subnet, with different limitations depending on the approach taken. The destination load balancer can be connected to one or more VMs in an unmanaged or managed instance group. It must also be in the same region, but can be in either the same VPC or a peered VPC. Packet mirroring policies can be configured to forward only ingress or only egress traffic, or to limit the traffic to only include certain source or destination IP ranges and/or protocols.

More aspects of Virtual Private Clouds (VPCs)

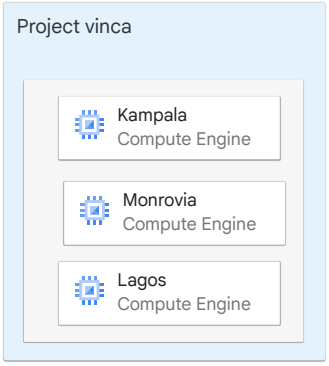


Next, let's explore some more aspects of Virtual Private Clouds or VPCs.

VPC networks

A Virtual Private Cloud (VPC) network is a virtual version of a physical network that:

- Provides connectivity for your compute engine virtual machine (VM) instances.
- Offers native Internal TCP/UDP load balancing and proxy systems for Internal HTTP(S) load balancing.
- Distributes traffic from Google Cloud external load balancers to backends.



A Virtual Private Cloud (VPC) network is a virtual version of a physical network that provides connectivity for your Compute Engine virtual machine (VM) instances, including Google Kubernetes Engine (GKE) clusters, App Engine flexible environment instances, and other Google Cloud products built on Compute Engine VMs.

You can configure native Internal TCP/UDP Load Balancing and proxy systems for Internal HTTP(S) Load Balancing with your VPC network.

A VPC network connects to on-premises networks by using Cloud VPN tunnels and Cloud Interconnect attachments and distributes traffic from Google Cloud external load balancers to backends.

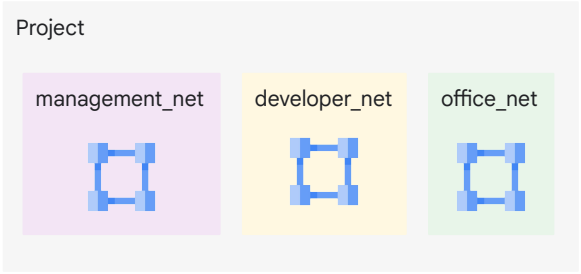
By default, every network has routes that let instances in a network send traffic directly to each other, even across subnets. In addition, every network has a default route that directs packets to destinations that are outside the network. Although these routes cover most of your normal routing needs, you can also create special routes that override these routes.

Just creating a route does not ensure that your packets will be received by the specified next hop. Firewall rules must also allow the packet.

The default network has pre-configured firewall rules that allow all instances in the network to talk with each other. Manually created networks do not have such rules, so you must create them.

VPC networks

- Projects can contain multiple VPC networks.
- New projects start with a default network (an auto mode VPC network) that has one subnetwork (subnet) in each region.
  - Google-recommended practice: create a custom mode VPC network.

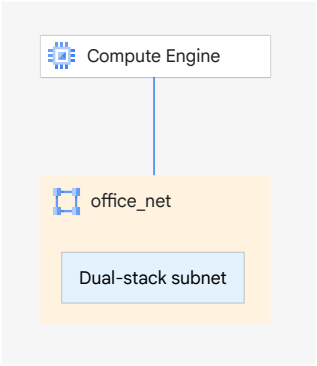


Projects can contain multiple VPC networks. Unless you create an organizational policy that prohibits it, new projects start with a default network (an auto mode VPC network) that has one subnetwork (subnet) in each region.

An auto mode VPC network can be useful when you start learning about Google Cloud. However, it's a best practice to create a custom mode network and include subnetworks only in desired regions.

Subnets and IPv6 support

- VPC networks now support IPv6 addresses.
- Support for IPv6 addresses can vary per subnet.
- To support IPv6, Google Cloud has introduced the concept of a subnet stack.
  - Single-stack subnets support IPv4.
  - Dual-stack subnets support IPv4 and IPv6.
- IPv6 addresses can be assigned to objects in a subnet that supports IPv6.



#NEWVIDEO

VPC networks now support IPv6 addresses.

Support for IPv6 addresses can vary per subnet. To support IPv6, Google Cloud has introduced the concept of a subnet stack. The subnet stack defines the type of address that can be assigned to objects in the subnet.

Single-stack subnets support IPv4. Dual-stack subnets support IPv4 and IPv6. There's no subnet that only supports IPv6.

IPv6 addresses can be assigned to objects in a subnet that supports IPv6. In other words, you can only assign IPv6 addresses to objects in a dual-stack subnet.

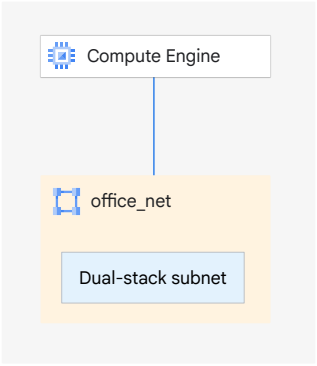
IPv6 in Google Cloud – Getting Started

You can use Google Cloud IPv6 in:

- Dual-stack subnets, VM instances, and instances with multiple interfaces (multi-NIC).
- VPC firewall rules and hierarchical firewall policy rules.
- Either internal IPv6 addresses (with Private Google Access) or external IPv6 addresses to access Google APIs and services from your VPC.
- Dual-stack VPC peering HA-VPN tunnels and exchange dynamic IPv6 routes.
- Cloud Load Balancing.

To use IPv6, set up a dual-stack subnet

- You can configure the IPv6 access type as internal or external.
- Internal IPv6 addresses are used for communication between VMs within VPC networks.
- External IPv6 addresses:
  - Can be used for communication between VMs within VPC networks.
  - Are also routable on the internet.
- Connected VMs inherit the IPv6 access type from the subnet.



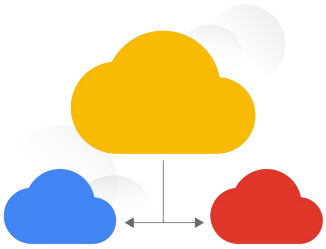
You can configure the IPv6 access type to be internal or external.

Internal IPv6 addresses are used for VM to VM communication within VPC networks. These use unique local addresses (ULAs), which can only be routed within VPC networks and cannot be routed to the internet.

External IPv6 addresses can be used for communication between VMs within VPC networks. These use global unicast addresses (GUAs) and are also routable on the internet
Connected VMs inherit the IPv6 access type from the subnet.

Assigning IPv6 address ranges to a VPC network

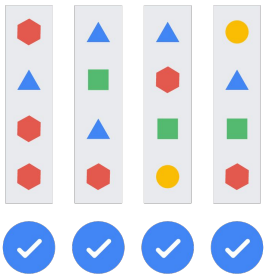
- To enable internal IPv6 on a subnet, you must first assign an internal IPv6 range on the VPC network.
- A /48 ULA range from within fd20::/20 is assigned to the network.
  - All internal IPv6 subnet ranges in the network are assigned from this /48 range.
  - The /48 range can be automatically assigned, or you can select a specific range from within fd20::/20.



To enable internal IPv6 on a subnet, you must first assign an internal IPv6 range on the VPC network. A /48 ULA range from within fd20::/20 is assigned to the network. All internal IPv6 subnet ranges in the network are assigned from this /48 range. The /48 range can be automatically assigned, or you can select a specific range from within fd20::/20.

Assigning IPv6 address ranges to a subnet

- When you enable IPv6 on a VM, the VM is assigned a /96 range from the subnet.
- The first IP address in that range is assigned to the primary interface.
- You don't configure whether a VM gets internal or external IPv6 addresses.
  - The VM inherits the IPv6 access type from the subnet.



When you enable IPv6 on a VM, the VM is assigned a /96 range from the subnet that it's connected to. The first IP address in that range is assigned to the primary interface. You don't configure whether a VM gets internal or external IPv6 addresses. The VM inherits the IPv6 access type from the subnet that it's connected to.

