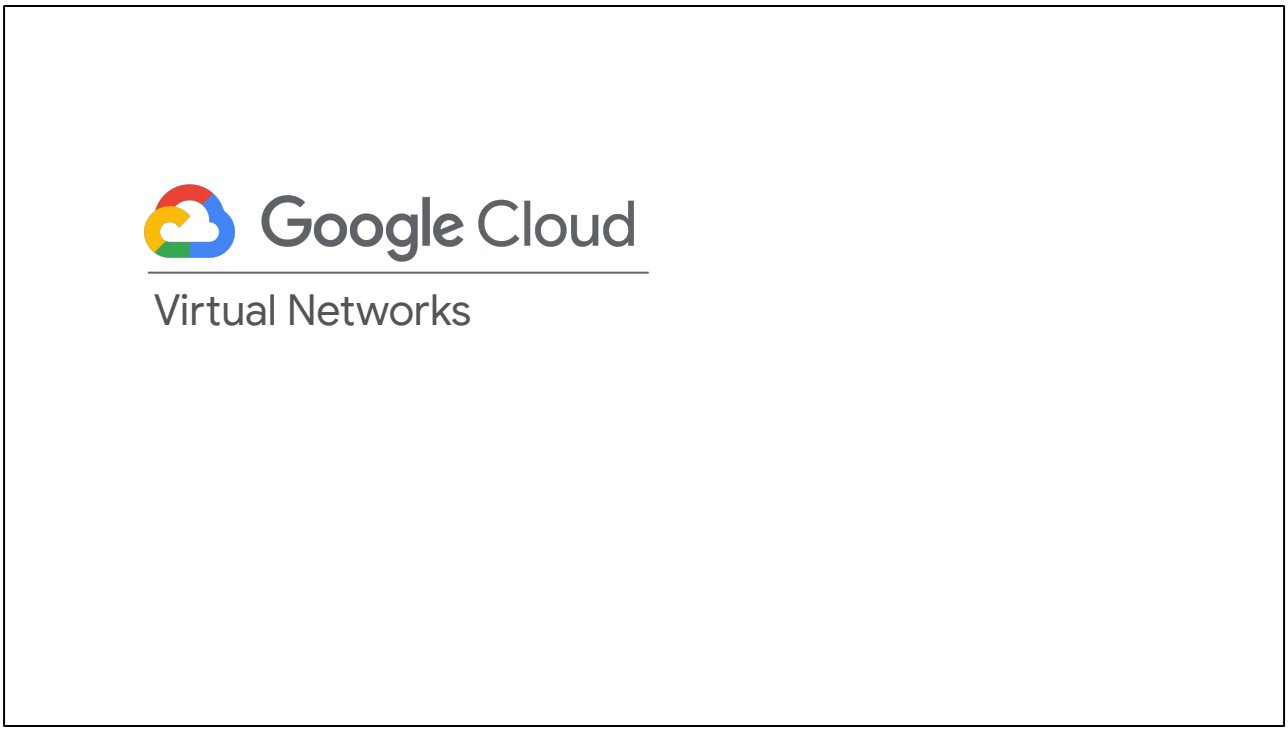
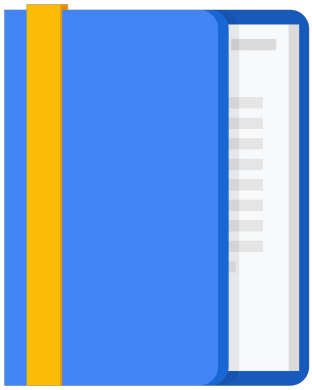


In this module, we will be covering virtual networks.

GCP uses a software-defined network that is built on a global fiber infrastructure. This infrastructure makes GCP 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.

Agenda

Virtual Private Cloud (VPC)
Projects, networks, and subnetworks
IP addresses
Routes and firewall rules
Pricing
Lab
Common network designs
Lab

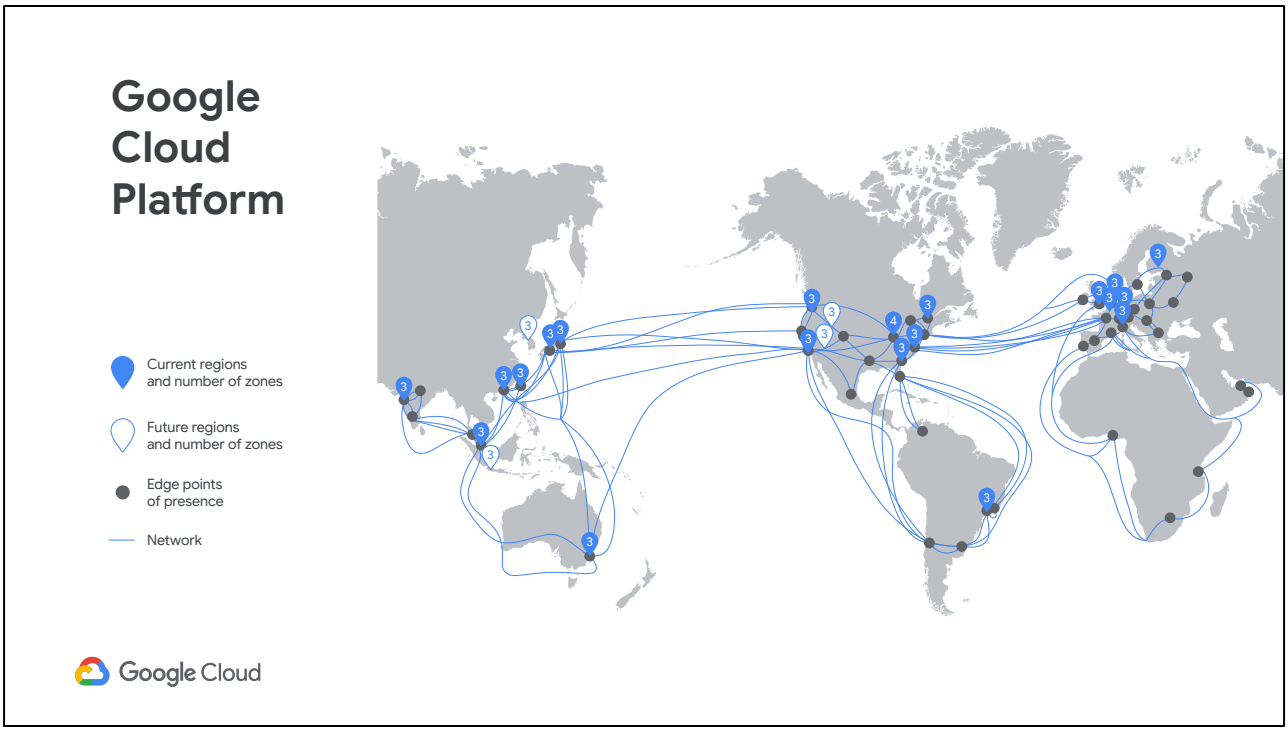


Google Cloud

In this module, we start by introducing Virtual Private Cloud, or VPC, which is Google's managed networking functionality for your Cloud Platform resources. Then we dissect networking into its fundamental components, which are projects, networks, subnetworks, IP addresses, routes and firewall rules, along with network pricing.

Next, you will explore GCP's network structure in a lab by creating networks of many different varieties and exploring the network relationships between them.

After that, we will look at common network designs, like a bastion host, which you will get to implement in a lab.



