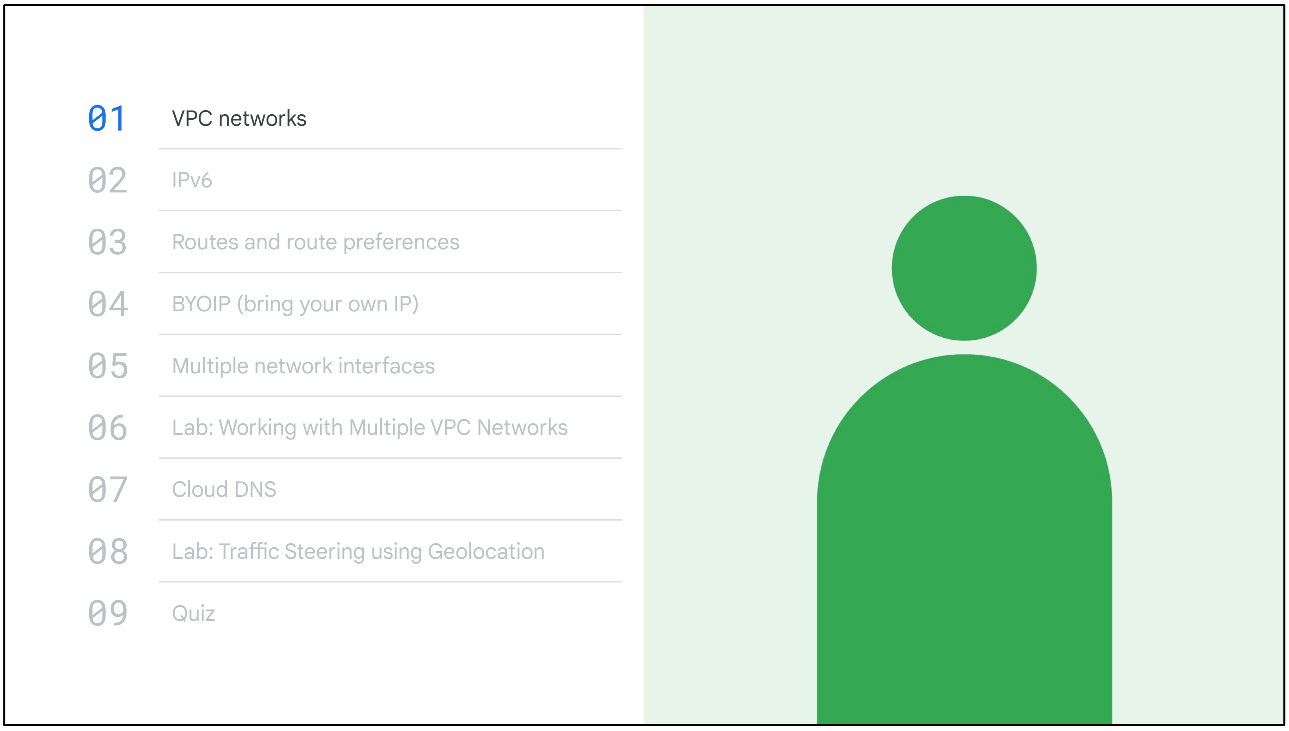
Networking in Google Cloud

Module 1: Google Cloud VPC (Virtual Private Cloud) Networking Fundamentals

Welcome to the Google Cloud VPC Networking Fundamentals module. This is the first module of the Networking in Google Cloud: Defining and Implementing Networks course.



01 VPC networks
02 IPv6
03 Routes and route preferences
04 BYOIP (bring your own IP)
05 Multiple network interfaces
06 Lab: Working with Multiple VPC Networks
07 Cloud DNS
08 Lab: Traffic Steering using Geolocation
09 Quiz

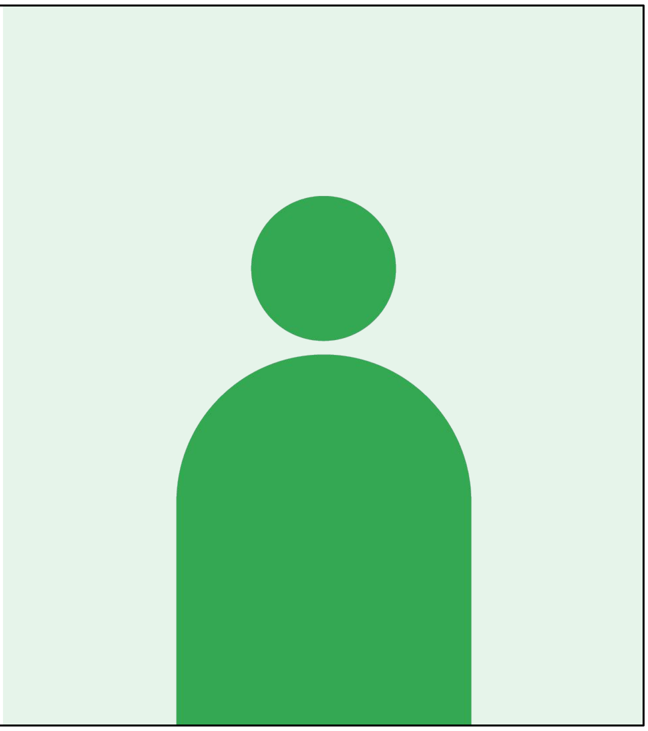
In this module, we will cover the topics listed on the screen.

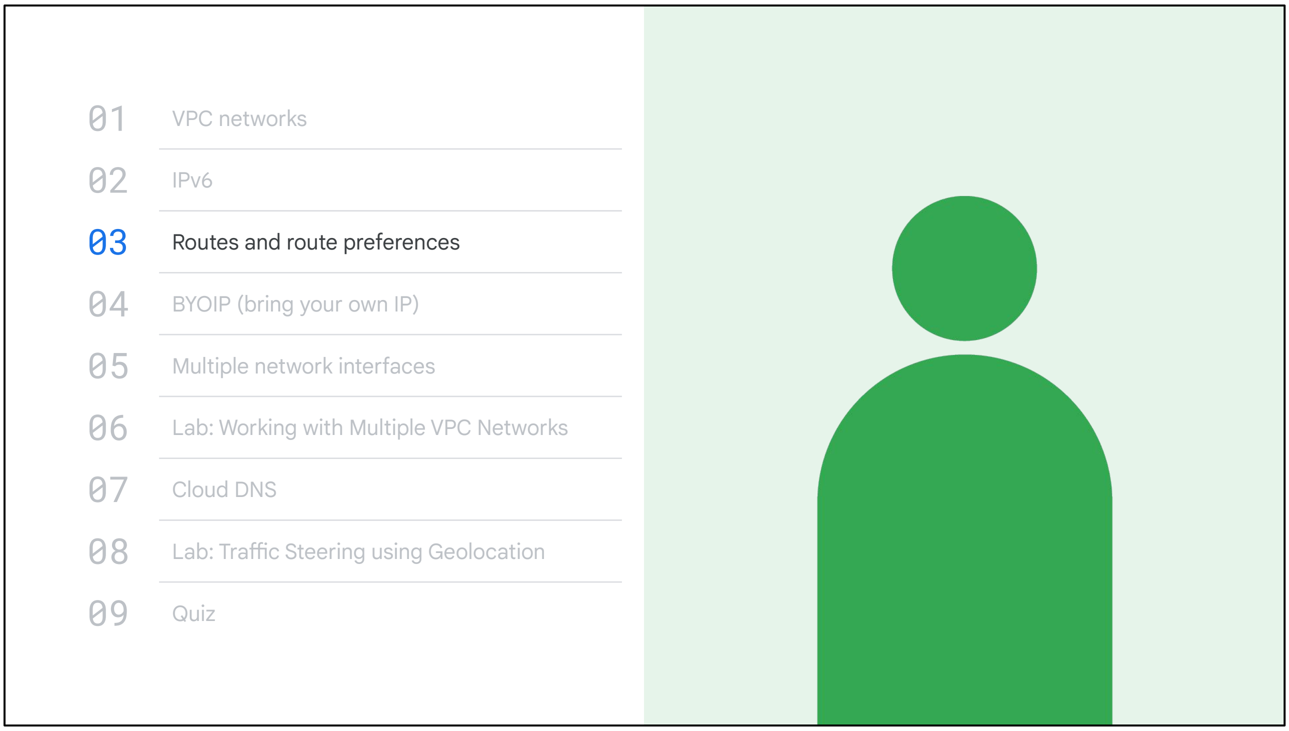
We will begin with a brief overview of VPC networks. After that, we will discuss how you can use IPv6 addressing within Google Cloud. We will also cover routes and route preferences, BYOIP (or bring your own IP address), multiple network interfaces, and Cloud DNS.

You will apply what you have learned in two lab exercises - Working with Multiple VPC Networks and Traffic Steering using Geolocation.

At the end of the module, there will be a short quiz. Now, let’s get started with the VPC networks.

01 VPC networks
02 IPv6
03 Routes and route preferences
04 BYOIP (bring your own IP)
05 Multiple network interfaces
06 Lab: Working with Multiple VPC Networks
07 Cloud DNS
08 Lab: Traffic Steering using Geolocation
09 Quiz





01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

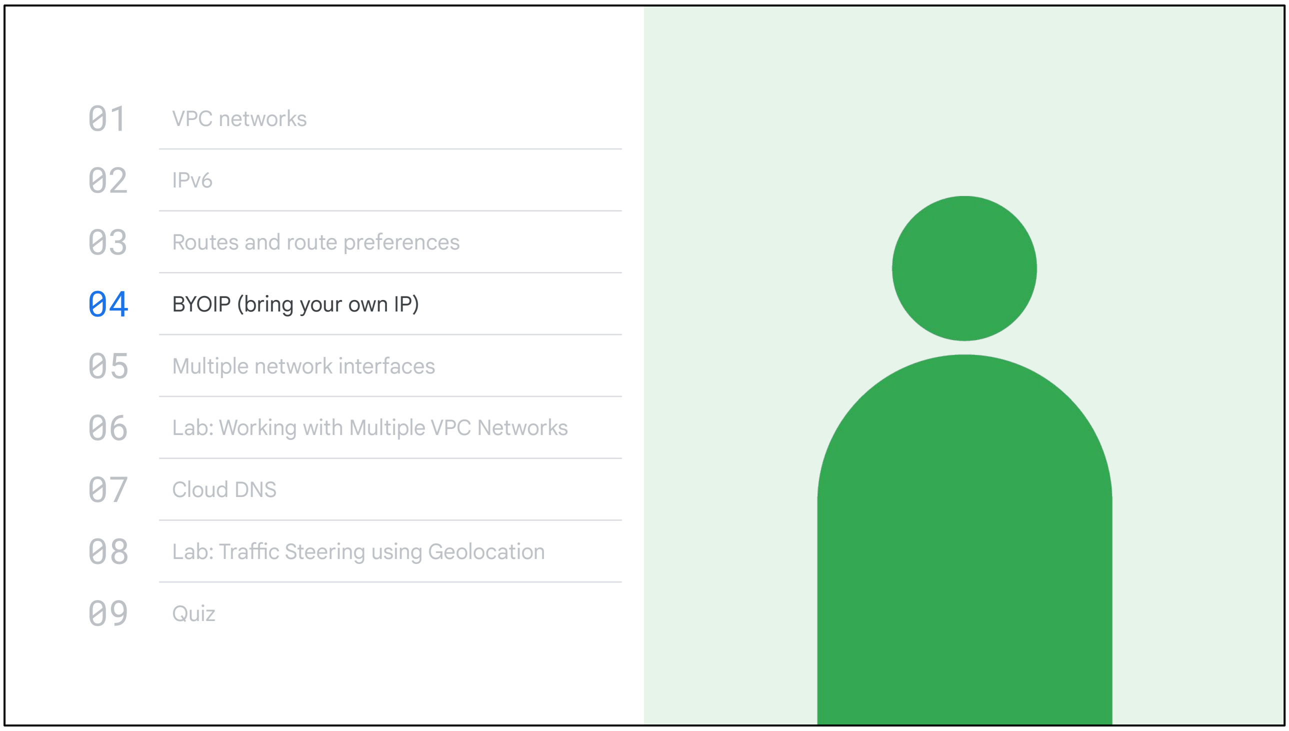
05 Multiple network interfaces

06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

08 Lab: Traffic Steering using Geolocation

09 Quiz



01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

05 Multiple network interfaces

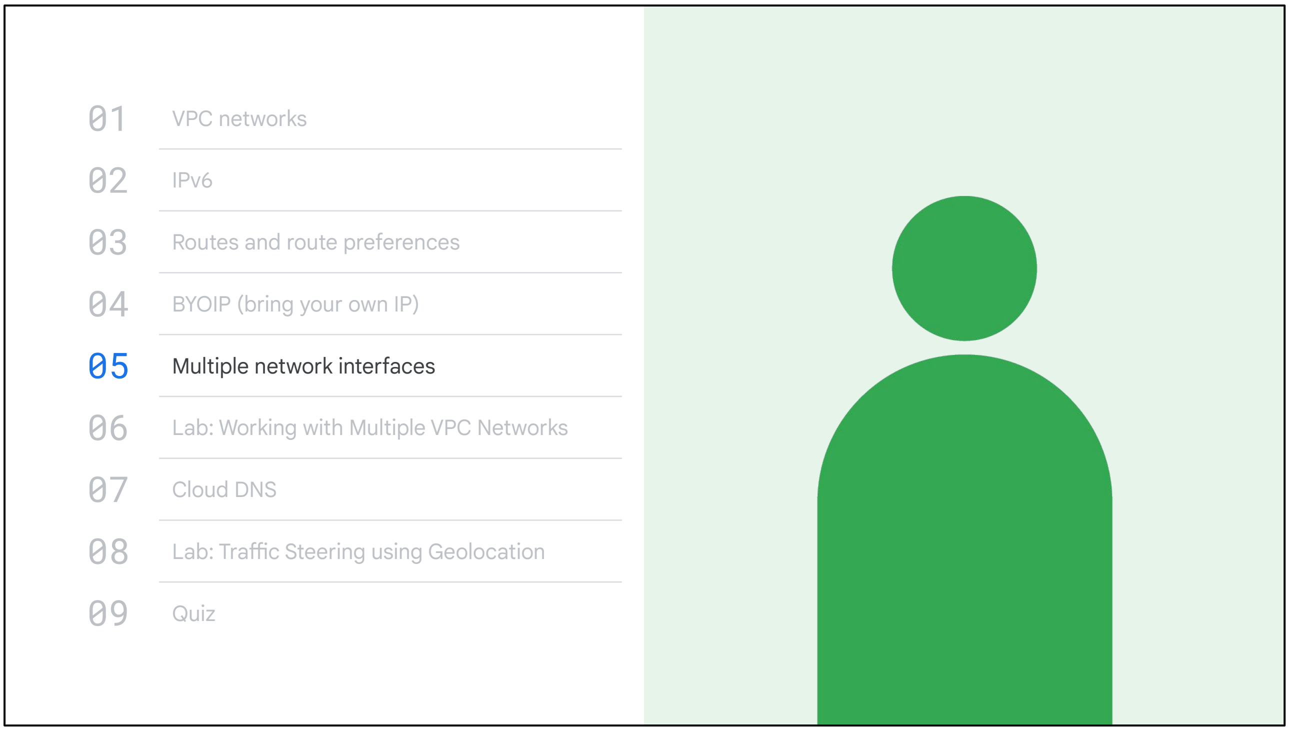
06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

08 Lab: Traffic Steering using Geolocation

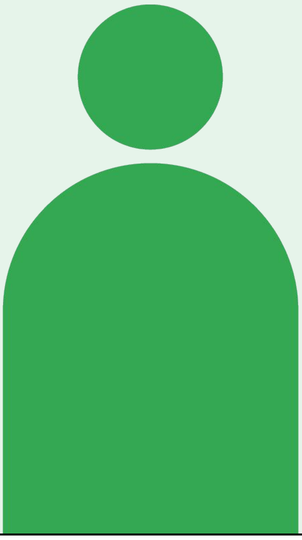
09 Quiz

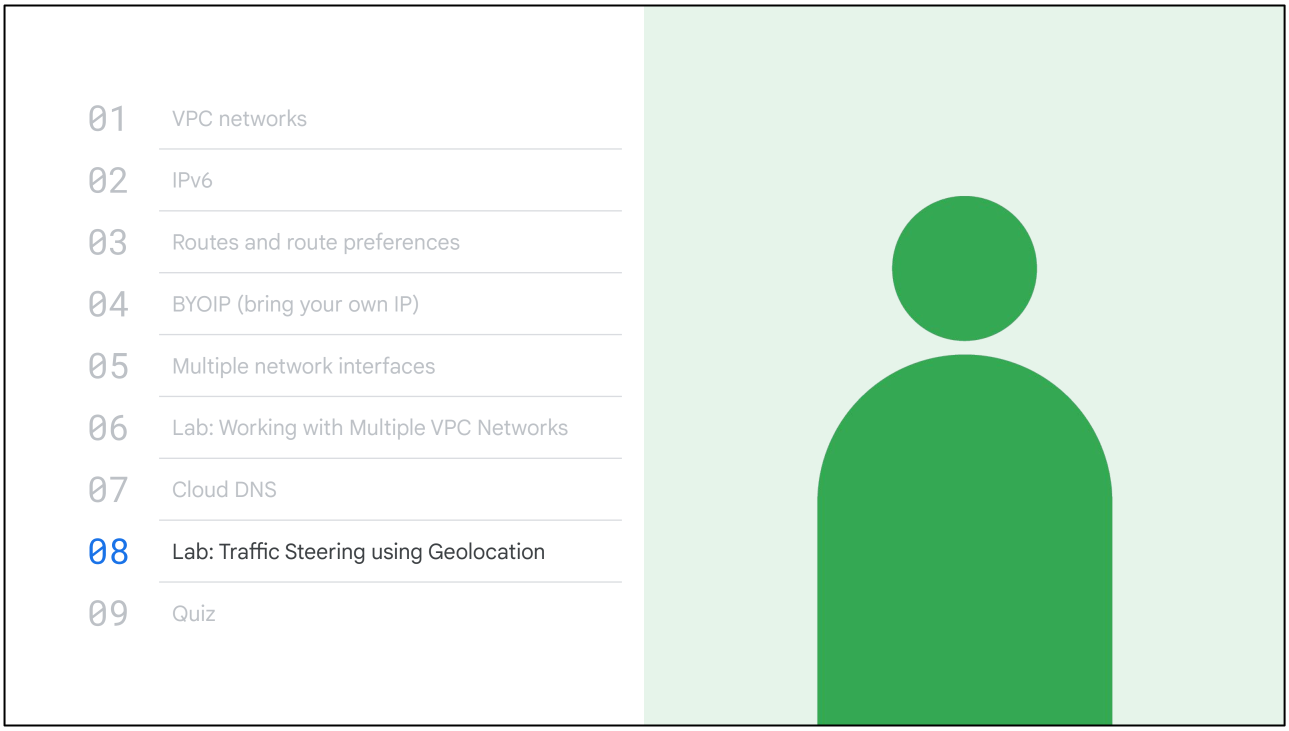
.



