

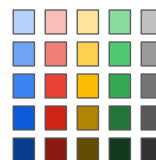
Google Cloud Platform

# Virtual Networks

v 1.0



Virtual  
Networks



# Agenda

- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# Cloud Virtual Network (CVN) Objects

- Projects
- Networks
  - Default, auto, custom
- Subnetworks
- Regions
- Zones
- IP addresses
  - Internal, External, range
- Virtual Machines (VMs)
- Routes
- Firewall rules

These 9 objects are all you need to know to understand the fundamentals of Cloud Virtual Networks.

IP Forwarding, Protocol Forwarding, Load Balancing, Cloud DNS, and VPN Tunnels are built on top of this framework and are covered separately, not in this module.

# Agenda

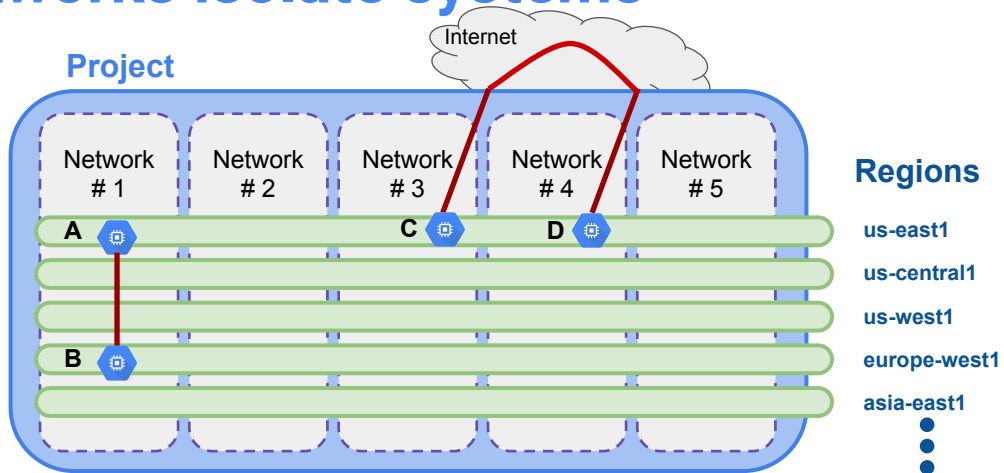
- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# Projects and networks

- Project
  - Associates objects and services with billing
  - Contains networks (quota max 5)
- Network
  - Has no IP address range
  - Is global and spans all available regions
  - Contains subnetworks
  - Types of Networks
    - Default, auto, custom
    - auto can be converted to custom but *once custom, always custom*

A fourth type of network called "Legacy Google Compute Engine (GCE)" exists but is not covered here and is not recommended.

# Networks isolate systems

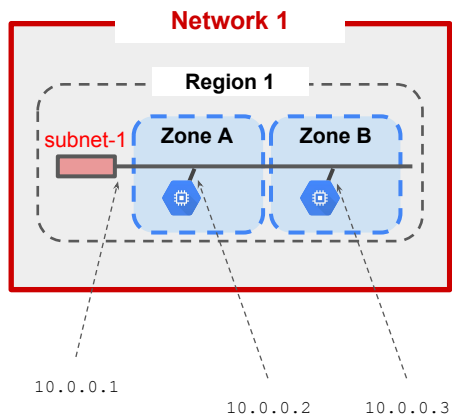


- A and B can communicate over Internal IPs *even though they are in different regions*
- C and D must communicate over External IPs *even though they are in the same region*

You can use networks to isolate systems. If you want to make sure that System C can never communicate with System D except over the Internet, then place them in separate networks.

Instances in the same network, such as A and B, can communicate with each other over Internal IPs even though they are in different regions.

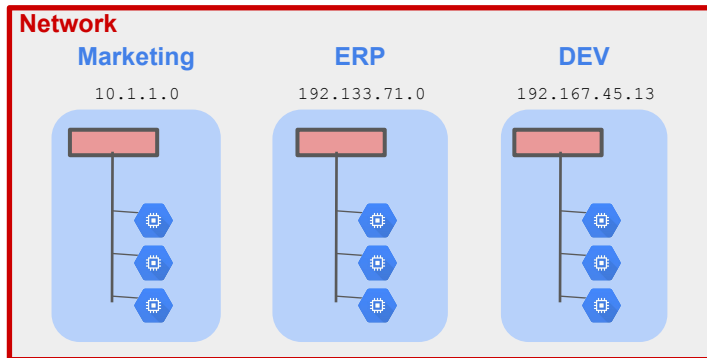
# Subnetworks cross zones



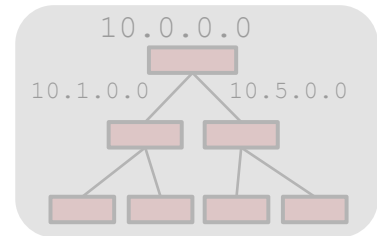
- Subnetworks can extend across zones in the same region
- One VM and an alternate VM can be on the same subnet but in different zones
- A single firewall rule can apply to both VMs even though they are in different zones

- Defined by Internal IP address prefix range
- Specified in CIDR notation
  - IP ranges cannot overlap between subnets
  - IP range can be expanded but can never shrink
  - Specific to one region
- Can cross zones within the region
- Notice that the first address in the range, 10.0.0.1 is reserved for the "router" address. And the last address in the range 10.0.0.255 is reserved for the "broadcast" address. However, broadcast and multicast traffic are currently not supported in CVN,

# Subnetworks are for managing resources



Physical Network Hierarchy



Networks have no IP range, so subnetworks don't need to fit into an address hierarchy. Instead, subnetworks can be used to group and manage resources. They can represent departments, business functions, or systems

In physical networks, subnetworks must fit into a tree-shaped hierarchy composed of progressively more specific IP prefix ranges.

In the example, the top-level network IP range is 10.0.0.0, with narrow subnets 10.1.0.0 and 10.5.0.0, and progressively narrower ranges. Data arrives at the network level and is routed to more specific subnets until it reaches the destination subnet. A subnet has to fit into the larger hierarchy to receive traffic. Machines that are components of a system are likely to be distributed within the network based on where in the network bandwidth is available or based on separation of a primary machine from its backup for availability. They are not likely to be located on one or a few subnets.

