Module - Networking

Overview

Resources in GCP projects are split across VPCs (Virtual Private Clouds)

VPCs support various networking features such as DNS, Load balancing, direct peering, VPNs and Firewalls

GCP has 3rd party partnerships for CDNs and ISPs

Several complex forms of load balancing are available at different levels of the OSI stack

VPC

Virtual Private Cloud

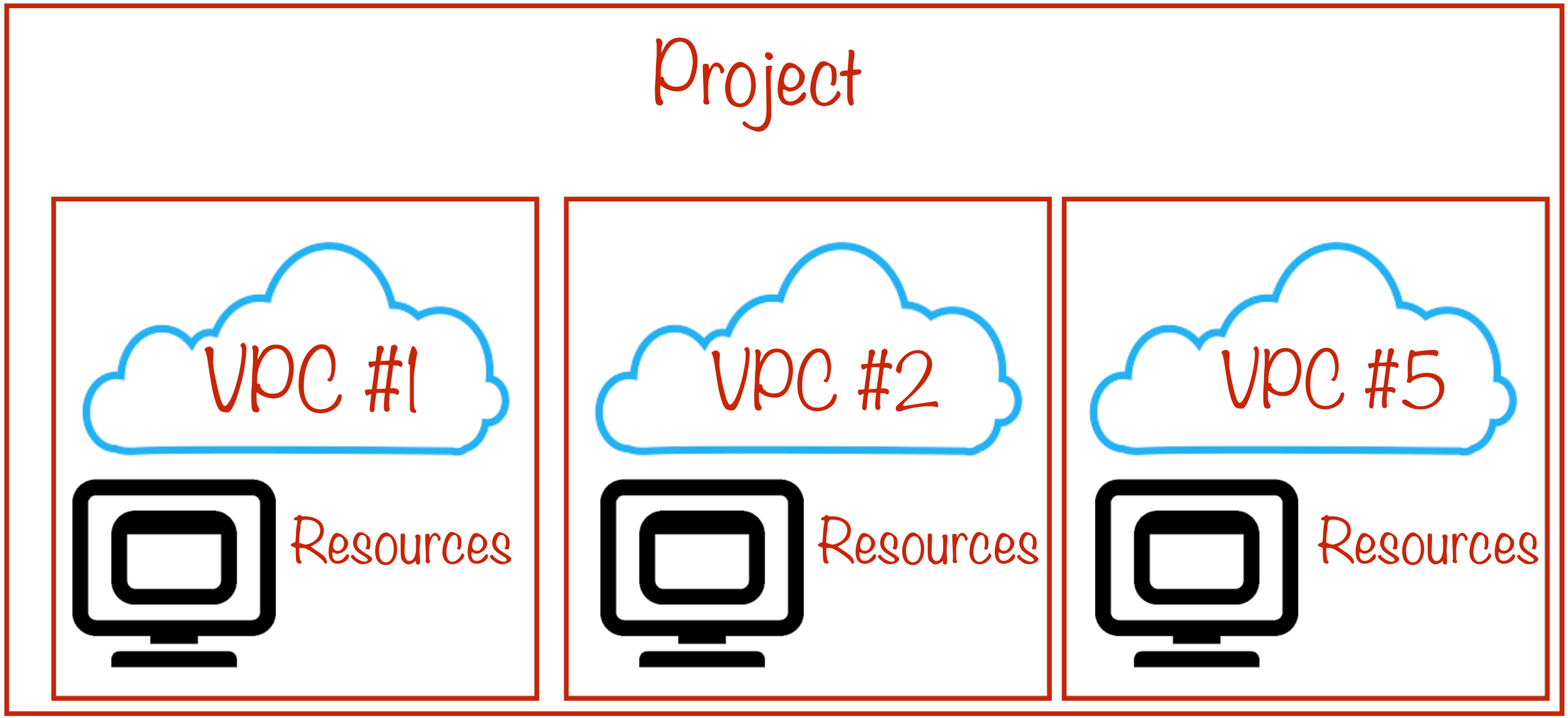
A global private isolated virtual network partition that provides managed networking functionality

(www.docker.com)

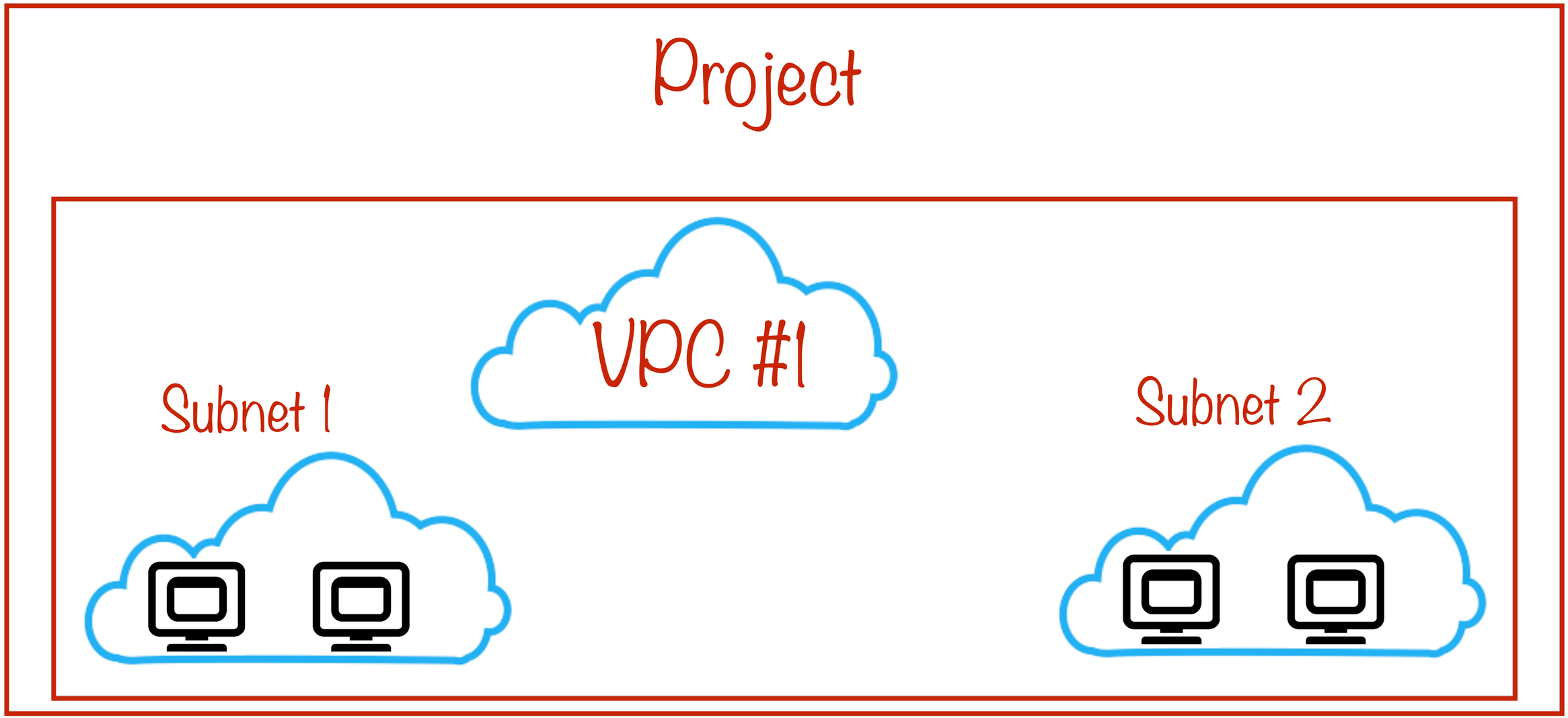
VPC

- Global
  - Resources from anywhere
- Multi-tenancy
  - VPC shared across GCP projects
- Private and secure
  - IAM, firewall rules

