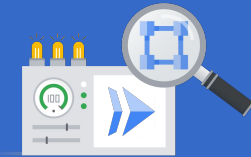


Serverless Networking Guide



January 2024

Meet our characters

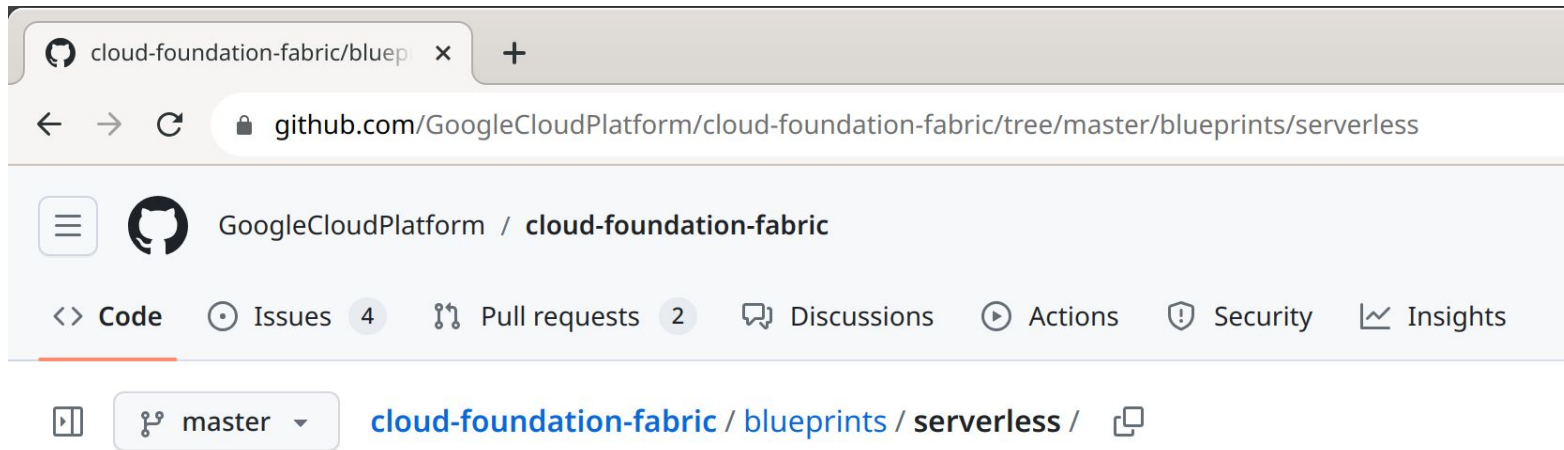


This is Ginny. She is a talented engineer working for ACME, interested in everything Cloud.

And this is her friend Nick, an experienced Google Cloud Engineer always willing to help.

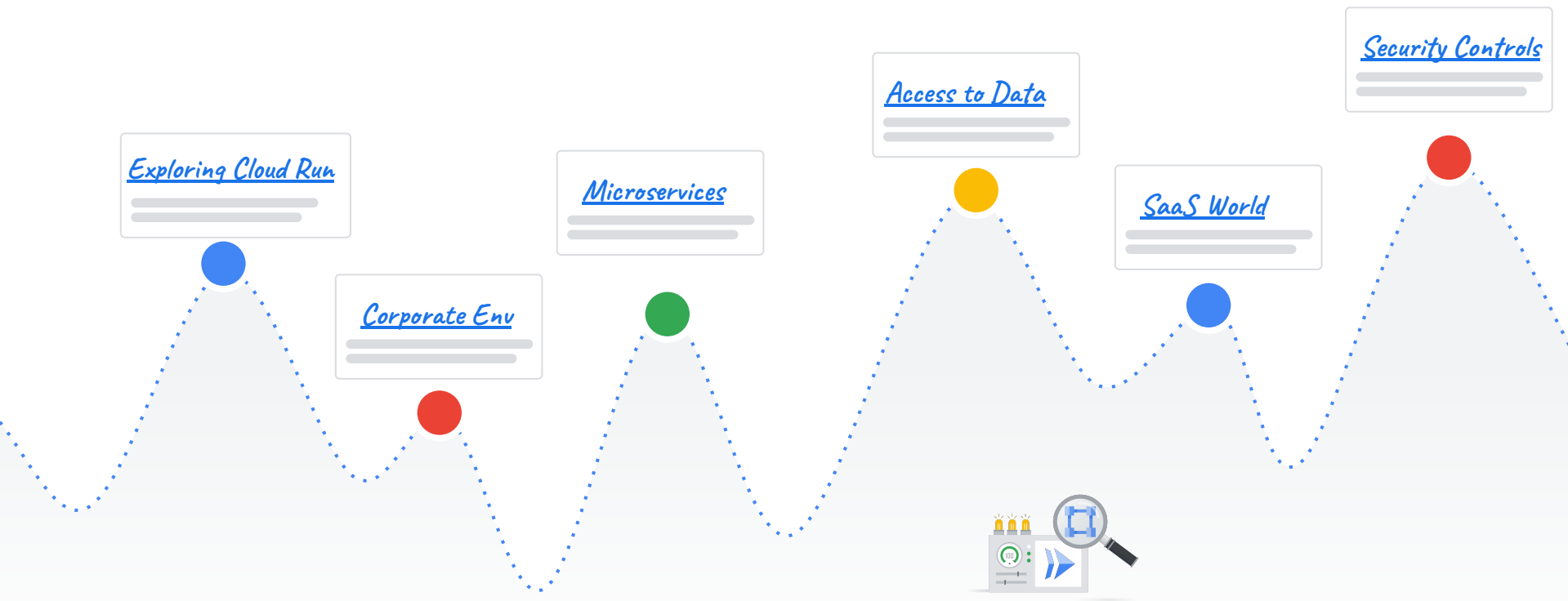


Check out the code



You can check out code with examples of this guide from the [cloud-foundation-fabric](#) repository. All serverless related examples are at [blueprints/serverless](#).

Section I: The Basics



Exploring Cloud Run

My serverless "Hello, World!"

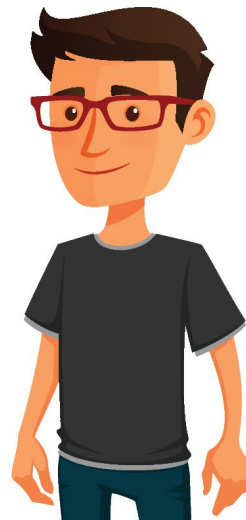
[Terraform code](#)

> Hello world

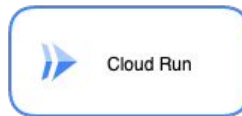


*Hey Nick, I heard about serverless and I want to make my first serverless "Hello, World!". But now I'm puzzled, how will I **connect to a serverless service?***

*Hahaha! Hi Ginny, yeah, networking may not be obvious in the serverless world. But first, there are several serverless options, hmmm... do you know **Cloud Run?***



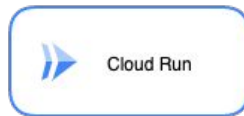
<https://hello-olc7uafoke-ew.a.run.app>



*Cloud Run doesn't run within a VPC, but when you deploy an app you get a **default URL** to access it based on your project, service name and region where it is deployed.*

~~<https://hello-olc7uafoka-ew.a-run.app>~~

<https://ginny-project.acme.org>



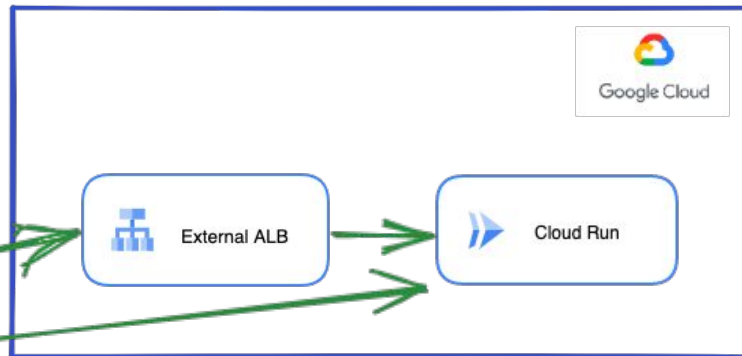
Oh, great, easy! Hmm... but what
if I want to use *my own domain*?



<https://ginny-project.acme.org>

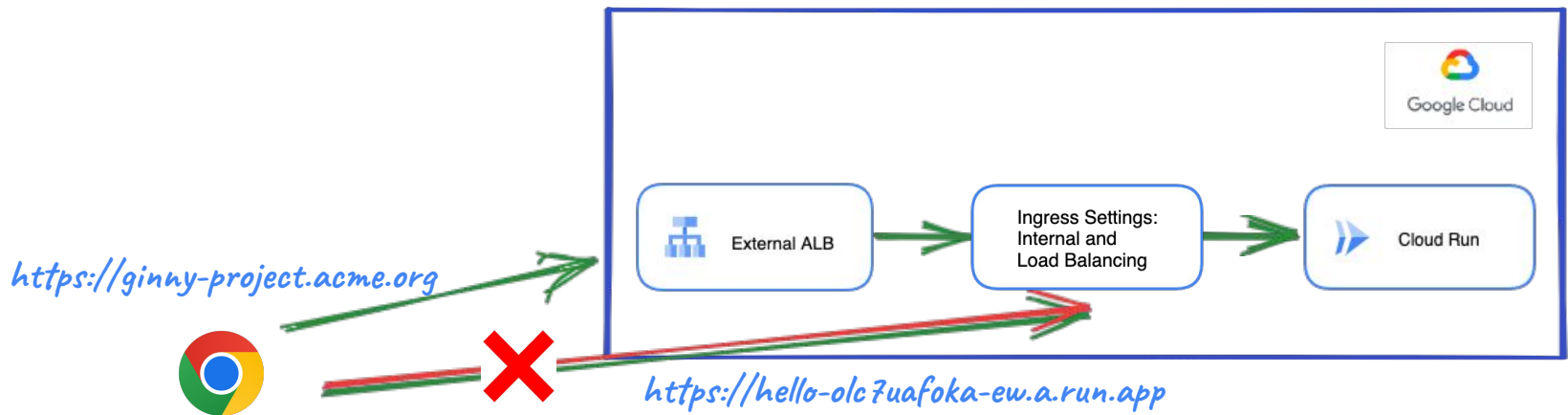


<https://hello-olc7uafoka-ew.a.run.app>

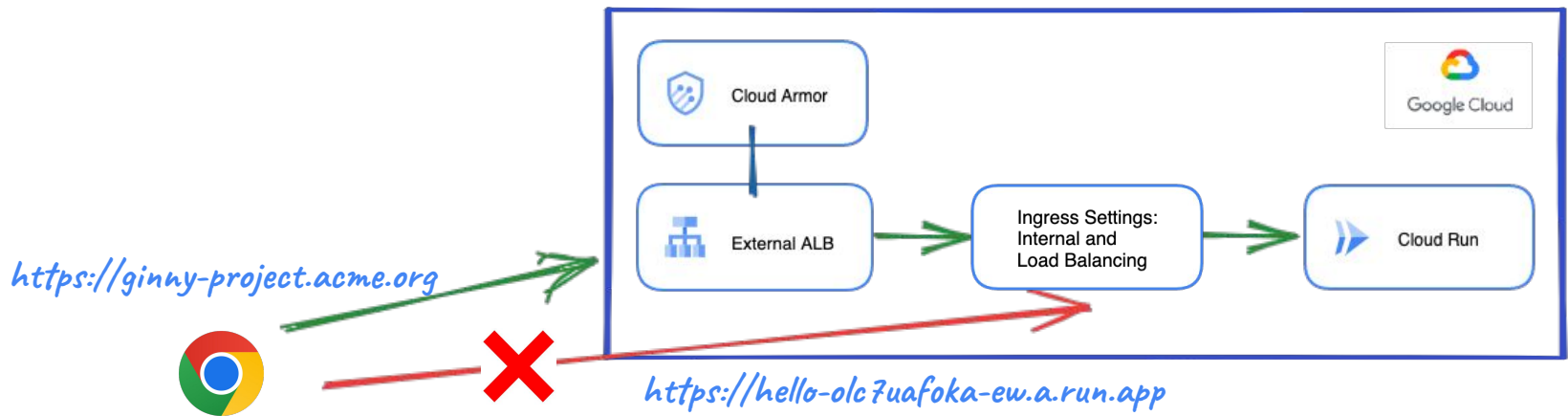


To use your own *custom domain* you need an *external Application Load Balancer* in front of your *Cloud Run* app.

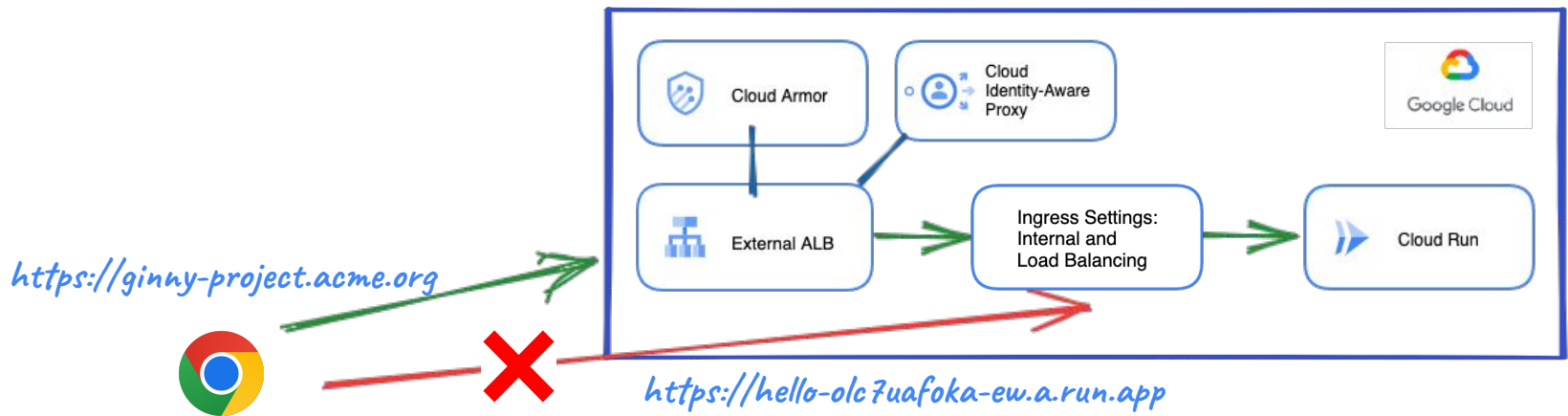
However, note that your app is *still accessible* through its *default URL*.