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

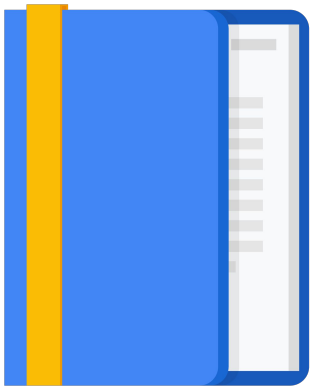
A region is a specific geographical location where you can run your resources. This map shows several regions that are currently operating, as well as future regions. The number on each region represents the zones within that region. For example, in Iowa there is a region called us-central1 which has 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 documentation in the links section of this video. [https://cloud.google.com/compute/docs/regions-zones/]

The PoPs are where Google's network is connected to the rest of the internet. GCP 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, see the links section of this video. [https://peering.google.com/#/infrastructure]

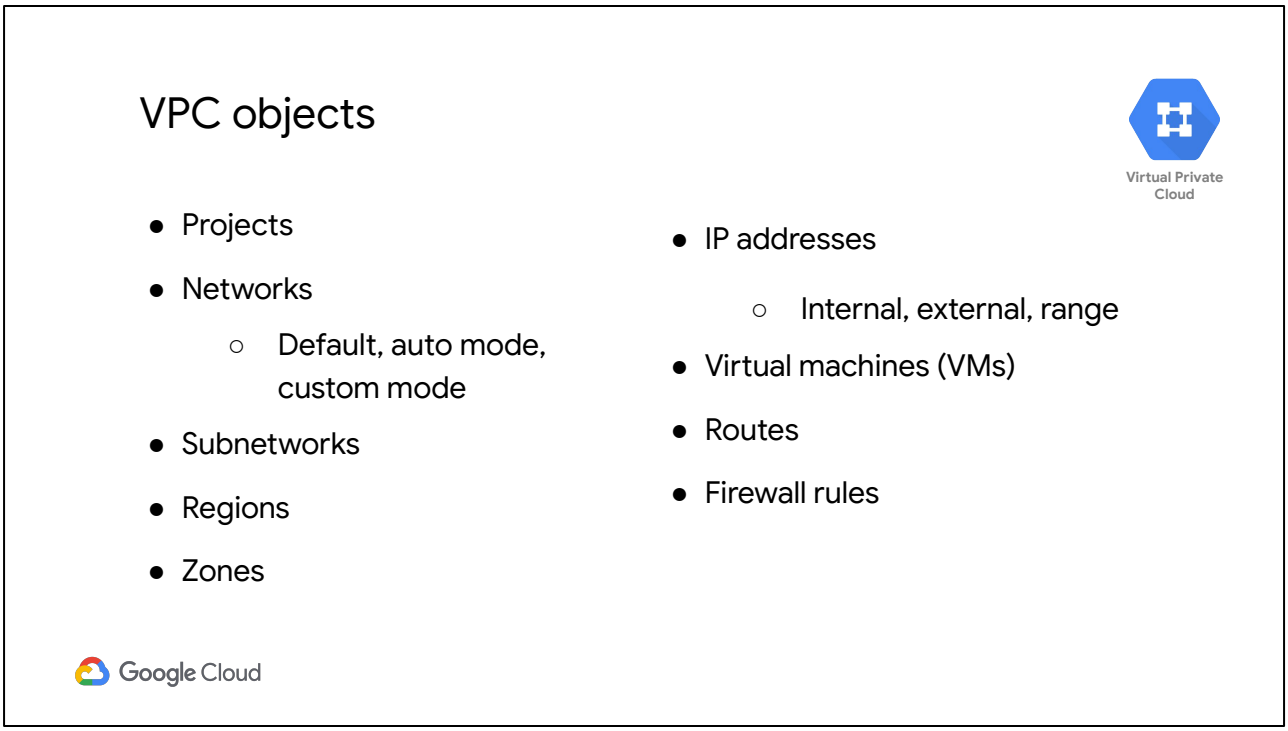
Agenda

- Virtual Private Cloud (VPC)
- Projects, networks, and subnetworks
- IP addresses
- Routes and firewall rules
- Pricing
- Lab
- Common network designs
- Lab



Google Cloud

Let's start by talking about GCP's network, and specifically Virtual Private Cloud, or VPC.



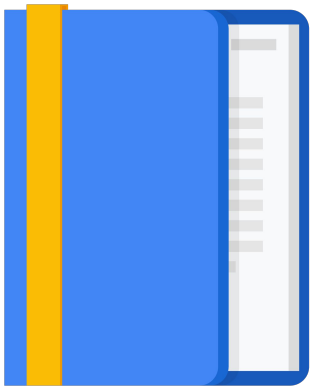
With GCP, you can provision your GCP resources, connect them to each other, and isolate them from each other in a Virtual Private Cloud. You can also define fine-grained networking policies within GCP, and between GCP and on-premises or other public clouds. Essentially, VPC is a comprehensive set of 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.

Agenda

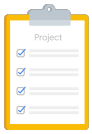
- Virtual Private Cloud (VPC)
- **Projects, networks, and subnetworks**
- IP addresses
- Routes and firewall rules
- Pricing
- Lab
- Common network designs
- Lab



Google Cloud

Let’s start exploring the VPC objects by looking at projects, networks, and subnetworks.

Projects and networks



A project:

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

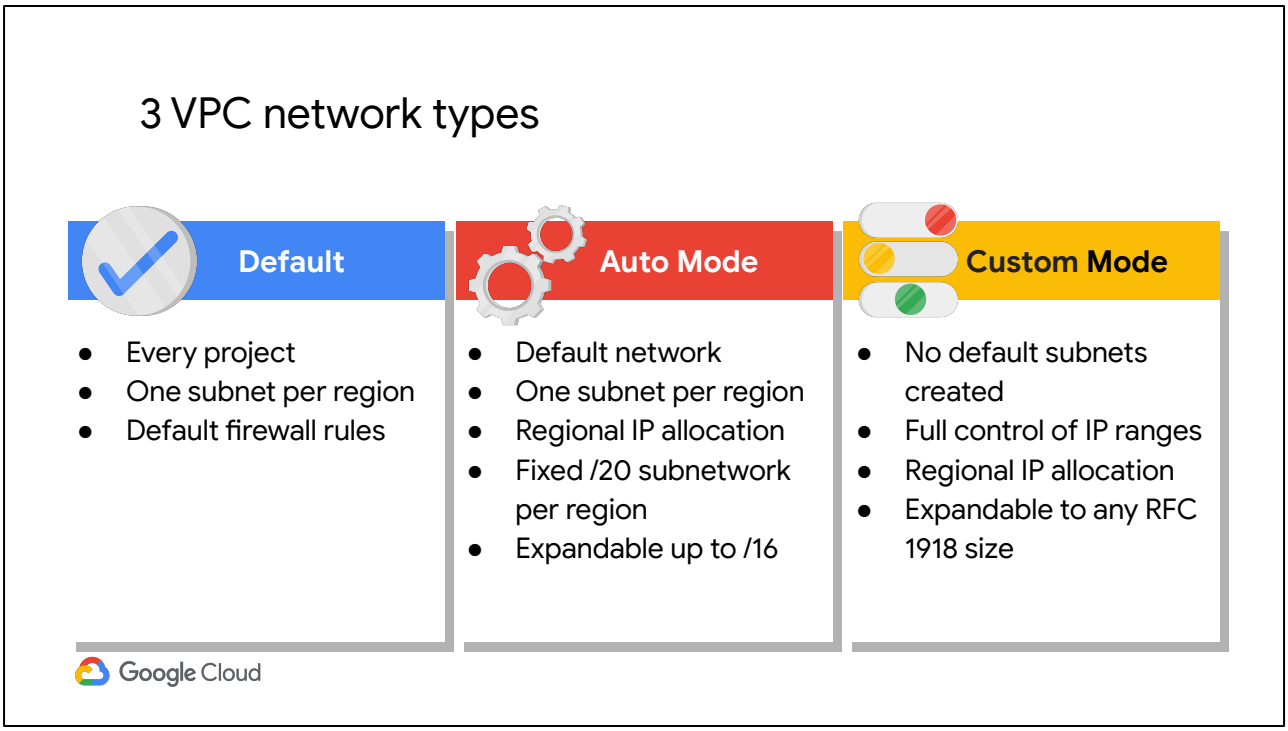
A network:

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

Google Cloud

Projects are the key organizer of infrastructure resources in GCP. 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 GCP 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 series.