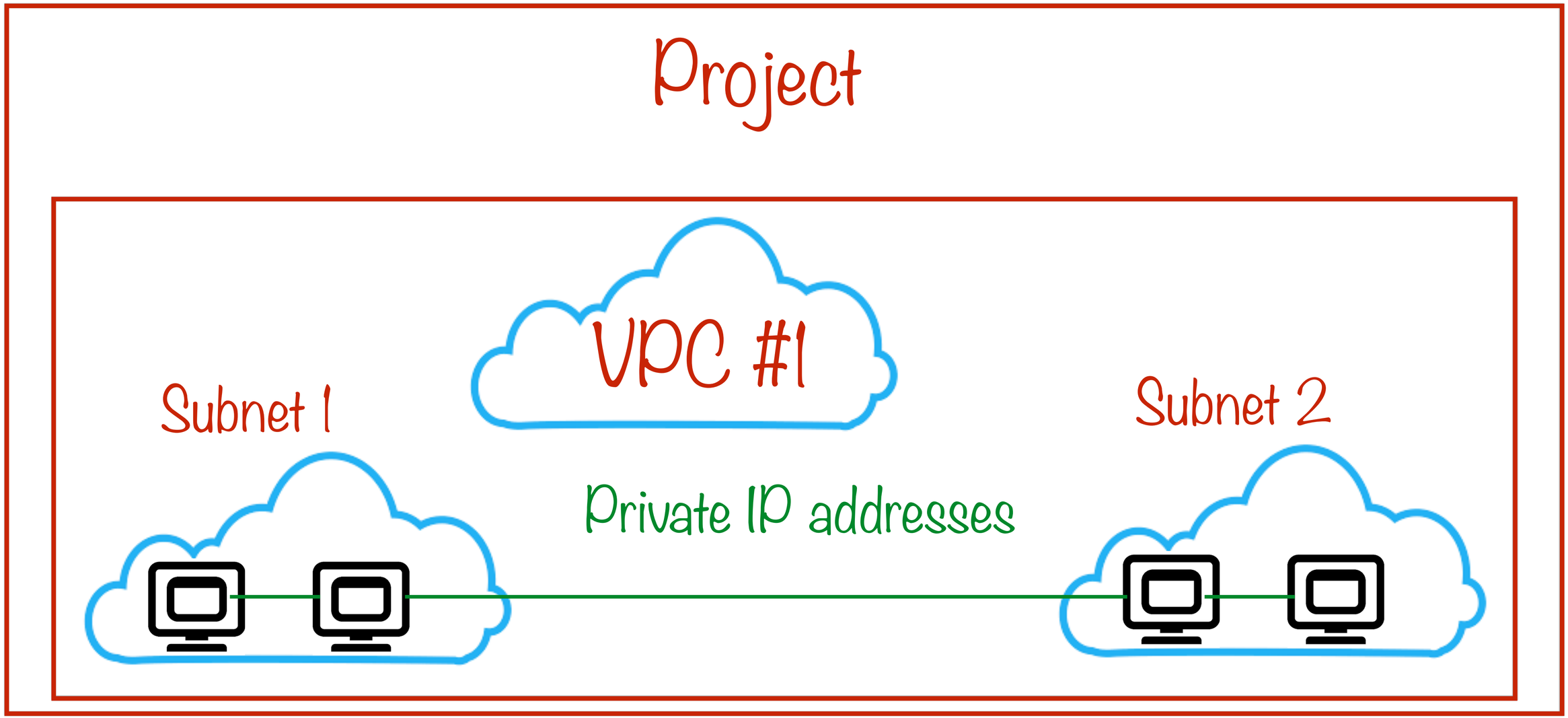
Projects and VPCs



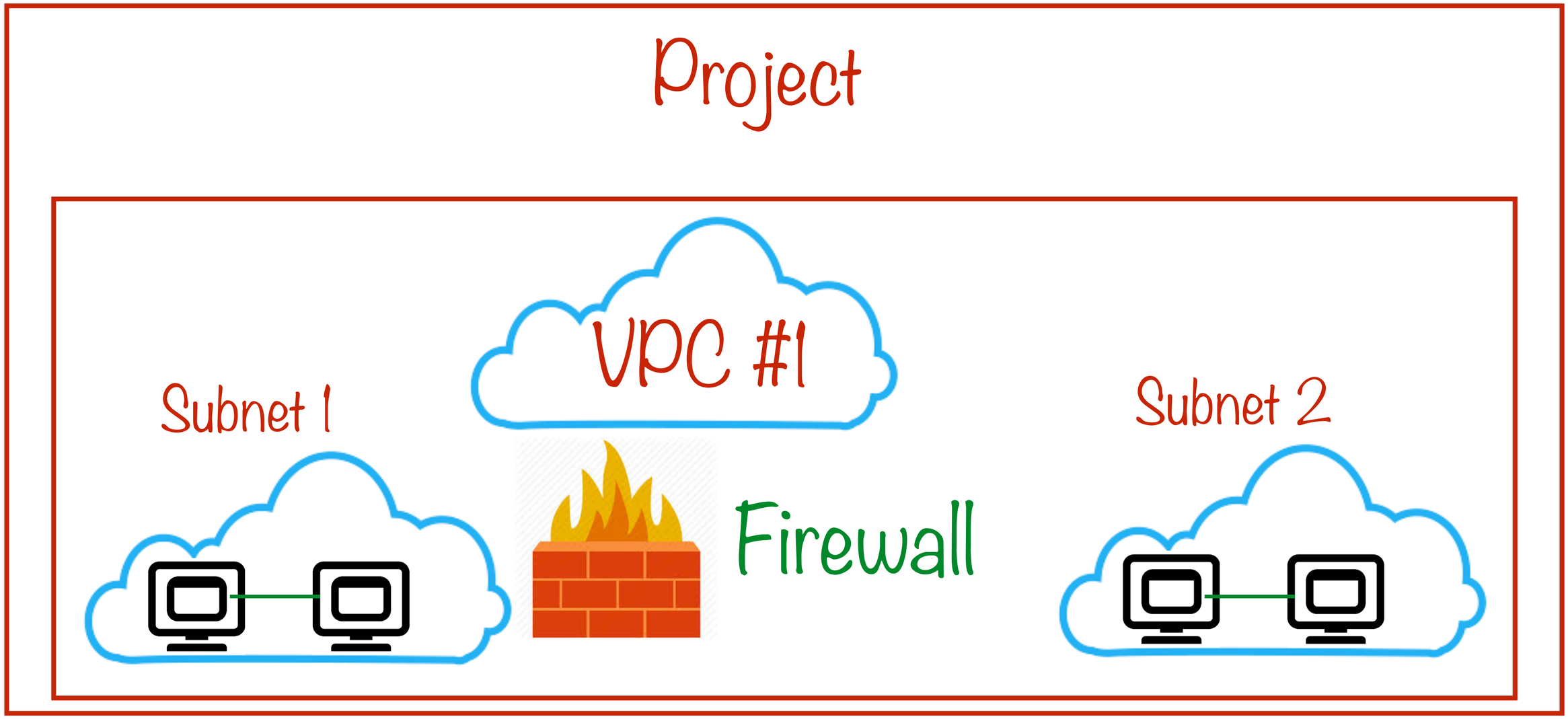
Projects and VPCs



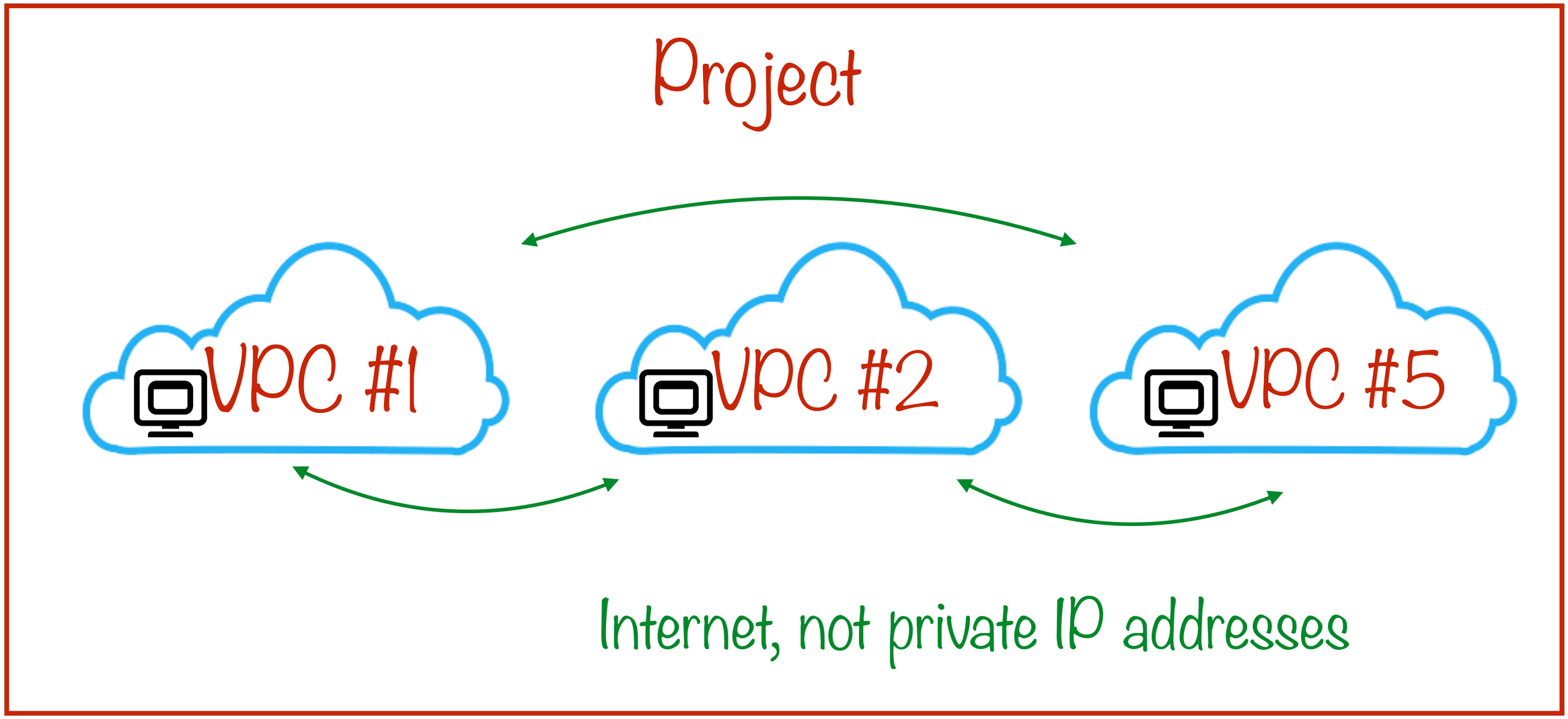
Projects and VPCs



Projects and VPCs



Projects and VPCs



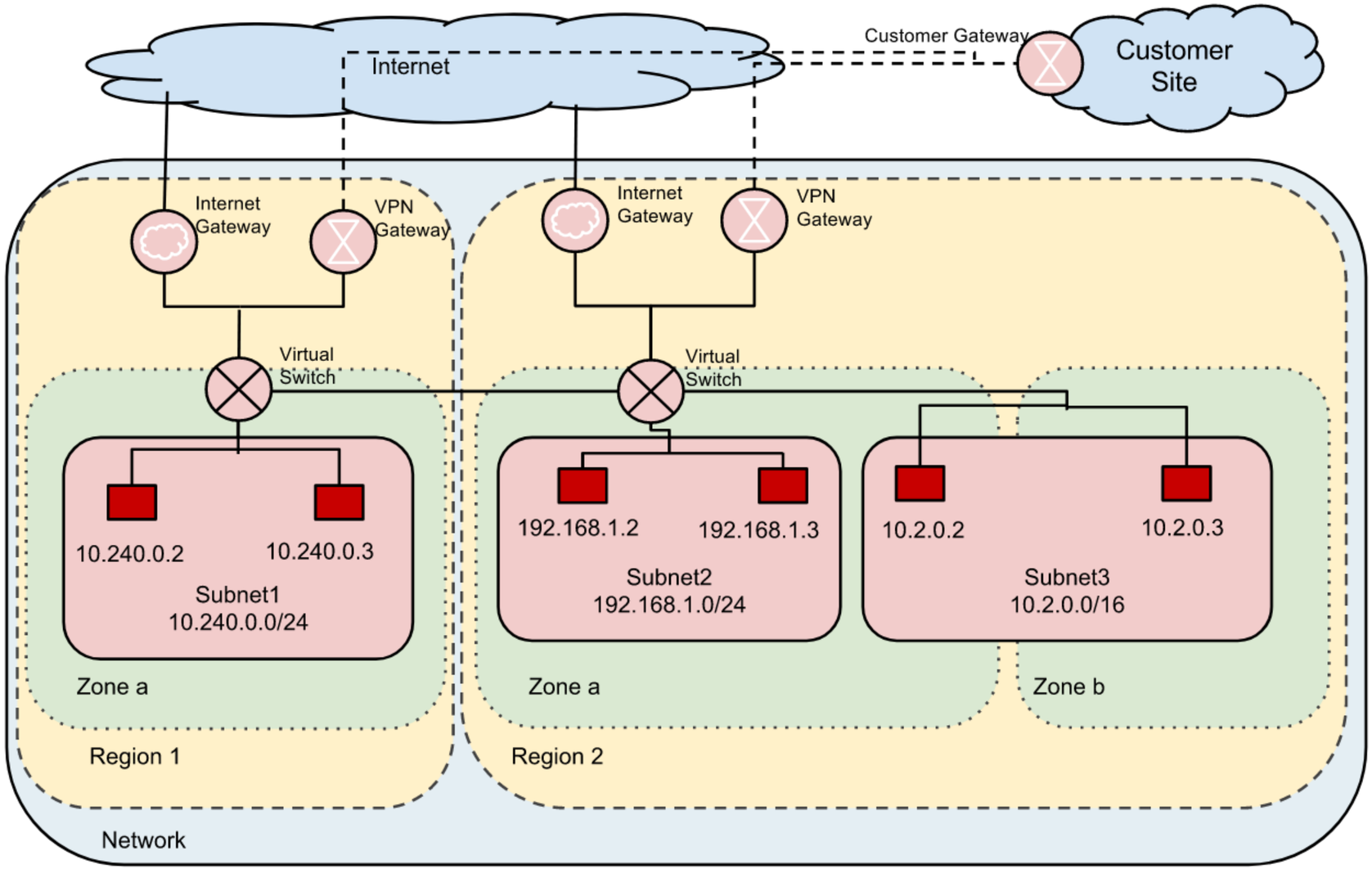
Resources and Rules

- Two types: auto-mode, custom-mode

- VPC have routes, firewall rules

- Nax 7000 VMs per VPC

IP Addresses



IP Addresses

- Can be assigned to resources e.g. VMs

- Each VM has internal IP address

- Can also have an external IP address

Internal vs. External

- Internal: Use within VPC

- External: Use across VPCs or across projects

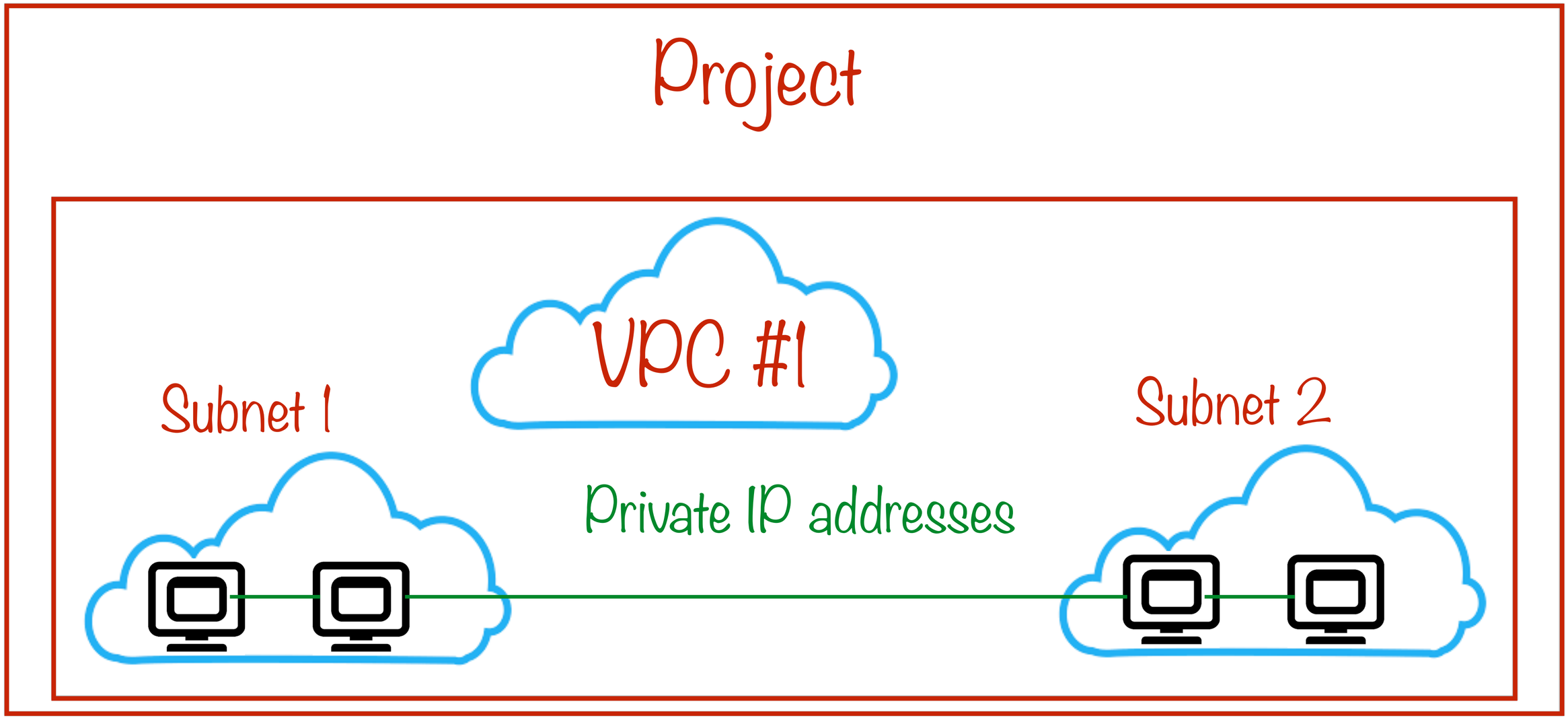
- Internal IPs - always ephemeral

- External IPs - ephemeral or static

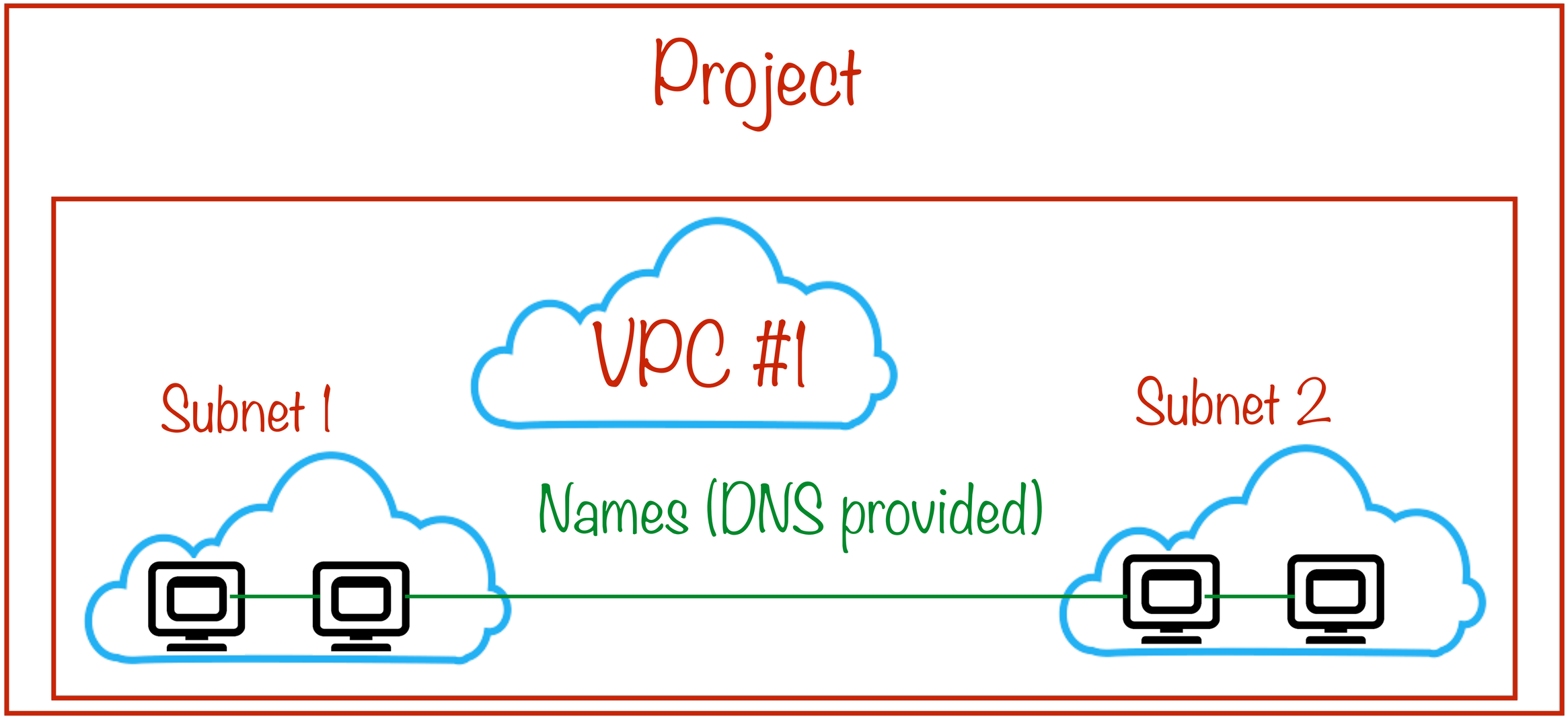
Ephemeral vs Static

- Ephemeral: Only till VM restart or start/stop
- Static: Available till explicitly released
- Internal IPs - always ephemeral
- External IPs - ephemeral or static
- Static external IPs - can global or regional

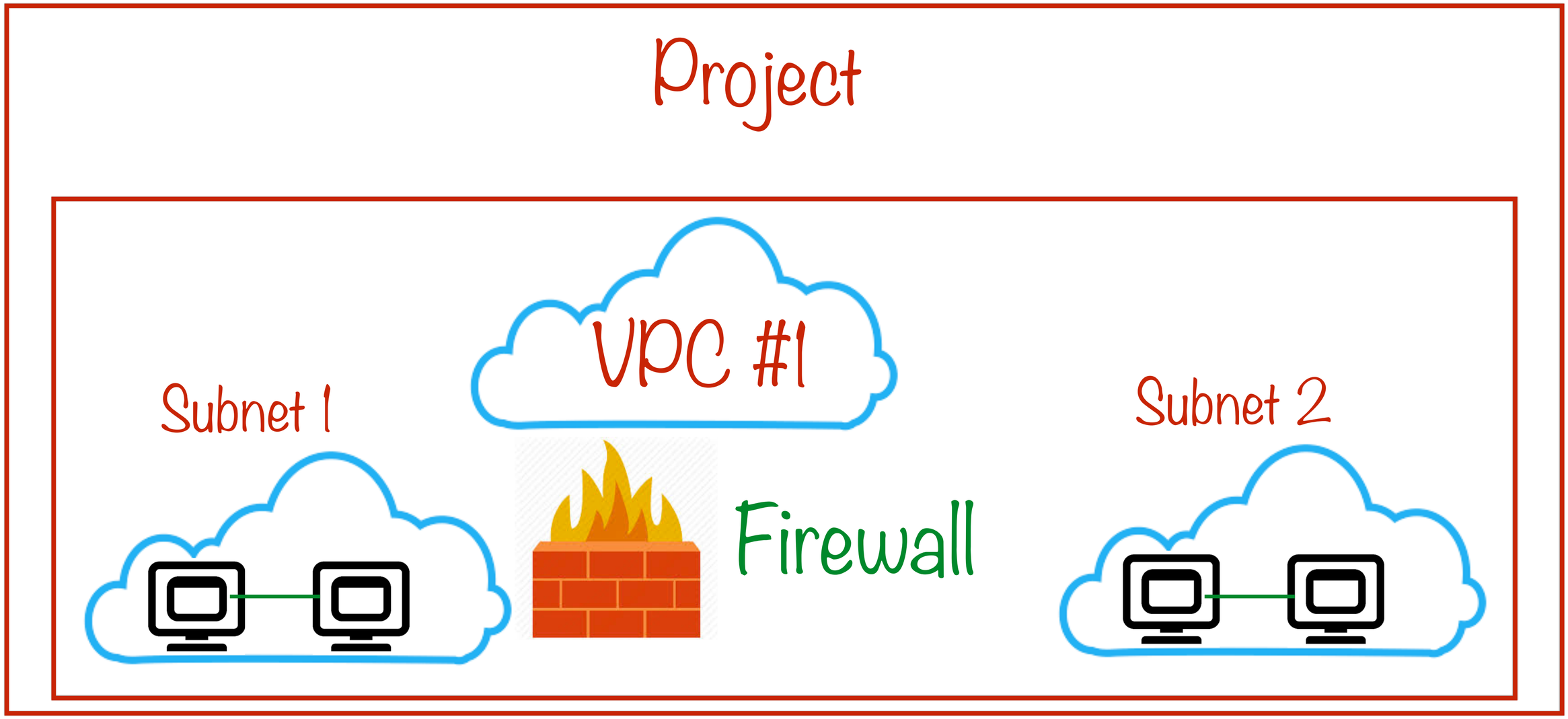
Projects and VPCs



Projects and VPCs



Projects and VPCs



Firewalls

- Each VPC network has its own firewall controlling access to the instances

- All traffic to instances, even from other instances, is blocked by the firewall unless firewall rules are created to allow it

- The exception is the default VPC network that is created automatically with each project

- For example, you can create a firewall rule that allows all traffic through port 80 to all instances, or only allows traffic from one specific IP or IP range to one specific instance

Firewall Rules

- Action: allow or deny
- Direction: ingress or egress
- Source and Destination IPs
- Protocol and port
- Specific instance names
- Priorities and tiebreakers

Routes

Where to send packets destined for an IP?

The answer lies in a route

Eg all internet-bound packets to proxy server first

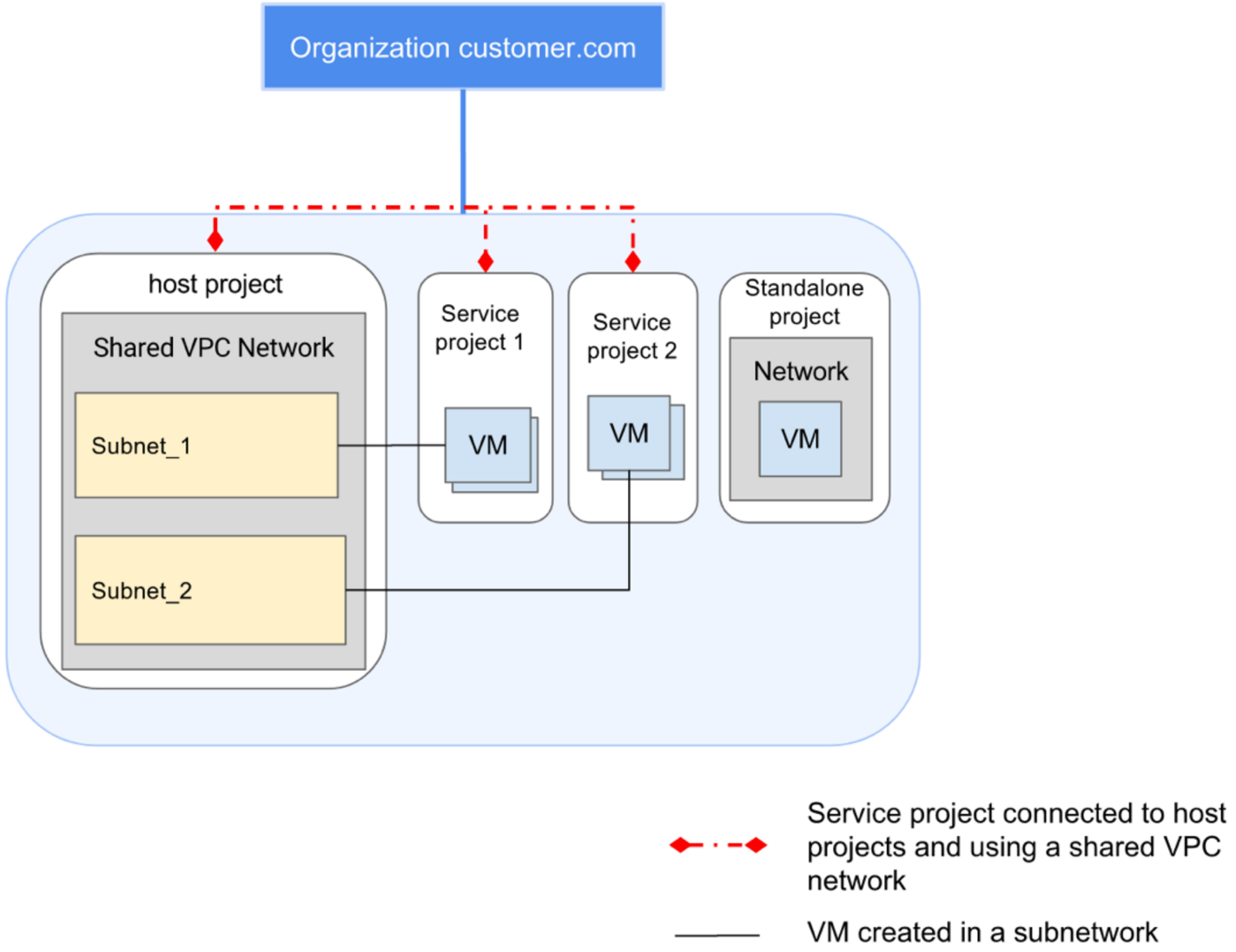
Shared VPC

Used to be called XPN (Cross-Project Networking)

So far - one project, multiple VPCs

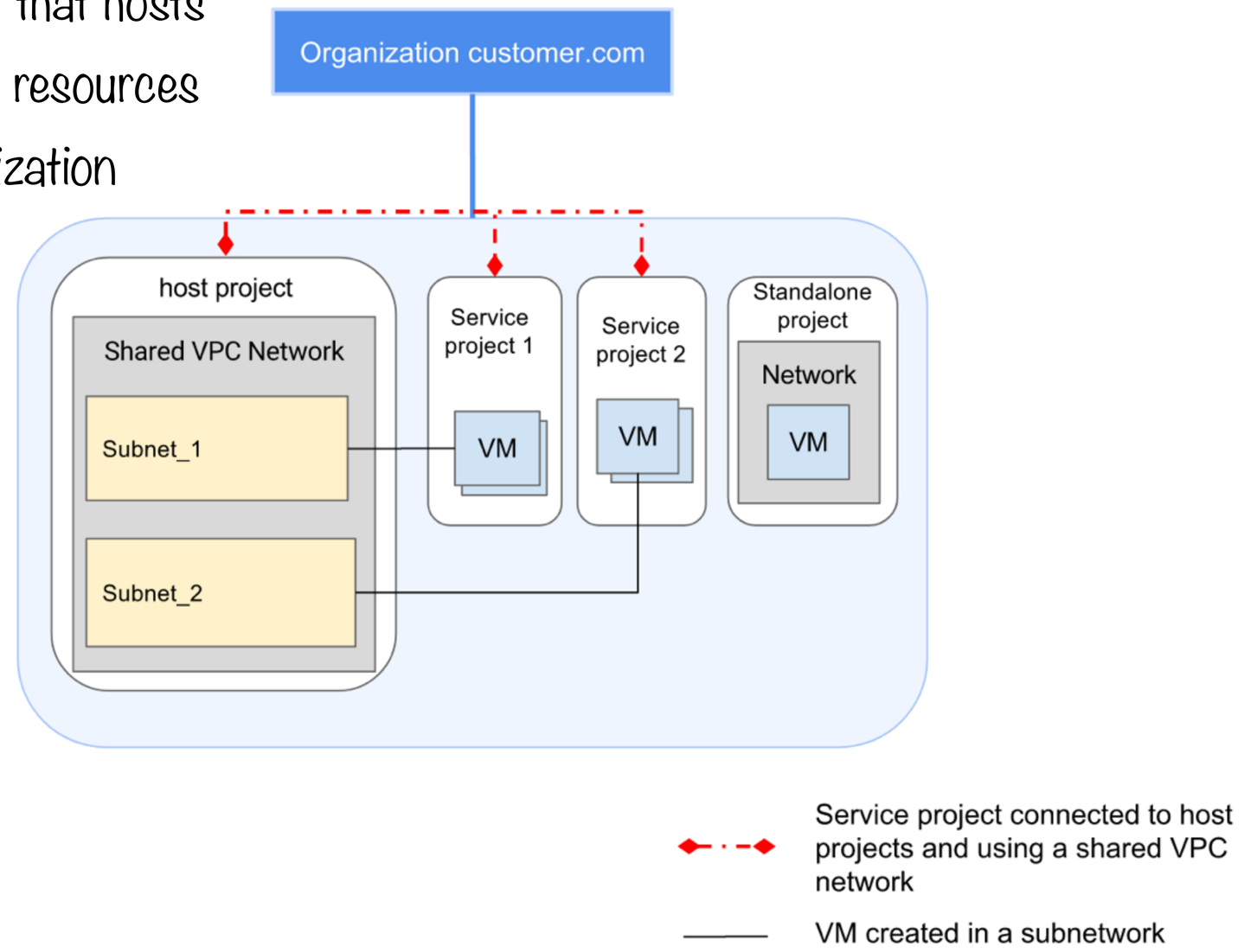
Shared VPC - multiple projects, one VPC

Shared VPC

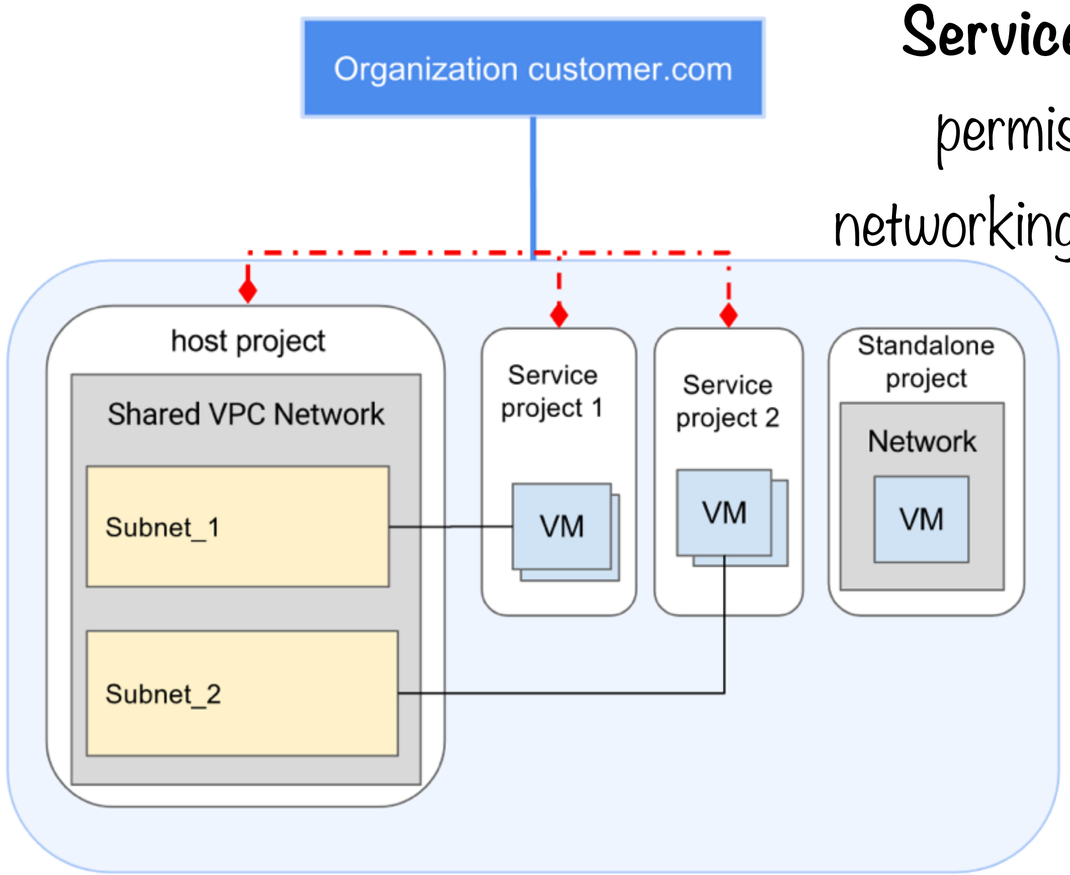

Organization customer.com
host project
Shared VPC Network
Subnet_1
Subnet_2
Service project 1
VM
Service project 2
VM
Standalone project
Network
VM
Service project connected to host projects and using a shared VPC network
VM created in a subnetwork