In the Cloud Virtual Network that physical network structure doesn't exist. The network has no top-level IP range. Subnetworks receive traffic without having to fit into a tree-shaped topology. And bandwidth does not depend on location in the hierarchy. Unbounded by traffic-routing requirements, subnetworks take on a different role in CVN; they become resource management tools.

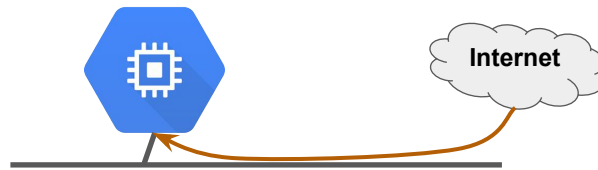
Subnets are ways to group similar or related resources. In the example, one subnet represents a company organization, "Marketing", another subnet represents a business function, Electronic Resource Planning (ERP), and another subnet represents a technology system, Continuous Integration Development Environment (CIDE). Resource monitoring and management, and policy implementation, becomes simplified when related VMs are located in a few subnets defined for that purpose.



# Agenda

- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# IP addresses




| Internal IP  | External IP   |
|--|---|
| Allocated from subnet range to VMs by DHCP         | Assigned from pool (ephemeral)                                |
| DHCP reservation is renewed every 24 hours         | Reserved (static)<br>Billed when not attached to a running VM |
| VM name + IP is registered with network-scoped DNS | VM doesn't know External IP, it is mapped to the Internal IP  |

VMs are allocated Internal IP addresses from the prefix range of a subnetwork by DHCP and renewed every 24 hours. VMs are assigned External IP addresses either from a pool (ephemeral) or can be assigned a reserved External IP (static). In either case, the external address is unknown to the VM and is mapped to the VM's internal address transparently by the Cloud Virtual Network.

Static IPs that are assigned to a VM or a Load Balancer are not charged at all.  
<https://cloud.google.com/compute/pricing>

When you create VMs in GCP, their symbolic name is registered with an internal DNS service that translates the name to the Internal IP address. DNS is scoped to the Network and connected to Internet DNS. 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.

# External IPs are mapped to Internal IPs

| <input type="checkbox"/> Name ^   | Zone       | Machine type    | Recommendation | In use by | Internal IP | External IP    | Connect |
|---|------------|-----------------|----------------|-----------|-------------|----------------|---------|
| <input type="checkbox"/>  instance-1   | us-east1-d | 1 vCPU, 3.75 GB |                |           | 10.142.0.2  | 104.196.149.82 | SSH ▾ ⋮ |
| <pre>\$ sudo /sbin/ifconfig eth0     Link encap:Ethernet  HWaddr 42:01:0a:8e:00:02     inet addr:10.142.0.2  Bcast:10.142.0.2  Mask:255.255.255.255     UP BROADCAST RUNNING MULTICAST  MTU:1460  Metric:1     RX packets:397 errors:0 dropped:0 overruns:0 frame:0     TX packets:279 errors:0 dropped:0 overruns:0 carrier:0     collisions:0 txqueuelen:1000     RX bytes:66429 (64.8 KiB)  TX bytes:41662 (40.6 KiB)  lo     Link encap:Local Loopback     inet addr:127.0.0.1  Mask:255.0.0.0     inet6 addr: ::1/128 Scope:Host     UP LOOPBACK RUNNING  MTU:65536  Metric:1     RX packets:0 errors:0 dropped:0 overruns:0 frame:0     TX packets:0 errors:0 dropped:0 overruns:0 carrier:0     collisions:0 txqueuelen:0     RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)</pre> |            |                 |                |           |             |                |         |

When an external IP address is assigned, the VM has no knowledge of the address, instead an Internal IP is mapped to the External IP

# DNS Resolution for Internal Addresses

- Each instance has a hostname that can be resolved to an internal IP address
  - Hostname is the same as the instance name
  - FQDN is `[hostname].c.[project-id].internal`
    - Example: `guestbook-test.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
  - You can use an alternative DNS resolver if you prefer

You can view instance names, which are used in hostnames, via...

- Console:  
[https://console.cloud.google.com/compute/instances?project=\[project\\_id\]](https://console.cloud.google.com/compute/instances?project=[project_id])
- gcloud: **compute instances list**
- API: **GET /project/zones/zone/instances**

When resolving hostnames for hosts outside of GCP, the resolver forwards queries to the standard Google DNS servers.

For more information, including how to set up your own resolver on instances, see:

[https://cloud.google.com/compute/docs/networking#internal\\_dns\\_and\\_resolution](https://cloud.google.com/compute/docs/networking#internal_dns_and_resolution)

# DNS Resolution for External Addresses

- Instances with external IP addresses can allow connections from hosts outside of the project
  - Users connect directly using external IP address
  - Admins can also publish public-DNS records pointing to 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 Google Cloud DNS
  - Create zone and configure domain DNS to use
  - Create, update, remove records manually or via API

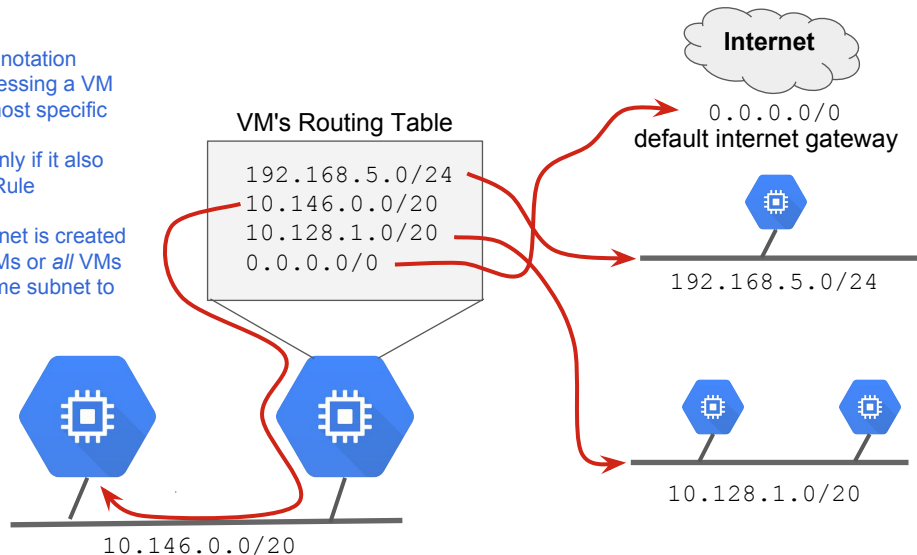
For more details on using Google Cloud DNS, see <https://cloud.google.com/dns/docs/>.