These networks do not have IP ranges but are simply a construct of all of the individual IP addresses and services within that network. GCP's networks are global, spanning all available regions across the world that I 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.
I just mentioned that there are different types of networks: default, auto, and custom. Let's explore these types of networks in more detail.



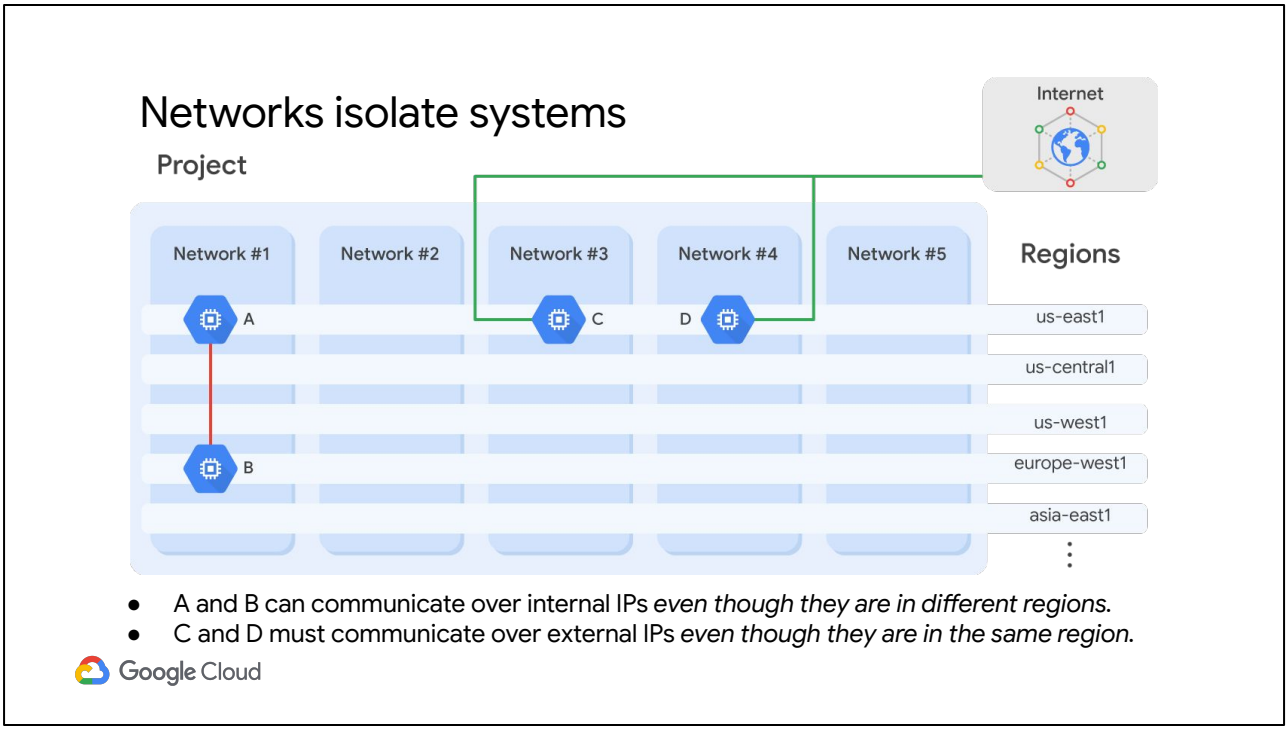
Every project is provided with a default VPC network with preset subnets and firewall rules. Specifically, a subnet is allocated for each region with non-overlapping CIDR blocks and firewall rules that allow ingress traffic for ICMP, RDP, and SSH traffic from anywhere, as well as ingress traffic from 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 GCP 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 within the RFC 1918 address space. 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

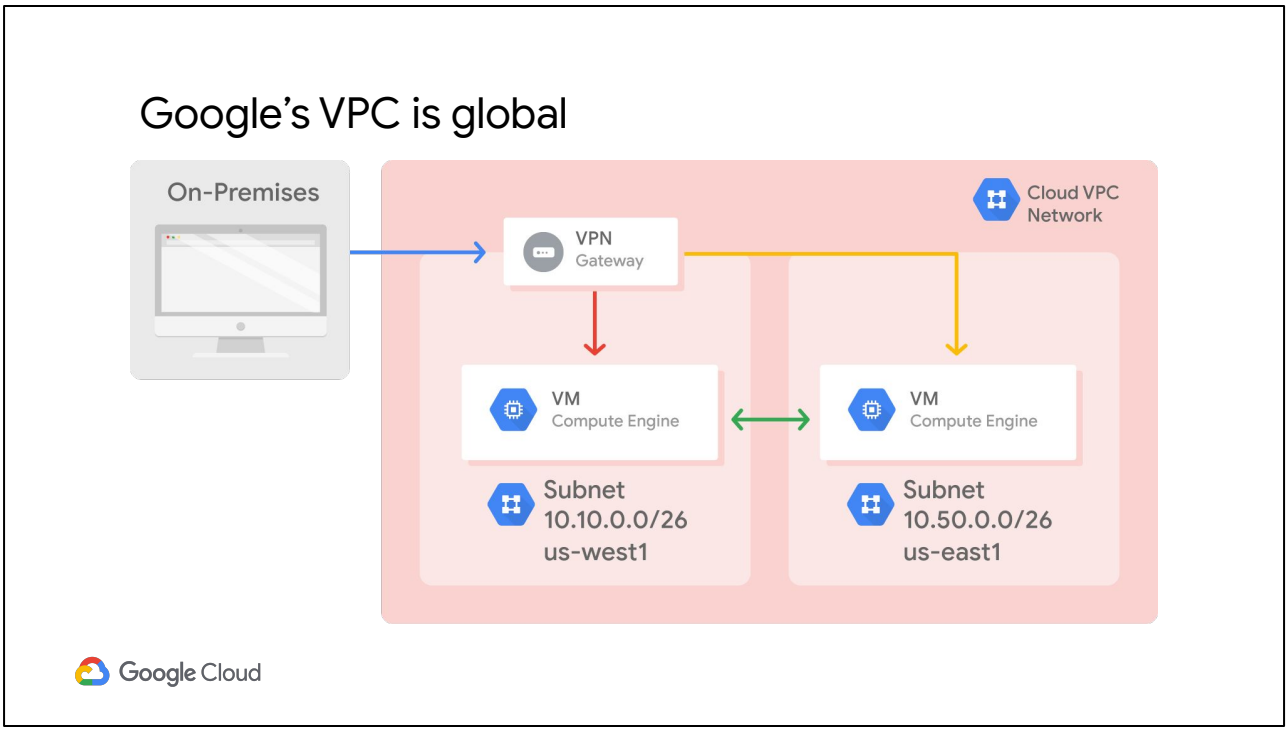
conversion is one way, meaning that custom mode networks cannot be changed to auto mode networks. So, carefully review the considerations for auto mode networks to help you decide which type of network meets your needs.



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.

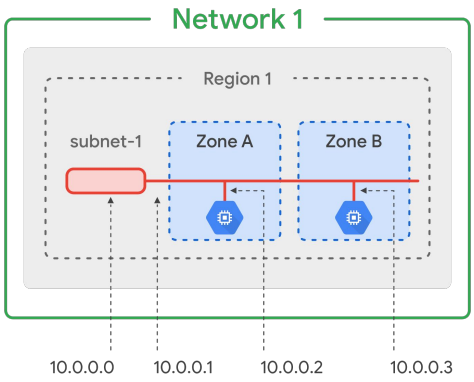
Each network contains separate virtual machines: A, B, C, and D. Because VMs A and B are in the same network, network 1, they can communicate using their internal IP addresses, even though they are in different regions. Essentially, your virtual machines, even if they exist in different locations across 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.



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 GCP 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.

