oduct or feature is in a pre-release state and might change or have limited support. For more information, see the pr
stages (/products/#product-launch-stages).

This page is an overview of Packet Mirroring. If you're looking for information about how to create and manage pack
ng policies, see Using Packet Mirroring (/vpc/docs/using-packet-mirroring).

Packet Mirroring clones the traffic of specified instances in your Virtual Private Cloud (VPC) network and forwards it for examination. Packet Mirroring captures all ingress and egress traffic and packet data, such as payloads and headers. The mirroring happens on the virtual machine (VM) instances, not on the network. Consequently, Packet Mirroring consumes additional bandwidth on the hosts.

Packet Mirroring is useful when you need to monitor and analyze your security status. It exports all traffic, not only the traffic between sampling periods. For example, you can use security software that analyzes mirrored traffic to detect all threats or anomalies. Additionally, you can inspect the full traffic flow to detect application performance issues. For more information, see the example use cases (/vpc/docs/packet-mirroring#use-cases).

Packet Mirroring copies traffic from *mirrored sources* and sends it to a *collector destination*. To configure Packet Mirroring, you create a *packet mirroring policy* that specifies the source and destination.

- Mirrored sources are Compute Engine instances that you can select by specifying subnets, network tags, or instance names. If you specify a subnet, all existing and future instances in that subnet are mirrored. You can specify one or more source types; if an instance matches at least one of them, it's mirrored.

  Packet Mirroring collects traffic from an instance's network interface in the network where the packet mirroring policy applies. In cases where an instance has multiple network interfaces, the other interfaces aren't mirrored unless another policy has been configured to do so.

- A collector destination is an instance group that is behind an internal load balancer. Instances in the instance group are referred to as collector instances.

When you specify the collector destination, you enter the name of a forwarding rule that is associated with the internal load balancer. Google Cloud then forwards the mirrored traffic to the collector instances. An internal load balancer for Packet Mirroring is similar to other internal load balancers except that the forwarding rule must be configured for Packet Mirroring. Any non-mirrored traffic that is sent to the load balancer is dropped.

By default, Packet Mirroring collects all incoming and outgoing traffic of mirrored instances. Instead of collecting all traffic, you can use filters to narrow the traffic that is mirrored. Using filters can help you limit bandwidth usage on mirrored instances.

You can configure filters to collect traffic based on protocol, IP address ranges, or both. You can use filters to specify only the mirrored traffic to collect, not the traffic to exclude.

Multiple packet mirroring policies can apply to an instance. Depending on each policy's filter and priority, Google Cloud chooses one for each flow. If you have distinct policies, Google Cloud uses the corresponding policy that matches the mirrored traffic. For example, you might have one policy that has the filter `1.1.1.1/24:TCP`, and another policy that has the filter `2.2.2.2/24:UDP`. Because the policies are distinct, there's no ambiguity about which policy Google Cloud uses.

However, if you have overlapping policies, Google Cloud evaluates their filters and priorities to choose one. For example, you might have two policies, one that has a filter for `10.0.0.0/24:TCP` and another for `10.0.0.0/16:TCP`. These policies overlap because their CIDR ranges overlap.

When choosing a policy, Google Cloud prioritizes policies by comparing their filter's CIDR range size. If overlapping policies have the same exact filter, Google Cloud compares their priority values. The following list details the evaluation that Google Cloud performs when multiple policies overlap:

- If policies have the exact same filters (CIDR range and protocol), Google Cloud uses the policy with the lowest priority value. For example, if you have multiple policies that use the default filter, meaning they match on `0.0.0.0/0` for all protocols, Google Cloud uses the policy with the lowest priority value. This is the only case when Google Cloud uses the priority value.

  If policies have equal priority, Google Cloud picks one by using a non-deterministic method. Each time matching traffic is re-evaluated against these policies, Google Cloud might pick the same policy or a different one.