01 VPC networks
02 IPv6
03 Routes and route preferences
04 BYOIP (bring your own IP)
05 Multiple network interfaces
06 Lab: Working with Multiple VPC Networks
07 Cloud DNS
08 Lab: Traffic Steering using Geolocation
09 Quiz

01 VPC networks
02 IPv6
03 Routes and route preferences
04 BYOIP (bring your own IP)
05 Multiple network interfaces
06 Lab: Working with Multiple VPC Networks
07 **Cloud DNS**
08 Lab: Traffic Steering using Geolocation
09 Quiz





01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

05 Multiple network interfaces

06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

08 Lab: Traffic Steering using Geolocation

09 Quiz

01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

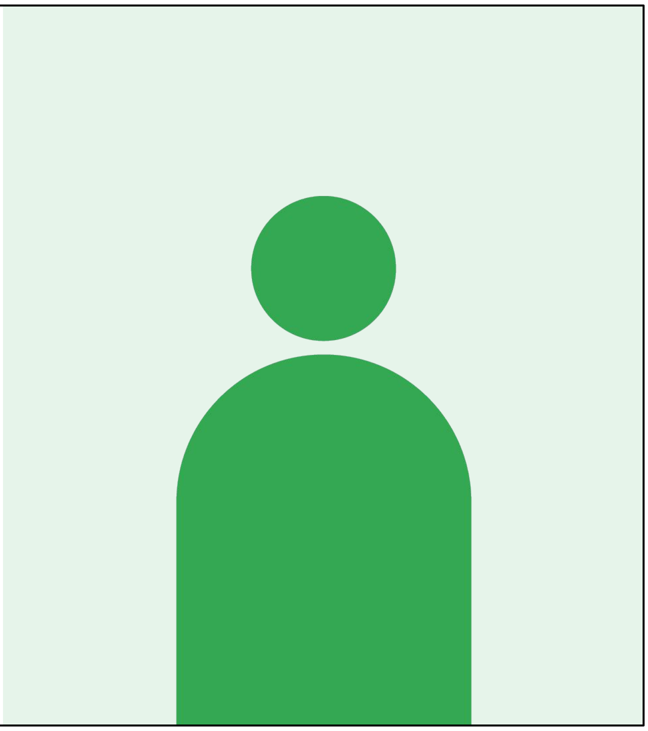
05 Multiple network interfaces

06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

08 Lab: Traffic Steering using Geolocation

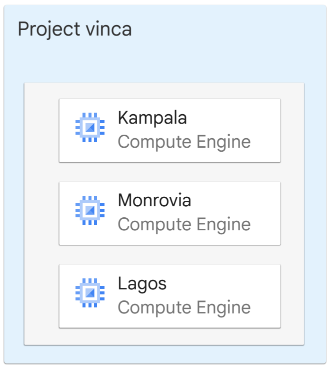
09 Quiz



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.

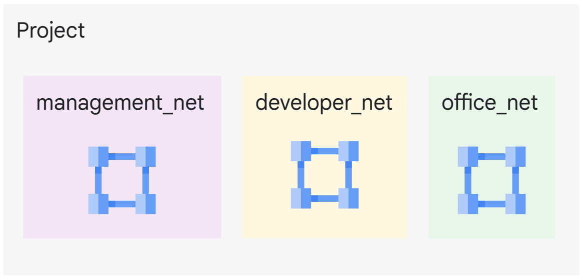
By default, every network has routes that let instances in a network send traffic directly to each other, even across subnets. In addition, every network has a default route that directs packets to destinations that are outside the network. Although these routes cover most of your normal routing needs,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

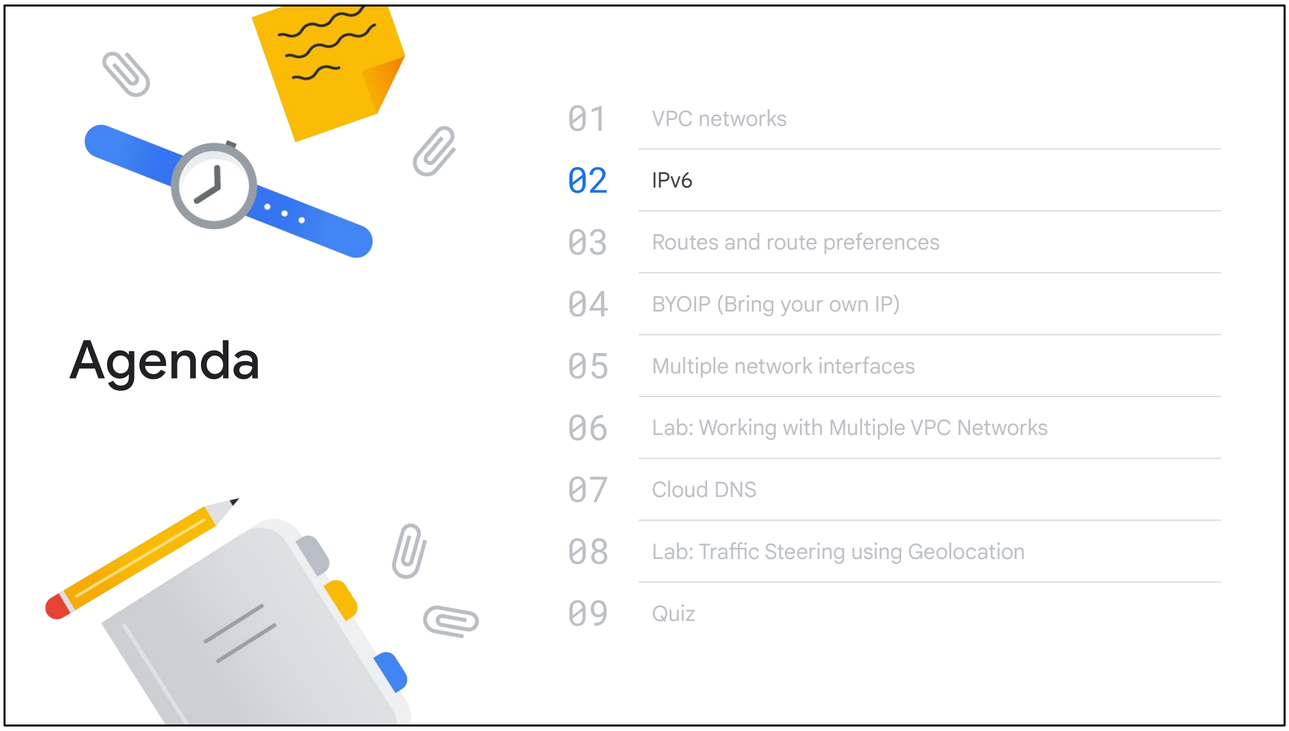
VPC networks

- 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.



Agenda

01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (Bring your own IP)

05 Multiple network interfaces

06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

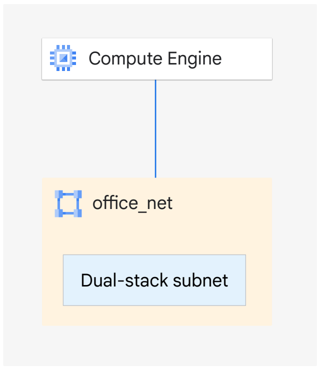
08 Lab: Traffic Steering using Geolocation

09 Quiz

Next, let's discuss how to use IPv6 addressing in Google Cloud.

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.



VPC networks now support IPv6 addresses.

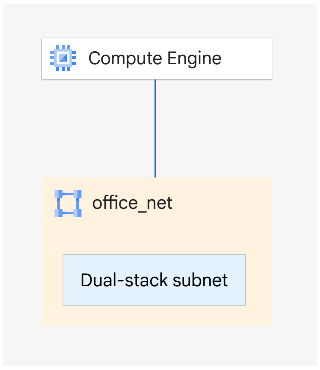
Support for IPv6 addresses can vary per subnet. To support IPv6, Google Cloud has introduced the concept of a subnet stack. The subnet stack defines the type of address that can be assigned to objects in the subnet.

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.

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 and are also routable on the internet Connected VMs inherit the IPv6 access type from the subnet.

External IPv6 addresses can be used for VM to VM communication 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

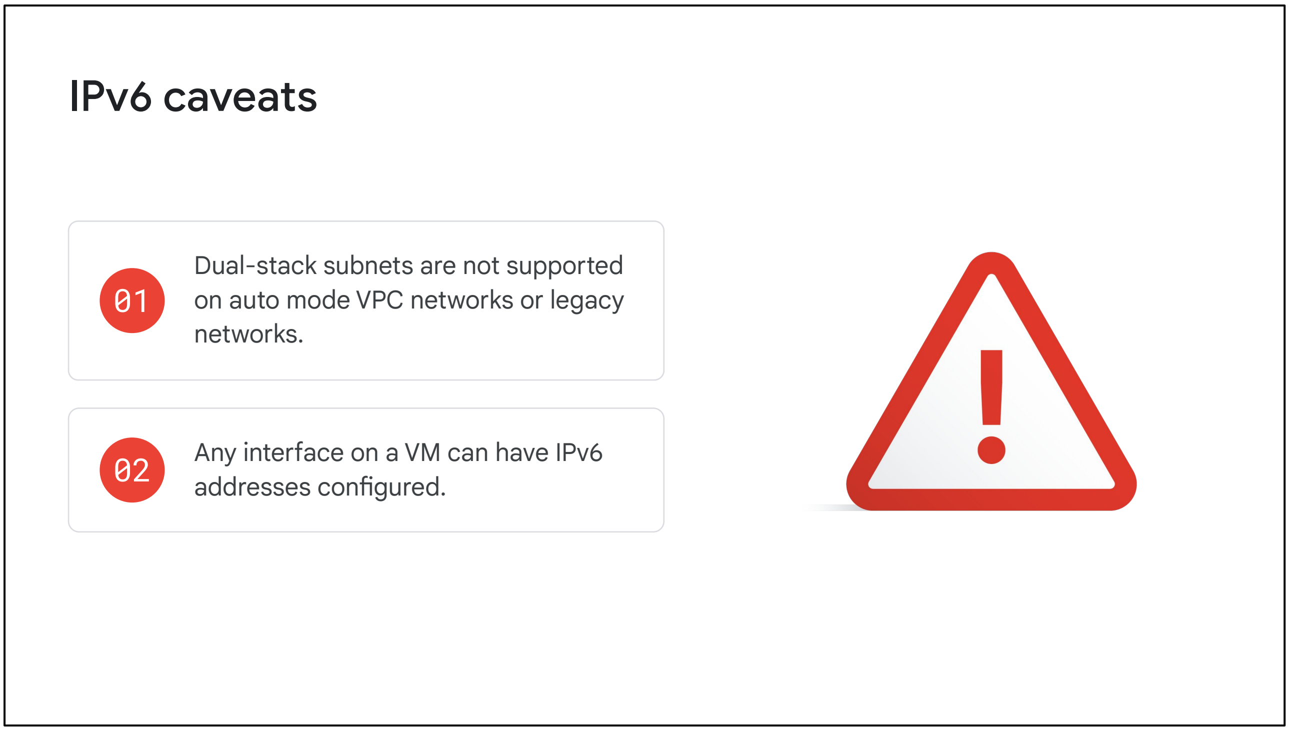
- To enable internal IPv6 on a subnet, you must first assign an internal IPv6 range on the VPC network.
- A /48 ULA range from within fd20::/20 is assigned to the network.
  - All internal IPv6 subnet ranges in the network are assigned from this /48 range.
  - The /48 range can be automatically assigned,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

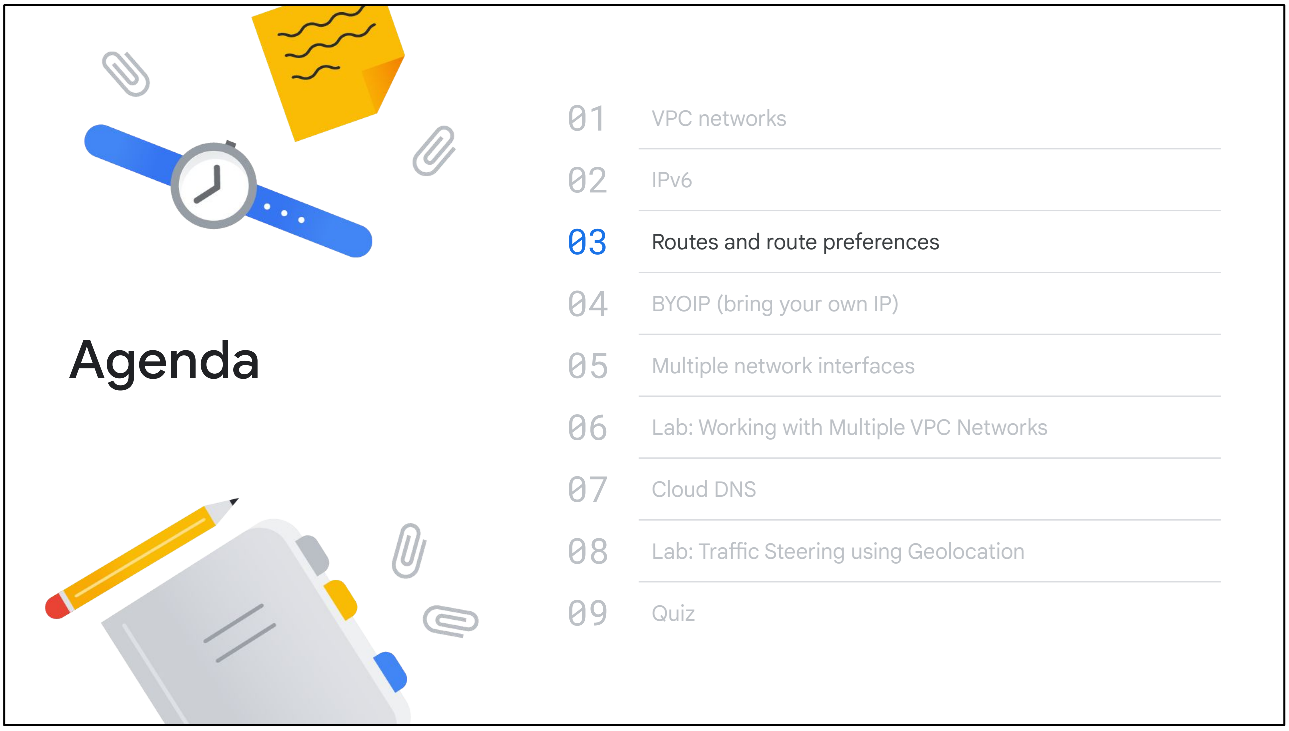
02 Any interface on a VM can have IPv6 addresses configured.

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.



Agenda

01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

05 Multiple network interfaces

06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

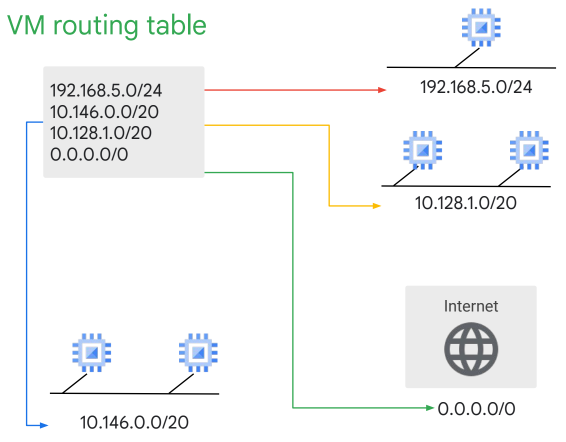
08 Lab: Traffic Steering using Geolocation

09 Quiz

Next, let’s talk about routes and route preferences. We will begin with a brief overview of routes and route types.

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.