Google Cloud

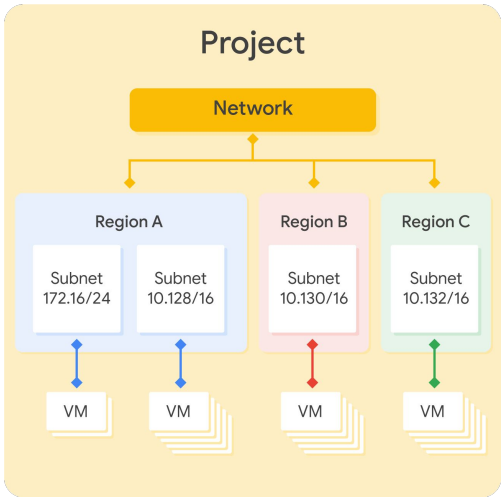
I mentioned that subnetworks work on a regional scale. Because a region contains several zones, subnetworks can cross zones.

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
- Must be inside the RFC 1918 address spaces
- Can expand but not shrink
- Auto mode can be expanded from /20 to /16
- Avoid large subnets



Google Cloud

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.
- Also, the new subnet must stay inside the RFC 1918 address spaces.
- 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.

Demo

Expand a subnet

Google Cloud

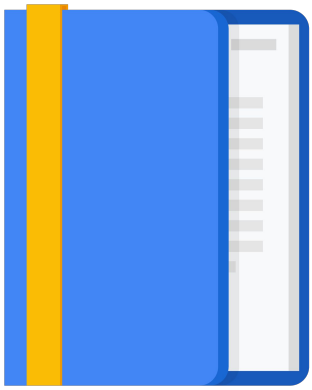
Let me show you how to expand a subnet within GCP.
I have already created a custom subnet with a /29 mask. A /29 mask provides you with 8 addresses, but of those, 4 are reserved by GCP, which leaves another 4 for your VM instances. Let's try to create another VM instance in this subnet.

[Demo]

That's how it easy it is to expand a subnet in GCP without any workload shutdown or downtime.

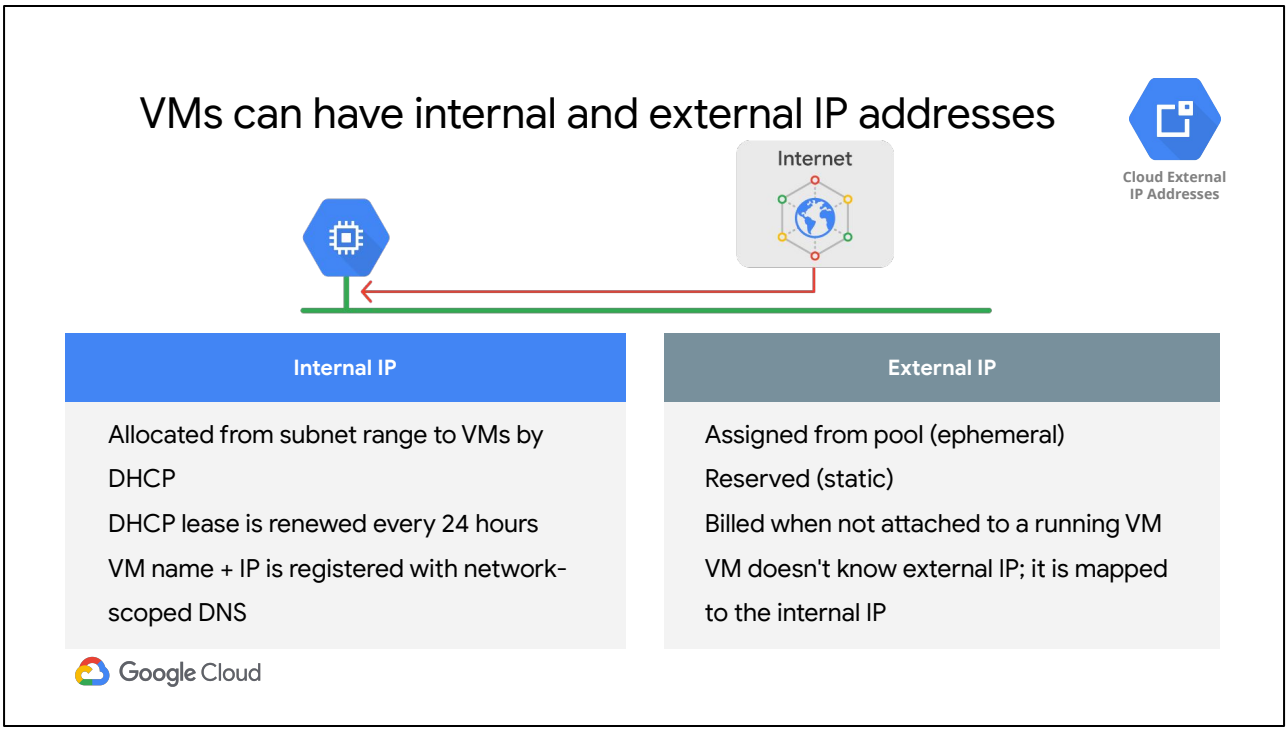
Agenda

- Virtual Private Cloud (VPC)
- Projects, networks, and subnetworks
- **IP addresses**
- Routes and firewall rules
- Pricing
- Lab
- Common network designs
- Lab



Google Cloud

Now that we covered GCP networks at a high level, let's go deeper by exploring IP addresses.



In GCP, 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 GCP, 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. Remember that you are billed for reserving external IP addresses when they are not attached to a running VM. For more information about this, see the links section of this video.
[https://cloud.google.com/compute/docs/ip-addresses/]

Demo

Internal and external IP

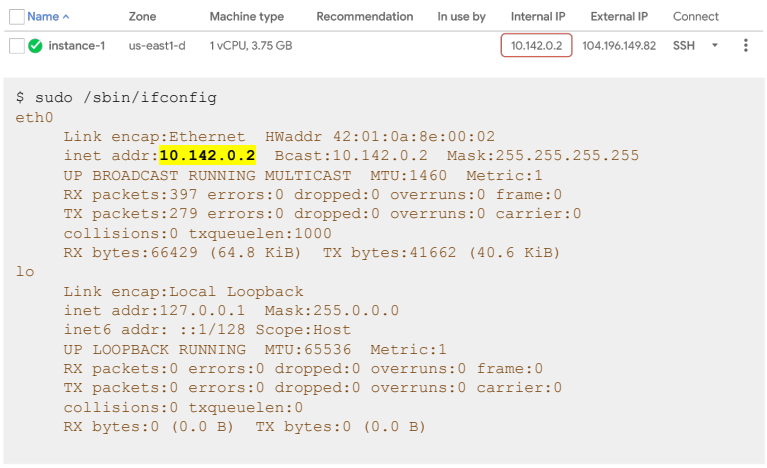
Google Cloud

I just mentioned that VMs can have internal and external IP addresses. Let's explore this in the GCP Console.

[Demo]

This demonstrates that every VM needs an internal IP address, but external IP addresses are optional, and by default they are ephemeral.

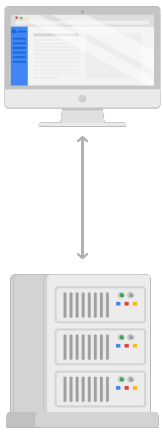
External IPs are mapped to internal IPs



Google Cloud

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. I am illustrating this here by running ifconfig within a VM in GCP, which only returns the internal IP address.
Let’s explore this further by looking at DNS resolution for both internal and external addresses.

DNS resolution for internal addresses



Each instance has a hostname that can be resolved to an internal IP address:

- The hostname is the same as the instance name.
- FQDN is [hostname].[zone].c.[project-id].internal

Example: my-server.us-central1-a.c.guestbook-151617.internal

Name resolution is handled by internal DNS resolver:

- Provided as part of Compute Engine (169.254.169.254).
- Configured for use on instance via DHCP.
- Provides answer for internal and external addresses.