- In the following cases, Google Cloud chooses a policy based on its filter. The priority values for each policy are ignored.
    - If policies have different but overlapping CIDR ranges and the same exact protocols, Google Cloud selects the policy that uses the most specific IP address range. Suppose the destination for a TCP packet leaving a mirrored instance is `10.240.1.4` and there are two policies with the following filters: `10.240.1.0/24:ALL` and `10.240.0.0/16:TCP`. Because the most specific match for `10.240.1.4` is `10.240.1.0/24:ALL`, Google Cloud uses the policy that has the filter `10.240.1.0/24:ALL`.
    - If policies have the same exact CIDR range but distinct protocols, these policies don't overlap. Google Cloud uses the policy that corresponds to the mirrored traffic's protocol. For example, you might have a policy for `10.240.1.0/24:TCP` and another for `10.240.1.0/24:UDP`. Depending on the mirrored traffic's protocol, Google Cloud uses either the TCP or UDP policy.
    - If policies specify the same exact CIDR range with overlapping protocols, Google Cloud picks a policy by using a non-deterministic method. For example, the following filters have the same range but overlapping protocols: `10.240.1.0/24:TCP` and `10.240.1.0/24:ALL`.

In cases where the selected policy is not deterministic, it's possible that Google Cloud captures mirrored traffic across multiple load balancers. To predictably and consistently send mirrored traffic to a single load balancer, create policies that have filters with non-overlapping address ranges. If ranges overlap, set unique filter protocols. If policies have matching filters or use the default filter, set unique priority numbers for those policies.

To collect mirrored traffic, Packet Mirroring requires you to use an internal load balancer as part of the collector destination. If you have multiple collector instances that are behind the load balancer, Google Cloud might send ingress and egress flows to different collector instances. However, for security tools to effectively analyze data and detect patterns, they require all packets for a single communication stream (ingress and egress flows).

To collect a stream on a single collector instance, use an unmanaged or managed instance group that always has a single collector instance.

If mirrored instances are in a subnet that also has VPC Flow Logs enabled, VPC Flow Logs doesn't report the cloned packets. VPC Flow Logs only logs non-mirrored traffic.

However, if collector instances are in a subnet that has VPC Flow Logs enabled, VPC Flow Logs captures the flow between the mirrored and collector instances. The logs show the source and destination IP addresses as the mirrored and collector instances. To stop collecting logs on mirrored traffic, disable VPC Flow Logs on the collector instances subnet.

For more information, see Using VPC Flow Logs (/vpc/docs/using-flow-logs).

The following list describes constraints or behaviors with Packet Mirroring that are important to understand before you use it:

- You can mirror only TCP, UDP, and ICMP traffic.
- The mirrored sources and collector destination must be in the same region. They can be in different VPC networks if they're connected through VPC Network Peering.
- You cannot mirror and collect traffic on the same network interface of a VM instance because doing this would cause a mirroring loop.
- Mirroring traffic consumes bandwidth on the mirrored instance. For example, if a mirrored instance experiences 1 Gbps of ingress traffic and 1 Gbps of egress traffic, the total traffic on the instances is 1 Gbps of ingress and 3 Gbps of egress (1 Gbps of normal egress traffic and 2 Gbps of mirrored egress traffic). To limit what traffic is collected, you can use filters.
- The cost of Packet Mirroring varies depending on the amount of egress traffic traveling from a mirrored instance to an instance group and whether the traffic travels between zones.
- Packet Mirroring applies to both ingress and egress direction. If two VM instances that are being mirrored send traffic to each other, Google Cloud collects two versions of the same packet.
- There is a maximum number of packet mirroring policies that you can create for a project. For more information, see the per-project quotas on the quotas (/vpc/docs/quota) page.
- For each packet mirroring policy, the maximum number of mirrored sources that you can specify depends on the source type:
    - 5 subnets
    - 5 tags
    - 50 instances

- The maximum number of packet mirroring filters is 30, which is the number of IP address ranges multiplied by the number of protocols. For example, you can specify 30 ranges and 1 protocol, which would be 30 filters. However, you cannot specify 30 ranges and 2 protocols, which would be 60 filters and greater than the maximum.
- For the **Beta** release, there is no charge for using Packet Mirroring. Charges start when Packet Mirroring is **Generally Available** (GA). You are charged for each packet mirroring forwarding rule and the amount of data processed, similar to existing forwarding rules and load balancers. For details, see the Load balancing and forwarding rules (/compute/network-pricing#lb) pricing.

★ **Note:** For beta, you are still charged for all the prerequisite components and egress traffic that are related to Packet Mirroring. For example, the instances that collect traffic are charged at the regular rate. Also, if packet mirroring traffic travels between zones, you are charged for the egress traffic. For pricing details, see the related pricing page (/compute/all-pricing).