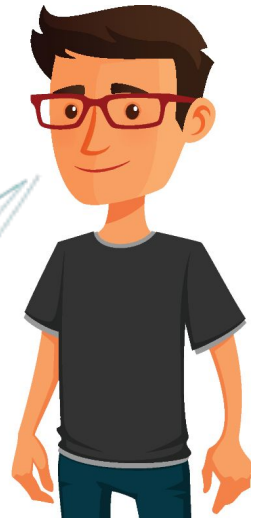
To block that access you can configure *Ingress Settings* to only accept Internet requests if they come through the Load Balancer.

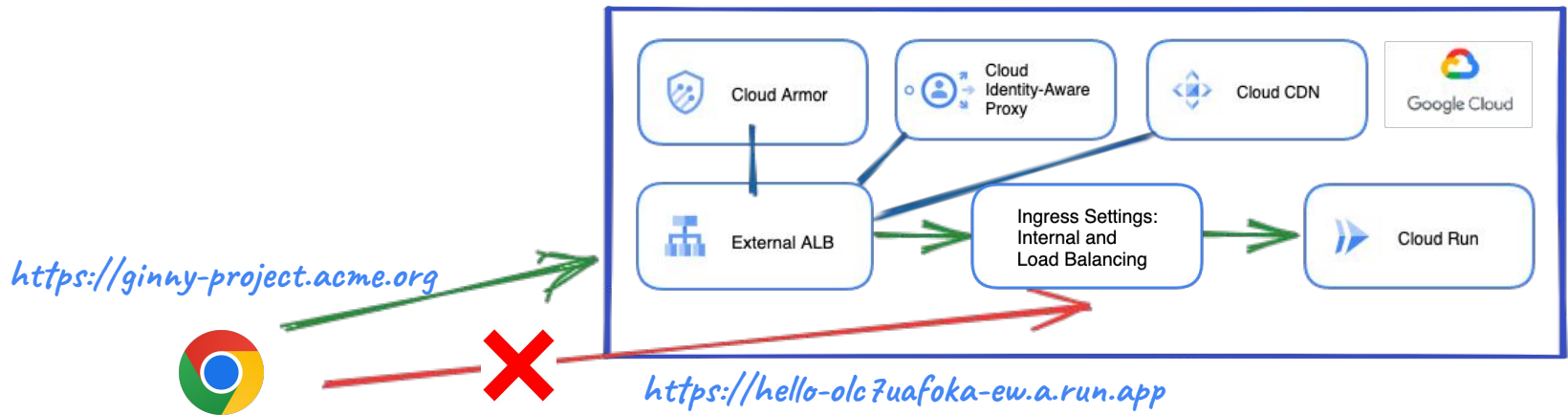


Got it! And since I put an external Application Load Balancer in front, I can use *Cloud Armor* to protect my web app, right? This is cool!



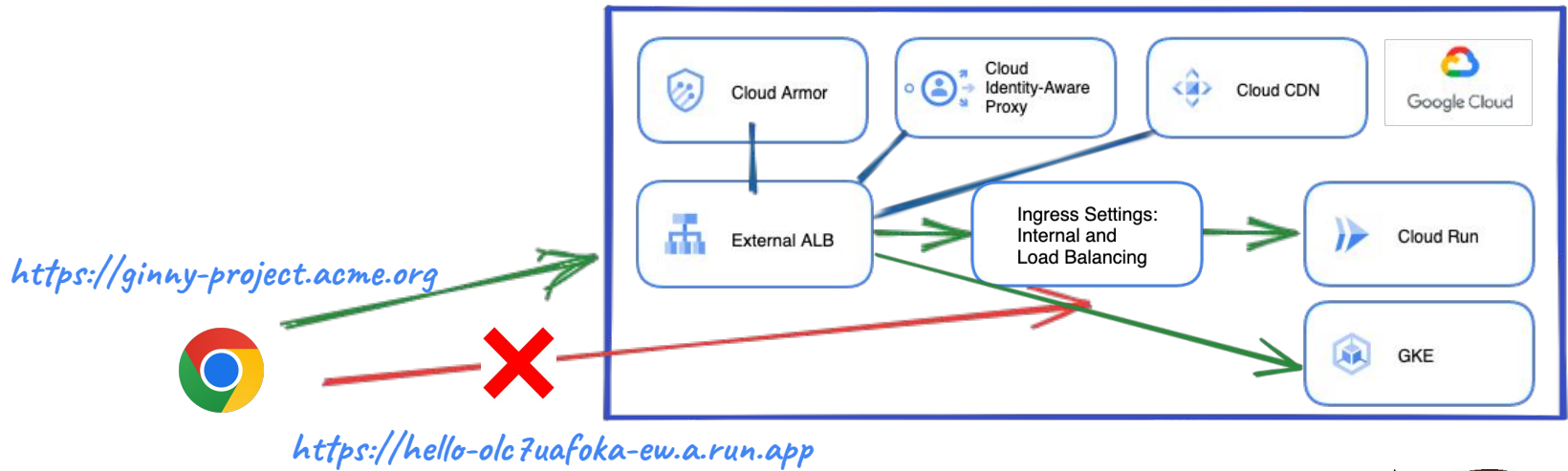
*Absolutely, you can protect your services with Cloud Armor. And if you want to **control access** using identity and context you can also enable **IAP** at the Load Balancer Backend Services. This will provide a central authorization layer for applications.*



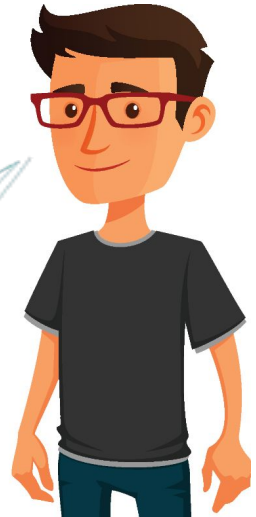


Not only that, you can also **cache** your static content through **Cloud CDN**, and Cloud Armor will protect the origin through Backend Policies as well as cached content through Edge Security Policies.



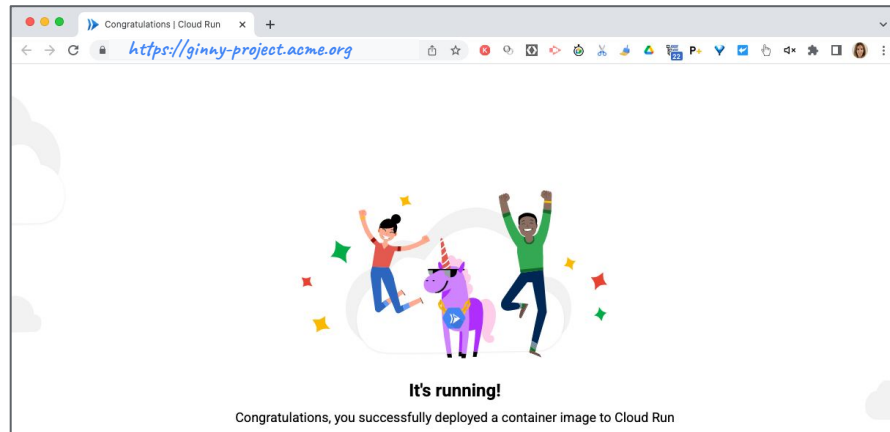


And you can use this same Load Balancer to load balance traffic to applications running in other type of backends like GCE or GKE besides Cloud Run.





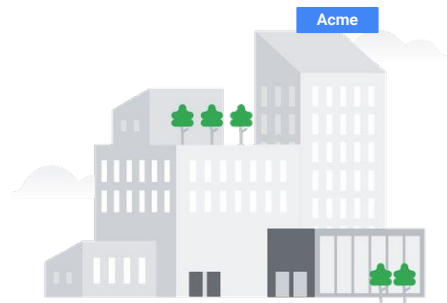
*That's it. And all of this can be easily created with
Terraform, isn't it great?*



The corporate environment

Developing an enterprise application

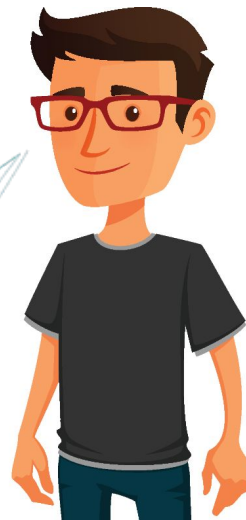
Terraform code



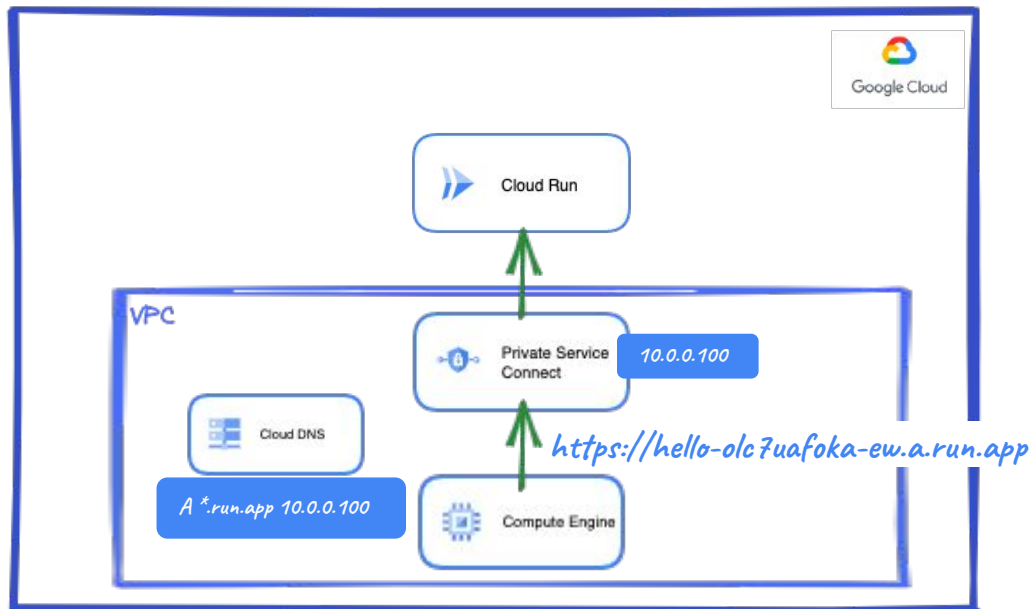


*Hi, Nick! You know? I showed my toy Cloud Run app and now my manager is interested in exploring it for **corporate use**. But I'm afraid Cloud Run doesn't fit, does it?*

You mean, privately access it? Sure, I'll show you how you can do that!

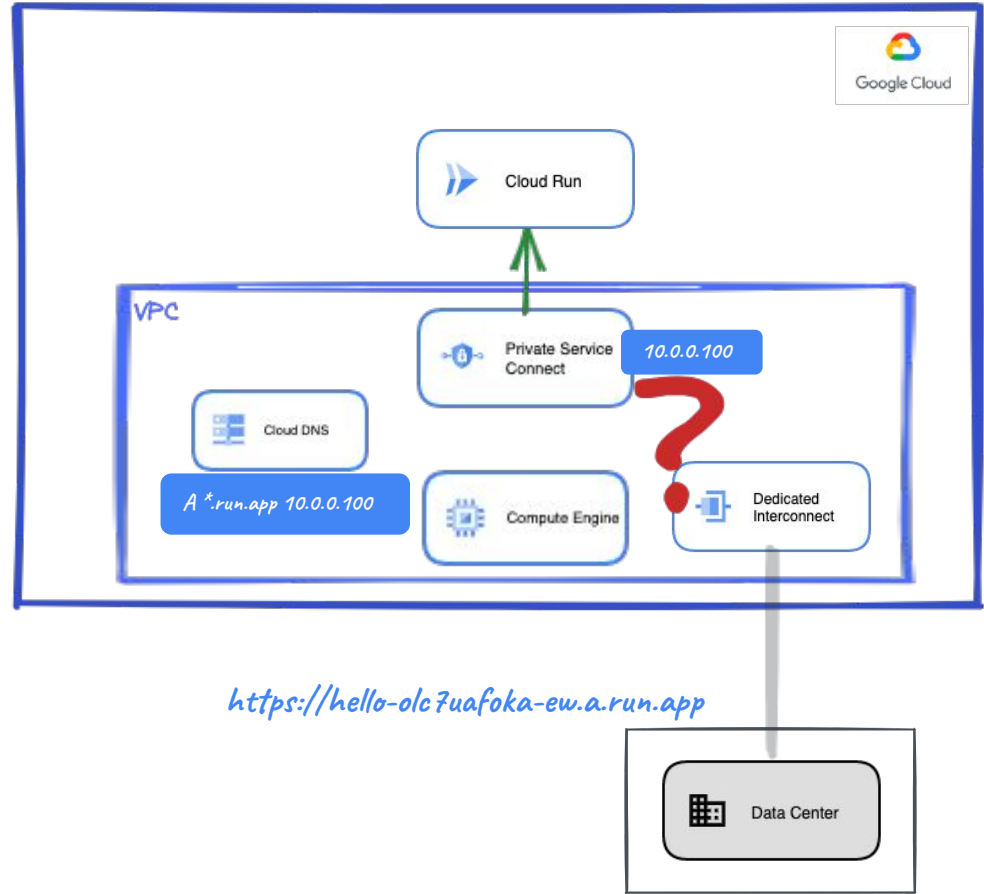


I imagine your company has deployed some apps in VMs and you want to access Cloud Run services from them, right?

