Google Cloud

Partner Certification Academy

Professional Cloud Network Engineer

The information in this presentation is classified:

Google confidential & proprietary

⚠ This presentation is shared with you under NDA.

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

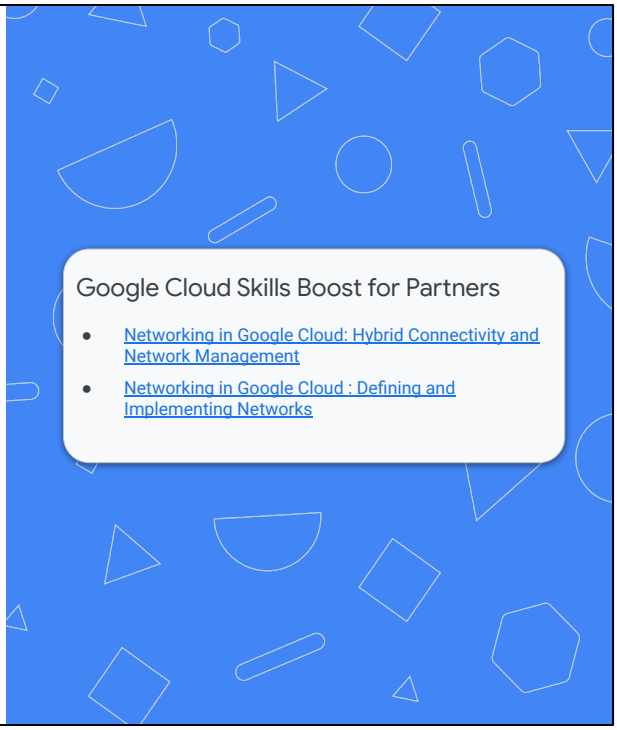
Thank you!

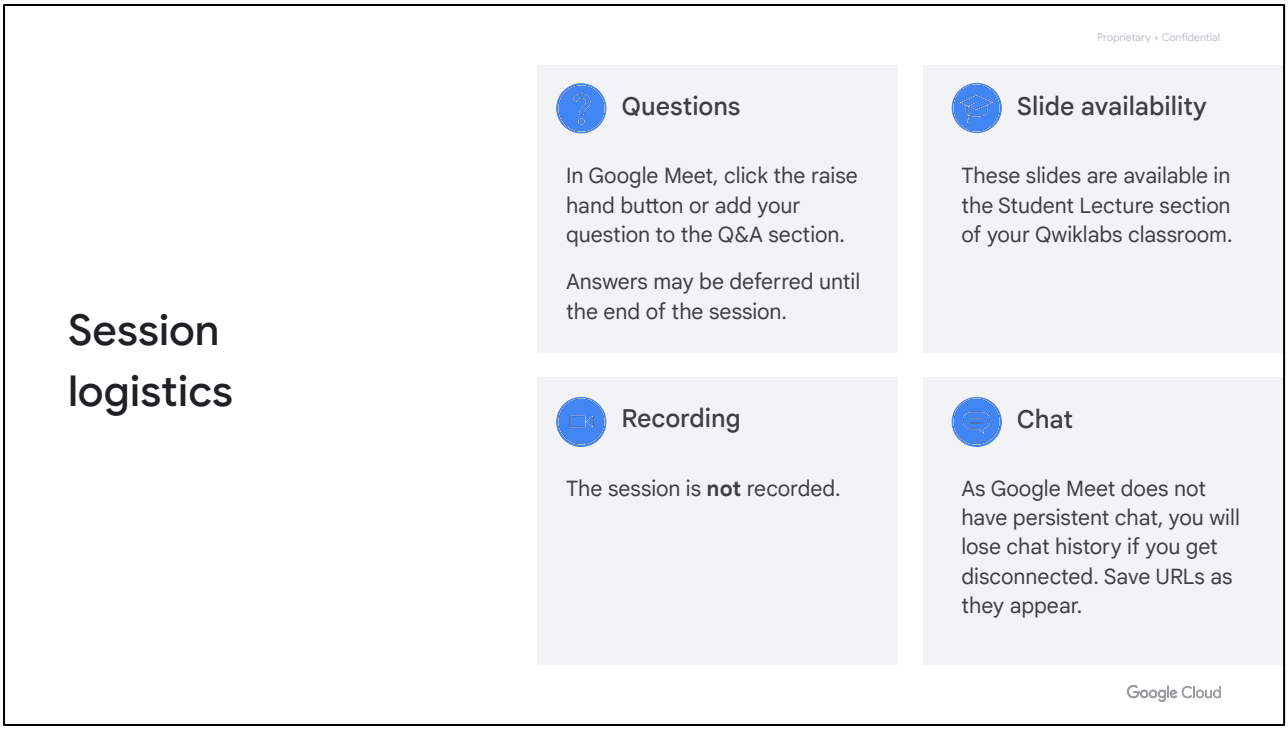
Google Cloud

Source materials

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

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





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

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

The session is not recorded.

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

Welcome to the Load Balancing, CDN and Hybrid Networking module.

Today's agenda

01 Load balancers

02 Internal load balancers

03 Content Delivery Network (CDN) options

04 Hybrid connectivity

AGENDA

Objectives

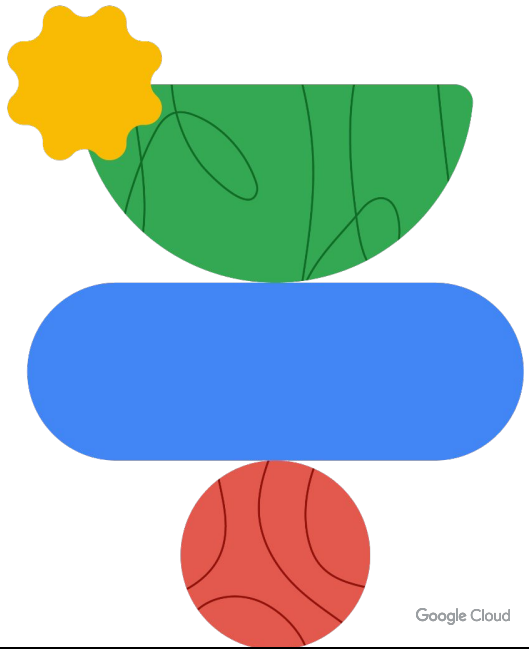
01 Provide an overview of load balancing in Google Cloud

02 Explain the different Content Delivery Network (CDN) options in Google Cloud

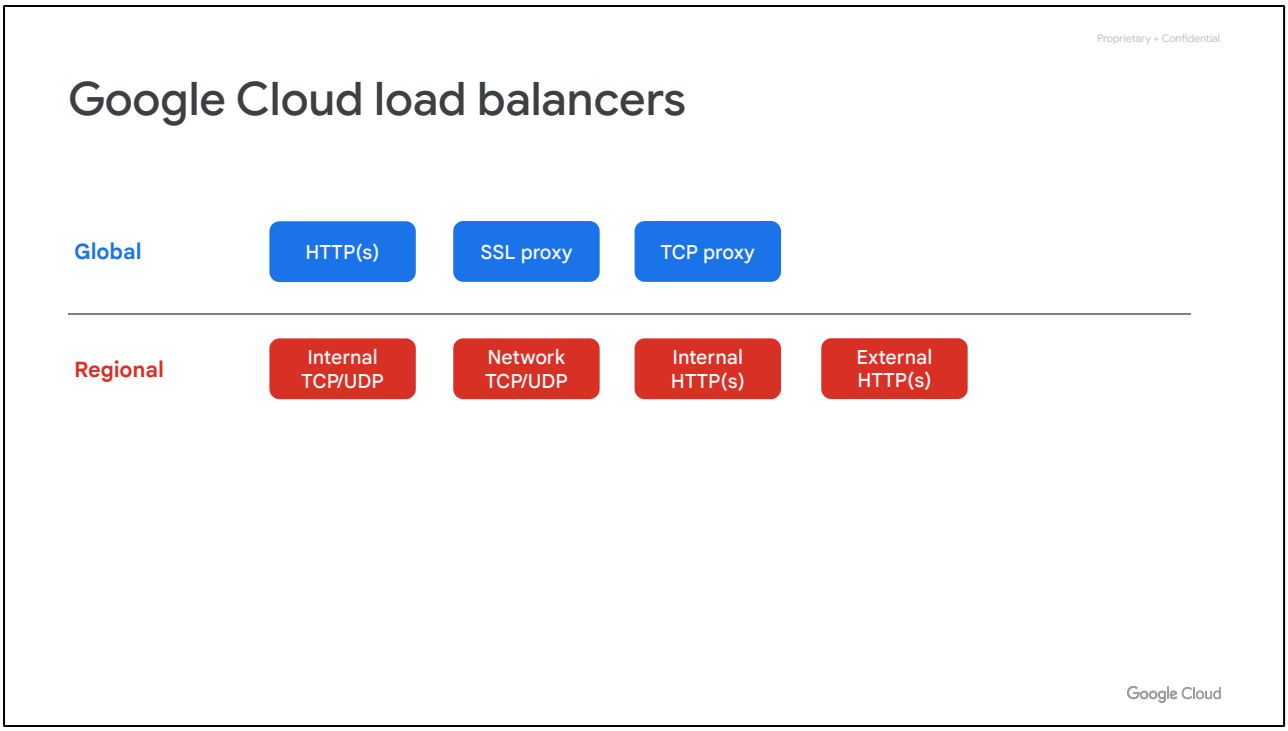
03 Describe the different hybrid connectivity options, such as Cloud Interconnect, Cloud VPN and Network Connectivity Center

Objectives

Load balancers



BREAK SLIDE



Google Cloud offers different types of load balancers that can be divided into two categories: global and regional.

The global load balancers are the HTTP(S), SSL proxy, and TCP proxy load balancers. These load balancers leverage the Google Front End (GFE) service. GFE is a software-defined, distributed system that is available from Google points of presence and is distributed globally. The global load balancer manages redundancy, such as routing a request to a nearby region when the desired region is unavailable.

Use a global load balancer when your users and instances are globally distributed, you need access to the same workloads and content, and you want to use a single anycast IP address to provide access.

The regional load balancers are the internal TCP/UDP, the network TCP/UDP, the internal HTTP(S), and the external HTTP(s) load balancers. These load balancers distribute traffic to instances that are in a single Google Cloud region.

The internal load TCP/UDP balancer uses Andromeda, which is a a network virtualization stack that is software-defined and made by Google Cloud.

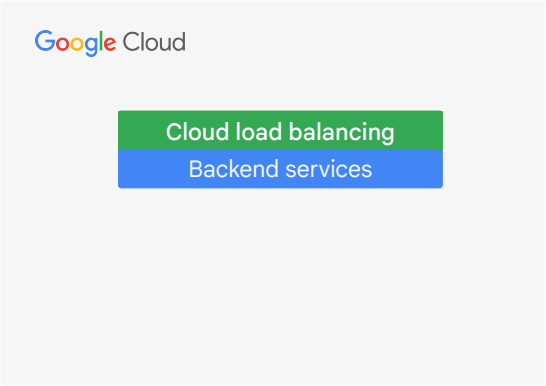
The network load balancer uses Maglev, which is a large, distributed software system.

The internal HTTP(S) load balancer is a proxy-based Layer 7 load balancer. This load balancer lets you run and scale your services behind a private load balancing IP

address. The IP address is accessible only in the region where the load balancer is located.

Overview of Cloud load balancing

- Cloud load balancing receives client traffic.
- Backend services define:
  - How the traffic is distributed.
  - Which health check to use.
  - If session affinity is used.
  - Which other services are used (such as Cloud CDN or Identity-Aware Proxy).



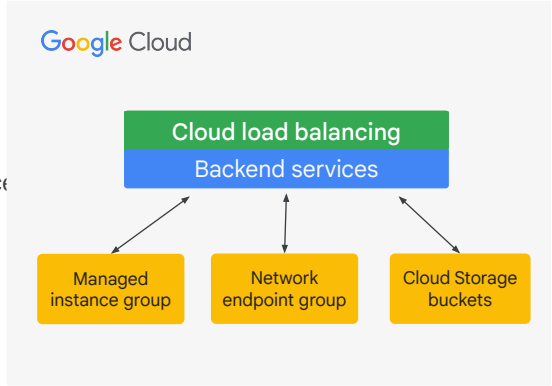
Cloud Load Balancing receives client traffic. This traffic can be external or external, depending on the load balancer you use.

The backend services define how to handle the traffic. For example, backend services define how the traffic is distributed, which health check to use, and if session affinity is used. Backend services also define which other Google Cloud services to use, such as Cloud CDN or Identity-Aware Proxy.

Overview of Cloud load balancing

Cloud load balancing can route traffic to:

- Managed instance groups: a group of virtual machines created from a template.
- Network endpoint groups (NEG): a group of service or workloads.
- Cloud Storage buckets

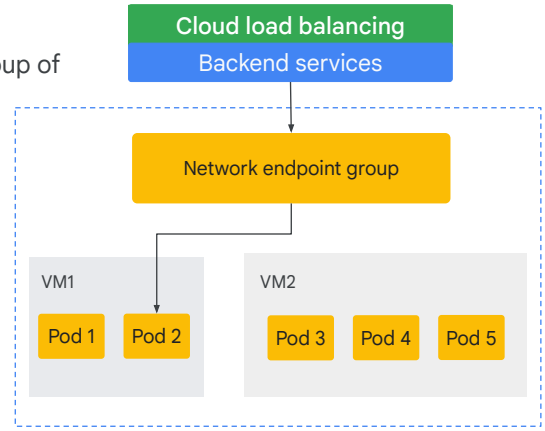


In previous training, you learned that Cloud Load Balancing can route traffic to a managed instance group, a network endpoint group (NEG), or Cloud Storage buckets.

In this module, we are going to look at some special features related to network endpoint groups.

Network endpoint groups (NEG)

- A NEG is a configuration object that specifies a group of backend endpoints or services.
- There are five types of NEGs:
  - Zonal
  - Internet
  - Serverless
  - Private service connect
  - Hybrid connectivity



A network endpoint group (NEG) is a configuration object that specifies a group of backend endpoints or services. A common use case for this configuration is deploying services in containers, as in Google Kubernetes Engine. The load balancer must be able to select a pod from the container, as shown in the example. You can also distribute traffic in a granular fashion to workloads and services that run on your backend hosts.

You can use NEGs as backends for some load balancers and with Traffic Director. For a complete list of supported load balancers, refer to the network endpoint groups overview in the Google Cloud documentation.

Zonal and internet NEGs define how endpoints should be reached, whether they are reachable, and where they are located.

A serverless NEG is a backend that points to a Cloud Run, App Engine, Cloud Functions, or API Gateway service.

A zonal NEG contains one or more endpoints that can be Compute Engine virtual machines (VMs) or services that run on the VMs. Each endpoint is specified either by an IP address or an IP:port combination.

An internet NEG contains a single endpoint that is hosted outside of Google Cloud. This endpoint is specified by hostname FQDN:port or IP:port.

A serverless NEG points to Cloud Run, App Engine, Cloud Functions services that reside in the same region as the NEG.

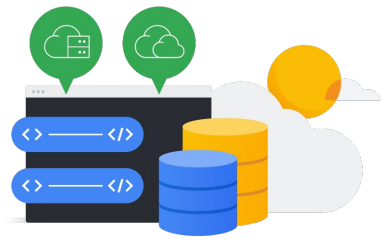
A Private Service Connect NEG contains a single endpoint. That endpoint that resolves to either a Google-managed regional API endpoint or a managed service published by using Private Service Connect.

A hybrid connectivity NEG points to Traffic Director services that run outside of Google Cloud. The focus in this module is on hybrid connectivity NEGs.

For more information on using NEGs, please refer to the Network endpoint groups overview in the Google Cloud documentation.

Hybrid connectivity and load balancing

- A hybrid strategy lets you extend Cloud load balancing to workloads that run on your existing infrastructure outside of Google Cloud.
- This strategy could be:
  - Permanent to provide multiple platforms for your workloads.
  - Temporary as you prepare to migrate your external workloads to Google Cloud.

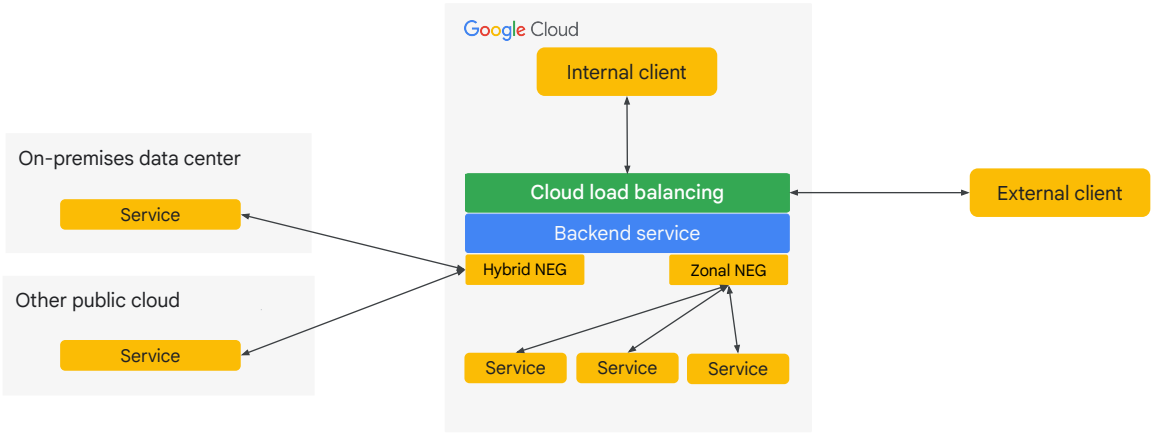


A hybrid load balancing lets you extend Cloud Load Balancing to workloads that run on your existing infrastructure outside of Google Cloud.

A hybrid strategy is a pragmatic solution for you to adapt to changing market demands and incrementally modernize the backend services that run your workloads. You can create a hybrid deployment to enable migration to a modern cloud-based solution or a permanent fixture of your organization IT infrastructure.

Next, let's look at a few general examples of hybrid load balancing.

Hybrid load balancing



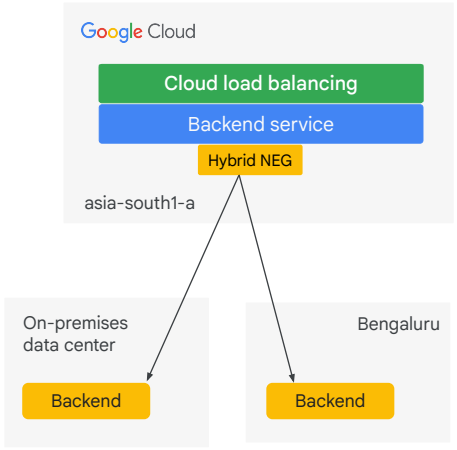
In this example, traffic from clients on the public internet enters your private on-premises environment, and traffic from another public cloud provider enters through a Google Cloud load balancer. The load balancer also gets requests from internal clients.