Google Cloud

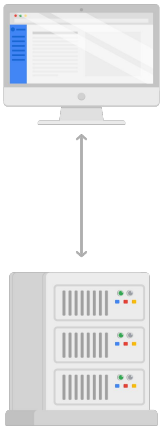
Let's start with internal addresses.

Each instance has a hostname that can be resolved to an internal IP address. This hostname is the same as the instance name. There is also an internal fully qualified domain name, or FQDN, for an instance that uses the format shown on the slide. If you delete and recreate an instance, the internal IP address can change. This change can disrupt connections from other Compute Engine resources, which must obtain the new IP address before they can connect again. However, the DNS name always points to a specific instance, no matter what the internal IP address is.

Each instance has a metadata server that also acts as a DNS resolver for that instance. The metadata server handles all DNS queries for local network resources and routes all other queries to Google's public DNS servers for public name resolution. I previously mentioned that an instance is not aware of any external IP address assigned to it. Instead, the network stores a lookup table that matches external IP addresses with the internal IP addresses of the relevant instances.

For more information, including how to set up your own resolver on instances, see the links section of this video. [https://cloud.google.com/compute/docs/vpc/internal-dns]

DNS resolution for internal addresses



- Instances with external IP addresses can allow connections from hosts outside the project.
  - Users connect directly using external IP address.
  - Admins can also publish public DNS records pointing to the instance.
    - Public DNS records are not published automatically.
- DNS records for external addresses can be published using existing DNS servers (outside of GCP).
- DNS zones can be hosted using Cloud DNS.

Google Cloud

Now, let's look at external addresses.

Instances with external IP addresses can allow connections from hosts outside of the project. Users can do so directly using the external IP address. Public DNS records pointing to instances are not published automatically; however, admins can publish these using existing DNS servers.

Domain name servers can be hosted on GCP, using Cloud DNS. This is a managed service that is definitely worth considering, so let's explore it in more detail.

Host DNS zones using Cloud DNS

Cloud DNS

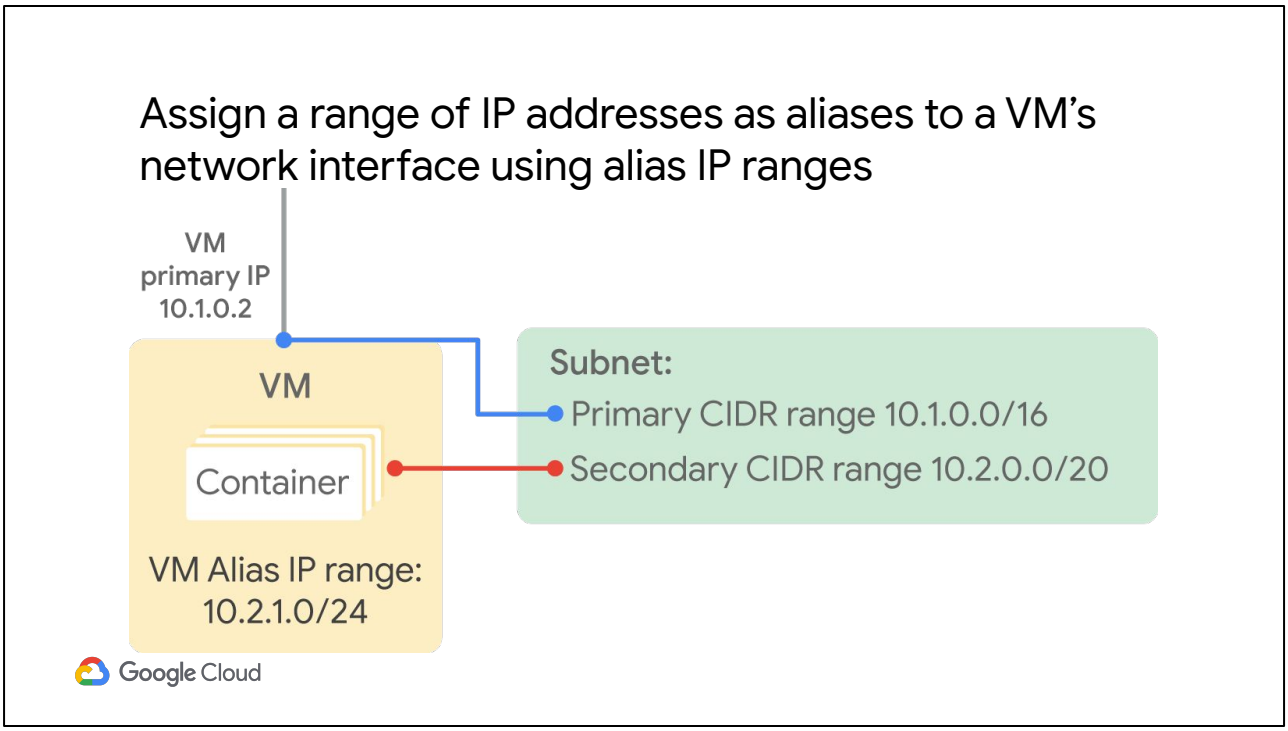
- Google's DNS service
- Translate domain names into IP address
- Low latency
- High availability (100% uptime SLA)
- Create and update millions of DNS records
- UI, command line, or API

Google Cloud

Cloud DNS is a scalable, reliable, and managed authoritative Domain Name System, or DNS, service running on the same infrastructure as Google. Cloud DNS translates requests for domain names like google.com into IP addresses.

Cloud DNS uses Google's global network of Anycast name servers to serve your DNS zones from redundant locations around the world, providing lower latency and high availability for your users. High availability is very important because if you can't look up a domain name, the internet might as well be down. That's why GCP offers a 100% uptime Service Level Agreement, or SLA, for domains configured in Cloud DNS. For more information about this SLA, see the links section of this video. [https://cloud.google.com/dns/sla]

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 about Cloud DNS, see the links section of this video. [https://cloud.google.com/dns/docs/]



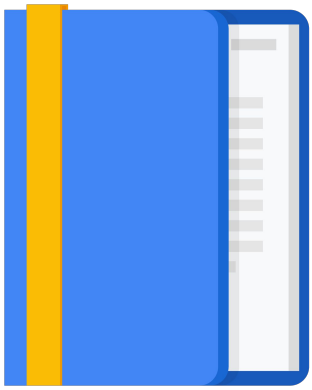
Another networking feature of GCP is Alias IP Ranges.

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, see the links section of this video.
[https://cloud.google.com/compute/docs/alias-ip/]

Agenda

- Virtual Private Cloud (VPC)
- Projects, networks, and subnetworks
- IP addresses
- **Routes and firewall rules**
- Pricing
- Lab
- Common network designs
- Lab



Google Cloud

So far you've learned about projects, networks, subnetworks, and IP addresses. Let's use what you learned to understand how GCP routes traffic.

A route is a mapping of an IP range to a destination

Every network has:

- Routes that let instances in a network send traffic directly to each other.
- A default route that directs packets to destinations that are outside the network.

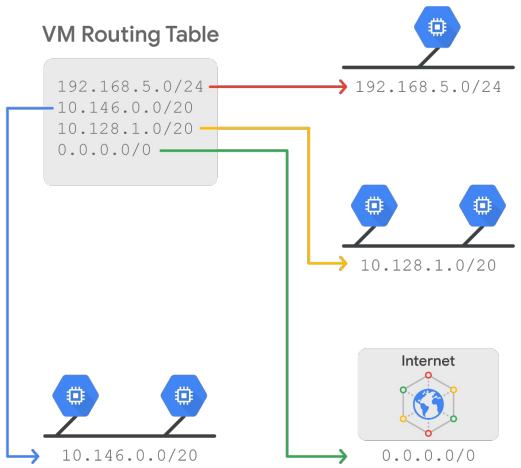
Firewall rules must also allow the packet.