IPv6 caveats

01 Dual-stack subnets are not supported on auto mode VPC networks or legacy networks.

02 Any interface on a VM can have IPv6 addresses configured.

When configuring your VPC networks and subnets to use IPv6 address, consider these caveats:

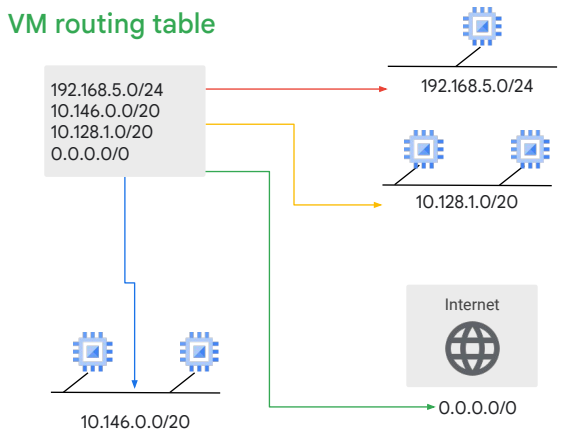
Dual-stack subnets are not supported on auto mode VPC networks or legacy networks. If you have an auto mode VPC network that you want to add dual-stack subnets to, you can convert the auto mode VPC network to custom mode.

If you're converting a legacy custom network, create new dual-stack subnets, or convert existing subnets to dual-stack.

Any interface on a VM can have IPv6 addresses configured.

Routes

- Define the paths that network traffic takes from a virtual machine (VM) instance to other destinations.
- Apply to traffic that egresses a VM.
- Forward traffic to most specific route.
- Deliver traffic only if it also matches a firewall rule.
- Can be fine-tuned using network tags.



Routes define the paths that network traffic takes from a virtual machine (VM) instance to other destinations. These destinations can be inside your Google Cloud Virtual Private Cloud (VPC) network (for example, in another VM) or outside it.

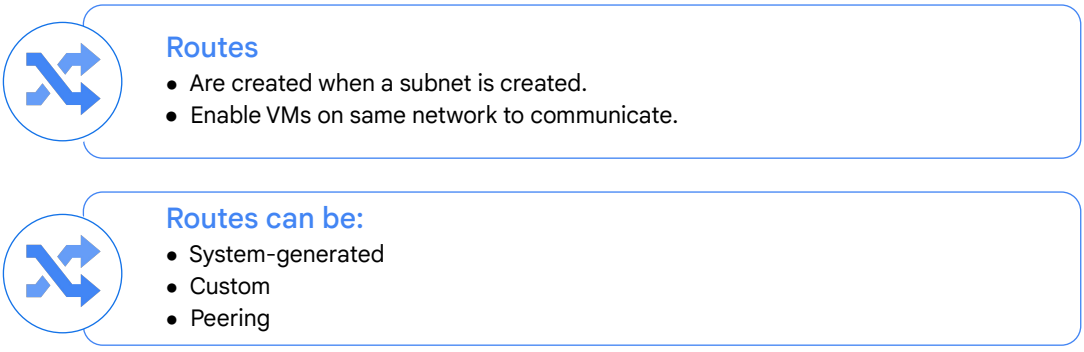
Routes match packets by destination IP address. However, no traffic will flow without also matching a firewall rule.

A route is created when a network is created, which enables traffic delivery from anywhere. Also, a route is created when a subnet is created. This is what allows VMs on the same network to communicate.

Network tags fine-tune which route is picked. If a route has a network tag, it can be applied only to instances that have the same network tag. Routes without network tags can apply to all instances in the network.

This slide shows a simplified routing table.

Routes and route types

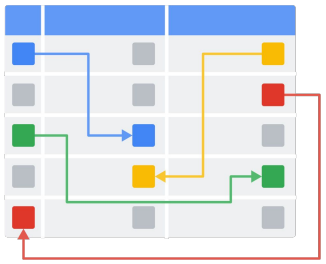


A route is created when a network or subnet is created, enabling traffic delivery from anywhere. This is what allows VMs on the same network to communicate.

A route can be system-generated, custom, or peering. System-generated routes are simple and can be used by default. When they do not provide the desired granularity, create custom routes. For example, custom routes can be used to route traffic between subnets through a network virtual appliance. Peering routes are used for network peering. Next, you'll learn more about system-generated and custom route types. Peering is covered in another module.

Overview of system-generated default routes

- When you create a VPC network, it includes a system-generated IPv4 default route (0.0.0.0/0).
- When you create a dual-stack subnet with an external IPv6 address range, a system-generated IPv6 default route (::/0) is added to the VPC network.
- The IPv4 and IPv6 default routes define a path to external IP addresses.
- System-generated routes can serve as a path to Google APIs and services when you are not using a Private Service Connect endpoint.



When you create a VPC network, it includes a system-generated IPv4 default route (0.0.0.0/0).

When you create a dual-stack subnet with an external IPv6 address range in a VPC network, a system-generated IPv6 default route (::/0) is added. If the default route doesn't exist, it isn't added.

The IPv4 and IPv6 default routes that serve these purposes define a path out of the VPC network to external IP addresses on the internet.

If you access Google APIs and services without using a Private Service Connect endpoint, the default route can serve as the path to Google APIs and services. Private Service Connect enables you to publish and consume services by using the internal IP addresses that you define. You'll learn more about Private Service Connect later in this course. For more information, in the Google Cloud documentation, refer to Configuring Private Google Access and Accessing APIs from VMs with external IP addresses.

Using system-generated default routes

- A default route is used only if a route with a more specific destination does not apply to a packet.
- To completely isolate a network from the internet or to replace the default route with a custom route, delete the default route:
  - **IPv4 only**: to route internet traffic to a different next hop, replace the default route with a custom static or dynamic route.
  - **IPv4 and IPv6**: if you delete the default route and don't replace it, packets destined to IP ranges that are not covered by other routes are dropped.

Google Cloud only uses a default route if a route with a more specific destination does not apply to a packet. For information about how destination specificity and route priority influence route selection, see Routing order in the Google Cloud documentation.

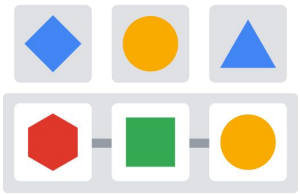
To completely isolate your network from the internet or to replace the default route with a custom route, you can delete the default route. For IPv4 only, to route internet traffic to a different next hop, you can replace the default route with a custom static or dynamic route. For example, you could replace it with a custom static route whose next hop is a proxy VM. If you delete the default route and do not replace it, packets to IP ranges not covered by other routes are dropped.

If you don't have custom static routes that meet the routing requirements for Private Google Access, deleting the default route might disable Private Google Access.

Some organizations do not want a default route pointing to the internet; instead, they want the default route to point to an on-premises network. To do that, you can create a custom route. You will learn about custom routes later in this module.

Overview of system-generated subnet routes

- When you create a subnet, system-generated subnet routes are automatically created.
- Subnet routes:
  - Apply to the subnet, not to the whole network.
  - Always have the most specific destinations.
  - Cannot be overridden by higher priority routes (lower number equals higher priority).
- Each subnet has at least one subnet route whose destination matches the subnet's primary IP range.
- If the subnet has secondary IP ranges, each secondary IP address range has a corresponding subnet route.



When you create a subnet, system-generated subnet routes are automatically created.

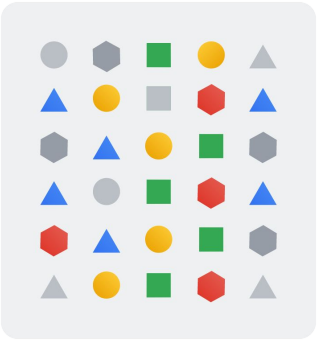
Subnet routes apply to the subnet, not the whole network. They always have the most specific destination and cannot be overridden by higher priority routes. Recall that lower priority number indicate higher priority, so 1 would have a higher priority than 10.

Each subnet has at least one subnet route whose destination matches the primary IP range of the subnet.

If the subnet has secondary IP ranges, each secondary IP address range has a corresponding subnet route.