Shared VPC

Host Project — Project that hosts sharable VPC networking resources within a Cloud Organization



Shared VPC



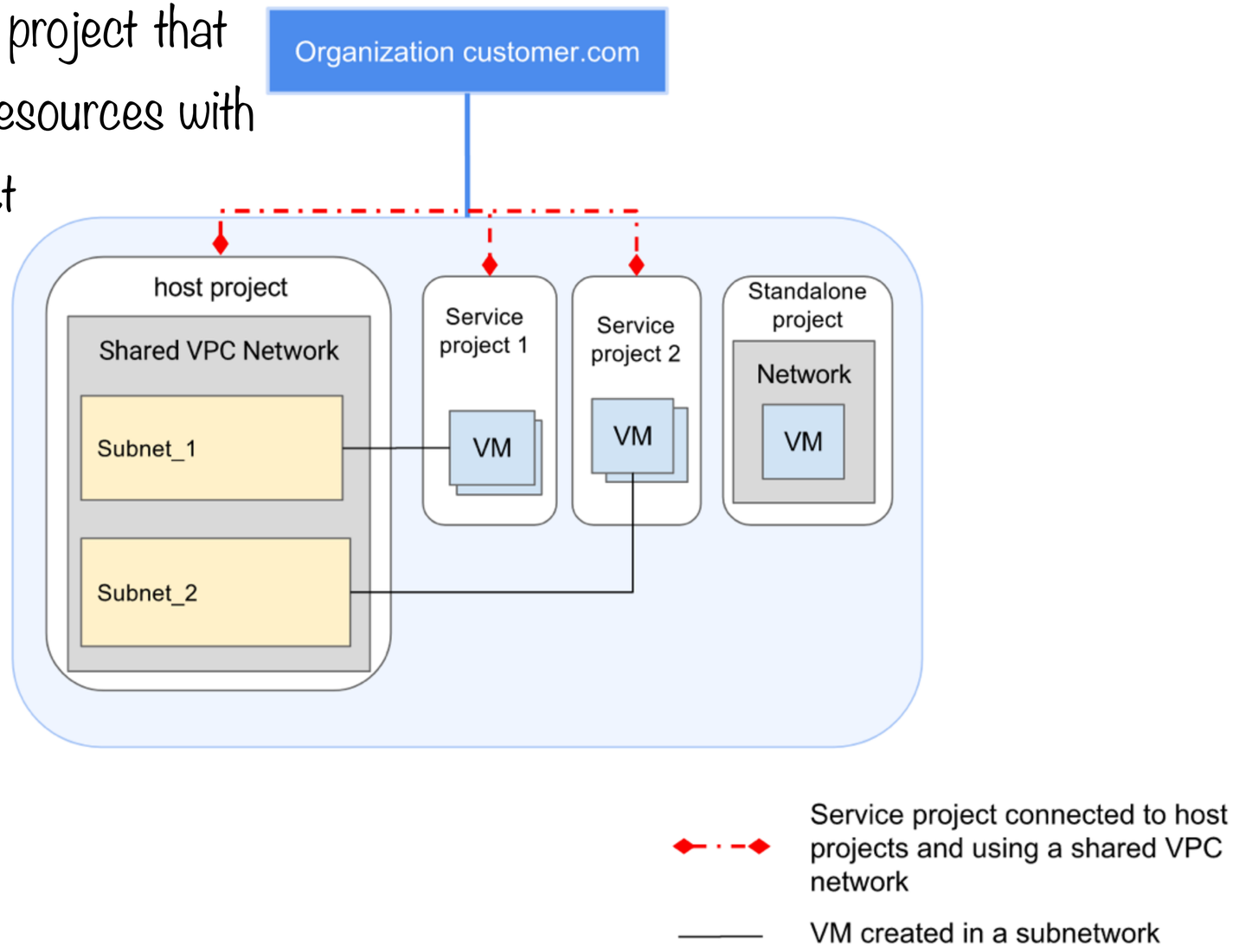
Service project — Project that has permission to use the shared VPC networking resources from the host project

Service project connected to host projects and using a shared VPC network

VM created in a subnetwork

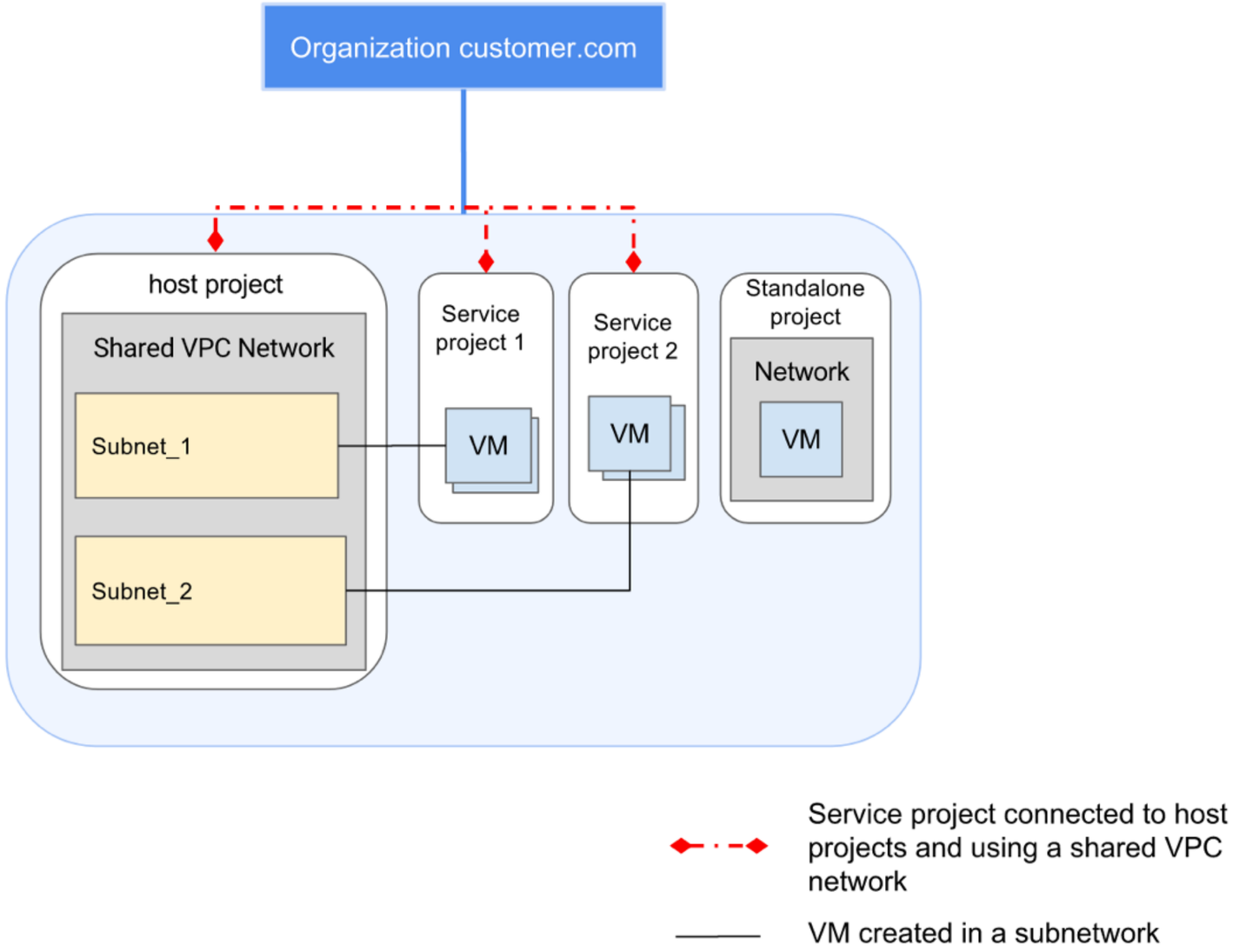
Shared VPC

Standalone project — A project that does not share networking resources with any other project



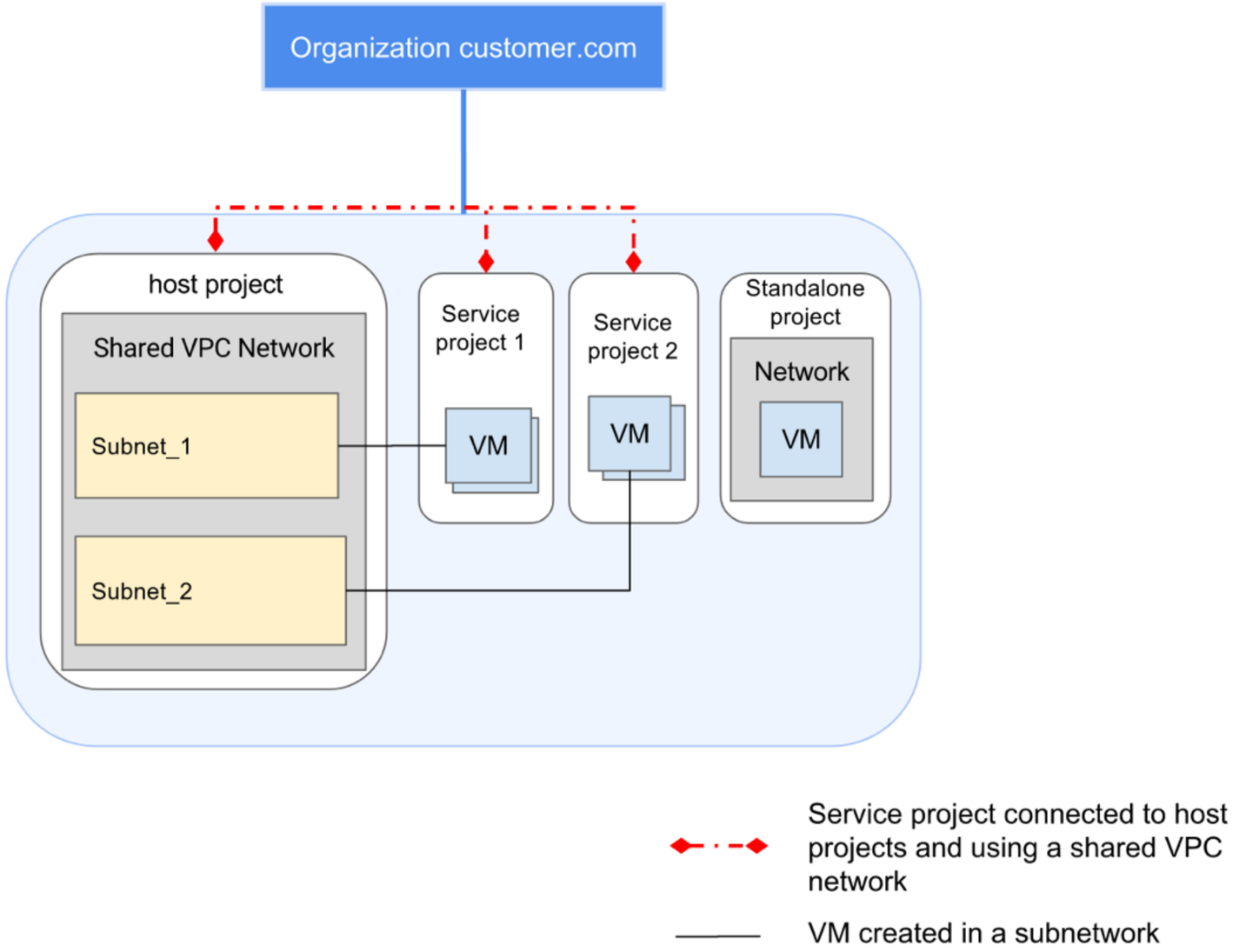
Shared VPC

Shared VPC network — A VPC network owned by the host project and shared with one or more service projects in the Cloud Organization



Shared VPC

Organization — The Cloud Organization is the top level in the Cloud Resource Hierarchy and the top-level owner of all the projects and resources created under it. A given host project and its service projects must be under the same Cloud Organization.



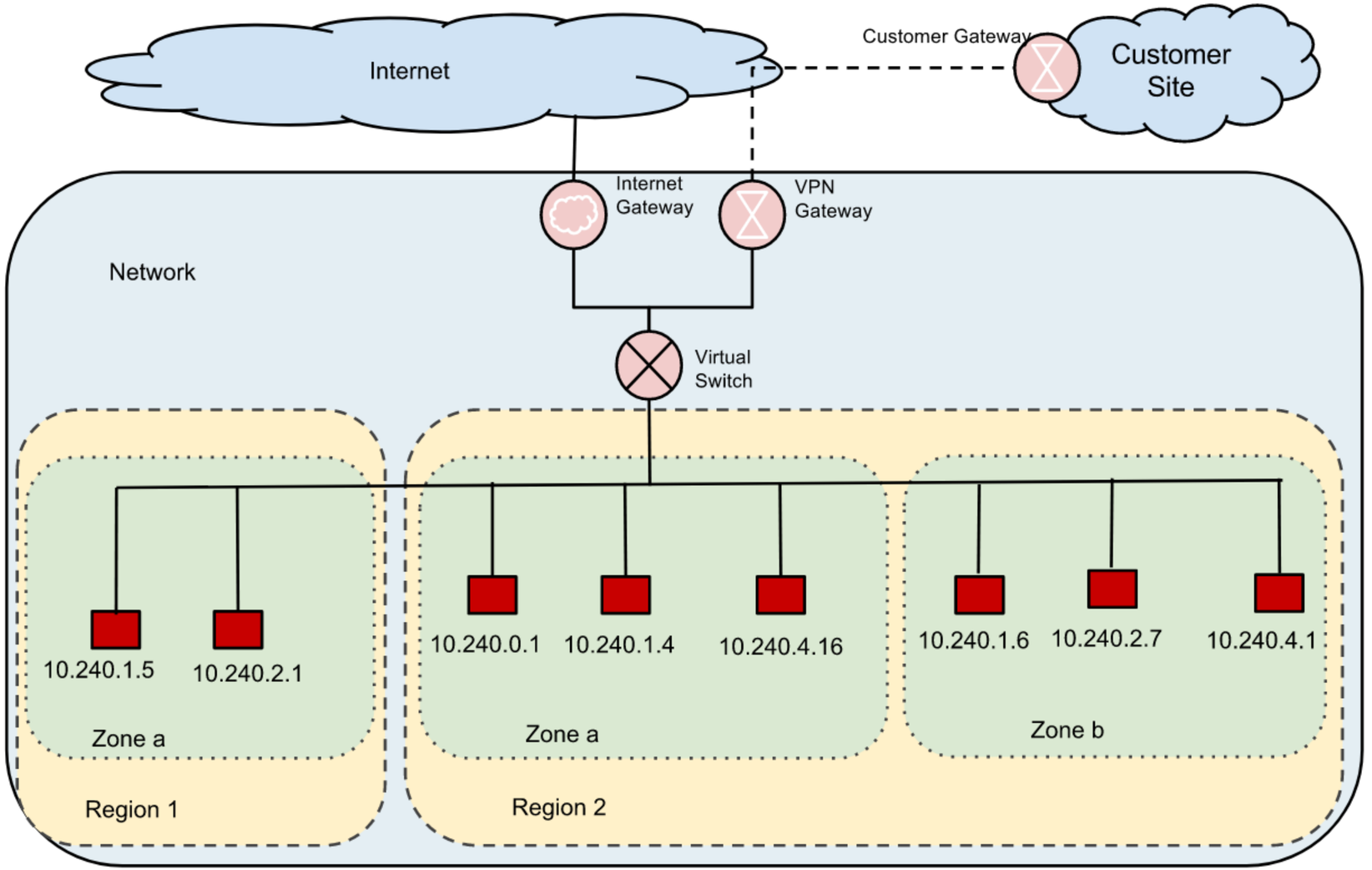
Legacy GCP Networks

Not recommended :-)

- Instance IP addresses are not grouped by region or zone
- No subnets
- Random and non-contiguous IP addresses

It is still possible to create legacy networks through the gcloud command-line tool and the REST API. It is not possible to create legacy networks using the Google Cloud Platform Console.