The following sections describe real-world scenarios that demonstrate why you might use Packet Mirroring.

Security and network engineering teams must ensure that they are catching all anomalies and threats that might indicate security breaches and intrusions. They mirror all traffic so that they can complete a comprehensive inspection of suspicious flows. Because attacks can span multiple packets, security teams must be able to get all packets for each flow.

For example, the following security tools require you to capture multiple packets:

- Intrusion detection system (IDS (https://wikipedia.org/wiki/Intrusion_detection_system)) tools require multiple packets of a single flow to match a signature so that the tools can detect persistent threats.
- Deep Packet Inspection engines inspect packet payloads to detect protocol anomalies.
- Network forensics for PCI compliance and other regulatory use cases. Packet Mirroring provides a solution for capturing different attack vectors, such as infrequent communication or attempted but unsuccessful communication.

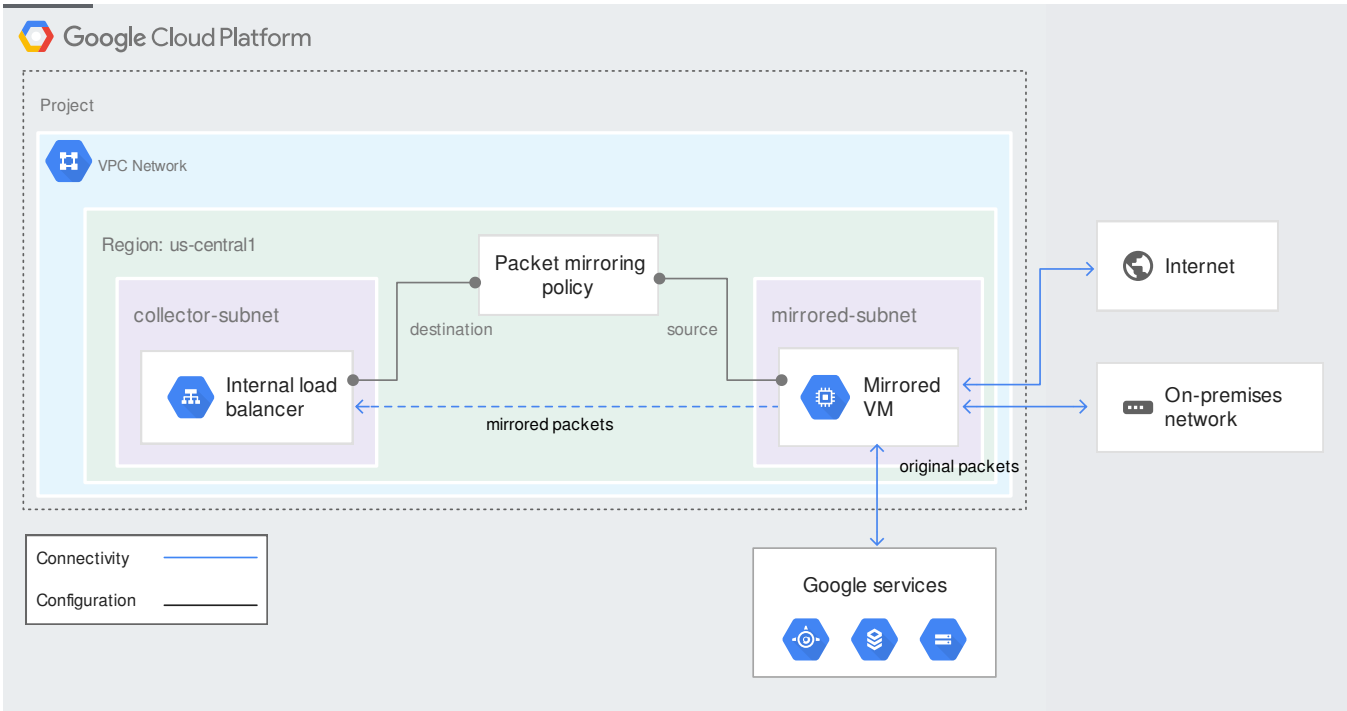
Network engineers can use mirrored traffic to troubleshoot performance issues reported by application and database teams. To check for networking issues, network engineers can view what's going over the wire rather than relying on application logs.