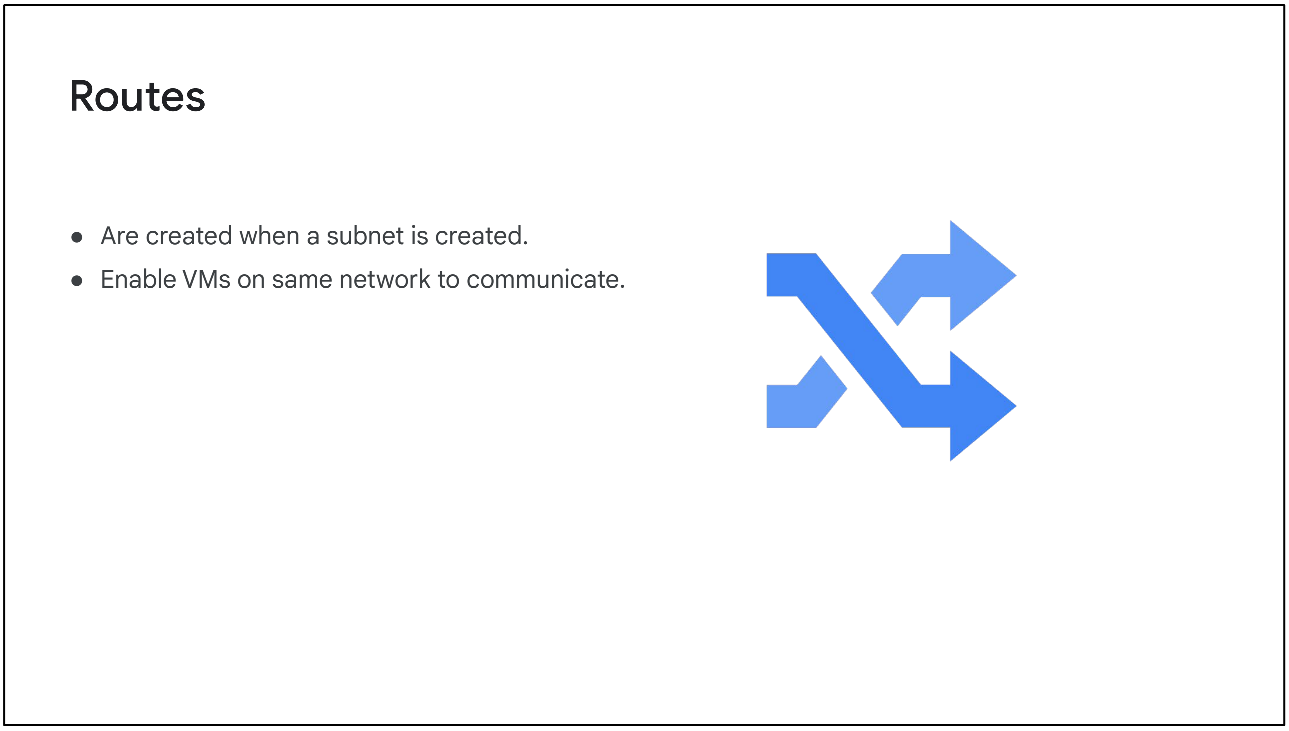
Routes define the paths that network traffic takes from a virtual machine (VM) instance to other destinations. These destinations can be inside your Google Cloud Virtual Private Cloud (VPC) network (for example, in another VM) or outside it.

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.

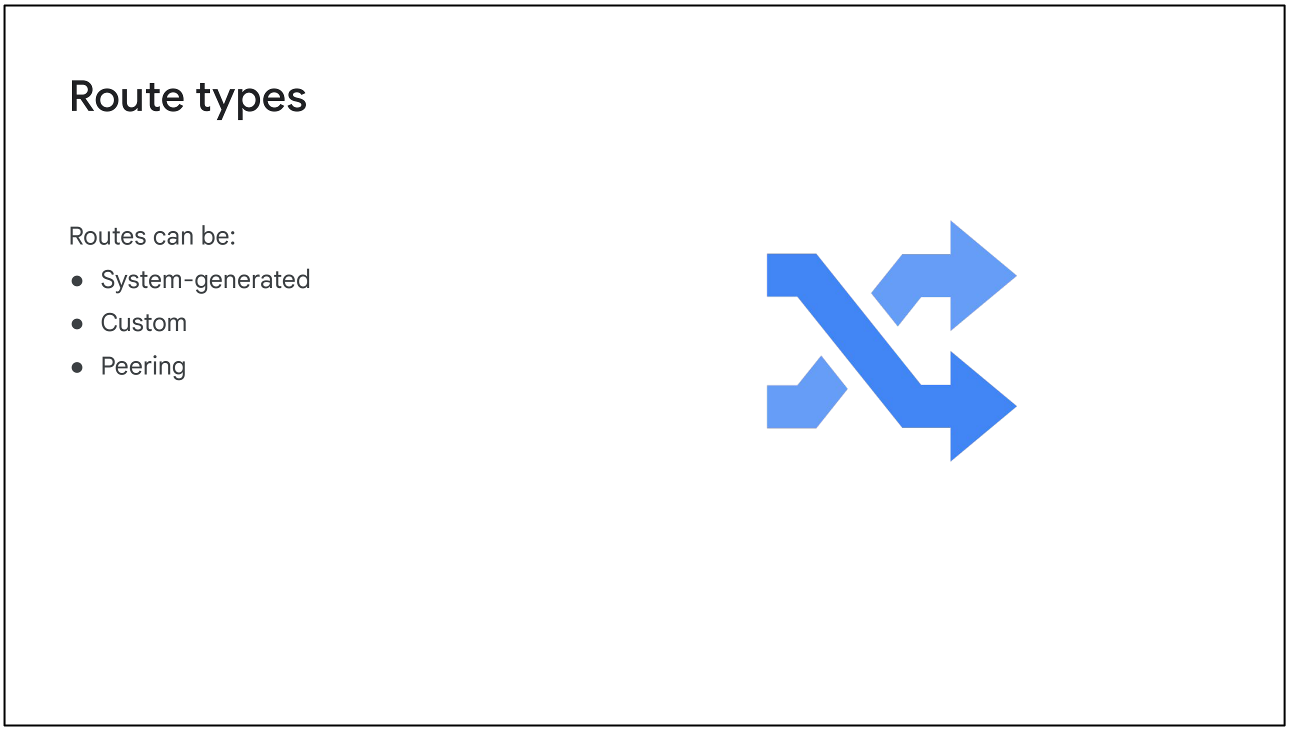
This slide shows a simplified routing table.



Routes

- Are created when a subnet is created.
- Enable VMs on same network to communicate.

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.

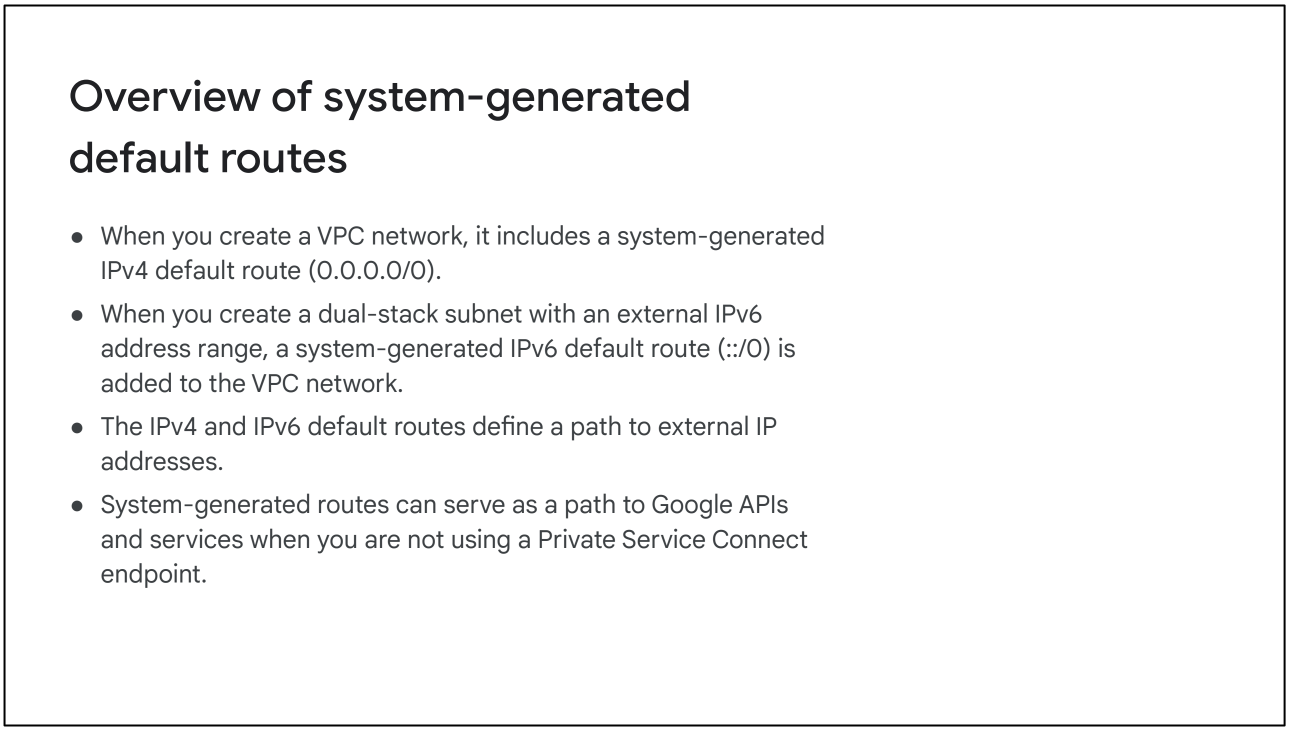


Route types

Routes can be:

- System-generated
- Custom
- Peering

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

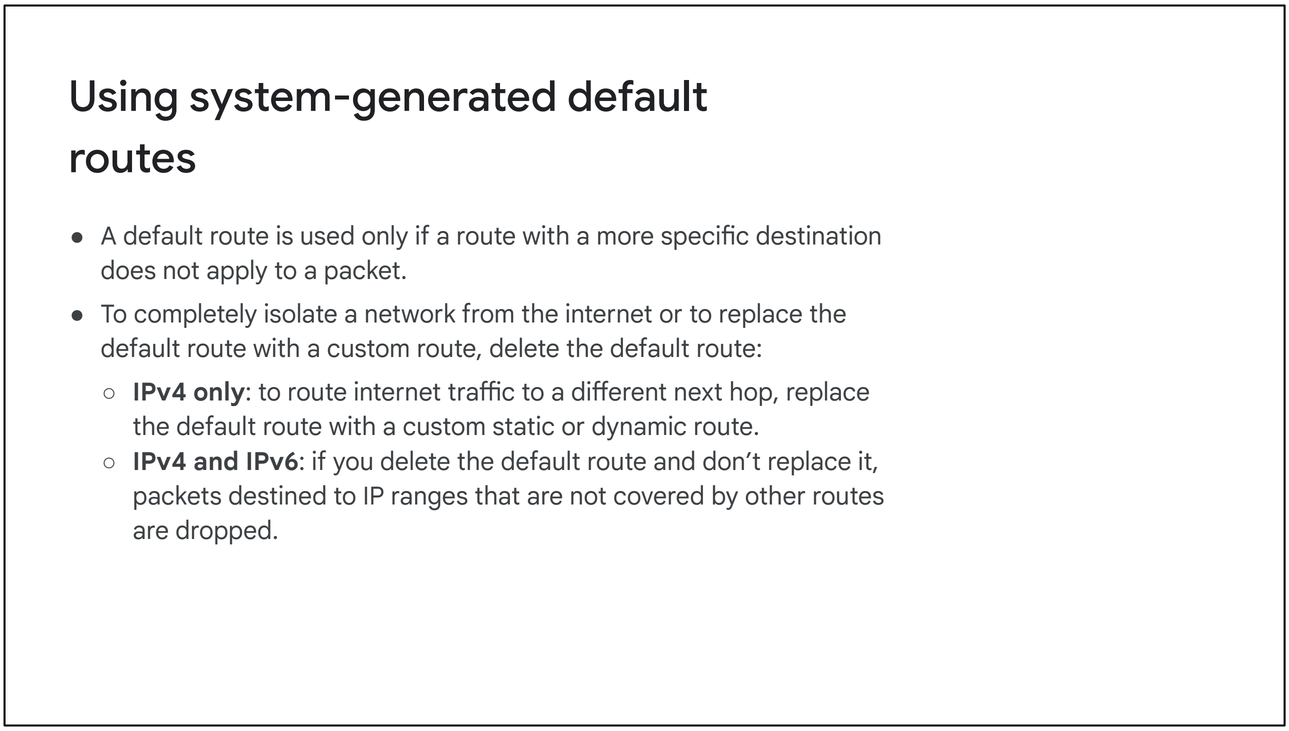
- When you create a VPC network, it includes a system-generated IPv4 default route (0.0.0.0/0).
- When you create a dual-stack subnet with an external IPv6 address range, a system-generated IPv6 default route (::/0) is added to the VPC network.
- The IPv4 and IPv6 default routes define a path to external 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

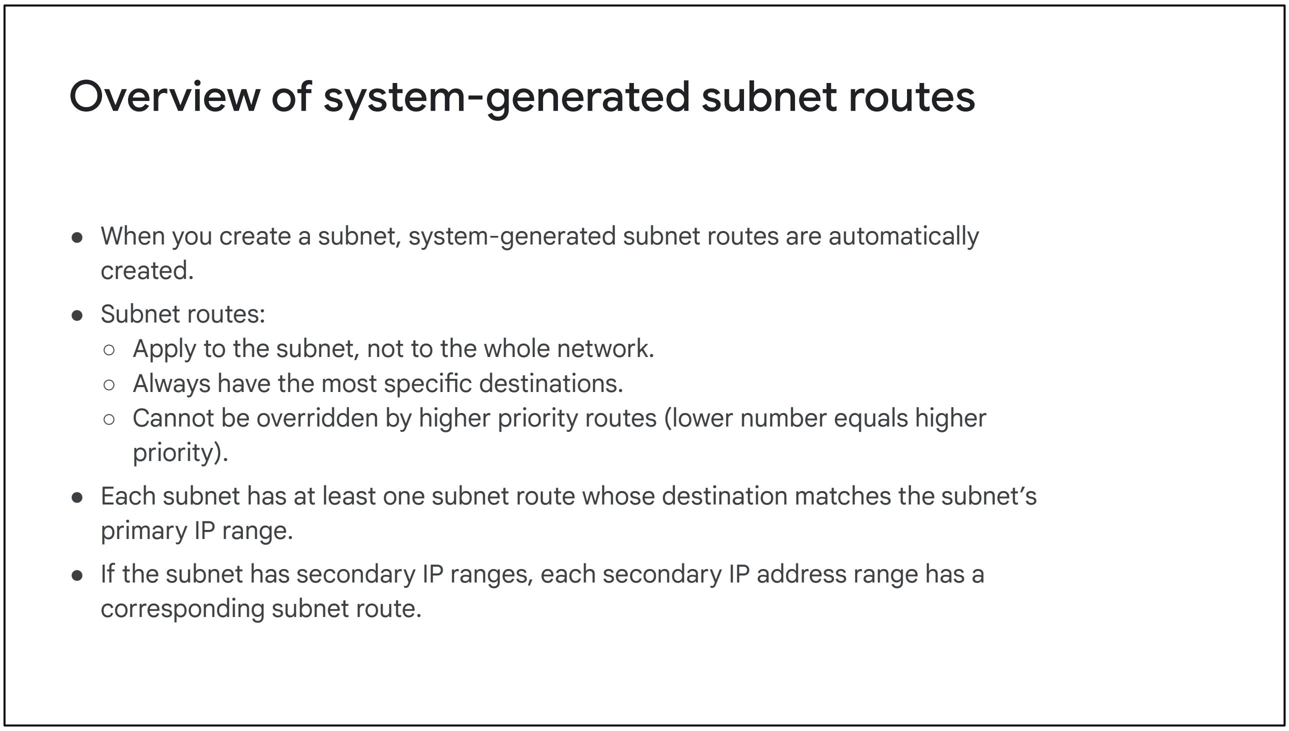
- A default route is used only if a route with a more specific destination does not apply to a packet.
- To completely isolate a network from the internet or to replace the default route with a custom route, delete the default route:
  - **IPv4 only**: to route internet traffic to a different next hop, 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:
  - Quicker routing performance (lower processing overhead).
  - More security (no route advertisement).
- Limitations:
  - Cannot point to a VLAN attachment.
  - Require more maintenance, because routes are not dynamically updated.

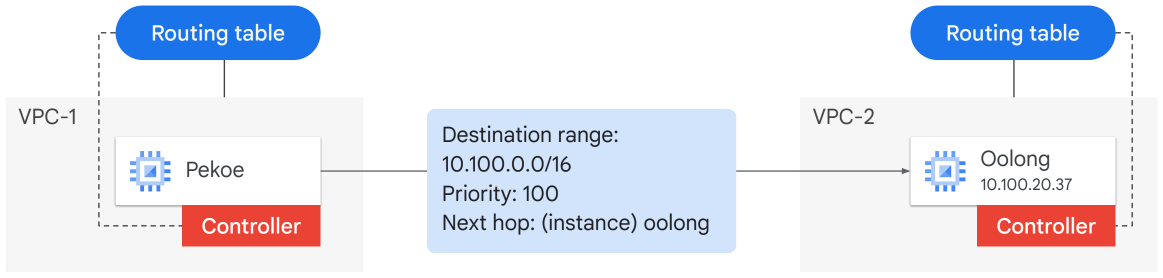
Custom static routes forward packets to a static route next hop and are useful for small, stable topologies.