Google Cloud

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

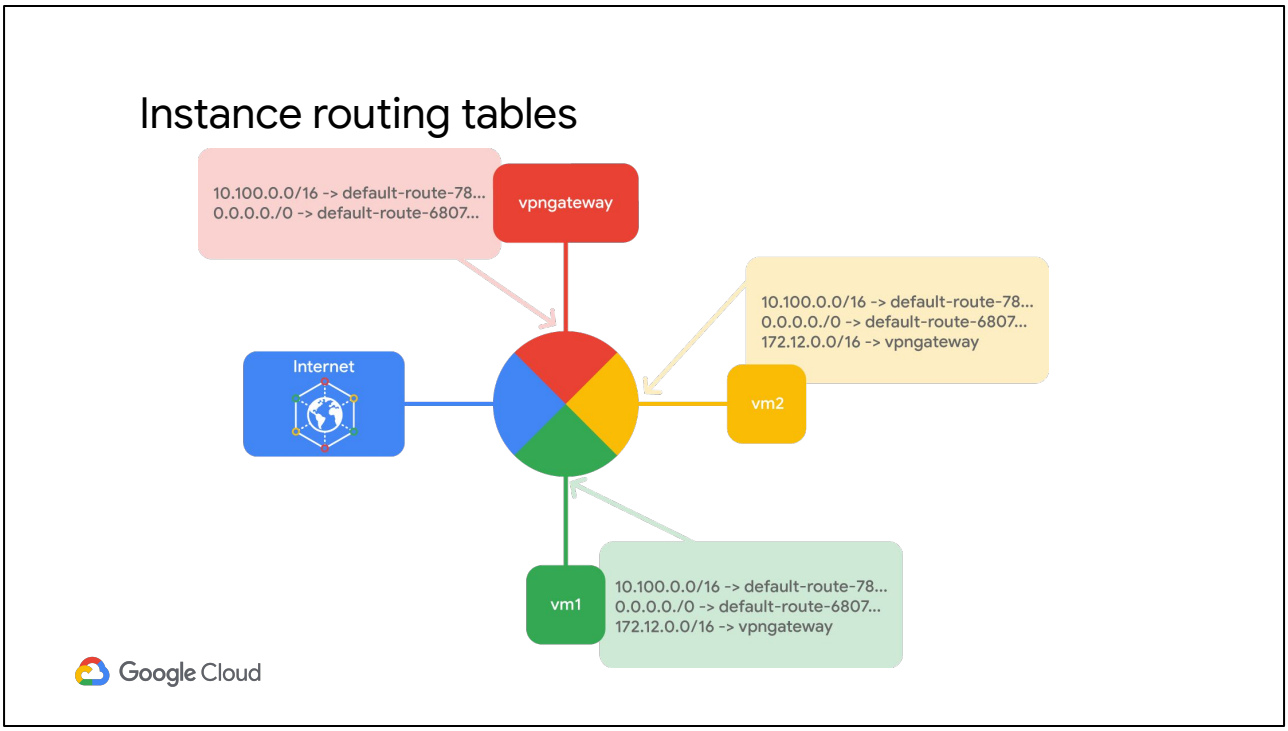
- Apply to traffic egressing a VM.
- Forward traffic to most specific route.
- Are created when a subnet is created.
- Enable VMs on same network to communicate.
- Destination is in CIDR notation.
- Traffic is delivered only if it also matches a firewall rule.



Google Cloud

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.



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.

Google Cloud

GCP 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.

GCP 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.

Routes map traffic to destination networks

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

Google Cloud

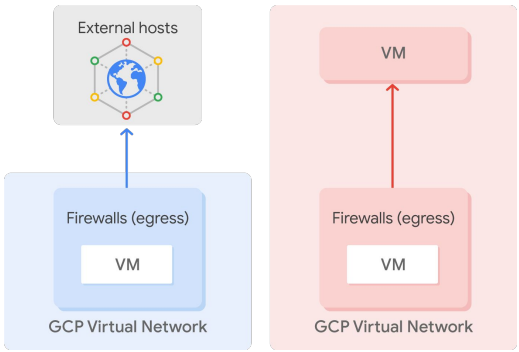
You can express your desired firewall configuration as a set of firewall rules. Conceptually, a firewall rule is composed of the following parameters:

- 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, please refer to links section of this video.
[https://cloud.google.com/compute/docs/vpc/firewalls#firewall_rule_components]

Let’s look at some GCP firewall use cases for both egress and ingress.

GCP firewall use case: Egress



Conditions:

- Destination CIDR ranges
- Protocols
- Ports

Action:

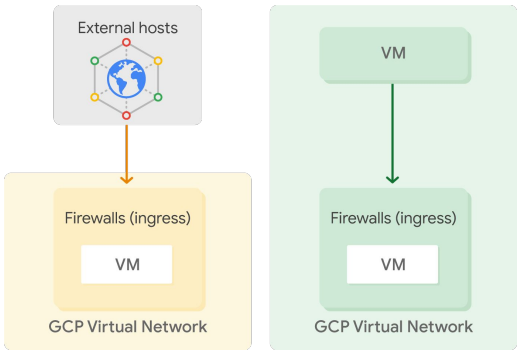
- Allow: permit the matching egress connection
- Deny: block the matching egress connection

Google Cloud

Egress firewall rules control outgoing connections originated inside your GCP 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 GCP 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.

GCP firewall use case: Ingress



Conditions:

- Source CIDR ranges
- Protocols
- Ports

Action:

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

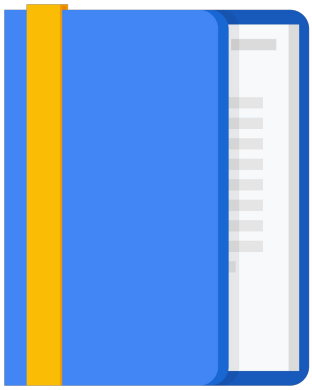
Google Cloud

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 GCP 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.

Agenda

- Virtual Private Cloud (VPC)
- Projects, networks, and subnetworks
- IP addresses
- Routes and firewall rules
- **Pricing**
- Lab
- Common network designs
- Lab



Google Cloud

Before you apply what you just learned, let's talk about network pricing. It is important that you understand the circumstances in which you are billed for GCP's network.

Network pricing (subject to change)

Traffic type	Price
Ingress	No charge
Egress to the same zone (internal IP address)	No charge
Egress to Google products (YouTube, Maps, Drive)	No charge
Egress to a different GCP service (within same region; exceptions)	No charge
Egress between zones in the same region (per GB)	$0.01
Egress to the same zone (external IP address, per GB)	$0.01
Egress between regions within the US and Canada (per GB)	$0.01
Egress between regions, not including traffic between US regions	Varies by region