For example, network engineers can use data from Packet Mirroring to complete the following tasks:

- Analyze protocols and behaviors so that they can find and fix issues, such as packet loss or TCP resets.
- Analyze (in real time) traffic patterns from remote desktop, VoIP, and other interactive applications. Network engineers can search for issues that affect the application's user experience, such as multiple packet resends or more than expected reconnections.

You can use Packet Mirroring in various setups. The following examples show the location of collector destinations and their policies for different packet mirroring configurations, such as VPC Network Peering and Shared VPC.

The following example shows a packet mirroring configuration where the mirrored source and collector destination are in the same VPC network.



(/vpc/images/packet-mirroring/same-network.svg)

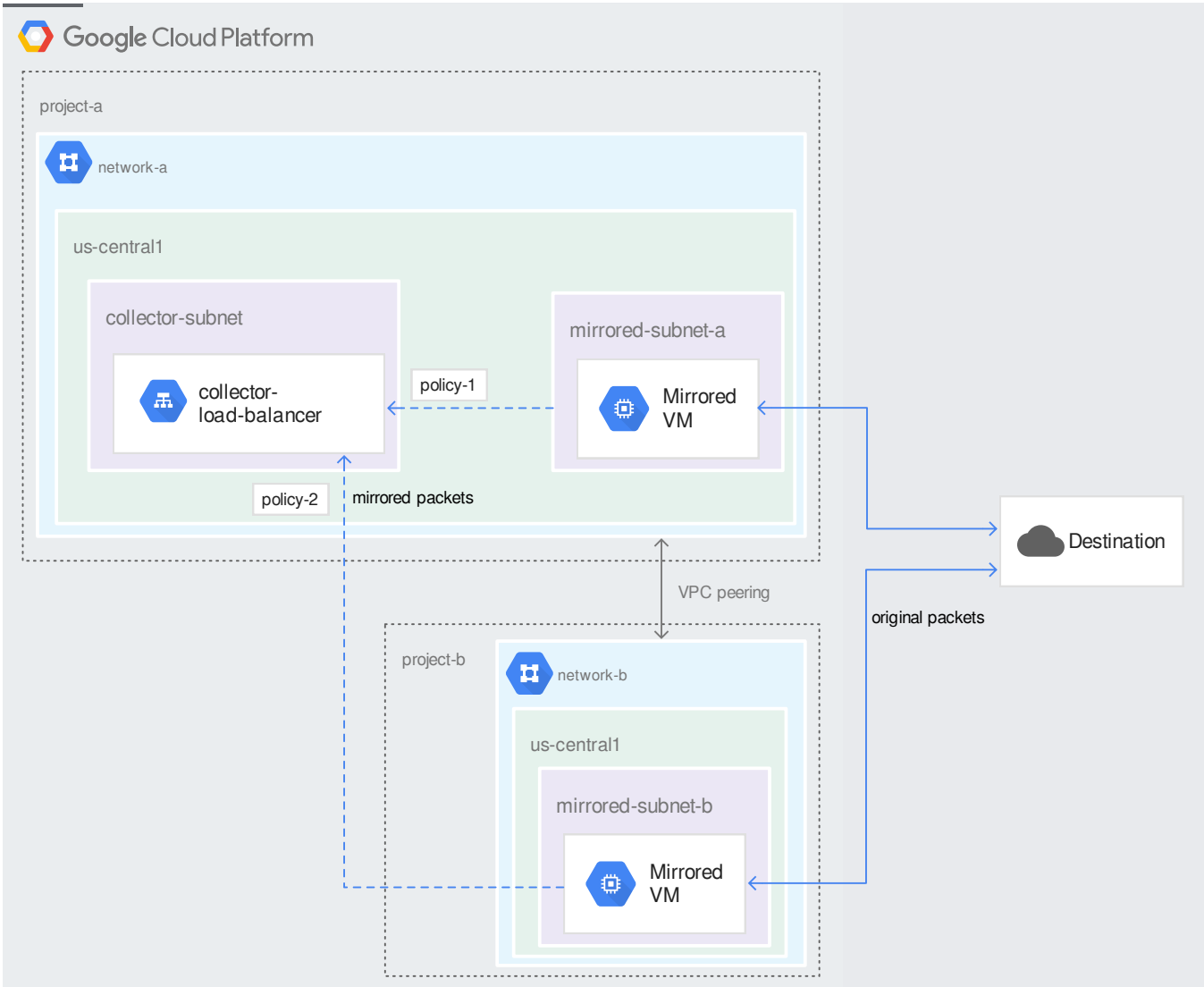
Packet mirroring policy that has all resources in the same VPC network (click to enlarge)

