Google Cloud VPC Network Peering allows private RFC 1918 (https://tools.ietf.org/html/rfc1918) connectivity across two Virtual Private Cloud (VPC) networks regardless of whether they belong to the same project or the same organization.

VPC Network Peering enables you to peer VPC networks so that workloads in different VPC networks can communicate in private RFC 1918 space. Traffic stays within Google's network and doesn't traverse the public internet.

VPC Network Peering is useful for:

- SaaS (Software-as-a-Service (https://wikipedia.org/wiki/Software_as_a_service)) ecosystems in Google Cloud. You can make services available privately across different VPC networks within and across organizations.
- Organizations with several network administrative domains can peer with each other.

If you have multiple network administrative domains within your organization, VPC Network Peering allows you to make services available across VPC networks in private RFC 1918 space. If you offer services to other organizations, VPC Network Peering allows you to make those services available in private RFC 1918 space to those organizations. The ability to offer services across organizations is useful if you want to offer services to other enterprises, and it is useful within your own enterprise if you have several distinct organization nodes (/resource-manager/docs/quickstart) due to your own structure or as a result of mergers or acquisitions.

VPC Network Peering gives you several advantages over using external IP addresses or VPNs to connect networks, including:

- Network Latency: Public IP networking suffers higher latency than private networking. All peering traffic stays within Google's network.
- Network Security: Service owners do not need to have their services exposed to the public Internet and deal with its associated risks.
- Network Cost: Google Cloud charges egress bandwidth pricing (/compute/network-pricing#internet_egress) for networks using external IPs to communicate even if the traffic is within the same zone. If however, the networks are peered they can use internal IPs to communicate and save on those egress costs. Regular network pricing (/compute/network-pricing) still applies to all traffic.

For information about creating peering connections, see Using VPC Network Peering (/vpc/docs/using-vpc-peering).

Peered VPC networks exhibit the following key properties:

- VPC Network Peering works with Compute Engine (/compute/docs/), GKE (/kubernetes-engine/docs/), and App Engine flexible environment (/appengine/docs/flexible/).
- Peered VPC networks remain administratively separate. Routes, firewalls, VPNs, and other traffic management tools are administered and applied separately in each of the VPC networks.
- Each side of a peering association is set up independently. Peering will be active only when the configuration from both sides matches. Either side can choose to delete the peering association at any time.
- Peering and the option to import and export custom routes can be configured for one VPC network even before the other VPC network is created. Although route exchange only occurs after both sides have been configured.
- VPC peers always exchange all subnet routes. You can also exchange custom routes (/vpc/docs/routes#custom-routes) (static and dynamic routes), depending on whether the peering configurations have been configured to import or export them. For more information, see Importing and exporting custom routes (/vpc/docs/vpc-peering#importing-exporting-routes).
- Subnet and static routes are global. Dynamic routes can be regional (#regional) or global (#global), depending on the VPC network's dynamic routing mode.
- A given VPC network can peer with multiple VPC networks, but there is a limit (/vpc/docs/quota#vpc-peering).
- IAM permissions (/iam/docs/) for creating and deleting VPC Network Peering are included as part of the project owner, project editor, and network admin roles.
- Peering traffic (traffic flowing between peered networks) has the same latency, throughput, and availability as private traffic in the same network.
- Billing policy for peering traffic is the same as the billing policy (/compute/network-pricing) for private traffic in same network.
- If you're an organization policy administrator, you can use an organization policy to constrain which VPC networks can peer with VPC networks in your organization. You can deny peering connections to particular VPC networks or to VPC networks in a particular folder or organization. The constraint applies to new peering configurations and doesn't affect existing connections (/resource-manager/docs/organization-policy/overview#violations). An existing peering connection can continue to work even if a new policy denies new connections. For more

information, refer to the `constraints/compute.restrictVpcPeering` (/resource-manager/docs/organization-policy/org-policy-constraints) constraint.

When peering with VPC networks, consider the following restrictions:

- A subnet CIDR range in one peered VPC network cannot overlap with a static route (/vpc/docs/routes) in another peered network. This rule covers both subnet routes and static routes. Google Cloud checks for overlap in the following circumstances and generates an error when an overlap occurs.
    - When you peer VPC networks for the first time
    - When you create a static route in a peered VPC network
    - When you create a new subnet in a peered VPC network
- A dynamic route can overlap with a subnet route in a peer network. For dynamic routes, the destination ranges that overlap with a subnet route from the peer network are silently dropped. Google Cloud uses the subnet route.
- Only VPC networks (/vpc/docs/vpc) are supported for VPC Network Peering. Peering is NOT supported for legacy networks.
- You can't disable the subnet route exchange or select which subnet routes are exchanged. After peering is established, all resources within subnet IP addresses are accessible across directly peered networks. VPC Network Peering doesn't provide granular route controls to filter out which subnet CIDR ranges are reachable across peered networks. You must use firewall rules to filter traffic if that's required. The following types of endpoints and resources are reachable across any directly peered networks:
    - Virtual machine (VM) internal IPs in all subnets
    - Internal load balanced IPs in all subnets
- Only directly peered networks can communicate. Transitive peering is not supported. In other words, if VPC network N1 is peered with N2 and N3, but N2 and N3 are not directly connected, VPC network N2 cannot communicate with VPC network N3 over VPC Network Peering.
- You cannot use a tag or service account from one peered network in the other peered network.
- Compute Engine internal DNS names created in a network are not accessible to peered networks. Use the IP address to reach the VM instances in peered network.

- By default, VPC Network Peering with GKE (/kubernetes-engine/) is supported when used with IP aliases (/kubernetes-engine/docs/ip-aliases). If you don't use IP aliases, you can export custom routes so that GKE containers are reachable from peered networks.

After peering with another network and exchanging routes, your VPC network might have routes with competing destinations. To understand how Google Cloud determines the routing order, see Routing order (/vpc/docs/routes#routeselection).

By default, subnet routes are always exchanged between peered networks. You can also exchange custom routes, which include static and dynamic routes, if network administrators in both networks have the appropriate peering configurations.