The load balancer sends requests to the services that run your workloads. These services are the load balancer endpoints, and they can be located inside or outside of Google Cloud. You configure a load balancer backend service to communicate to the external endpoints by using a hybrid NEG. The external environments can use Cloud Interconnect or Cloud VPN to communicate with Google Cloud. The load balancer must be able to reach each service with a valid IP address:Port combination.

The example shows a load balancer backend service with a hybrid and a zonal NEG. The hybrid NEG connects to endpoints that are on-premises and in other public clouds. The zonal NEG points to Cloud Endpoints in the same subnet and zone.

Configuring backend services outside of Google Cloud

- Configure one or more hybrid connectivity network endpoint groups (NEG):
  - Add the IP address: Port for each backend service to a hybrid connectivity NEG.
  - Specify a Google Cloud zone that is as close as possible to your other environment.
  - Add a health check to the NEG.
- Add the hybrid connectivity NEGs to a hybrid load balancer backend service.



To configure backend services outside of Google Cloud, first configure one or more hybrid connectivity network endpoint groups (NEG).

Add each non-Google Cloud network endpoint IP:Port combination to a hybrid connectivity network endpoint group (NEG). Ensure that the IP address and port are reachable from Google Cloud. For hybrid connectivity NEGs, you set the network endpoint type to NON_GCP_PRIVATE_IP_PORT.

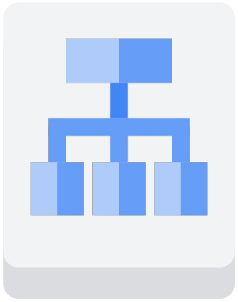
Create the NEG in a Google Cloud zone that is as close as possible to your other environment. For example, if you're hosting a service in an on-premises environment in Bengaluru, India, you can place the NEG in the asia-south1-a Google Cloud zone, as shown in the example.

Add the hybrid connectivity NEGs to a hybrid load balancer backend. A hybrid connectivity NEG must only include endpoints outside Google Cloud. Traffic might be dropped if a hybrid NEG includes endpoints for resources within a Google Cloud VPC network.

Enabled for hybrid load balancing

These Cloud Load Balancing support hybrid load balancing:

- Global external HTTP(S) load balancer
- Global external HTTP(S) load balancer (classic)
- Regional external HTTP(S) load balancer
- Internal HTTP(S) Load Balancing
- External TCP Proxy Load Balancing
- External SSL Proxy Load Balancing



You can use hybrid load balancing with the following:

- Global external HTTP(S) load balancers
- Global external HTTP(S) load balancer (classic)
- Regional external HTTP(S) load balancers
- Internal HTTP(S) load balancers
- External TCP proxy load balancers
- External SSL proxy load balancers

You choose a load balancer depending on your needs, such as where the clients and workloads are located.

Caveats: Hybrid load balancing

01 To create, delete, or manage hybrid connectivity NEGs, use the Google Cloud CLI or the REST API.

02 Regional dynamic routing and static routes are not supported.

03 Internal HTTP(S) Load Balancing and hybrid connectivity must be configured in the same region.

04 Ensure that you also review the security settings on your hybrid connectivity configuration.

To create, delete, or manage hybrid connectivity NEGs, you must use the Google Cloud CLI or the REST API. You can't use the Google Cloud console to create, delete, or manage hybrid connectivity NEGs.

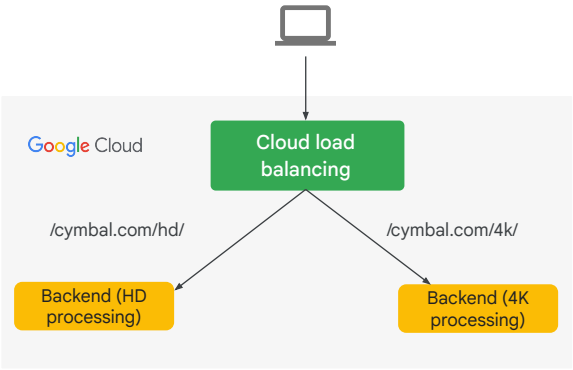
Regional dynamic routing and static routes are not supported. The Cloud Router used for hybrid connectivity must be enabled with global dynamic routing.

Internal HTTP(S) Load Balancing and hybrid connectivity must be configured in the same region. If they are configured in different regions, you might see backends as healthy, but client requests will not be forwarded to the backend.

Ensure that you also review the security settings on your hybrid connectivity configuration. Currently, HA Cloud VPN connections are encrypted by default, using IPsec encryption. Cloud Interconnect connections are not encrypted by default. For more details, go to Encryption in Transit in Google Cloud on the Google Cloud website.

Traffic management

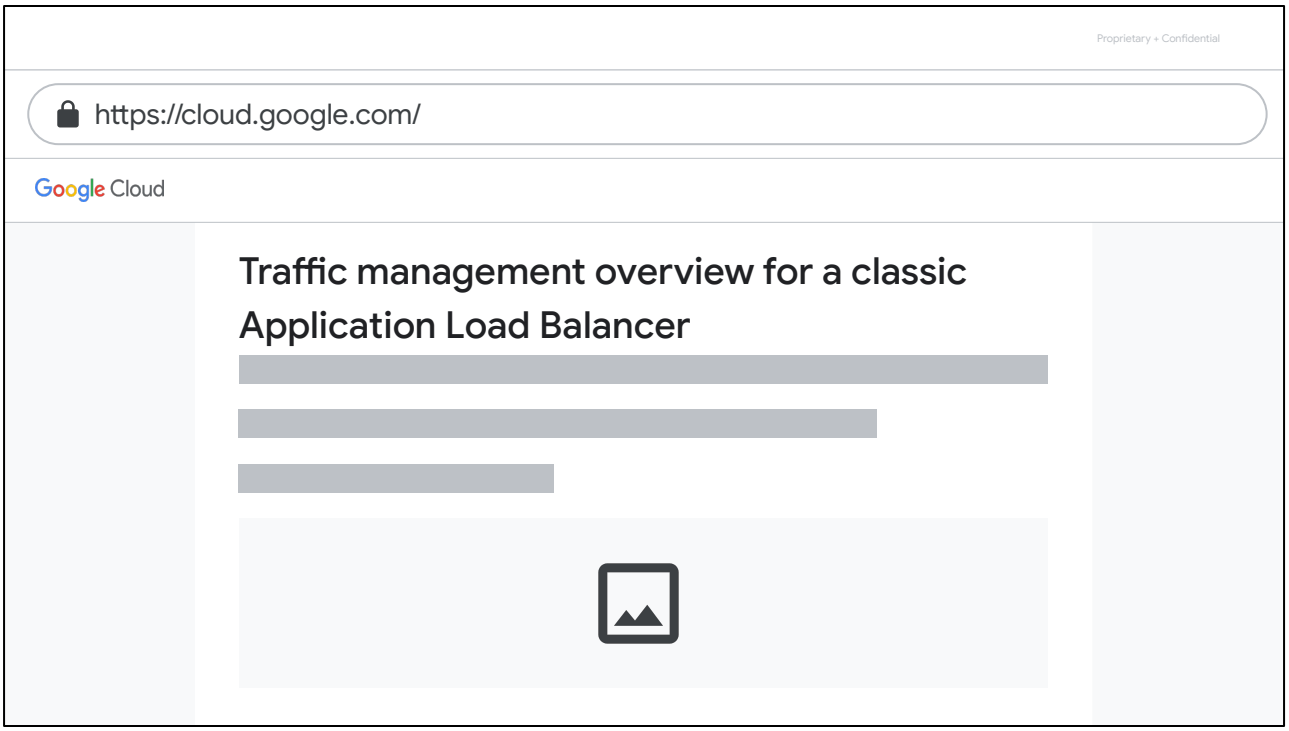
- Traffic management provides enhanced features to route load balancer traffic based on criteria that you specify.
- With traffic management, you can:
  - Direct traffic to a backend based on HTTPS parameters.
  - Perform request-based and response-based actions.
  - Use traffic policies to fine-tune load balancing behavior.



Traffic management provides enhanced features to route load balancer traffic based on criteria that you specify.

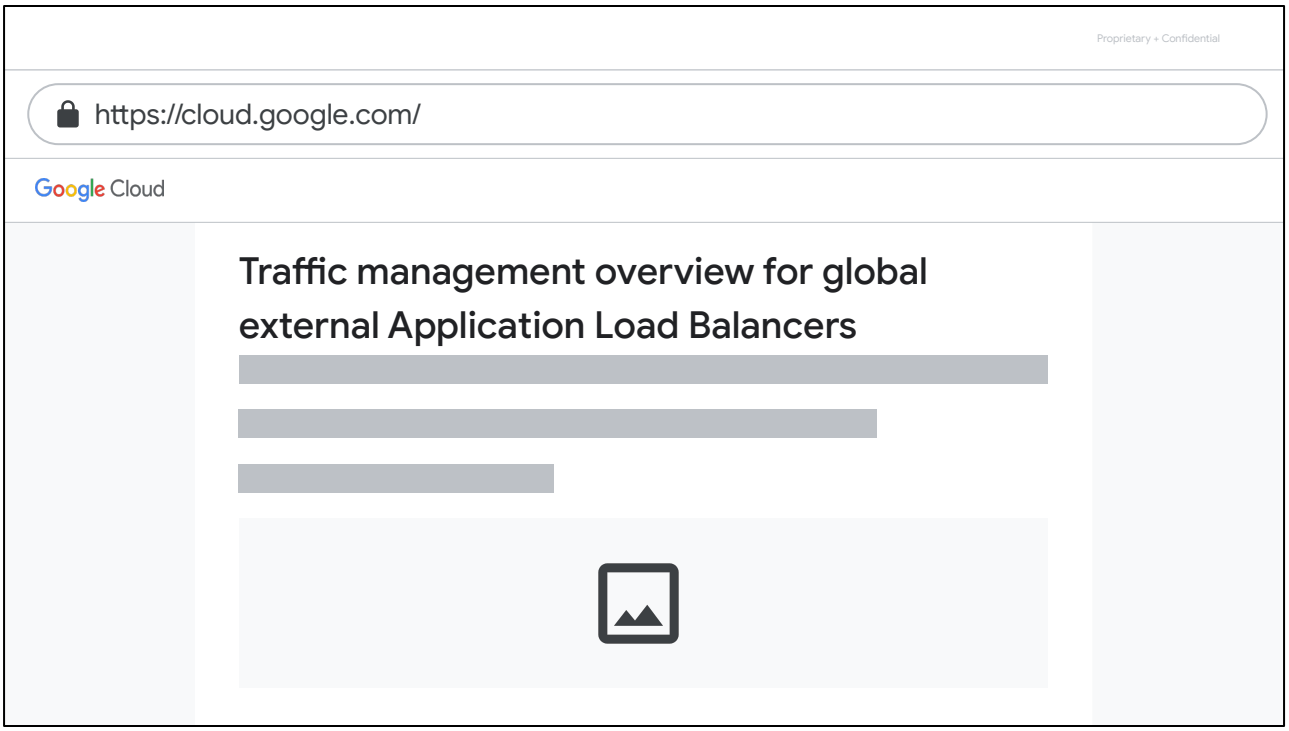
With traffic management, you can:

- Implement traffic steering based on HTTPS parameters, such as the host, path, headers, and other request parameters.
- Perform request-based and response-based actions, such as redirects and header transformations.
- Use traffic policies to fine-tune load balancing behavior, such as retry policies, request mirroring, and cross-origin resource sharing (CORS).

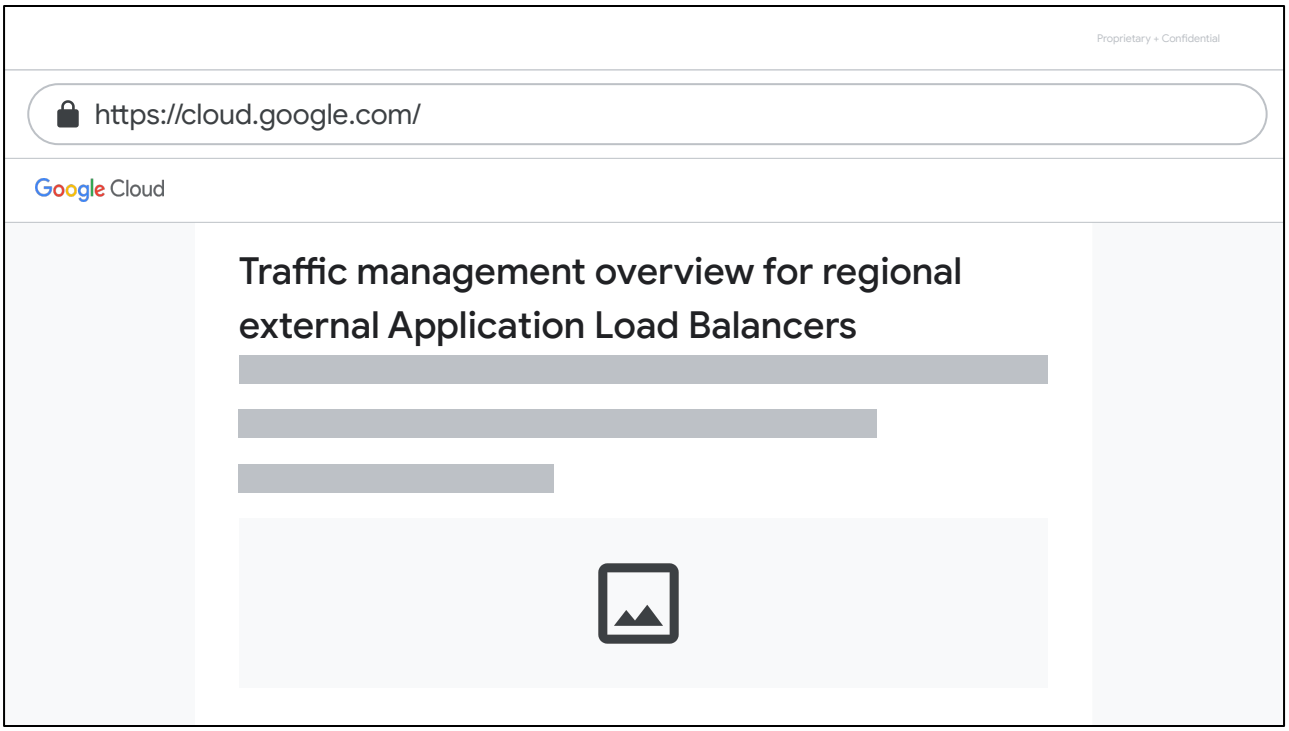


The traffic features that are available can vary per load balancer.

For more information, visit the following documentation: Traffic management overview for a classic Application Load Balancer,



Traffic managements overview for global external Application Load Balancers, and

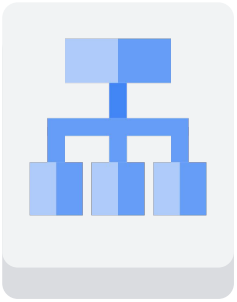


Traffic management overview for regional external Application Load Balancers.

Recall that, in addition to traffic management, Cloud Load Balancing offers backend services like health checks, session affinity, balancing mode, and capacity scaling.

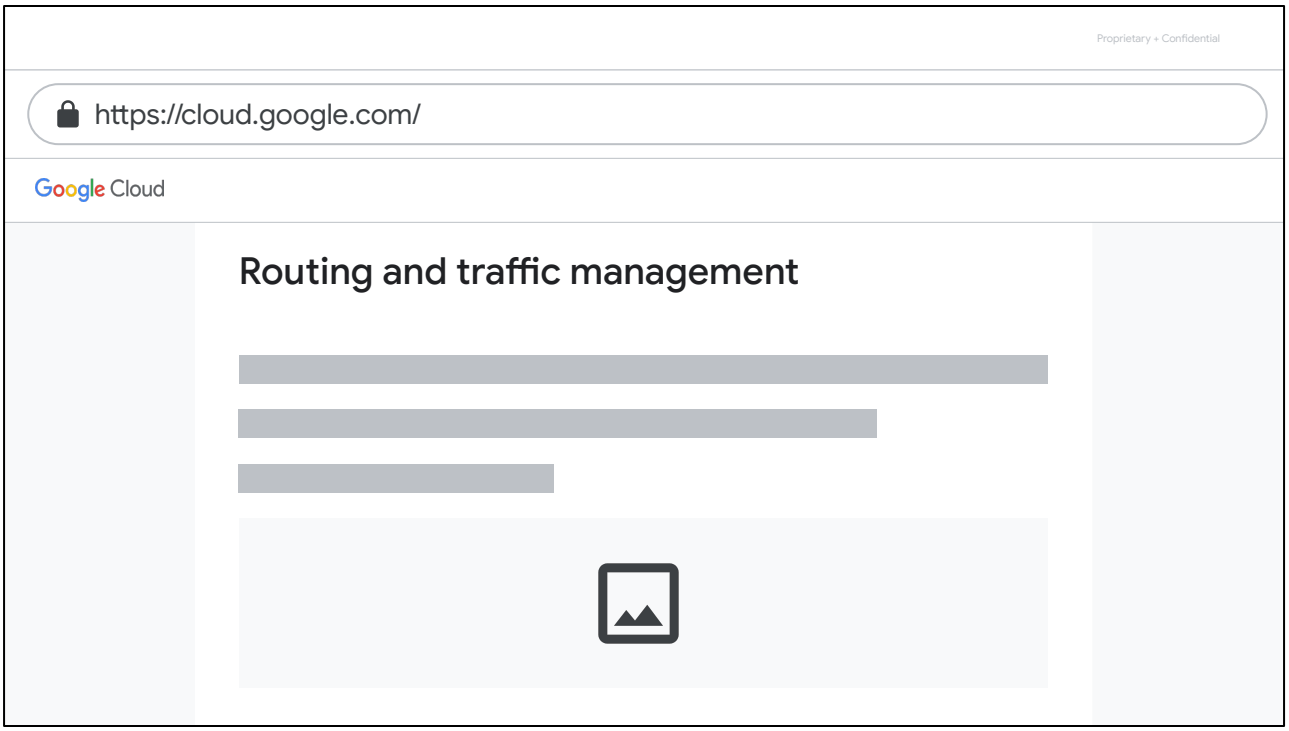
Supported load balancers

- These load balancers support traffic management features:
  - Global external HTTP(S) load balancer
  - Global external HTTP(S) load balancer (classic)
  - Regional external HTTP(S) load balancer
- Other load balancers have access only to traffic features that are available in backend services, such as balancing mode and session affinity.