# Agenda

- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# Routes map traffic to destination networks

- Destination in CIDR notation
- Applies to traffic egressing a VM
- Forwards traffic to most specific route
- Traffic is delivered only if it also matches a Firewall Rule
- Created when a subnet is created
- Applies to *tagged* VMs or *all* VMs
- Enables VMs on same subnet to communicate



- Specified by CIDR notation
- A route is created when a network is created
  - Enables delivery of traffic from "anywhere"
- A route is created when a subnet is created
  - Enables VMs on the same subnet to communicate
- Traffic is forwarded to the most specific route

However, no traffic will flow without also matching a firewall rule

Routes match packets by destination IP address. More specific IP ranges are preferred over larger ranges. If there is a tie, the route with the smallest priority value is chosen. If there is still a tie, layer 3 and 4 headers are used to select one route.

[https://cloud.google.com/compute/docs/networking#network\\_routes](https://cloud.google.com/compute/docs/networking#network_routes)

# Firewall rules allow traffic

- Positive logic; "allow" rules, no "deny" rules
  - Rules can match IP address or IP range
- Tags
  - Rules can match Tags (source and target)
  - Tags user-defined strings
  - Tags are applied to VMs (*you can't tag an IP address*)
  - Group VMs for policy
    - Not limited by topology (as are IPs)
- Default rules are created for auto-type networks
  - but not custom-type networks

Firewall Rules use positive logic; they allow traffic, there are no deny Rules. A Rule can match either IP addresses or match Source and Destination Tags.

Before March 7, 2017, a Tag was simply the Key portion of the Key:Value pair of a Label. A Tag or Label could be applied to a VM. And the two namespaces were kept in synch.

After March 7, 2017, the two namespace were separate. Labels are used to identify resources across GCP and propagate through billing.

VM Tags are used to apply network policy (firewall rules).

Tags are not limited to the topology as are IP addresses. For example, you could Tag primary and failover VMs to be part of "production" even though they are in separate networks on different subnets. That makes firewall rules more efficient to develop and apply.



# Routes and firewall rules

- Routes and firewall rules are network resources
  - Cannot be shared between projects or networks
  - Maintained in network tables
- Traffic Engineering (TE)
  - Subset of routes and rules for one VM is derived from tables
  - Applied to traffic as it exits the VM
  - Traffic not matching both a route and a rule is dropped
  - Allowed traffic with a route is delivered to its target
    - VM, service, or Internet PoP
    - *Not routing or switching*

It is convenient to think of each subnetwork as having a virtual gateway that contains a virtual router and a virtual firewall. The gateway address is the lowest IP address on the subnetwork.

So you can think of configuring routes and rules in this virtual gateway. However, in the network infrastructure, the routes and rules are contained in network-wide tables and filtered to produce VM-specific views that are used when traffic exits the VM.

It is convenient to think of packets being routed or switched through a concealed network topology within the cloud. However, packets actually enter the Cloud Virtual Network fabric where they are directly delivered to the target VM, target service, or external Point of Presence (PoP) on the Internet. So no "hops" actually occur and there is no physical "path".

# Firewalls Connection Control <sup>Alpha</sup>

- Default & custom network
  - default-deny-all-ingress
  - default-allow-all-egress
- Ingress firewall rules
  - deny
  - Second source filter allow source range and tag for simpler rules
- Egress firewall rules
  - Matches destination IP CIDR ranges, protocols, ports and target tags
- Priority
  - Ranks rules

# Agenda

- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → **Billing**
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# Network billing

- Billed for traffic *egress*
  - to the Internet (varies by region)
  - from one region to another (in the same network)
    - different rates for same continent regions vs intercontinental
  - between zones within a region
- Not billed for
  - traffic *ingress*
  - VM to VM traffic in a single zone (same region, network)
  - traffic to GCP services (*limits apply, see documentation*)

<https://cloud.google.com/compute/pricing#network>

Note: about "between zones within a region".

If you have a High Availability solution with components in different zones, and part of the design requires making frequent RPC calls from one zone to the other, zone egress costs could accumulate. In this circumstance, consider using Cloud Pub/Sub instead of the RPC calls for a more efficient solution.

# Network considerations

- CVN throughput and round-trip latency between VMs
  - Varies with location
  - Consider requirements
  - See documentation for current specifics
- CVN is constantly evolving
  - Any feature marked BETA has no Service Level Agreement (SLA)
  - For more information see SLA for specific feature

Throughput and latency are critical factors in conventional networks.