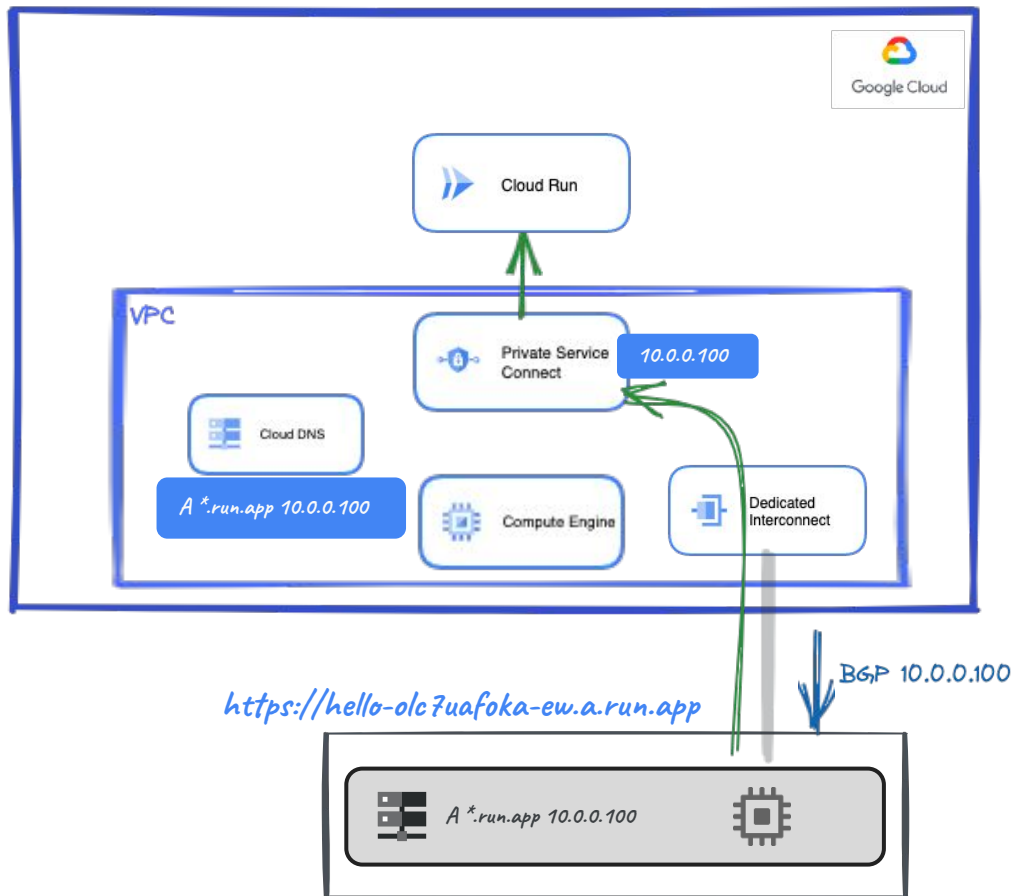


*You can use either **Private Google Access** or **Private Service Connect**. For **PSC**, just enable **Private Google Access** at the subnet where the VM is, create a **PSC endpoint** and make **DNS** point to it.*

Well, ahem... We don't have workloads in GCP yet. I was thinking accessing from onprem through Cloud VPN or Interconnect. Is that possible?



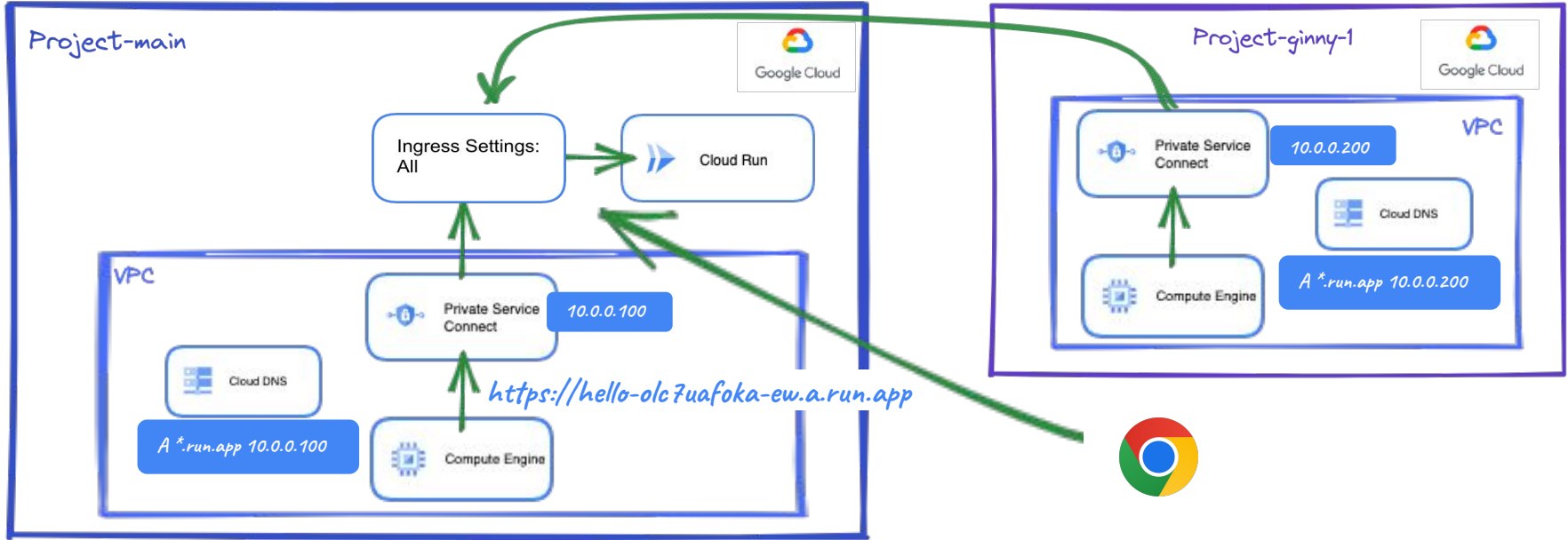
*Absolutely! **Announce the PSC endpoint to onprem**, make DNS changes if needed and you are good to go!*





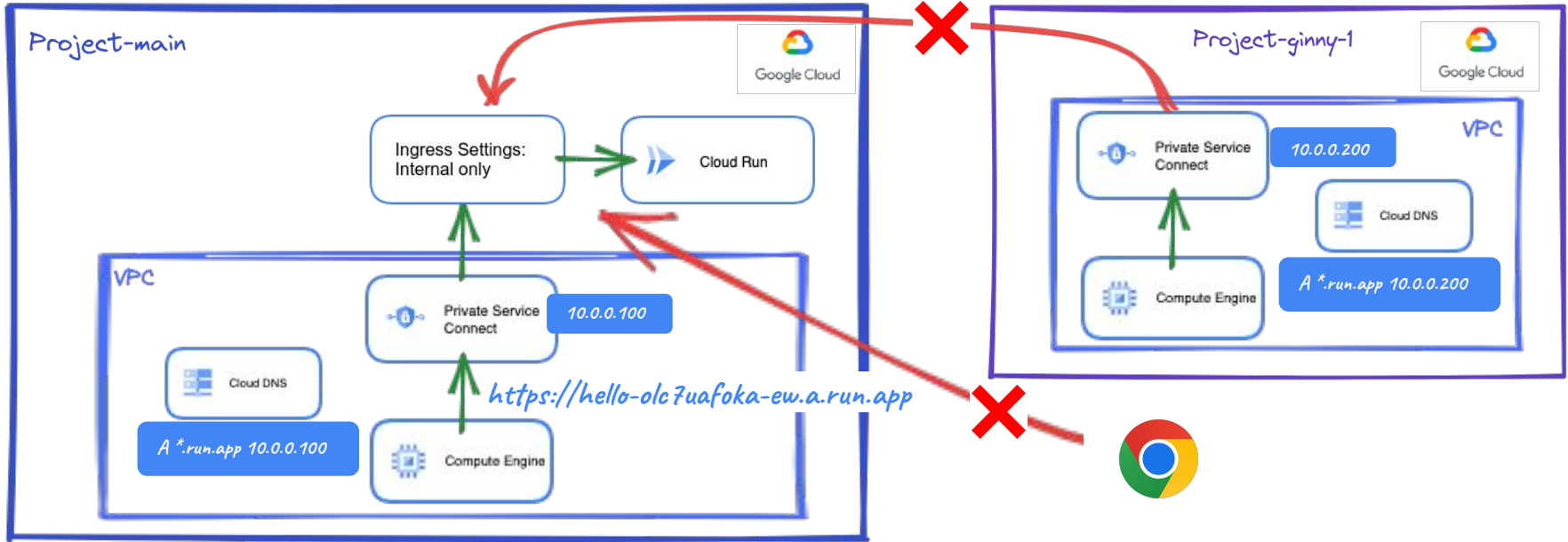
*Looks simple! Ok, we've considered one project but corporate apps are intended to be used by **multiple teams and projects**. Would that work?*

<https://hello-olc7uafoka-ew.a.run.app>



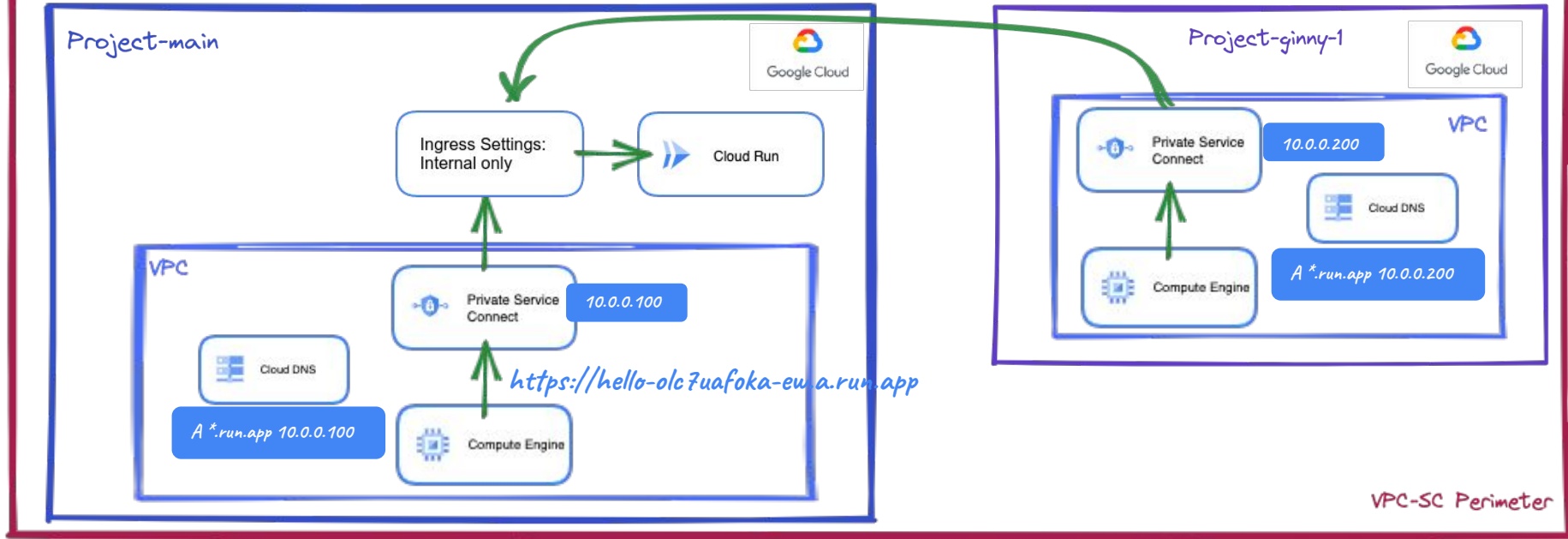
Yeah, good point. By *default*, you can *access* your service *from anywhere*: the same project, another project, or... the Internet, remember?

<https://hello-olc7uafoka-ew.a.run.app>

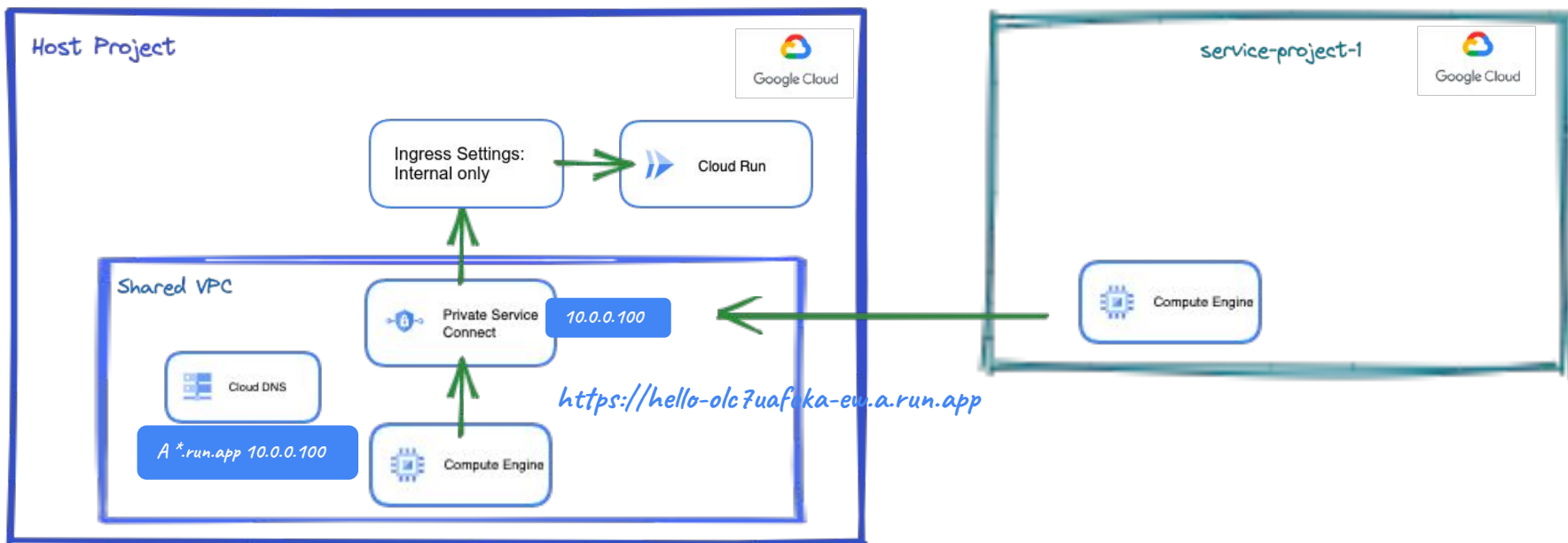


Hmmm... Configuring *Ingress settings* to *Internal only* you could block access from the Internet, but access from another project will not work as this call is **NOT** considered *internal*.