These load balancers support traffic management: global external HTTP(S) load balancer, global external HTTP(S) load balancer (classic), and the regional external HTTP(S) load balancer. Other load balancers have access to only traffic features available in backend services, such as balancing mode and session affinity.

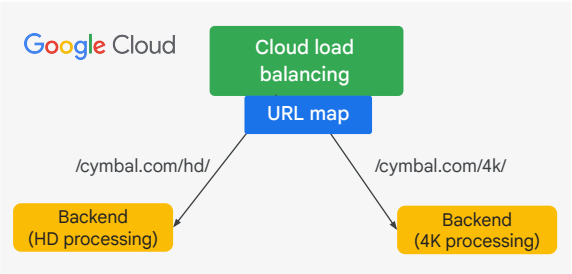
Not all load balancers support all traffic management features.



For a complete list of traffic management features supported for each load balancer, refer to Routing and traffic management in the official Google Cloud documentation.

URL map

- The URL map contains rules that define the criteria to use to route incoming traffic to a backend service.
- Traffic management features are configured in a URL map.
- The load balancer uses the URL map to determine where to route incoming traffic.

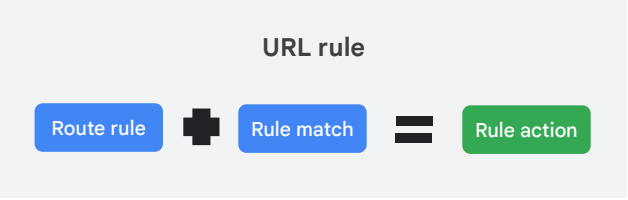


The URL map contains rules that define the criteria to use to route incoming traffic to a backend service. Traffic management features are configured in a URL map. In other words, the load balancer uses the URL map to determine where to route incoming traffic.

URL rule

Each URL rule is composed of:

- A route rule
- A rule match
- A rule action



Each URL rule in a URL map is composed of a route rule, a rule match, and a rule action. Let's look at an example.

A simple URL map

- In this example, video traffic goes to the `video-backend-service`.
- All other traffic goes to the `defaultService`, which is the `web-backend-service`.
- This rule applies for all hosts.

```
defaultService: /pathToTheService/web-backend-service
hostRules:
- hosts:
  - '*'
  pathMatcher: pathmap
name: lb-map
pathMatchers:
- defaultService:/pathToTheService/web-backend-service
  name: pathmap
  pathRules:
  - paths:
    - /video/hd
    - /video/hd/*
    service: /pathToTheService/hd-video-service
- paths:
    - /video/4K
    - /video/4K/*
    service: /pathToTheService/4K-video-service
```


On this slide, you see a URL map that routes video traffic to the video-backend-service. All other traffic is routed to the default service, which is the web-backend-service.

This example is shown by using a YAML file. You can also use the Google Cloud console to configure URL maps.

Let’s look at how this example works.

A simple URL map: `hostRules`

- `hostRules` defines a list of hostnames that are processed by this rule
- `host` defines one or more valid host values.
- `pathMatcher` defines where to find the matching logic to use.
- If there's no applicable host rule, traffic is sent to the `defaultService`.

```
defaultService: /pathToTheService/web-backend-service
hostRules:
- hosts:
  - '*'
  pathMatcher: pathmap
name: lb-map
pathMatchers:
- defaultService:/pathToTheService/web-backend-service
  name: pathmap
  pathRules:
  - paths:
    - /video/hd
    - /video/hd/*
    service: /pathToTheService/hd-video-service
  - paths:
    - /video/4K
    - /video/4K/*
    service: /pathToTheService/4K-video-service
```


The `defaultService` defines a service where traffic should be routed when no matching URL rule is found. You must specify a `defaultService` or a `backendBucket`.

The `hostRules` defines a list of hostnames that are processed by this rule. In this example, there's only one item in the list, which means that only one host rule is defined. This host rule contains an asterisk (*). The asterisk is a wildcard, which matches all hosts.

To see where to find the matching logic to use, look at the value of `pathMatcher`. For this host rule, `pathMatcher` is set to `pathmap`.

A simple URL map: pathMatchers

- pathMatchers defines a list of match rules.
- pathRules define all valid matches, each defined by values within paths.
- If any of the values in path match the traffic, the traffic is directed to service.
- If there are no matches, traffic is routed to the defaultService

```
defaultService: /pathToTheService/web-backend-service
hostRules:
- hosts:
  - '*'
  pathMatcher: pathmap
name: lb-map
pathMatchers:
- defaultService:/pathToTheService/web-backend-service
  name: pathmap
  pathRules:
  - paths:
    - /video/hd
    - /video/hd/*
    service: /pathToTheService/hd-video-service
  - paths:
    - /video/4K
    - /video/4K/*
    service: /pathToTheService/4K-video-service
```


Now, let's explore this example in detail. There's a pathMatchers list, which contains a list of path matching rules. The only element in this list is pathmap, which in this example is also specified in the hostRules.

Each match rule defines logic to process the traffic that is sent to the load balancer.

In this example, there are two sets of paths. One paths list defines valid URL paths for the hd-video-service. The other paths list defines valid URL paths for the 4K-video-service. If the URL contains a match for one of these paths lists, the load balancer routes the traffic to the corresponding service.

If the traffic contains a path that matches none of the paths lists, then it's sent to the default backend service, web-backend-service. In other words, the traffic is sent to the service denoted by pathMatchers/defaultService.

Path rule evaluation

- Path rules are evaluated on a longest-path-matches-first basis.
- Specify path rules in any order.

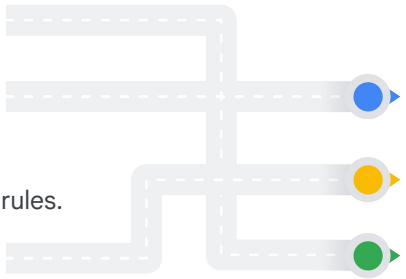
```
pathMatchers:
- defaultService:/pathToTheService/web-backend-service
  name: pathmap
  pathRules:
  - paths:
3   - aShortRule
    service: /pathToTheService/hd-video-service
  - paths:
2   - aLongerRule
    service: /pathToTheService/4K-video-service
  - paths:
1   - anEvenLongerRule
    service: /pathToTheService/4K-video-service
```


Path rules are evaluated on a longest-path-matches-first basis. You can specify the path rules in any order.

In the example, each path rule is labeled to show the order of evaluation. The rule labeled with 1 is evaluated first. The rule labeled with 3 is evaluated last.

Advanced routing mode

- The advanced routing mode:
  - Can choose a rule based on a defined priority.
  - Includes additional configuration options.
  - Uses route rules instead of path rules.
  - Can't include any path rules if a URL map includes route rules.



The advanced routing mode can choose a rule based on a defined priority and includes additional configuration options. Instead of path rules, advanced routing uses route rules.

Each URL map can include either simple or advanced rules, but not both.

An advanced routing mode URL map

This URL map contains rules that route 95% of the traffic to `service-a`, and 5% of the traffic is routed to `service-b`.

```
defaultService: global/backendServices/service-a
hostRules:
- hosts:
  - '*'
  pathMatcher: matcher1
name: lb-map
pathMatchers:
- defaultService: global/backendServices/service-a
  name: matcher1
  routeRules:
  - matchRules:
    - prefixMatch: ''
    routeAction:
      weightedBackendServices:
      - backendService: global/backendServices/service-a
        weight: 95
      - backendService: global/backendServices/service-b
        weight: 5
```


This URL map contains rules that route 95% of the traffic to `service-a`, and 5% of the traffic is routed to `service-b`.

The example shows a YAML implementation of an advanced routing mode. You can also use the Google Cloud console to configure URL maps.

Let’s look at how this example works.

Advanced routing mode: hostRules

- When no matching host rule is found, the `defaultService` defines a default service to use.
- The `hostRules` works the same way as for simple routing mode.

```
defaultService: global/backendServices/service-a
hostRules:
- hosts:
  - '*'
  pathMatcher: matcher1
name: lb-map
pathMatchers:
- defaultService: global/backendServices/service-a
  name: matcher1
  routeRules:
  - matchRules:
    - prefixMatch: ''
    routeAction:
      weightedBackendServices:
      - backendService: global/backendServices/service-a
        weight: 95
      - backendService: global/backendServices/service-b
        weight: 5
```


When no matching host rule is found, the `defaultService` defines a service to use. `defaultService` is a required field.

The `hostsRules` works the same way as for simple routing mode. As in the previous example, this host rule uses the asterisk to match all hosts. Because `pathMatcher` is set to `matcher1`, `/pathMatchers/matcher1` defines the matching logic.

Advanced routing mode: pathMatchers

- routeRules contains a list of one or more matchRules and a routeAction.
- When URLs satisfy the matchRules, their traffic is processed by the routeAction.
- The routeAction defines where traffic is routed.

```
defaultService: global/backendServices/service-a
hostRules:
- hosts
  - '*'
  pathMatcher: matcher1
name: lb-map
pathMatchers:
- defaultService: global/backendServices/service-a
  name: matcher1
  routeRules:
  - matchRules:
    - prefixMatch: ''
    routeAction:
      weightedBackendServices:
      - backendService: global/backendServices/service-a
        weight: 95
      - backendService: global/backendServices/service-b
        weight: 5
```


`/pathMatchers/matcher1` contains a list of `routeRules`. The `routeRules` contain a list of one or more `matchRules` and a `routeAction`. When URLs satisfy the `matchRules`, their traffic is processed by the `routeAction`.

In this example, there's only one item in `matchRules`, where `prefixMatch` equals an empty string. The `prefixMatch` condition matches the URL path prefix; URLs that start with the same string match. In th example, the `prefixMatch` is the empty string, which matches all URLs. In other words, all URLs trigger this match rule, and the `routeAction` is applied.

The `routeAction` defines how the traffic is routed. In the example, the `routeAction` is set to `weightedBackendServices`. `weightedBackedServices` is a list of backend services. A weight value is specified for each backend service; representing a percentage of the total traffic. 95% of the traffic is sent to `service-a`, and 5% of the traffic is sent to `service-b`.

The `routeAction` can also define traffic policies, such as retry policies and CORS.

For a complete list of `routeAction` values, refer to the Google Cloud documentation for the load balancer that you're using.

defaultService

Used if there's no matching host rule.

Used if there's a matching host rule but there's no matching route rule.

1
```
defaultService: global/backendServices/service-a
hostRules:
- hosts
  - '*'
  pathMatcher: matcher1
name: lb-map
```
2
```
pathMatchers:
- defaultService: global/backendServices/service-a
  name: matcher1
  routeRules:
  - matchRules:
    - prefixMatch: ''
    routeAction:
      weightedBackendServices:
      - backendService: global/backendServices/service-a
        weight: 95
      - backendService: global/backendServices/service-b
        weight: 5
```


In this example, you might notice that there are two `defaultService` key-value pairs. One `defaultService` is associated with the `hostRules`, and the other is associated with the `routeRules`.

If there's no matching host rule, the first `defaultService` is used.

If there's no matching route rule, the second `defaultService` is used.

Caveats: Traffic routing

- Not all load balancers support all traffic management features.
- Wildcards are supported, but only after a forward slash (/), for example:
  - Valid: `/video/*`
  - Invalid: `/video*`
- Substring matching and regular expressions are not supported, for example:
  - `/videos/hd*` doesn't match `/videos/hd-pdq`.
  - `/videos/*` does match `/videos/hd-pdq`.

Not all load balancers support all traffic management features. For a complete list of traffic management features supported for each load balancer, refer to Routing and traffic management in the Google Cloud documentation.

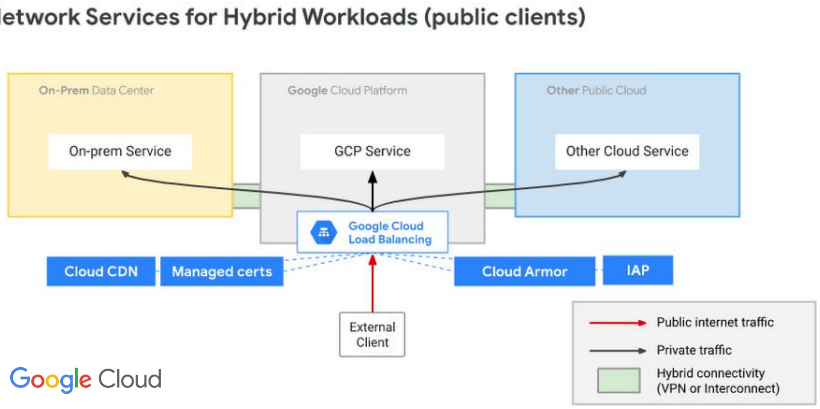
Wildcards are supported, but only after a forward slash (/). For example, /video/* is valid, and /video* is invalid.

Rule matching does not use regular expressions or substring matching. For example, `/videos/hd/*` does not match `/videos/hd-pdq`, because -pdq is a substring and also because it comes after the forward slash. `/videos/*` matches `/videos/hd-pqd`.

Cloud load balancing in multicloud/hybrid environments

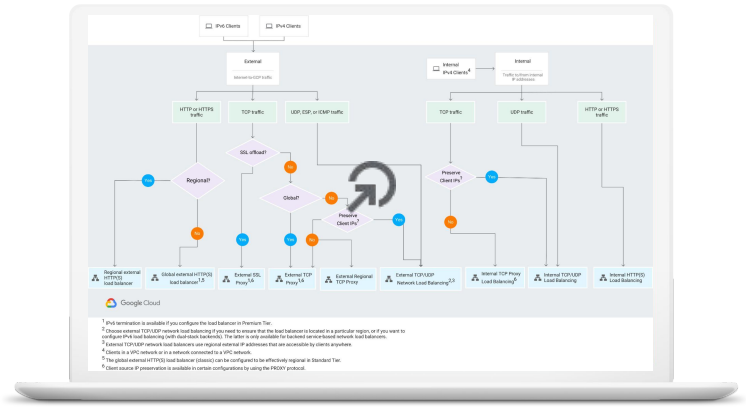
Protocols supported:

- HTTP(s)
- HTTPS/2
- HTTPS/3 with gRPC
- TCP/SSL,
- UDP
- QUIC



https://cloud.google.com/blog/products/networking/how-cloud-load-balancing-supports-hybrid-and-multicloud

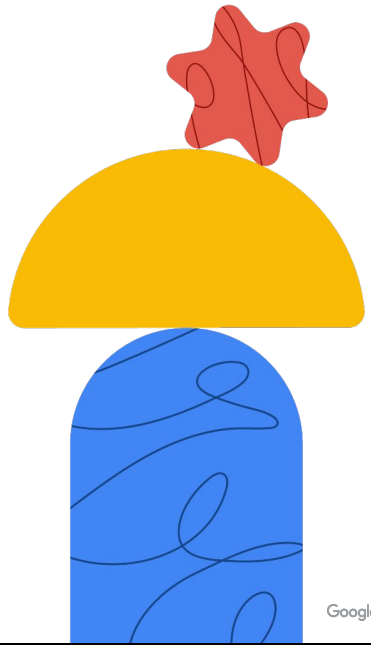
Decision tree



Note to speaker: Click on image on screen

Source: https://cloud.google.com/load-balancing/images/choose-lb.svg

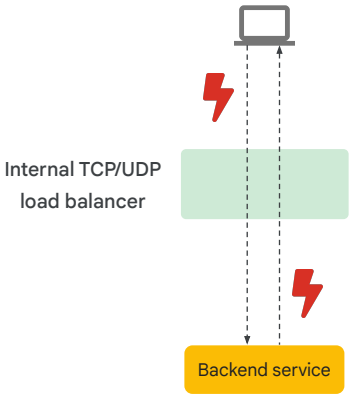
Internal load balancers



BREAK SLIDE

Internal TCP/UDP load balancers are fast

- An internal TCP/UDP load balancer routes connections directly from clients to the healthy backends without any interruption.
- There's no intermediate device or single point of failure.
- Client requests to the load balancer IP address go directly to the healthy backend VMs.
- Responses from the healthy backend VMs go directly to the clients, not back through the load balancer.



Before we cover how to use an Internal TCP/UDP load balancer as a routing next hop, let's discuss why these load balancers are useful: they're fast.

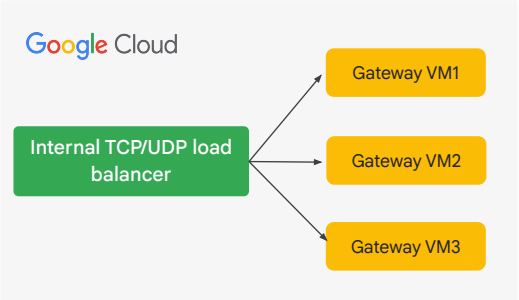
Internal TCP/UDP load balancers don't have the overhead associated with other types of Cloud load balancers; the reduced overhead makes them fast.

An internal TCP/UDP load balancer routes connections directly from clients to the healthy backend without any interruption. There's no intermediate device or single point of failure. Client requests to the load balancer IP address go directly to the healthy backend VMs. Unlike other types of load balancers, there's minimal processing of the incoming traffic.

Responses from the healthy backend VMs go directly to the clients, not back through the load balancer. TCP responses use direct server return. For more information, see IP addresses for request and return packets in the Google Cloud documentation.

Use cases

- Load-balance traffic across multiple VMs that are functioning as gateway or router VMs.
- Use gateway virtual appliances as a next hop for a default route.
- Send traffic through multiple load balancers in two or more directions by using the same set of multi-NIC gateway or router VMs as backends.



Let's consider some use cases for internal TCP/UDP load balancers.

You can load-balance traffic across multiple VMs that are functioning as gateway or router VMs.

You can use gateway virtual appliances as the next hop for a default route. With this configuration, VM instances in your virtual private cloud (VPC) network send traffic to the internet through a set of load balanced virtual gateway VMs.

You can send traffic through multiple load balancers in two or more directions by using the same set of multi-NIC gateway or router VMs as backends. To accomplish this result, you create a load balancer and use it as the next hop for a custom static route in each VPC network. Each internal TCP/UDP load balancer operates within a single VPC network; distributing traffic to the network interfaces of backend VMs in that network.

In these use cases, the backend services are the gateway VMs, gateway virtual appliances, multi-NIC gateways, and router VMs. Because these resources are all internal, it makes sense to access them through an internal TCP/UDP load balancer. As we discussed a moment ago, these load balancers have lower overhead than other load balancers that Google Cloud offers.