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.

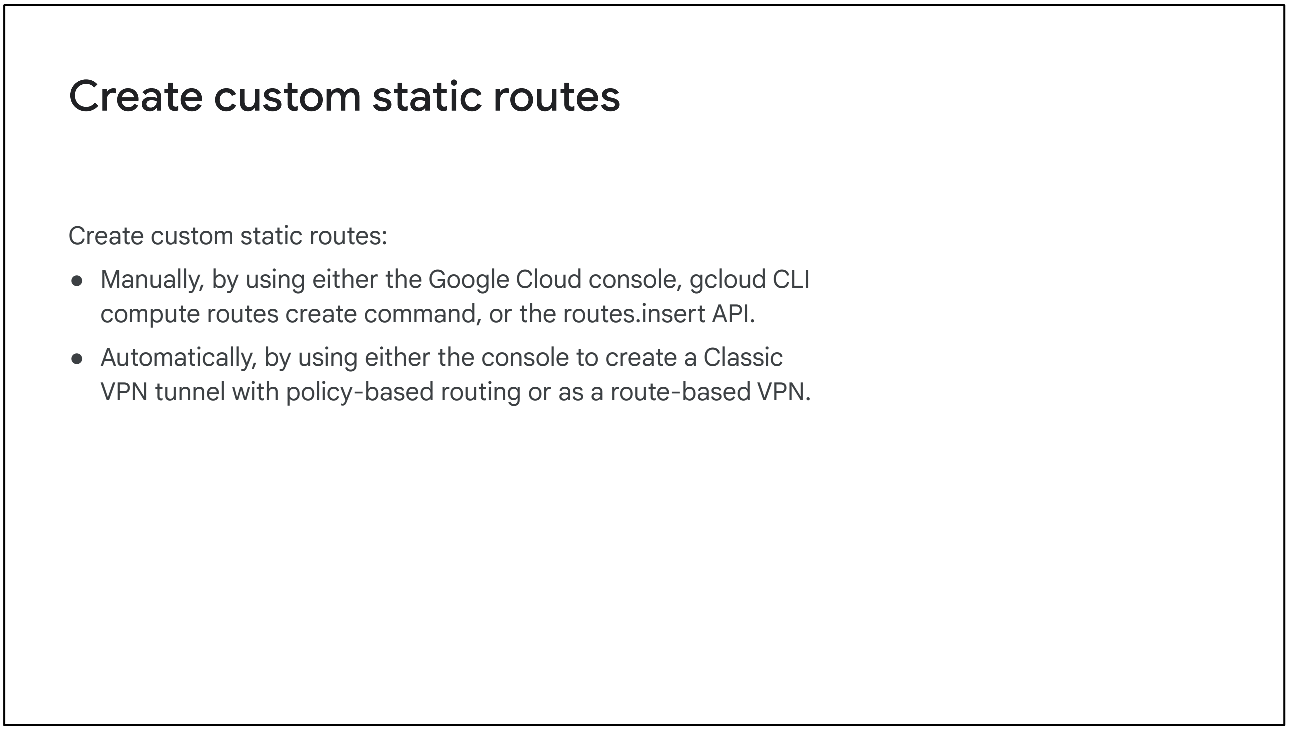
Custom static routes

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



Destination range:
10.100.0.0/16
Priority: 100
Next hop: (instance) oolong

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.

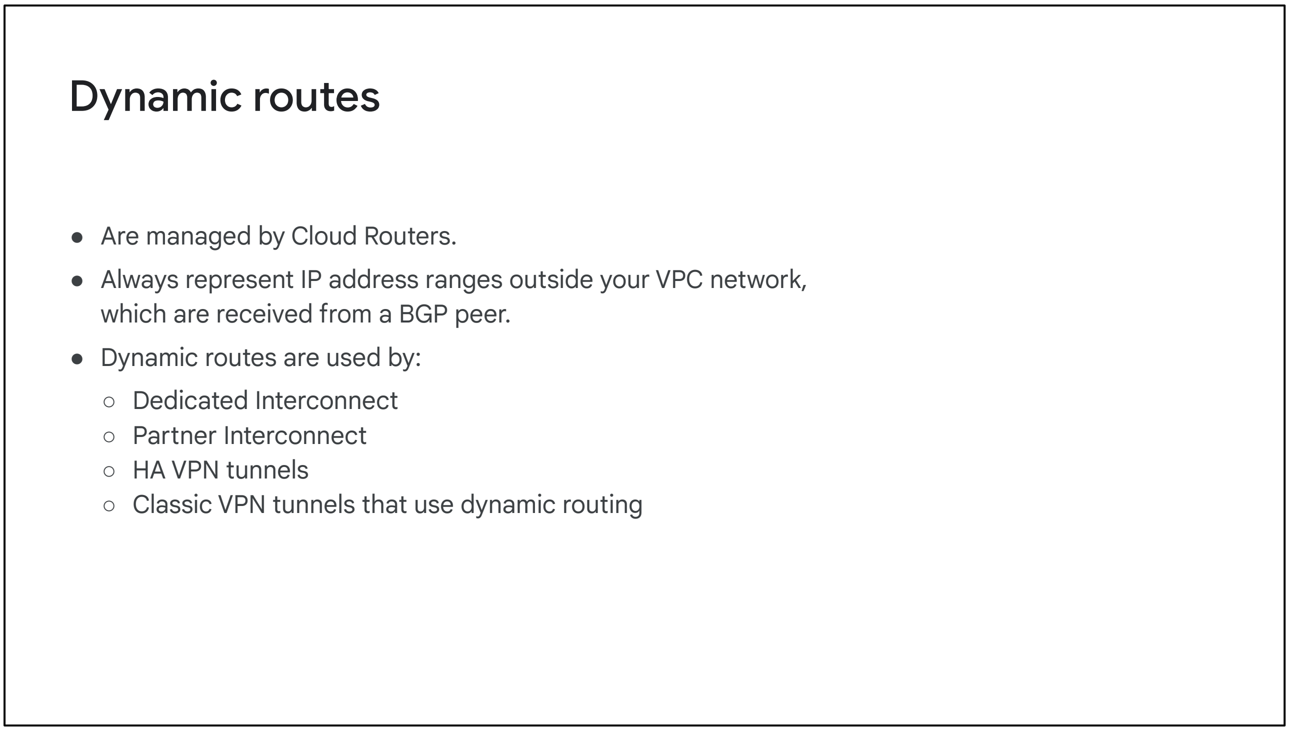


Create custom static routes

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 Console, 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

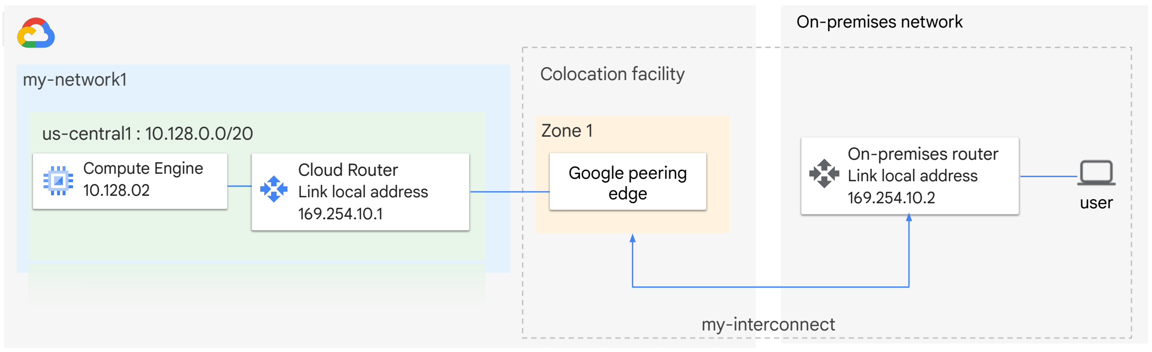
Dynamic routes are managed by Cloud Routers in the VPC network. Their destinations always represent IP address ranges outside your VPC network, which are received from a BGP peer router. BGP peer routers are typically outside the Google network (like on-premises or another cloud provider).

Dynamic routes are used by:

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

A dynamic routing example

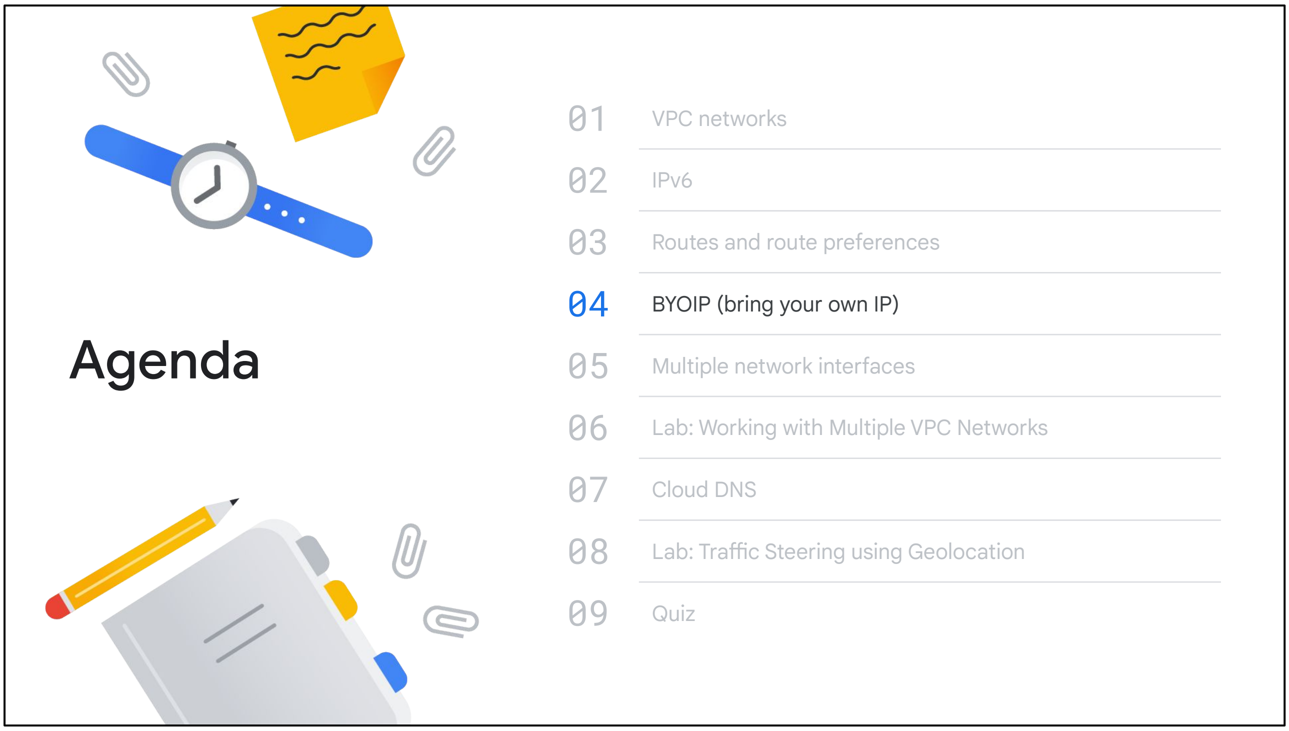
- Routes are added and removed automatically by Cloud Routers in your VPC network.
- Routes apply to VMs according to the VPC network's dynamic routing mode.



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.



Agenda

01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

05 Multiple network interfaces

06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

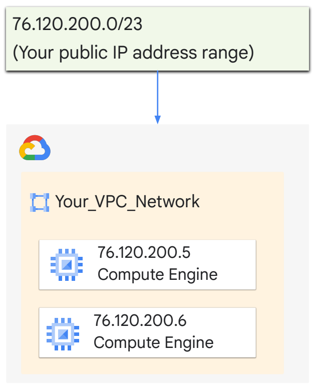
08 Lab: Traffic Steering using Geolocation

09 Quiz

In your VPC networks, you are not limited to IP addresses that are assigned by Google Cloud. Next, let's discuss BYOIP, or Bring Your Own IP addresses into Google Cloud.

Introduction to BYOIP (bring your own IP address)

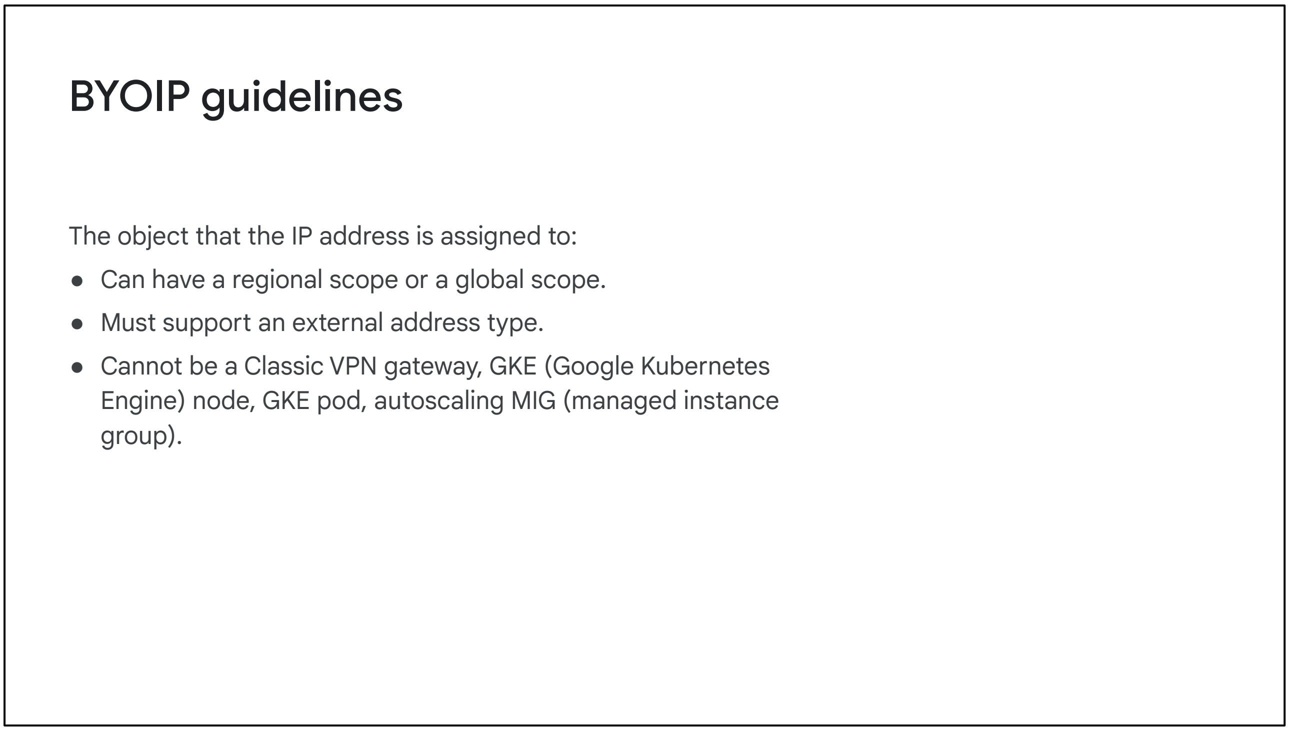
- BYOIP enables customers to:
  - Assign IP addresses from a public IP range that they own to Google Cloud resources.
  - Route traffic directly from the internet to their VMs.
- Google Cloud manages these BYOIP addresses in the same way as Google-provided IP addresses, except that:
  - The IP addresses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



BYOIP guidelines

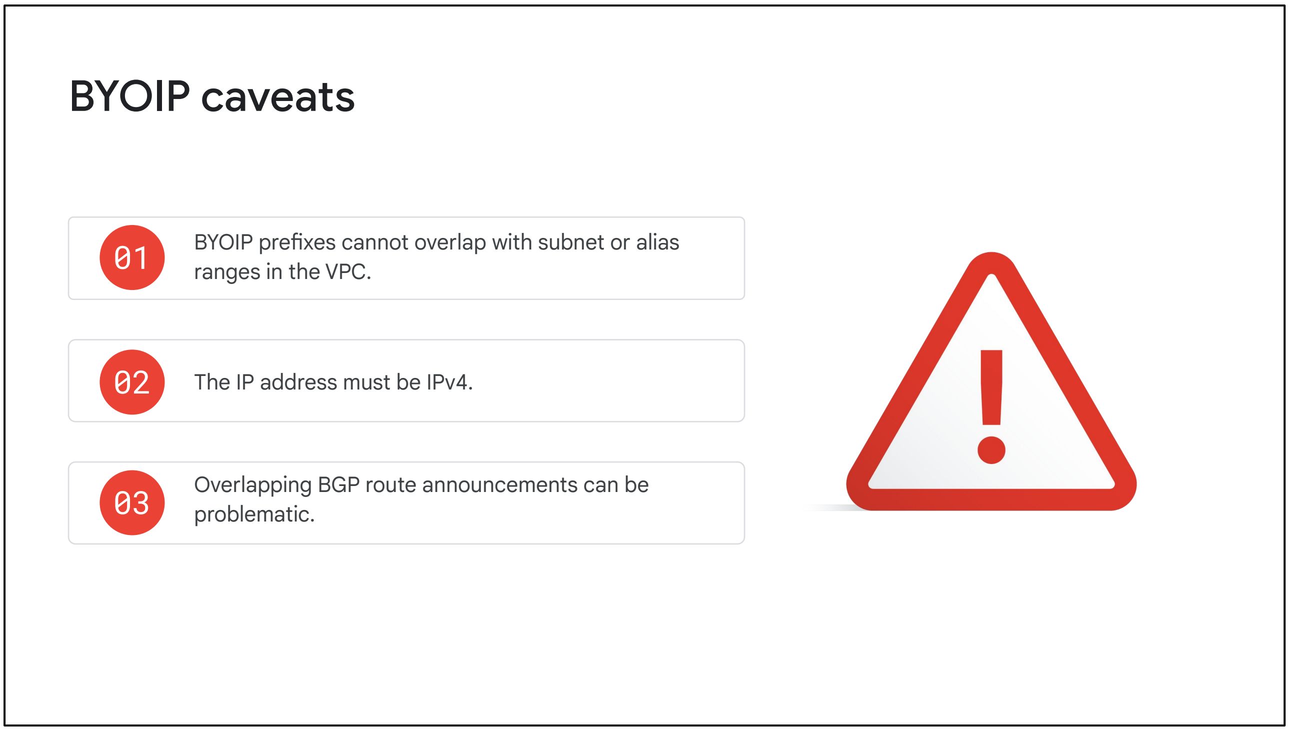
The object that the IP address is assigned to:

- 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.