<https://hello-olc7uafoka-ew.a.run.app>

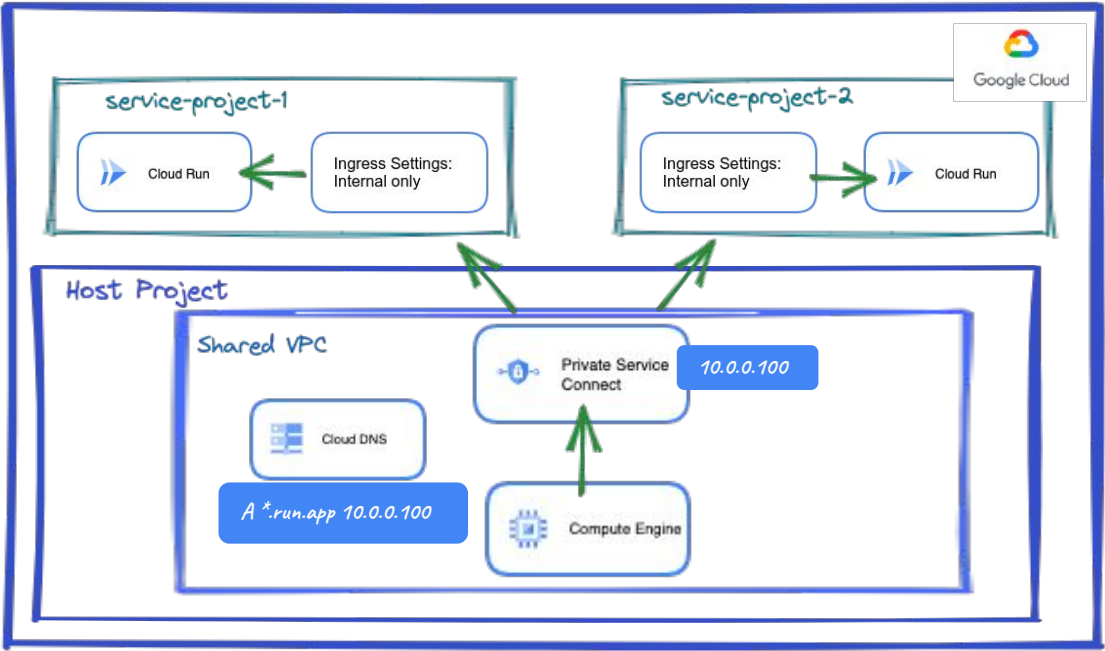


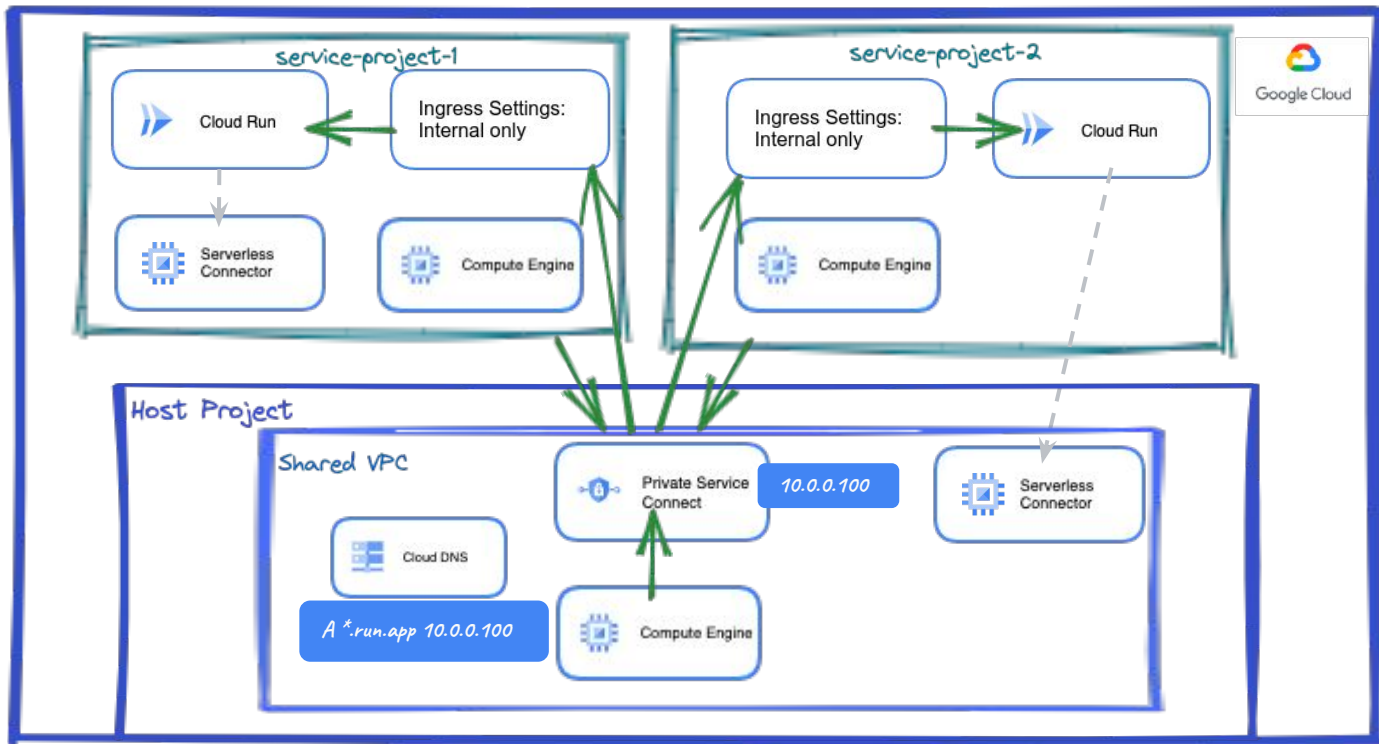
However, if you include the projects within the same **VPC-SC perimeter**, this call will be considered **internal** and will work!



Accessing from a Service Project of the Host Project where Cloud Run service is running is also considered *internal* traffic.

*In reality, you can deploy Cloud Run in any project using the Shared VPC. Calls between these projects are considered **internal**.*



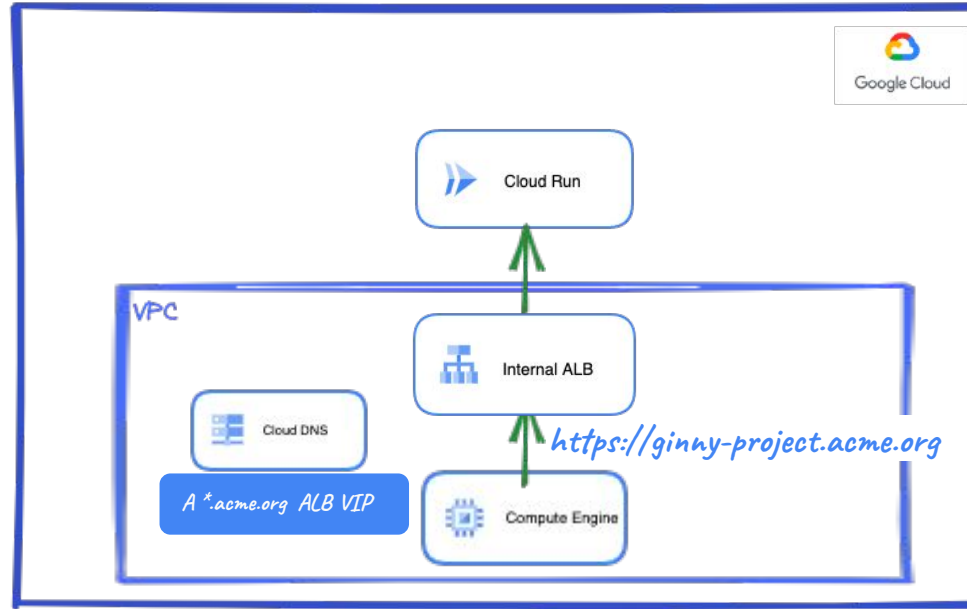


That includes calls between service projects. Note: in a service project, Cloud Run needs a serverless connector even though this is Ingress Traffic.

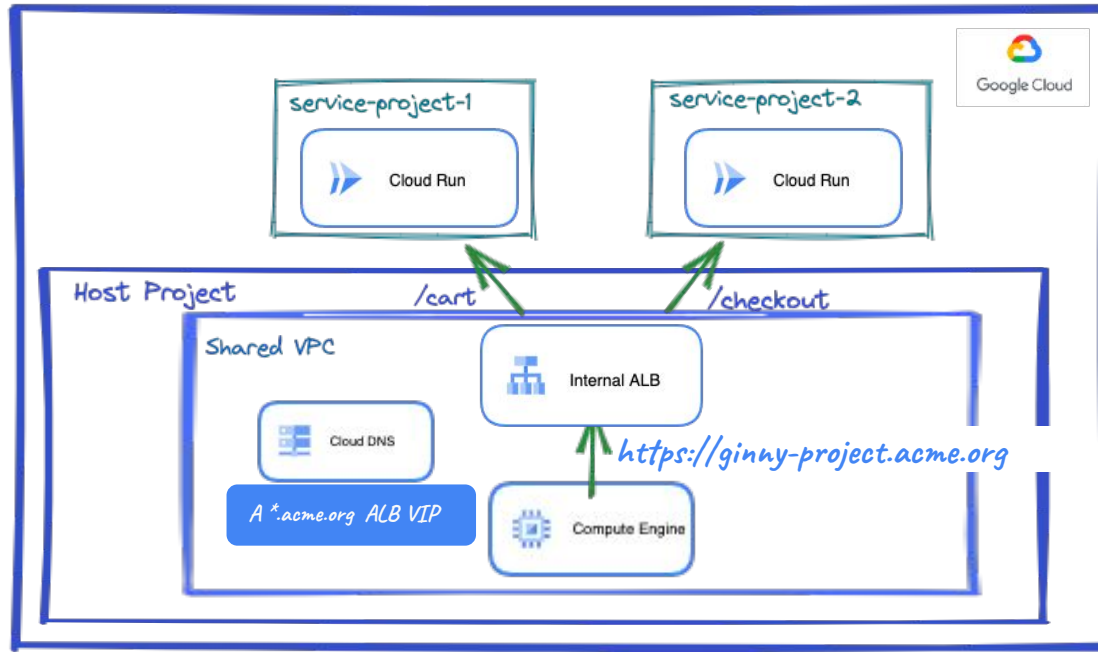




*Wow, we've added quite some pieces! I think now I can tell my manager Cloud Run may fit our corporate use cases. One more thing, could we access Cloud Run services using our own **custom domain**?*



You cannot use your own domain with PSC but you can use internal Application Load Balancer with Serverless NEG's to set it. And you could even use the same domain to access externally and internally!



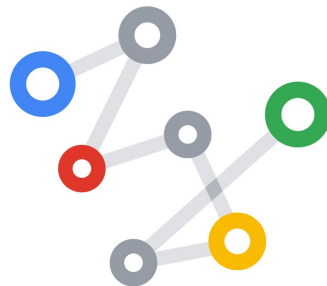
Great! We're considering using different Service projects for each team. Could the infrastructure team **centrally manage this internal ALB?**

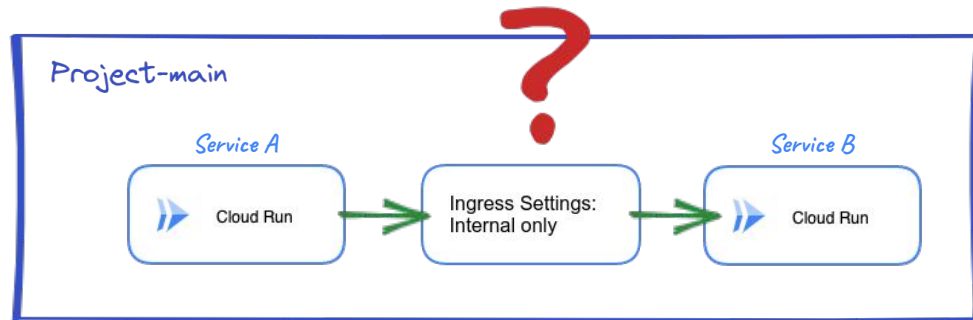
Yes, **cross-project service referencing** allows you to create Backend Services and Frontend resources in different projects.

