Why do you bother locking doors?

Google Cloud

The principal illustrated in this image is very important and will help you understand the next several slides about security.

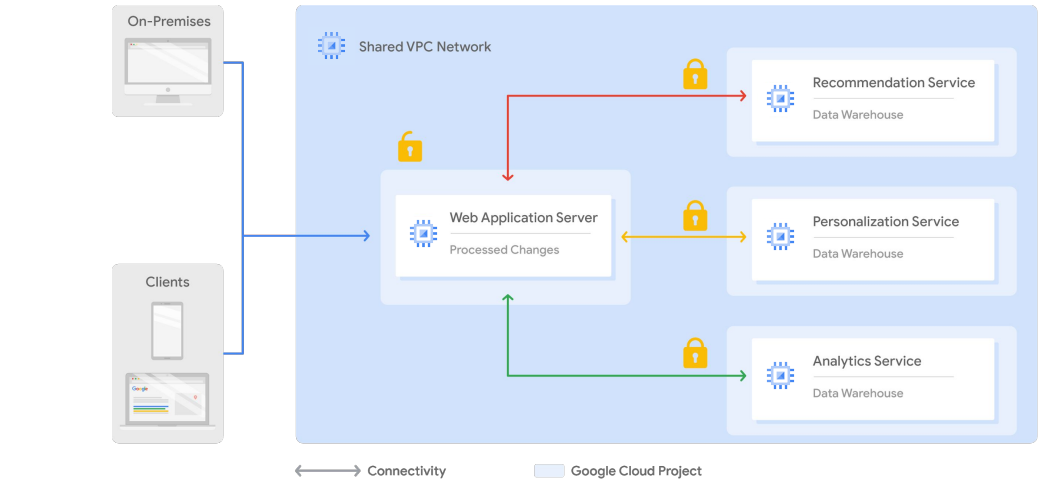
This is a very simple door lock. But it is on the **inside** of this door. The person or people inside can unlock this very easily at any time. For them it does nothing but puts a step in the way of using the door. But the lock is not supposed to do anything *for* them. The purpose of the lock is to keep *other people out*. So to do that, you need to lock your own people *in*.

This principal is on display in the way networking contributes to security.

Consider an internal firewall between two VMs. Why would you want to do that?

If you think about it functionally it makes no sense. Because the VMs can already communicate. So adding a firewall between them does nothing for them. It only makes it inconvenient if they want to communicate using a different protocol. Now you have to go change the firewall rule to allow that protocol. And it seems like invented work. Because if you just didn't put the firewall rule in place to begin with you wouldn't have to modify it later. However, the firewall rule is not for those VMs. It doesn't do anything for them. It is to keep others out of their communications. To prevent someone from spoofing or injecting bad traffic as part of an attack to either violate privacy or deny service.

Shared VPC: Keep others out by locking you in



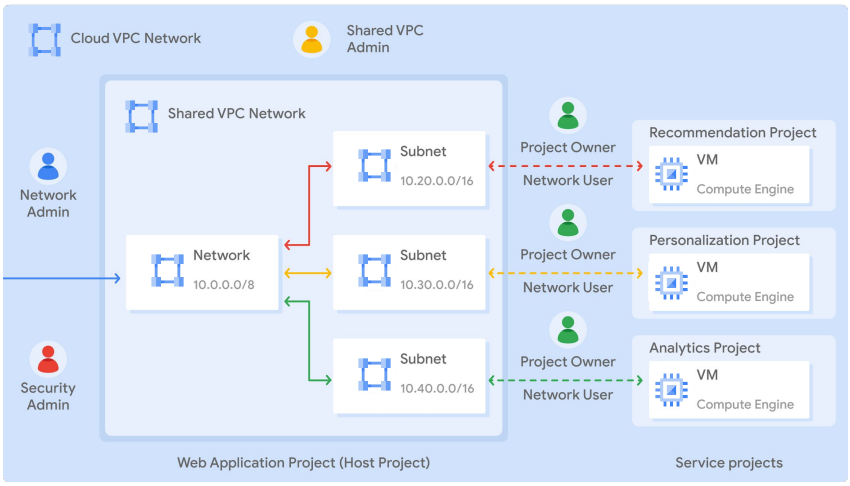
Google Cloud

Shared VPC allows an organization to connect resources from multiple projects to a common VPC network. This allows the resources to communicate with each other securely and efficiently using internal IPs from that network.