Next, let's consider how to make access to these backends even faster.

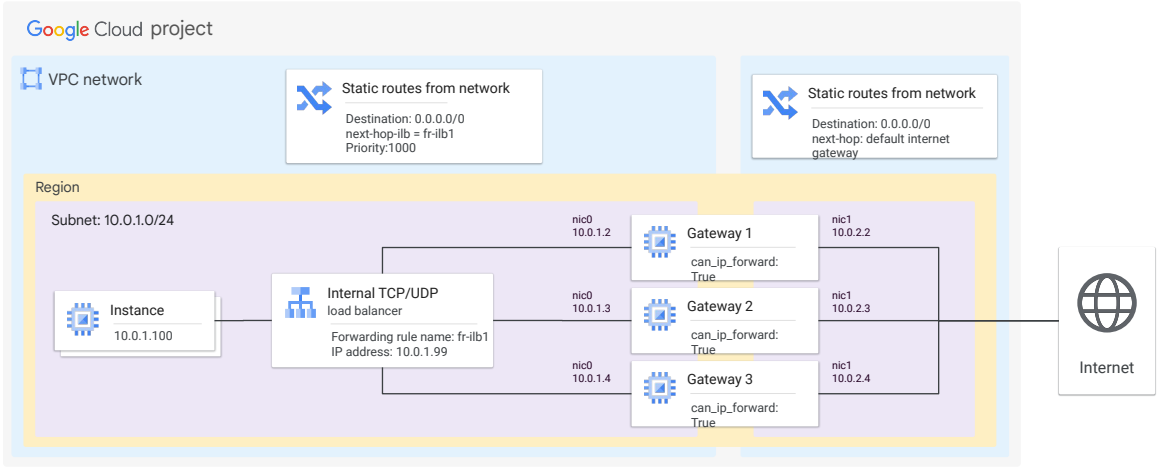
Specifying the next hop

Specification option	Next hop network
Forwarding rule name and the load balancer region	Next hop load balancer and route must be in the same VPC network.
Internal IP address of the forwarding rule	Next hop load balancer can be in the same VPC network as the route or in a peered VPC network.

To specify the next hop, you have two choices, as shown in the table. The main difference concerns the location of the next hop load balancer.

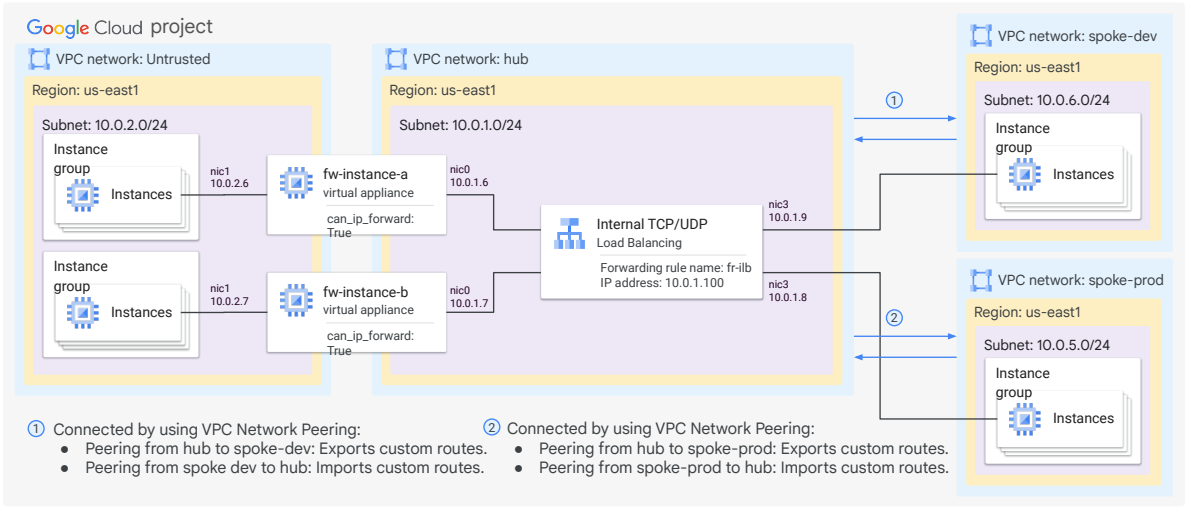
If the next hop load balancer is in the same VPC network, you can specify the forwarding rule name and the load balancer region. To use a next hop load balancer in a peered VPC network, specify the internal IP address of the forwarding rule.

Next hop to a NAT gateway



This use case load balances traffic from internal VMs to multiple network address translation (NAT) gateway instances that route traffic to the internet. In this example, an internal TCP/UDP load balancer 1 has next hops configured to three Compute Engine VMs. Each Compute Engine VM has a NAT gateway that runs on it, and has can_ip_forward set to true. These VMs then forward traffic to the internet. Optionally, you can set up the gateways to apply custom logic to fine-tune access to the internet.

Using a hub and spoke topology



In addition to exchanging subnet routes, you can configure VPC Network Peering to export and import custom static and dynamic routes. Custom static routes that have a next hop of the default internet gateway are excluded. Custom static routes that use next-hop internal TCP/UDP load balancers are included.

You can configure a hub-and-spoke topology with your next-hop firewall virtual appliances located in the hub VPC network by doing the following:

1. In the hub VPC network, create an internal TCP/UDP load balancer with firewall virtual appliances as the backends.

2. In the hub VPC network, create a custom static route, and set the next hop to be the internal TCP/UDP load balancer.

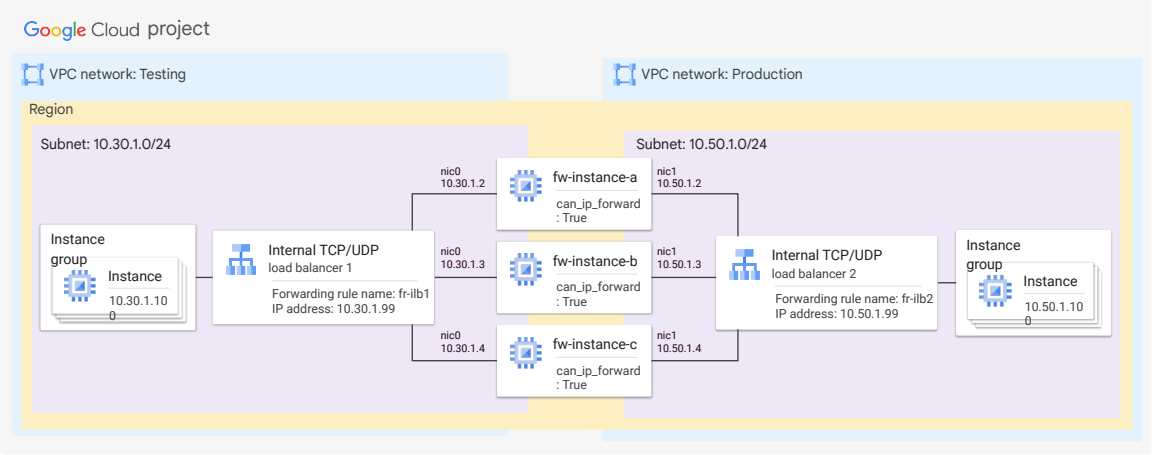
3. Use VPC Network Peering to connect the hub VPC network to each of the spoke VPC networks.

For each peering, configure the hub network to export its custom routes, and configure the corresponding spoke network to import custom routes. The route with the load balancer next hop is one of the routes that the hub network exports.

Subject to the routing order, the next hop firewall appliance load balancer in the hub VPC network is available in the spoke networks. If global access is enabled, the firewall appliance is available according to the routing order. If global access is

disabled, then resources are only available to requestors in the same region.

Load balancing to multiple NICs



Internal TCP/UDP load balancer 1, shown on the left, distributes traffic from the clients to nic0, the primary interface on the backend services. The internal TCP/UDP load balancer 2, shown on the right, distributes traffic from the clients to nic1, the secondary interface on the backend services. The result is that clients can connect to the backend services through nic0 or nic1.

Benefits

When the load balancer is a next hop for a static route:

- No special configuration is needed within the guest operating systems of the client VMs in the VPC network where the route is defined.
- Client VMs send packets to the load balancer backends through VPC network routing, in a bump-in-the-wire fashion.
- It also provides the same benefits as Internal TCP/UDP Load Balancing.

When the load balancer is a next hop for a static route, no special configuration is needed within the client VMs. Client VMs send packets to the load balancer backends through VPC network routing, in a bump-in-the-wire fashion.

Using an internal TCP/UDP load balancer as a next hop for a static route provides the same benefits as Internal TCP/UDP Load Balancing. The health check ensures that new connections are routed to healthy backend VMs. By using a managed instance group as a backend, you can configure auto scaling to grow or shrink the set of VMs based on service demand.

Caveats: Internal TCP/UDP load balancers as next hops

01 Enable global access on the VPC network so that the next hope is usable from all regions.

02 Even if all health checks fail, the load balancer next hop is still in effect.

03 The load balancer must use an IP address that is unique to a load balancer forwarding rule.

You must enable global access on the VPC network so that the next hope is usable from all regions. Whether the next hop is usable depends on the global access setting of the load balancer. With global access enabled, the load balancer next hop is accessible in all regions of the VPC network. With global access disabled, the load balancer next hop is only accessible in the same region as the load balancer. With global access disabled, packets sent from another region to a route that uses an internal TCP/UDP load balancer next hop are dropped.

Even if all health checks fail, the load balancer next hop is still in effect. Packets processed by the route are sent to one of the next hop load balancer backends. If needed, configure a failover policy.

A next hop internal TCP/UDP load balancer must use an IP address that is unique to a load balancer forwarding rule. Only one backend service is unambiguously referenced.

Caveats: Internal TCP/UDP load balancers as next hops

04 Two or more custom static route next hops with the same destination that use different load balancers are never distributed by using ECMP.

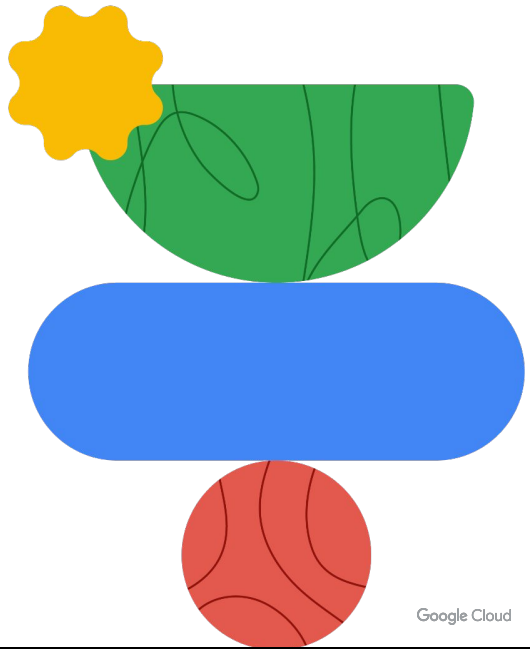
05 To route identical source IP addresses to the same backend, use the client IP, no destination (CLIENT_IP_NO_DESTINATION) session affinity option.

Two or more custom static route next hops with the same destination that use different load balancers are never distributed by using ECMP. If the routes have unique priorities, Google Cloud uses the next hop internal TCP/UDP load balancer from the route with the highest priority. If the routes have equal priorities, Google Cloud still selects just one next hop internal TCP/UDP load balancer.

For packets with identical source IP addresses routed to the same backend, use the client IP, no destination (CLIENT_IP_NO_DESTINATION) session affinity option.

There are some additional caveats for using an internal TCP/UDP load balancer as a next hop, for example, pertaining to the use of network tags. For additional information on this and other caveats, refer to Additional considerations on the Internal TCP/UDP load balancers as next hops page in the Google Cloud documentation.

CDN (Content Delivery Network) options



BREAK SLIDE

Cloud CDN (Content Delivery Network)

Cloud CDN:

- Caches content at the edges of the Google network.
- Provides faster content delivery to users while reducing transmission costs.

Cloud CDN (Content Delivery Network) caches content at the edges of the Google network. This caching provides faster content delivery to users while reducing transmission costs.

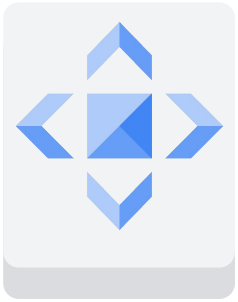
Content can be cached at CDN nodes as shown on this map. There are over 90 of these cache sites spread across metropolitan areas in Asia Pacific, Americas, and EMEA. For an updated list, please refer to Cache locations in the Google Cloud documentation.

For Cloud CDN performance measured by Cedexis, please refer to these reports on the Citrix website.

When setting up the backend service of a HTTP(S) load balancer, you can enable Cloud CDN with a checkbox.

Cloud CDN cache modes

- Cache modes control the factors that determine whether Cloud CDN caches your content.
- Cloud CDN offers three cache modes:
  - USE_ORIGIN_HEADERS
  - CACHE_ALL_STATIC
  - FORCE_CACHE_ALL



Using cache modes, you can control the factors that determine whether Cloud CDN caches your content.

Cloud CDN offers three cache modes. The cache modes define how responses are cached, whether Cloud CDN respects cache directives sent by the origin, and how cache TTLs are applied.

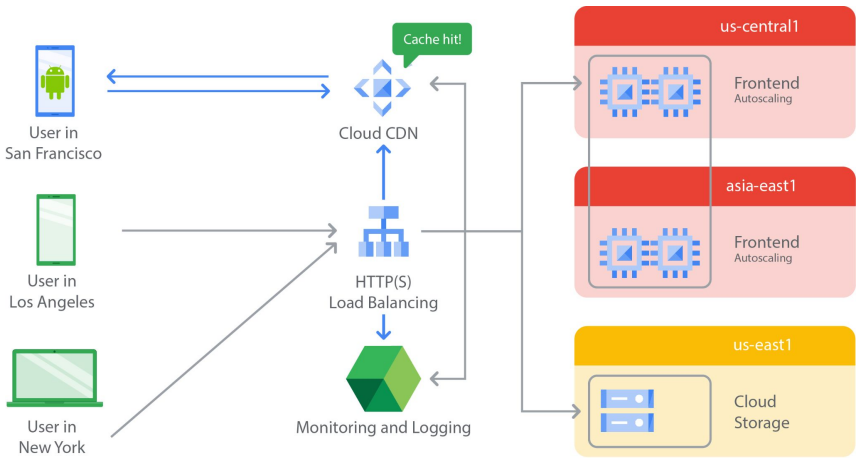
The available cache modes are USE_ORIGIN_HEADERS, CACHE_ALL_STATIC and FORCE_CACHE_ALL.

USE_ORIGIN_HEADERS mode requires origin responses to set valid cache directives and valid caching headers.

CACHE_ALL_STATIC mode automatically caches static content that doesn't have the no-store, private, or no-cache directive. Origin responses that set valid caching directives are also cached.

FORCE_CACHE_ALL mode unconditionally caches responses; overriding any cache directives set by the origin. If you use a shared backend with this mode configured, ensure that you don’t cache private, per-user content (such as dynamic HTML or API responses).

Caching content with Cloud CDN



Let's walk through the Cloud CDN response flow with this diagram.

In this example, the HTTP(S) load balancer has two types of backends. There are managed VM instance groups in the us-central1 and asia-east1 regions, and there's a Cloud Storage bucket in us-east1. A URL map decides which backend to send the content to: the Cloud Storage bucket could be used to serve static content and the instance groups could handle PHP traffic.

When a user in San Francisco is the first to access content, the cache site in San Francisco sees that it can't fulfill the request. This situation is called a cache miss. If content is in a nearby cache, Cloud CDN might attempt to get the content from it, for example if a user in Los Angeles has already accessed the content. Otherwise, the request is forwarded to the HTTP(S) load balancer, which in turn forwards the request to one of your backends.

Depending on the content that is being served, the request will be forwarded to the us-central1 instance group or the us-east1 storage bucket.

If the content from the backend is cacheable, the cache site in San Francisco can store it for future requests. In other words, if another user requests the same content in San Francisco, the cache site might now be able to serve that content. This approach shortens the round trip time and saves the origin server from having to process the request. This is called a cache hit.

For more information on what content can be cached, please refer to Caching overview in the Google Cloud documentation.

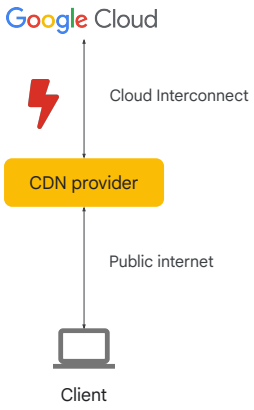
Each Cloud CDN request is automatically logged within Google Cloud. These logs will indicate a “Cache hit” or “Cache miss” status for each HTTP request of the load balancer. You will explore such logs in the next lab.

Cache modes let you control how content is cached.

CDN Interconnect

CDN Interconnect lets you:

- Select third-party Cloud CDN providers to establish Direct Interconnect links at edge locations in the Google network.
- Direct your traffic from your VPC networks to a provider network.
- Optimize your Cloud CDN cache population costs.



CDN Interconnect lets select third-party Content Delivery Network (CDN) providers establish Direct Interconnect links at edge locations in the Google edge network. These connections let you direct your traffic from your VPC networks to a CDN provider network. For a complete list of CDN providers, refer to CDN Interconnect overview in the Google Cloud documentation.

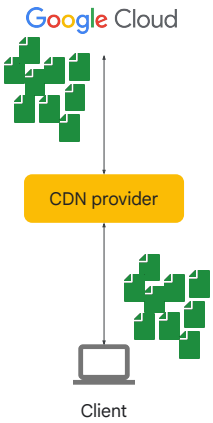
CDN Interconnect lets you connect directly to select CDN providers from Google Cloud. Your network traffic that egresses from Google Cloud through one of these links benefits from the direct connectivity to supported CDN providers.

CDN Interconnect reduces your Cloud CDN cache population costs.

Typical use cases for CDN Interconnect

01 High-volume egress traffic.

01 Frequent content updates.



If you have a high-volume of egress traffic, consider using CDN Interconnect. You can use the CDN Interconnect links between Google Cloud and selected providers to automatically optimize the egress traffic and save money. If you're populating the Cloud CDN cache locations with large data files from Google Cloud, this optimization can be especially helpful.

Frequent content updates are another typical CDN Interconnect use case. Cloud workloads that frequently update data stored in Cloud CDN cache locations benefit from using CDN Interconnect. The direct link to the Cloud CDN provider reduces latency.