When you import custom routes, your VPC network can receive custom routes from the peer network only if that network is exporting them. Similarly, if you export custom routes, the peer network can receive custom routes only if that network is importing them.

When you import or export custom routes, networks only exchange custom routes with direct peers. For example, if your VPC network is peered with multiple networks, routes that your network imports from one peered network aren't exported to the other peered networks.

When you create or modify a peering configuration, you can choose to import routes, export routes, or both. The peer network administrator must similarly configure their peering configuration before routes are exchanged. This process ensures that both network administrators explicitly agree to exchange custom routes before they are exchanged.

Sharing custom routes with peered VPC networks allow networks to learn routes directly from their peered networks. For example, if a custom route in a peered network is updated, your VPC network automatically learns and uses the updated custom route without requiring any action from you.

Exchanging custom routes can be helpful in the following scenarios:

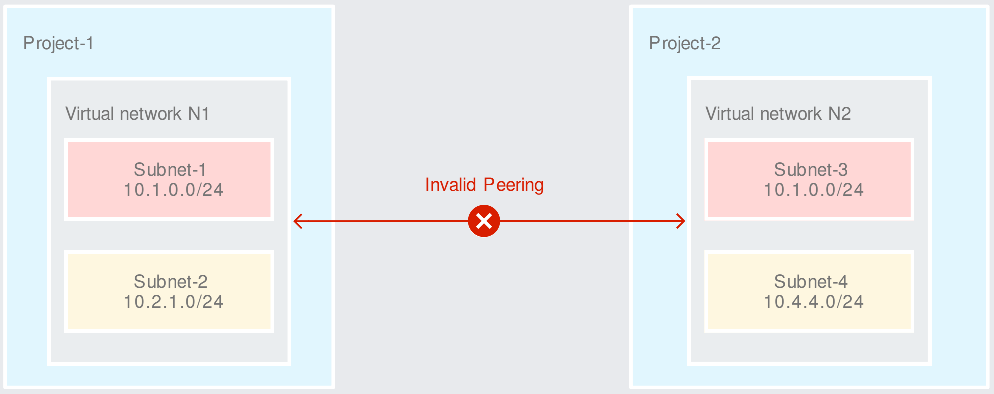
- If you have GKE clusters (/kubernetes-engine/docs/) without VPC native addressing, you might have multiple static routes to direct traffic to VM instances that are hosting your containers. You can export these static routes so that the containers are reachable from peered networks.
- If you have a VPN tunnel (/vpn/docs) or interconnect (/interconnect/docs), you can share custom routes so that peer networks can reach your on-premises network. For dynamic routes, you must add Cloud Router custom route advertisements (/router/docs/concepts/overview#route-advertisement) in your VPC network to announce peered network subnets to your on-premises network.

When you configure importing or exporting custom routes, consider the following behaviors:

- Both static and dynamic routes are exported or imported. You can't choose to import or export only one type of route.
- Custom routes imported from one VPC network can't be exported to another peered VPC network transitively.
- The following routes are excluded from being imported and exported:
    - Tagged routes are never exchanged between peer networks. Tagged routes are custom static routes scoped to specific VM instances by using network tags. Network tags can only be resolved in the VPC network in which they're created.
    - Static routes with a next hop to the default Internet gateway are never exchanged. For example, the default route (/vpc/docs/routes#types_of_routes) (0.0.0.0/0) with a next hop of default Internet gateway isn't exchanged between peer networks.
- Imported routes could lead to unintended changes to traffic flow, such as changes to the routing order. For more information, see Routing order (/vpc/docs/routes#routeselection).
- When a VPC network imports custom routes from a peer network, the destination ranges are imported as-is. However, the next hop for an imported routes is set to the name of the peering connection. Traffic is routed to the peered network where the actual next hop is defined.

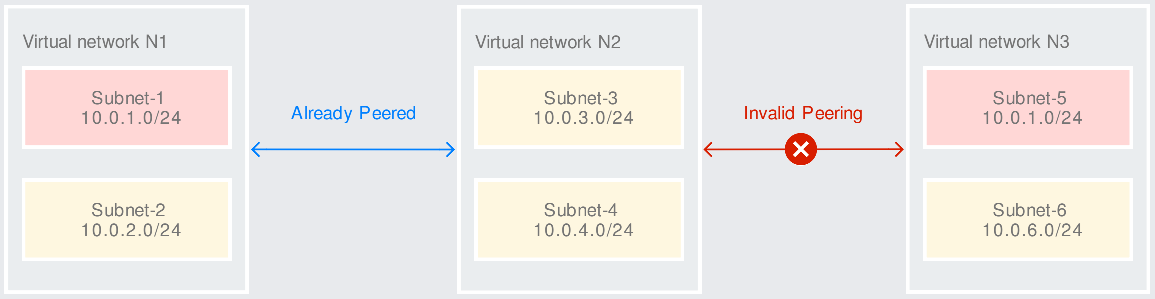
The following sections demonstrate how VPC Network Peering behaves in certain scenarios.

At the time of peering, Google Cloud checks to see if there are any subnets with overlapping IP ranges between the two VPC networks or any of their peered networks. If there is an overlap, peering is not established. Since a full mesh connectivity is created between VM instances, subnets in the peered VPC networks can't have overlapping IP ranges as this would cause routing issues.



(/vpc/images/peering/network-peering-11.svg)
Overlapping subnet IP ranges between two peers (click to enlarge)

If there were any subnets with overlapping IP ranges between peers of a given VPC network, it would cause a routing conflict. For example, suppose VPC network N1 has already peered with VPC network N2, then VPC network N3 tries to peer with N2. This is an invalid peering because N3 has a subnet Subnet_5 whose IP range overlaps with Subnet_1 in network N1.