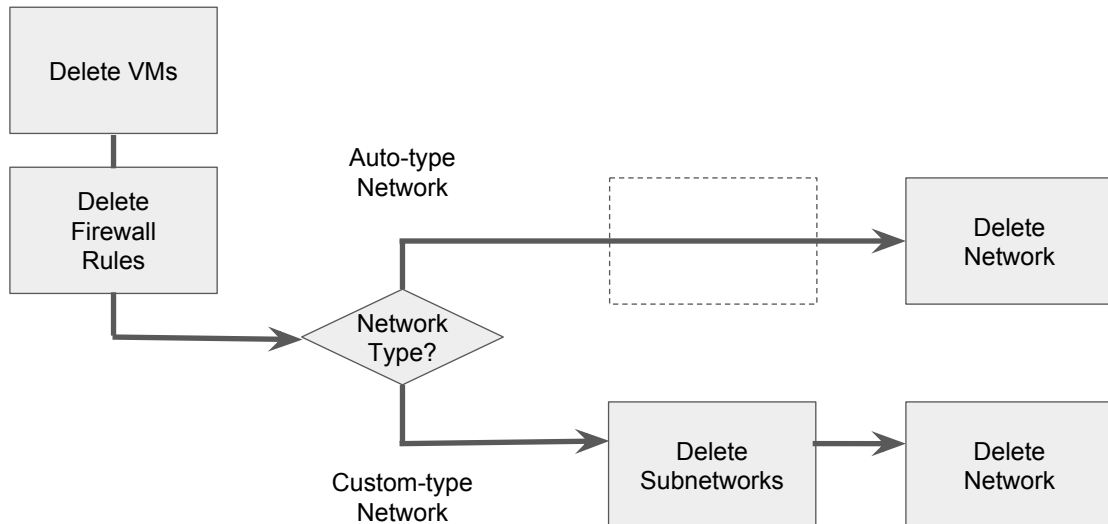
CVN's high-throughput and low round-trip latency may influence these factors in your design.

Check your application's requirements against current specifics for CVN.

**Note:**

VM-to-VM within a single zone has much better performance and much more consistent performance than VM-to-VM between regions in a single continent, or communications that span continents.

# How to delete networks and subnetworks



- Subnetworks
  - all VMs must be deleted first
  - all firewall rules must be deleted first
  - can be deleted on custom-type networks
  - cannot be deleted on auto-type networks
- Networks
  - All VMs and firewall rules on all subnetworks must be deleted first
  - On an auto-type network, only the entire network can be deleted, not the subnetworks.

# Agenda

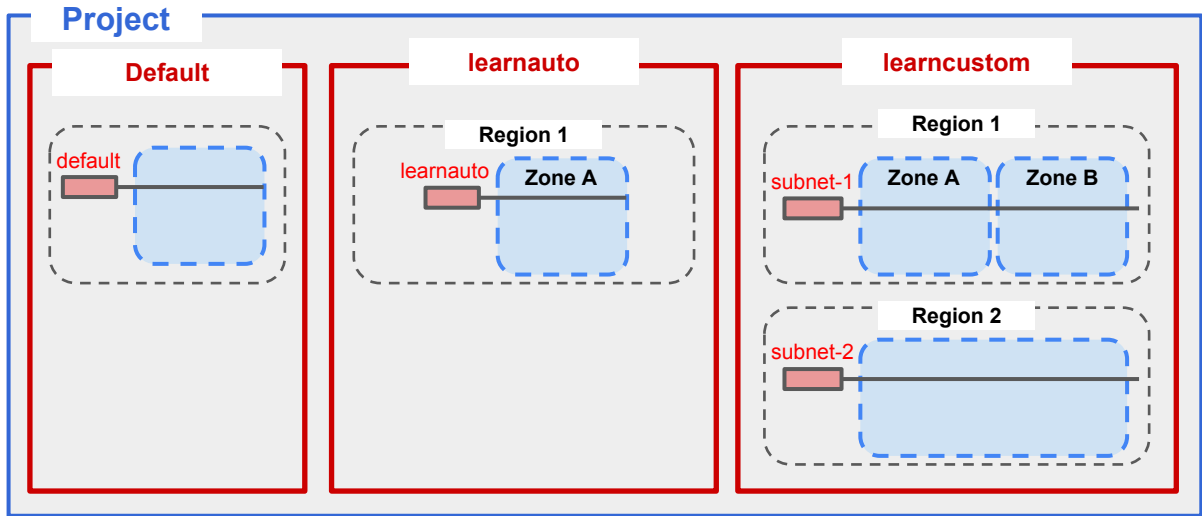
- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# Lab #1 - Cloud Virtual Networking

In this lab you will create a project, networks of various types, and subnetworks. You will launch VMs in various locations in the topology. You will use `ping`, `tracert`, and `ssh` to evaluate connectivity. You will manage the network routes and firewall rules to meet policy requirements. You will also learn to expand the number of VMs that a subnet can support.



# Networks and Subnets

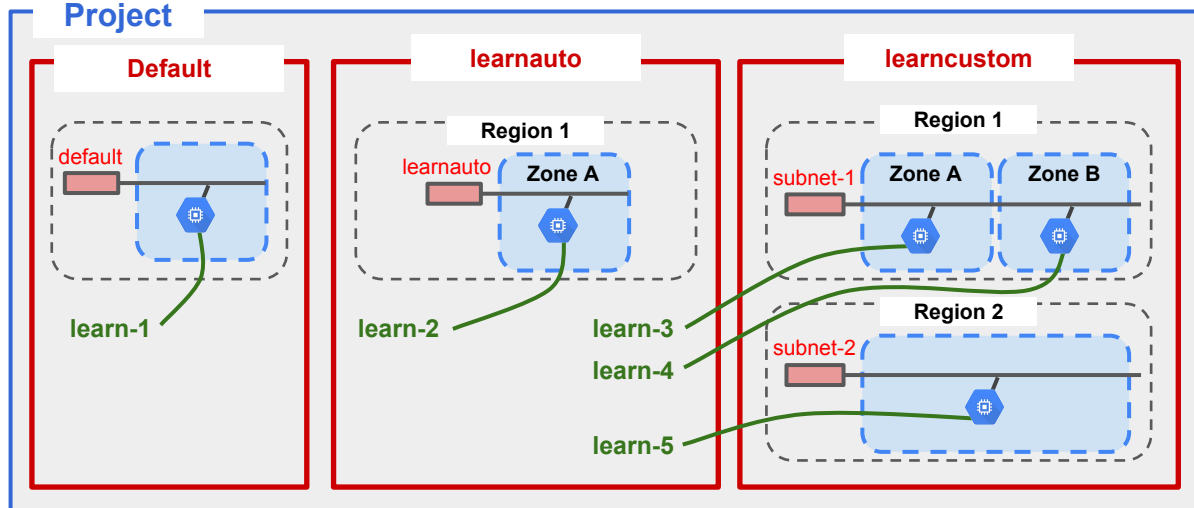


In the first part of the lab you will build this complex multiple-network topology.