In the preceding diagram, the packet mirroring policy is configured to mirror `subnet-a` and send mirrored traffic to the internal load balancer. Google Cloud mirrors the traffic on existing and future instances in the subnet. All traffic to and from the internet, on-premises hosts, or Google services is mirrored.

You can build a centralized collector model, where instances in different VPC networks send mirrored traffic to a central VPC network. That way, you don't need to build a destination collector in each network. The central VPC network hosts one or more collector destinations where all mirrored traffic is sent.

You can achieve this scenario by using VPC Network Peering. All VPC networks that include mirrored sources must be peered with the central network.

VPC networks can be in the same project or in different projects. For each peer VPC network, you must create a packet mirroring policy. In this case, the policy is in the project that contains the central network. The policy can also be in a project that has a network that is peered with the central network.



(/vpc/images/packet-mirroring/peer-network.svg)

Packet mirroring policies in a central network peered with other networks that have mirrored sources (click to enlarge)

In the preceding diagram, the collector destination collects mirrored traffic from subnets in two different networks. All resources (the source and destination) must be in the same region. The setup in `network-a` is similar to the example where the mirrored source and collector destination are in the same VPC network (#same-network). `policy-1` is configured to collect traffic from `subnet-a` and send it to `collector-ilb`.

`policy-2` is configured in `project-a` but specifies `subnet-b` as a mirrored source. Because `network-a` and `network-b` are peered, the destination collector can collect traffic from `subnet-b`.

The networks are in different projects and might have different owners. It's possible for either owner to create the packet mirroring policy if they have the right permissions:

- If the **owners of** `project-a` create the packet mirroring policy, they must have the `compute.packetMirroringAdmin` role on the network, subnet, or instances to mirror in `project-b`.
- If the **owners of** `project-b` create the packet mirroring policy, they must have `compute.packetMirroringUser` role in `project-a`.

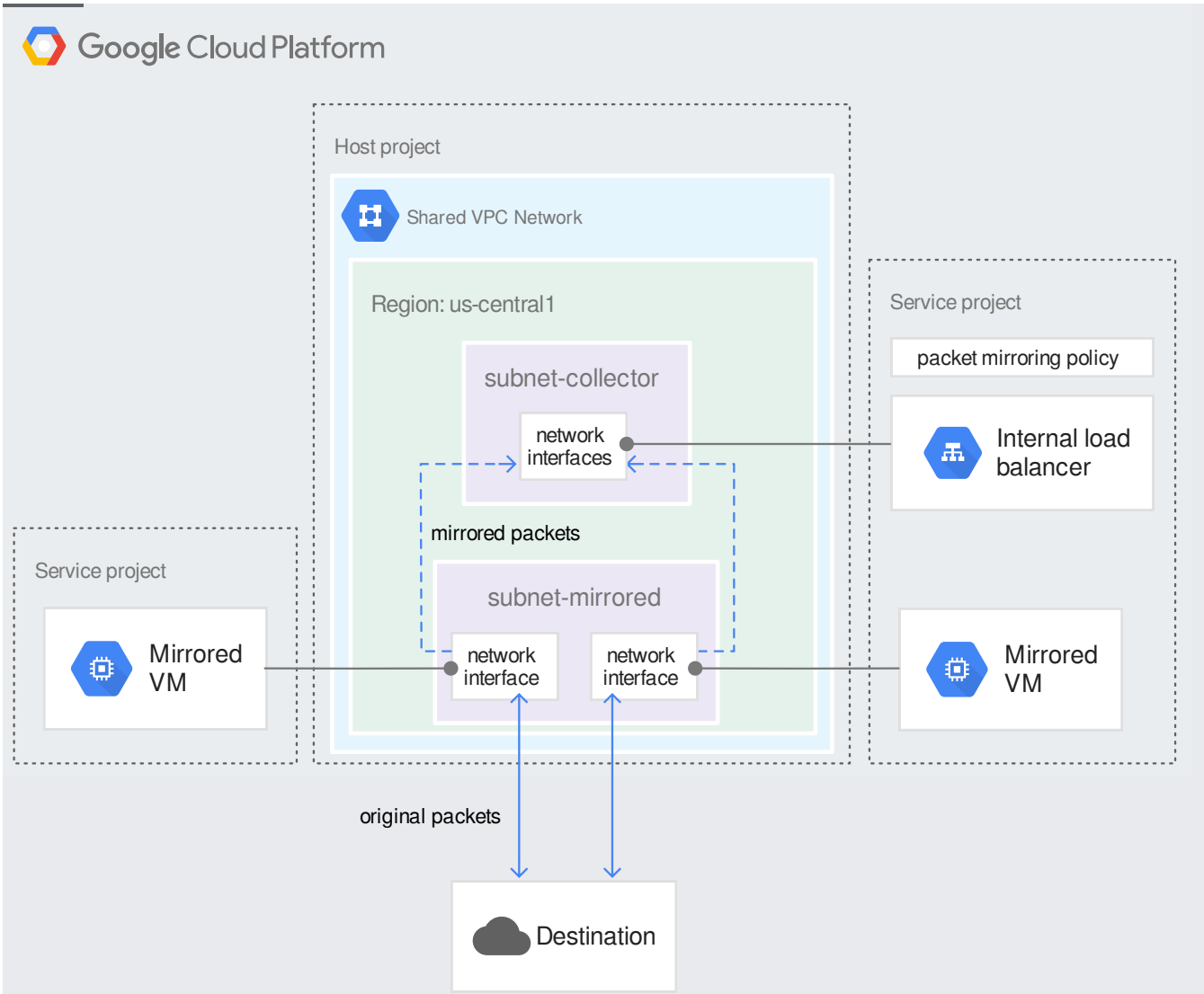
For more information about enabling private connectivity across two VPC networks, see VPC Network Peering (/vpc/docs/vpc-peering).