Legacy GCP Networks



Cloud DNS

Can refer to instances by name rather than IP

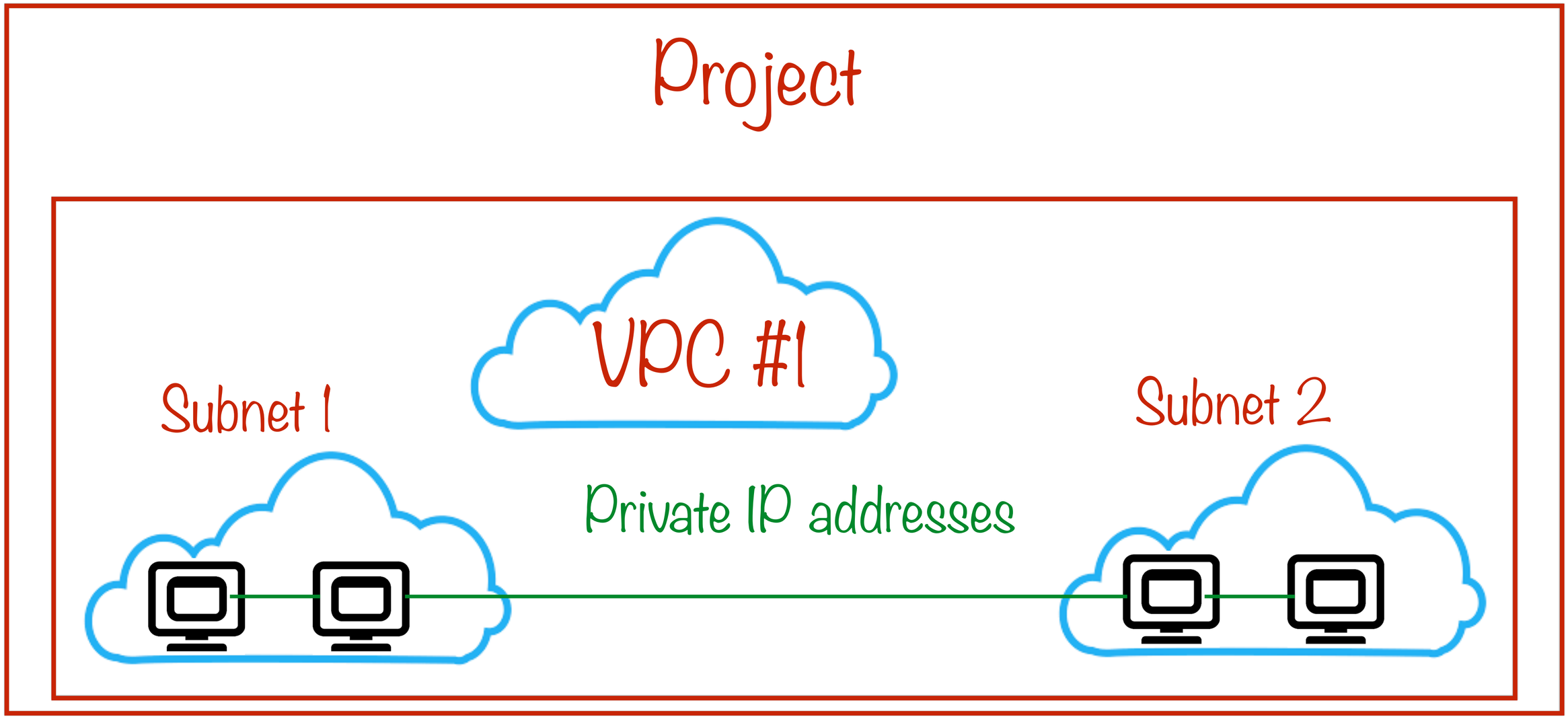
Each VM has a DNS resolver to allow this

Cloud DNS - no need to manage servers or lookup

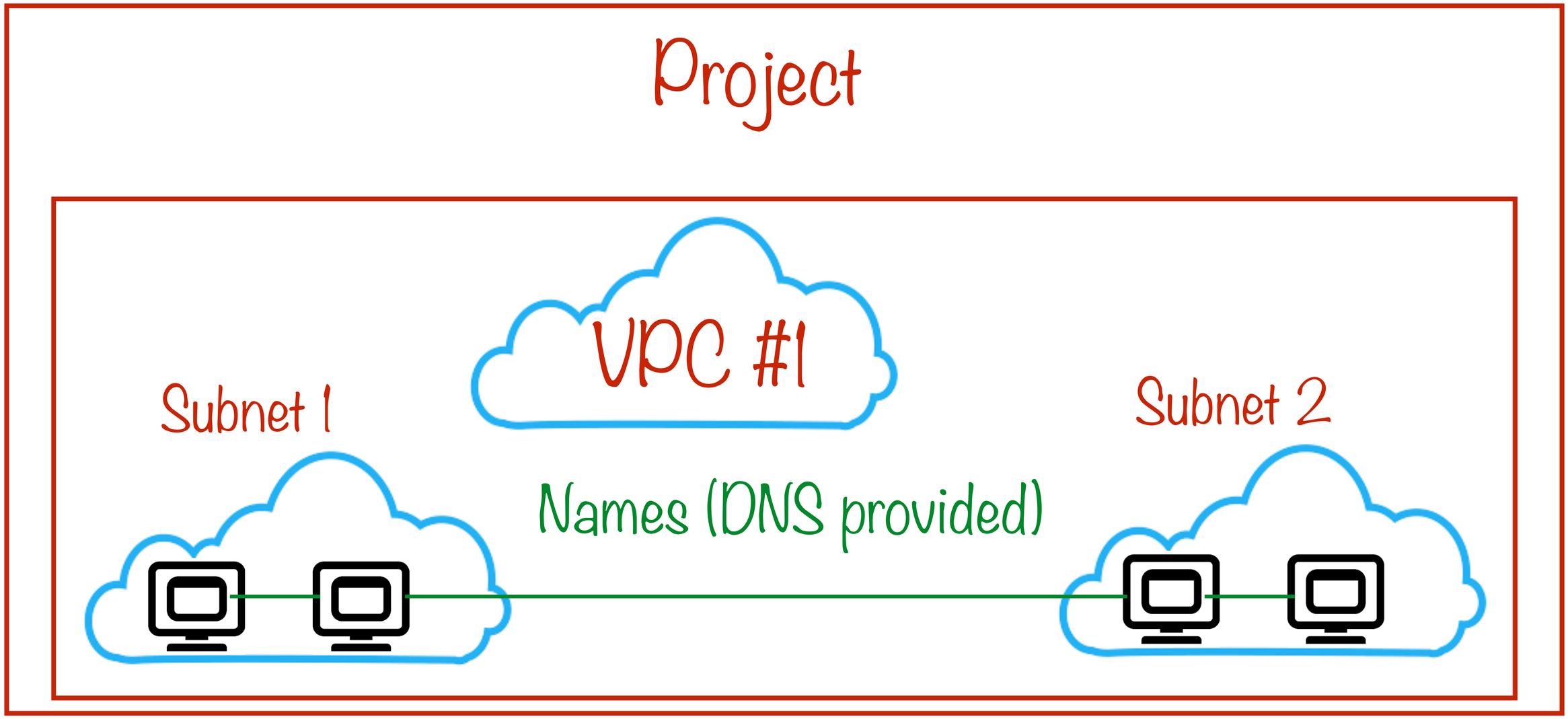
Cloud DNS

A global Domain Name System (DNS) service that publishes your domain names to the global DNS in a cost-effective way

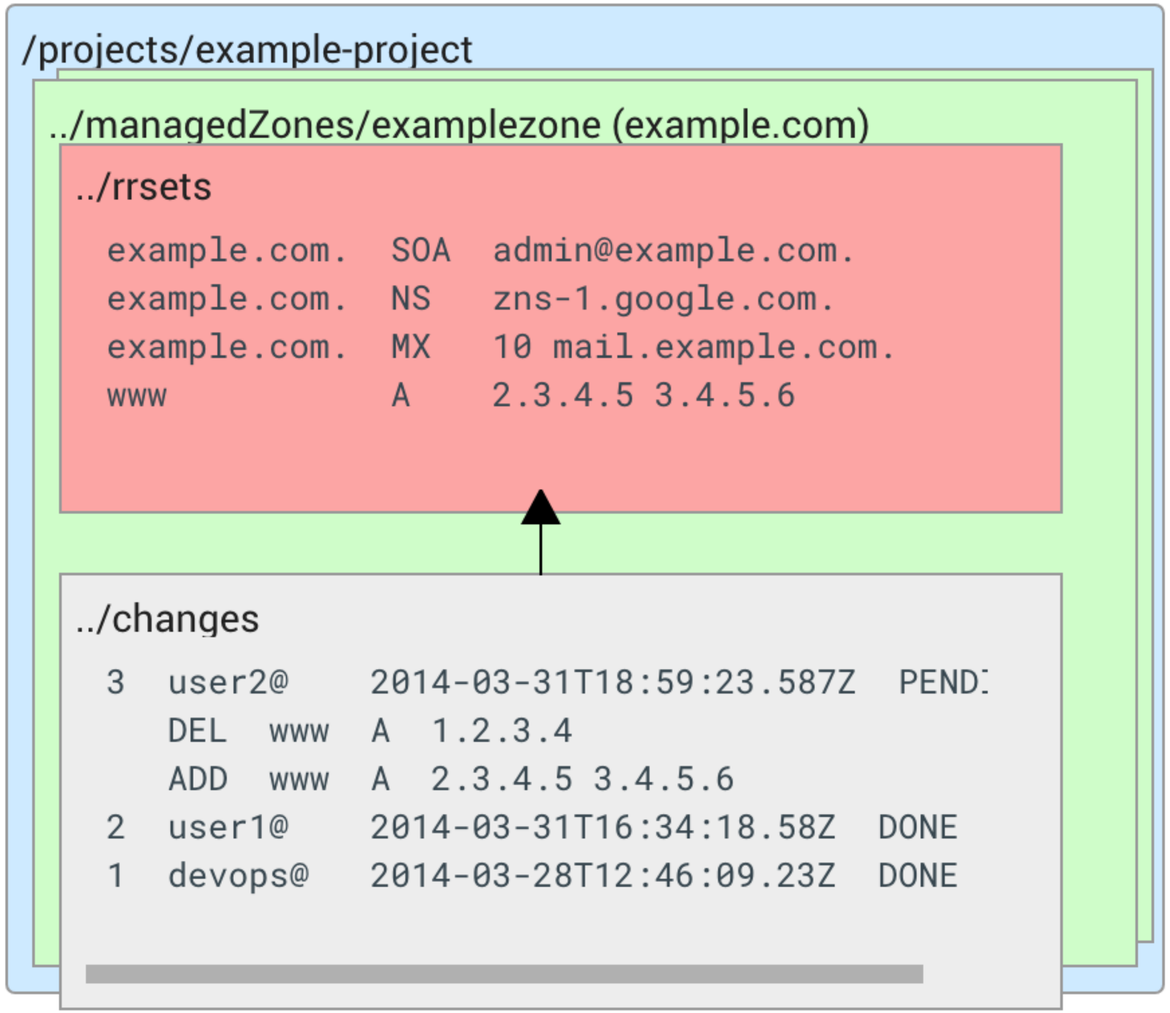
Cloud DNS



Cloud DNS



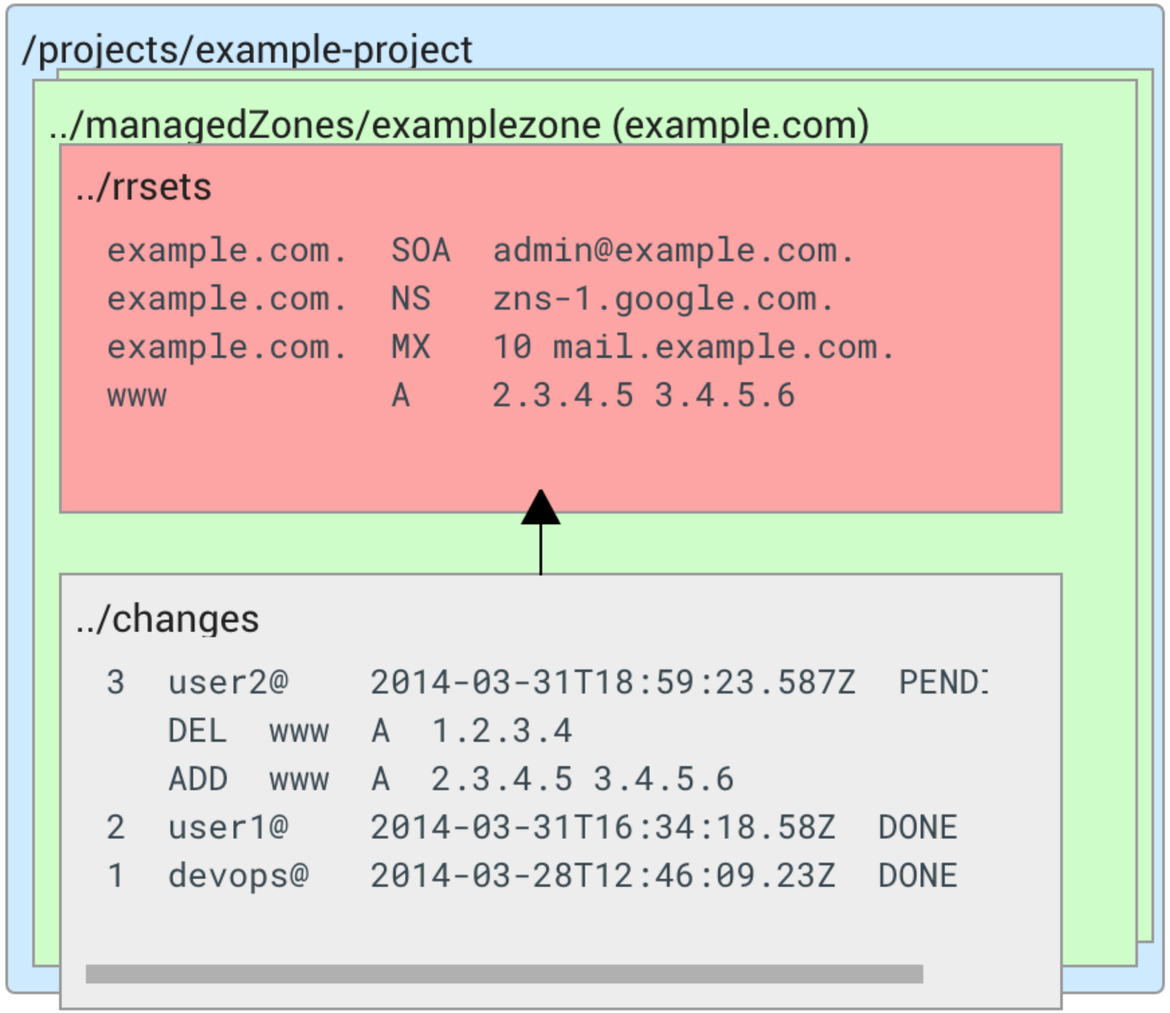
Cloud DNS



Managed Zone

- Entity that manages DNS records for a given suffix (example.com)
- Maintained by Cloud DNS

Cloud DNS



Record types -

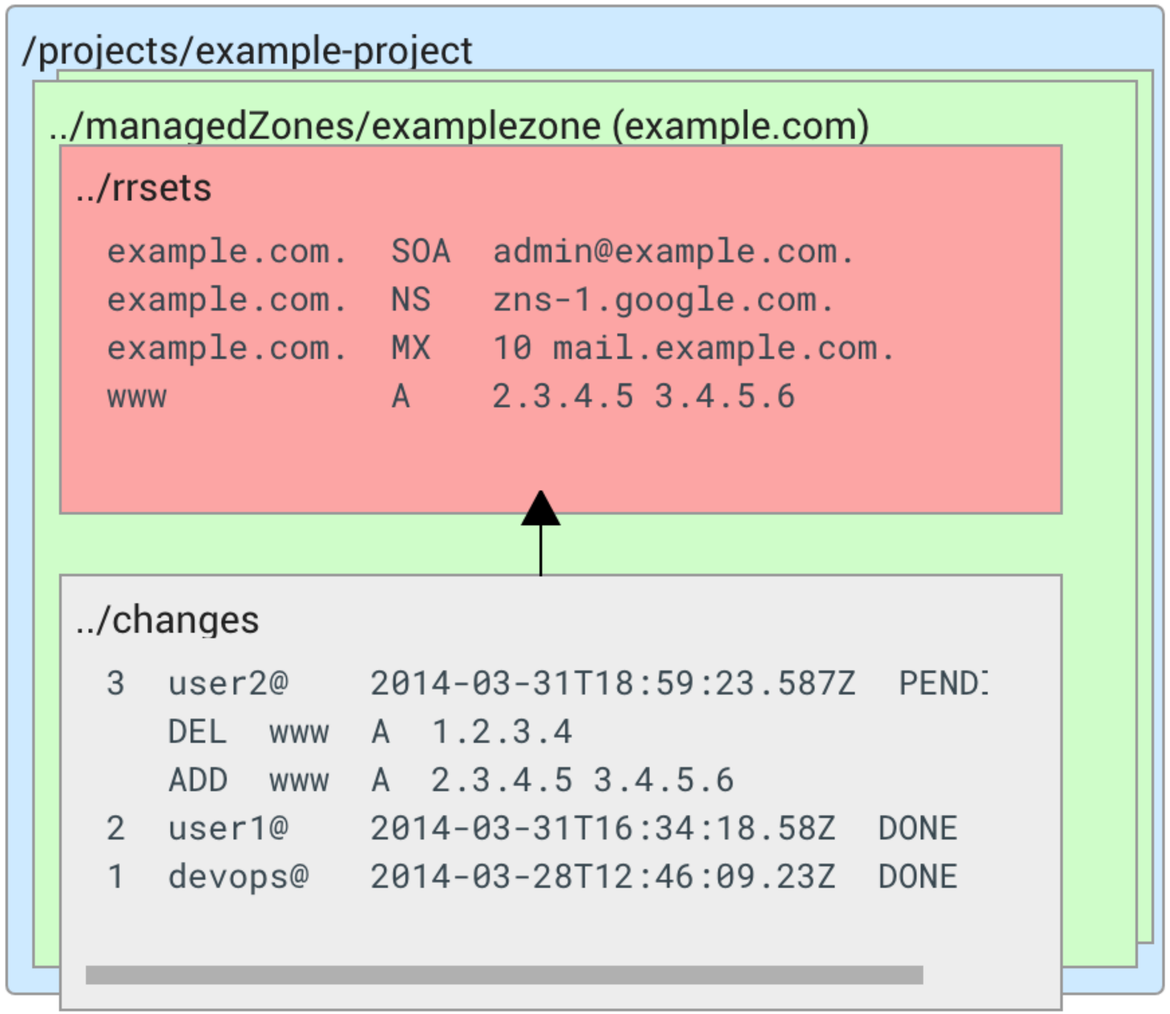
A - Address record

SOA - Start of authority (needed to create managed zone)