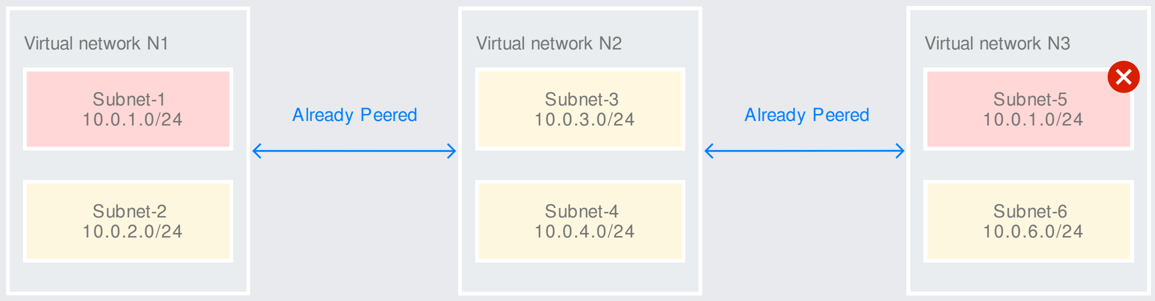


(/vpc/images/peering/network-peering-12.svg)
Overlapping subnet IP ranges with three peers (click to enlarge)

When a VPC subnet is created or a subnet IP range is expanded, Google Cloud performs a check to make sure the new subnet range does not overlap with IP ranges of subnets in the same VPC network or in directly peered VPC networks. If it does, the creation or expansion action fails. For example, when a new subnet subnet_3 is created in network N2 in the following figure, the IP ranges must not overlap with the IP ranges defined in the directly peered network N1.

Subnet creation check (click to enlarge)
(/vpc/images/peering/network-peering-13.svg)
Subnet creation check (click to enlarge)

Google Cloud also ensures that no overlapping subnet IP ranges are allowed across VPC networks that have a peered network in common. If it does, the creation or expansion action fails. For example, when a new subnet subnet_5 is created in network N3 in the following figure, the IP ranges must not overlap with the IP ranges defined in directly peered network N2, or with network N1, because N1 is already peered with N2.



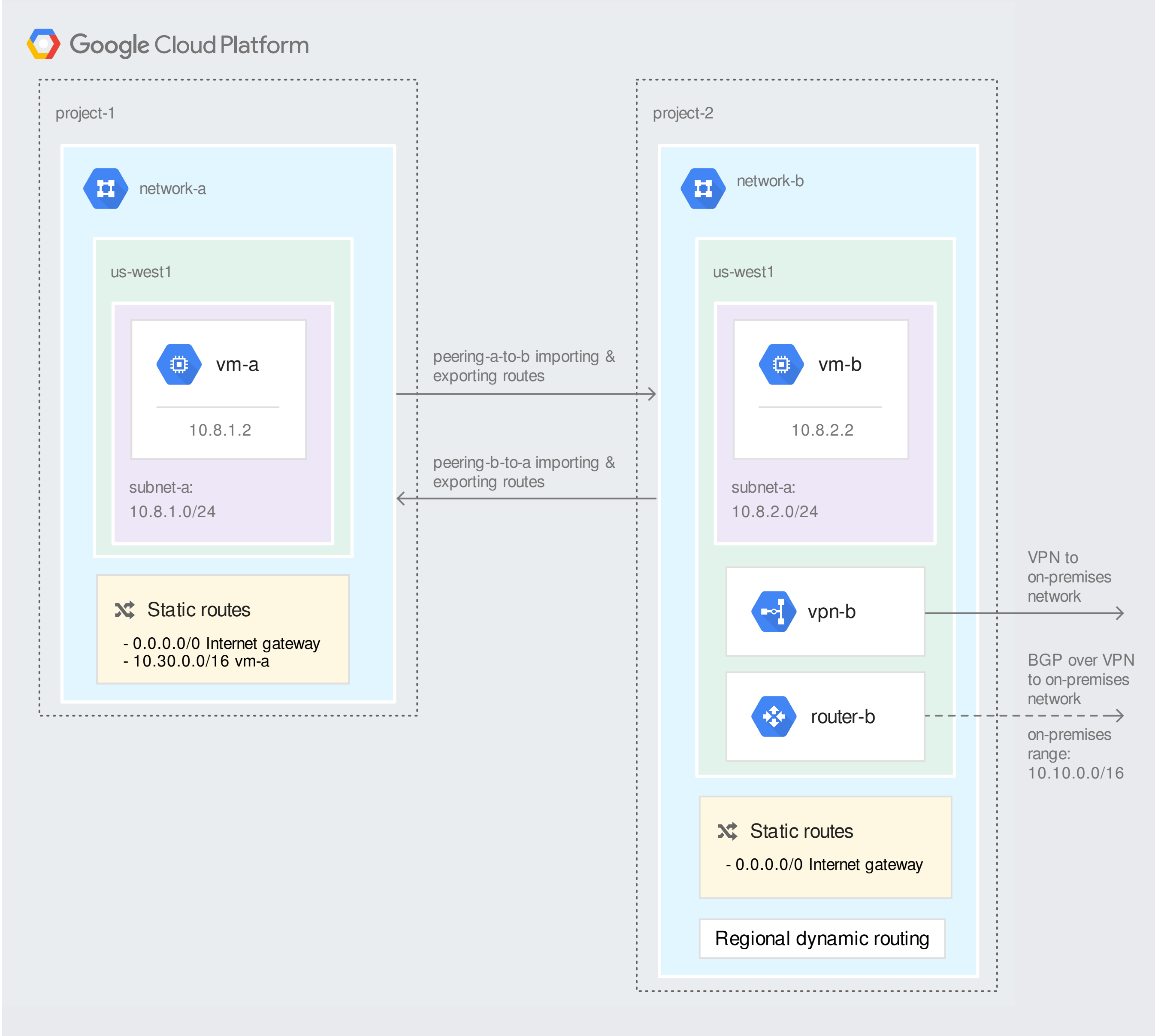
(/vpc/images/peering/network-peering-14.svg)
Subnet creation check with three peers (click to enlarge)

You can use either Cloud VPN or Cloud Interconnect to securely connect your on-premises network to your VPC network. If you export custom routes, peered VPC networks can also connect to your on-premises network. On the on-premises side, you must create routes so that traffic going to VPC networks is directed to the VPN tunnel.