For example, in this diagram there is one network that belongs to the Web Application Server's project. This network is shared with three other projects, namely the Recommendation Service, the Personalization Service, and the Analytics Service. Each of those service projects has instances that are in the same network as the Web Application Server, allowing for private communication to that server, using internal IP addresses. The Web Application Server communicates with clients and on-premises using the server's external IP address. The backend services, on the other hand, cannot be reached externally because they only communicate using internal IP addresses.

When you use shared VPC, you designate a project as a host project and attach one or more other service projects to it. In this case, the Web Application Server's project is the host project, and the three other projects are the service projects. The overall VPC network is called the shared VPC network.

An example of Shared VPC keeping a system safe



Google Cloud

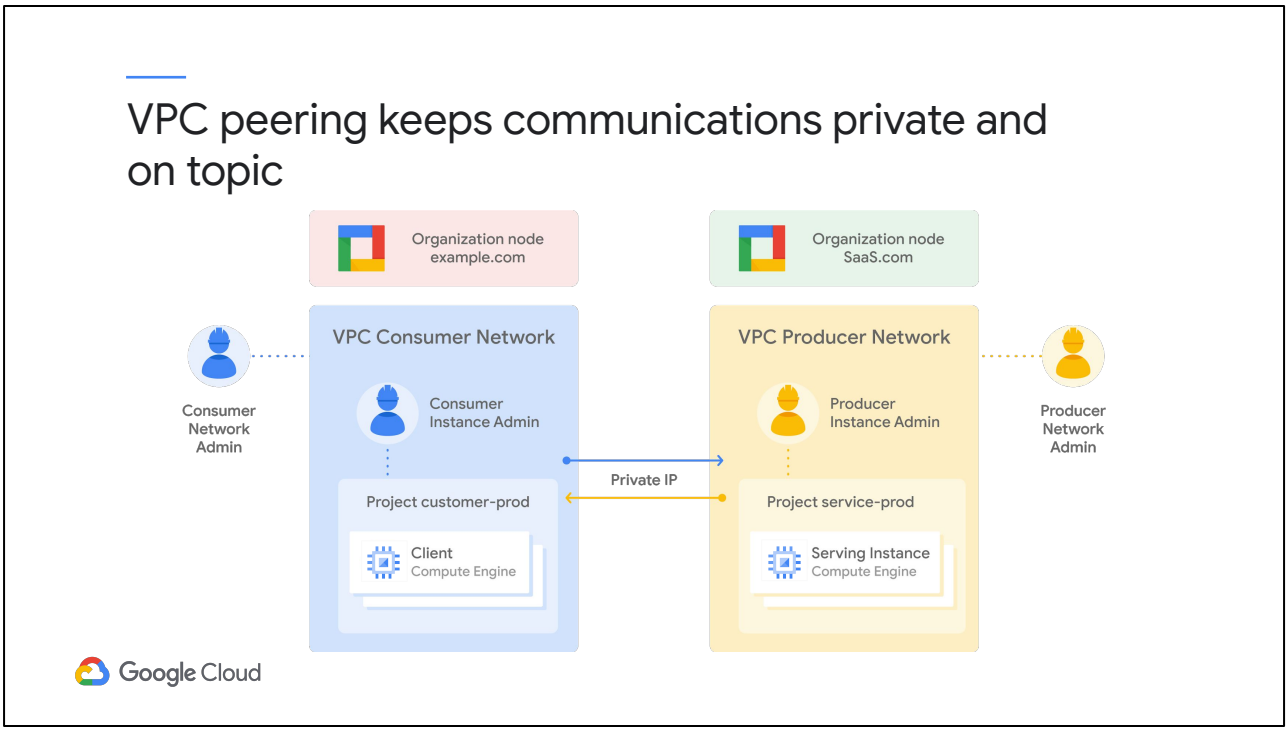
This example has one host project and 3 service projects. The Shared VPC Admin, which was nominated by an organization admin, configured the Web Application Project to be a host project with subnet-level permissions. Doing so allowed the Shared VPC Admin to selectively share subnets from the VPC network.