MX - Mail exchange

NS - Name Server record

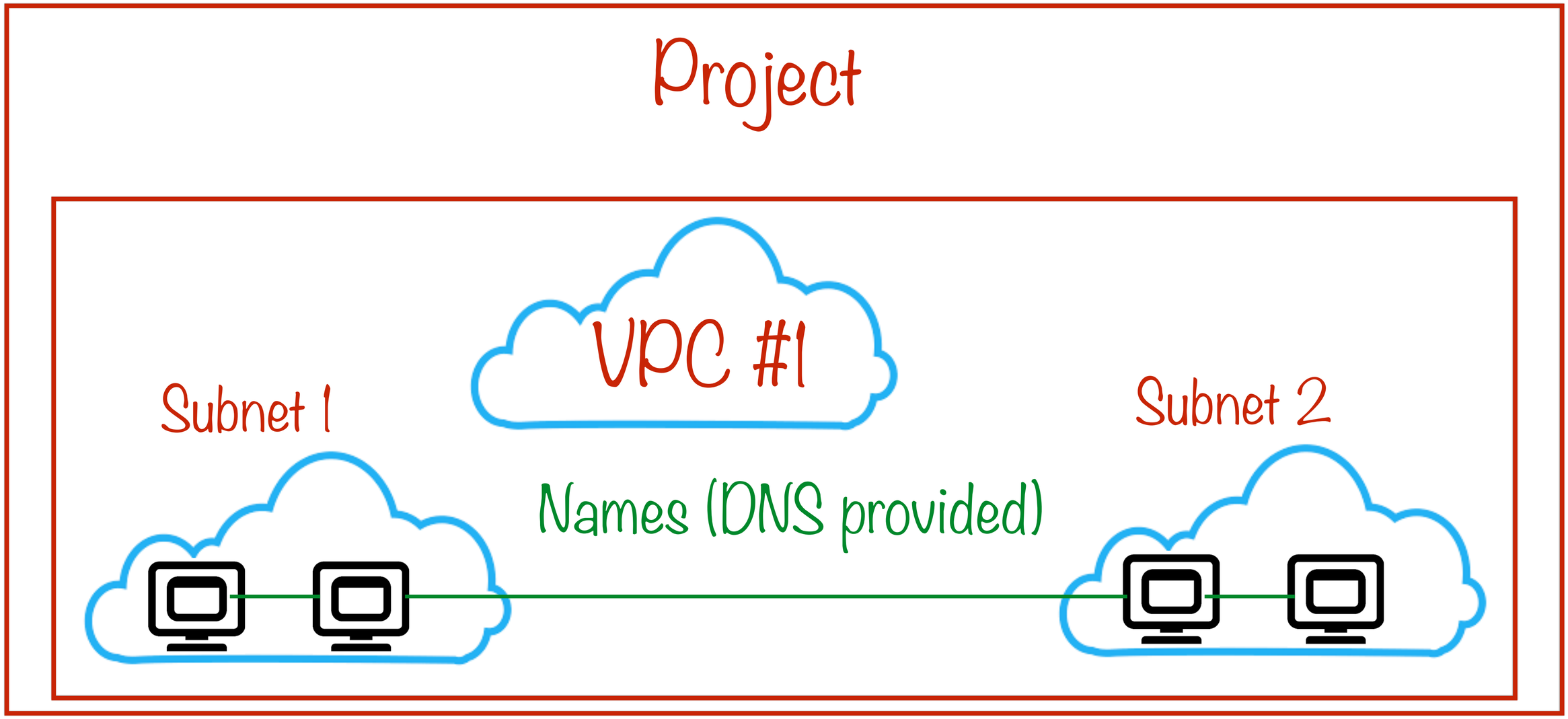
Cloud DNS



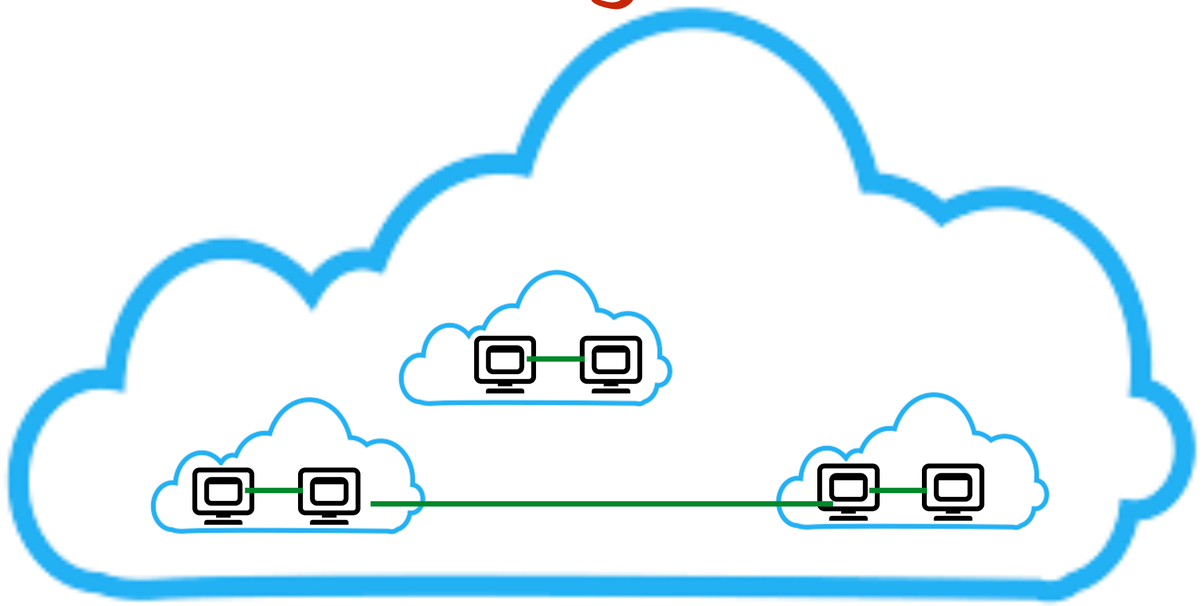
Resource Record Changes

Changes to the record sets

Cloud Interconnect



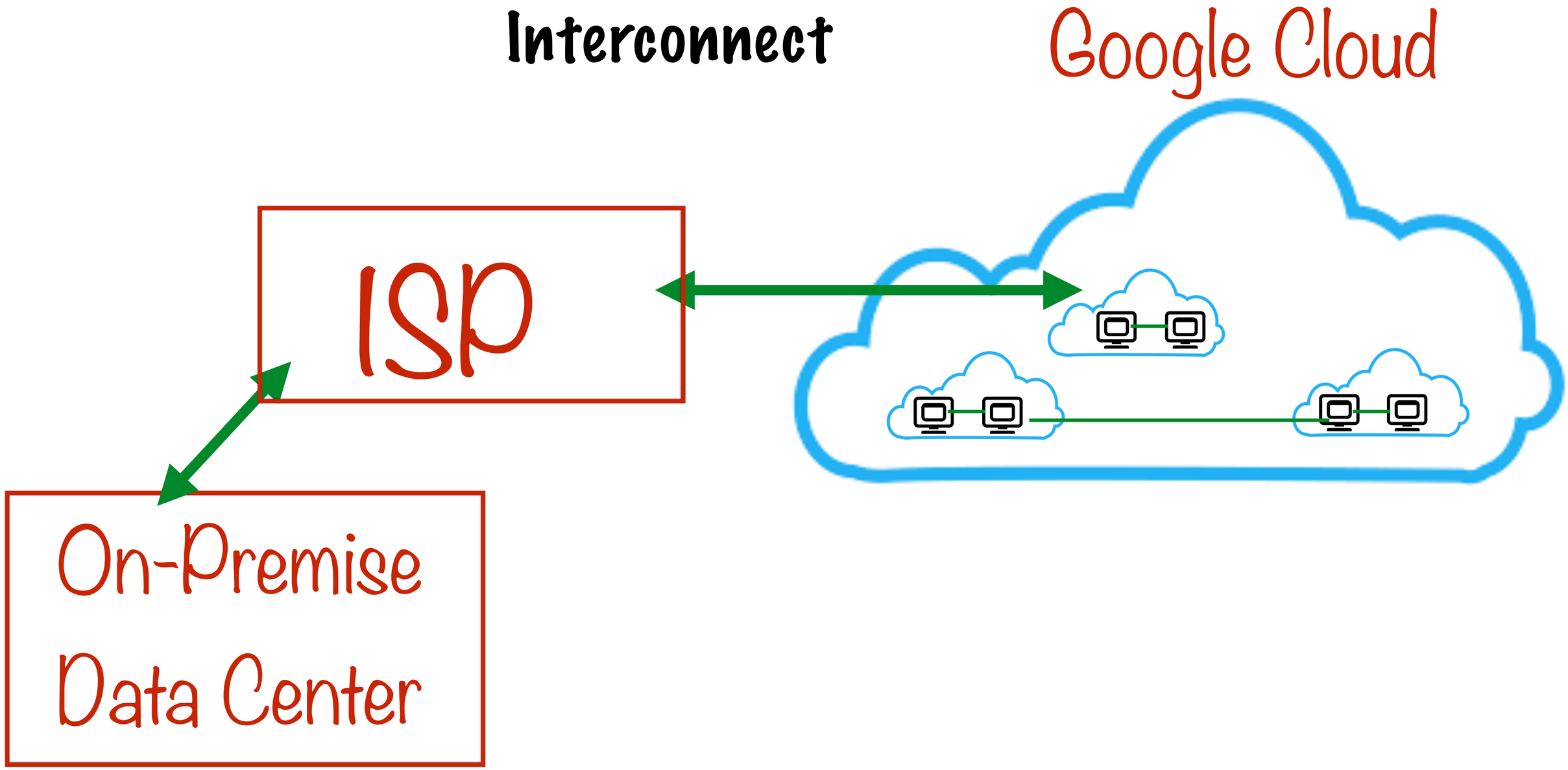
Google Cloud



Cloud
Interconnect

On-Premise
Data Center

Cloud Interconnect

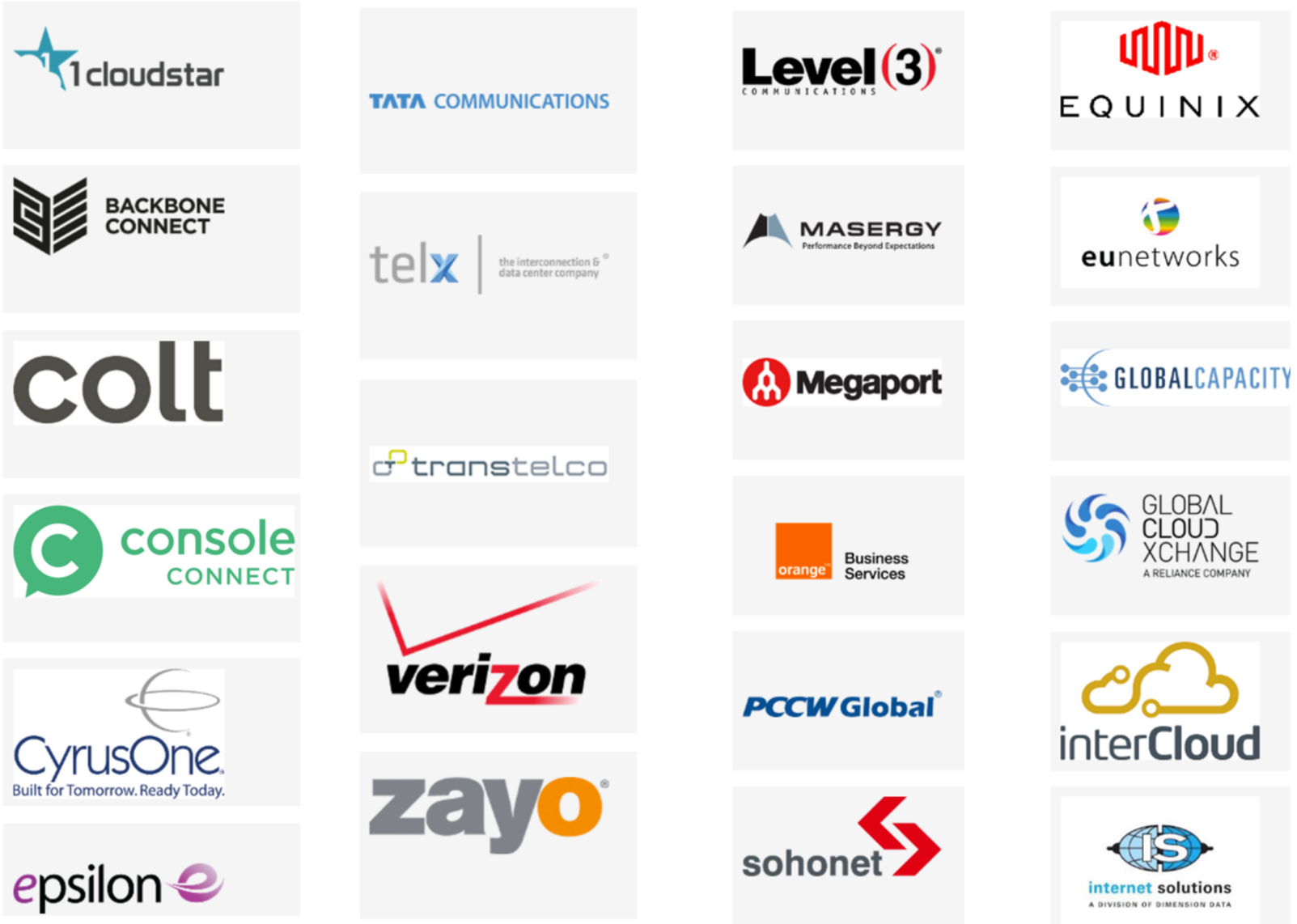


Cloud Interconnect

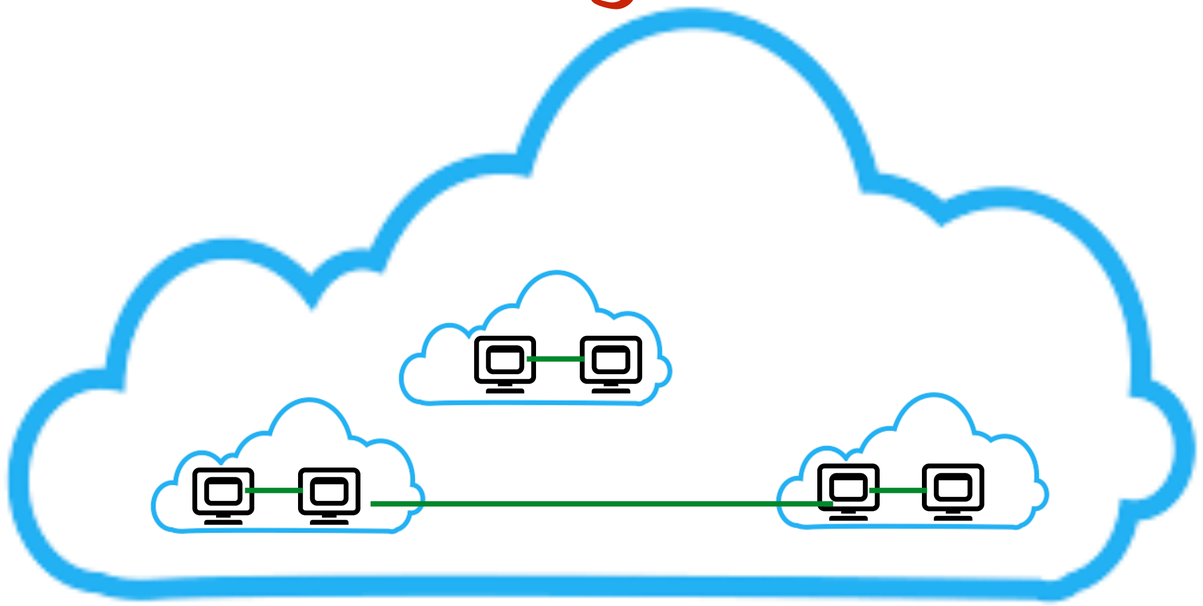
Data-intensive applications

Latency-sensitive applications

Cloud Interconnect Providers



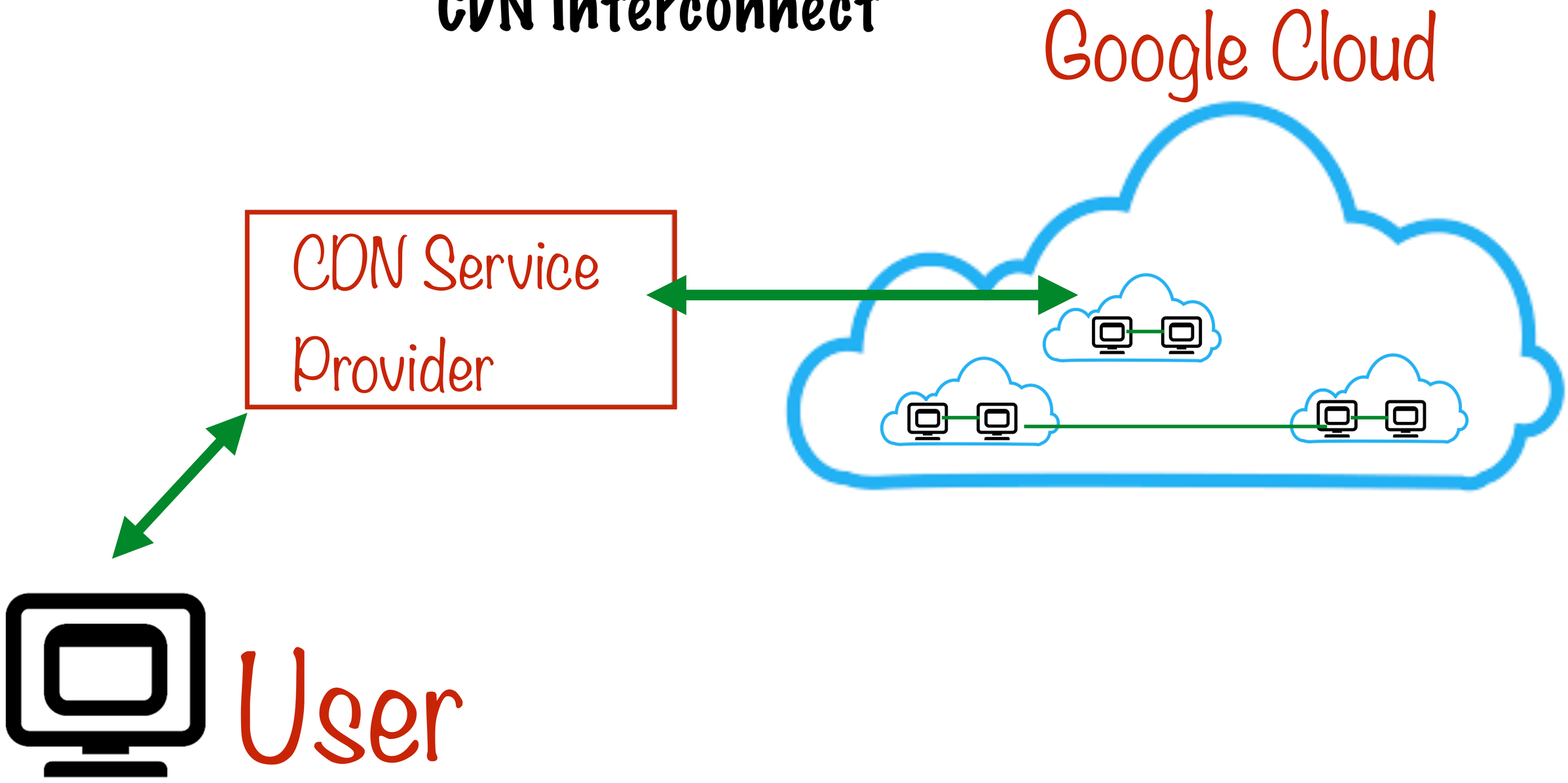
Google Cloud



Cloud
Interconnect

On-Premise
Data Center

CDN Interconnect



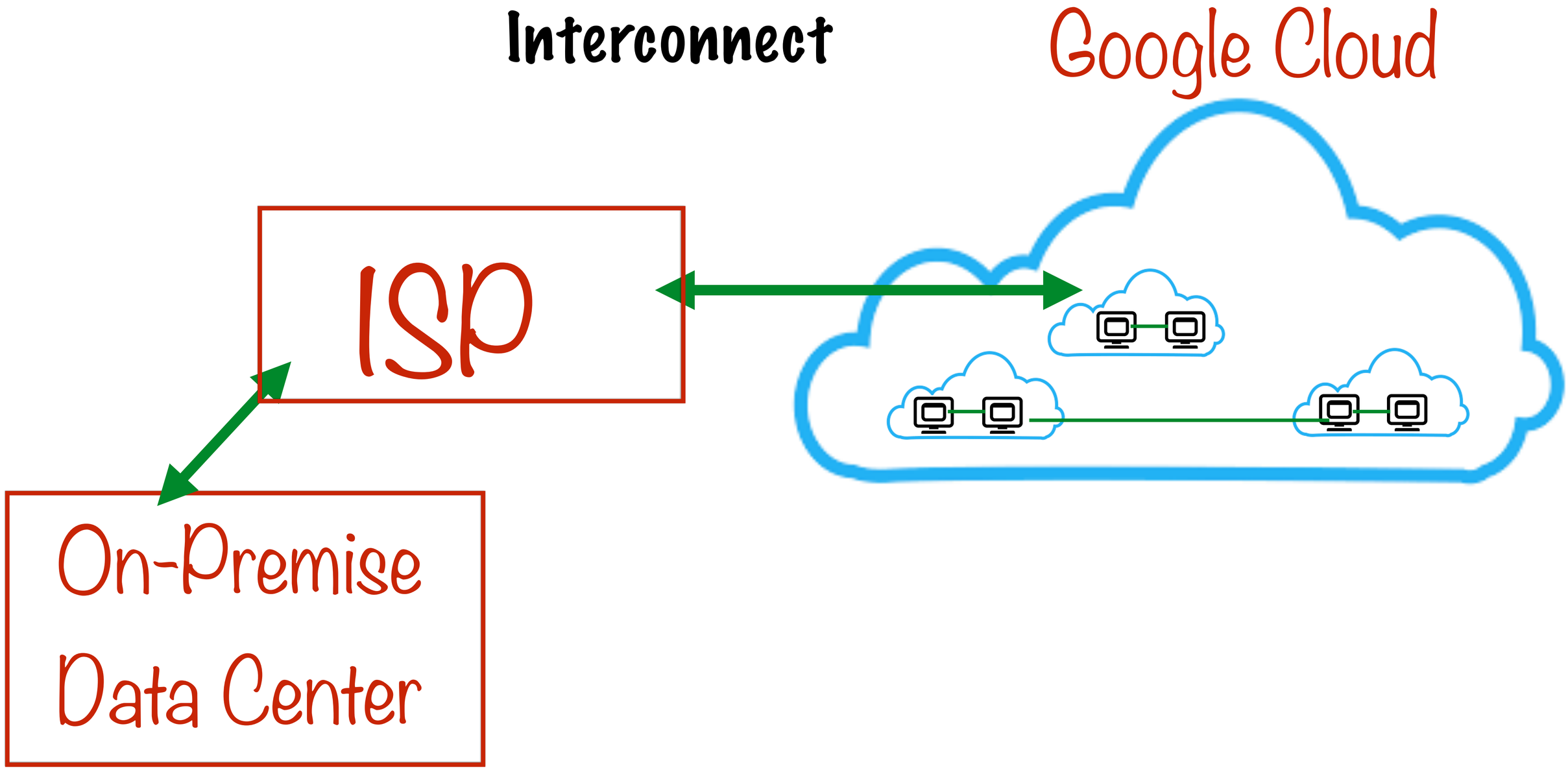
CDN Interconnect

High-volume egress traffic

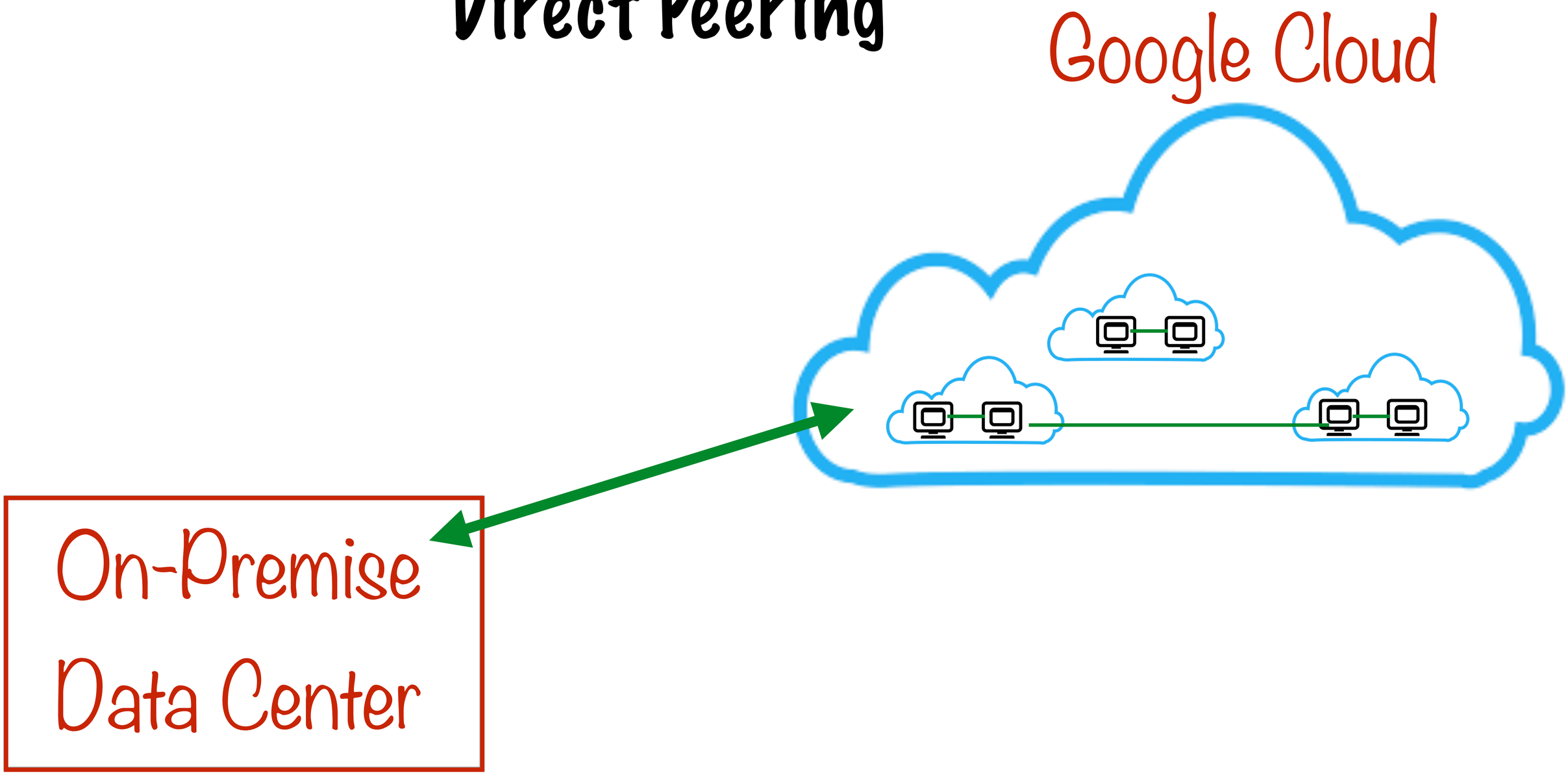
Frequent content updates

CDN Interconnect Service Providers

Cloud Interconnect



Direct Peering



Direct Peering

"Free"