Cloud VPN supports static and dynamic routing, but Cloud Interconnect supports dynamic routing only. For dynamic routing, use Cloud Router to dynamically update routes between your VPC network and your on-premises network by using the Border Gateway Protocol (BGP). Cloud Routers can learn and propagate routes within its region or for all regions within a VPC network. This behavior depends on the VPC network's dynamic routing mode, which can be regional or global.

The following use cases shows how resources in peered VPC networks are accessible after they've imported and exported custom routes. Each example shows two networks (`network-a` and `network-b`) that are peered to one another. In one example, the dynamic routing mode for `network-b` is regional, and in the other example it's global.

If you use regional dynamic routing, only resources in the same region as the Cloud Router can access the on-premises network. This restriction applies to the Cloud Router's VPC network and any peered VPC networks, regardless of the peered VPC network's dynamic routing mode. In the following example, `network-b` contains a VPN tunnel (which could also be an interconnect) and a Cloud Router. The dynamic routing mode of `network-b` is set to regional. In the peered network, `subnet-a` is in the same region as the Cloud Router in `network-b`.


Google Cloud Platform
project-1
network-a
us-west1
vm-a
10.8.1.2
subnet-a:
10.8.1.0/24
Static routes
- 0.0.0.0/0 Internet gateway
- 10.30.0.0/16 vm-a
peering-a-to-b importing & exporting routes
peering-b-to-a importing & exporting routes
project-2
network-b
us-west1
vm-b
10.8.2.2
subnet-a:
10.8.2.0/24
vpn-b
router-b
VPN to on-premises network
BGP over VPN to on-premises network
on-premises range: 10.10.0.0/16
Static routes
- 0.0.0.0/0 Internet gateway
Regional dynamic routing

(/vpc/images/peering/network-peering-routes-regional.svg)

Peered network with access to on-premises network

After a peering connection is established, `network-a` can access the VPN tunnel in `network-b`. This access is limited to subnets that are in the same region as the Cloud Router. For example, the VM instance `vm-a` can reach the VPN tunnel because it's in the same region as the Cloud Router. VM instances in other regions can't reach the tunnel.

Cloud Router doesn't learn routes and propagate routes from peered VPC networks. As a result, you must have a custom route advertisement on the Cloud Router that propagates the `10.8.1.0/24` range to the on-premises network on the BGP session.

The following table summarizes the resulting routes for `network-a` and `network-b` if they both import and export custom routes:

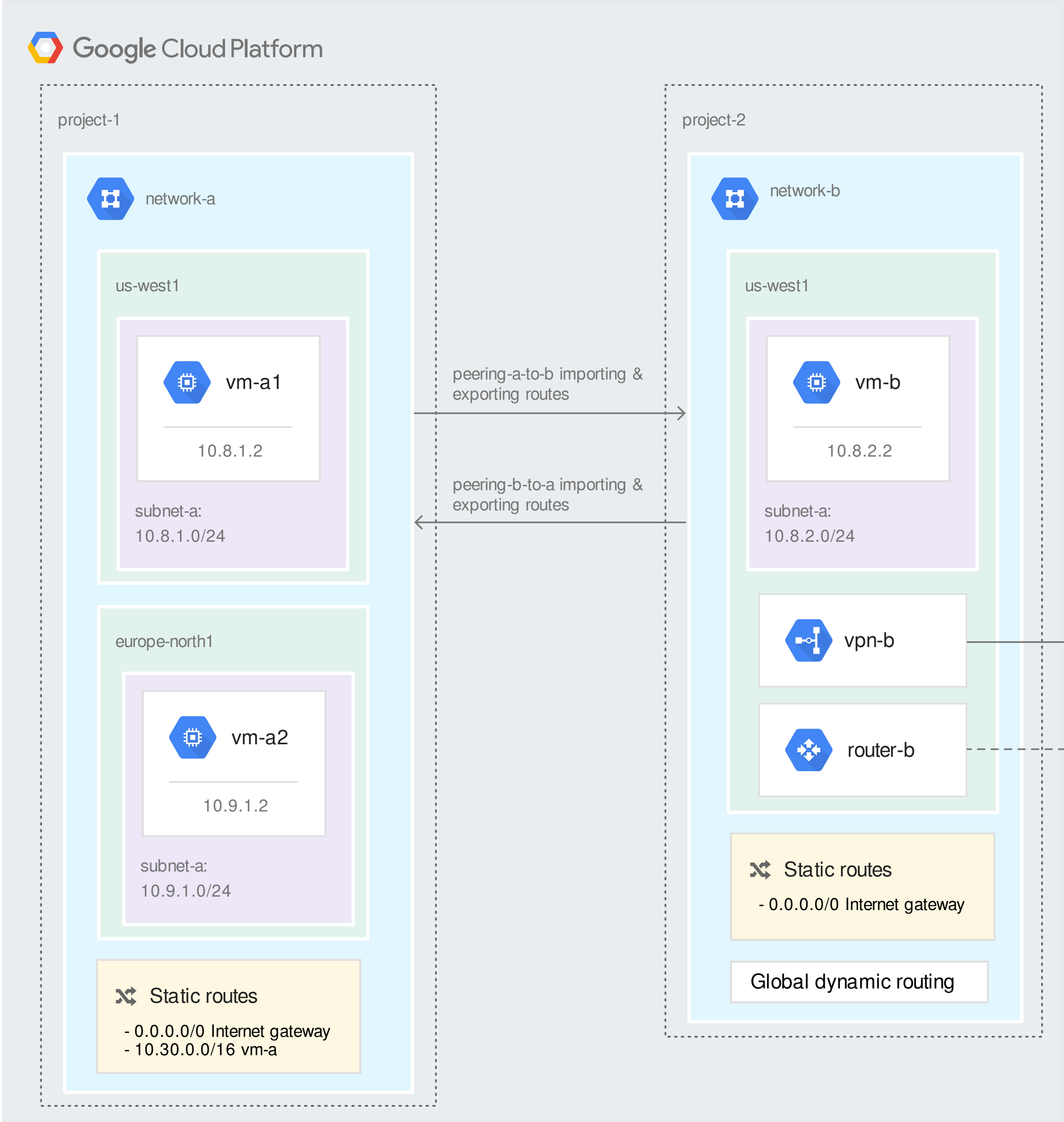
Network	Destination	Next Hop	Origin
network-a	0.0.0.0/0	Internet gateway	local
	10.8.1.0/24	network-a	local
	10.30.0.0/16	vm-a1	local
	10.8.2.0/24	peering-a-to-b	peer
	10.10.0.0/16 (dynamic route limited to `us-west1`)	peering-a-to-b	peer
network-b	0.0.0.0/0	Internet gateway	local
	10.8.2.0/24	network-b	local
	10.10.0.0/16 (dynamic route limited to `us-west1`)	vpn-b	local
	10.8.1.0/24	peering-b-to-a	peer
	10.30.0.0/16	peering-b-to-a	peer