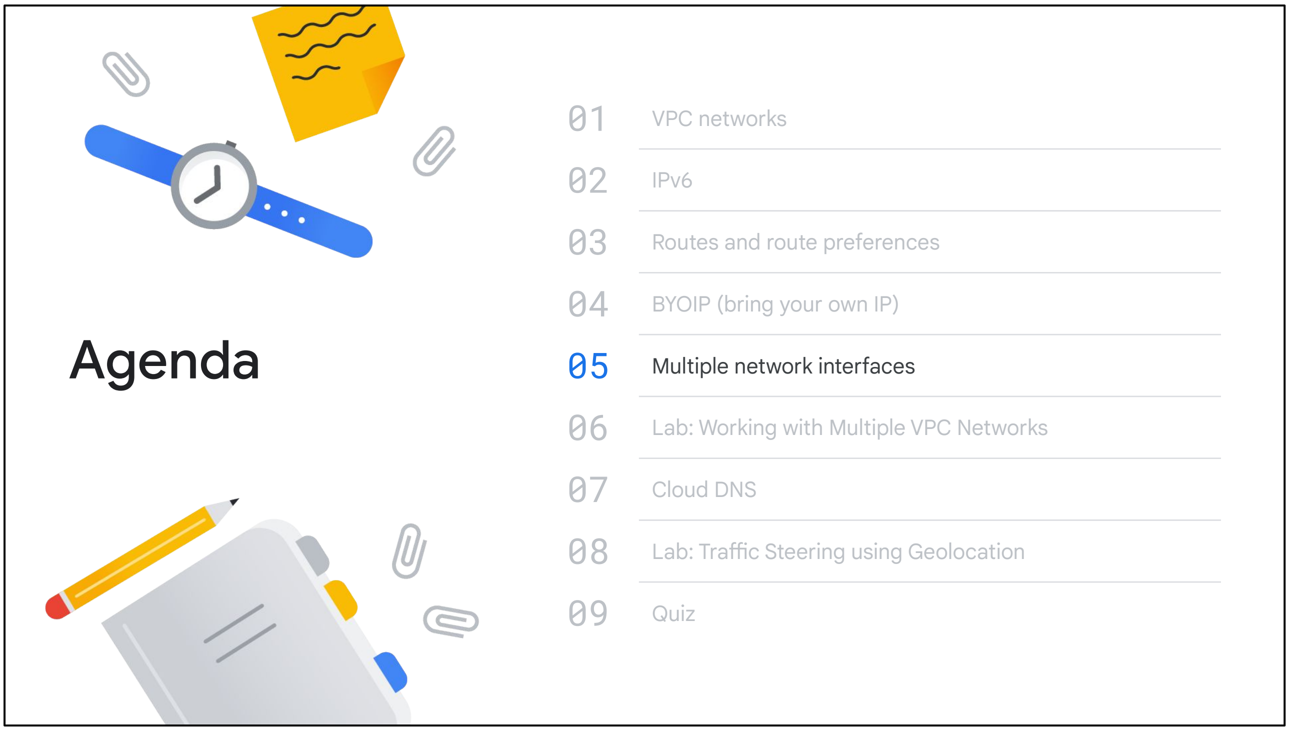
03 Overlapping BGP route announcements can be problematic.

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.



Agenda

01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

05 Multiple network interfaces

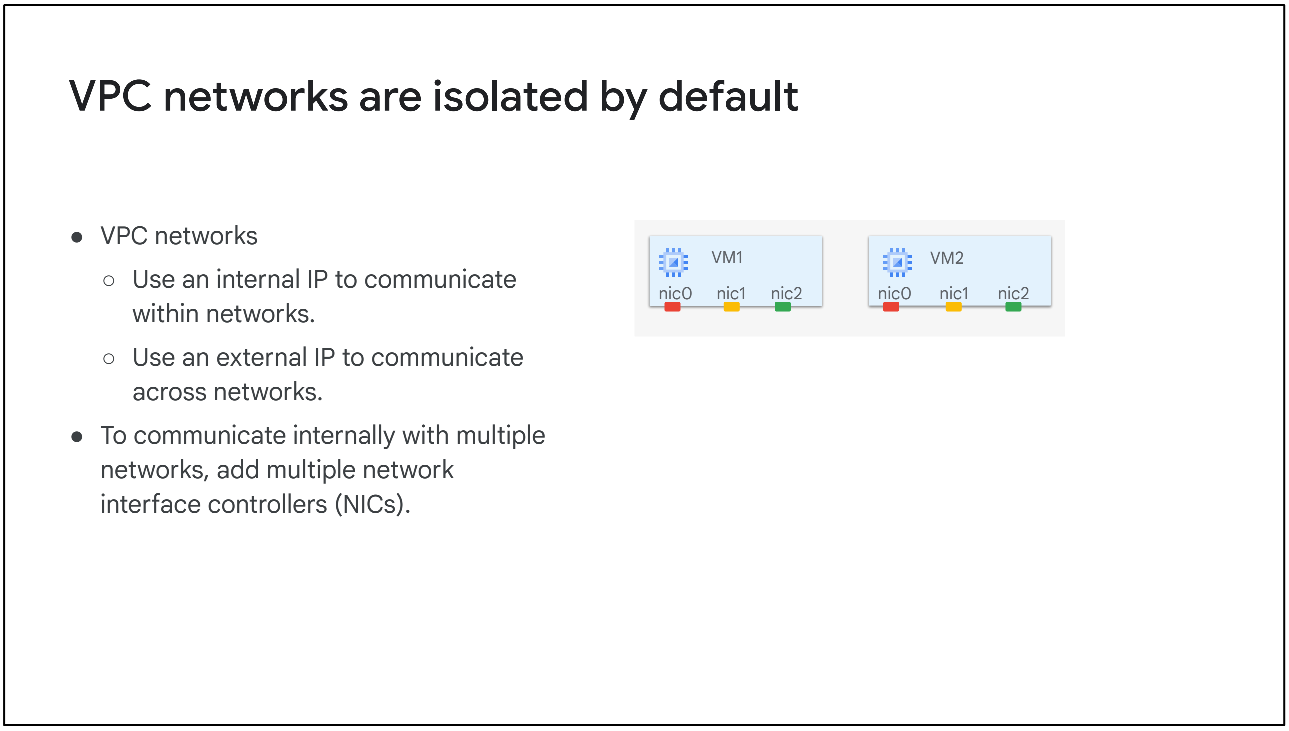
06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

08 Lab: Traffic Steering using Geolocation

09 Quiz

In conventional networking, devices can use multiple network interfaces to communicate with multiple networks. Next, let's discuss using multiple network interfaces in a Google Cloud VPC network.



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).

VM1
nic0
nic1
nic2
VM2
nic0
nic1
nic2

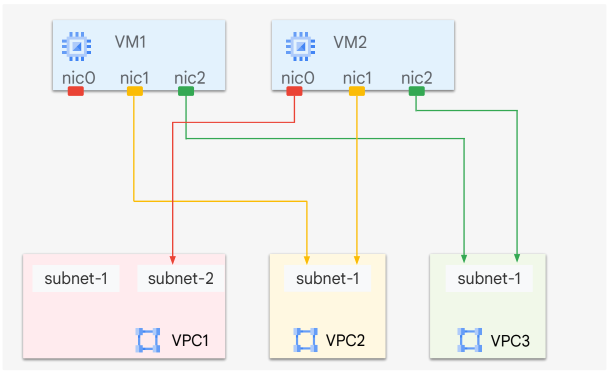
VPC networks are isolated private networking domains by default. As we mentioned earlier, VM instances within a VPC network can communicate among themselves by using internal IP addresses as long as firewall rules permit. However, no internal IP address communication is allowed between networks unless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

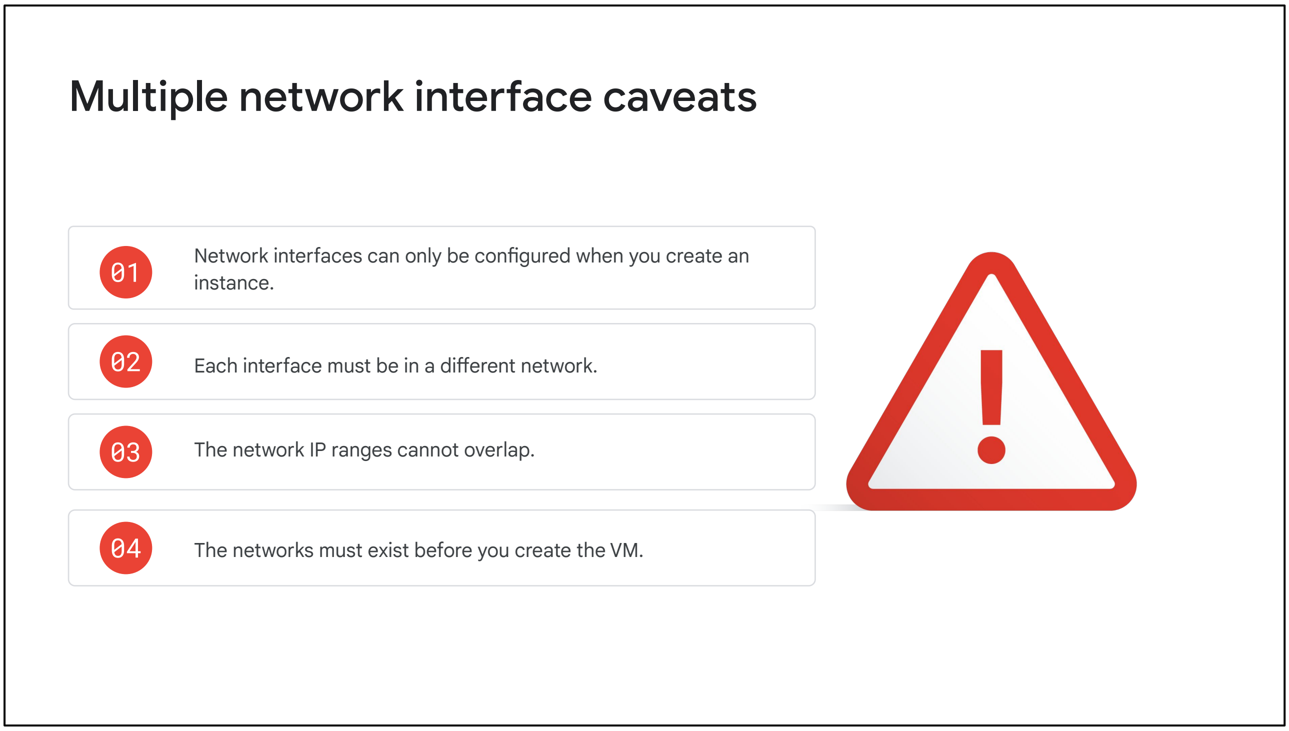
- Is attached to a separate VPC network.
- Uses an internal IP to communicate across networks.



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.

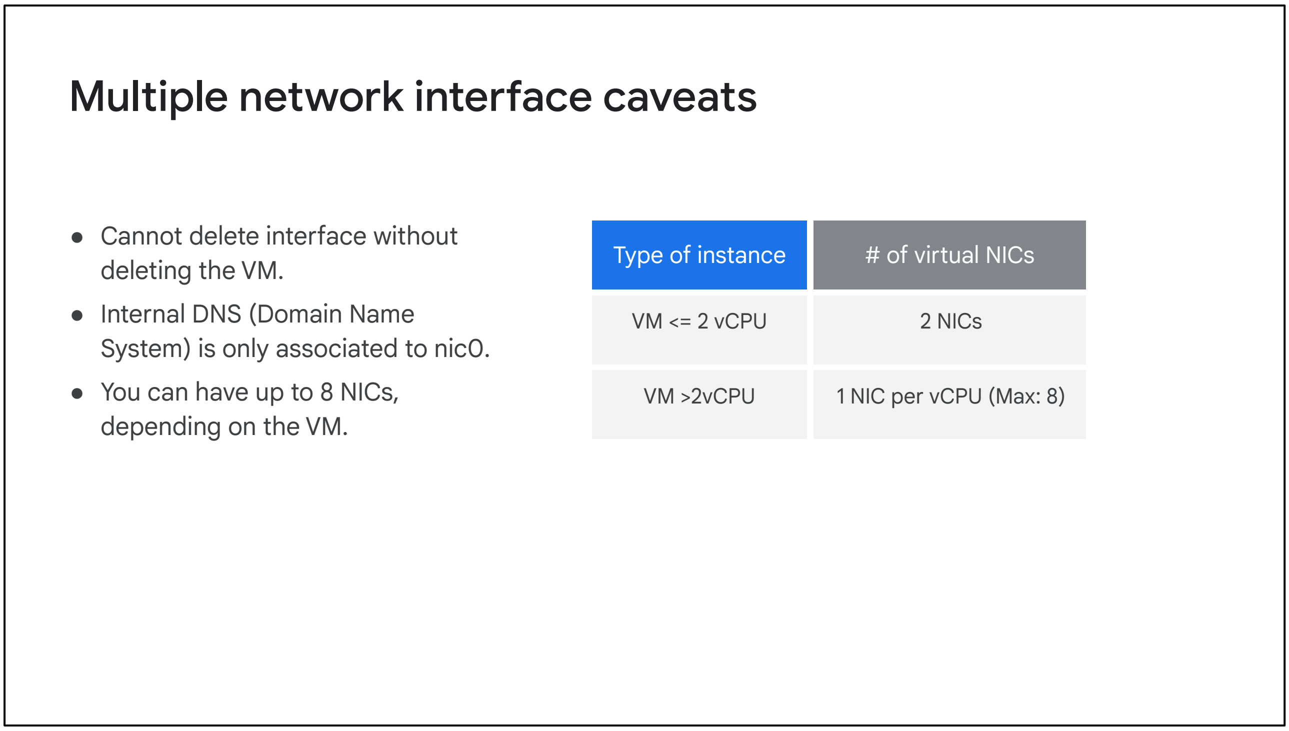
04 The networks must exist before you create the VM.

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)

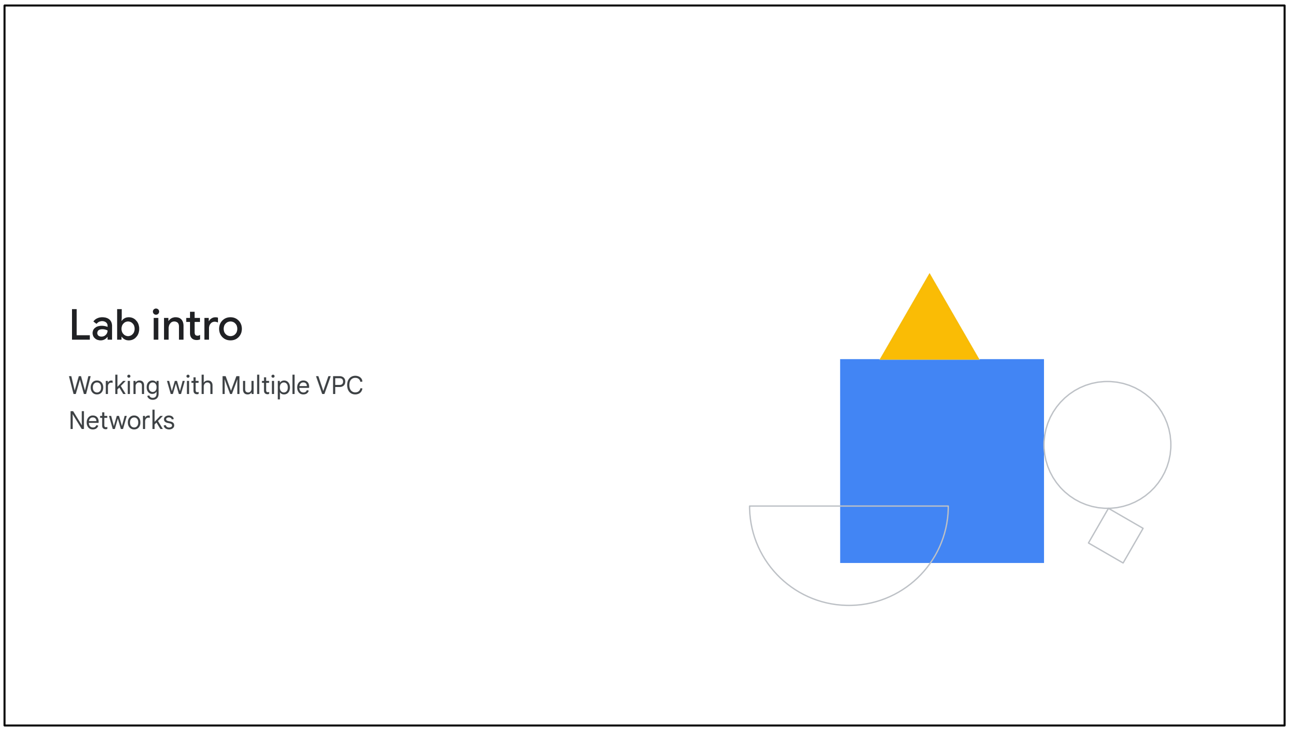
You cannot delete a network interface without deleting the instance.