Overview of custom static routes

- Custom static routes forward packets to a static route next hop and are useful for small, stable topologies.
- Benefits over dynamic routing include:
  - Quicker routing performance (lower processing overhead).
  - More security (no route advertisement).
- It has its limitations:
  - Cannot point to a VLAN attachment.
  - Require more maintenance, because routes are not dynamically updated.



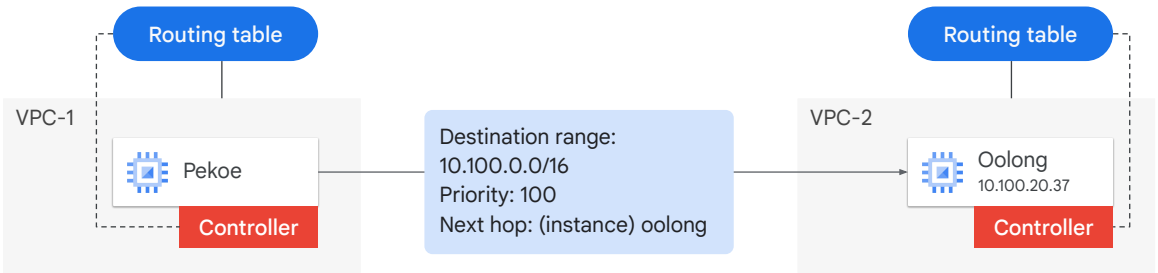
Custom static routes forward packets to a static route next hop and are useful for small, stable topologies.

Dynamic routing generally provides quicker routing performance. Unlike dynamic routing, no processing power is devoted to maintaining and modifying the routes (hence the quicker performance). Custom static routing is more secure than dynamic routing, because there's no route advertisement.

Note these custom static routing limitations. A custom static route cannot point to a VLAN attachment. It also requires more maintenance, because routes are not dynamically updated. For example, a topology change on either network requires you to update static routes. Also, if a link fails, static routes can't reroute traffic automatically. For small, stable topologies, this is not always a significant concern.

Custom static routes

- The controller is kept informed of all routes from the network's routing table.
- Route changes are propagated to the VM controllers.



The controller is kept informed of all routes from the network's routing table. Route changes are propagated to the VM controllers. When you add or delete a route, the set of changes is propagated to the VM controllers. In this example, if you change any of the routes to the oolong VM, pekoe can still route packets to oolong.

Create custom static routes

Create custom static routes:

- Manually, by using either the Google Cloud console, gcloud CLI compute routes create command, or the routes.insert API.
- Automatically, by using either the console to create a Classic VPN tunnel with policy-based routing or as a route-based VPN.

You can create custom static routes either manually or automatically. To create custom static routes manually, use the Google Cloud app, the gcloud CLI compute routes create command, or the routes.insert API. To create the routes automatically, you can use the Google Cloud app to create a Classic VPN tunnel with policy-based routing or as a route-based VPN. For more information, see Cloud VPN networks and tunnel routing.

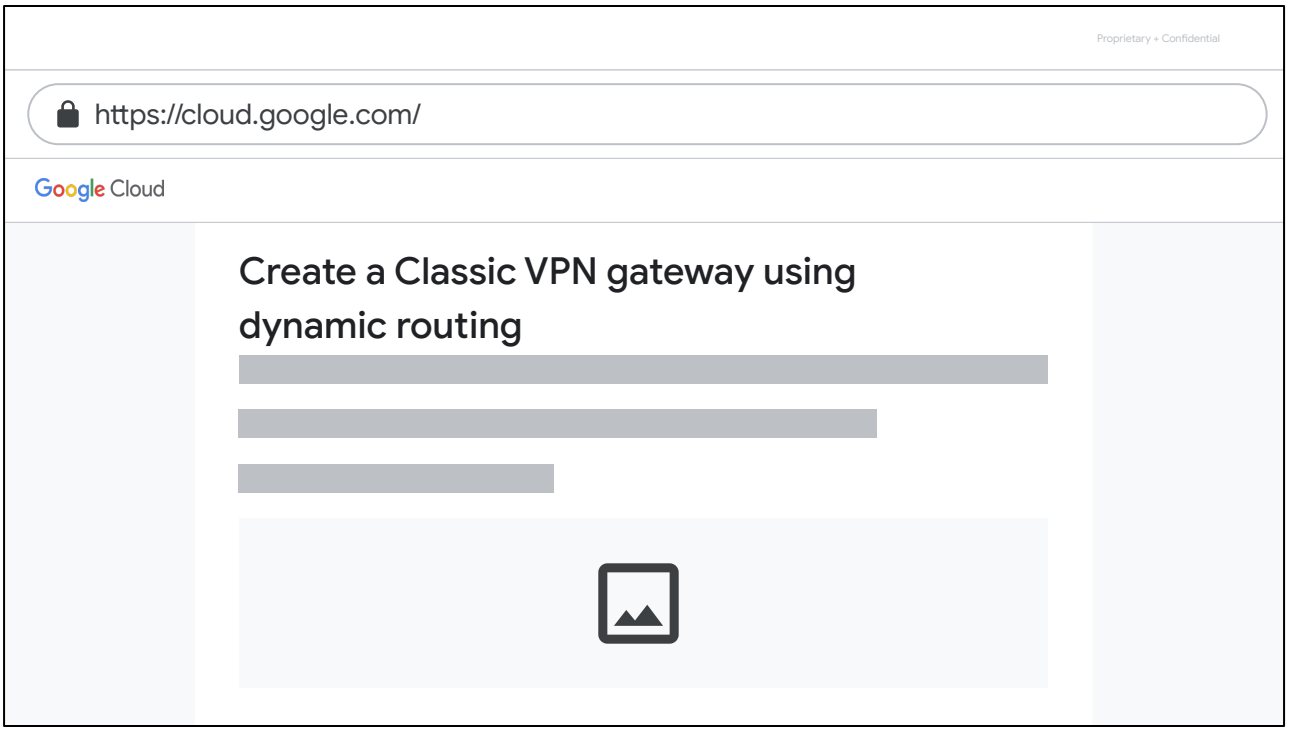
Dynamic routes

- Are managed by Cloud routers.
- Always represent IP address ranges outside your VPC network, which are received from a BGP peer.
- Dynamic routes are used by:
  - Dedicated Interconnect
  - Partner Interconnect
  - HA VPN tunnels
  - Classic VPN tunnels that use dynamic routing (very limited in scope)

Dynamic routes are managed by Cloud Routers in the VPC network. Their destinations always represent IP address ranges outside your VPC network, which are received from a BGP peer router. BGP peer routers are typically outside the Google network (like on-premises or another cloud provider).

Dynamic routes are used by:

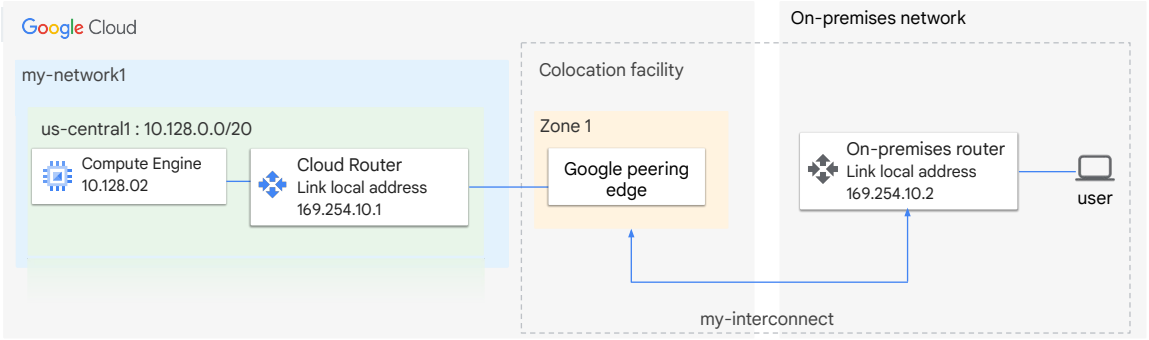
- Dedicated Interconnect
- Partner Interconnect
- HA VPN tunnels
- Classic VPN tunnels that use dynamic routing



For more information visit the document titled ‘Create a Classic VPN gateway using dynamic routing’.