If `network-b` enables global dynamic routing, resources in all regions can access the on-premises network. This applies to the Cloud Router's VPC network and any peered VPC networks, regardless of the peered VPC network's dynamic routing mode.

In the following example, resources in `network-a` can access the VPN tunnel in `network-b`, regardless of their region. For example, the VM instances `vm-a1` and `vm-a2` can reach the on-premises network even through `vm-a2` is in a different region than the VPN tunnel.

Cloud Router must also include custom route advertisement to announce the `10.8.1.0/24` and `10.9.1.0/24` ranges to the on-premises network on the BGP session.

Global dynamic routing mode doesn't change the regional restriction of internal load balancers. For more informatior
ɔud Load Balancing documentation (/load-balancing/docs/internal/internal-lb-and-other-networks).



(/vpc/images/peering/network-peering-routes-global.svg)

Peered network with access to on-premises network with global dynamic routing

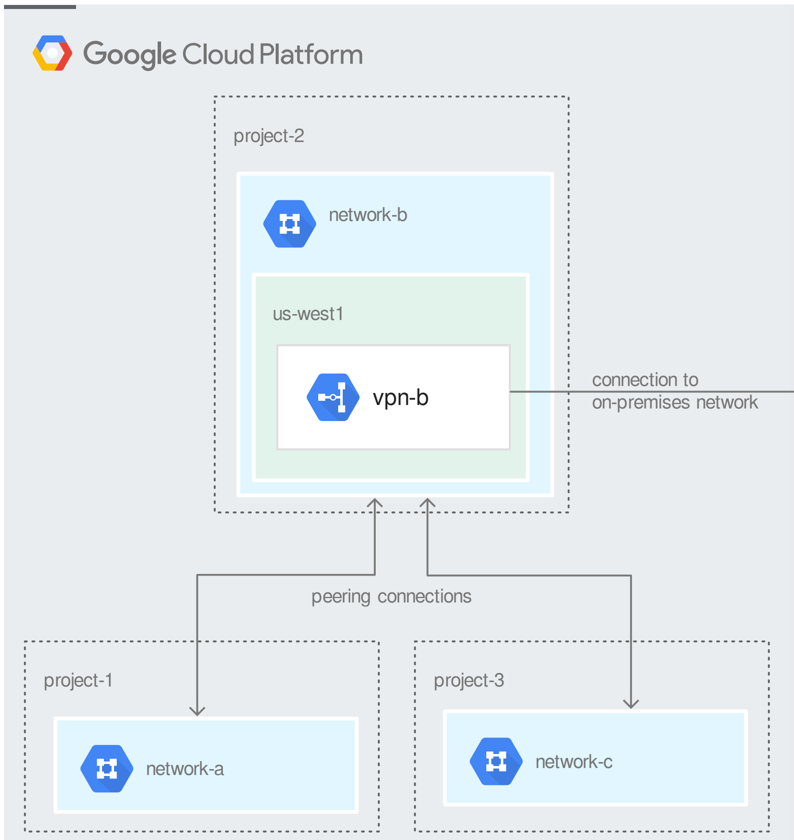
The following table summarizes the resulting routes for `network-a` and `network-b` if they both import and export custom routes:

Network	Destination	Next Hop	Origin
network-a	0.0.0.0/0	Internet gateway	local
	10.8.1.0/24	network-a	local
	10.9.1.0/24	network-a	local
	10.30.0.0/16	vm-a1	local
	10.8.2.0/24	peering-a-to-b	peer
	10.10.0.0/16 (global dynamic route)	peering-a-to-b	peer
network-b	0.0.0.0/0	Internet gateway	local
	10.8.2.0/24	network-b	local
	10.10.0.0/16 (global dynamic route)	vpn-b	local
	10.8.1.0/24	peering-b-to-a	peer
	10.9.1.0/24	peering-b-to-a	peer
	10.30.0.0/16	peering-b-to-a	peer

Imagine that you have a single on-premises connection, such as a VPN tunnel or interconnect, between your VPC network and on-premises network. You want to share this connection with other VPC networks so that you don't have to recreate an on-premises connection for all of the other VPC networks.

You can have other VPC networks peer with your VPC network (also known as a transit network) so that the other networks can use the on-premises connection. All peered networks can leverage the on-premises connection, but they won't be able to route traffic to another peer through the transit network.

In the following example, there are three VPC networks. `network-b` is peered with `network-a` and `network-c`. All networks are exporting and importing custom routes. `network-b` acts as the transit network, where the VPN tunnel is located.



(/vpc/images/peering/network-peering-vpc-transit.svg)

VPC transit network

- `network-b` has the relevant static VPN routes to route traffic to the connected on-premises network. The static route is exported and then imported by the peered VPC networks. If you're using Cloud Router for dynamic routing, you must advertise the subnet IP addresses of the peer networks. Cloud Router doesn't automatically advertise routes from VPC Network Peering.
- Hosts in the on-premises network can send and receive traffic to and from hosts in each of the VPC networks. The on-premises network must contain routes that have a next hop to the VPN gateway if traffic is destined to a VPC network.
- Hosts in `network-c` can reach hosts in `network-b` and the on-premises network but can't reach hosts in `network-a`. Similarly, hosts in `network-a` can reach `network-b` and the on-premises network but not `network-c`. This is because peering routes are not transitive.