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.

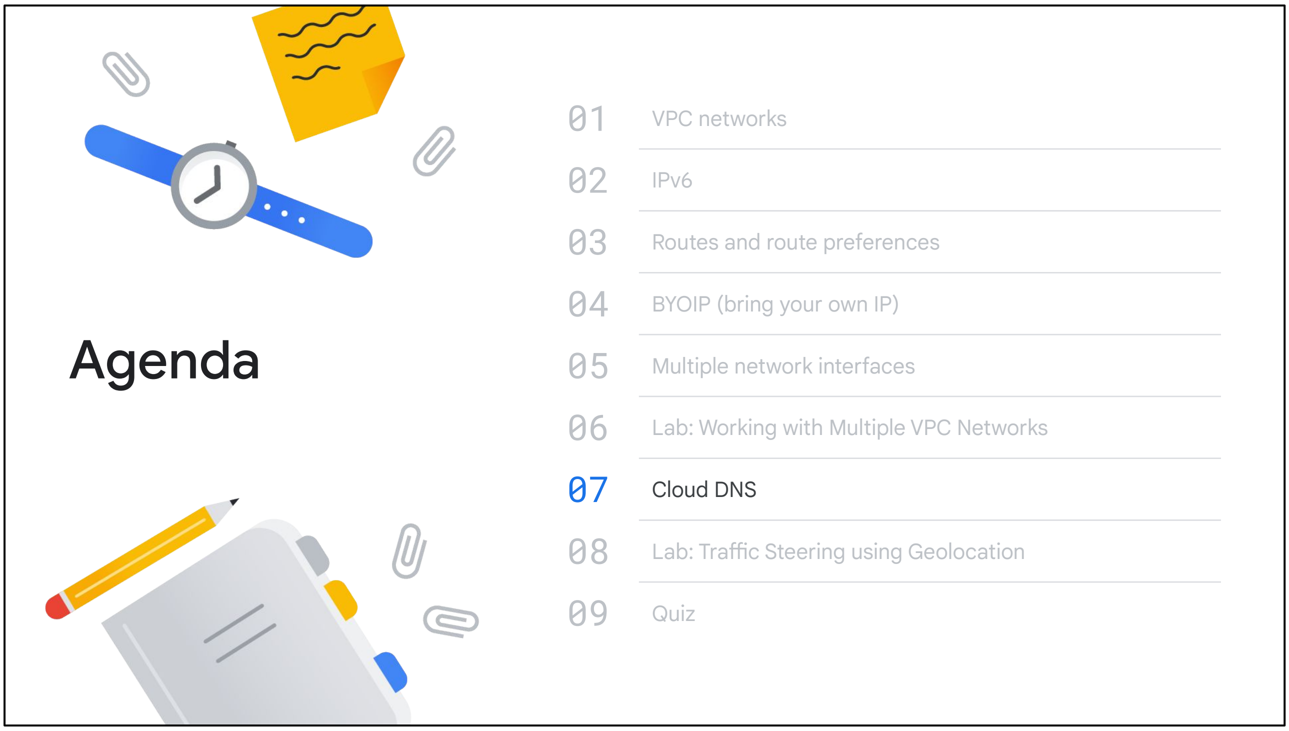


Lab intro

Working with Multiple VPC Networks

In this lab, you create several VPC networks and VM instances and test connectivity across networks. The lab tasks are to:

- Create custom mode VPC networks with firewall rules.
- Create VM instances by using Compute Engine.
- Explore the connectivity for VM instances across VPC networks.
- Create a VM instance with multiple network interfaces.



Agenda

01 VPC networks

02 IPv6

03 Routes and route preferences

04 BYOIP (bring your own IP)

05 Multiple network interfaces

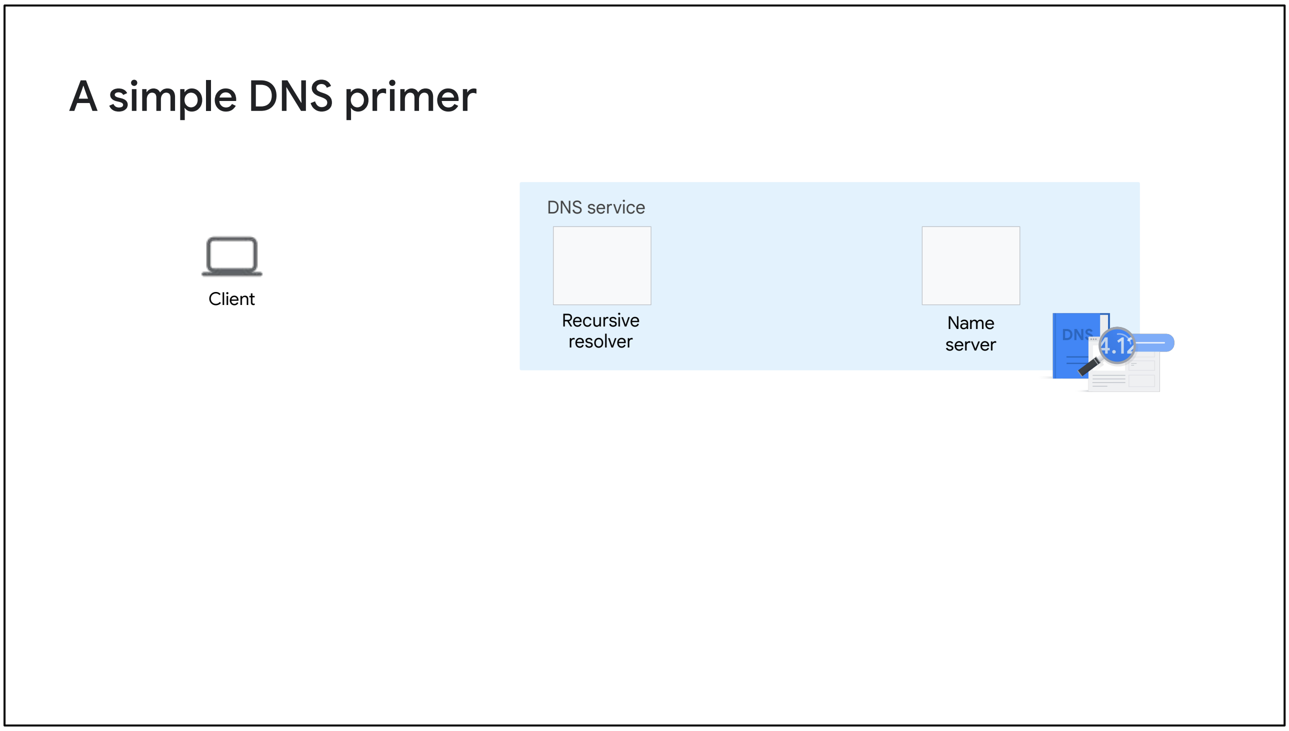
06 Lab: Working with Multiple VPC Networks

07 Cloud DNS

08 Lab: Traffic Steering using Geolocation

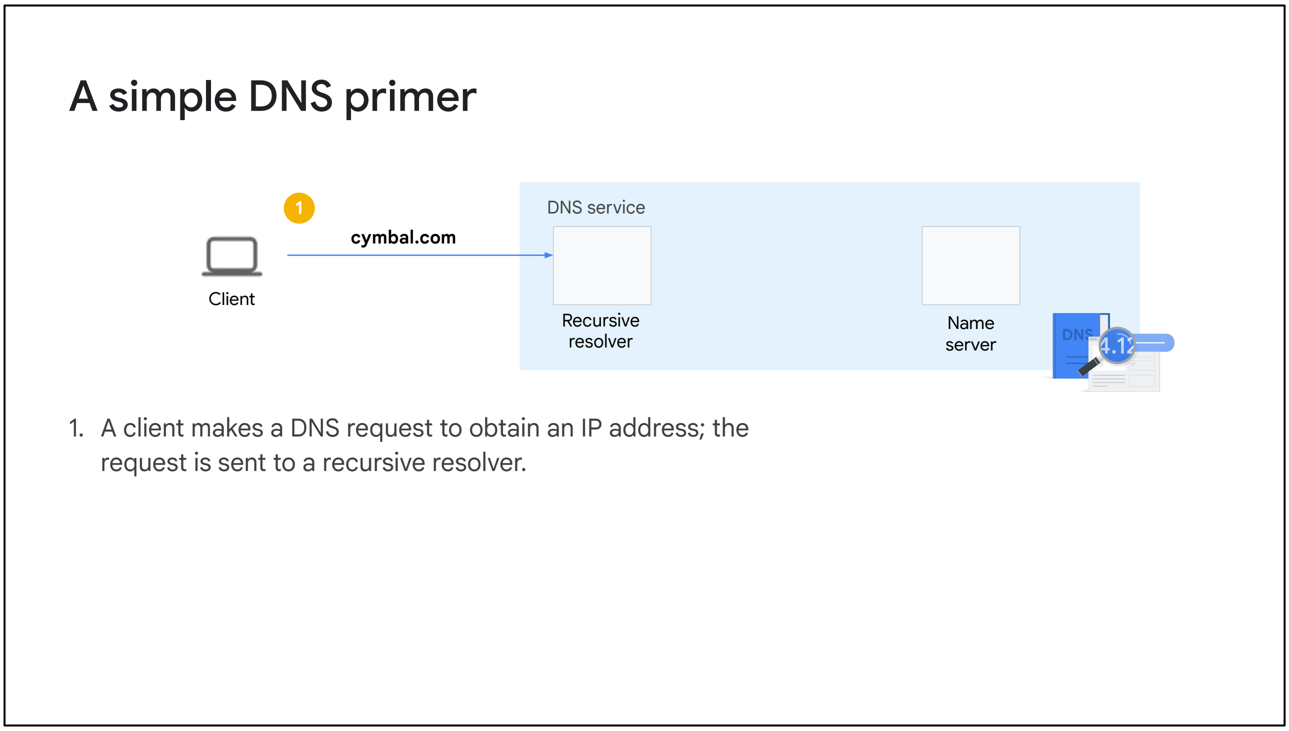
09 Quiz

Next, let’s discuss about Cloud DNS. After a brief overview of DNS (Domain Name Service), we’ll cover how to use Cloud DNS policies to refine how you can Cloud DNS with your VPC networks.

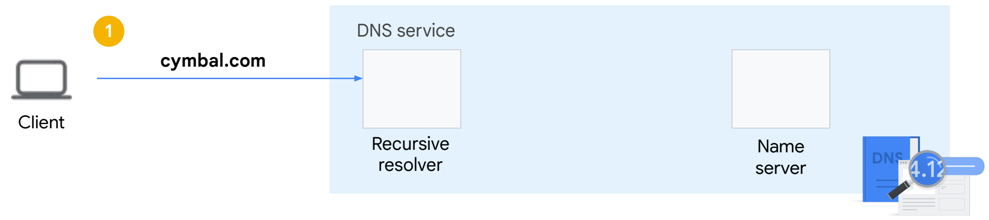

A simple DNS primer

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).



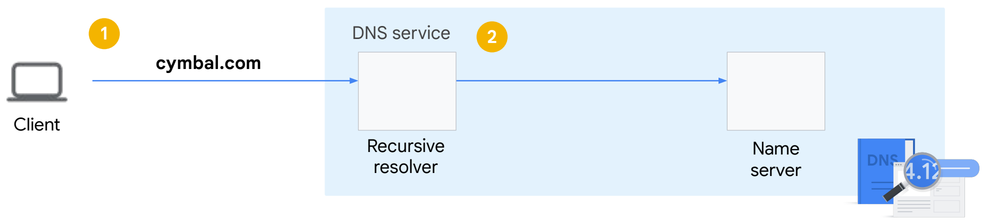
A simple DNS primer



1. A client makes a DNS request to obtain an IP address; the request is sent to a recursive resolver.

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.

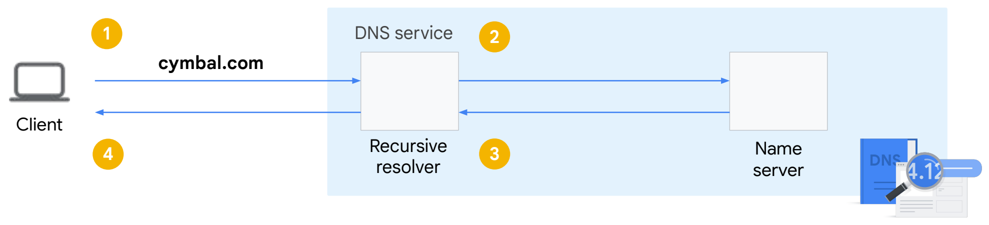
A simple DNS primer



1. A client makes a DNS request to obtain an IP address; the request is sent to a recursive resolver.
2. A recursive resolver requests the IP address from a name server.

The DNS service recursive resolver issues a request to look up the IP address of cymbal.com from one of its name servers.

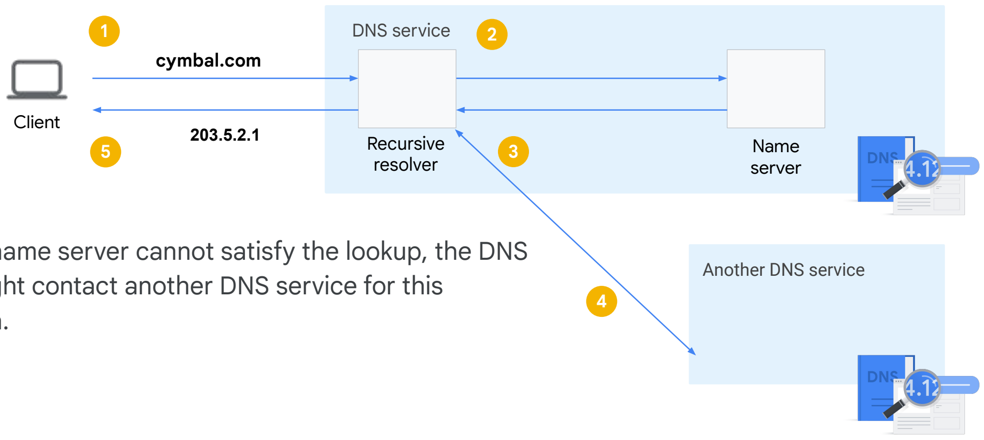
A simple DNS primer



1. A client makes a DNS request to obtain an IP address; the request is sent to a recursive resolver.
2. A recursive resolver requests the IP address from a name server.
3. The name server responds with the IP address.
4. The recursive resolver sends the IP to the client.

The name server responds with the ISP, and the recursive resolver sends the IP to the client.

A simple DNS primer



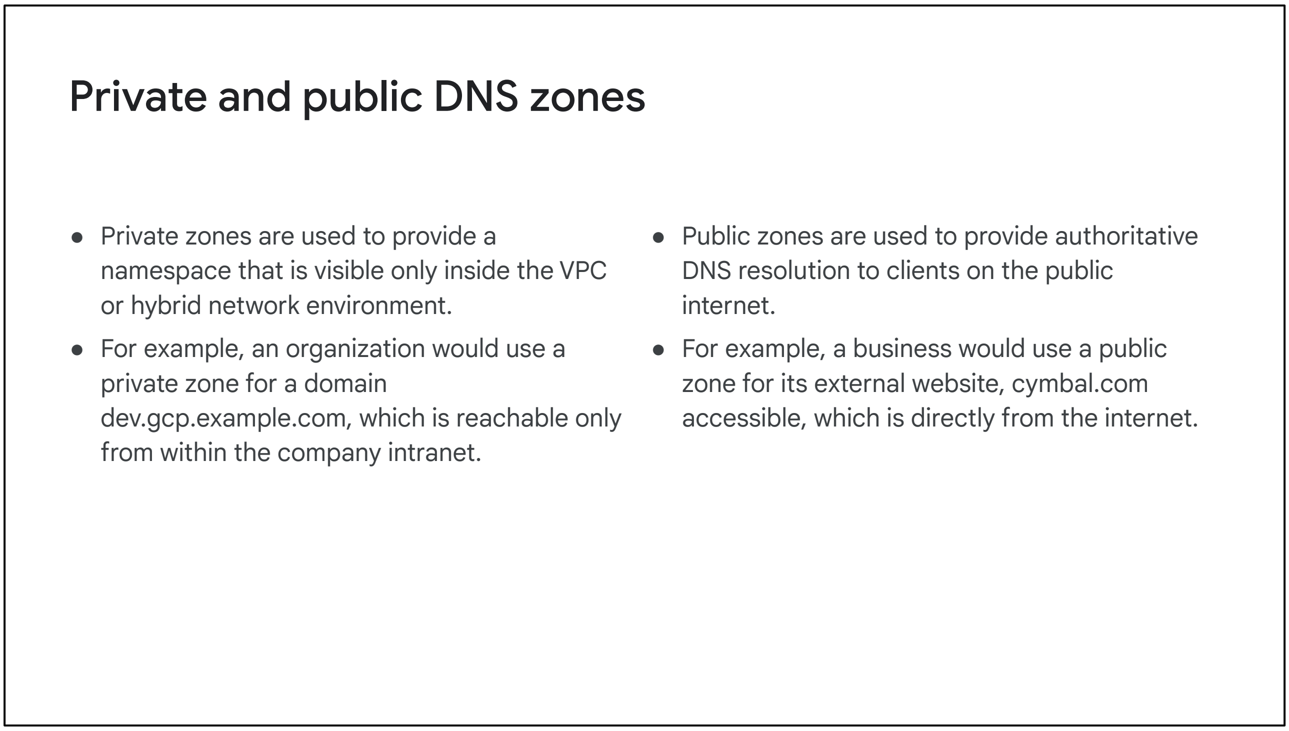
When the name server cannot satisfy the lookup, the DNS service might contact another DNS service for this information.