A dynamic routing example

- Routes are added and removed automatically by Cloud routers in your VPC network.
- Routes apply to VMs according to the VPC network's dynamic routing mode.



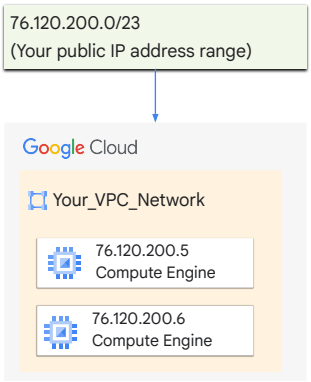
Routes are added and removed automatically by Cloud Routers in your VPC network. The routes apply to VMs according to the VPC network's dynamic routing mode.

This example shows a VPC network connected to an on-premises network that uses Dedicated Interconnect.

Cloud Router handles the BGP advertisements and adds them as custom routes. Cloud Router creates a BGP session for the VLAN attachment and its corresponding on-premises peer router. The Cloud Router receives the routes that your on-premises router advertises. These routes are added as custom dynamic routes in your VPC network. The Cloud Router also advertises routes for Google Cloud resources to the on-premises peer router.

Introduction to BYOIP (bring your own IP address)

- BYOIP enables customers to:
  - Assign IP addresses from a public IP range that they own to Google Cloud resources.
  - Route traffic directly from the internet to their VMs.
- Google Cloud manages these BYOIP addresses in the same way as Google-provided IP addresses, except that:
  - The IP addresses are available only to the customer who bought them.
  - Idle or in-use IP addresses incur no charges.



BYOIP enables customers to assign IP addresses from a public IP range that they own to Google Cloud resources. With BYOIP, customers can route traffic directly from the internet to their VMs without having to go through their own physical networks.

After the IP addresses are imported, Google Cloud manages them in the same way as Google-provided IP addresses, with these exceptions:

- The IP addresses are available only to the customer who bought them.
- Idle or in-use IP addresses incur no charges.

BYOIP guidelines

The object that the IP address is assigned to:

- Can have a regional scope or a global scope.
- Must support an external address type.
- Cannot be a Classic VPN gateway, GKE (Google Kubernetes Engine) node, GKE pod, autoscaling MIG (managed instance group).

The object that the IP address is assigned to can have a regional scope, like a VM or the forwarding rule of a network load balancer. It can also have a global scope, like the forwarding rule of a global external HTTP(S) load balancer.

It must support an external address type, because BYOIP ranges will be advertised by Google to the public internet.

A BYOIP address can't be assigned to a Classic VPN gateway, GKE (Google Kubernetes Engine) node, GKE pod, or an autoscaling MIG (managed instance group).

BYOIP caveats

01 BYOIP prefixes cannot overlap with subnet or alias ranges in the VPC.

02 The IP address must be IPv4.

03 Overlapping BGP route announcements can be problematic.

BYOIP prefixes cannot overlap with subnet or alias ranges in the VPC used by the customer.

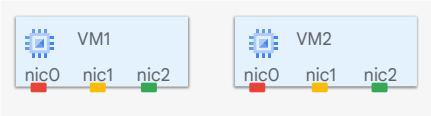
For BYOIP, the IP address must be IPv4. Importing IPv6 addresses is not supported.

Overlapping BGP route announcements can be problematic. BGP is a routing protocol that picks the most efficient route to send a packet. If Google and another network advertise the same route with matching or mismatched prefix lengths, BGP cannot work properly. You might experience unexpected routing and packet loss.

For example: suppose you're advertising a 203.0.112.0/20 address block and you're using BGP to route packets. You could bring a 203.0.112.0/23 address block that you own to Google using BYOIP, and set it up to route externally. Because the /23 block is contained within the /20 block, BGP route announcements can/might overlap. If you're maintaining the routing registry correctly, BPG routing practices cause the more specific route to take precedence. Thus, the /23 block will take precedence over the /20 block. However, if the /23 route ever stopped being advertised, the /20 block could be used.

VPC networks are isolated by default

- VPC networks
  - Use an internal IP to communicate within networks.
  - Use an external IP to communicate across networks.
- To communicate internally with multiple networks, add multiple network interface controllers (NICs).
- VPC Peering (discussed later) also allows instances in different VPC to communicate via internal IP addresses



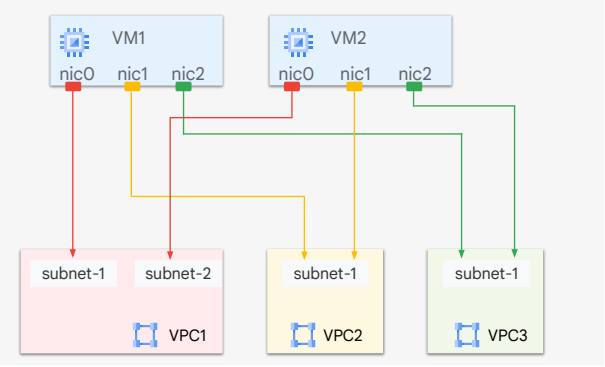
VPC networks are isolated private networking domains by default. As we mentioned earlier, VM instances within a VPC network can communicate among themselves by using internal IP addresses as long as firewall rules permit. However, no internal IP address communication is allowed between networks unless you set up mechanisms such as VPC peering or VPN.

Every instance in a VPC network has a default network interface. You can create additional network interfaces attached to your VMs through network interface controllers (NICs).

Network interface controllers

Each NIC:

- Is attached to a separate VPC network.
- Uses an internal IP to communicate across networks.



Multiple network interfaces let you create configurations in which an instance connects directly to several VPC networks. Each of the interfaces must have an internal IP address, and each interface can also have an external IP address.

For example, in this diagram, you have two VM instances. Each instance has network interfaces to a subnet within VPC1, VPC2, and VPC3.

For some situations, you might require multiple interfaces; for example, to configure an instance as a network appliance for load balancing. Multiple network interfaces are also useful when applications running in an instance require traffic separation, such as separation of data plane traffic from management plane traffic.

Multiple network interface caveats

01 Network interfaces can only be configured when you create an instance.

02 Each interface must be in a different network.

03 The network IP ranges cannot overlap.

04 The networks must exist before you create the VM.

When creating VM instances with multiple network interfaces, note these caveats.

You can only configure a network interface when you create an instance.

Each network interface configured in a single instance must be attached to a different VPC network. Each interface must belong to a subnet whose IP range does not overlap with the subnets of any other interfaces.

The additional VPC networks that the multiple interfaces will attach to must exist before you create the instance.

Multiple network interface caveats

- Cannot delete interface without deleting the VM.
- Internal DNS (Domain Name System) is only associated to nic0.
- You can have up to 8 NICs, depending on the VM.

Type of instance	# of virtual NICs
VM <= 2 vCPU	2 NICs
VM >2vCPU	1 NIC per vCPU (Max: 8)

You cannot delete a network interface without deleting the instance.

When an internal DNS (Domain Name System) query is made with the instance hostname, it resolves to the primary interface (nic0) of the instance. If the nic0 interface of the instance belongs to a different VPC network than the instance that issues the internal DNS query, the query will fail. You will explore this in the upcoming lab.

The maximum number of network interfaces per instance is 8, but this depends on the instance's machine type, as shown in this table:

Instances with less than or equal to 2 vCPU can have up to 2 virtual NICs. Examples include the f1-micro, g1-small, n1-standard-1, and any other custom VMs with 1 or 2 vCPUs.

Instances with more than 2 vCPU can have 1 NIC per vCPU, with a maximum of 8 virtual NICs.

Inter VPC connectivity: How will you connect your VPCs?

01

VPC Peering

- Global
- High throughput, low latency
- Fast and easy to configure
- Works between projects and organizations
- Doesn't add any cost
- NOT transitive
- Exchanges all subnet routes
- Custom and dynamic routes can be exchanged
- Peered VPCs share some limits and quotas

@trainer: please use this link for context regarding the next four slides:
VPC Network Peering

Inter VPC connectivity: How will you connect your VPCs?

02

HA VPN

- Regional in nature (can still route to all regions inside a VPC)
- Works between projects and organizations
- Added costs
- Around 3 Gbs per tunnel
- Can provide transitive routing
- Routes exchanged via BGP
- Does not share limits and quotas between VPCs

Inter VPC connectivity: How will you connect your VPCs?