# VMs



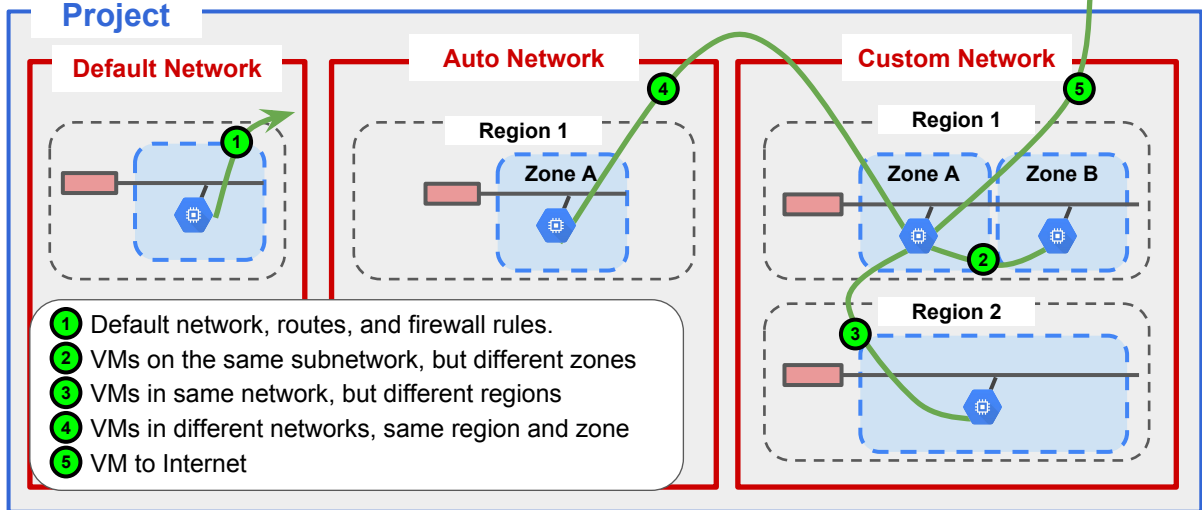
## Project



In the second part of the lab, you will launch VMs in the various regions and subnets. Having the VMs in a variety of locations will enable you to explore connectivity across and within the multiple networks.

# Routes and Firewall Rules

## Project



In the third part of the lab, you will use ping, traceroute, and ssh to test connectivity. You will modify the firewall rules to meet policy requirements, and test to verify that the changes worked.

## Lab Review

- You created a topology with multiple networks and subnetworks
- Create routes
- Created firewall rules
- Converted an auto to a custom network
- Expanded a subnet range

## Lab #2 - Bastion Host

Best practices with firewall rules and Bastion Host.

In the previous lab you explored how Cloud Virtual Networks work. In this lab you perform basic tasks to secure access to your VMs building on what you learned in the previous lab.

# Agenda

- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → **Common network designs**
- 8 → Review

# Common network designs

How do all these elements work together?

- Projects
- Networks
- Subnetworks
- Regions
- Zones

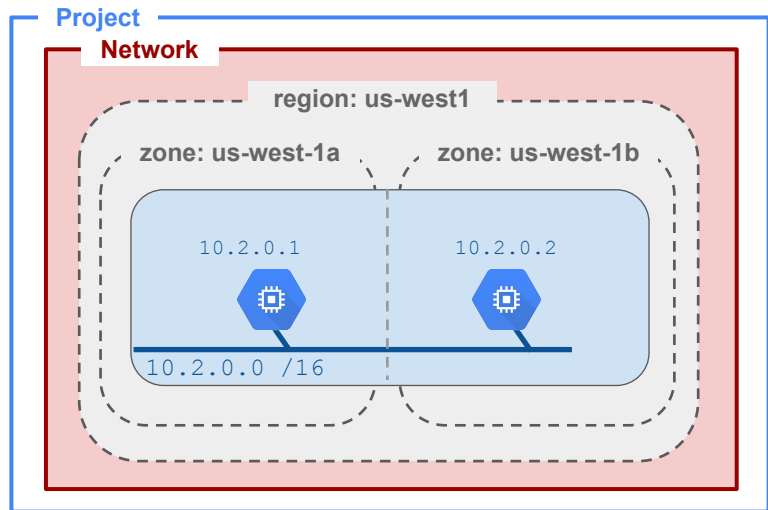
They provide a rich set of alternatives for managing groups of resources with varying availability and access control requirements

# Availability

One Project  
One Network  
One Region  
**One Subnetwork**  
**Multiple zones**

Increased availability due to multiple zones

Simplified security due to a single subnetwork



The point here is that you can write firewall rules that apply to all VMs on a single subnet. By allocating VMs on a single subnet to separate zones, you get improved availability without additional security complexity.

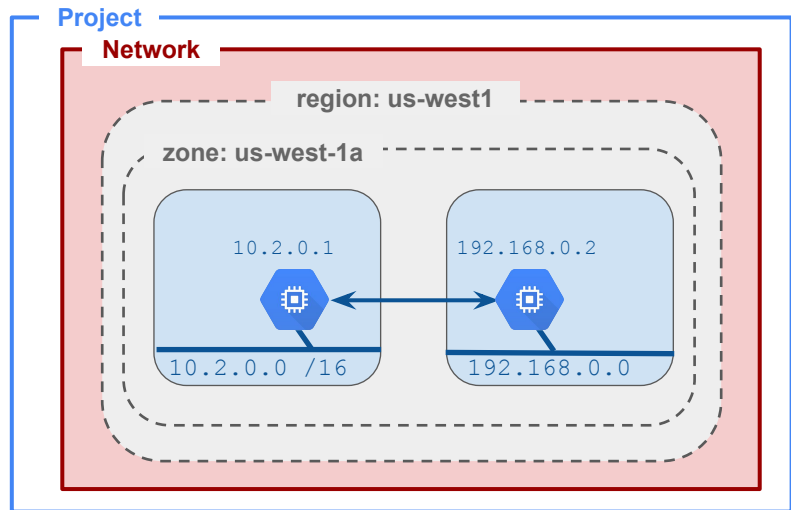
10.2.0.0 crosses zones within a region. Allocating the VMs in different zones provides fault isolation and increases availability. Additionally, a firewall rule can be written against subnetwork 10.2.0.0 and it will apply to both servers even though that are in different zones. Routes and Firewalls are global resources in CVN.