VM instances in peer networks can access the internal IP addresses of internal TCP/UDP load balancers in your VPC network if the following conditions are met:

- The VMs are located in the same region as the internal TCP/UDP load balancer.

- Ingress firewall rules that apply to the load balancer's backend VMs allow traffic from sources in peer networks. These ingress firewall rules must be created in the VPC network that contains the load balancer.

Refer to using VPC Network Peering (/load-balancing/docs/internal/internal-lb-and-other-networks#using_network_peering) for additional information.

When you connect networks using VPC Network Peering, firewall rules are **not** exchanged between them. To allow ingress traffic from VM instances in a peer network, you must create ingress allow firewall rules. By default, ingress traffic to VMs is blocked by the implied deny ingress rule (/vpc/docs/firewalls#default_firewall_rules).

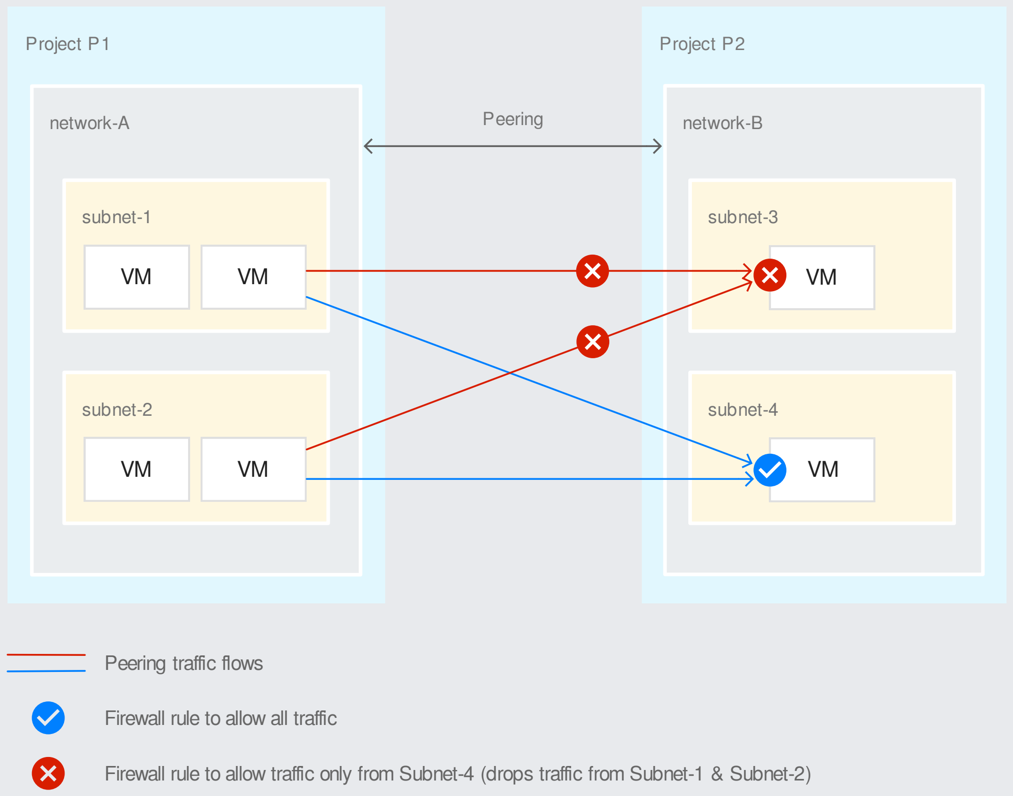
If you need to restrict access to VMs such that only other VMs in your VPC network have access, ensure that the sources (/vpc/docs/firewalls#sources_or_destinations_for_the_rule) for your ingress allow firewall rules only identify VMs in your VPC network, not ones from peer networks. For example, you can specify source IP ranges for just the subnets in your VPC network.

To restrict access to an internal TCP/UDP load balancer, create ingress firewall rules that apply to the load balancer's backend VMs.

Firewall rules are not imported into peered networks. You can configure firewall rules in each network separately to control the traffic you want to allow or block from peered networks.

If you have peering between your VPC network and another VPC network, you may want to block traffic to a given set of VM instances or Internal Load Balancing endpoints. You must use firewall rules to do this as there is no way to exclude certain VM instances or Internal load balancers from the peering. If you want to disallow communication with certain VM instances or Internal load balancers, you can install ingress firewall rules on the network you want to block the communication to.

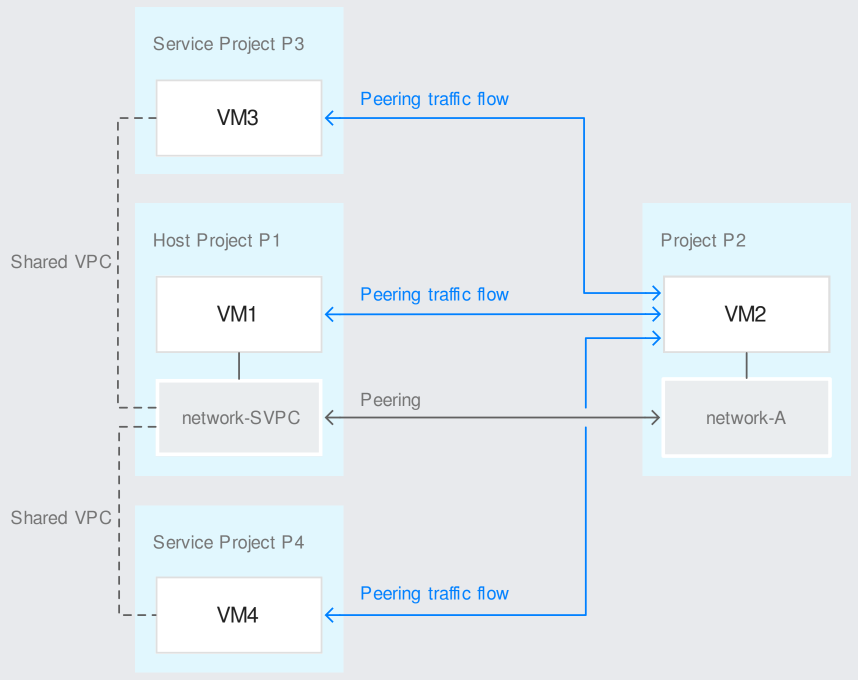
- VM instances: In this case, you can install an ingress firewall that only allows traffic from certain source IPs. These source IPs can map to the subnet CIDRs in your own VPC network. This blocks any traffic originating from the peered VPC networks.