CDN Interconnect traffic billing

- Ingress traffic is free for all regions.
- Egress traffic rates apply only to data that leaves Compute Engine or Cloud Storage.
- The reduced price applies only to IPv4 traffic.
- Egress charges for CDN Interconnect appear on the invoice as *Compute Engine Network Egress via Carrier Peering Network*.

Ingress traffic is free for all regions.

Egress traffic rates apply only to data that leaves Compute Engine or Cloud Storage. Egress charges for CDN Interconnect appear on the invoice as *Compute Engine Network Egress via Carrier Peering Network*.

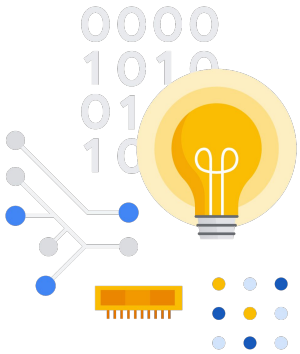
The special pricing for your traffic that egresses from Google Cloud to a CDN provider is automatic. Google works with approved CDN partners in supported locations to accept provider IP addresses. Any data that you send to your allowlisted CDN provider from Google Cloud is charged at the reduced price.

This reduced price applies only to IPv4 traffic. It does not apply to IPv6 traffic.

Intra-region pricing for CDN Interconnect applies only to intra-region egress traffic that is sent to Google-approved CDN providers at specific locations.

Setting up CDN Interconnect

- CDN Interconnect does not require any configuration or integration with Cloud load balancing.
- Work with your supported CDN provider to:
  - Learn which locations are supported.
  - Correctly configure your deployment to use intra-region egress routes.



CDN Interconnect does not require any configuration or integration with Cloud Load Balancing. If your CDN provider is already part of the program, you don't need to do anything. Traffic from supported Google Cloud locations to your CDN provider automatically benefits from the direct connection and reduced pricing.

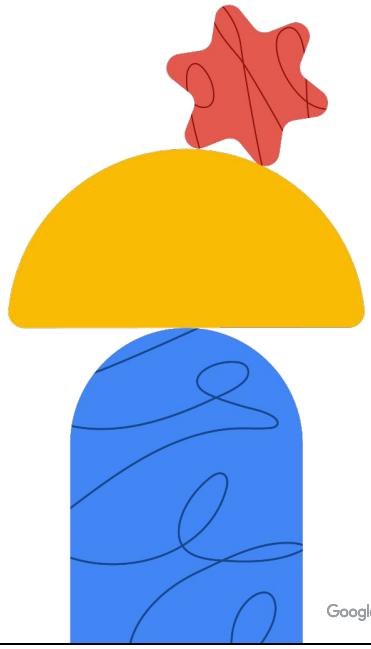
Work with your supported CDN provider to learn what locations are supported. Your supported CDN service provider can also help you correctly configure your deployment to use intra-region egress routes, which cost less than inter-region egress traffic.

Media CDN: for immersive streaming experiences

- CDN designed for streaming video applications
- Integrated AI/ML capabilities
- Secured with Cloud Armor
- Comprehensive API
- Deploy with Terraform

https://cloud.google.com/blog/products/networking/introducing-media-cdn

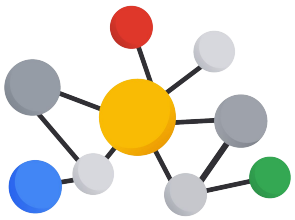
Hybrid connectivity



BREAK SLIDE

Cloud Interconnect and Cloud VPN

- Cloud Interconnect provides a fast connection to the Google network.
- Google offers two different Cloud Interconnect products.
  To physically connect to the Google network:
  - At a Google colocation facility, use Dedicated Interconnect.
  - Through a supported service provider, use Partner Interconnect.
- Google also offers Cloud VPN, for lower bandwidth needs.



Cloud Interconnect provides a fast connection to the Google network. Google offers two different Cloud Interconnect products; choose a product based on your situation and your needs.

When you can physically connect to the Google network at a Google colocation facility, use Dedicated Interconnect. When you cannot connect to the Google network at a Google colocation facility but can connect through a service provider, use Partner Interconnect.

Google Cloud also offers Cloud VPN. Cloud VPN can be useful when you don't need a high-speed internet connection or when you must encrypt data in transit.

In this module, you will learn more about Cloud VPN and both types of Cloud Interconnect.

Comparison of connection options

Connection	Provides	Capacity	Requirements	Access Type
VPN tunnel	Encrypted tunnel to VPC networks through the public internet	1.5–3 Gbps per tunnel	Remote VPN gateway	Internal IP addresses
Dedicated Interconnect	Dedicated, direct connection to VPC networks	10 Gbps or 100 Gbps per link	Connection in colocation facility	
Partner Interconnect	Dedicated bandwidth, connection to VPC network through a service provider	50 Mbps – 50 Gbps per connection	Service provider	

Let's compare these connection options. All of these options provide internal IP address access between resources in your on-premises network and in your VPC network. The main differences are the connection capacity and the requirements for using a service.

The IPsec VPN tunnels that Cloud VPN offers have a capacity of 1.5 Gbps to 3 Gbps per tunnel. The tunnels connect to a VPN device in your on-premises network. The 1.5 Gbps capacity applies to traffic that traverses the public internet, and the 3 Gbps capacity applies to traffic that is traversing a direct peering link. If you want to scale this capacity, you can configure multiple tunnels.

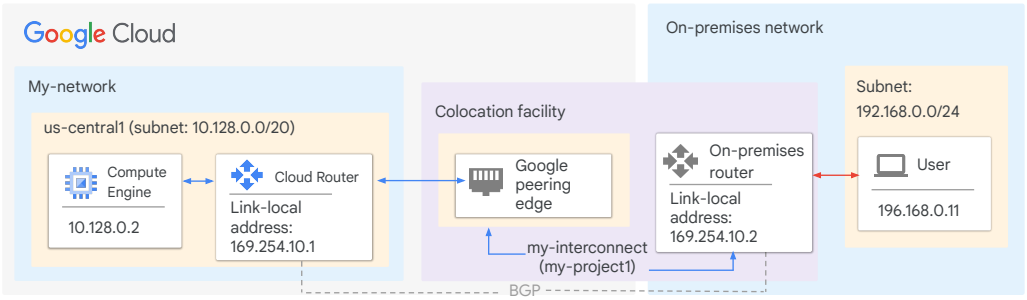
Dedicated Interconnect has a capacity of 10 Gbps or 100 Gbps per link and requires you to have a connection in a Google-supported colocation facility. You can have up to eight links to achieve multiples of 10 Gbps, or up to two links to achieve multiples of 200 Gbps, but 10 Gbps is the minimum capacity.

Partner Interconnect has a capacity of 50 Mbps to 50 Gbps per connection, and requirements depend on the service provider.

Dedicated Interconnect and Cloud Interconnect do not encrypt data in transit. Secure data in transit at the application layer using TLS, for example.

Dedicated Interconnect provides direct physical connections

Dedicated Interconnect provides direct physical connections between your on-premises network and the Google network.

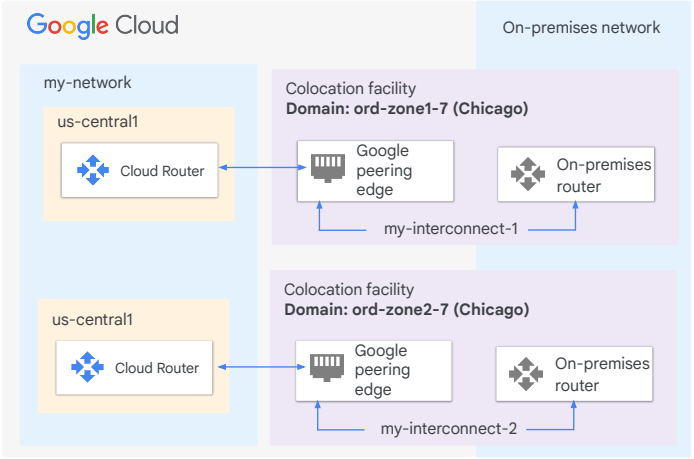


Dedicated Interconnect provides direct physical connections between your on-premises network and the Google network. Dedicated Interconnect enables you to transfer large amounts of data between networks, which can be more cost-effective than purchasing additional bandwidth over the public internet.

Peering edge placement

In order to provide redundancy, consider peering edge placement:

- Every metropolitan area with colocation facilities has at least two edge availability domains.
- Placing a Dedicated Interconnect connection in more than one edge availability domain in a metropolitan area provides redundancy.
- Peering edge placement is also a factor in planning Partner Interconnect connections.



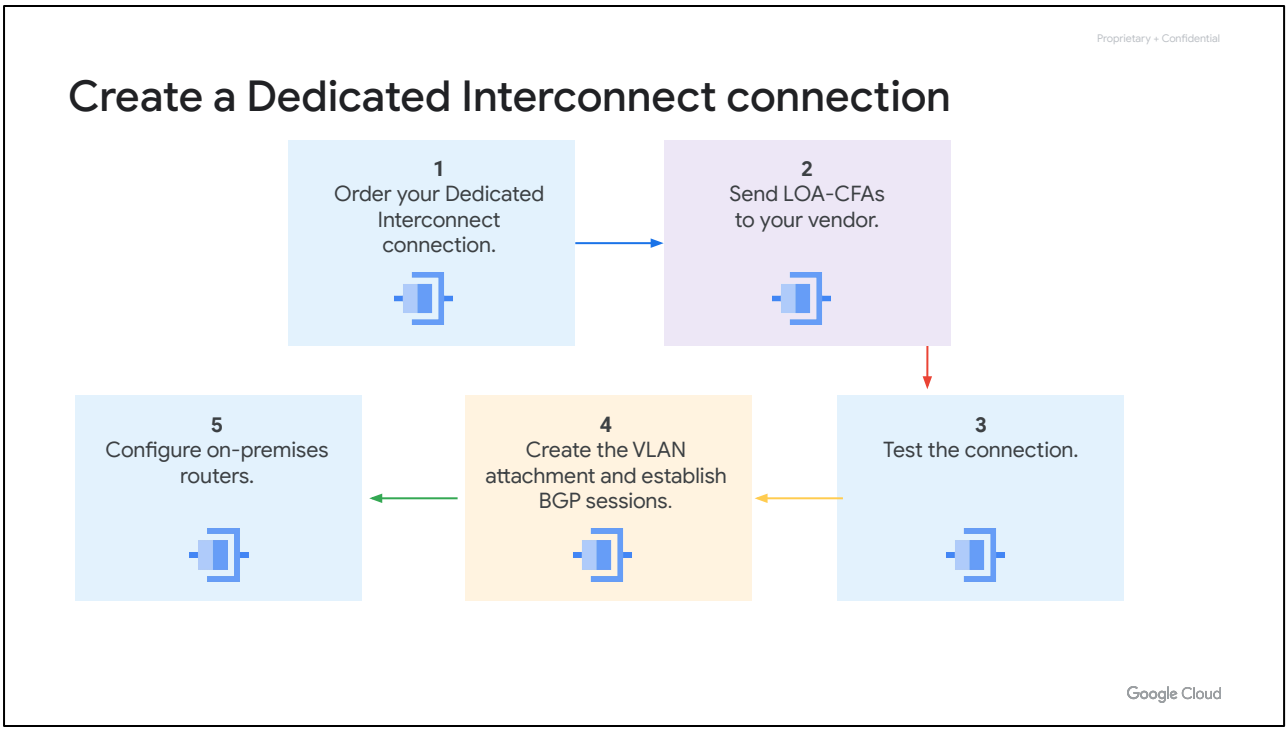
In order to provide redundancy, consider peering edge placement.

Every metropolitan area with colocation facilities has at least two edge availability domains.

Placing a Dedicated Interconnect connection in more than one domain in a metropolitan area provides redundancy. The domains provide isolation during scheduled maintenance, which means that two domains in the same metropolitan area are not down for maintenance at the same time. If you have a Dedicated Interconnect connection defined in each of the two domains, scheduled maintenance can only affect a single connection at any given time.

In the example shown on the slide, there are two Dedicated Interconnect connections in Chicago. One connection is located in a colocation facility in the edge availability domain ord-zone1-7. The other connection is located in a colocation facility in the edge availability domain ord-zone2-7.

For a complete list of colocation facilities and edge availability zones, see All colocation facilities in the Google Cloud documentation. Note that the documentation also refers to an edge availability zone as an Interconnect location name.



To create a Dedicated Interconnect connection, follow these steps:

1. Order your connection; you can do this within Google Cloud. Next, Google sends you a LOA-CFA—that is, a Letter of Authorization and Connecting Facility Assignment. The LOA-CFA identifies the connection ports that Google has assigned for your Dedicated Interconnect connection. The LOA-CFA also grants permission for a vendor in a colocation facility to connect to them.

2. Send LOA-CFAs to your vendor, so they can complete your connection setup. Your vendor will let you know when this setup is complete.

3. Test the connection. Google sends you automated emails with configuration information for two different tests. First, Google sends an IP address configuration to test light levels on every circuit in a Dedicated Interconnect connection. After those tests pass, Google sends the final IP address configuration to test the IP connectivity of each connection. Apply these configurations to your Cloud Routers so that Google can confirm connectivity. After all tests have passed, your Dedicated Interconnect connection is ready to use.

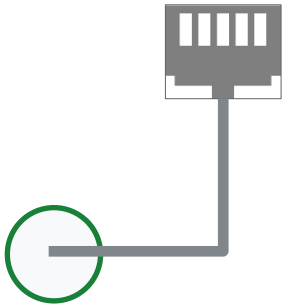
4. Create VLAN attachments and establish BGP sessions. You can do this using the Google Cloud console.

5. Configure the on-premises routers to establish a BGP session with your Cloud

1. Router. To configure your on-premises router, use the VLAN ID, interface IP address, and peering IP address provided by the VLAN attachment.

Connection bandwidth and circuits

- A Dedicated Interconnect connection consists of one or more circuits.
- The circuits in a connection can be 10 Gbps or 100 Gbps, but not both.
- A connection can have one of the following maximum capacities:
  - Eight 10-Gbps circuits (80 Gbps total)
  - Two 100-Gbps circuits (200 Gbps total)

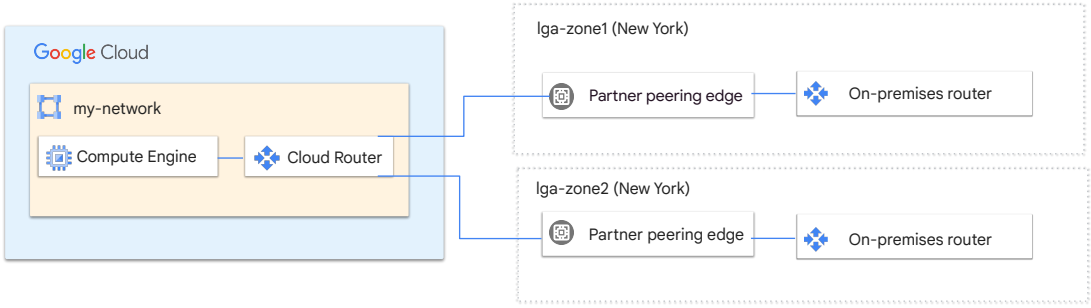


Next, let's talk about the bandwidth of your connection. You can purchase bandwidth as one or more circuits. Each circuit can be either 10 Gbps or 100 Gbps, but not both. You cannot have different types of circuits in the same connection.

A connection can have a maximum of eight 10-Gbps circuits, or two 100-Gbps circuits. Therefore, the maximum connection capacity is either 80 Gbps or 200 Gbps, depending on which type of circuit you choose.

Partner Interconnect

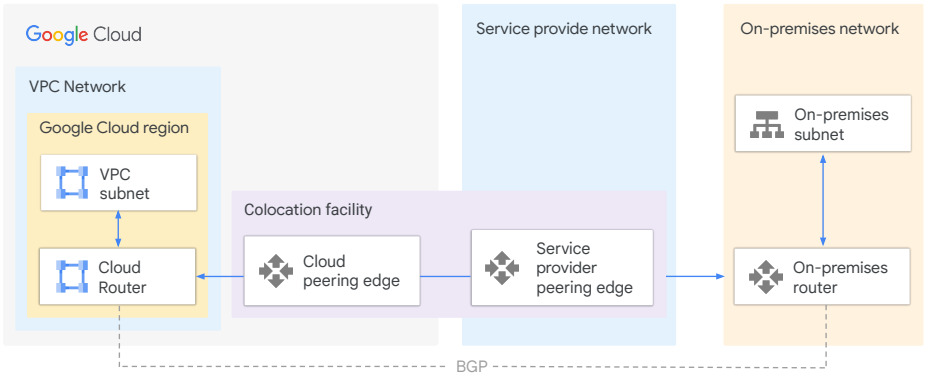
- Partner Interconnect is similar to Dedicated Interconnect.
- The physical connection is not made through a supported service provider.



Partner Interconnect is similar to Dedicated Interconnect. Dedicated Interconnect and Partner Interconnect have technical feature parity.

However, the physical connection for Dedicated Interconnect is not made through a supported service provider. Let's look at a few more differences on the next slide.