No intermediate ASNs - control over paths etc

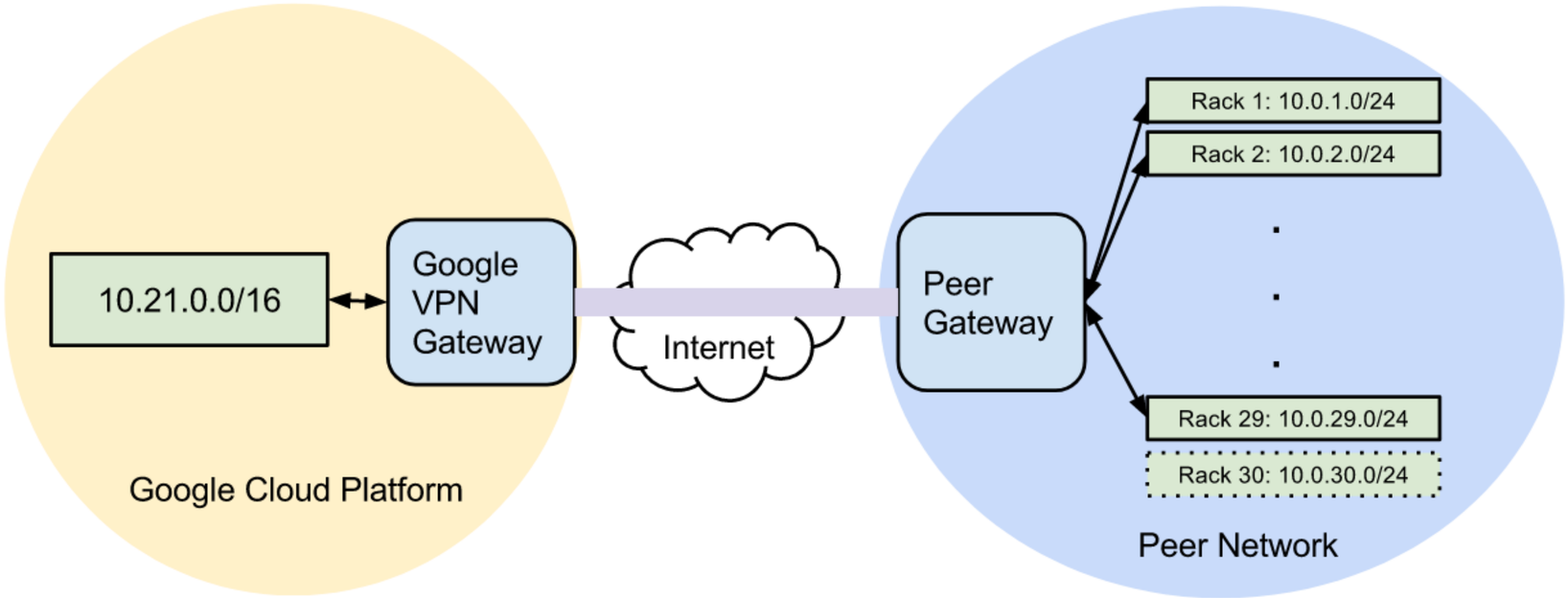
Performance tuning - edge nodes etc

Direct Peering Requirements

Networks wishing to peer with Google must have the following:

- Publicly routable ASN
- Publicly routable address space (at least one /24 of IPv4 and/or one /48 of IPv6 space)
- ASN record completed in PeeringDB
- 24x7 NOC contact capable of resolving BGP routing issues
- Presence at one or more of the 90+ Internet Exchanges, or 100+ private peering interconnection facilities listed for Google in PeeringDB
- Minimum traffic requirements as below

VPN

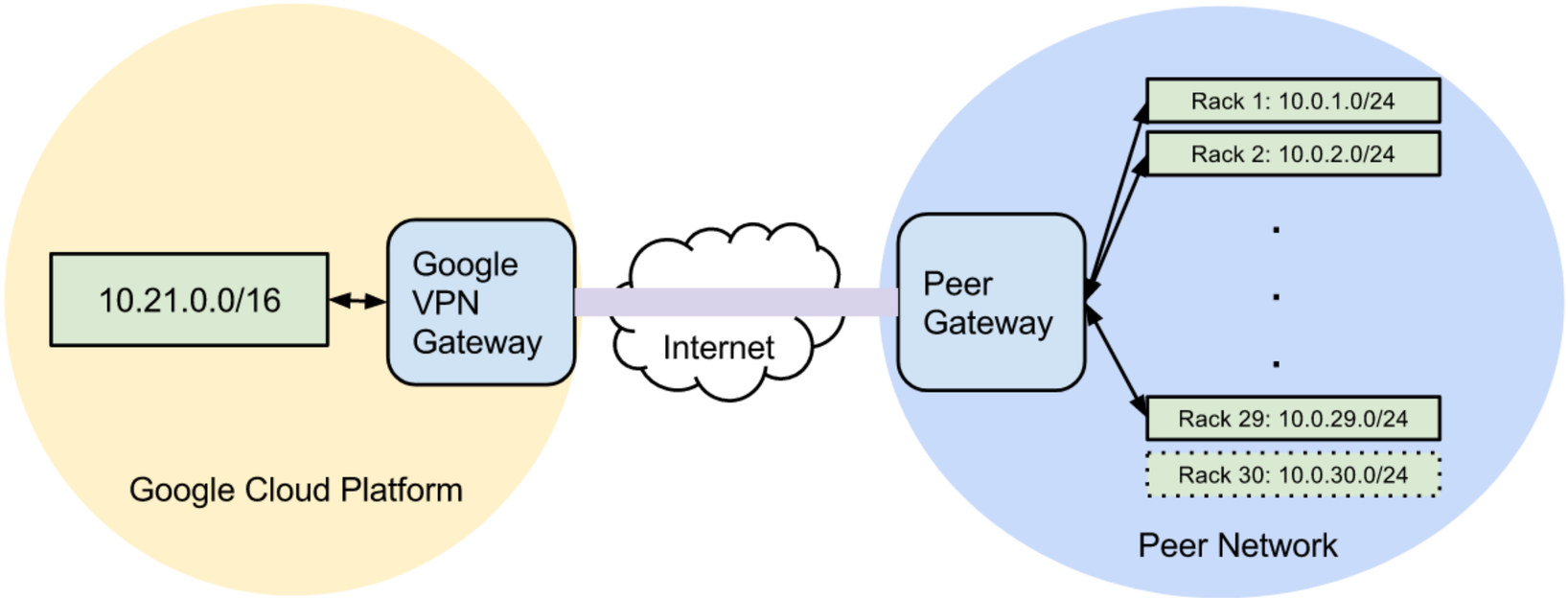

10.21.0.0/16
Google
VPN
Gateway
Internet
Peer
Gateway
Rack 1: 10.0.1.0/24
Rack 2: 10.0.2.0/24
Rack 29: 10.0.29.0/24
Rack 30: 10.0.30.0/24
Google Cloud Platform
Peer Network

VPN

- Securely connects your on-premises network to Virtual Private Cloud (VPC) network through an IPsec VPN connection.

- Traffic traveling between the two networks is protected - encrypted by one VPN gateway, then decrypted by the other VPN gateway

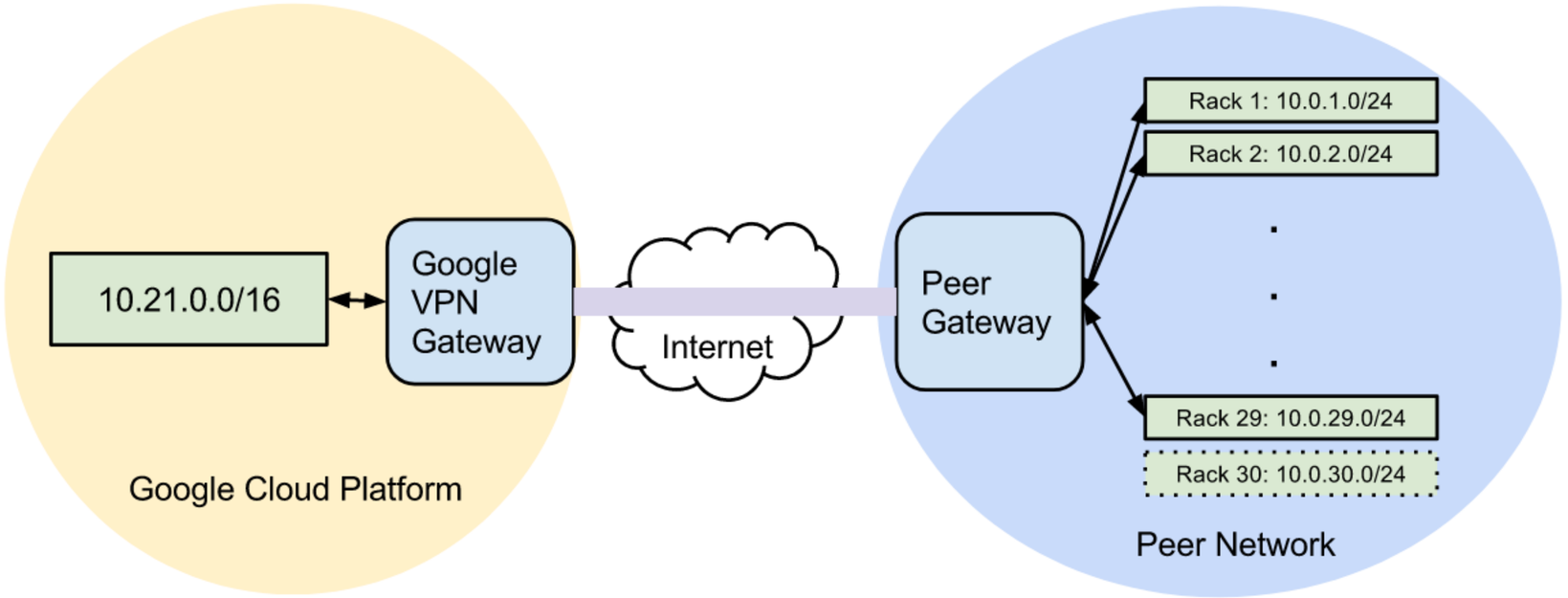
VPN



Cloud VPN gateway: The virtual VPN gateway running in GCP.

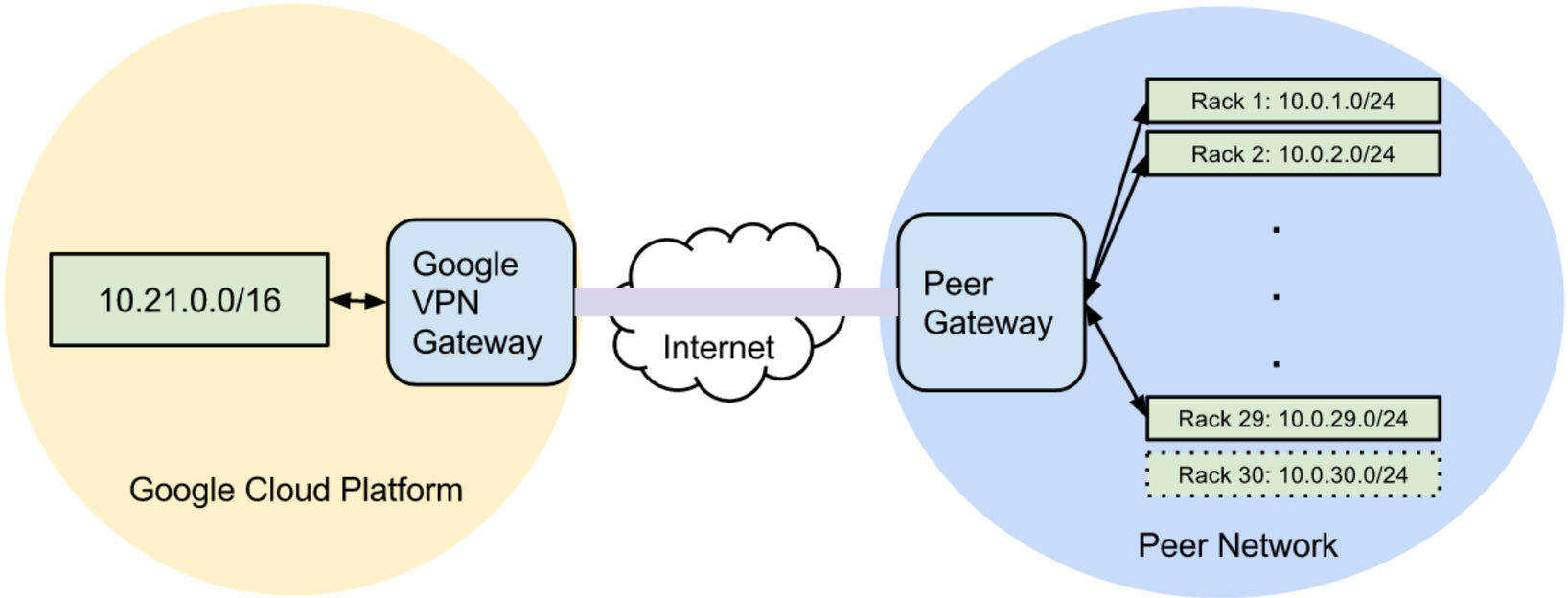
This virtual device is managed by Google, but used only by you.

VPN



Peer VPN gateway: Frequently, a physical device on your premises.
(However, it can be a second Cloud VPN gateway or a virtual gateway running in another provider's network)

VPN


10.21.0.0/16
Google
VPN
Gateway
Internet
Peer
Gateway
Rack 1: 10.0.1.0/24
Rack 2: 10.0.2.0/24
Rack 29: 10.0.29.0/24
Rack 30: 10.0.30.0/24
Google Cloud Platform
Peer Network

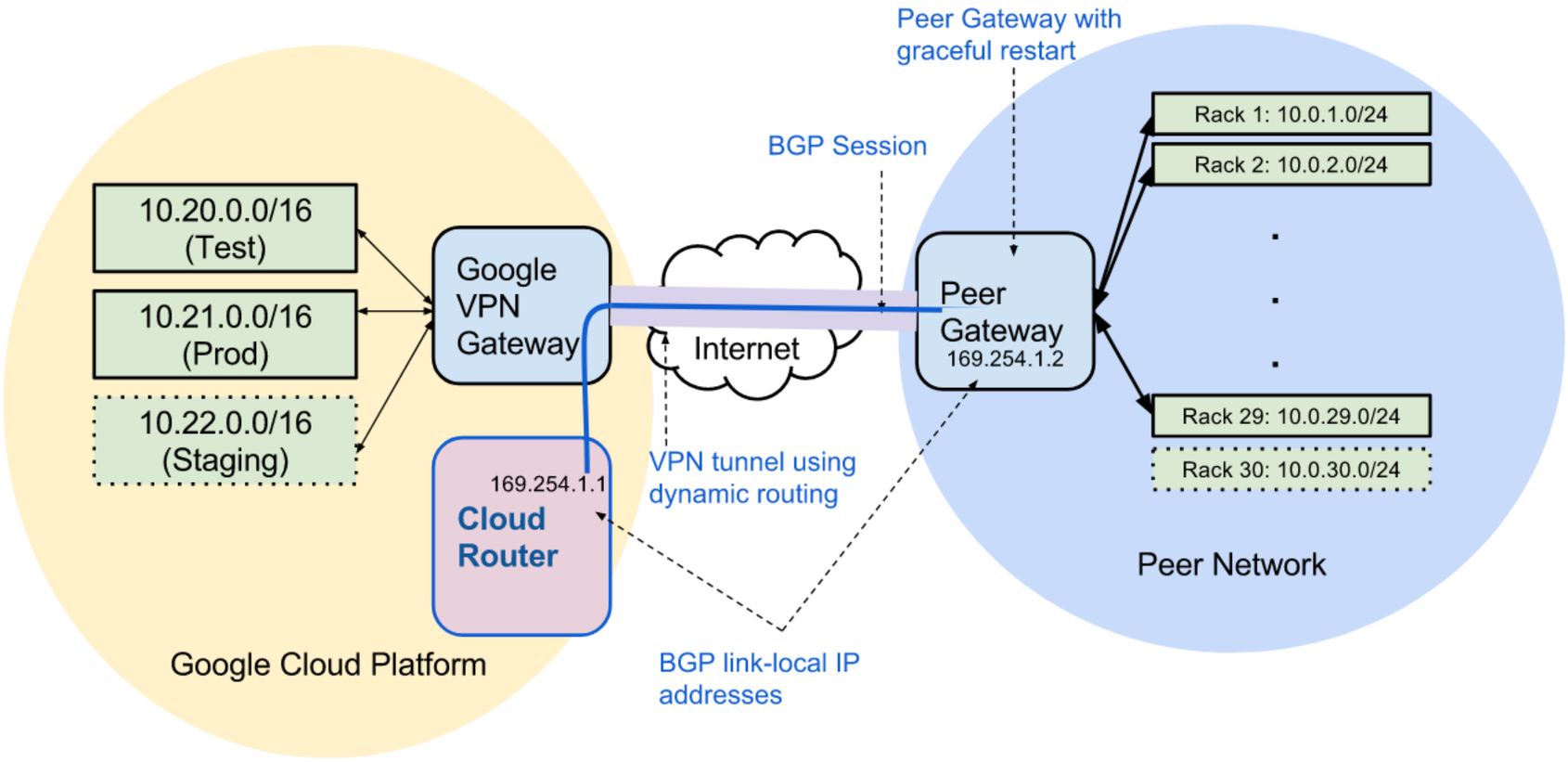
Static Routes on VPN

- A network configuration change on either side of a VPN tunnel requires manually creating and deleting static routes corresponding to these changes.
- Static routes changes are slow to converge.
- Every time the routes must change, the VPN tunnel has to be updated (deleted and recreated) with the new routes, which disrupts existing traffic
- There is no standard way to configure static routes. Different vendors use different commands.

Cloud Router

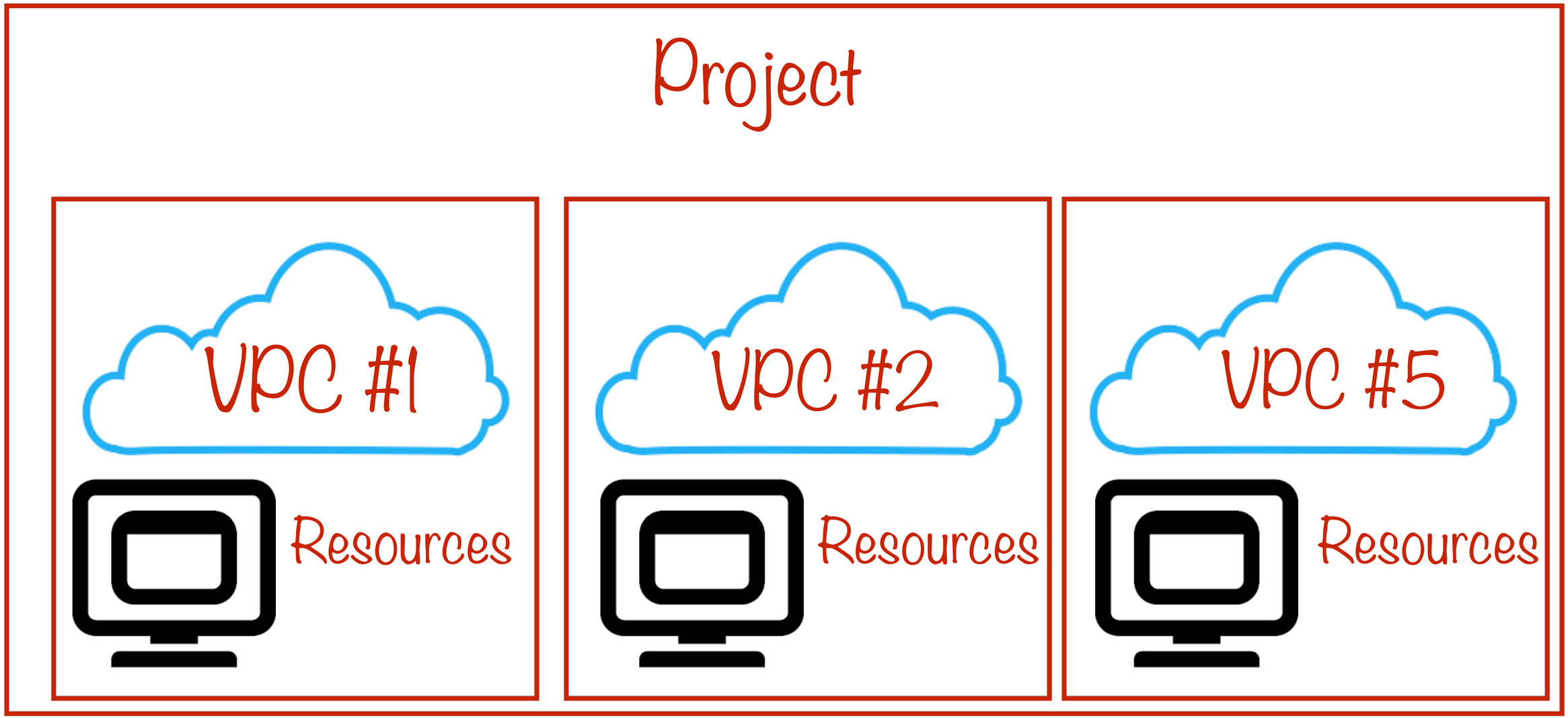
- Enables dynamic Border Gateway Protocol (BGP) route updates between your VPC and on-premises network
- Essential requirement for large organizations with enterprise-grade VPNs.
- Eliminates the need to configure static routes for VPN tunnels
- Provides an automatic, rapid, and interoperable way of discovering network topology changes.
- Eliminates the need to restart VPN tunnels on topology changes; enables seamless network topology changes without traffic disruption.