In the following Shared VPC scenarios, the mirrored instances for the collector destination are all in the same Shared VPC network. Even though the resources are all in the same network, they can be in different projects, such as the host project or several different service projects. The following examples show where packet mirroring policies must be created and who can create them.

If both the mirrored sources and collector destination are in the same project, either in a host project or service project, the setup is similar to having everything in the same VPC network (#same-network). The project owner can create all the resources and set the required permissions in that project.

For more information, see Shared VPC overiew (/vpc/docs/shared-vpc).

In the following example, the collector destination is in a service project that uses a subnet in the host project. In this case, the policy is also in the service project. The policy could also be in the host project.



(/vpc/images/packet-mirroring/service-project.svg)

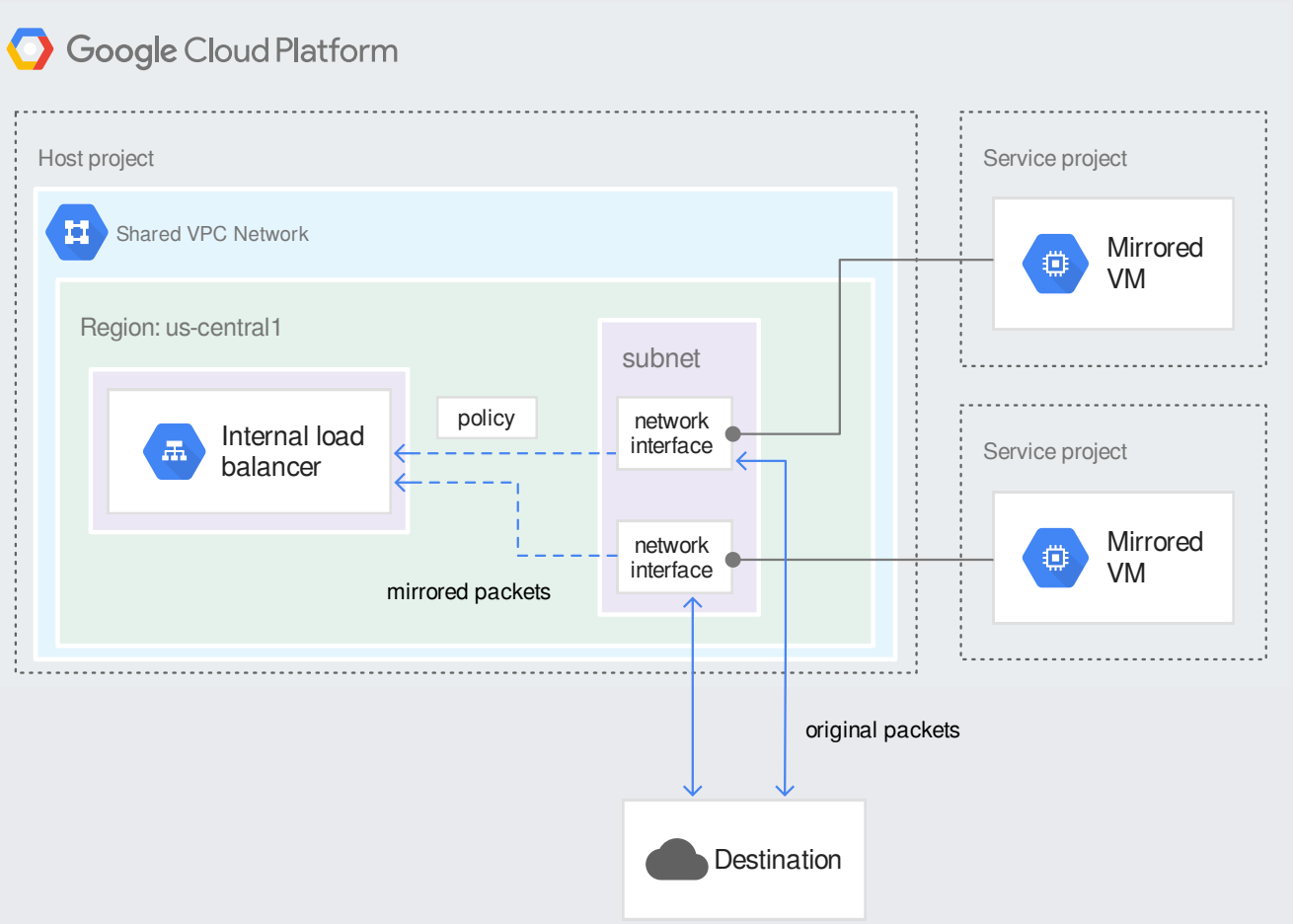
Collector destination in service project (click to enlarge)

In the preceding diagram, the service project contains the collector instances that use the collector subnet in the Shared VPC network. The packet mirroring policy was created in the service project and is configured to mirror instances that have a network interface in `subnet-mirrored`.

Service or host project users can create the packet mirroring policy. To do so, users must have the `compute.packetMirroringUser` role in the service project where the collector destination is located. Users must also have the `compute.packetMirroringAdmin` role on the mirrored sources.

In the following example, the collector destination is in the host project and mirrored instances are in the service projects.