Microservices architectures

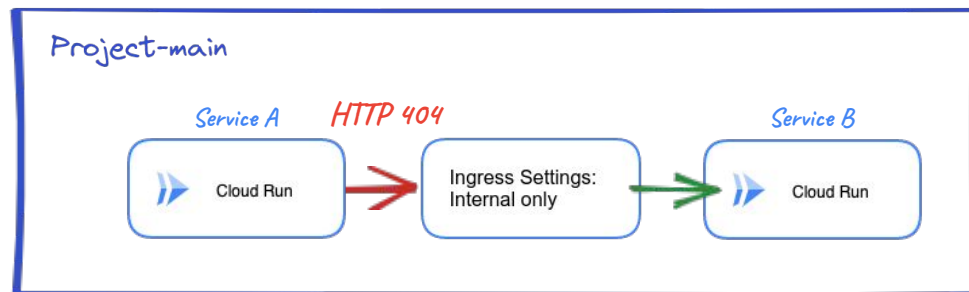
Developing Microservices applications

Terraform code

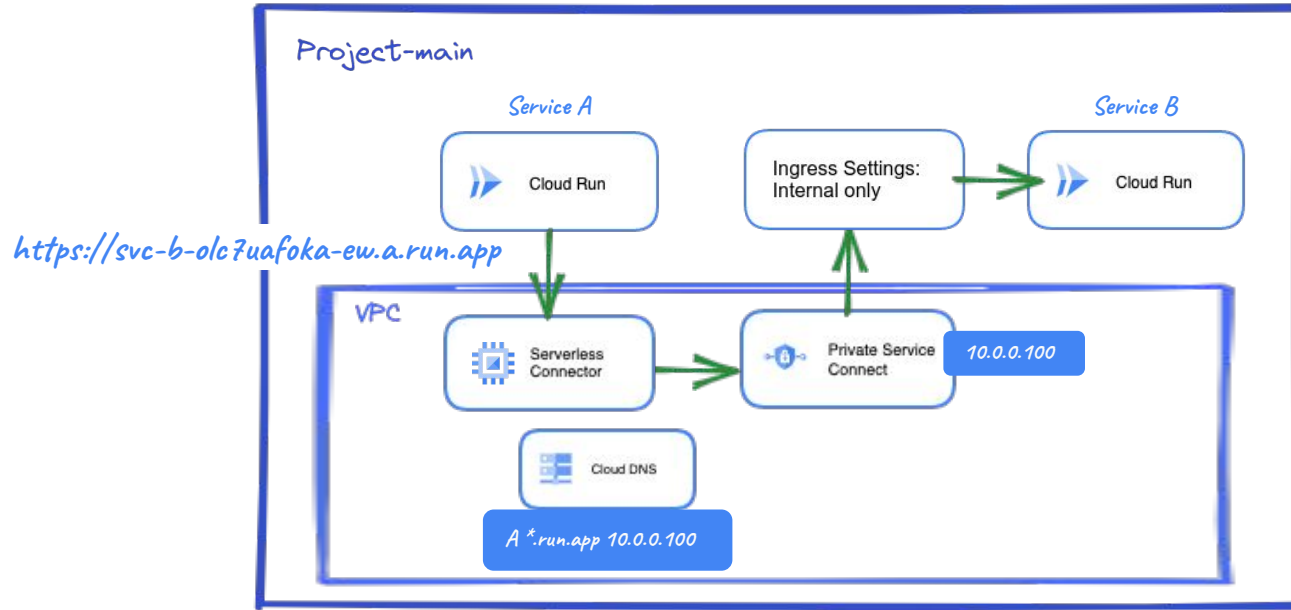




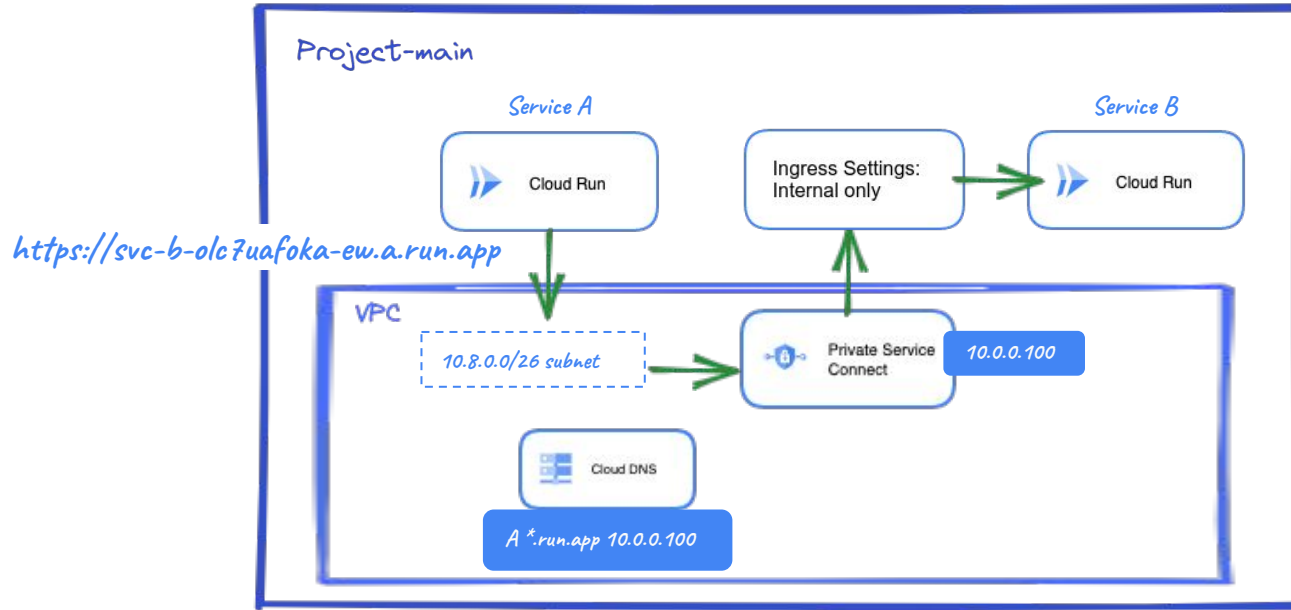
*Hey, Nick! I would like to discuss microservices. For instance, when a Cloud Run service calls another Cloud Run service within the same project, it is considered **internal**, right?*



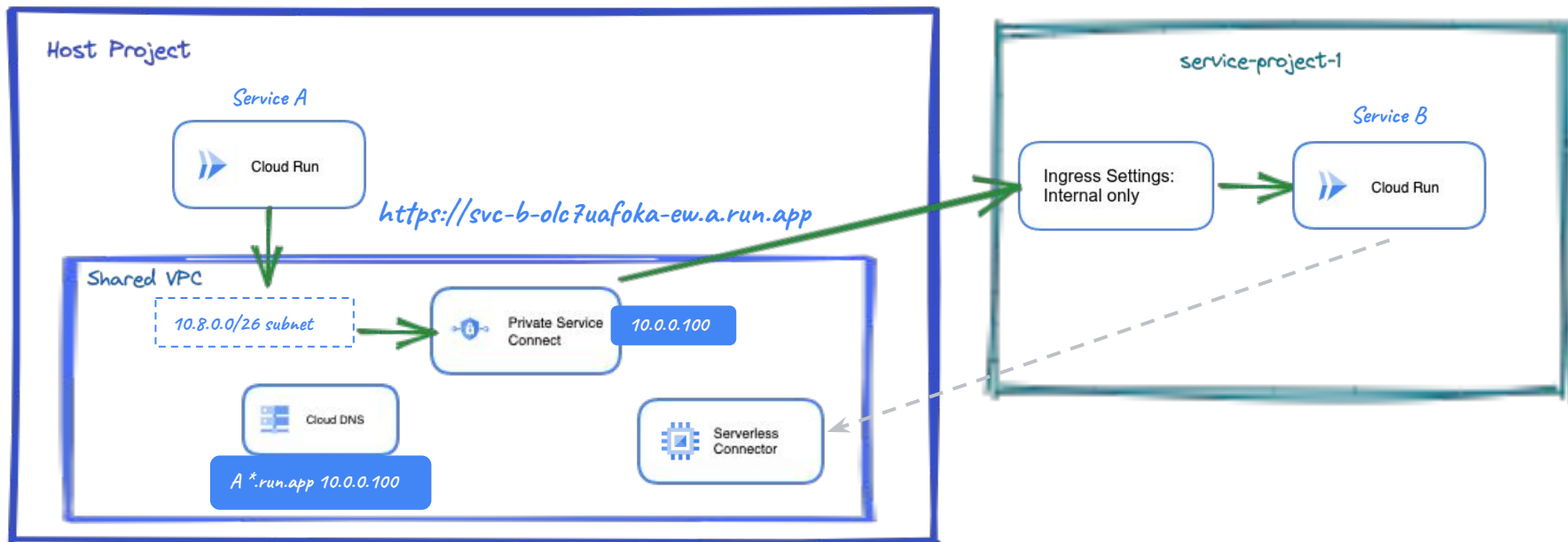
Great question! Actually this is **NOT** considered **internal**! You will get a HTTP 404 Not Found status code.



For this call to be considered *internal*, you have two options.
One is accessing through *PGA/PSC*. You will need a *Serverless VPC connector* to reach *PGA/PSC*.



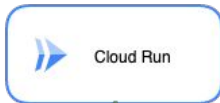
Cloud Run also supports *Direct VPC Egress*, that is, directly using IPs from a subnet in your VPC. This provides *lower latency, higher throughput, lower costs and ease of use*.



PGA/PSC also works with calls between services in projects associated to a *Shared VPC*.

Project-main

Service A



<https://svc-b-olc7uafoka-ew.a.run.app>

VPC

10.8.0.0/26 subnet



Private Service Connect

10.0.0.100



Cloud DNS

A *.run.app 10.0.0.100

Project-ginny-1

Service B

Ingress Settings:
Internal only

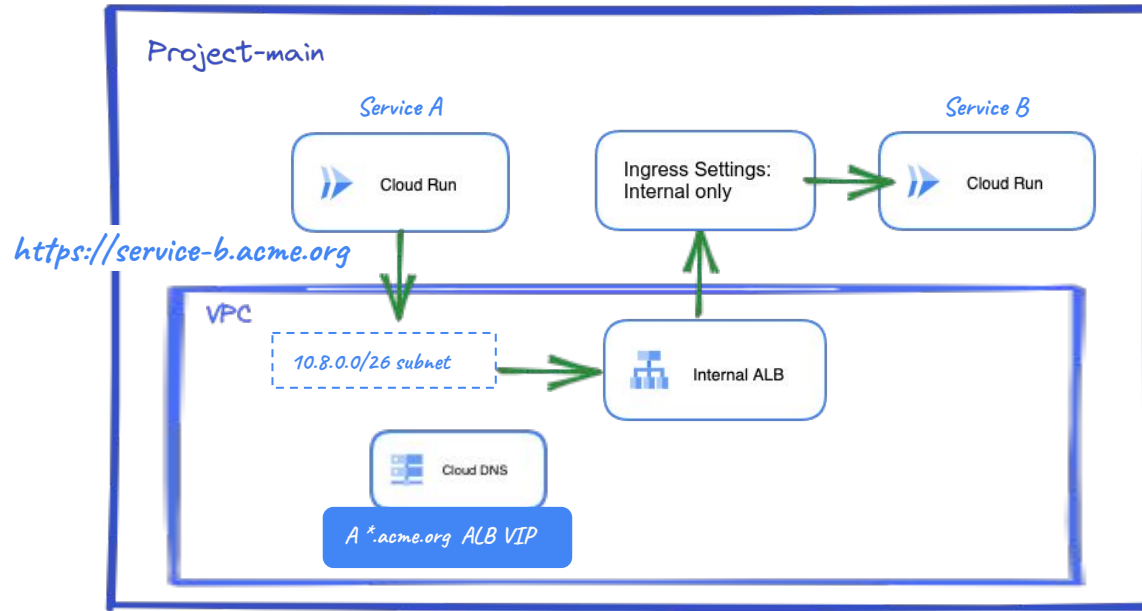


Cloud Run

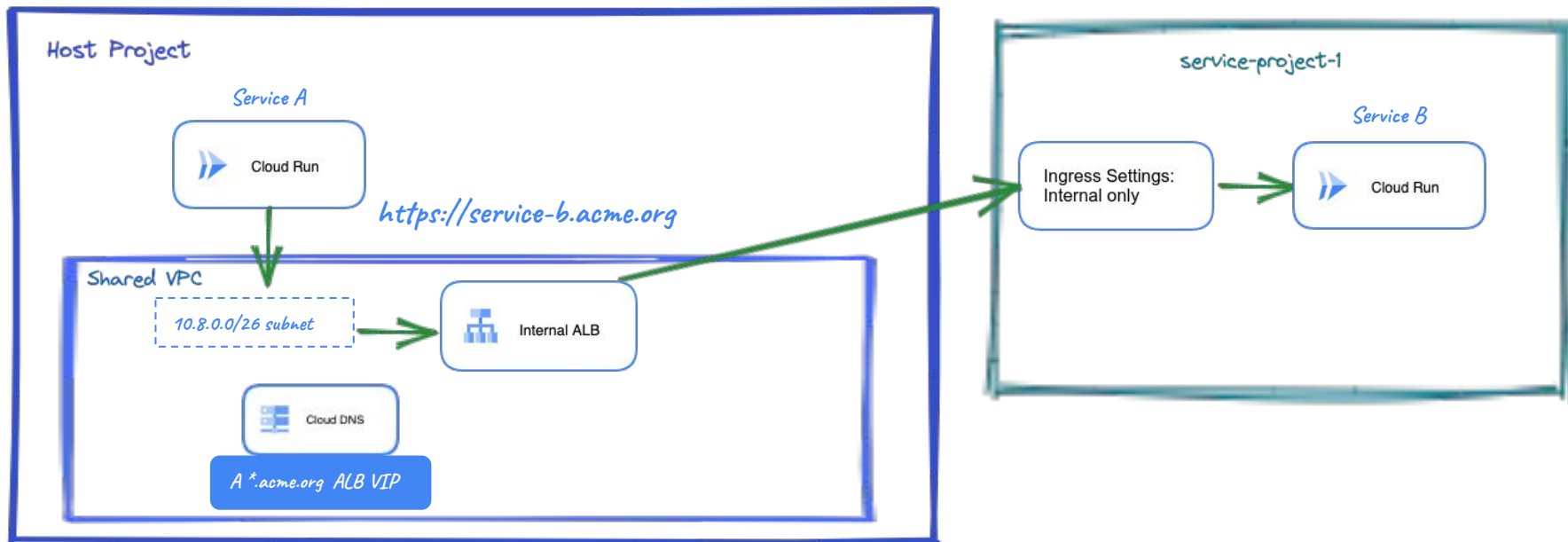
VPC-SC Perimeter



If projects are not associated to the same Shared VPC, you need to include them within the same VPC-SC perimeter.