VPN with Cloud Router

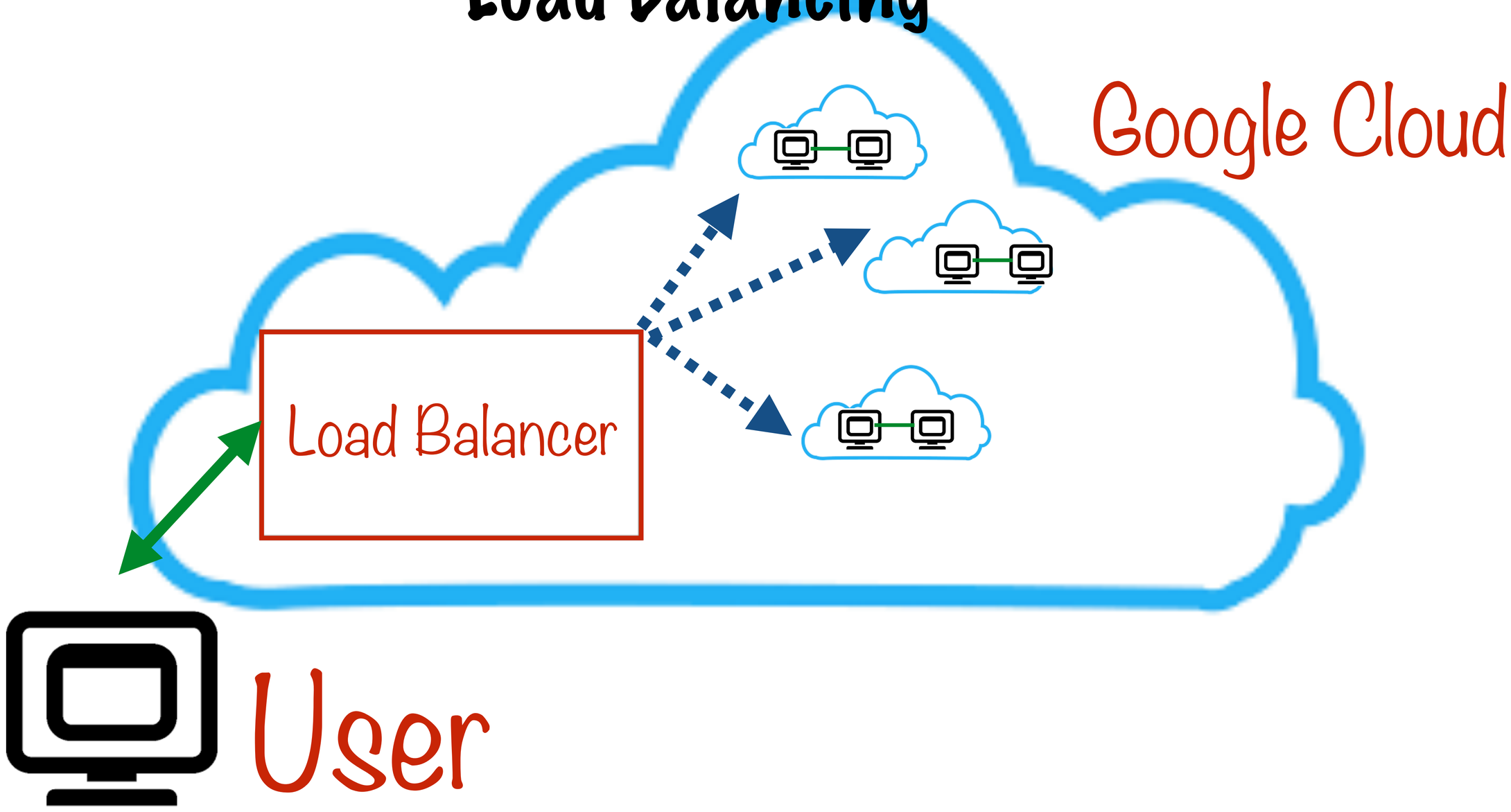


Load Balancing

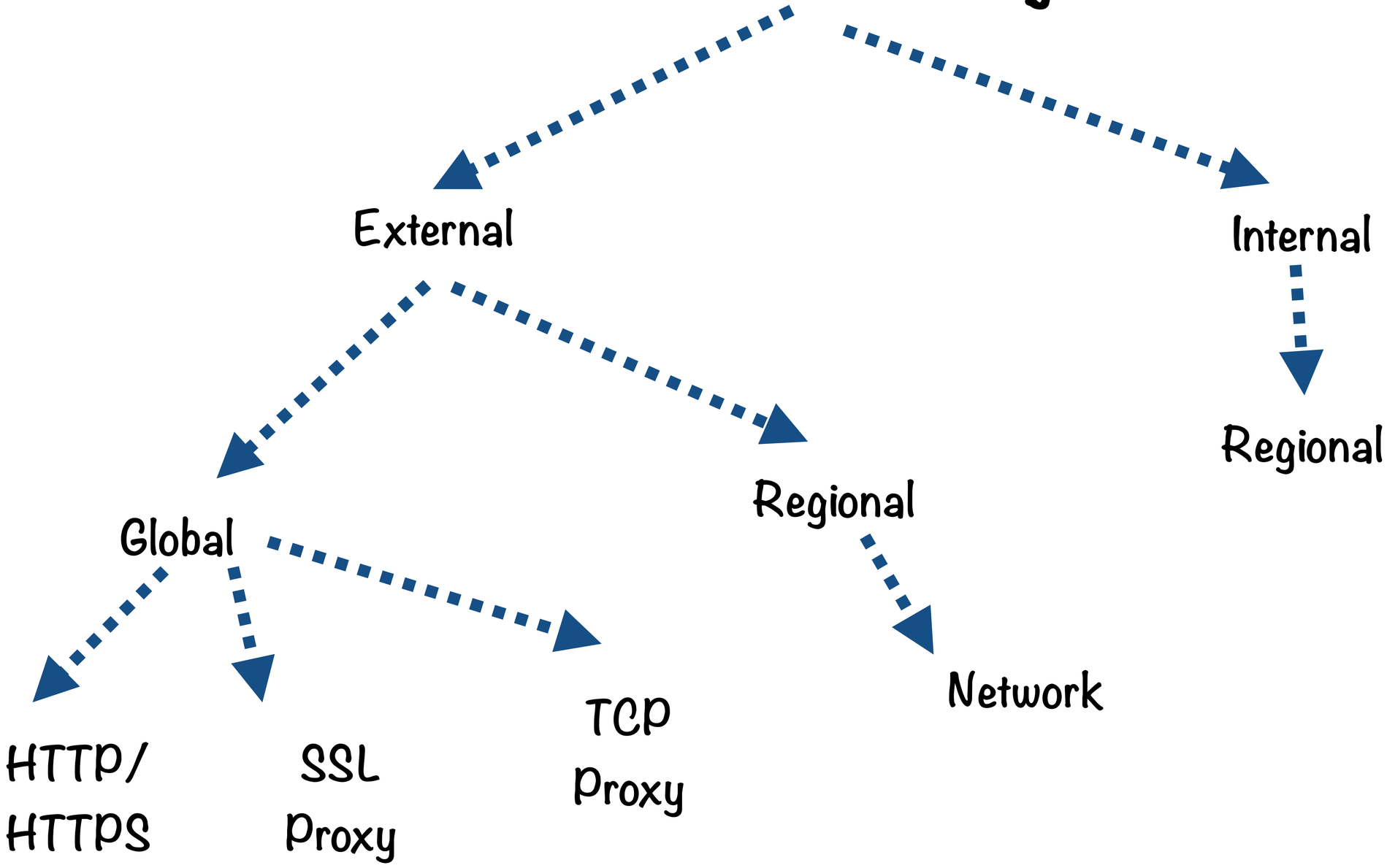
Projects and VPCs



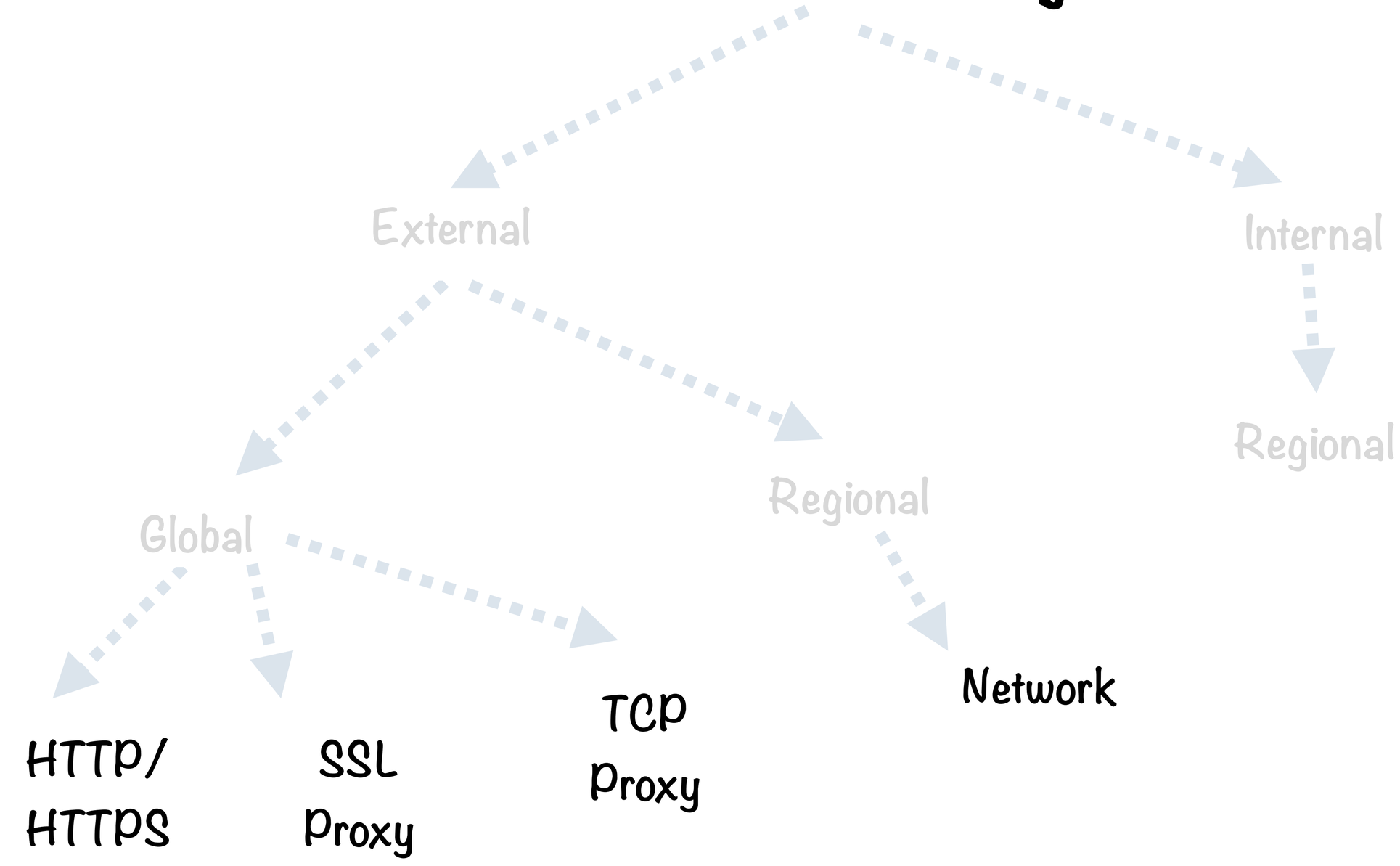
Load Balancing



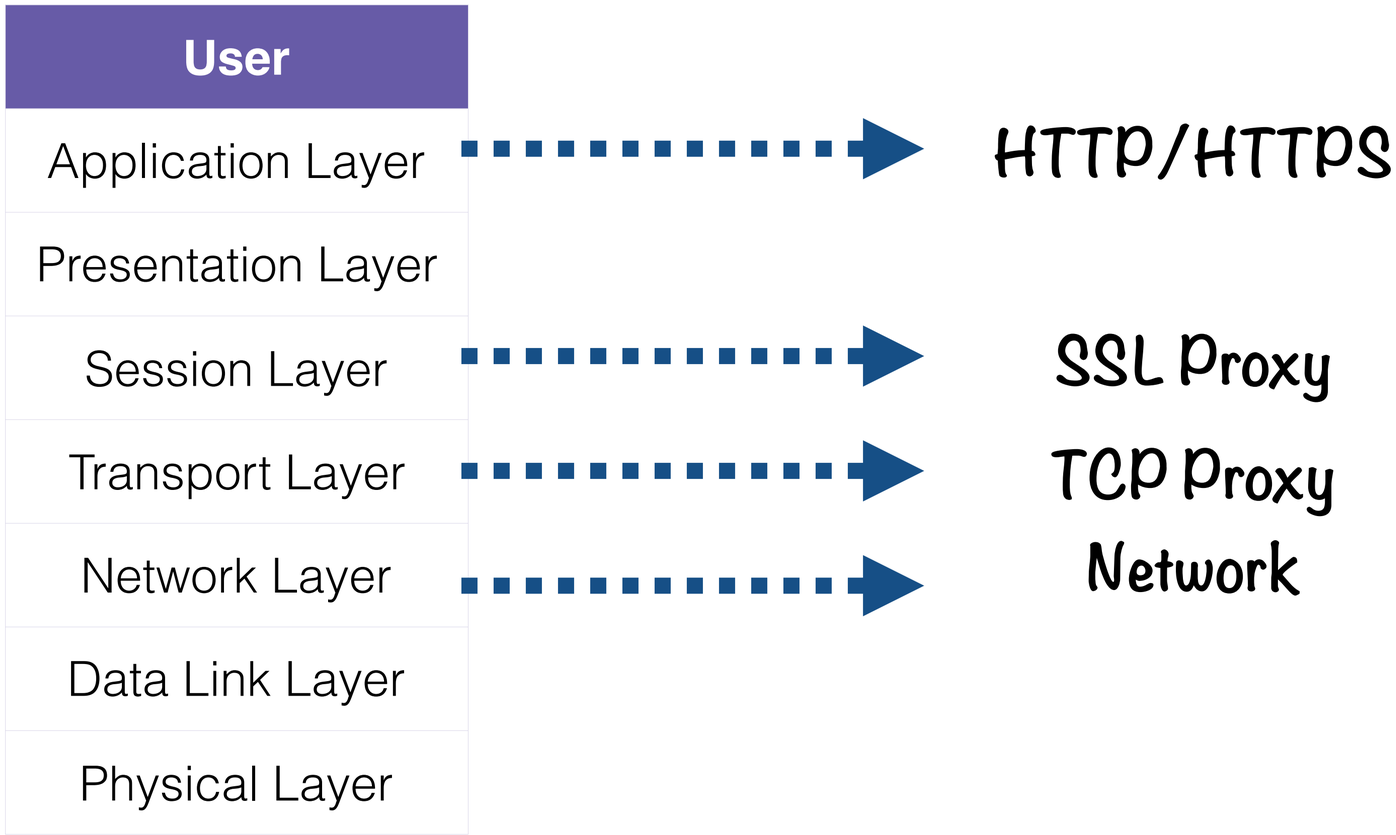
Load Balancing



Load Balancing



OSI Network Stack



OSI Network Stack

User
Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

HTTP/HTTPS

SSL Proxy

TCP Proxy

Network

Rule-of-thumb: Load balancer in the highest layer possible

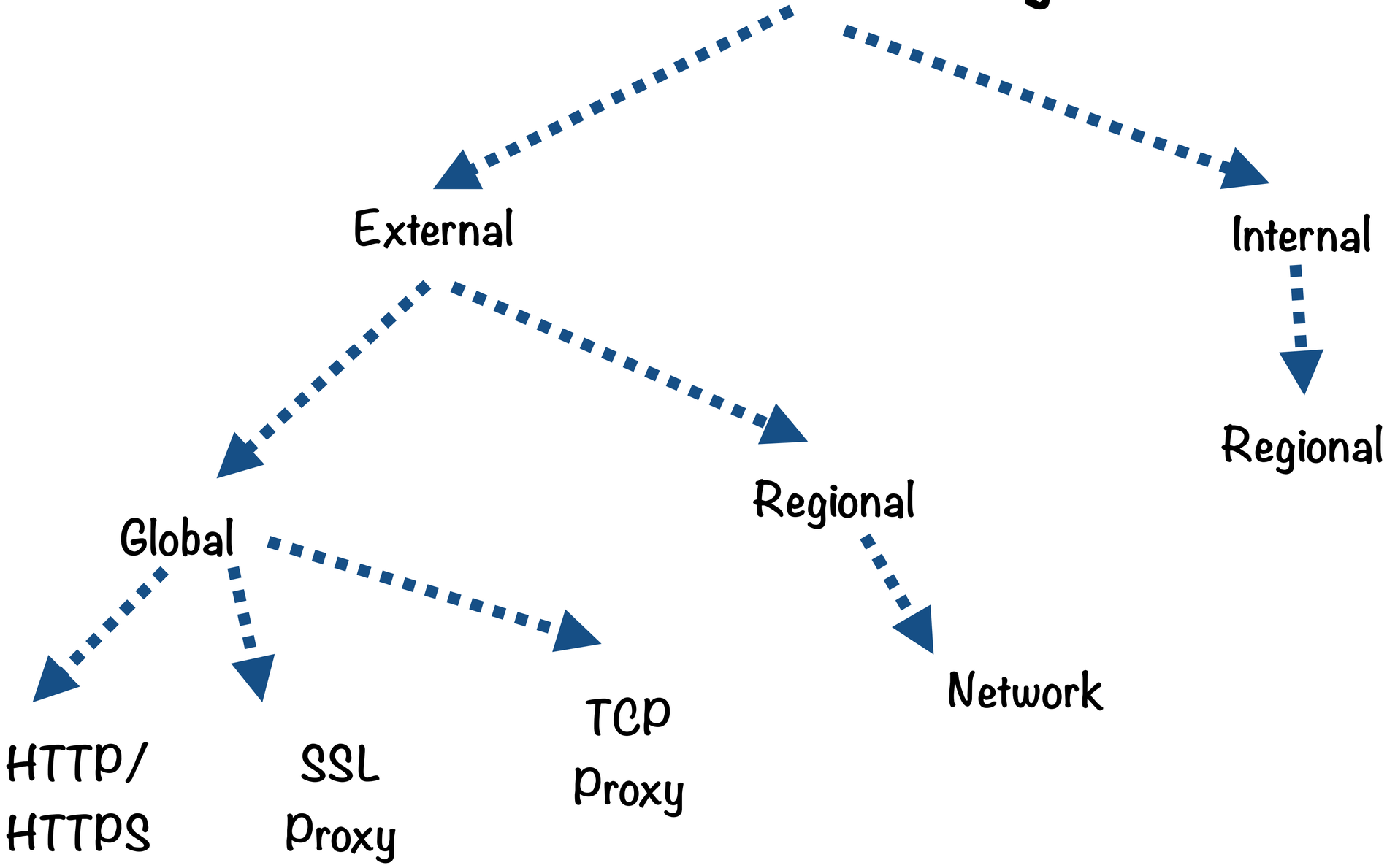
Health Checks

- HTTP, HTTPS health checks: If your traffic is HTTP(S), then HTTP(S) health checks provide the highest fidelity check because they verify that the web server is up and serving traffic, not just that the instance is healthy.