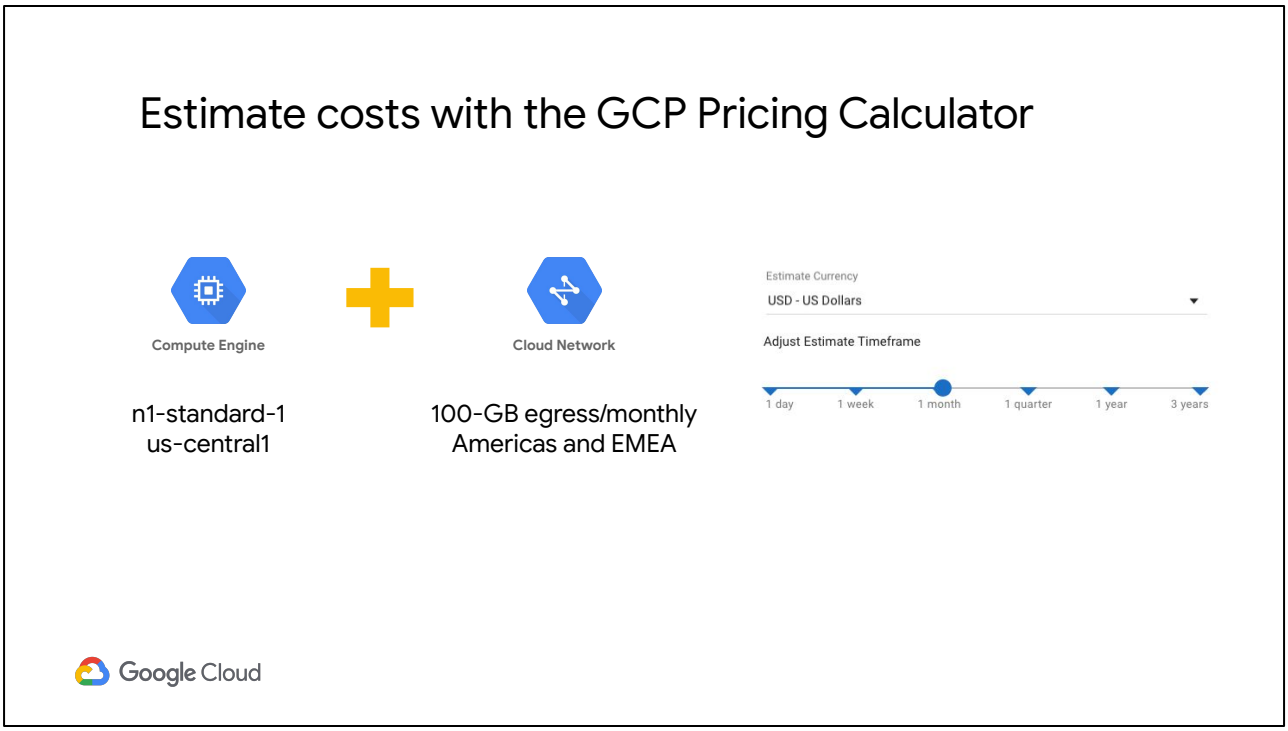
Google Cloud

This table is from the Compute Engine documentation, and it lists the price of each traffic type.

First of all, ingress or traffic coming into GCP's network is not charged, unless there is a resource such as a load balancer that is processing ingress traffic. Responses to requests count as egress and are charged.

The rest of this table lists egress or traffic leaving a virtual machine. Egress traffic to the same zone is not charged, as long as that egress is through the internal IP address of an instance. Also, egress traffic to Google products, like YouTube, Maps, Drive, or traffic to a different GCP service within the same region is not charged for. However, there is a charge for egress between zones in the same region, egress within a zone if the traffic is through the external IP address of an instance, and egress between regions.

As for the difference in egress traffic to the same zone, Compute Engine cannot determine the zone of a virtual machine through the external IP address. Therefore, this traffic is treated like egress between zones in the same region. Also, there are some exceptions, and pricing can always change, so see the documentation in the links section of this video [https://cloud.google.com/compute/all-pricing#general_network_pricing]



I recommend using the GCP pricing calculator to estimate the cost of a collection of resources, because each GCP service has its own pricing model. The pricing calculator is a web-based tool that you use to specify the expected consumption of certain services and resources, and it then provides you with an estimated cost.

For example, you can specify a specific instance type in a specific region along with 100-GB of monthly egress traffic to Americas and EMEA. The pricing calculator then returns the total estimated cost. You can adjust the currency and time frame to meet your needs, and when you finish, you can email the estimate or save it to a specific URL for future reference.

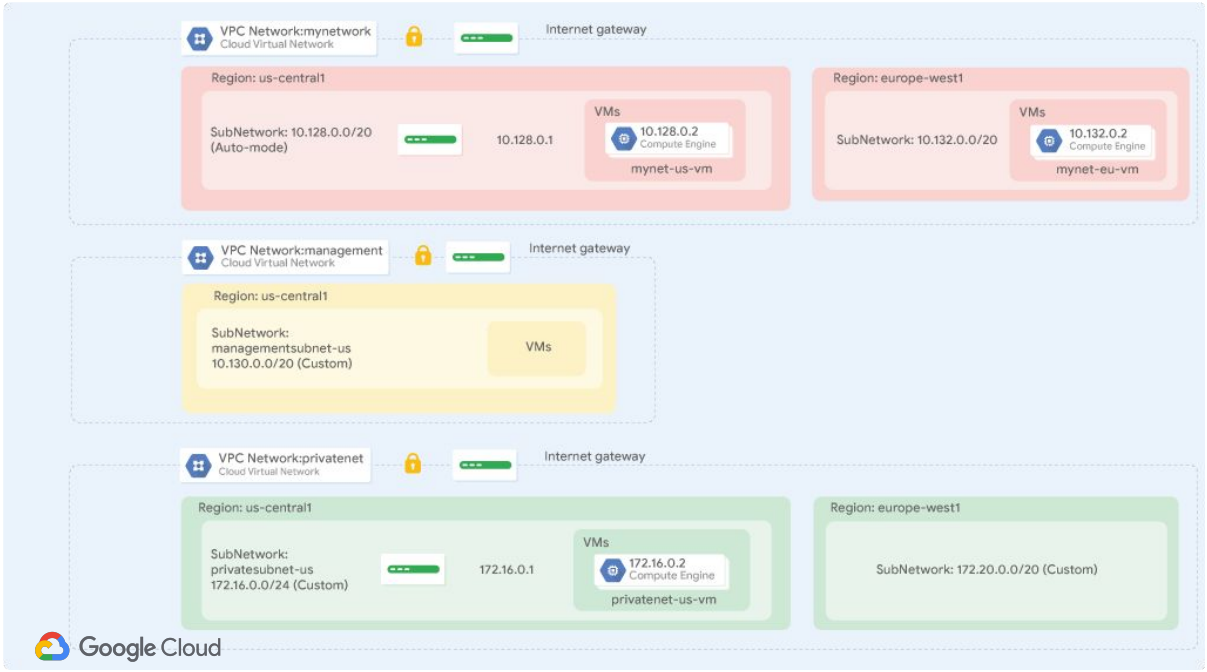
To use the pricing calculator today, refer to the links section of this video. [https://cloud.google.com/products/calculator/]

Lab

VPC Networking

Google Cloud

Let’s apply some of the network features we just discussed in a lab.



In this lab, you create an auto mode VPC network with firewall rules and two VM instances. Then, you convert the auto mode network to a custom mode network and create other custom mode networks, as shown in the network diagram. You also explore the connectivity across networks.

Lab review

VPC Networking

Google Cloud

In this lab, you explored the default network and determined that you cannot create VM instances without a VPC network. So you created a new auto mode VPC network with subnets, routes, firewall rules, and two VM instances and tested the connectivity for the VM instances. Because auto mode networks aren't recommended for production, you converted the auto mode network to a custom mode network.

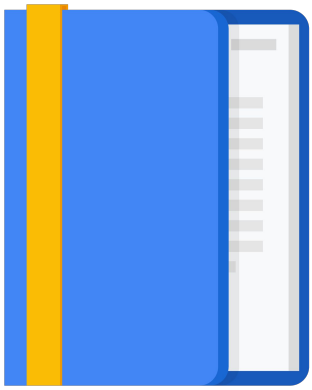
Next, you created two more custom mode VPC networks with firewall rules and VM instances using the GCP Console and the gcloud command line. Then you tested the connectivity across VPC networks, which worked when you pinged external IP addresses, but not when you pinged internal IP addresses.

VPC networks are by default isolated private networking domains. Therefore, no internal IP address communication is allowed between networks unless you set up mechanisms such as VPC peering or a VPN connection.

You can stay for a lab walkthrough, but remember that GCP's user interface can change, so your environment might look slightly different.

