Packet Mirroring

Archived: 2026-04-05 22:30:52 UTC

Packet Mirroring Stay organized with collections Save and categorize content based on your preferences.

This page is an overview of Packet Mirroring in the Virtual Private Cloud (VPC) network. If you want to analyze your workloads' network traffic at scale and monitor network traffic using third-party virtual appliances, use Network Security Integration Packet Mirroring. For more information, see Out-of-band integration overview.

Packet Mirroring clones the traffic of specified instances in your VPC network and forwards it for examination. Packet Mirroring captures all traffic and packet data, including payloads and headers. The capture can be configured for both egress and ingress traffic, only ingress traffic, or only egress traffic.

The mirroring happens on the virtual machine (VM) instances, not on the network. Consequently, Packet Mirroring consumes additional bandwidth on the VMs.

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.

How it works

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 VM 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 passthrough Network 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.

Filtering

By default, Packet Mirroring collects all IPv4 traffic of mirrored instances. Instead of collecting all IPv4 traffic, you can use filters to expand the traffic that's collected to include all or some IPv6 traffic. You can also use filters to narrow the traffic that's mirrored, which can help you limit the bandwidth that's used by mirrored instances.

You can configure filters to collect traffic based on protocol, CIDR ranges (IPv4, IPv6, or both), direction of traffic (ingress-only, egress-only, or both), or a combination.

Policy order

Multiple packet mirroring policies can apply to an instance. The priority of a packet mirroring policy is always `1000` and cannot be changed. Identical policies are not supported. Google Cloud can send traffic to any of the load balancers that have been configured with identical packet mirroring policies. 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.

Depending on each policy's filter, Google Cloud chooses a policy 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 `198.51.100.3/24:TCP` and another policy that has the filter `2001:db8::/64:TCP: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 to choose which policy to use. 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.

Google Cloud chooses a policy based on a filter:

- If policies have different but overlapping CIDR ranges and the same exact protocols, Google Cloud chooses the policy that uses the most specific CIDR 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 specify the same exact CIDR range with overlapping protocols, Google Cloud chooses a policy with the most specific protocol. 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`. For matching TCP traffic, Google Cloud uses the `10.240.1.0/24:TCP` policy. The `10.240.1.0/24:ALL` policy applies to matching traffic for all other protocols.

- 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 `2001:db8::/64:TCP` and another for `2001:db8::/64:UDP`. Depending on the mirrored traffic's protocol, Google Cloud uses either the TCP or UDP policy.

- If overlapping policies have the same exact filter, they are identical. In this case, Google Cloud might choose the same policy or a different policy each time that matching traffic is re-evaluated against these policies. We recommend that you avoid creating identical packet mirroring policies.

VPC Flow Logs

VPC Flow Logs doesn't log mirrored packets. If a collector instance is on a subnet that has VPC Flow Logs enabled, traffic that is sent directly to the collector instance is logged, including traffic from mirrored instances. That is, if the original destination IPv4 or IPv6 address matches the IPv4 or IPv6 address of the collector instance, the flow is logged.

For more information about VPC Flow Logs, see Using VPC Flow Logs.

Key properties

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

- Each packet mirroring policy defines *mirrored sources* and a *collector destination*. You must adhere to the following rules:
    - All mirrored sources must be in the same project, VPC network, and Google Cloud region.
    - A collector destination must be in the same region as the mirrored sources. A collector destination can be located in either the same VPC network as the mirrored sources or a VPC network connected to the mirrored sources' network using VPC Network Peering.
    - Each mirroring policy can only reference a single collector destination. However, a single collector destination can be referenced by multiple mirroring policies.

- All layer 4 protocols are supported by Packet Mirroring.

- You cannot mirror and collect traffic on the same network interface of a VM instance because doing this would cause a mirroring loop.

- To mirror traffic passing between Pods on the same Google Kubernetes Engine (GKE) node, you must enable Intranode visibility for the cluster.

- To mirror IPv6 traffic, use filters to specify the IPv6 CIDR ranges of the IPv6 traffic that you want to mirror. You can mirror all IPv6 traffic by using a CIDR range filter of `::/0`. You can mirror all IPv4 and IPv6 traffic by using the following comma-separated CIDR range filter: `0.0.0.0/0,::/0`.

- 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. You can alter this behaviour by specifying that only ingress or only egress packets are mirrored.
- 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 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 IPv4 and IPv6 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.
- Mirrored traffic is encrypted only if the VM encrypts that traffic at the application layer. While VM-to-VM connections within VPC networks and peered VPC networks are encrypted, the encryption and decryption happens in the hypervisors. From the perspective of the VM, this traffic is not encrypted.

Use cases

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

Enterprise security

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) 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 require that most packets be examined. Packet Mirroring provides a solution for capturing different attack vectors, such as infrequent communication or attempted but unsuccessful communication.

Application performance monitoring

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.

Example collector destination topologies

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.

Collector destination in the same network

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


A packet mirroring policy with a mirrored source and a destination collector in the same VPC network.
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 `mirrored-subnet` and send mirrored traffic to the internal passthrough Network 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.

Collector destination in a peer network

You can build a centralized collector model, where instances in different VPC networks send mirrored traffic to a collector destination in a central VPC network. That way, you can use a single destination collector.

In the following example, the `collector-load-balancer` internal passthrough Network Load Balancer is in the `us-central1` region in the `network-a` VPC network in `project-a`. This destination collector can be used by

two packet mirroring policies:

- `policy-1` collects packets from mirrored sources in the `us-central1` region in the `network-a` VPC network in `project-a` and sends them to the `collector-load-balancer` destination.
- `policy-2` collects packets from mirrored sources in the `us-central1` region in the `network-b` VPC network in `project-b` and sends them to the same `collector-load-balancer` destination.

Two mirroring policies are required because mirrored sources exist in different VPC networks.


A packet mirroring policy in a central network where the collector destination lives. The network is peered with other networks where the mirrored sources live.
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. `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.

Shared VPC

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. The project owner can create all the resources and set the required permissions in that project.

For more information, see Shared VPC overiew.

Collector destination in service project

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.


The relationship between the host and service projects for Packet Mirroring.
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.

Collector destination in host project

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.


The relationship between the host and service projects for Packet Mirroring.
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.

Multi-interface VM instances

You can include VM instances that have multiple network interfaces in a packet mirroring policy.

A policy can mirror resources only from a single network. If your multi-NIC instance has network interfaces in different networks, you cannot create one policy to mirror traffic for all of the network interfaces. If you need to mirror additional network interfaces that are attached to different networks, you must create one packet mirroring policy for each interface.

Pricing

You are charged for the amount of data processed by Packet Mirroring. For details, see Packet Mirroring pricing.

You are also 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.

What's next

- Use Packet Mirroring.
- Monitor Packet Mirroring.
- Internal passthrough Network Load Balancer overview.
- Packet Mirroring partner providers.
- Out-of-band integration overview.

Source: https://cloud.google.com/vpc/docs/packet-mirroring