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.



Private and public DNS zones

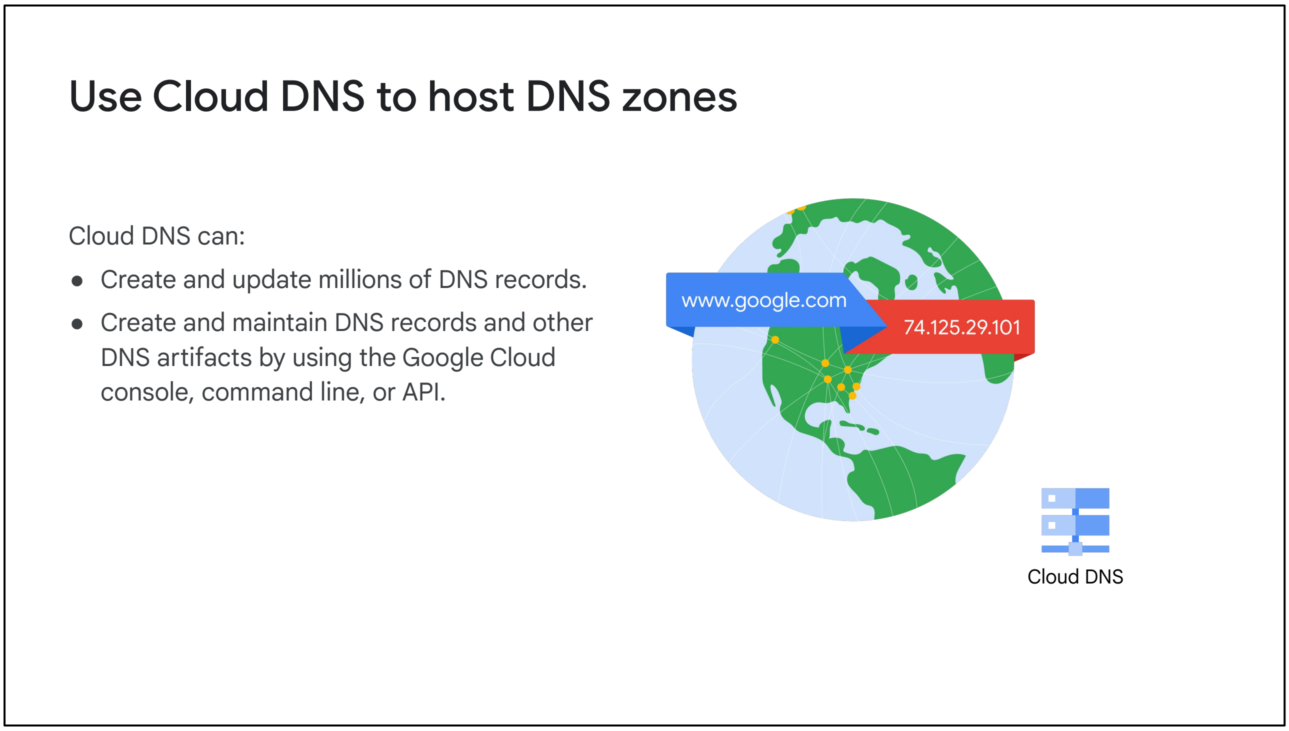
- Private zones are used to provide a namespace that is visible only inside the VPC or hybrid network environment.
- For example, an organization would use a private zone for a domain dev.gcp.example.com, which is reachable only from within the company intranet.

- 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 accessible, which is directly from 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.

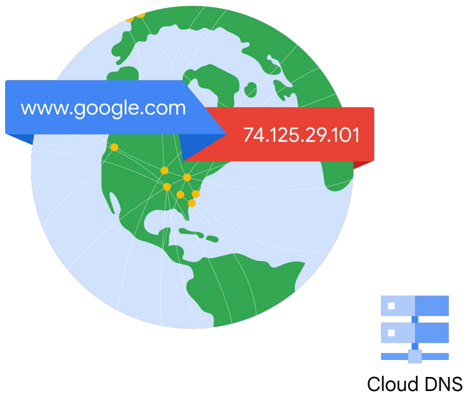
Don’t confuse the concept of a public zone with Google Public DNS (8.8.8.8). Google Public DNS is just a public recursive resolver.



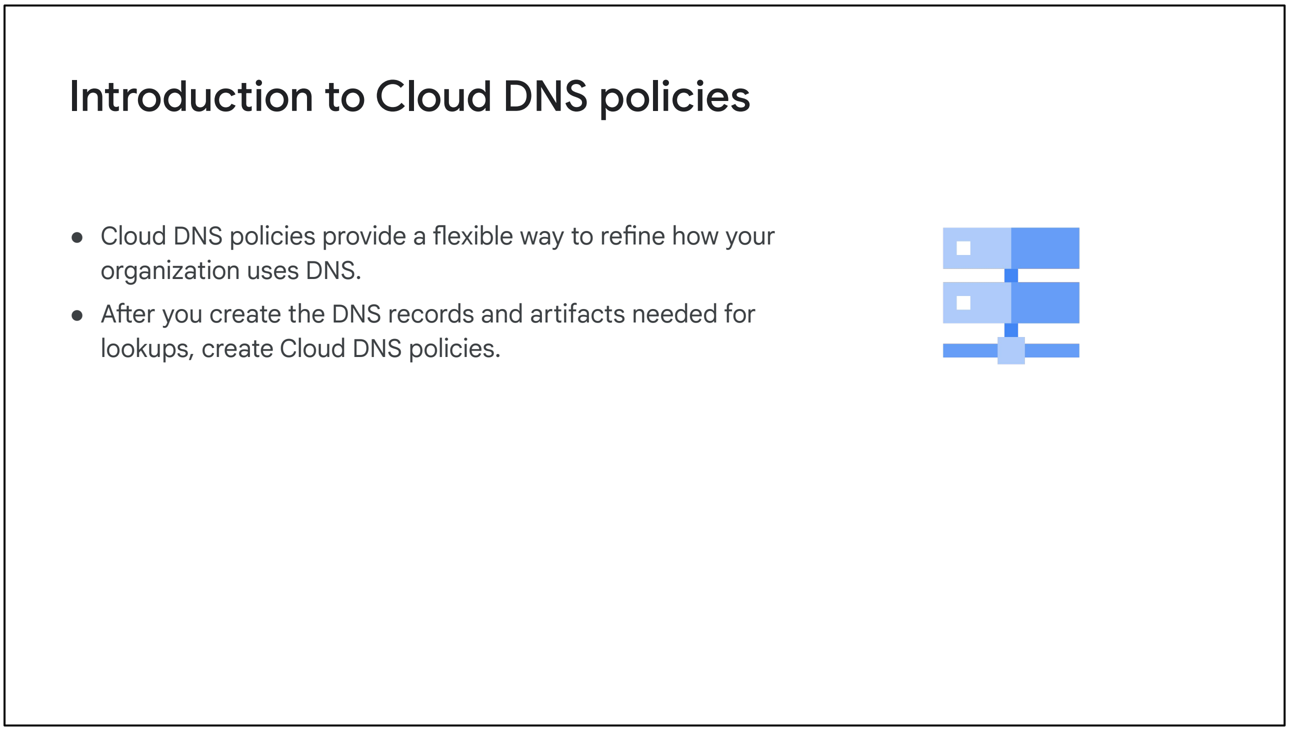
Use Cloud DNS to host DNS zones

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.

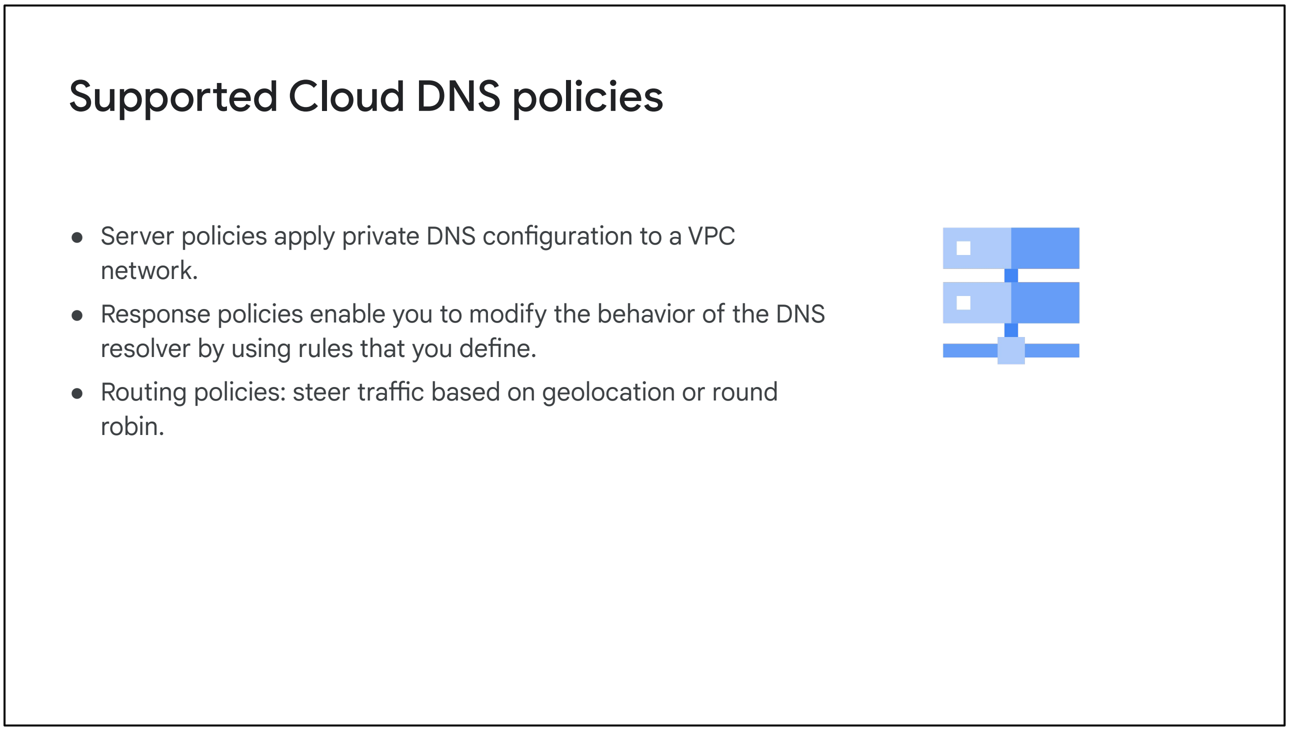


Introduction to Cloud DNS policies

- 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.



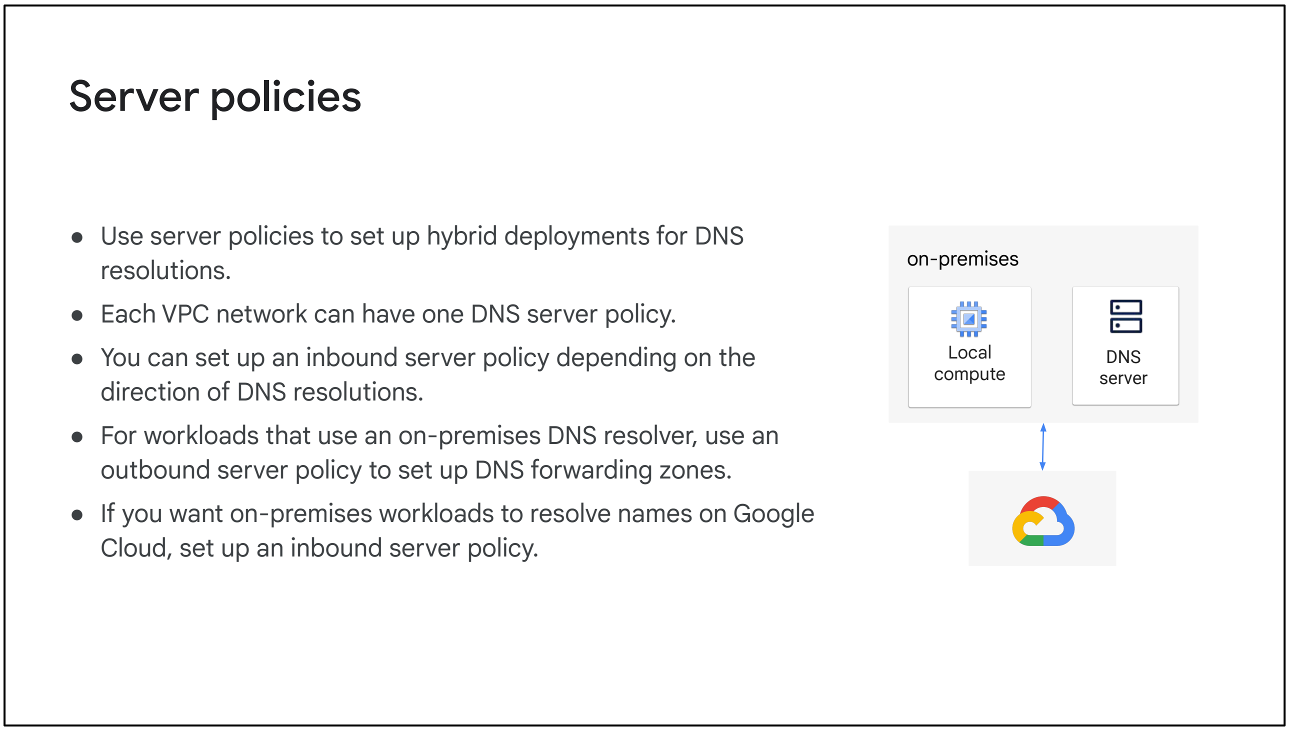
Supported Cloud DNS policies

- 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:

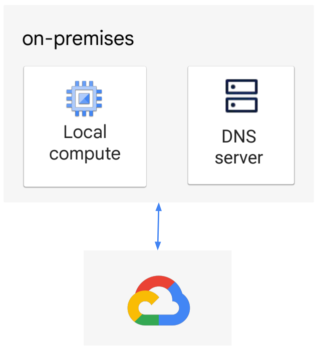
- Server policies apply private DNS configuration to a VPC network.
- Response policies enable you to modify the behavior of the DNS resolver by using rules that you define.
- Routing policies steer traffic based on geolocation or round robin.

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 workloads that use an on-premises DNS resolver, use an outbound server policy to set up DNS forwarding zones.
- If you want on-premises workloads to resolve names on Google Cloud, set up an inbound server policy.