Next, the Shared VPC Admin attached the three service projects to the host project and gave each project owner the Network User role for the corresponding subnets. Each project owner then created VM instances from their service projects in the shared subnets.

The billing for those VM instances is attributed to the project where the resources are created, which are the service projects.

Shared VPC Admins have full control over the resources in the host project, including administration of the shared VPC network. They can optionally delegate the Network Admin and Security Admin roles for the host project. Overall, shared VPC is a centralized approach to multi-project networking because security and network policy occurs in a single designated VPC network.

For a demo on how to create VM instances in a Shared VPC network, please refer here:
https://storage.googleapis.com/cloud-training/gcpnet/student/M3%20-%20Shared%20VPC.mp4



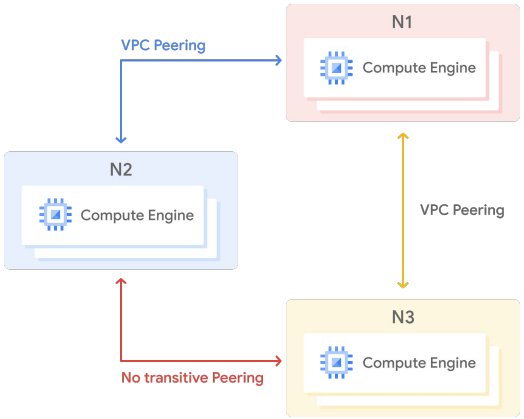
VPC Network Peering allows private RFC 1918 connectivity across two VPC networks, regardless of whether they belong to the same project or the same organization. Now, remember that each VPC network will have firewall rules that define what traffic is allowed or denied between the networks.

For example, in this diagram there are two organizations that represent a consumer and a producer, respectively. Each organization has its own organization node, VPC network, VM instances, Network Admin and Instance Admin. In order for VPC Network Peering to be established successfully, the Producer Network Admin needs to peer the Producer Network with the Consumer Network, and the Consumer Network Admin needs to peer the Consumer Network with the Producer Network. When both peering connections are created, the VPC Network Peering session becomes Active and routes are exchanged This allows the VM instances to communicate privately, using their internal IP addresses.

VPC Network Peering is a decentralized or distributed approach to multi-project networking, because each VPC network may remain under the control of separate administrator groups and maintains its own global firewall and routing tables. Historically, such projects would consider external IP addresses or VPNs to facilitate private communication between VPC networks. However, VPC Network Peering does not incur the network latency, security, and cost drawbacks that are present when using external IP addresses or VPNs.

Tips on designing solutions that use VPC peering

- Compute Engine, Google Kubernetes Engine, and App Engine flexible environments
- Peered VPC networks remain administratively separate
- Each side of a peering association is set up independently
- No subnet IP range overlap across peered VPC networks
- Transitive peering is not supported