03

Multi NIC instances

- Regional in nature (can still route to all regions inside a VPC)
- VPCs must be in the same organization and Google Cloud project
- Cost and performance based on the type/flavor of VM
- Can provide additional services offered by the instance, ie advanced firewall services
- Does not share limits and quotas between VPCs but shares project-level quotas

Best practices

VPC Peering / VPC Peering / Multi NIC instances

Best practices

Use VPC peering to get started. HA VPN can be used when you don't want to share network limits across VPCs and Multi-NIC VM when you need inspection/security appliances between VPCs.

VPC network best practices

Custom mode VPCs	Prevent overlapping IPs and control subnet creation by creating VPCs with custom subnet creation mode.
Use Shared VPC	Centralize security and reduce management and topology complexity by making use of Shared VPC where fit.
Fewer subnets	Group similar applications into fewer, more manageable and larger subnets.
Restrict network configuration	Apply organization policies to 1) skip creation of default network for new projects, 2) restrict shared VPC host projects and subnets, 3) restrict public IP address usage.
Consider quota limitations	Ensure the design scales to your needs by considering limitations on each network component.

TL;DR / Purpose of the slide:

- Review **VPC networks best practices**

Key points:

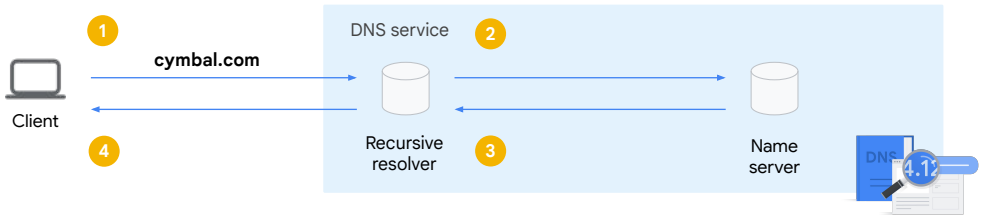
- **Custom mode VPCs** for **production environments**
  - With auto mode, you cannot control IP ranges to **minimise IP ranges** to block when connecting environments, and **prevent overlapping IPs**.
- **Simplify topology**
  - Try to make use of **Shared VPC** where possible to minimize management overhead and overall network design complexity.
  - As said, we will review a few **reference architectures** to discuss this in details.
- **Fewer subnets**
  - With fewer subnets there is less management overhead. A good approach is grouping similar applications into larger subnets.
- **Restrict network configuration** - apply **org policies** on **appropriate folders** to:
  - Control which Shared VPCs projects can use (constraints/compute.restrictSharedVpcHostProjects)
  - Control which subnets part of a Shared VPC projects can use (subnets (constraints/compute.restrictSharedVpcSubnetworks)
  - Control which networks underlying projects can be VPC peered with

    - (constraints/compute.restrictVpcPeering)
- **Quota limitations**
    - **Scalability** is very often an important factor in a network design.
    - **Define** your scalability targets, and consider the **soft/hard limits of every component** in your network design to ensure it can scale as required.
    - **Some examples**:
        - Max VPCs per project
        - Max Shared VPCs per organization
        - Max service projects per Shared VPC
        - Max VPN gateways, and tunnels per gateway
        - Max VPC peered networks
        - Max subnets per VPC
        - Max routes (learned through Cloud Router, and static) per VPC
        - Max FW rules per VPC
        - Max forwarding rules per VPC
        - Max forwarding rules per peering group of VPCs
        - Max VMs per VPC
        - IP range ability to accommodate growth

Probing questions (optional):

- In case a network topology was discussed and drafted by this point, discuss the scaling requirements of each component:
    - See list of examples above
    - Consider limitations on any other component that is used

A simple DNS primer



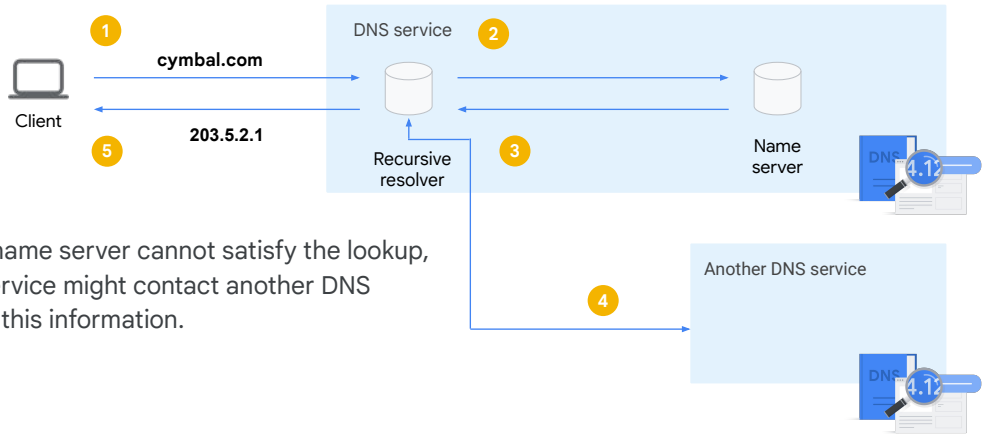
01. A client makes a DNS request to obtain an IP address; the request is sent to a recursive resolver.
02. A recursive resolver requests the IP address from a name server.
03. The name server responds with the IP address.
04. The recursive resolver sends the IP to the client.

Before we talk about Google Cloud, let's quickly review how DNS (Domain Name System) works.

DNS provides a lookup for sites on the internet. You can think of it as a phone book, but instead of using the name of an organization to look up its phone number, you use the name of an organization to find an IP address. A DNS service is provided by your ISP (internet service provider).

For example, suppose a request comes from a client computer to access cymbal.com. To direct the client computer to the cymbal.com site, the internet service provider needs the IP address of cymbal.com. The ISP connects to get this information from its DNS service. The DNS service recursive resolver issues a request to look up the IP address of cymbal.com from one of its name servers. The name server responds with the ISP.

A simple DNS primer



When the name server cannot satisfy the lookup, the DNS service might contact another DNS service for this information.

When the name server cannot satisfy the lookup, the DNS service might contact another DNS service for this information.

Some organizations don't rely on their ISP to provide DNS service, so they create and maintain their own DNS servers. Organizations sometimes do this to limit or customize the information that is returned, or because they can achieve better performance if they use their own DNS servers. Alternately, they can purchase DNS services from another organization.

Obviously, there's a lot more that can be said about DNS and its components, but that's not covered in this course.

Various companies provide DNS services. Google Cloud is one of them.

DNS options

An internal metadata server acts as DNS resolver, and is automatically set up as part of DHCP leases.

Internal Compute Engine DNS

Records are automatically created for a VM's primary internal IP with the following FQDN:

- [INSTANCE_NAME].[ZONE].c.[PROJECT_ID].internal (default)
- [INSTANCE_NAME].c.[PROJECT_ID].internal

Used for resolution within the same project **and** VPC

Cloud DNS

Scalable, reliable (**100% SLA**), and managed authoritative DNS service for public and private records offering

Private: Used for providing a namespace that is only visible inside Google Cloud

Public: Used for providing authoritative DNS resolution to clients on the public internet.

TL;DR / Purpose of the slide:

- Review **DNS options** in Google Cloud

Key points:

- **Metadata server**
  - Every VM instance has a metadata server used for querying instance info, for example: name, ID, startup/shutdown scripts, custom metadata, service account
  - The metadata server also acts as the **DNS resolver** for both **internal and external resolutions** (i.e resolving hostnames in the public internet)
  - It is set on VMs as part of default **DHCP leases**.
  - **Overriding** it is **possible** by **customizing DHCP** configuration (dhclient.conf), however it is **not a common pattern**
- **Internal DNS**
  - **Records** are automatically created for **VM's primary IP** (not Alias IP's), for example:
    - instance1.europe-west1b.c.projectx.internal
  - Used **within the same VPC**
- **Cloud DNS**

    - Managed DNS service. 100% SLA
    - **Public** as an **authoritative DNS** resolution accessible from the public internet
    - **Private** for **DNS resolution** inside a **VPC**
- In the next slides, we will discuss **Cloud DNS Private** in details

Probing questions (optional):

- Would you like to use your own custom DNS names for VM resources, or would you like to use the default internal DNS names? (i.e [instance_name].[zone].c.[project_id].internal)

Private and public DNS zones

Private

- Private zones are used to provide a namespace that is visible only inside the VPC or hybrid network environment.
- For example, an organization would use a private zone for a domain dev.gcp.example.com, which is reachable only from within the company intranet.