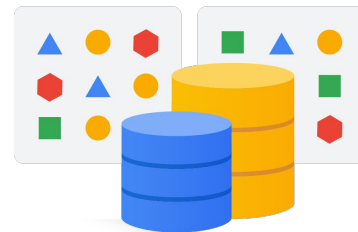
The second option is to use an *internal ALB* with Cloud Run as a *Serverless NEG* backend. The beauty of this is that you can also set your own *custom domain*.



With an internal ALB you can also access a Cloud Run service in a service project.

Access to Data

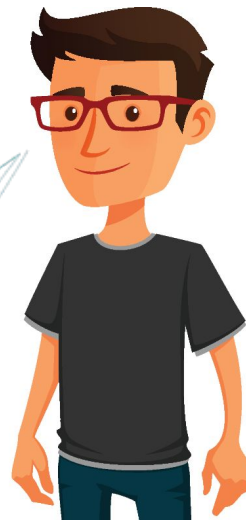
Databases, filesystems, oh my!





*Hi Nick. I'm excited about Cloud Run! As my team and I continue to develop the app, we're realizing that we will need **access to data**. How can we do this?*

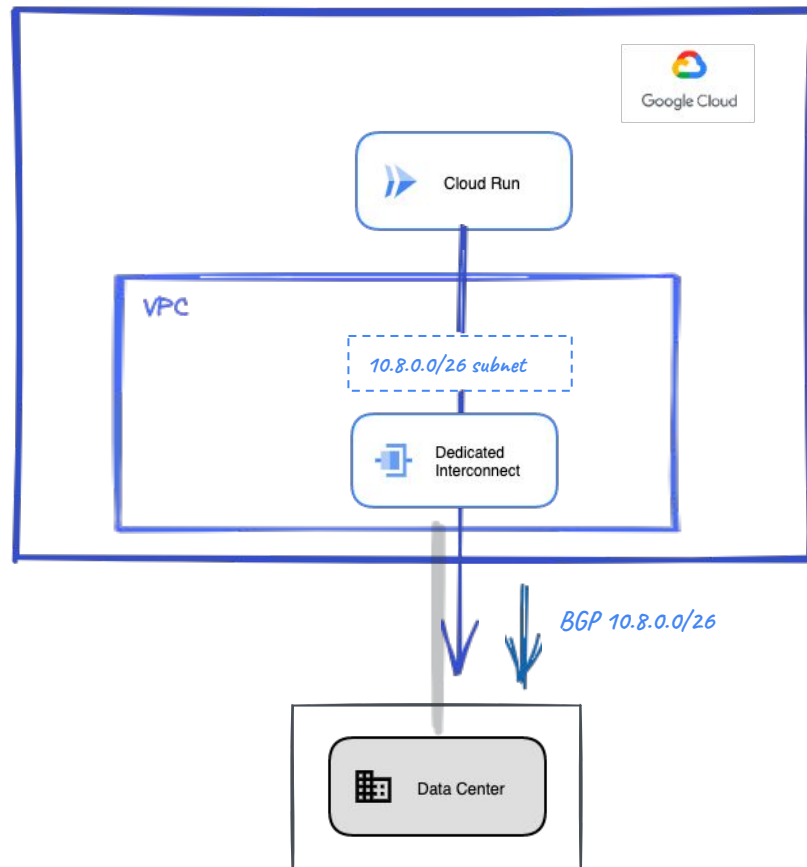
Hi Ginny. Well, it depends where your data is but let's take a look at it!

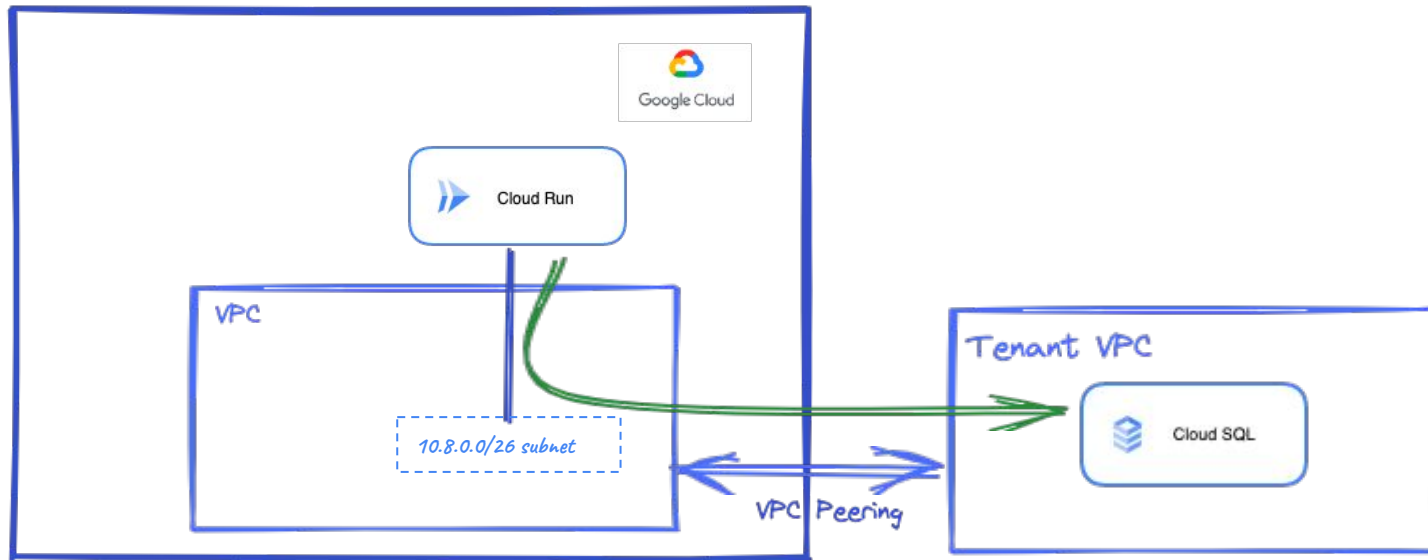


Let's assume your data is on an *onprem DB or network file systems*. Cloud Run can access it through *Direct VPC Egress*.



Make sure you *announce the subnet where Cloud Run is running over Interconnect for onprem connectivity*.

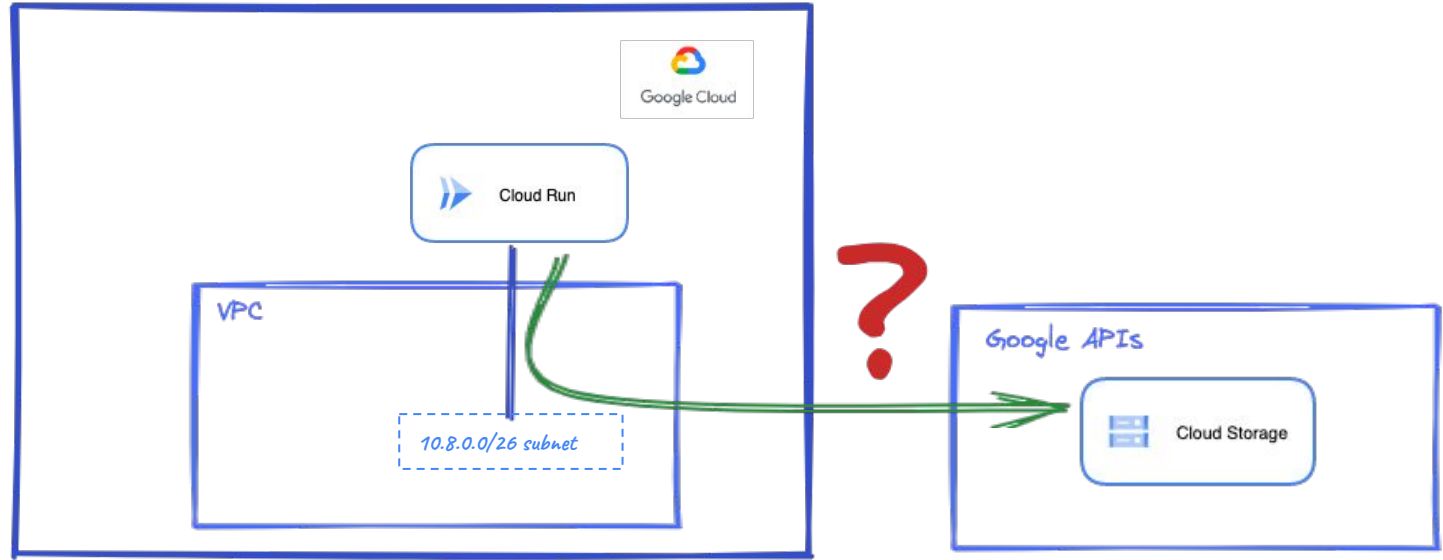




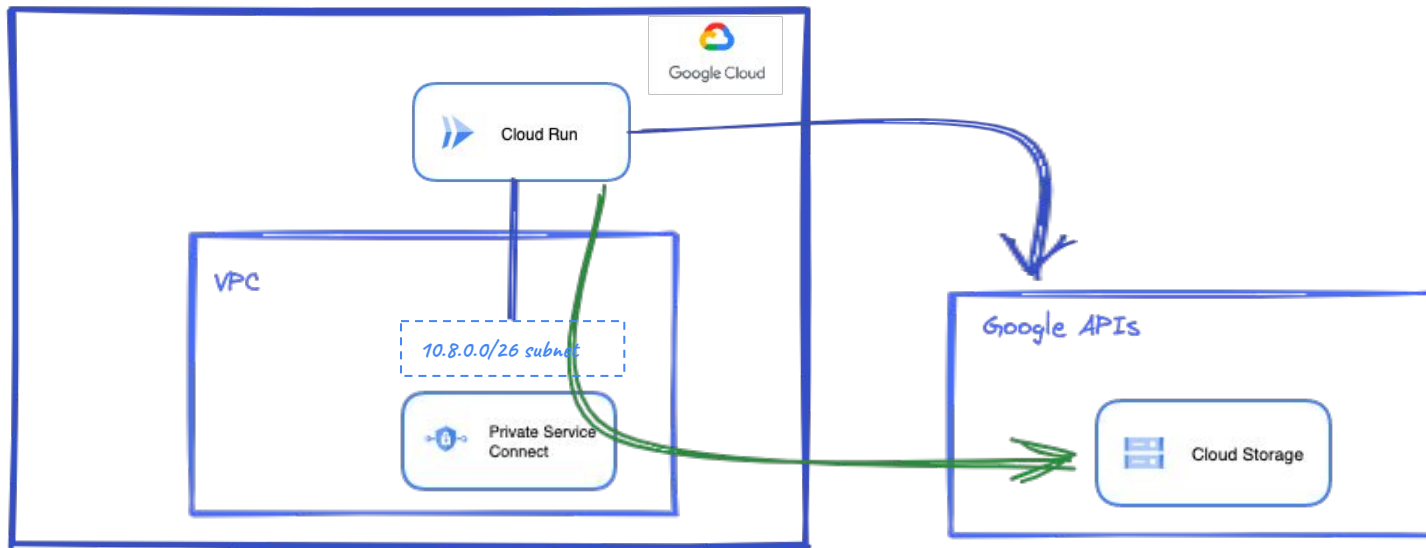
*I guess that using Direct VPC Egress, Cloud Run can also access data in a **Cloud SQL Database**, right?*

You got it, Ginny!





Ok, cool. We may need to access files too. Could we access a *Storage bucket*?



You can access Google APIs like GCS or BigQuery *directly* using *public* IP addressing, or *privately* using Direct VPC Egress if you enable *PGA/PSC*.