Project P1
Project P2
network-A
network-B
Peering
subnet-1
subnet-2
subnet-3
subnet-4
VM
Peering traffic flows
Firewall rule to allow all traffic
Firewall rule to allow traffic only from Subnet-4 (drops traffic from Subnet-1 & Subnet-2)

(/vpc/images/peering/network-peering-07.svg)
Firewall with VPC Network Peering (click to enlarge)

- Internal load balancers: In this case, you can install ingress firewall rules in the VPC network with the Internal load balancer. These source IPs can map to all or part of the subnet CIDRs in your own network. If ingress firewall rules are put in place for all subnet CIDR ranges in the peered network, then no instance in that network would be able to reach the Internal load balancer backend VMs.

VPC Network Peering allows peering with a Shared VPC (/vpc/docs/shared-vpc). A Shared VPC host project is a project that allows other projects to use one of its networks. The following diagram shows this setup.



(/vpc/images/peering/network-peering-10.svg)
Shared VPC with network peering (click to enlarge)

Network-SVPC is in a Shared VPC network in host project P1. Service projects P3 and P4 are able to attach VM instances to Network-SVPC. Network-SVPC peers with Network-A. As a result:

- VM instances in Shared VPC service projects that are using the Network-SVPC (VM3 and VM4) have private, internal IP connectivity with any endpoints associated to Network-A.
- VM instances associated to network-A will have private, internal IP connectivity with any endpoints associated to Network-SVPC, regardless of whether those endpoints live in the host project or in a service project.

It is possible to set up VPC Network Peering between two Shared VPC networks.

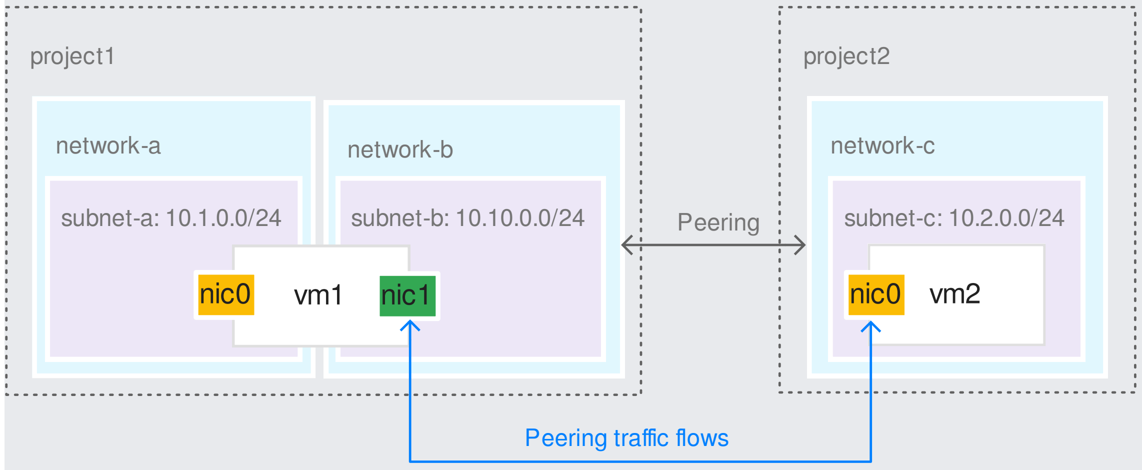
A VM instance can have multiple network interfaces (/vpc/docs/multiple-interfaces-concepts), one per VPC network. For those instances, Google Cloud assigns destination-based IP routes, where each interface gets a route from the subnet that it is in. Additionally, Google Cloud provides a default route to the primary network interface. With destination-based routing, any traffic that's not destined to any of the instance's subnets egresses from the primary network interface. For more information, see

DHCP behavior with multiple network interfaces (/vpc/docs/multiple-interfaces-concepts#dhcp_behavior_with_multiple_network_interfaces).

When you have peer networks that include VM instances with multiple network interfaces, you might need to change the default destination-based routing policy to a source-based routing policy. For more information, see Configuring policy routing (/vpc/docs/create-use-multiple-interfaces#configuring_policy_routing).

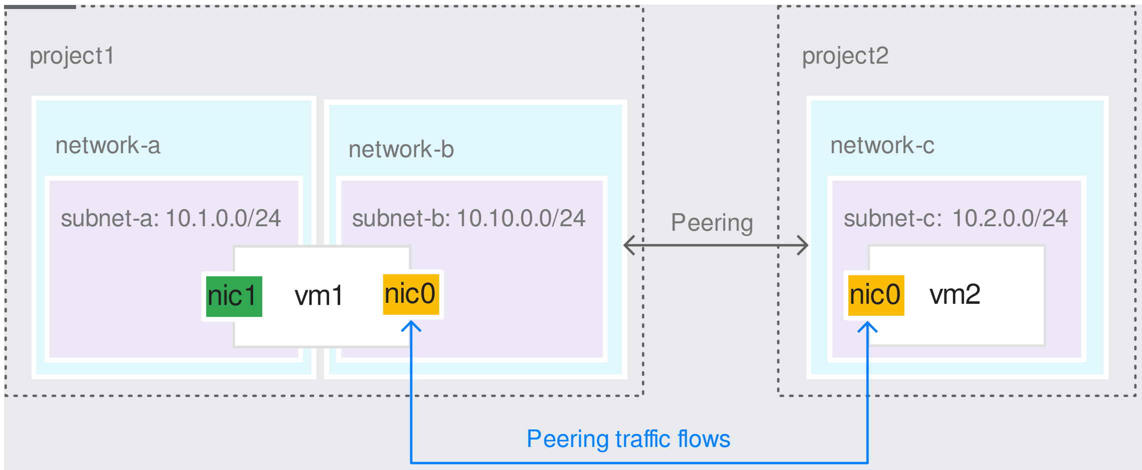
The following scenarios demonstrate when a VM instance might or might not require a source-based routing policy

In the following example, `vm1` requires a source-based routing policy so that `vm1` and `vm2` can successfully communicate. Without a routing policy, all traffic egresses through `vm1-nic0` to `network-a` unless traffic is destined to `subnet-b` where `nic1` is located. This means that traffic from `vm1` destined to `network-c` is dropped or sent to the incorrect destination because the VM instance always sends traffic out of its primary interface.