Public

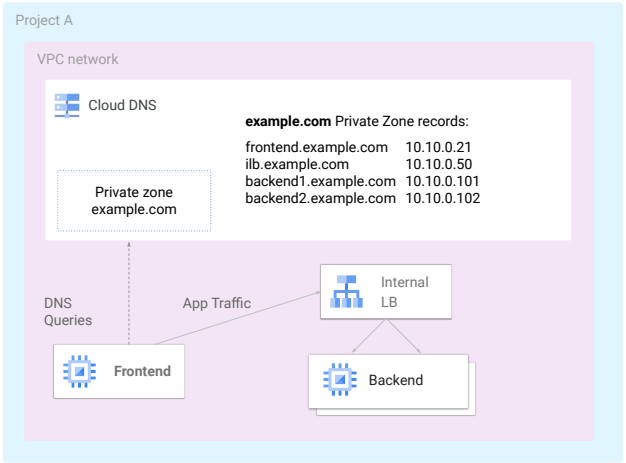
- Public zones are used to provide authoritative DNS resolution to clients on the public internet.
- For example, a business would use a public zone for its external website, cymbal.com, to make the site accessible on the internet.

Private zones are used to provide a namespace that is visible only inside the VPC or hybrid network environment. For example, an organization would use a private zone for a domain dev.gcp.example.com, which is reachable only from within the company intranet.

Public zones are used to provide authoritative DNS resolution to clients on the public internet. For example, a business would use a public zone for its external website, cymbal.com, which is accessible directly from the internet.

Don’t confuse the concept of a public zone with Google Public DNS (8.8.8.8). Google Public DNS is just a public recursive resolver.

Cloud DNS: Private DNS zones



- Internal facing DNS records (for example, VMs, LBs)
- Requests must be submitted through the metadata server
- Can only be queried by authorized VPC networks in the same project, unless DNS peering is configured

TL;DR / Purpose of the slide:

- Review **Cloud DNS private zones**

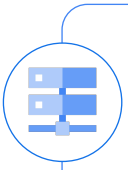
Key points:

- **Private zones** can be configured within **Cloud DNS** to manage **internal facing** records (for example: VMs, LBs)
- All **requests** must be **made to** the **metadata server**
- **VPC** must be in the **same project,** AND **authorized** to use the **private zone**
- For cross project resolution, **DNS peering** needs to be configured.

Probing questions (optional):

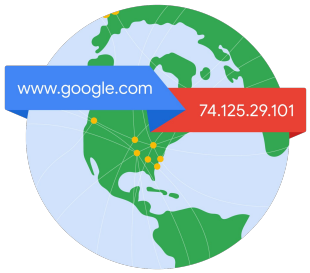
- None

Use Cloud DNS to host DNS zones



Cloud DNS can:

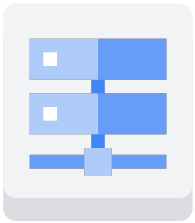
- Create and update millions of DNS records.
- Create and maintain DNS records and other DNS artifacts by using the Google Cloud console, command line, or API.



Cloud DNS lets you create and update millions of DNS records without the burden of managing your own DNS servers and software. Instead, you use a simple user interface, command-line interface, or API. For more information, refer to the Cloud DNS documentation, https://cloud.google.com/dns/docs/.

Introduction to Cloud DNS policies

- Cloud DNS policies provide a flexible way to refine how your organization uses DNS.
- After you create the DNS records and artifacts needed for lookups, create Cloud DNS policies.

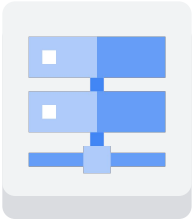


Cloud DNS policies provide a flexible way to define how your organization uses DNS.

After you create the DNS records and artifacts needed for lookups, create Cloud DNS policies.

Supported Cloud DNS policies

- Server policies apply private DNS configuration to a VPC network.
- Response policies enable you to modify the behavior of the DNS resolver by using rules that you define.
- Routing policies: steer traffic based on geolocation or round robin.



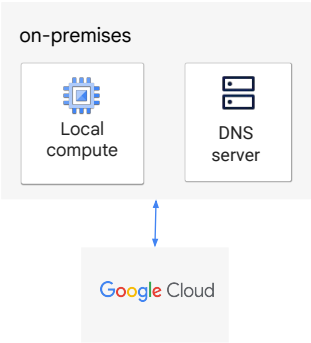
Cloud DNS supports different types of policies:

- Server policies apply private DNS configuration to a VPC network.
- Response policies enable you to modify the behavior of the DNS resolver by using rules that you define.
- Routing policies steer traffic based on geolocation or round robin.

Next, let's look at each of these types of policies.

Server policies

- Use server policies to set up hybrid deployments for DNS resolutions.
- Each VPC network can have one DNS server policy.
- You can set up an **inbound server policy** depending on the direction of DNS resolutions. For example, if you want on-premises workloads to resolve names on Google Cloud, set up an inbound server policy.
- For workloads that use an on-premises DNS resolver, use an **outbound server policy** to set up DNS forwarding zones.



Use server policies to set up hybrid deployments for DNS resolution. You can set up an inbound server policy depending on the direction of the DNS resolutions. If your workloads plan to use an on-premises DNS resolver, you can set up DNS forwarding zones by using an outbound server policy. If you want your on-premises workloads to resolve names on Google Cloud, you can set up an inbound server policy.

You can configure one DNS server policy for each Virtual Private Cloud (VPC) network. The policy can specify inbound DNS forwarding, outbound DNS forwarding, or both. In this section, inbound server policy refers to a policy that permits inbound DNS forwarding. Outbound server policy refers to one possible method for implementing outbound DNS forwarding. If a policy implements the features of both, it can be an inbound server policy and an outbound server policy.

DNS server policies are not available for legacy networks. DNS server policies require VPC networks.

For detailed information about server policies, see Server policies overview in the Google Cloud documentation. To configure and apply DNS server policies, see Apply Cloud DNS server policies in the Google Cloud documentation.

Response policies

- A response policy:
  - Is a Cloud DNS private zone concept that contains rules instead of records.
  - Lets you introduce customized rules in DNS servers within your network that the DNS resolver consults during lookups.
- If a rule in the response policy affects the incoming query, It's processed (otherwise, the lookup proceeds normally).
- The rules enable you to return modified results to DNS clients.

A response policy is a Cloud DNS private zone concept that contains rules instead of records. These rules can be used to achieve effects similar to the DNS response policy zone (RPZ) draft concept. In other words, you can use response policies to create a DNS firewall by returning modified DNS results to clients. For example, you can use response policies to block access to specified HTTP servers.

The response policy feature lets you introduce customized rules in DNS servers within your network that the DNS resolver consults during lookups.

If a rule in the response policy affects the incoming query, it's processed. Otherwise, the lookup proceeds normally. For more information, see *Manage response policies and rules* in the Google Cloud documentation.

A response policy is different from an RPZ (response policy zone). An RPZ is an otherwise normal DNS zone with specially formatted data that causes compatible resolvers to do special things. Response policies are not DNS zones and are managed separately in the API. To create and modify response policies in Cloud DNS, use the ResponsePolicies API. Response policies are separate from ManagedZones and cannot be managed by using either the ManagedZones API or the RRSet API.