The SaaS world

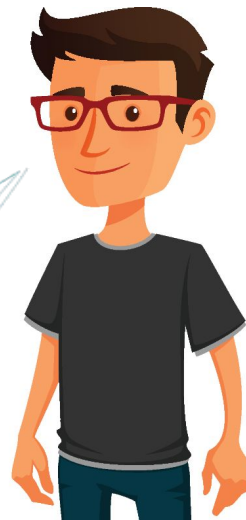
Connecting with providers

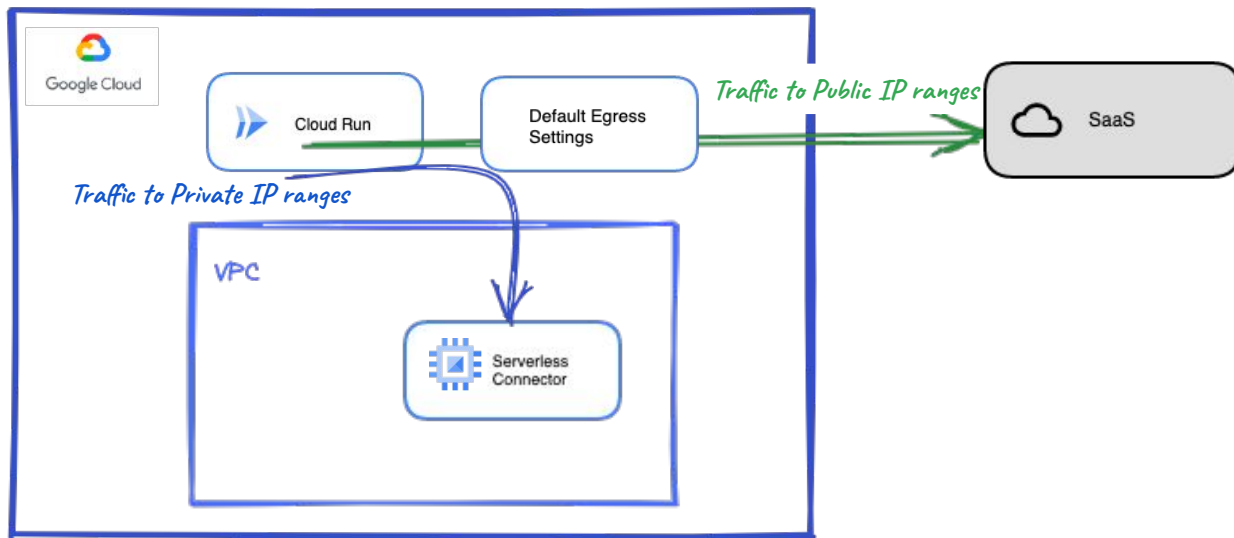




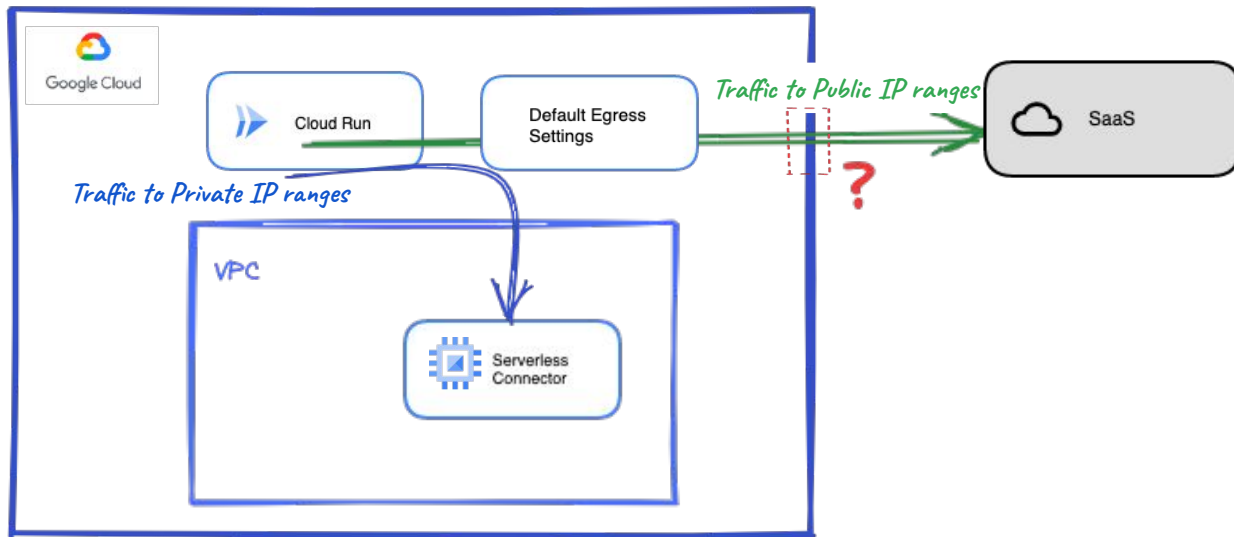
*Hi again Nick. My team is thrilled to use Cloud Run and we realize we have more requirements. E.g. we use **SaaS solutions** running in other clouds, could we **access them**?*

Sure, you can access SaaS solutions, let me show you how!

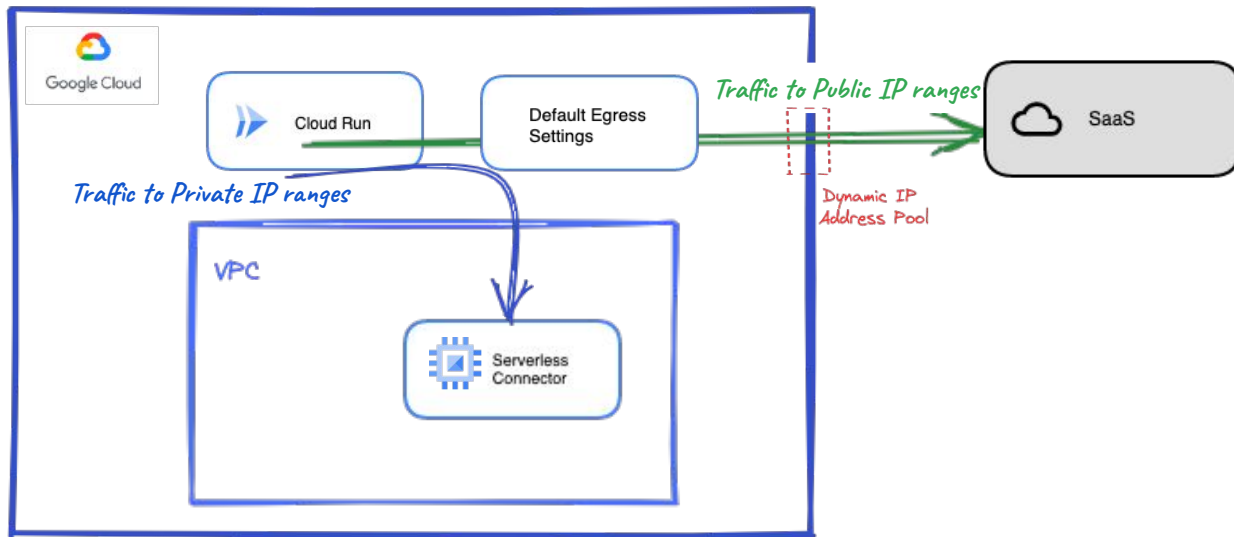




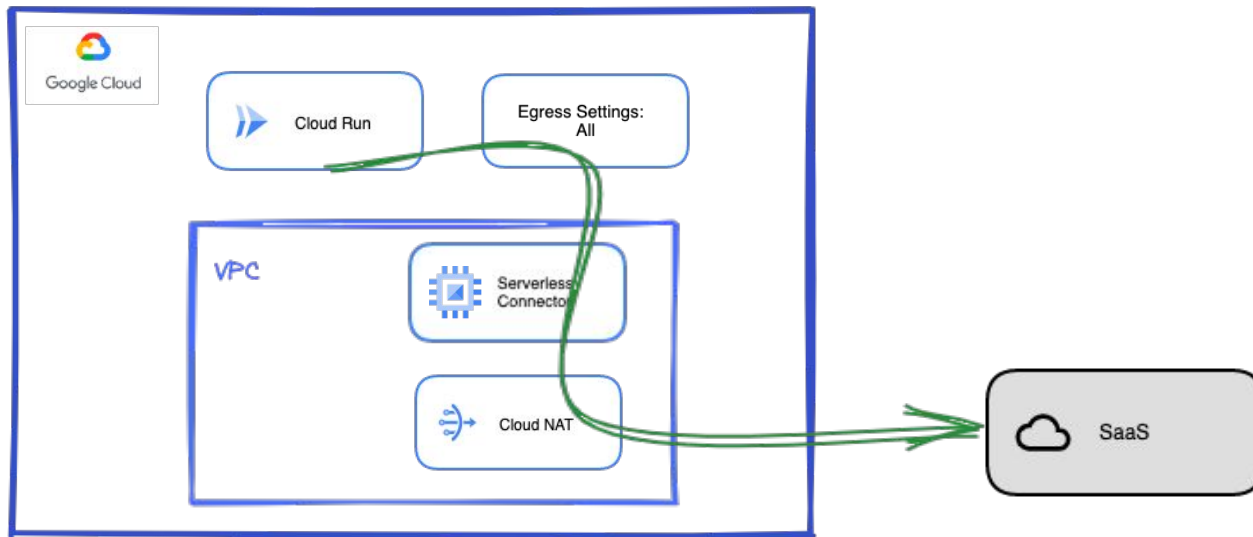
*Cloud Run services have direct Internet access. If you configure a Serverless Connector or Direct VPC Egress, default **Egress Settings** send traffic through them only to private IP ranges. The rest will reach **Internet directly**.*



Alright, so *direct access to the Internet*. But my provider has several customers and they allow access filtering by IP. *Which IP does Cloud Run use, can I assign one or...?*



No, you can't. Cloud Run IP addresses belong to a *Dynamic IP Address Pool* shared by all instances of all customers, and they can change!



*If you need a **static public IP** address, you can configure **Egress Settings** to send **all outgoing traffic** through a connector and set up **Cloud NAT** (Direct VPC Egress doesn't support it for now).*

Security Controls

Applying Security in the corporate environment

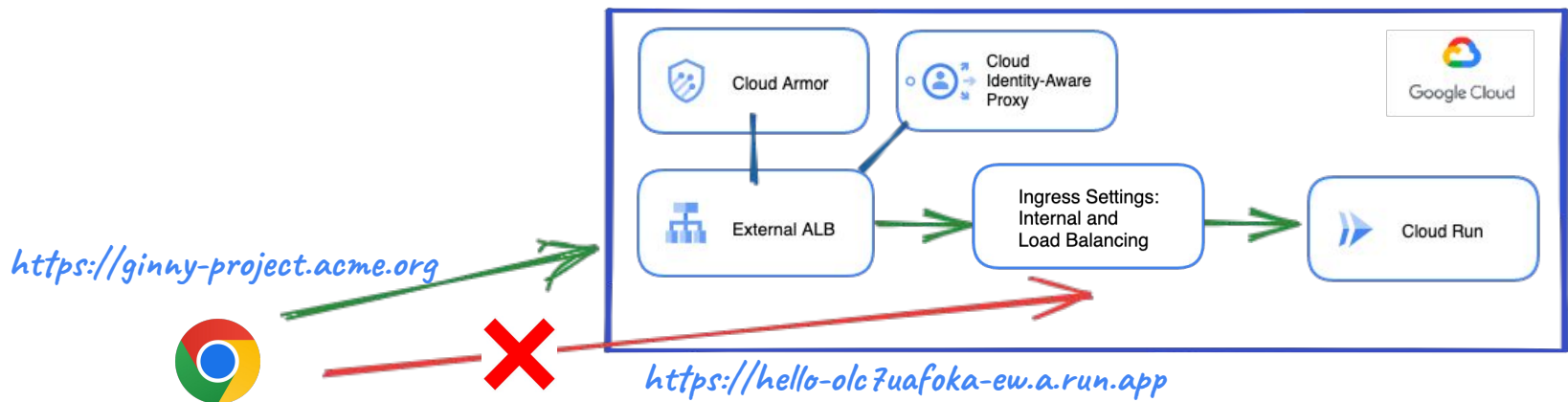




*Hi, Nick! You know? I showed Cloud Run capabilities to the **Security Team** and they are not sure it meets their security policies. Could we review Cloud Run security controls?*

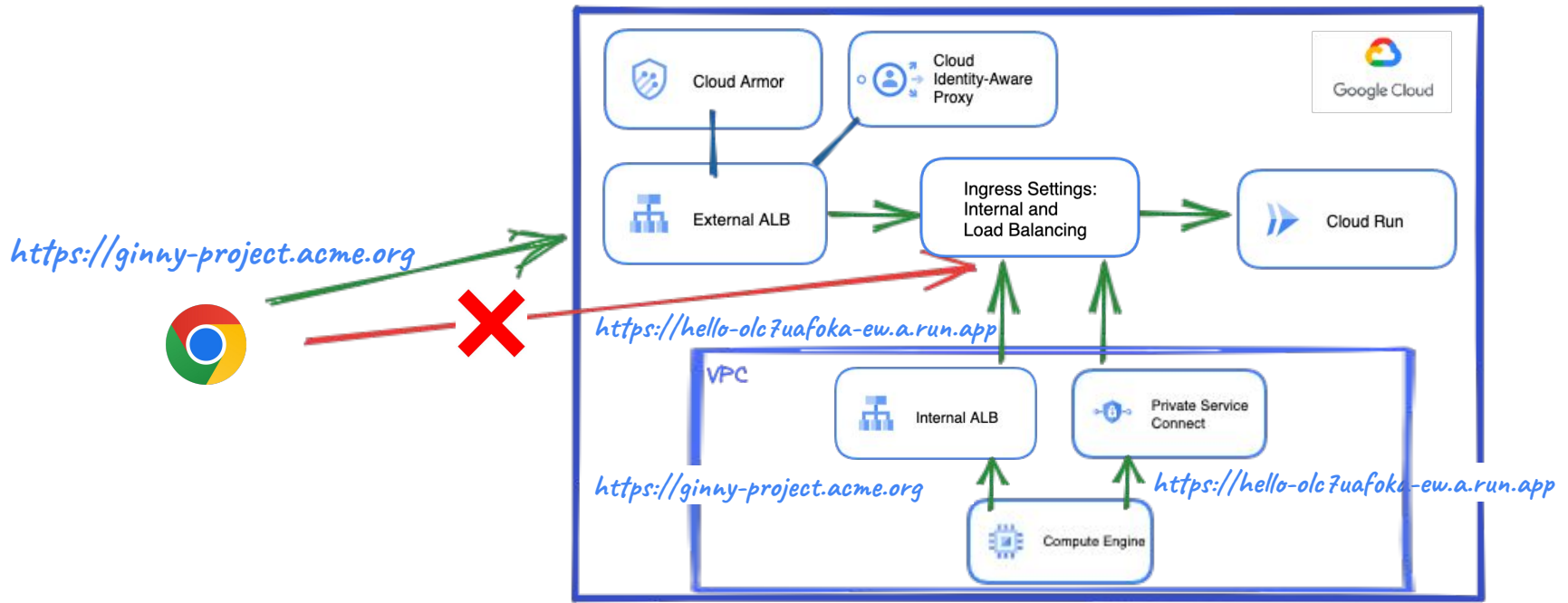
Absolutely, let's review different security controls!