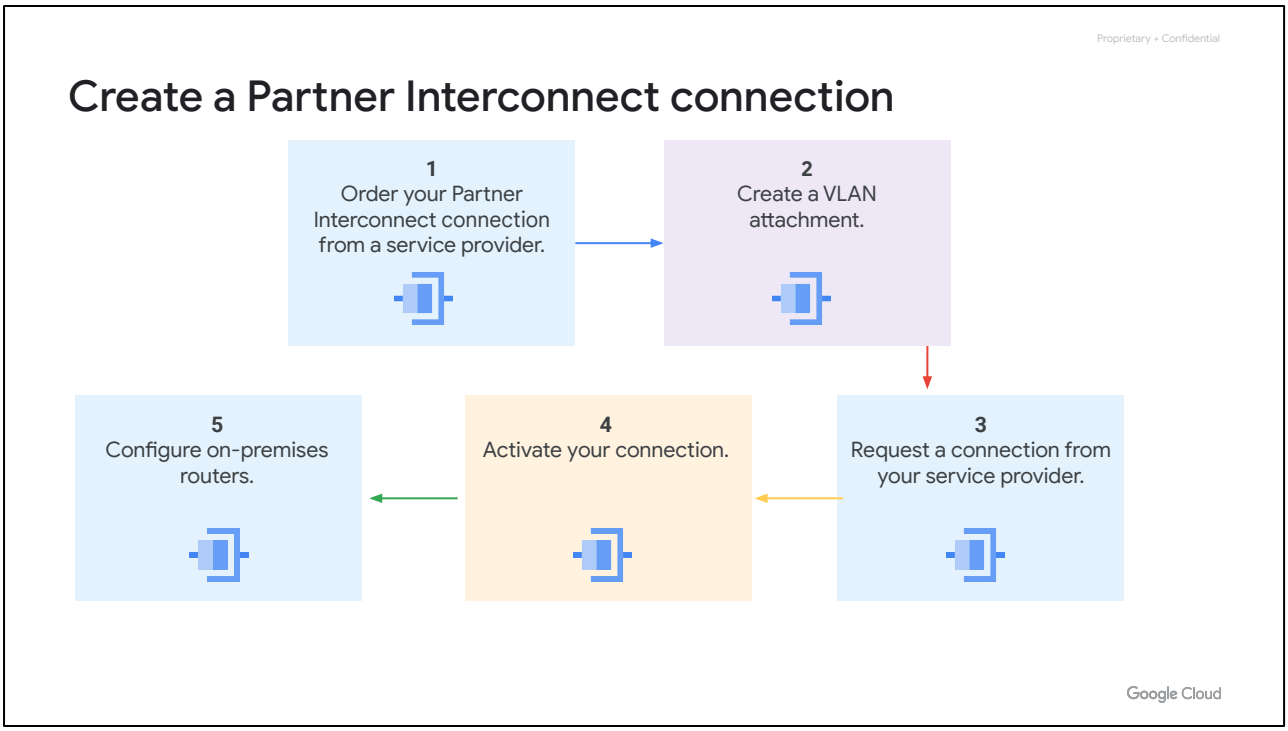
Partner Interconnect provides connectivity through a supported service provider



Partner Interconnect provides connectivity between your on-premises network and your VPC network through a supported service provider. If your data center is in a physical location that can't reach a Dedicated Interconnect colocation facility, Partner Interconnect is a good option. If your data needs don't warrant using Dedicated Interconnect, consider using Partner Interconnect. Work with a supported service provider to connect your VPC and on-premises networks.

Consider placing the Partner Interconnect connection in multiple edge availability domains for redundancy.

For a full list of service providers, see *Supported service providers* in the Google Cloud documentation at https://cloud.google.com/interconnect/docs/concepts/service-providers.



To create a Partner Interconnect connection, follow these steps:

1. Order your connection from a supported service provider. For a list of service providers in your area, see Supported service providers in the Google Cloud documentation. The service provider will then provide the connectivity needed to create a VLAN attachment.

2. Create a VLAN attachment, which creates a pairing key. The pairing key is unique and lets a service provider identify and connect to the associated Cloud Router. The service provider uses this key to finish configuring your VLAN attachment.

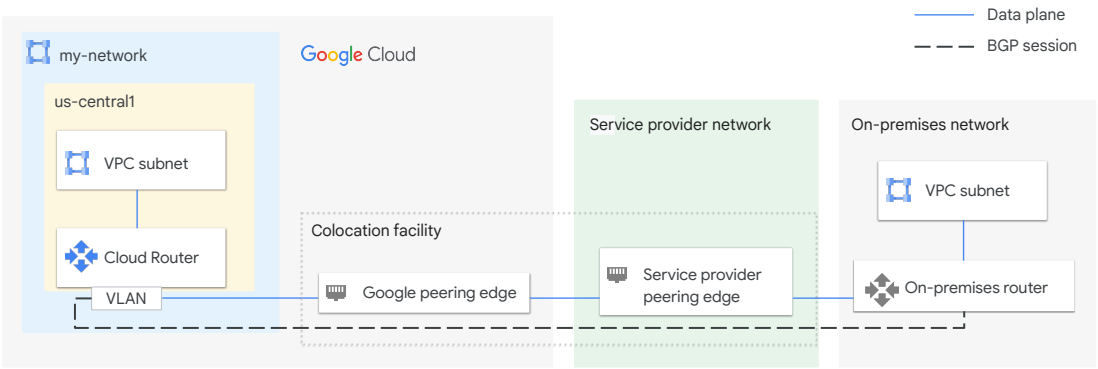
3. Request a connection from your service provider. Submit the pairing key and other connection details, such as the connection capacity and location. Your service provider configures your connection; they must confirm that they can serve your requested capacity. When the configuration is complete, you'll receive an email.

4. In the VLAN attachment, activate your connection. After the connection is activated, it can start passing traffic.

5. Configure the on-premises routers to establish a BGP session with your Cloud Router. To configure your on-premises routers, use the VLAN ID, interface IP address, and peering IP address provided by the VLAN attachment.

Layer 2 connections

For each VLAN attachment, configure and establish a BGP session between your Cloud Routers and on-premises routers.



For Layer 2 connections, traffic passes through the service provider network to reach the VPC network or on-premises network. BGP is configured between the on-premises router and a Cloud Router in the VPC network, as shown in the graphic.

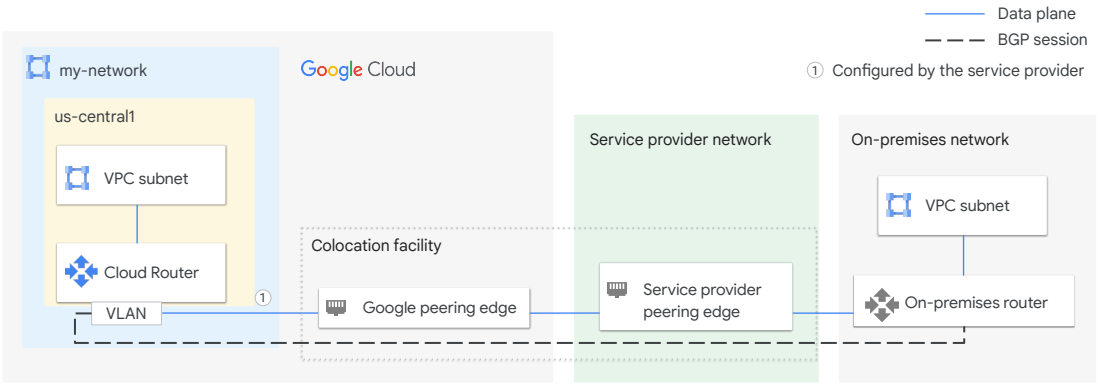
For Layer 2 connections, you must configure and establish a BGP session between your Cloud Routers and on-premises routers. When you configure Cloud Router, you configure VLAN (virtual local area network) attachments. Each VLAN attachment is a logical connection between your on-premises network and a single region in your VPC network.

When creating a VLAN attachment, specify a Cloud Router in the region that contains the subnets that you want to reach. The VLAN attachment automatically allocates a VLAN ID and BGP peering IP addresses. Use that information to configure your on-premises router and establish a BGP session with your Cloud Router.

For Partner Interconnect, the VLAN attachment uses a connection that your service provider sets up and manages. The service provider completes the circuit.

Layer 3 connections

Your service provider completes VLAN connections from the service provider peering edge to Cloud Router.



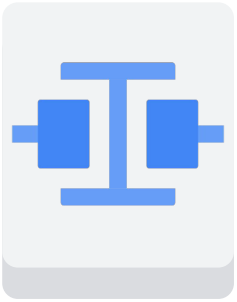
For Layer 3 connections, traffic is passed to the service provider network. Their network then routes the traffic to the correct destination, either to the on-premises network or to the VPC network. Connectivity between the on-premises network and the service provider network depends on the service provider. For example, the service provider might request that you establish a BGP session with them or configure a static default route to their network.

For Layer 3 connections, your service provider establishes a BGP session between your Cloud Routers and their on-premises routers for each VLAN attachment. You don't need to configure BGP on your local router. Google and your service provider automatically set the correct BGP configurations.

Partner Interconnect recommendations

Use Partner Interconnect when you:

- Cannot physically connect from a Google colocation facility.
- Need to procure a connection quickly.
- Want a single port for multi-cloud use.
- Have lower bandwidth needs (50 Mbps - 10 Gbps).



Dedicated Interconnect and Partner Interconnect have technical feature parity. The biggest difference is where you interconnect; from a Google colocation facility or from a partner facility. However, there are some other points to consider.

Use Partner Interconnect when you cannot physically connect from a Google colocation facility but can use a partner colocation facility.

Partner Interconnect can be procured quickly. The physical configuration already exists at the service provider, so there's less infrastructure to set up.

If you want a single port for multi-cloud use, Partner Interconnect may be a good choice. For example, Partner Interconnect is useful when you connect one VLAN to Google, and another to another cloud provider. The SLA (Service Level Agreement) to support this usage would be through the service provider, not Google.

Partner Interconnect also is sufficient to support bandwidth needs less than 10 Gbps. Dedicated Interconnect is only available in 10-Gbps or 100-Gbps circuits. A Partner Interconnect connection can be scaled based on the number and capacity of your VLAN attachments; thus, the smallest connection is 50 Mbps. If you need less than 50 Mbps, consider using Cloud VPN.

For 99.99% availability, consider the number of Cloud Interconnect connections

- Set the VPC network's dynamic routing to global.
- Configure at least four Interconnect connections: two connections in one metropolitan area and two connections in another.

To achieve 99.99% availability for Dedicated Interconnect and Partner Interconnect - in other words, for Cloud Interconnect in general - you must consider how and where you connect.

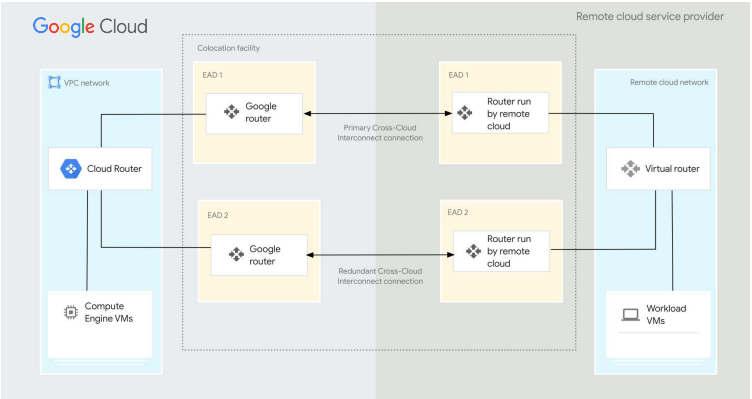
The dynamic routing mode for the VPC network must be set to global.

You must have at least four Cloud Interconnect connections: two connections in one metropolitan area and two connections in another. Connections that are in the same metro must be placed in different edge availability domains. If a region-wide issue occurs, Google Cloud can reroute traffic through the other region to your VMs. Each Cloud Router must be attached to a pair of Cloud Interconnect connections in a metropolitan area, with two VLAN attachments for each Cloud Router.

In the slide example, you can see that there are four Cloud Interconnect connections - two in Tokyo and two in Osaka. If one of the connections in Tokyo goes down, the other one can take over. If both Tokyo connections go down, the Osaka connections can take over. When planning the number of Cloud Interconnect connections to install, you should also consider your bandwidth needs.

Cross Cloud Interconnect

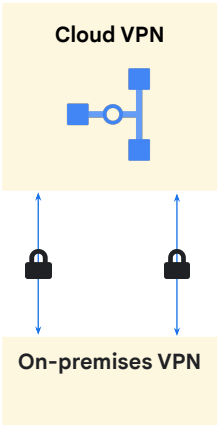
- High speed connection with other public clouds
- Reduces complexity
- 10Gbps or 100Gbps links
- Encryption supported



https://cloud.google.com/network-connectivity/docs/interconnect/concepts/cci-overview

Cloud VPN securely connects your on-premises network to Google Cloud

- Use Cloud VPN when you:
  - Don't need the connection speed of Cloud Interconnect.
  - Must encrypt data in transit.
  - Want to selectively advertise routes between VPC networks.
- Cloud VPN securely connects your peer network to a Virtual Private Cloud (VPC) network through IPsec tunnels.
- Traffic traveling between the two networks is encrypted.



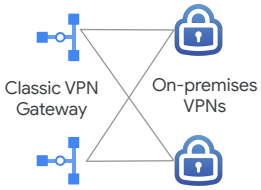
Use Cloud VPN when you don't need the connection speed of Cloud Interconnect. Cloud VPN is cheaper and easier to set up than Cloud Interconnect. Thus, Cloud VPN is useful for low-volume or low-bandwidth data connections. You also use Cloud VPN to encrypt data in transit.

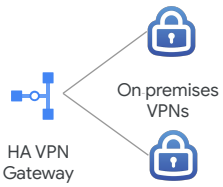
With Cloud VPN, you can selectively advertise routes between VPC networks. In contrast, when you set up peering between two VPC networks, all the subnet routes are advertised.

Cloud VPN securely connects your peer network to your Virtual Private Cloud (VPC) network through IPsec tunnels. The data is encrypted as it passes through the tunnels. The traffic traveling between the two networks is encrypted by one VPN gateway and then decrypted by the other VPN gateway. This action protects your data as it travels over the internet.

HA VPN and Classic VPN

- There are two types of Cloud VPN: HA (high availability) VPN and Classic VPN.
- For static routing, use Classic VPN.
- For BGP routing, use HA VPN.





There are two types of Cloud VPN: HA (high availability) VPN and Classic VPN. Although the Google Cloud documentation shows that Classic VPN has been partially deprecated, it is still a valid option for static route connectivity. For BGP routing, you must use HA VPN.

Next, let's discuss both Cloud VPN products.

HA VPN topologies

HA VPN supports site-to-site VPN for different configuration topologies:

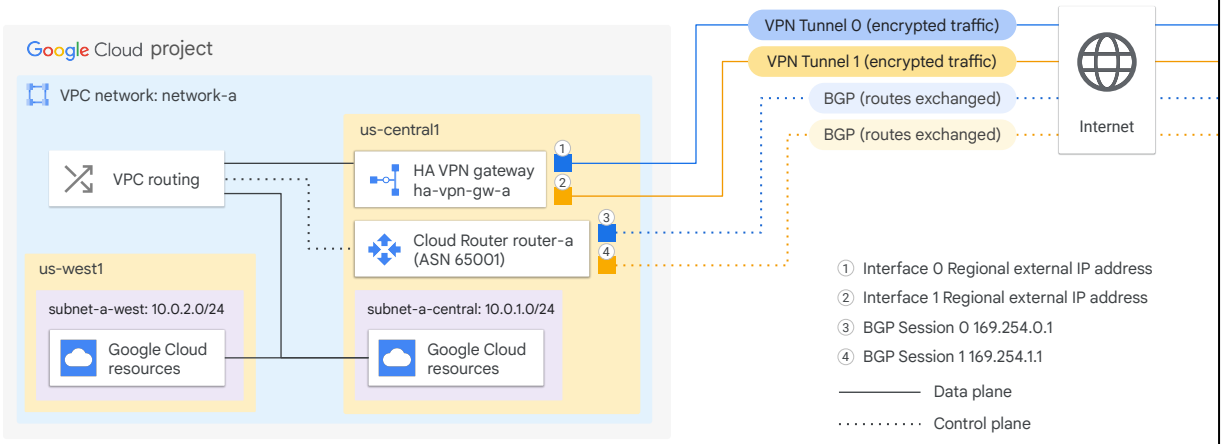
- An HA VPN gateway to peer VPN devices
- An HA VPN gateway to an Amazon Web Services (AWS) virtual private gateway
- Two HA VPN gateways connected to each other

HA VPN supports site-to-site VPN for different configuration topologies. These topologies are:

- An HA VPN gateway to peer VPN devices
- An HA VPN gateway to an Amazon Web Services (AWS) virtual private gateway
- Two HA VPN gateways connected to each other

Let’s look at each of these topologies.

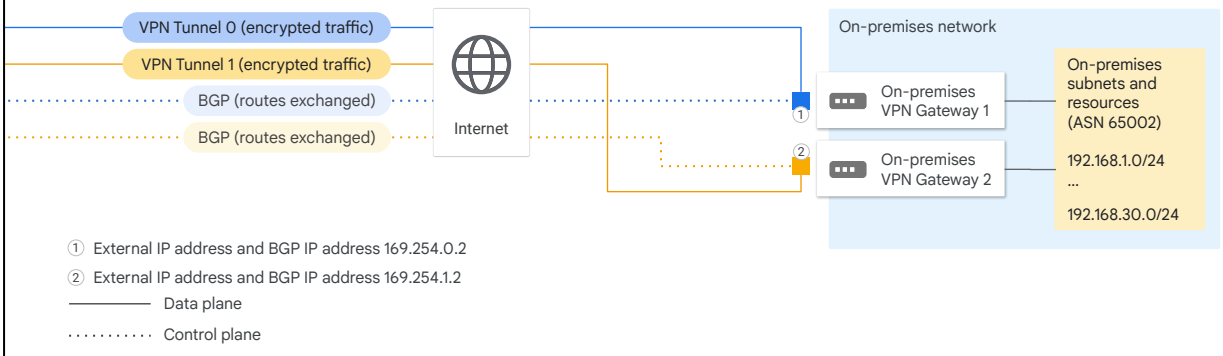
HA VPN to VPN peer gateway: Google Cloud to internet



HA VPN has three typical peer gateway configurations: an HA VPN gateway to two separate peer VPN devices, each with its own IP address; one peer VPN device that uses two separate IP addresses; and one peer VPN device that uses one IP address.

Let's walk through an example. In this topology, one HA VPN gateway connects to two peer devices. Here, we see the HA VPN gateway and two VPN tunnels that connect to the peer devices. Next, we'll look at the peer devices in the on-premises network.