# Security

One Project  
One Network  
One Region  
One Zone  
**Two Subnetworks**

Increased security  
control due to separate  
subnetworks



The point here is that by isolating the VMs on separate subnetworks, it becomes easier to write firewall rules that control access between them.

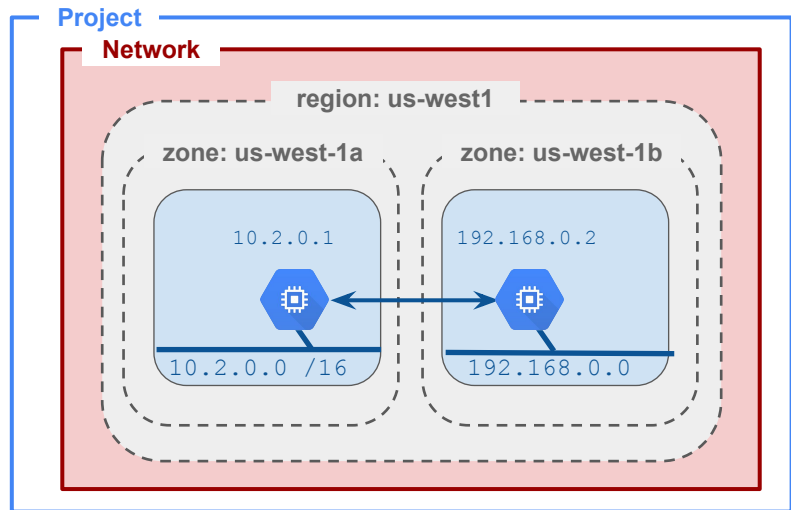
If the VMs were on the same subnet as in the previous slide, we would be writing firewall rules to the individual hosts or using VM tags and the policy wouldn't be as obvious.

Putting the VMs on separate subnets surfaces security policies in the firewall rules.

# Security + availability

One Project  
One Network  
One Region  
**Multiple Zones**  
**Two Subnetworks**

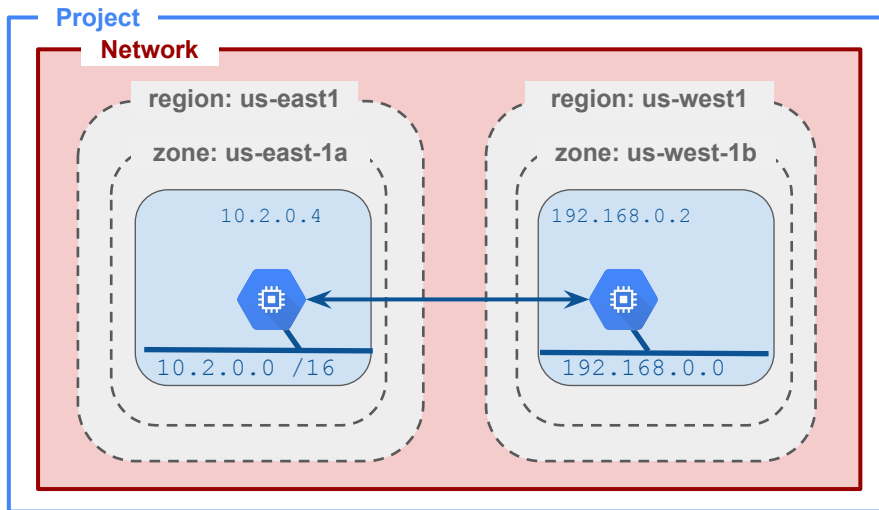
Increased security control  
(from separate  
subnetworks)  
Increased availability  
(from separate zones)



The point here is that you can get the benefits of separate zones without having to reconfigure the addressing or the firewall rules. You just allocate VMs in different zones.

The benefits of how CVN has broken with conventional networking starts to become evident. If availability zones were associated with a physical IP address, as they would be in a Datacenter, the firewall rules to implement policy would become complicated. In the cloud, with virtual networking, the availability zone is no longer coupled to the IP address, simplifying routing policies and security policies.

# Globalization



One Project  
One Network  
**Multiple Regions**  
Multiple Zones  
Two Subnetworks

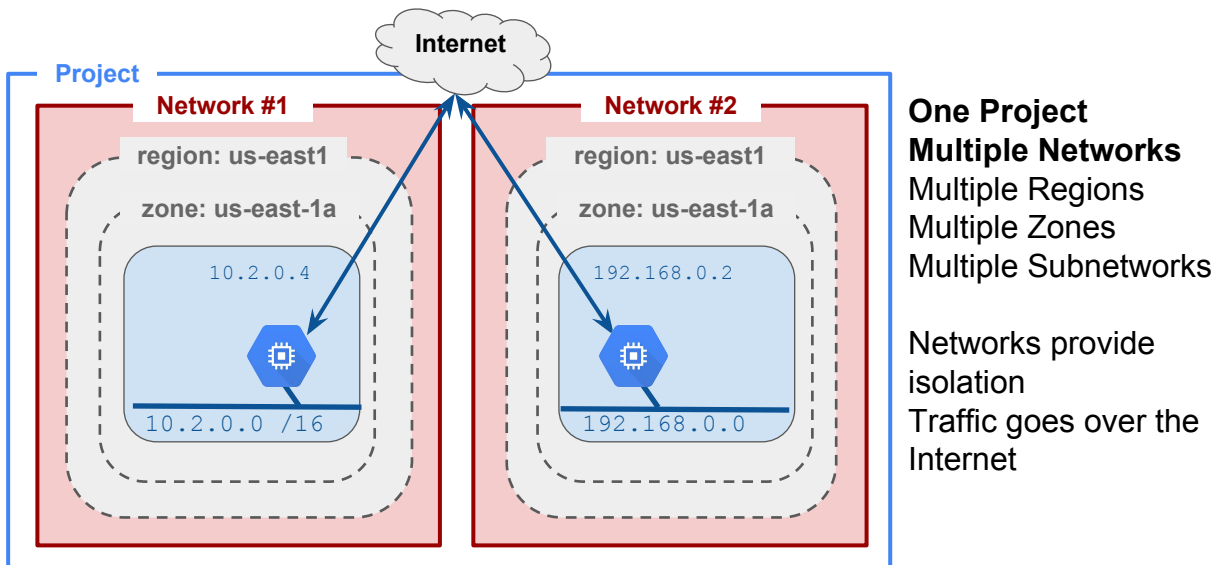
Increased  
availability through  
globalization