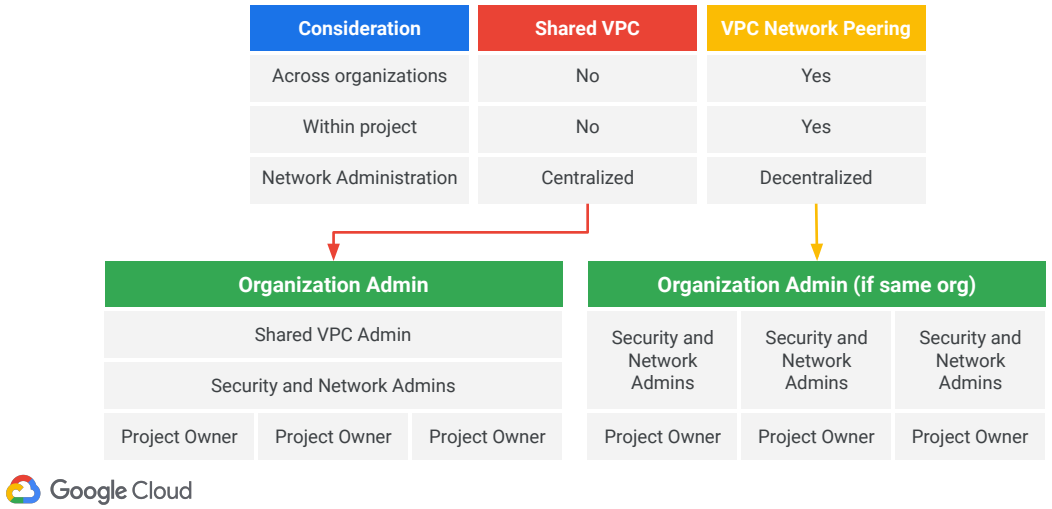


Google Cloud

Now, there are some things that we want you to remember when using VPC Network Peering:

- First of all, VPC Network Peering works with Compute Engine, Google Kubernetes Engine, and App Engine flexible environments.
- Peered VPC networks remain administratively separate. This means that 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. This allows either side to delete the peering association at any time.
- A subnet CIDR prefix in one peered VPC network cannot overlap with a subnet CIDR prefix in another peered network. This means that two auto mode VPC networks that only have the default subnets cannot peer.
- Only directly peered networks can communicate, meaning that 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 the peering. This is critical if N1 is a SaaS organization offering services to N2 and N3.

Should you use Shared VPC or VPC peering ?



Google Cloud

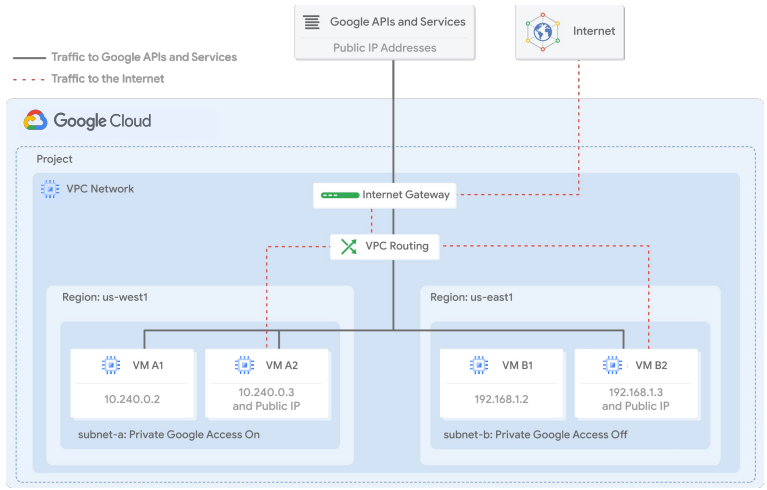
Should you use Shared VPC or VPC peering for a particular situation? That depends on the business administration requirements.

The biggest difference between the two configurations is the network administration models. Shared VPC is a centralized approach to multi-project networking, because security and network policy occurs in a single designated VPC network. In contrast, VPC Network Peering is a decentralized approach, because each VPC network can remain under the control of separate administrator groups and maintains its own global firewall and routing tables.