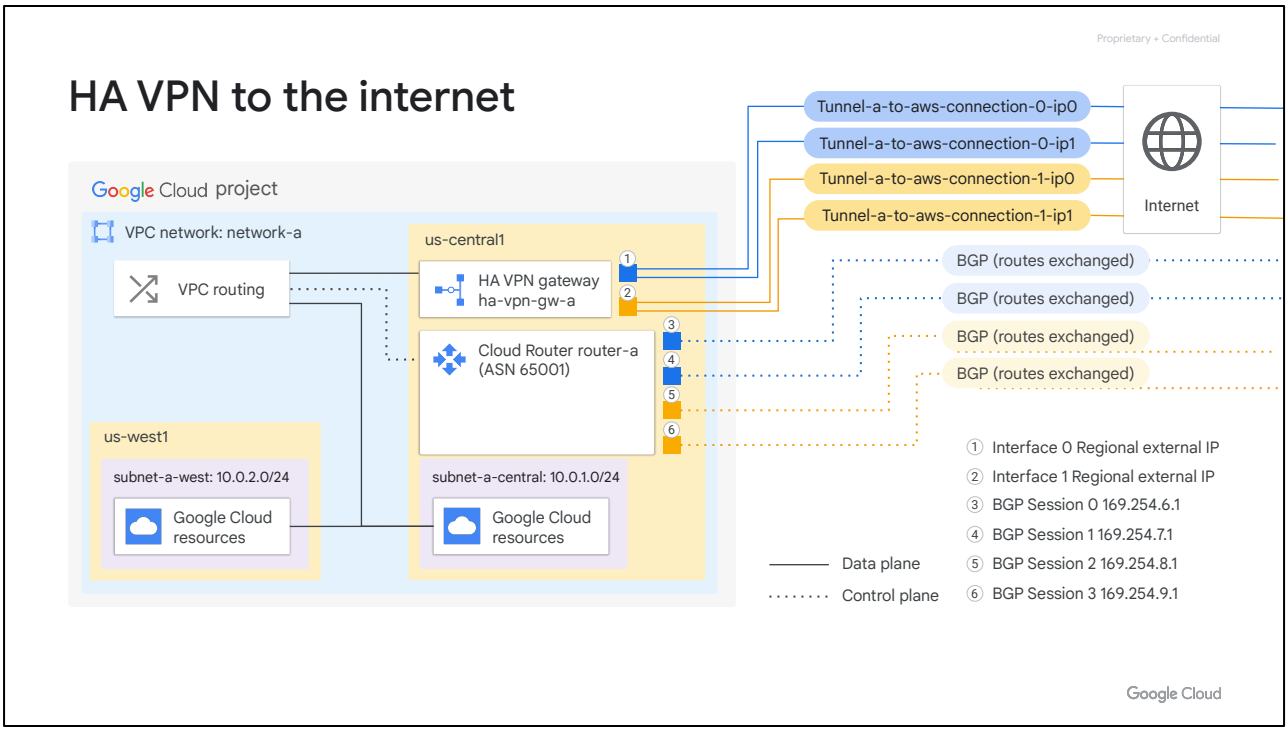
HA VPN to VPN Peer gateway: Internet to the on-premises network



Here you can see the on-premises network and the tunnels that come from the HA VPN gateway.

Each peer device has one interface and one external IP address. The HA VPN gateway uses two tunnels, one tunnel to each peer device. If your peer-side gateway is hardware-based, having a second peer-side gateway provides redundancy and failover on that side of the connection.

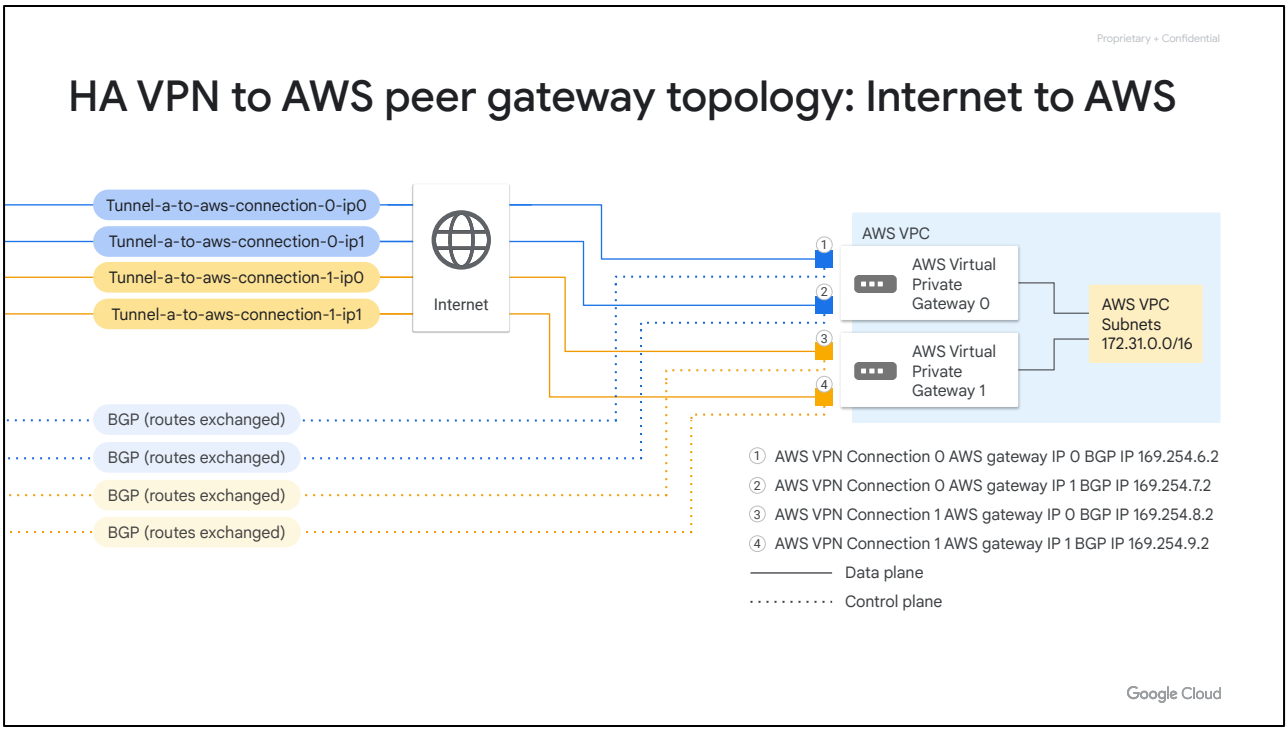
In Google Cloud, the REDUNDANCY_TYPE for this configuration takes the value TWO_IPS_REDUNDANCY. The example shown here provides 99.99% availability.



When configuring an HA VPN external VPN gateway to Amazon Web Services (AWS), you can use either a transit gateway or a virtual private gateway. Only the transit gateway supports equal-cost multipath (ECMP) routing. When enabled, ECMP distributes traffic equally across active tunnels. Let's walk through an example.

In this topology, you configure three major gateway components: an HA VPN gateway in Google Cloud with two interfaces; two AWS virtual private gateways that connect to your HA VPN gateway; and an external VPN gateway resource in Google Cloud that represents your AWS virtual private gateway. This resource provides information to Google Cloud about your AWS gateway.

Here you can see the Google Cloud components and their connections through the internet to the AWS components.

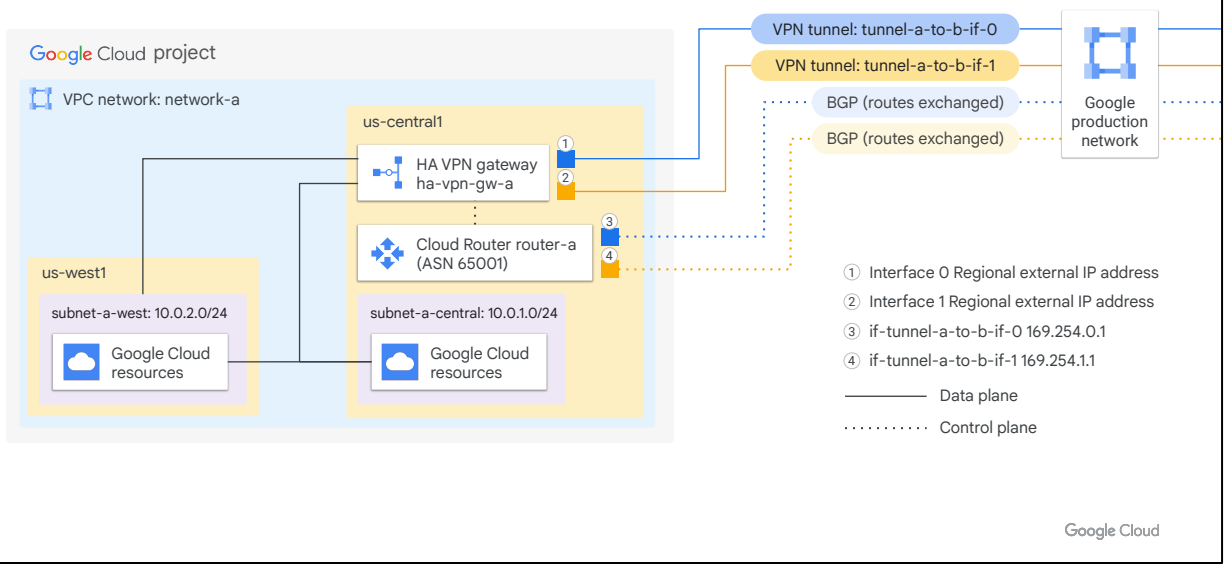


The AWS components in this topology are shown here, along with their connections to the HA VPN Gateway and Cloud Router in Google Cloud.

The supported AWS configuration uses a total of four tunnels: two tunnels from one AWS virtual private gateway to one interface of the HA VPN gateway, and two tunnels from the other AWS virtual private gateway to the other interface of the HA VPN gateway.

For information regarding using HA VPN to connect to a Microsoft Azure gateway, see Using Cloud VPN With Microsoft Azure(TM) VPN Gateway.

HA VPN to peer VPN gateway topology: Left side



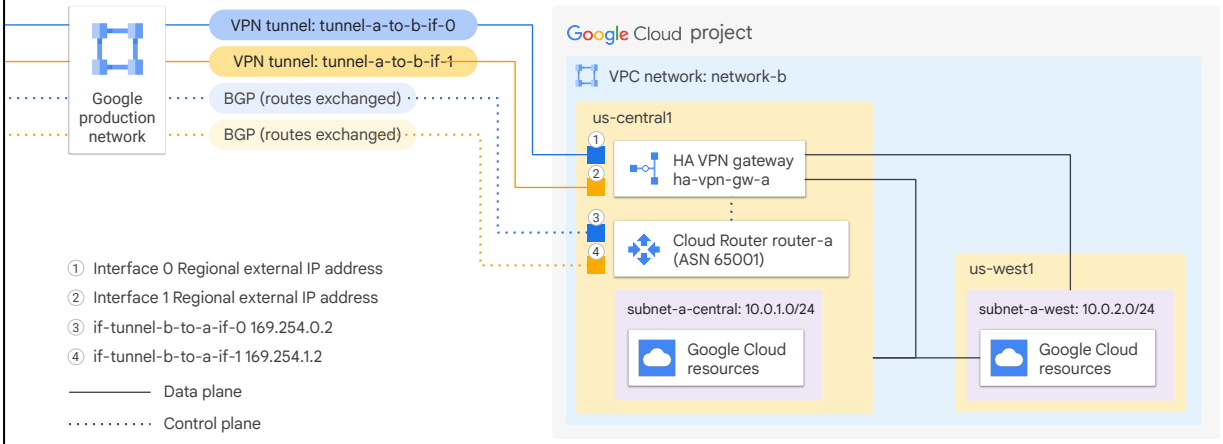
You can connect two Google Cloud VPC networks together by using an HA VPN gateway in each network. Let's walk through a sample topology. Like the other two samples that you've seen, it's divided into two parts.

Here you see a Google Cloud project with a VPC network called network-a. There's an HA VPN gateway and a Cloud Router instance that connects to VPC network-b, which is not visible here.

Each HA VPN gateway has two interfaces, shown in the graphic. You connect `interface 0` on the HA VPN gateway in network_a to `interface 0` on the HA VPN in network_b. You do the same for the `interface 1`, connecting from the HA VPN gateway in network-a to the HA VPN gateway in network-b.

Next, let's look at network-b.

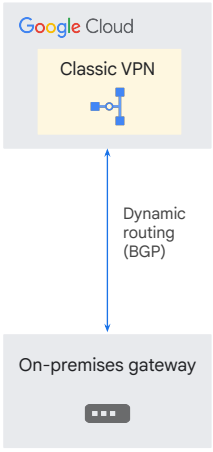
HA VPN to peer VPN gateway topology: Right side



Here you see the Google Cloud project that contains network-b. You see all the connections from the HA VPN gateway that link to network-a.

HA VPN recommendations

- HA VPN
  - Provides 99.99% service availability (instead of 99.9% for Classic VPN.)
  - Supports multiple VPN tunnels.
- Google Cloud automatically chooses two external IP addresses.
- VPN tunnels connected to HA VPN gateways must use dynamic (BGP) routing.
- Use an Active/Passive configuration for a consistent bandwidth experience.



HA VPN also provides 99.99% service availability.

Use HA VPN for BGP routing and to support multiple tunnels. When you create an HA VPN gateway, Google Cloud automatically chooses two external IPv4 addresses, one for each of its fixed number of two interfaces. Each IPv4 address is automatically chosen from a unique address pool to support high availability.

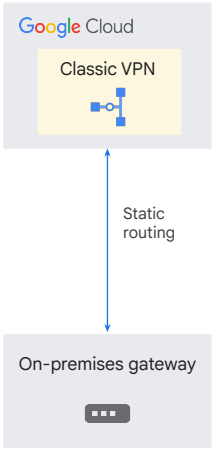
VPN tunnels connected to HA VPN gateways must use dynamic (BGP) routing.

Use an Active/Passive configuration for a consistent bandwidth experience.

Active/Active configurations may offer a less consistent experience. Unless combined traffic for both tunnels is within single tunnel capacity, failure can cause the available bandwidth to be cut in half.

Classic VPN recommendations

- Classic VPN is meant only for any of these situations:
  - When static routing must be used
  - To connect to VPN gateway software running inside a Compute Engine instance
  - To connect to gateways that don't support BGP
- Supports 99.9% service availability (instead of 99.99 for HA VPN).
- For all other situations, use HA VPN.



Google recommends that you use HA VPN whenever possible. However, there are some situations where Classic VPN is useful.

You can continue to create Classic VPN tunnels that use static routing.

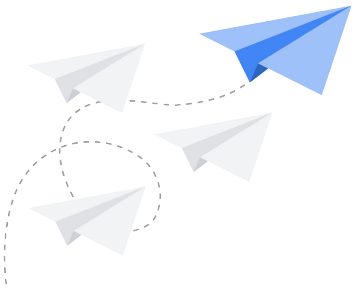
Using Classic VPN for dynamic routing is no longer supported—with one exception. To connect to VPN gateway software running inside a Compute Engine instance, you can still use Classic VPN. Classic VPN also remains an option for connecting to gateways that don't support BGP. For more information about this and other Classic VPN use cases, see Classic VPN topologies in the Google Cloud documentation.

Classi VPN supports 99.9% service availability. HA VPN supports a higher service availability of 99.99%.

For all other situations, use HA VPN. For more information, see Classic VPN partial deprecation in the Google Cloud documentation.

Influence the best path selection by setting a base priority

- When a Cloud Router advertises prefixes to a BGP peer, it advertises a route priority for each prefix.
- You can change the base priority on each Cloud VPN tunnel or VLAN attachment.
- The base priority is added to the region-to-region cost to calculate the value of the BGP MED attribute.



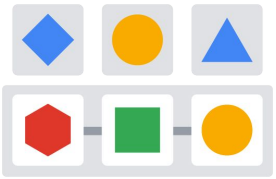
When a Cloud Router advertises prefixes to a BGP peer, it includes a priority for each prefix in the advertisement. The advertisement is the BGP message, and it includes a route priority. The route priority is stored in the BGP MED attribute, which influences route selection.

You can change the advertised route priority value on each Cloud VPN tunnel or VLAN attachment. This value is then assigned to the BGP MED (Multi-Exit Discriminator) attribute. The MED value affects the BGP best path selection. By changing the advertised route priority, you influence the best path selection. For more information about other factors that influence the best path selection, see the BGP protocol standard on the IETF.org website or to a networking tutorial.

The advertised route priority value applies to all prefixes advertised by the BGP session associated with the Cloud VPN tunnel or VLAN attachment.

Setting a base priority

- Base priorities are whole numbers from 0 to 65535.
- The highest possible base priority is 0.
- The default base priority is 100.
- If you don't specify a base priority, the default priority is used.



Base priorities are whole numbers from 0 to 65535. The highest possible base priority is 0. In other words, a lower number indicates a higher priority.

The default base priority is 100. If you don't specify a base priority, the default priority is used.

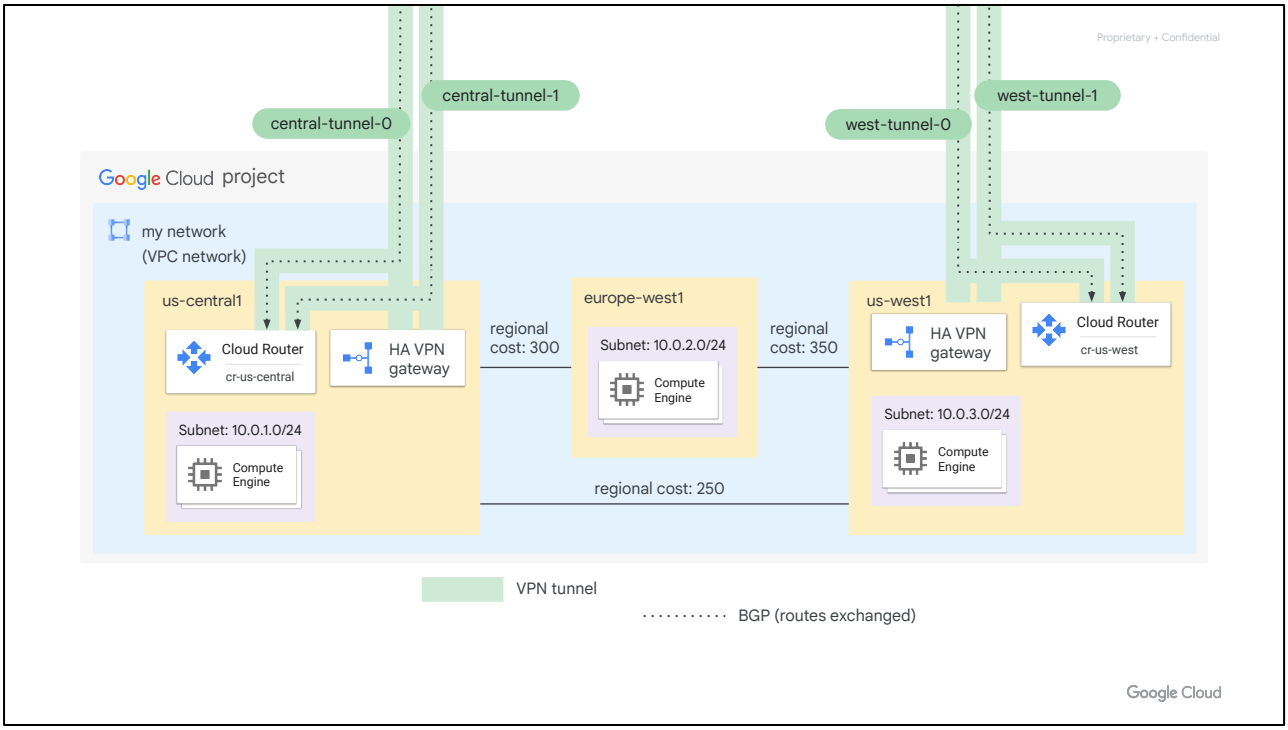
Next, let’s talk about the region-to-region costs that can be added to the base priority to set the MED attribute on the BGP session.