Subnetworks  
cannot span  
regions

In this example, the VMs are now in two different regions. Subnetworks are limited to a single region. So we can't get the benefits of a single subnetwork across regions as we could with the single subnetwork encompassing zones.

If we want global availability -- alternatives and failover VMs that are in a different geographic region -- there is a bit more complexity involved. Notice that because these VMs are in a single Network, even though they are in different regions, they can still communicate through GCP's internal global network.

# Isolation



In this case, the VMs are in separate networks. Even though both VMs are located in the same region (us-east1) and in the same zone (us-east-1a), they cannot communicate except over the Internet.

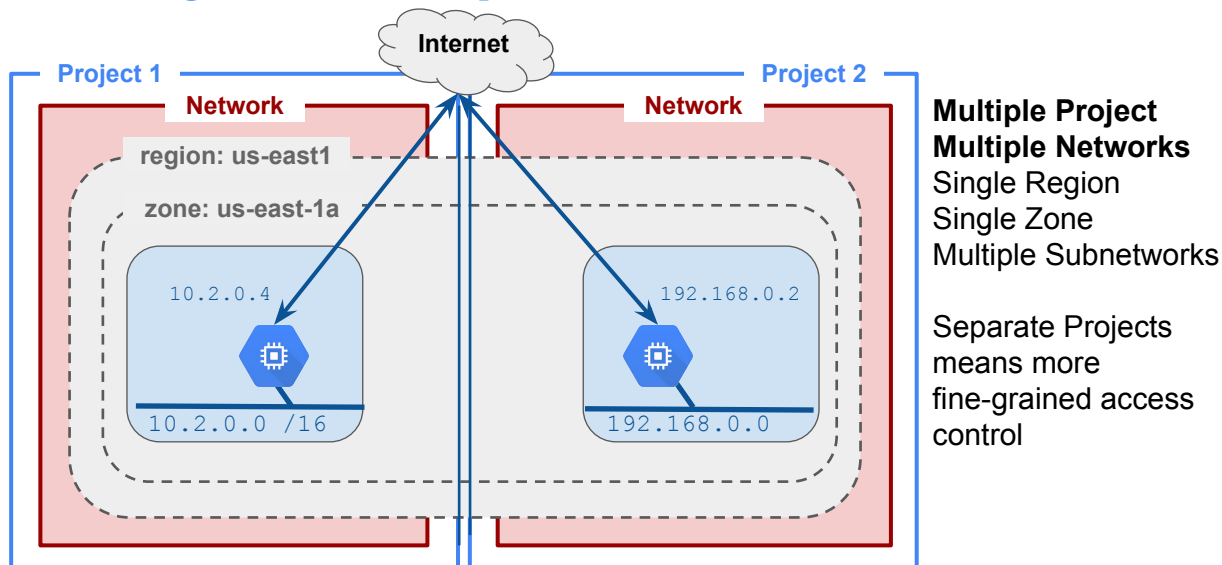
Use separate networks when you need absolute network traffic isolation.

Why would you put resources in separate networks? For *resource management* reasons.

Separate networks gives *separate resource accountability*. By putting organization #1 in network #1, and organization #2 in network #2 you would be assured that they could *never* use one another's resources. Later on, if they had some reason to share resources - for example using a common server - they would not be able to do over internal IP.

Note that all the resources in both networks would still roll up to a single billing account. If you wanted completely separate billing - *separate financial accountability* - then they would have to be in different projects.

# Management separation



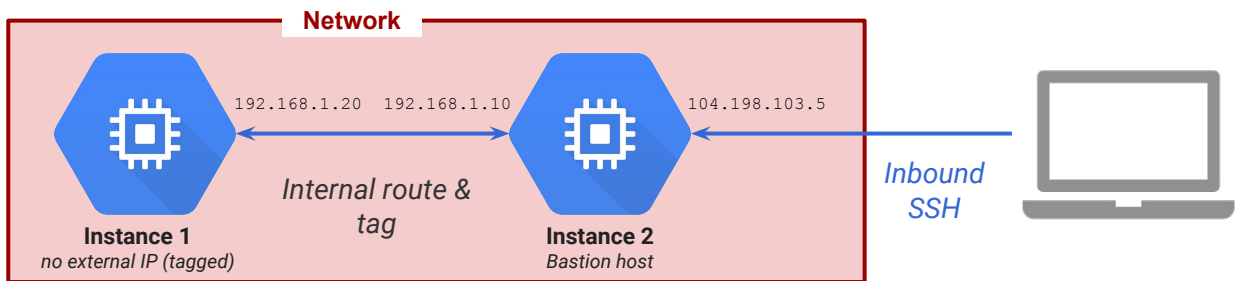
Finally, in this case, we have isolated the VMs in separate projects. This can be useful for Identity and Access Management.

For example, if Software Development is Project 1 and Test Engineering is Project 2, you can assign different people to different roles in the projects.

Consider dividing a system up into multiple projects for better access control. But remember that a Network cannot span Projects. So using separate Projects implies that the VMs must communicate via the Internet.