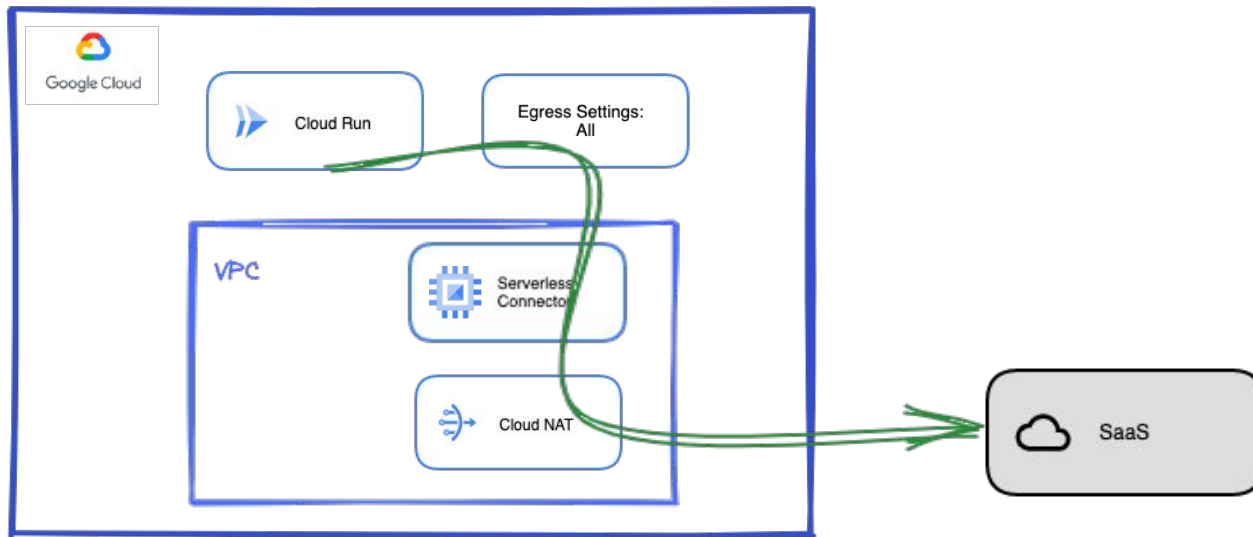


*In previous chapters we mentioned that a service can be exposed through an external Load Balancer, secured with **Ingress Settings** to Internal and Load Balancing, **Cloud Armor** and **IAP**.*





Regardless of the *Ingress Settings*, the service can also be reached through PGA/PSC or through an internal ALB.



You can configure *Egress Settings* to send *all outgoing traffic* through a *VPC*, apply *Firewall rules*, and set up *Cloud NAT* to get a *static Public IP* address to access *Internet*.



*Ok great, I got it! You told me about Networking specific security controls. But what about **authentication and authorization**? Can we also build them on top?*

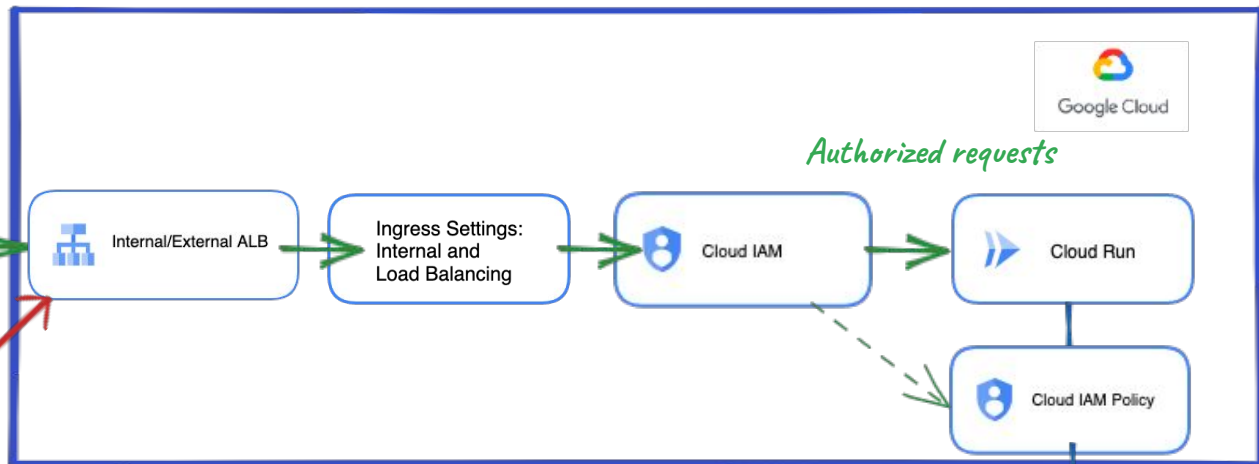
*Great point! Let's review what **Access Control** methods you have available.*



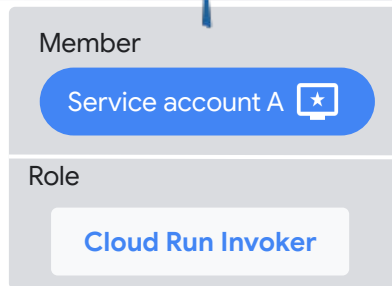
<https://ginny-project.acme.org>

Service account A 

Service account B 



<https://ginny-project.acme.org>

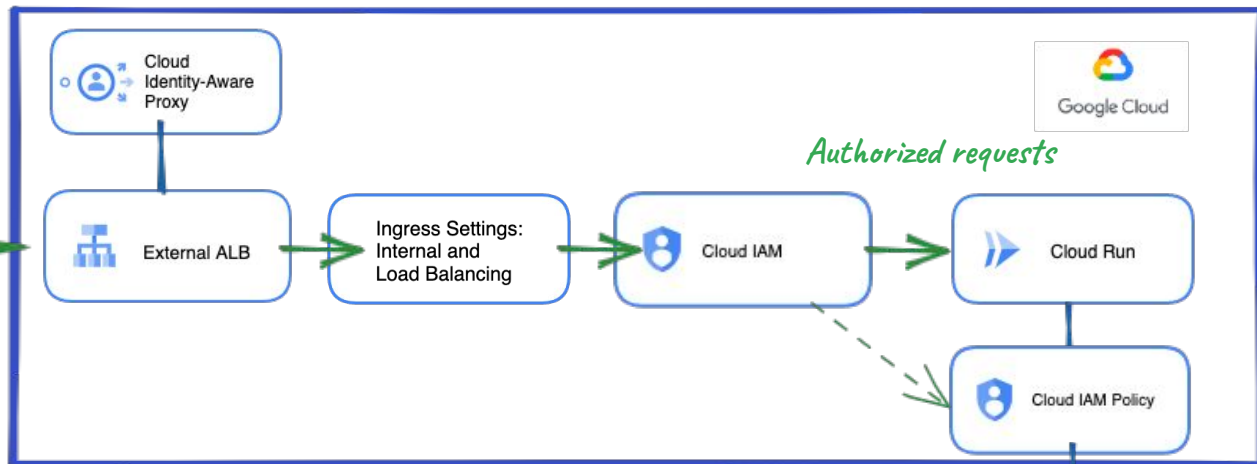


Cloud IAM protects Cloud Run service, checking every incoming request. Only requests authorized with Cloud Run Invoker role are accepted based on its IAM policy.

<https://ginny-project.acme.org>



End-user account



Member

IAP service agent



Role

Cloud Run Invoker

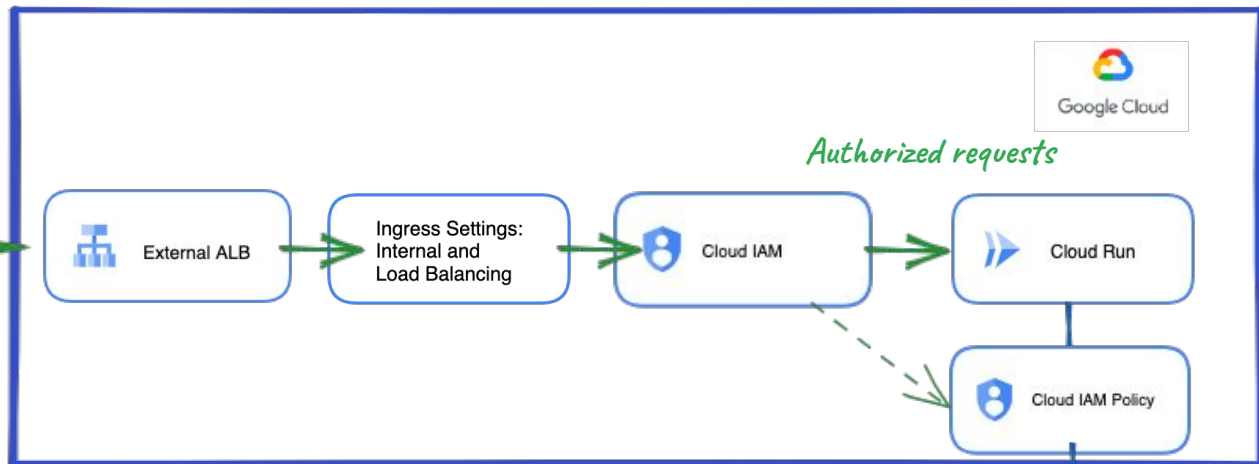
And for *internal Web Apps* with external access, how do we manage authentication?

You can authenticate users via *IAP*, and grant the invoker role to IAP's service agent.

<https://ginny-project.acme.org>



Anonymous



Member

allUsers



Role

Cloud Run Invoker

Alright, but what about *public APIs*, do you also need to authenticate?

No, for public APIs or websites, you can assign the *Cloud Run Invoker* role to *allUsers*.

All together

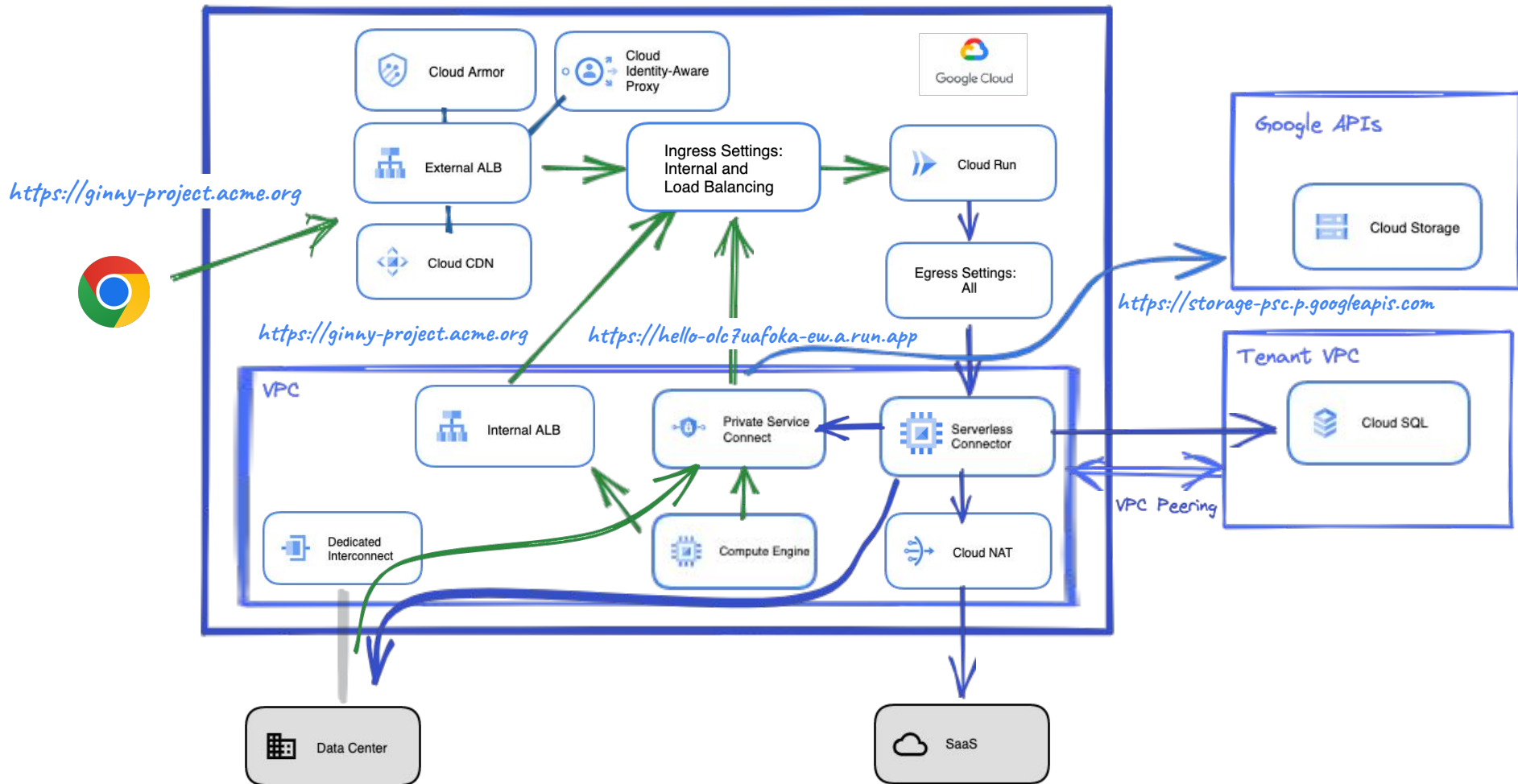




*Thanks a lot, Nick! I think now I have all the pieces in the puzzle to use Cloud Run in my company. I've **summarized** it here to see if I got it right!*

Hey, this diagram is great! It has all the main elements we have discussed.





THANK YOU