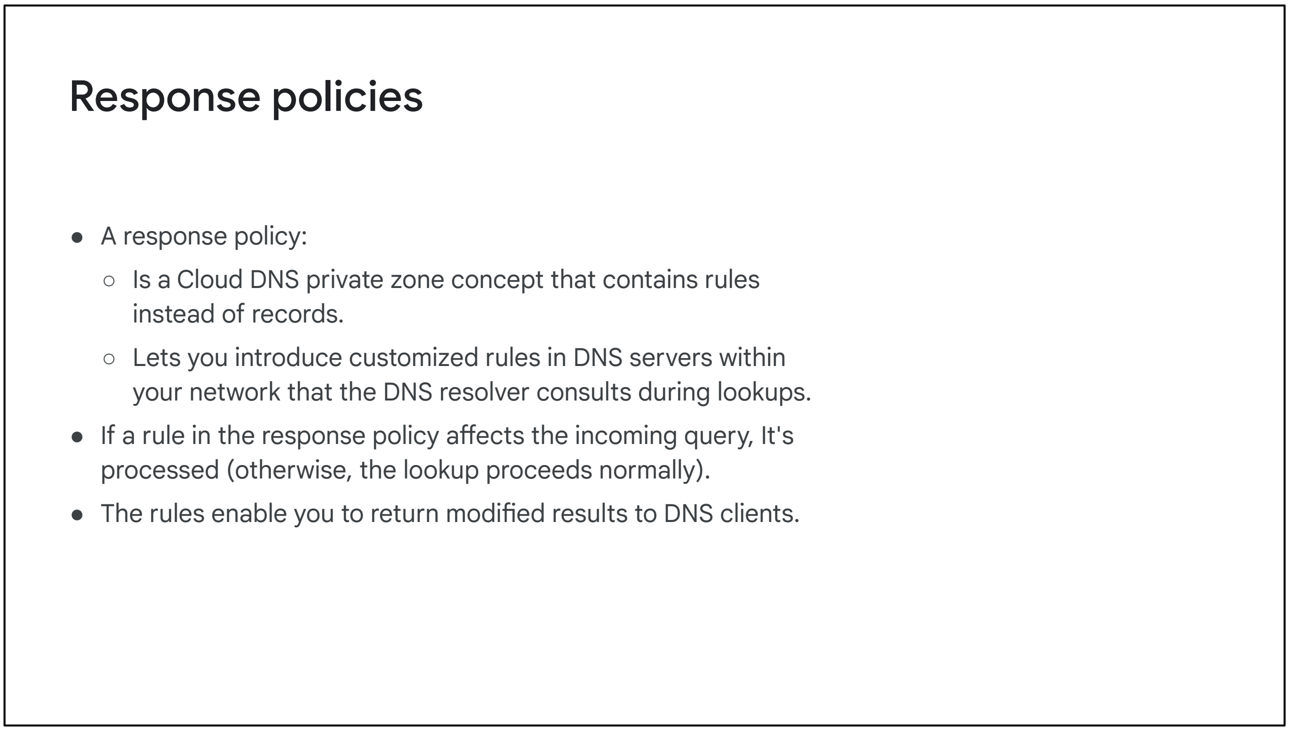


Use server policies to set up hybrid deployments for DNS resolution. You can set up an inbound server policy depending on the direction of the DNS resolutions. If your workloads plan to use an on-premises DNS resolver, you can set up DNS forwarding zones by using an outbound server policy. If you want 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

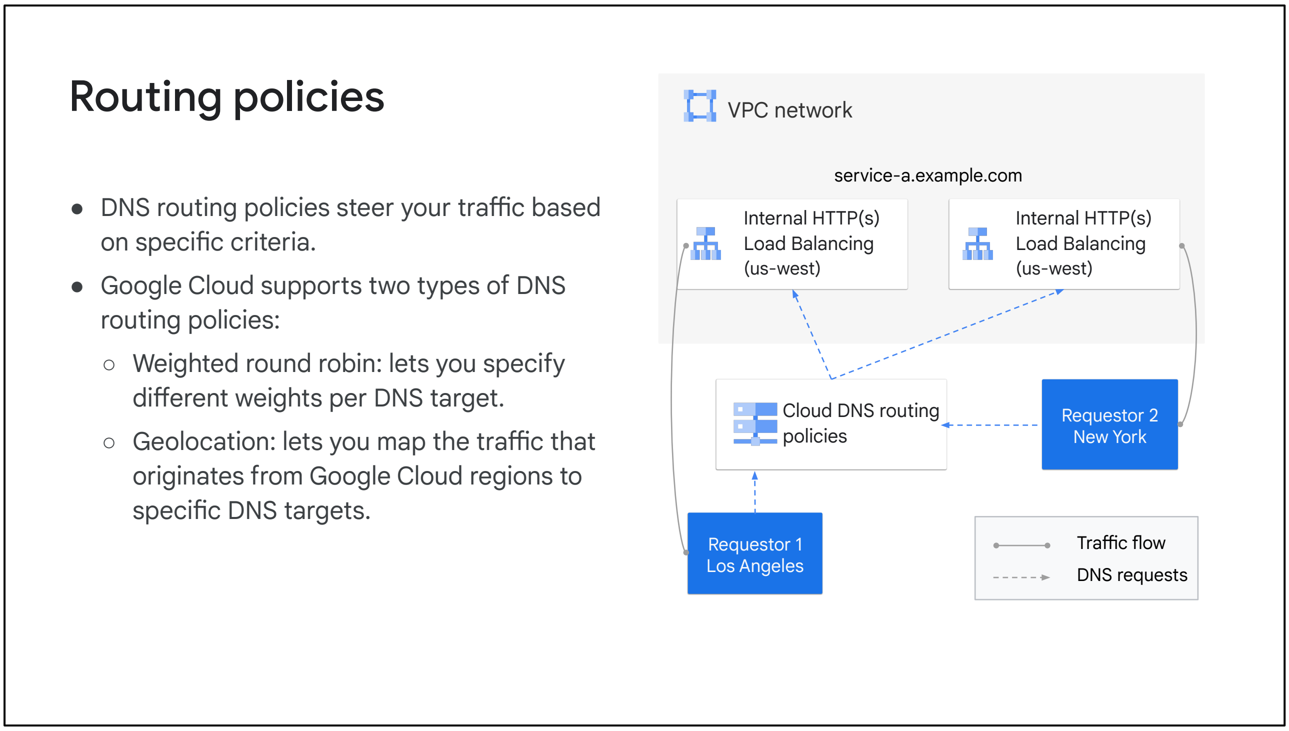
- A response policy:
  - Is a Cloud DNS private zone concept that contains rules instead of records.
  - Lets you introduce customized rules in DNS servers within your network that the DNS resolver consults during lookups.
- If a rule in the response policy affects the incoming query, It's processed (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.

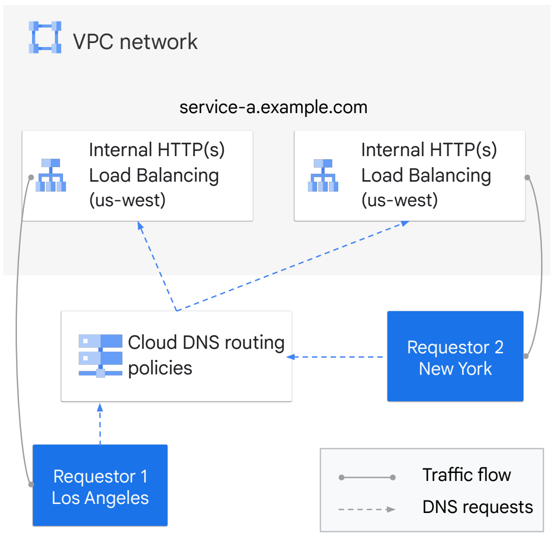
If a rule in the response policy affects the incoming query, it's processed. Otherwise, the lookup proceeds normally. For more information, see *Manage response policies and rules* in the Google Cloud documentation.

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.



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.


VPC network
service-a.example.com
Internal HTTP(s) Load Balancing (us-west)
Internal HTTP(s) Load Balancing (us-west)
Cloud DNS routing policies
Requestor 2 New York
Requestor 1 Los Angeles
Traffic flow
DNS requests

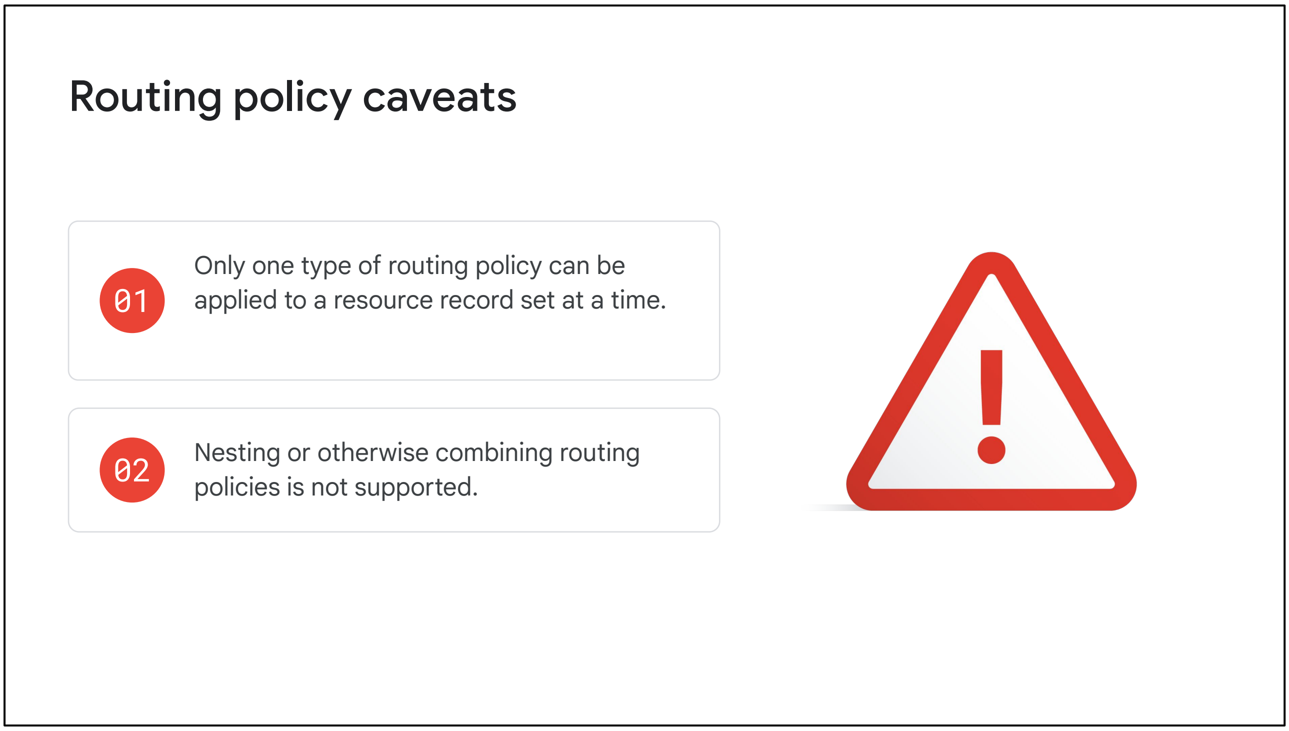
DNS routing policies let you steer your traffic based on specific criteria. Google Cloud supports two types of DNS routing policies: weighted round robin and geolocation.

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.



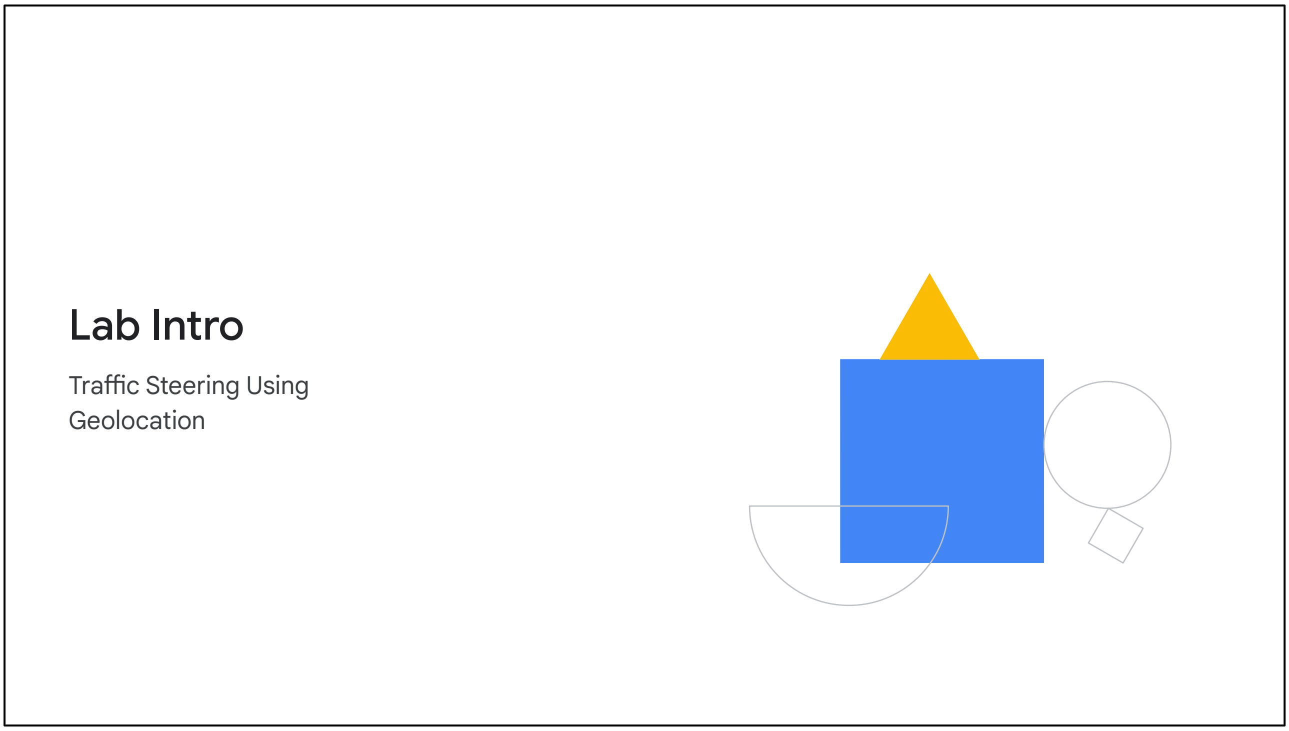
Routing policy caveats

01 Only one type of routing policy can be applied to a resource record set at a time.

02 Nesting or otherwise combining routing policies is not supported.

When configuring routing policies, consider these caveats:

- Only one type of routing policy can be applied to a resource record set at a time.
- Nesting or otherwise combining routing policies is not supported.



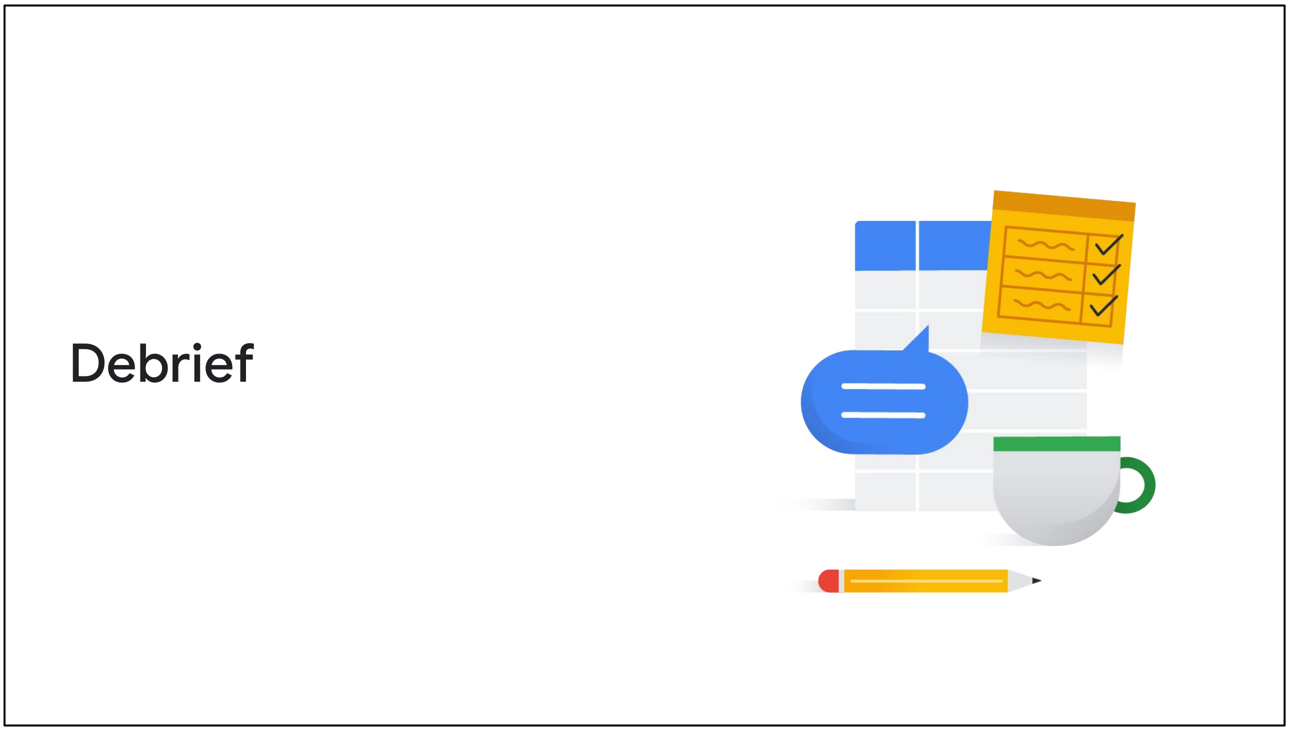
Lab Intro

Traffic Steering Using Geolocation

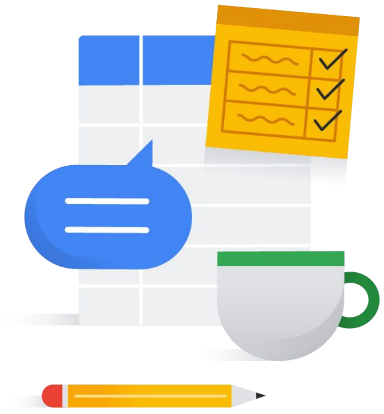
In this lab, you will configure and test the geolocation routing policy. The geolocation routing policy applies the nearest match for the source location when the traffic source location doesn't match any policy items exactly.

The lab tasks are to:

- Launch client VMs, one in each region.
- Launch server VMs, one in each region except asia-south1.
- Create a private zone, for example.com.
- Create a Geolocation routing policy using gcloud commands.
- Test the configuration.



Debrief



In this module, you learned about some fundamental Google Cloud VPC networking concepts.

We began with a brief overview of VPC networks. After that, you learned how you can use IPv6 addressing within Google Cloud. You also learned about routes and route preferences, BYOIP (or bring your own IP address), multiple network interfaces, and Cloud DNS.

You applied your knowledge in two lab exercises - Working with Multiple VPC Networks and Traffic Steering using Geolocation.

At the end of the module, you took short quiz to check your understanding of the lecture material. Thanks for watching this lecture!