# Bastion Host Isolation

- Instance used as “jump host”
  - External connections via SSH, used to connect to internal instances
- Be sure to harden bastion host
  - Limit CIDR range of source IPs connecting to bastion
  - Firewall rules allow SSH traffic to private instances only from bastion



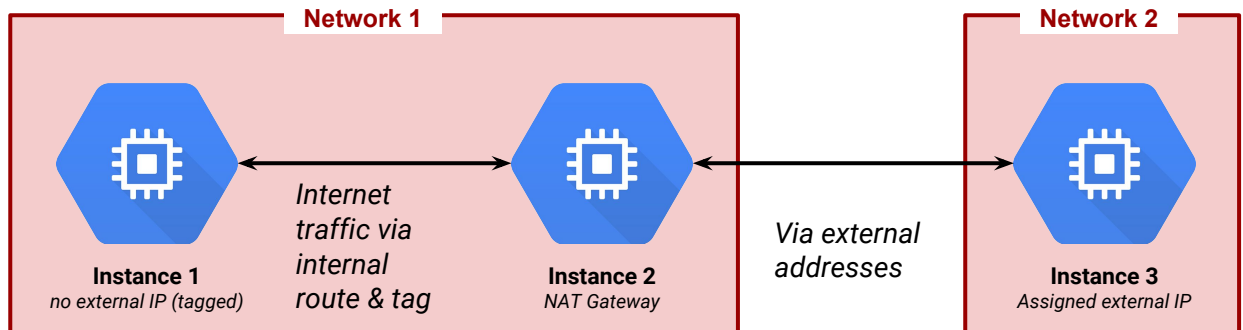
For more information on configuring a bastion host, see:  
<https://cloud.google.com/solutions/connecting-securely#bastion>.

## Best Practice

Use VPN or some other more secure form of connection for ordinary activities.  
Use SSH with the Bastion Host as the maintenance avenue of last resort.

# NAT Gateway Host Isolation

- Instance 1 has no external IP address and is tagged to match the route
- Instance 2 is configured as a NAT gateway with IP forwarding
- Instance 1 communicates with instance 3 via gateway
- Note: Networks 1 and 2 could be in the same or separate projects



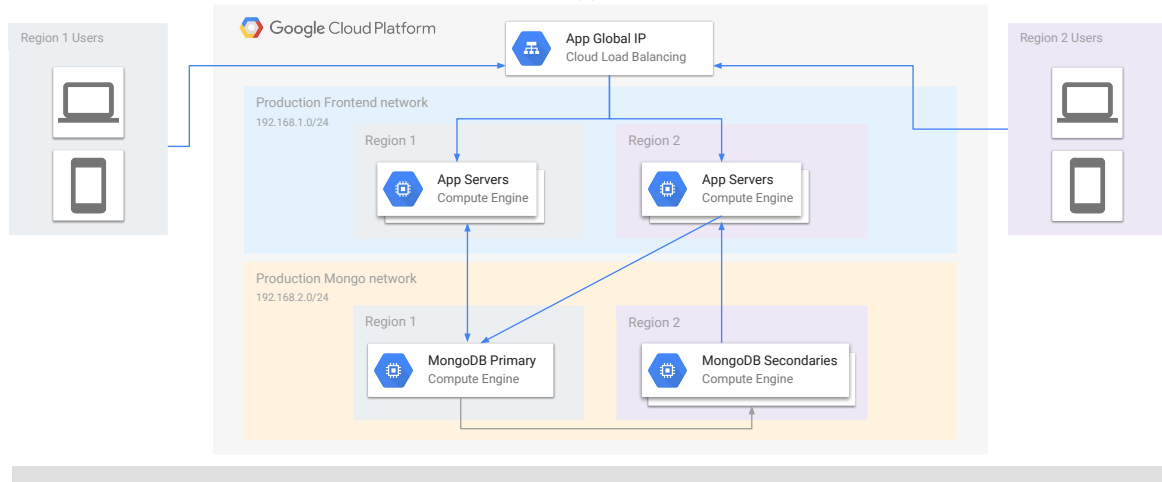
1. Create a Compute Engine instance to act as the gateway
  - Enable IP forwarding with the `--can-ip-forward` flag
2. Create additional Compute Engine instances with no external IP address
  - Disable external IP addresses with the `--no-address` flag
  - Also add a tag to identify instances that will use the gateway routing
3. Create a route to send traffic destined to the Internet through your gateway instance
4. Login to the gateway instance and configure iptables to NAT internal traffic to the Internet

For more information on configuring a NAT gateway, see:

<https://cloud.google.com/compute/docs/networking#natgateway>

# Network Isolation

Architecture: Isolated application and DB networks



The figure above shows a network with custom subnets.

Users access the web application via a global IP address (HTTP(s) load balancing will be addressed in Chapter 13). Requests are forwarded to app servers running in different regions, routing users to the closest available resources. Each of the frontend servers has external IP addresses and can communicate with devices outside Google Cloud Platform. The frontend servers read/write data from/to backend database servers.

The backend servers run in a separate network that is secured to only allow communications to/from the production frontend network. This configuration ensures that no users outside the project can access the database servers directly. This isolation is accomplished via firewall rules that allow traffic into production Mongo network only from hosts in the production frontend network.



# Agenda

- 1 → Cloud Virtual Networking (CVN)
- 2 → Projects, networks and subnetworks
- 3 → IP addresses
- 4 → Routes and rules
- 5 → Billing
- 6 → Lab
- 7 → Common network designs
- 8 → Review

# Review of Routes and Firewall rules

- Firewall rules and routes
  - Can be added to each network
  - Apply to a single network
  - Cannot be moved from one network to another
  - Are subject to quota
  - Can be updated or deleted
  - Can use **tags** to filter the instances they apply to
  - Expect to create several firewall rules for a network
- Default routes do not usually need to be modified

New: Egress rules, Deny rules, and Priorities

## More...

- Using networks and firewalls
  - <https://cloud.google.com/compute/docs/networking>
- Using subnetworks
  - <https://cloud.google.com/compute/docs/subnetworks>

More to learn on this subject. Here are some suggestions and links.