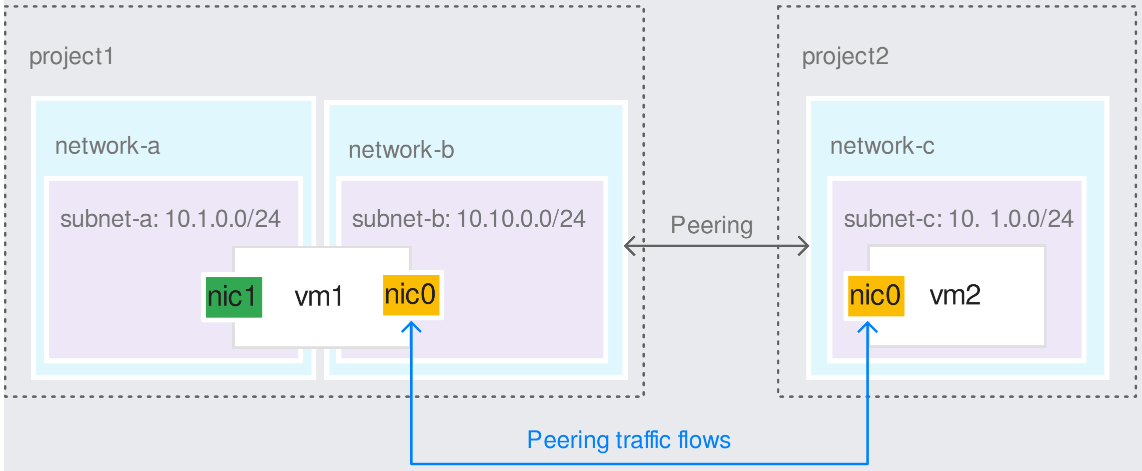
(/vpc/images/peering/network-peering-multinic1.svg)
Multiple network interfaces with network peering (click to enlarge)

In the following example, the primary network interface of `vm1` is in a network that is peered with another network. You don't need to configure policy routing on `vm1`.



(/vpc/images/peering/network-peering-multinic2.svg)
Primary network interfaces in peered networks (click to enlarge)

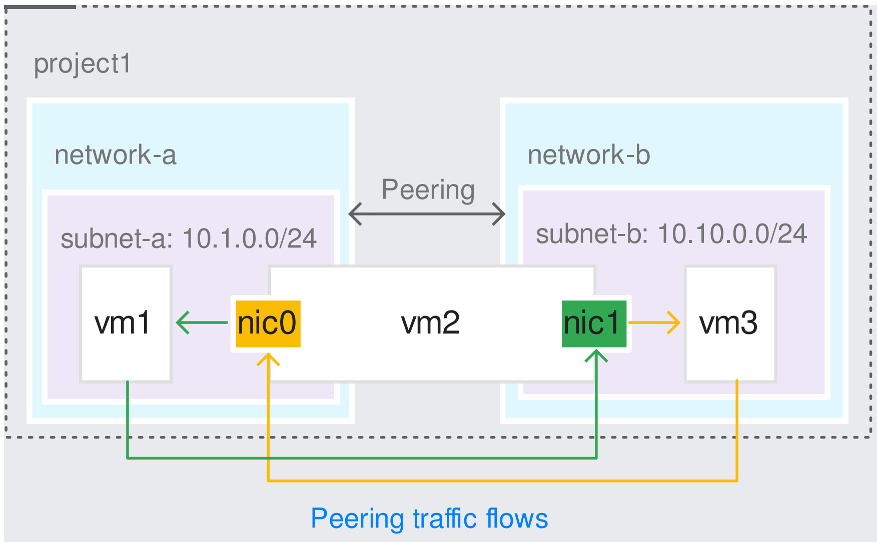
In the following example, `vm1-nic1` and `vm2-nic0` are in overlapping subnets. When peering is established, Google Cloud checks for overlapping IP ranges only between peers and not between other networks where instances contain network interfaces. To ensures that communication between `vm1` and `vm2` works, you must add a source-based routing policy on `vm1-nic0`.



(/vpc/images/peering/network-peering-multinic2-overlap.svg)
Secondary network interface is in a subnet that overlaps with the peered network (click to enlarge)

The following example shows a VM instance with multiple network interfaces, where each one is in a network that's peered with each other. In this case, `vm1` doesn't require a source-based routing policy. Traffic leaving the VM instance to each subnet uses the corresponding network interface.

If `vm1` sends traffic to the IP address of `vm2-nic1`, traffic goes into `nic1` but egresses out of `nic0`. Similarly, if `vm3` sends traffic to the IP address of `vm2-nic0`, traffic goes into `nic0` but egresses out of `nic1`.



(/vpc/images/peering/network-peering-multinic3.svg)
A VM instance with multiple network interfaces in peered networks (click to enlarge)

A subnet has a single primary IP address range and, optionally, one or more secondary IP address ranges (/vpc/docs/alias-ip#subnet_primary_and_secondary_cidr_ranges). For each subnet IP address range, Google Cloud creates a subnet route (/vpc/docs/routes#subnet-routes). When you use VPC Network Peering, Google Cloud always exchanges the subnet routes between the two peered networks. If firewall rules in each network permit communication, VM instances in one network can communicate with instances in the peered network.

When you establish a peering relationship, Google Cloud checks that a subnet's primary and secondary ranges don't overlap with other ranges in peered networks.

- To set up and troubleshoot VPC Network Peering, see Using VPC Network Peering (/vpc/docs/using-vpc-peering).