Region-to-region costs

- Region-to-region costs are from 201 through 9999, inclusive.
- The value depends on the distance, latency, and other factors between two regions.
- Google generates the region-to-region cost values, and you can't modify them.

Region-to-region costs are from 201 through 9999. The value depends on the distance, latency, and other factors between two regions. Google generates the region-to-region cost values, and you can't modify them.

Next, let’s look at a sample topology to see how this works.



In the graphic, you see an example of region-to-region costs that Google calculates. Regions that are closer have a lower region cost for traffic that flows between them. For example, the region cost between us-central1 and europe-west1 is lower than the cost between europe-west1 and us-west1.

For more information, see Advertised prefixes and priorities in the Google Cloud documentation.

Manage multi-site connectivity with Network Connectivity Center

Network Connectivity Center lets you:

- Integrate third-party virtual network appliances from:
  - External sites and VPC networks.
  - VPC networks to other VPC networks.
- Configure site-to-site data transfer, using Google's network.
- Centrally manage multi-site configurations.

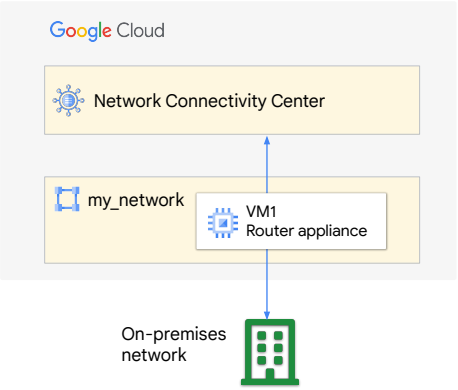
Network Connectivity Center adds some useful features to Google Cloud networking. You can integrate third-party network appliances between external sites to your VPC networks or even between VPC networks.

For supported regions, you can use the Google Cloud network to configure site-to-site data transfer. After configuration, external sites use Google Cloud network as a WAN, which reduces latency.

Network Connectivity Center also lets you centrally manage multi-site configurations between your external sites and Google Cloud.

Router appliance

- Network Connectivity supports virtual router appliances.
- With this feature, you can use a third-party SD-WAN router or another appliance to:
  - Connect an external network to Google Cloud.
  - Implement enhanced security and dynamic routing protocols with BGP.
- Router appliance lets you use Google's backbone network as a WAN to interconnect remote sites.



Network Connectivity Center also supports virtual router appliances. You install the third-party virtual appliance on a Compute Engine VM. You can install your own virtual appliance image or use an image provided by a supported Network Connectivity Center partner. Router appliance also supports using a third-party SD-WAN (software-defined wide area network) router.

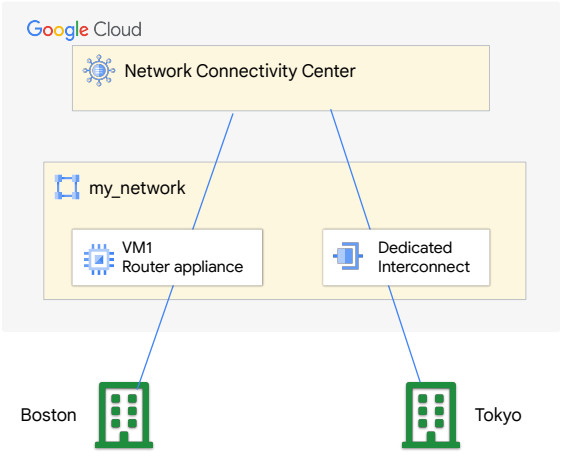
Using Router appliance and Network Connectivity Center, you can connect an external network to Google Cloud by using a third-party SD-WAN router or another appliance. This approach is known as site-to-cloud connectivity.

Third-party network appliances and SD-WAN router products can manage connectivity between your on-premises and VPC networks. These products enable you to implement security or any custom logic pertaining to network connections. For example, you can integrate third-party firewalls to fine-tune your security posture.

With Router appliance, you can use the Google network as a WAN (wide area network) to connect sites that are outside Google Cloud. This approach is known as site-to-site data transfer.

Site-to-site data transfer

- The site-to-site data transfer feature lets your on-premises sites or other cloud workloads use Google Cloud as a WAN.
- This feature can result in lower latency and greater reliability than connecting over the public internet.
- The sites connect to Google Cloud using a router appliance instance, Cloud Interconnect, or Cloud VPN.

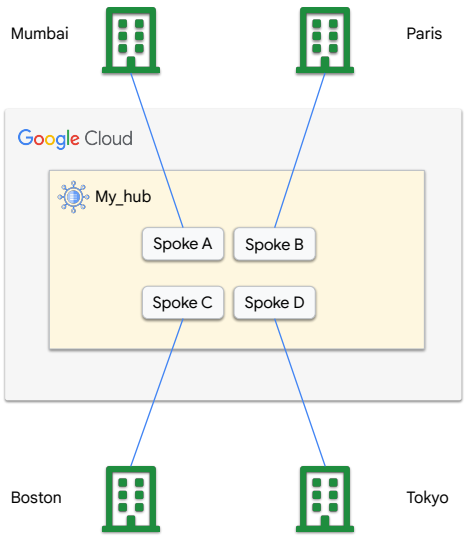


Site-to-site data transfer lets your on-premises sites or other cloud workloads use Google Cloud as a wide area network (WAN). This feature results in lower latency and greater reliability than connecting over the public internet.

The sites can connect to Google Cloud using a router appliance instance, Cloud Interconnect, or Cloud VPN.

The hub-and-spoke model

- Network Connectivity Center uses a hub-and-spoke model to manage hybrid connections.
- Each connectivity resource is represented as a spoke.
- Spokes are centrally managed by a hub.

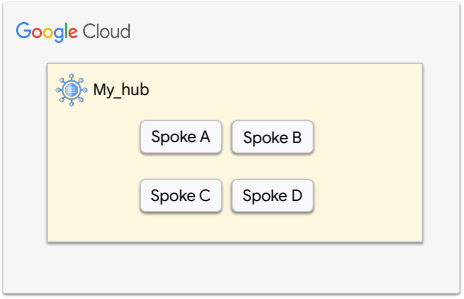


Network Connectivity Center uses a hub-and-spoke model to manage hybrid connections inside and outside Google Cloud. With this architecture, each connectivity resource is represented as a spoke.

Each spoke is attached to a central management resource known as a hub.

The hub

- The hub contains routing table entries for a group of spokes.
- The hub also:
  - Facilitates route updates between the spokes and the sites they represent.
  - Provides full mesh connectivity between spokes with site-to-site data transfer enabled.



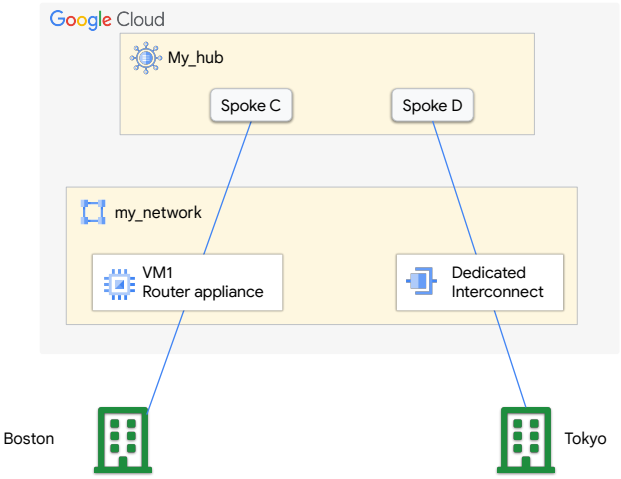
The hub contains routing table entries for a group of spokes. Each Google Cloud project can have only one hub.

The hub function varies depending on whether any of the spokes use the site-to-site data transfer feature. When you use this feature, the hub provides full mesh connectivity between all spokes that have the feature enabled. If data transfer is not enabled for any spokes, the hub provides connectivity only to Google Cloud resources. The hub does not establish connectivity between these spokes.

Next, let's look at the spokes.

The spokes

- Spoke represents a collection of network resources that injects network prefixes into the hub.
- For each spoke, you assign a connection type.
- The connection types are:
  - Dedicated Interconnect
  - Partner Interconnect
  - HA VPN
  - Router appliance



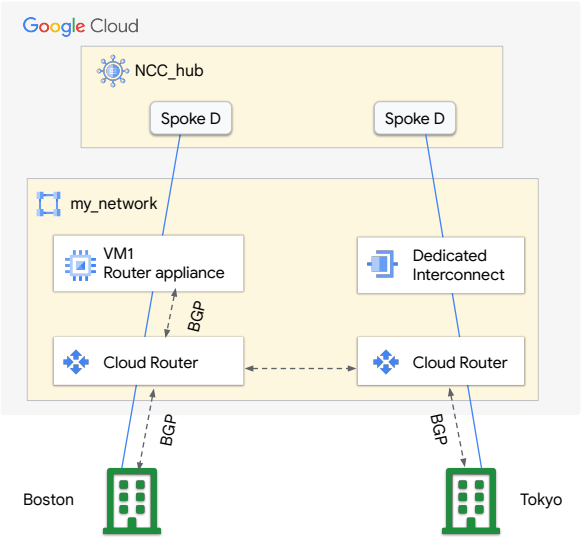
A spoke represents a collection of network resources that injects network prefixes into the hub. For best performance, you create each spoke in the region closest to the network resource that you're connecting. For example, if you were creating a spoke for a site in Tokyo, Japan, you would use the asia-northeast-1 region.

For each spoke, you associate a VPC network and assign a connection type. In the example shown on the slide, spokes C and D are both associated with the same VPC network, my_network. However, except for site-to-site data transfer, spokes do not need to use the same VPC network. The VPC network isn't required to be the same on all the spokes.

In addition to Router appliance, the sites can connect to Google Cloud using Dedicated Interconnect, Partner Interconnect, or HA VPN. Each of these connections is sometimes referred to as a backing resource.

Exchanging routes throughout the hub

- Router appliance establishes BGP peering between the VM and a Cloud Router.
- Route exchange enables connectivity between a VPC network and other networks.



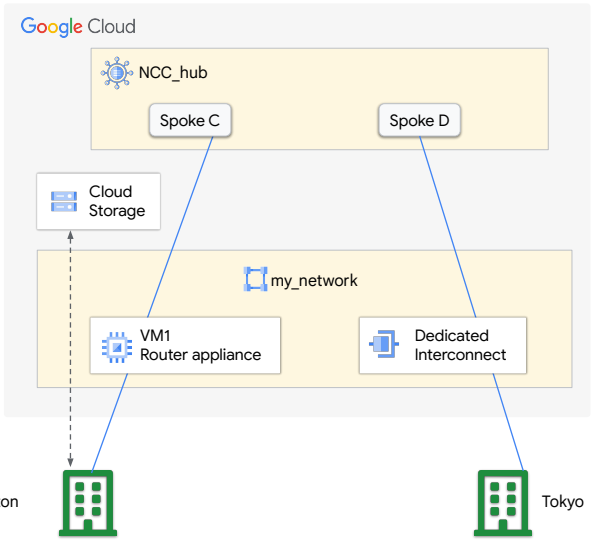
Router appliance establishes BGP peering between the VM and a Cloud Router.

To create a Router appliance instance, you install a virtual appliance image on a Compute Engine virtual machine and complete some other setup steps. This setup includes establishing Border Gateway Protocol peering between the VM and a Cloud Router. BGP enables the dynamic exchange of routes between the Cloud Router and the Router appliance instance.

Route exchange lets you establish connectivity between your VPC network and other networks.

Site-to-cloud connectivity

When you configure a spoke, site-to-cloud connectivity is automatically included.



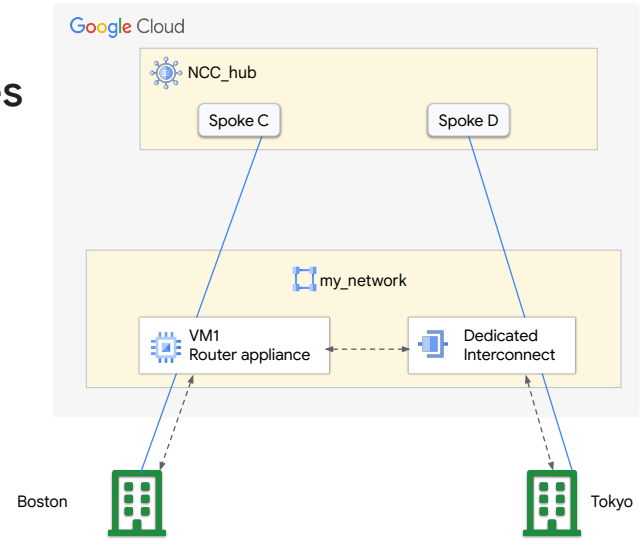
Control plane

Site-to-cloud connectivity

When you configure a spoke, site-to-cloud connectivity is automatically included. On-premises sites can connect to resources in Google Cloud. In the example, you can see that an on-premises site in Boston connects to Google Cloud using the Router appliance feature. You can also see that the on-premises sites also have site-to-cloud connectivity. For example, you see here that the Boston remote site connects to Cloud Storage.

Site-to-site data transfer is configured at the spokes

Optionally, you can configure a spoke to support site-to-site data transfer.



Site-to-site data transfer can be enabled on a spoke, using the Google Cloud console or the Google Cloud CLI. The site-to-site data transfer feature is optional; you only configure this feature for desired spokes. Spokes with this feature enabled can move data between them. In the example, you see that on-premises sites in Boston and Tokyo can move data between them. This data goes through Google Cloud.

Caveats: Network Connectivity Center

Network Connectivity Center does not support:

- IPv6
- BGP communities
- Legacy VPC networks
- Classic VPN tunnels

Classic VPN tunnels are not supported. Routes installed by the hub are treated as dynamic routes.

Network Connectivity Center does not support IPv6. For example, if a spoke has site-to-site data transfer enabled, the resources associated with the spokes cannot exchange IPv6 traffic. When you create a VM to use as a Router appliance instance, the VM must use an IPv4 address - specifically, an RFC 1918 address.

BGP communities are not supported.

Legacy networks are not supported. Legacy networks are older networks that do not support the latest Google Cloud features. To use Network Connectivity Center with a legacy network, you must recreate it or convert it to a VPC network.

Caveats: Site-to-site data transfer

- Site-to-site data transfer is only available in supported regions.
- There are no bandwidth or latency guarantees.
- Spokes using site-to-site data transfer must all be part of a single VPC network.
- To exchange routes between spokes in multiple regions, the VPC networks must have the dynamic routing mode set to global.
- For each spoke, ensure that all associated on-premises routers advertise identical routes to Cloud Router.

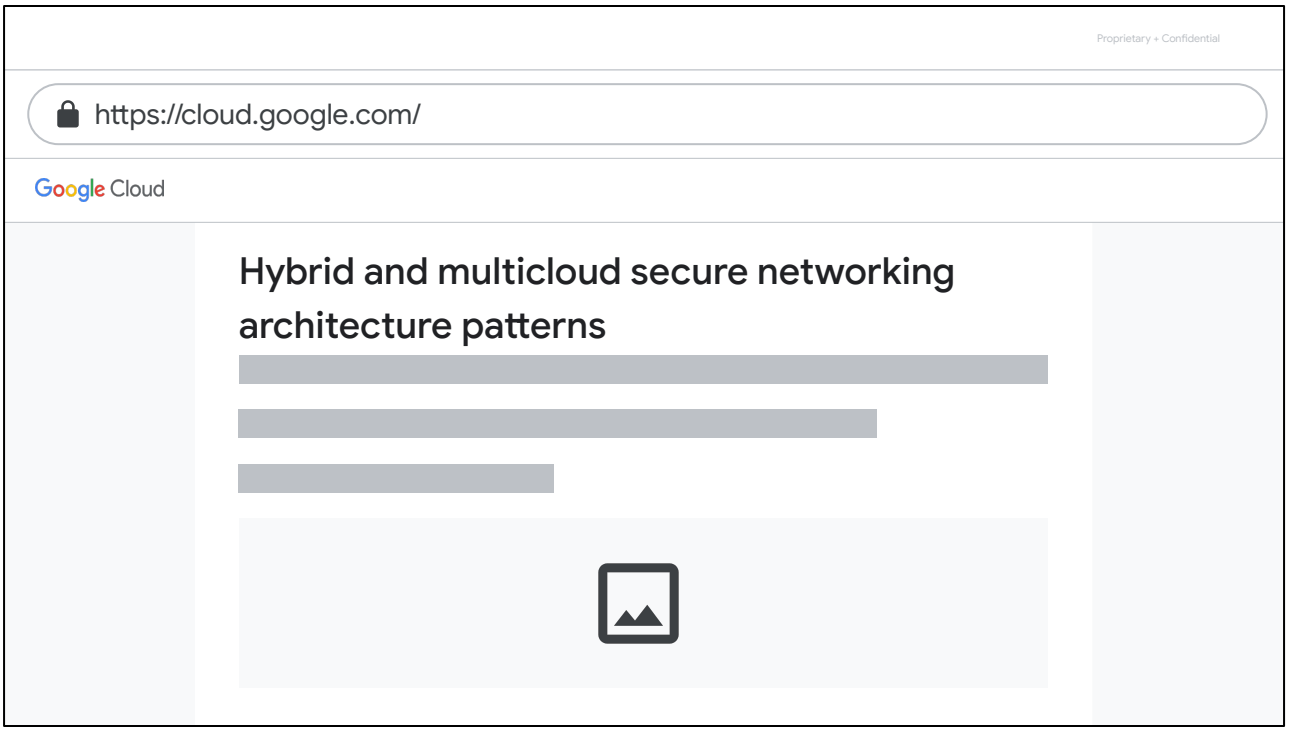
Site-to-site data transfer is only available in supported regions. For more information, see Locations supported for data transfer in the Google Cloud documentation.

Data transfer traffic between sites is on a best-effort basis. There are no bandwidth or latency guarantees.

When data transfer is enabled for one or more spokes, all connectivity resources associated with these spokes must be part of a single VPC network.

If you want to exchange routes between spokes in multiple regions, the VPC network for the spokes must have its dynamic routing mode set to global.

For each spoke, ensure that all associated on-premises routers advertise identical routes to Cloud Router.



For more information on hybrid Networking, visit the ‘Hybrid and multicloud secure networking architecture patters’ documentation found in the Cloud Architecture Center.

Welcome to the Load Balancing, CDN and Hybrid Networking module.