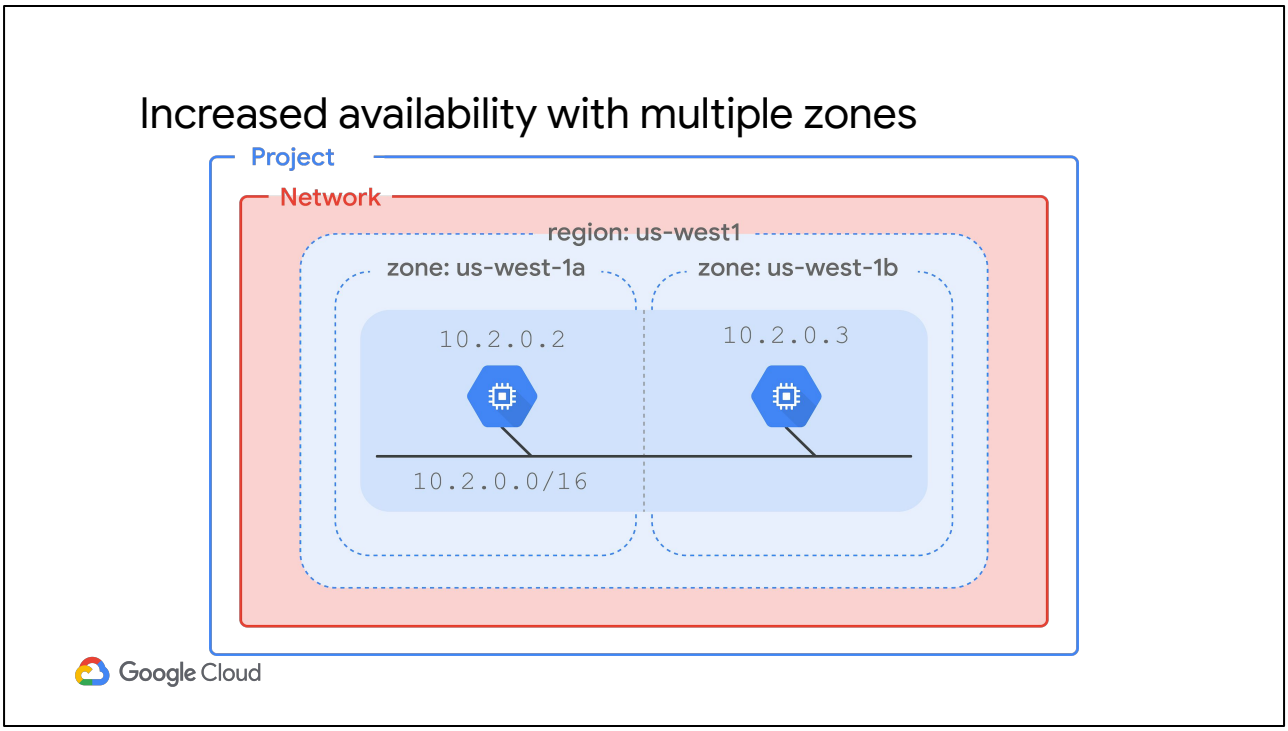
Agenda

- Virtual Private Cloud (VPC)
- Projects, networks, and subnetworks
- IP addresses
- Routes and firewall rules
- Pricing
- Lab
- **Common network designs**
- Lab



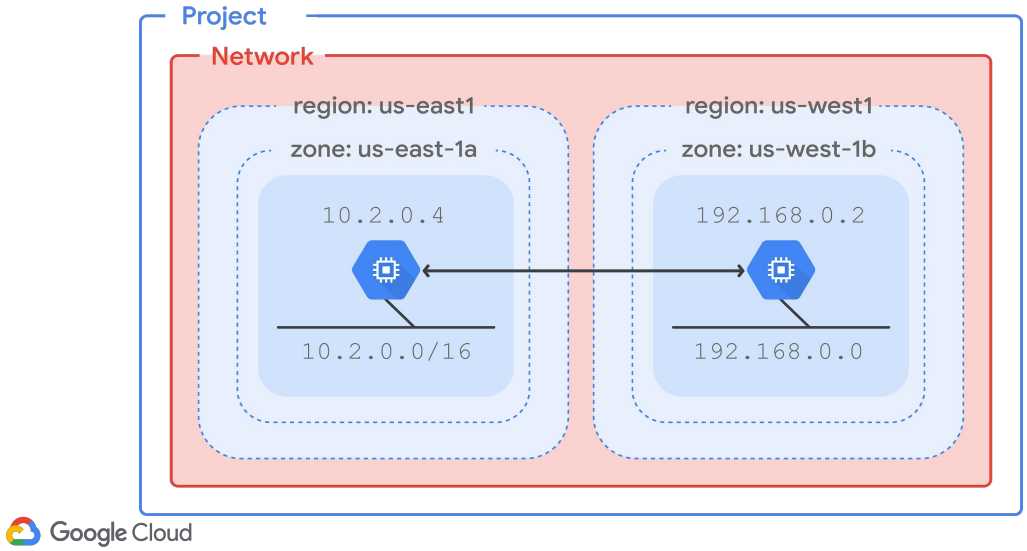
Google Cloud

Let’s use what we have learned so far and look at common network designs. Now, “common” is a fairly relative term. While I could spend all day talking about network designs, I have picked a handful of designs that best relate to this module.



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

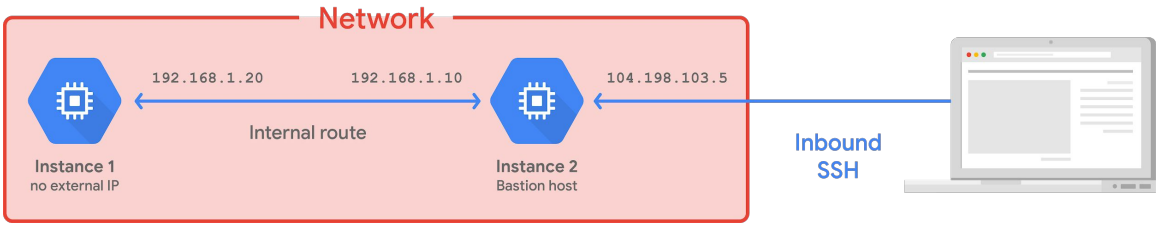


Google Cloud

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 series.

Instance isolation with bastion host



Google Cloud

Another common network design is a bastion host isolation. A bastion host provides an external facing point of entry into a network containing private network instances. This host can provide a single point of fortification or audit and can be started and stopped to enable or disable inbound SSH communication from the internet.
For example, on this slide, instance 1 represents a service provided to an internal corporate audience. Therefore, this instance does not have an external IP address. In order to gain access to this instance, you create a maintenance host, known as a bastion host.

You will configure such a network design in the upcoming lab and verify connectivity.

Lab

Bastion Host

Google Cloud

In this lab, you create a bastion host to enable maintenance access to a virtual machine. Specifically, you create an application web server to represent a service provided to an internal corporate audience. Then, to prevent the web server from access to or from the internet, you remove its external IP address. Finally, you create the maintenance server to gain access to and verify internal connectivity to the application server.

Lab review

Bastion Host

Google Cloud

In this lab you created a webserver VM and restricted access to it by removing the external IP address. Then, you created a bastion host to gain access to the webserver VM over its internal IP address. Normally, you would harden the bastion host by editing the firewall rules to restrict the source IPs that can access the bastion host. Also, when you're not using the bastion host, you would shut it down.

Using Cloud IAP, you can now enable context-aware access to VMs via SSH and RDP without bastion hosts. For more information on this, refer to the links section of the video:
[https://cloud.google.com/blog/products/identity-security/cloud-iap-enables-context-aware-access-to-vms-via-ssh-and-rdp-without-bastion-hosts]

You can stay for a lab walkthrough, but remember that GCP's user interface can change, so your environment might look slightly different.

Review

Virtual Networks

Google Cloud

In this module, I gave you an overview of Google's Virtual Private Cloud. We looked at the different objects within VPC, like projects, networks, IP addresses, routes, and firewall rules. I also provided a brief overview of how your network design choices can affect billing.
Then you applied the different concepts that we covered in a thorough lab.
Next, we looked at common network designs, such as, a bastion host which you got to implement in a lab.

Now that you have a solid understanding of how GCP has implemented networking, let's move on to learn more about other services. Next up is Compute Engine, which offers scalable, high-performance virtual machines