This example might apply to scenarios where developers deploy applications in service projects and use the Shared VPC network. They don't have to manage the networking infrastructure or Packet Mirroring. Instead, a centralized networking or security team, who have control over the host project and Shared VPC network, are responsible for provisioning packet mirroring policies.



(/vpc/images/packet-mirroring/host-project.svg)

Collector destination in host project (click to enlarge)

In the preceding diagram, the packet mirroring policy is created in the host project, where the collector destination is located. The policy is configured to mirror instances in the mirrored subnet. VM instances in service projects can use the mirrored subnet, and their traffic is mirrored.

Service or host project users can create the packet mirroring policy. To do so, users in the service project must have the `compute.packetMirroringUser` role in the host project. Users in the host project require the `compute.packetMirroringAdmin` role for mirrored sources in the service projects.

You can include VM instances that have multiple network interfaces in a packet mirroring policy. Because a policy can mirror resources from a single network, you cannot create one policy to mirror

traffic for all network interfaces of an instance. If you wanted to mirror other network interfaces, you must create a policy for each interface because each one is in a different network.

- To create and manage packet mirroring policies, see Using Packet Mirroring (/vpc/docs/using-packet-mirroring).
- To view metrics and check your existing packet mirroring policies, see Monitoring Packet Mirroring (/vpc/docs/monitoring-packet-mirroring).
- For information about internal load balancers, see Internal TCP/UDP Load Balancing concepts (/load-balancing/docs/internal/).