DNS response policy rules

01

Directing specific names to restricted Google API VIP addresses

02

Directing all names (wildcard) except some to restricted Google API VIP addresses

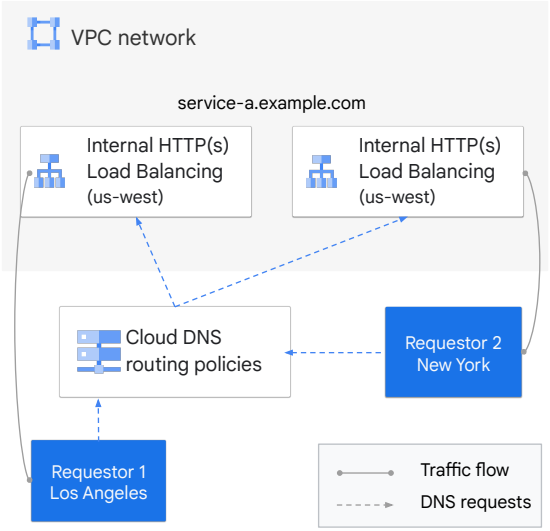
03

BlackHole certain DNS queries to known bad domain names

@trainer: please use this link for context regarding this slide:
Common use cases

Routing policies

- DNS routing policies steer your traffic based on specific criteria.
- Google Cloud supports two types of DNS routing policies:
  - Weighted round robin: lets you specify different weights per DNS target.
  - Geolocation: lets you map the traffic that originates from Google Cloud regions to specific DNS targets.



DNS routing policies let you steer your traffic based on specific criteria. Google Cloud supports two types of DNS routing policies: weighted round robin and geolocation.

A weighted round robin routing policy lets you specify different weights per DNS target, and Cloud DNS ensures that your traffic is distributed according to the weights. You can use this policy to support manual active-active or active-passive configurations. You can also split traffic between production and experimental versions of software.

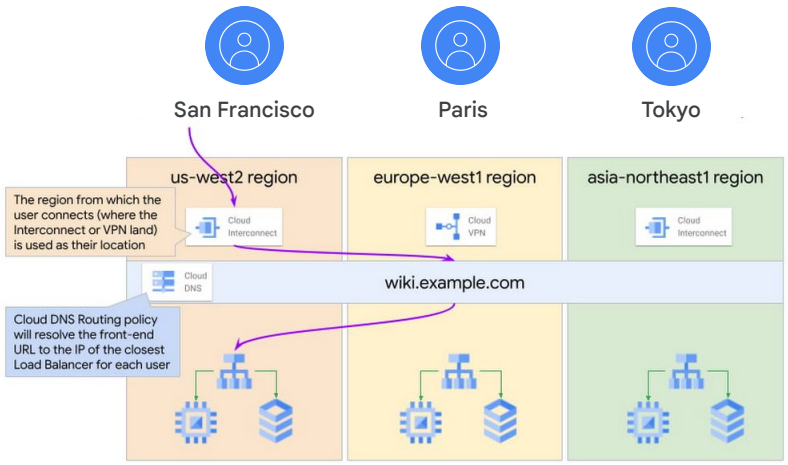
A geolocation routing policy lets you map traffic originating from source geographies (Google Cloud regions) to specific DNS targets. Use this policy to distribute incoming requests to different service instances based on the traffic's origin. You can use this feature with the internet, with external traffic, or with traffic originating within Google Cloud and bound for internal load balancers. Google Cloud uses the region where queries enter Google Cloud as the source geography. Next, you will implement a geolocation routing policy as part of a lab exercise. An example is shown on the screen; routing policies use geolocation to route requests to the closest load balancer.

In a lab exercise, you will configure a routing policy that uses geolocation.

To create, edit, or delete DNS routing policies, see Manage DNS routing policies in the Google Cloud documentation.

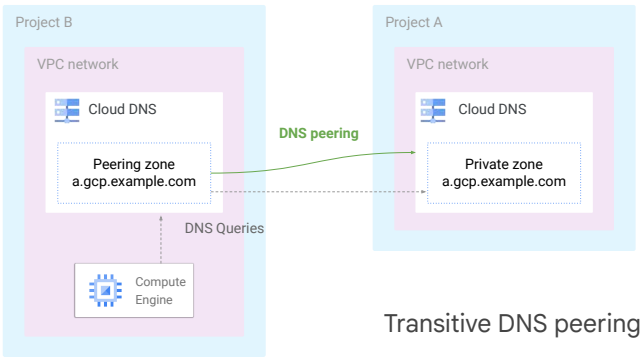
DNS Routing Policies and multi-regional apps

- Configure the DNS 'A' record for multiple destinations
- Set the DNS routing policy to GEO



https://cloud.google.com/blog/products/networking/dns-based-traffic-routing-for-global-application-deployment

Cloud DNS: DNS peering



- Allows DNS queries to be sent from one zone's namespace to another VPC
- Does not require connectivity between the VPCs (i.e. VPC Peering)
- One way only

Transitive DNS peering supported through a single hop only.

TL;DR / Purpose of the slide:

- Discuss how **DNS peering** works

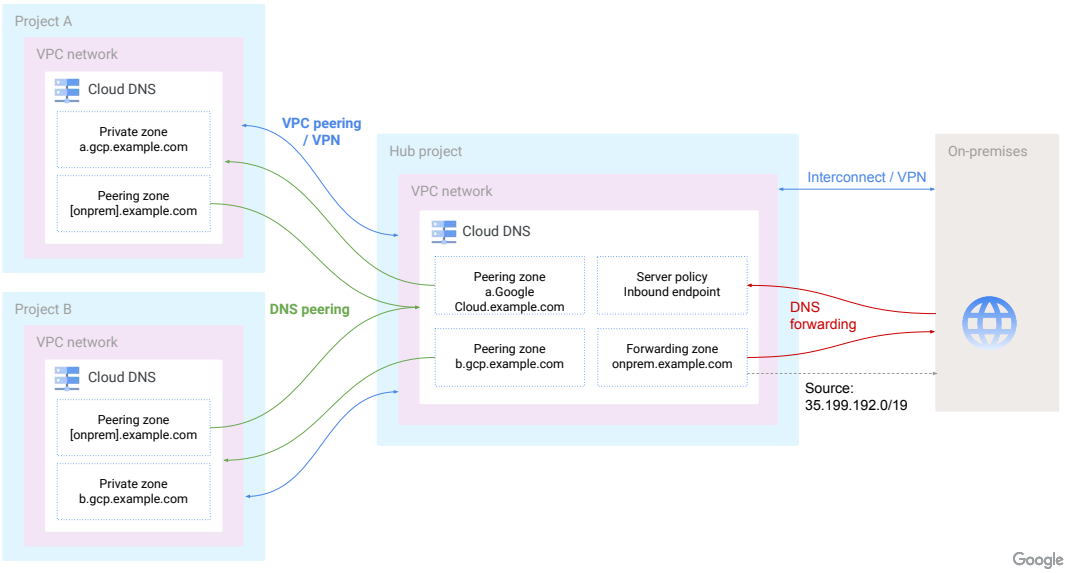
Key points:

- **Why**
  - DNS peering allows querying **private zones** in **different VPC networks**
  - Combined with **DNS forwarding**, it allows **DNS between spoke and on-premises** in a hub-and-spoke model.
- **What**
  - **One-way**. Two peerings are needed for bi-directional.
  - **Network connectivity (**for example. VPC peering**)** is not needed
  - **Recursion depth** of 2.
    - This means transitive **DNS peering across 2 DNS peering hops** is **supported**.
    - This is needed in a **hub-and-spoke** model to allow spokes query each other's private zones without creating a full-mesh peering topology. We will show an **example very soon.**

Probing questions (optional):

- None

Cloud DNS: Hub-and-spoke model



TL;DR / Purpose of the slide:

- Review a **hub-and-spoke model DNS topology** based on **Cloud DNS**

Key points:

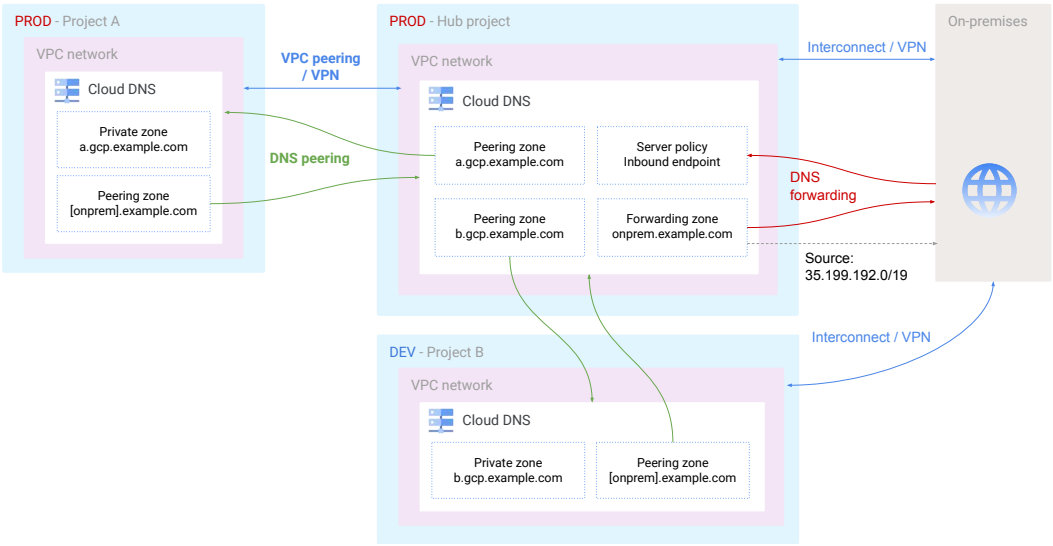
- Let's take a look at how we can use **Cloud DNS** to answer **DNS requirements** in a **classical hub-and-spoke model**.
- **Connectivity**
  - **Hub to on-premise**: VPN or Interconnect
  - **Spokes to hub**: VPC peering or VPN
- **DNS: Google Cloud to on-premises**
  - The **hub** project is configured with a **forwarding zone** to on-premises. Every request to ***.onprem.example.com** is forwarded to on-premises DNS resolvers.
  - Each **spoke** has a **peering zone** for example.com to the hub project's VPC. This allows spokes to **use the forwarding zone** configured in the hub project, and ultimately **resolve on-premises hostnames.**
- **DNS**: **On-premises to Google Cloud**
  - The hub project is configured with an **inbound server policy**. **On-premises** DNS servers are configured with DNS forwarding to **forward all queries for *.gcp.example.com** to the **inbound forwarder IP addresses**.

    - This would only **allow resolving records** in the **hub project**, but **not the spokes.** To solve that, we will configure a **peering zone** in the hub project for the **private zone of each spoke.**
- **DNS**: **Google Cloud to Google Cloud**
    - It might be the case **spokes** need to **resolve hostnames** in each other.
    - **DNS peering** is **transitive across 2 peering hops**, which means a VM in project A can resolve a VM in project B.
    - **Tenants use case:** if projects are tenants and DNS resolution between them is not desired, peering zones can be configured to resolve only on-premises hostnames (onprem.example.com).

Probing questions (optional):

- None

Cloud DNS: prod & non-prod (I)



TL;DR / Purpose of the slide:

- Adding environments (prod, dev) to **hub-and-spoke model DNS topology**

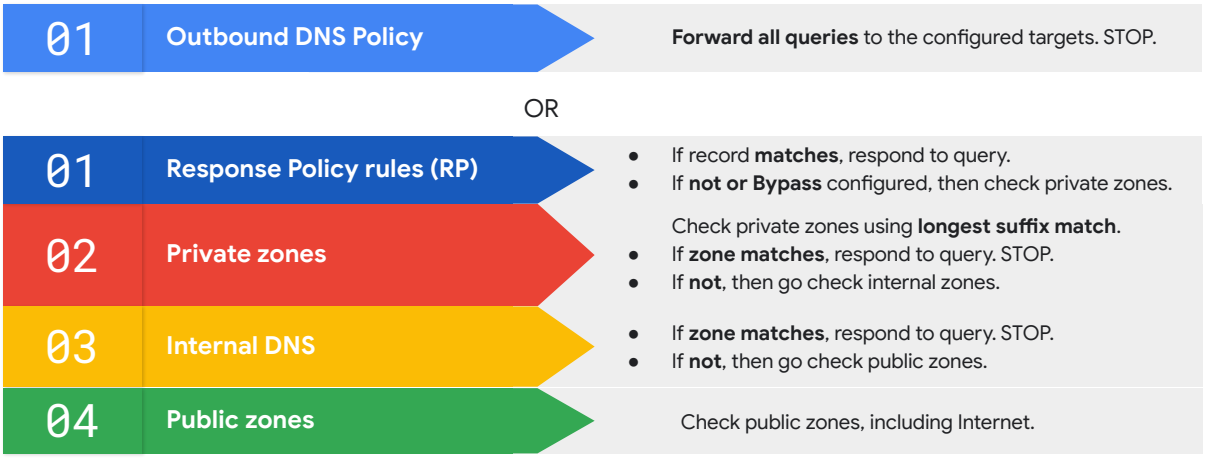
Key points:

- Only one environment can configure a **forwarding zone** to on-premises.
- **DNS peering** between environments.
- Each environment has its own Interconnect to on-premises.
- No network connectivity required between environments.
- Project B could be Hub project for DEV environment, diagram simplified.

Probing questions (optional):

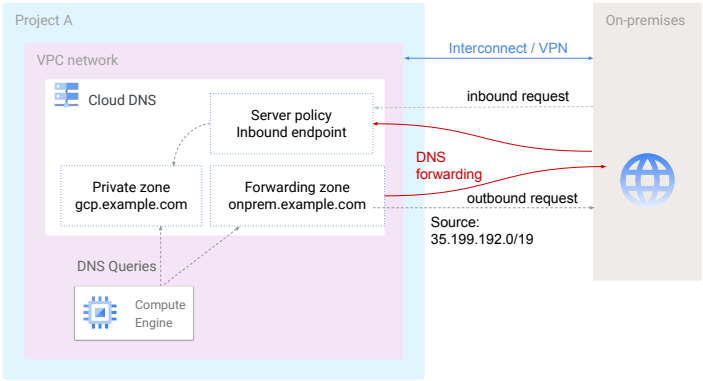
- None

DNS resolution order



@trainer: please use this link for context regarding this slide:
Name resolution order

Cloud DNS: DNS forwarding



TL;DR / Purpose of the slide:

- Discuss how **DNS forwarding** works (continuation of previous slide)

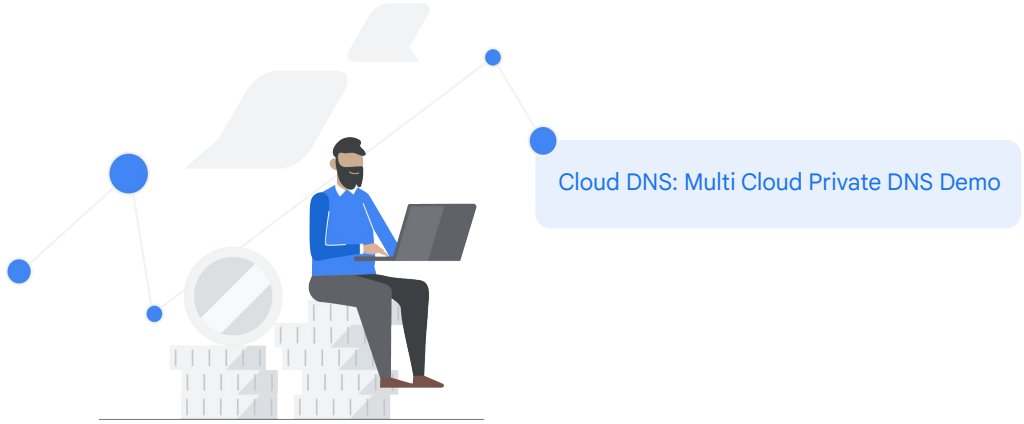
Key points:

- DNS forwarding allows querying to/from on-premises DNS servers

Probing questions (optional):

- None

Cloud DNS: Multi Cloud Private DNS Demo



The Advanced Networking Demo playlist has many helpful videos for PCNE study

@trainer: please click on the words ‘Cloud DNS Multi Cloud Private DNS Demo’ on screen to play the video.

You have now completed the Google Cloud Networks: Fundamentals and Knowledge Assessment module.