Also, consider the limits of VM instances per network and total among peered networks: https://cloud.google.com/vpc/docs/quota#per_network

Now that we've compared both of these configurations for sharing VPC networks across Google Cloud projects, let's look at one last use case.

Remove external IPs using Private Google Access



Google Cloud

The goal of Private Google Access is to remove external IPs from your VMs. Every time you remove an external IP you reduce the number of opportunities for an attacker to gain entry or try to deny service.

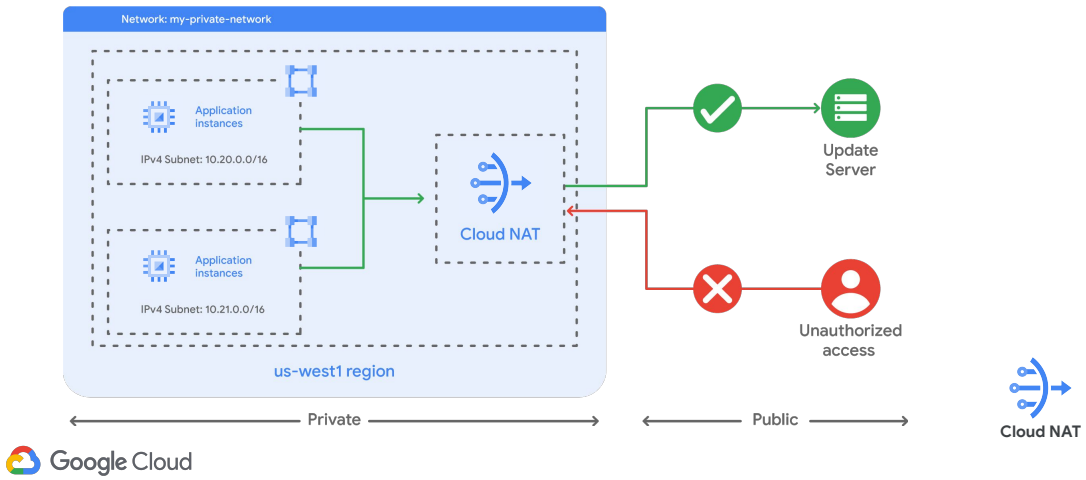
Private Google Access allows VM instances that only have internal IP addresses to reach the external IP addresses of Google APIs and services. For example, if your private VM instance needs to access a Cloud Storage bucket, you need to enable Private Google Access.

You enable Private Google Access on a subnet-by-subnet basis. As you can see in this diagram, subnet-a has Private Google Access enabled, and subnet-b has it disabled. This allows VM A1 to access Google APIs and services, even though it has no external IP address.

Private Google Access has no effect on instances that have external IP addresses. That's why VMs A2 and B2 can access Google APIs and services. The only VM that can't access those APIs and services is VM B1. This VM has no public IP address, and it is in a subnet where Google Private Access is disabled.

For a list of the services supported by Private Google Access, see: https://cloud.google.com/vpc/docs/private-access-options#pga-supported

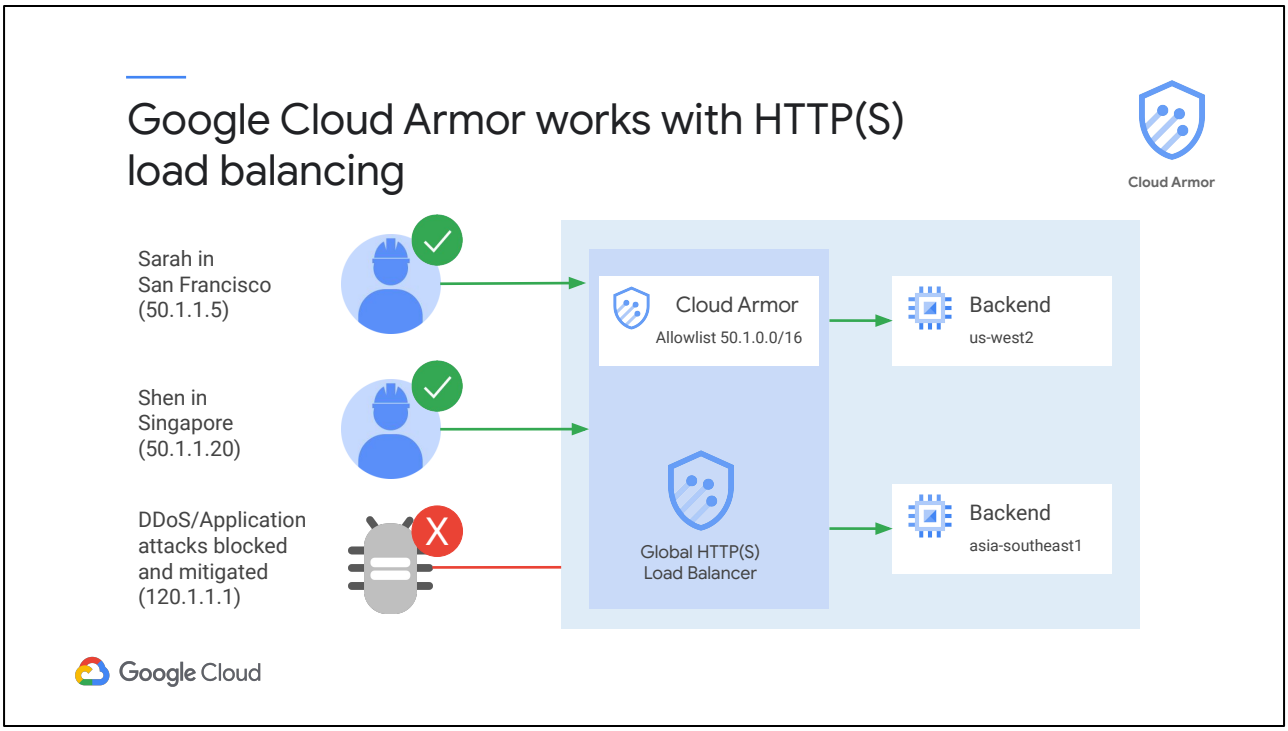
Cloud NAT provides internet access to private instances



Network Address Translation enables you to provide limited internet access without adding an external IP to your VM, and without exposing your internal IP to the world.

Cloud NAT is Google's managed network address translation service. It lets you provision your application instances without public IP addresses, while also allowing them to access the internet in a controlled and efficient manner. This means that your private instances can access the internet for updates, patching, configuration management, and more.

In this diagram, Cloud NAT enables two private instances to access an update server on the internet, which is referred to as outbound NAT. However, Cloud NAT does not implement inbound NAT. In other words, hosts outside your VPC network cannot directly access any of the private instances behind the Cloud NAT gateway. This helps you keep your VPC networks isolated and secure.



Physical hardware load balancers are a target for denial of service attacks. That is because they have physical hardware network interfaces that have a high but limited capacity. If an attacker can overrun an interface with traffic, or trigger an overrun from the internal service out to the internet, they can cause congestion that will slow or halt traffic.

Google Cloud Armor works with global HTTP(S) load balancing to provide built-in defenses against Infrastructure Distributed Denial of Service or DDoS attacks.Google Cloud Armor benefits from over a decade of experience protecting some of the world's largest internet properties like Google Search, Gmail, and YouTube.

Google Cloud Armor enables you to restrict or allow access to your HTTP(S) load balancer at the edge of the Google Cloud network, meaning as close as possible to the user and to malicious traffic. For example, in this diagram, Sarah in San Francisco and Shen in Singapore are two employees who are allowed to access your HTTP load balancer. Therefore, their traffic will be forwarded to the backend in the us-west2 region and the backend in the asia-southeast1 region, respectively. A DDoS attack, on the other hand, can be blocked directly at the edge without consuming resources or entering your VPC network.