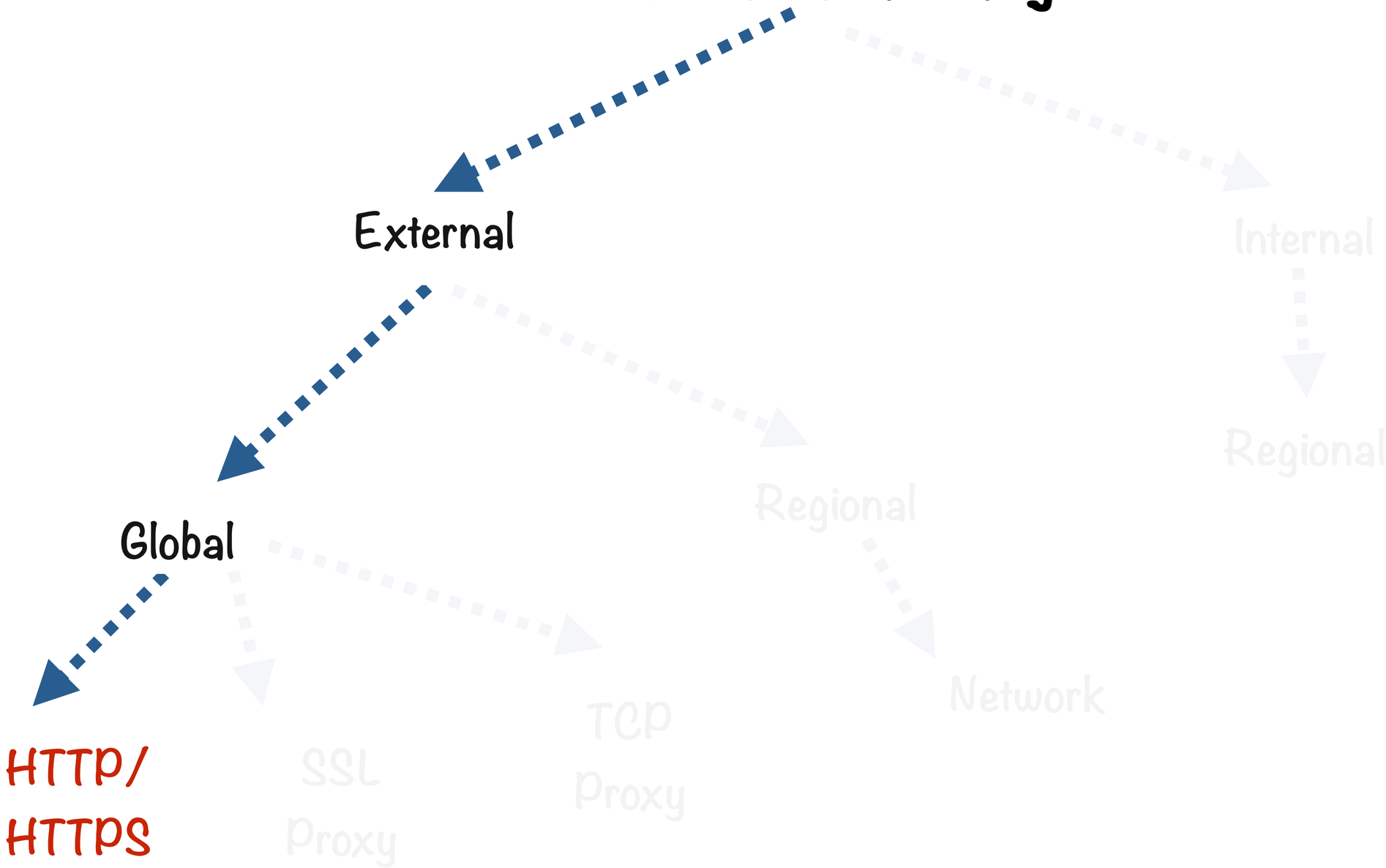
- TCP health checks: Configure the SSL health checks if your traffic is not HTTPS but is encrypted via SSL(TLS)

- SSL (TLS) health checks: For all TCP traffic that is not HTTP(S) or SSL(TLS), you can configure a TCP health check

Load Balancing



Load Balancing



OSI Network Stack

User
Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

HTTP/HTTPS

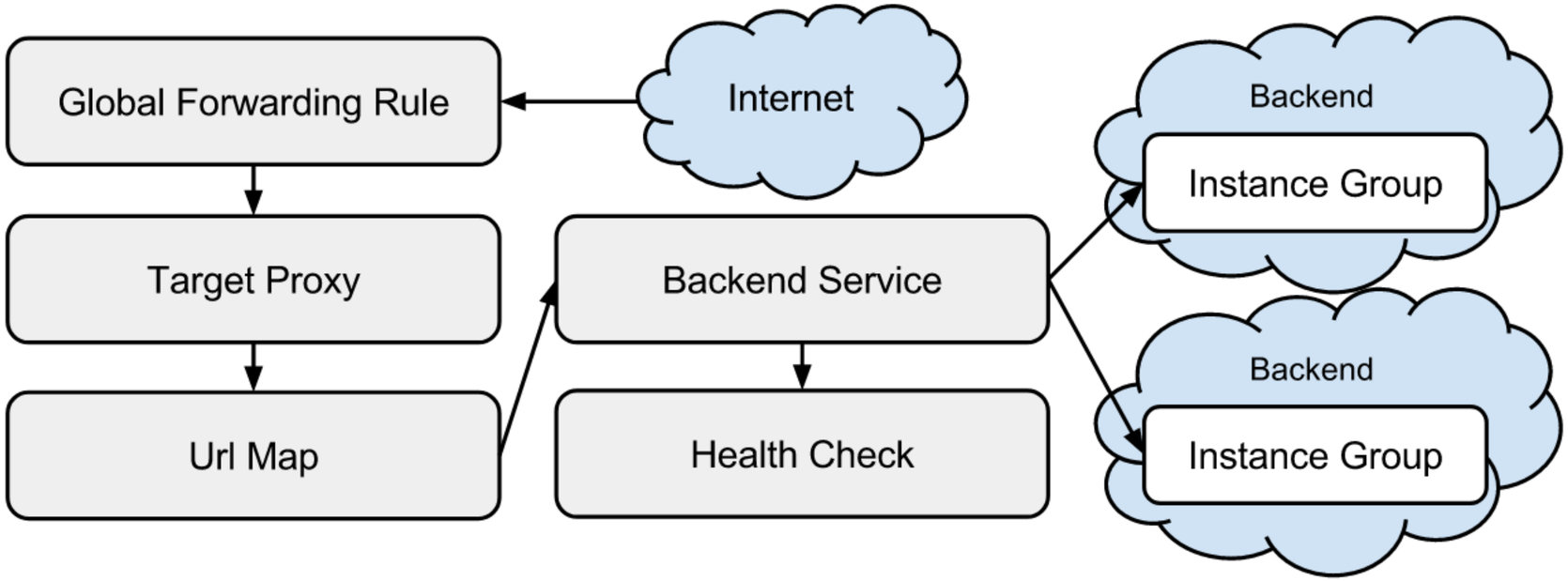
SSL Proxy

TCP Proxy

Network

HTTP(S) load balancing is the "smartest"

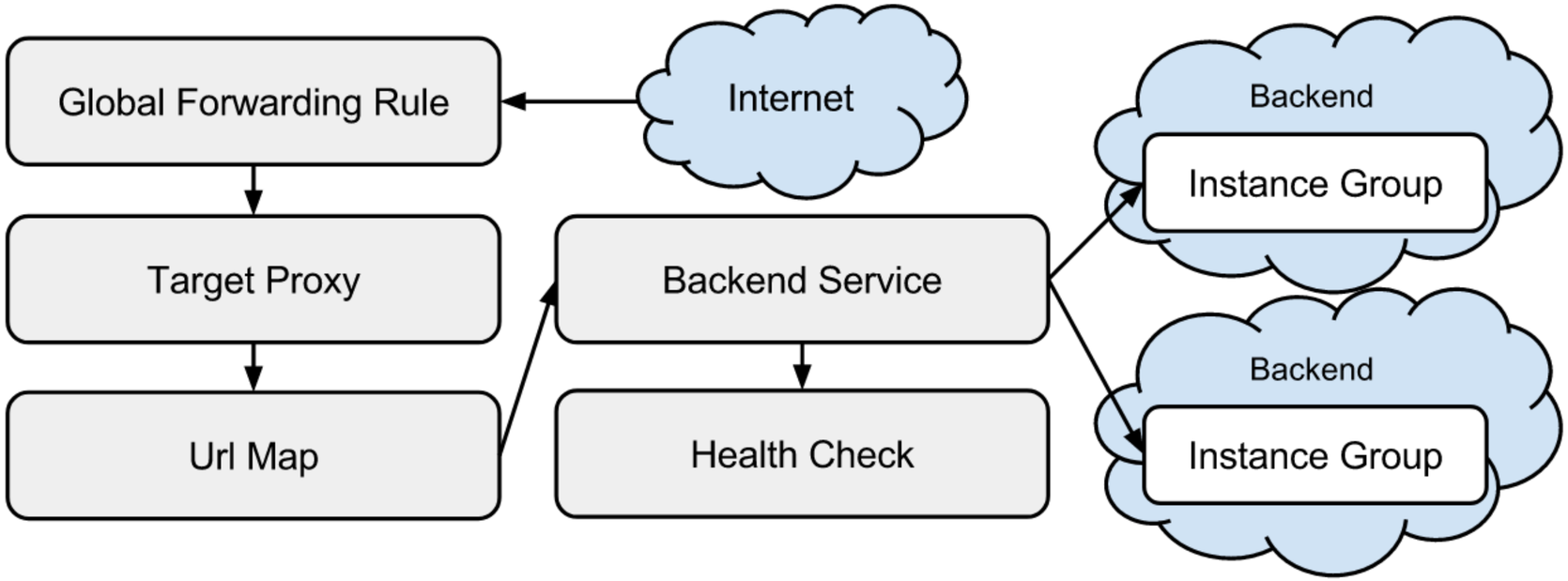
HTTP/HTTPS Load Balancing



Distributes HTTP(S) traffic among groups of instances based on:

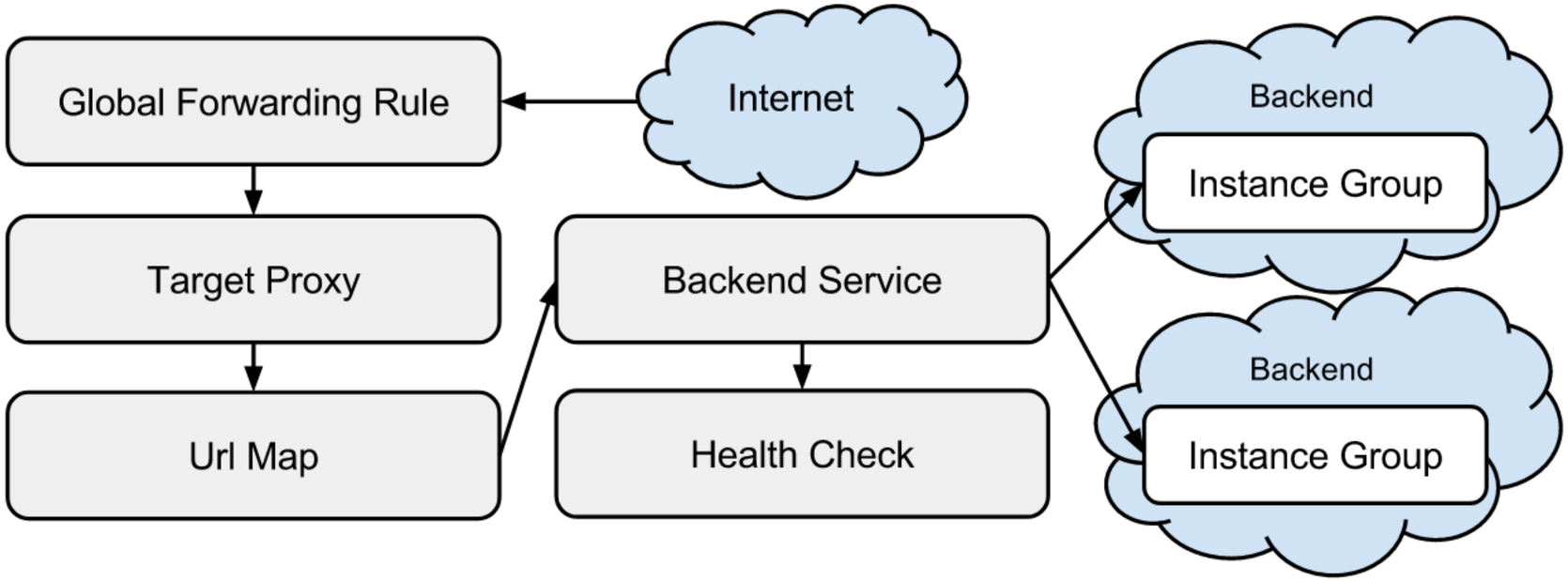
- proximity to the user
- requested URL
- or both.

HTTP/HTTPS Load Balancing



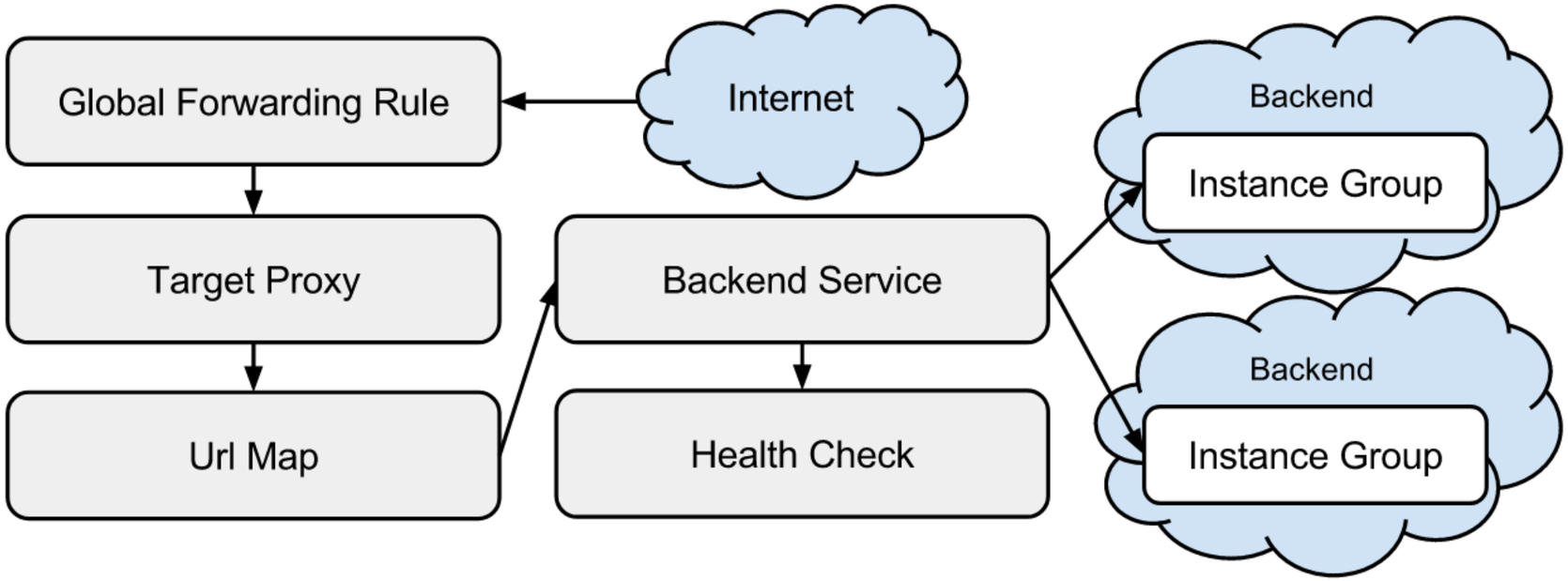
Step 1: A global forwarding rule directs incoming requests to a target HTTP proxy

HTTP/HTTPS Load Balancing



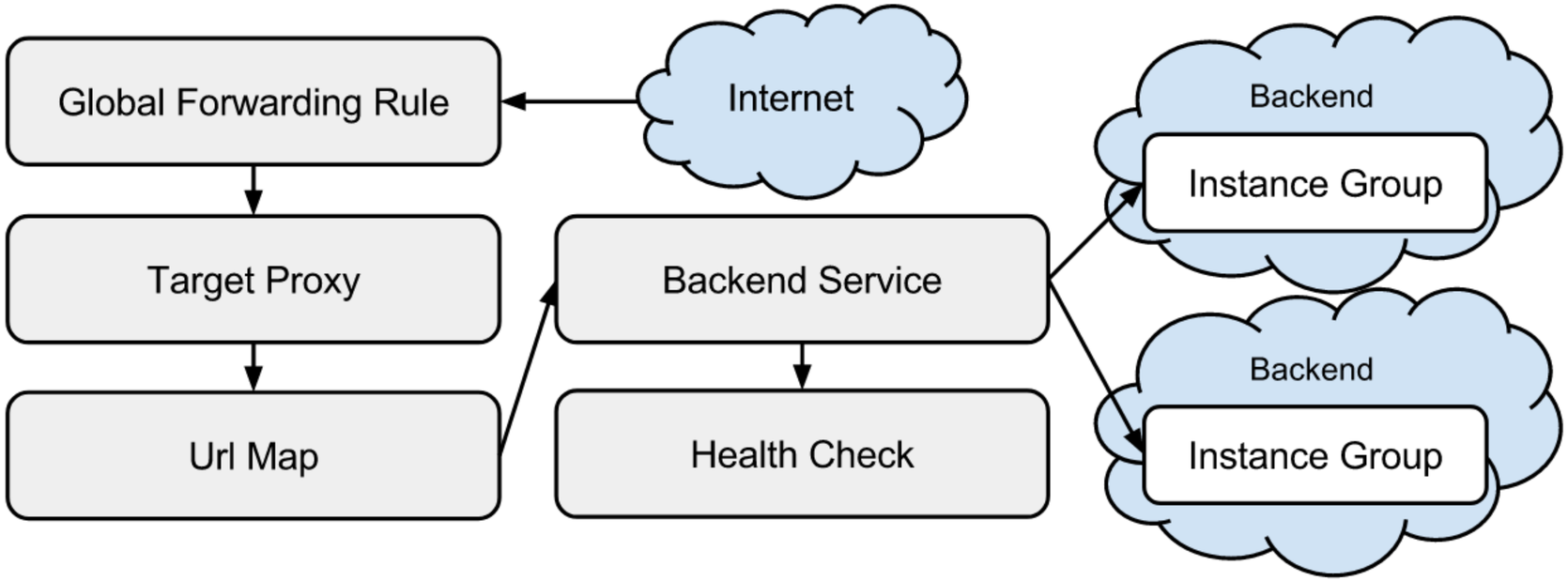
Step 2: The target HTTP proxy checks each request against a URL map to determine the appropriate backend service for the request

HTTP/HTTPS Load Balancing



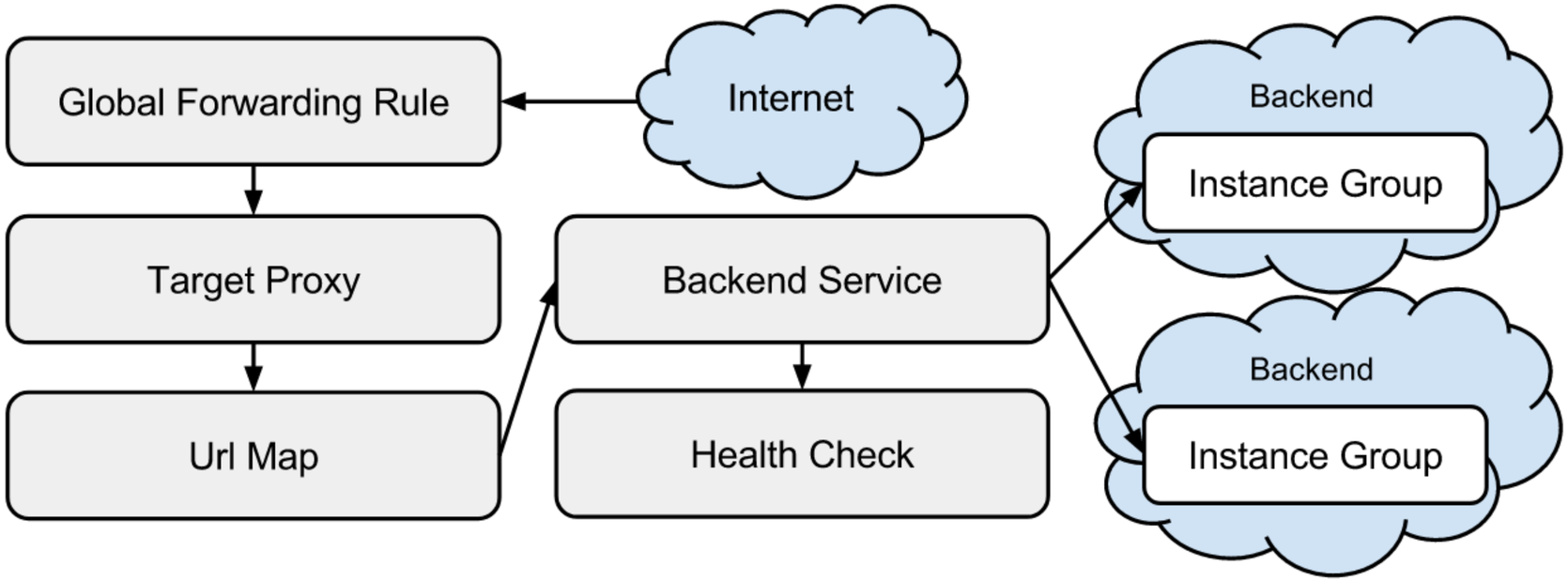
Step 3: The backend service directs each request to an appropriate backend based on serving capacity, zone, and instance health of its attached backends

HTTP/HTTPS Load Balancing



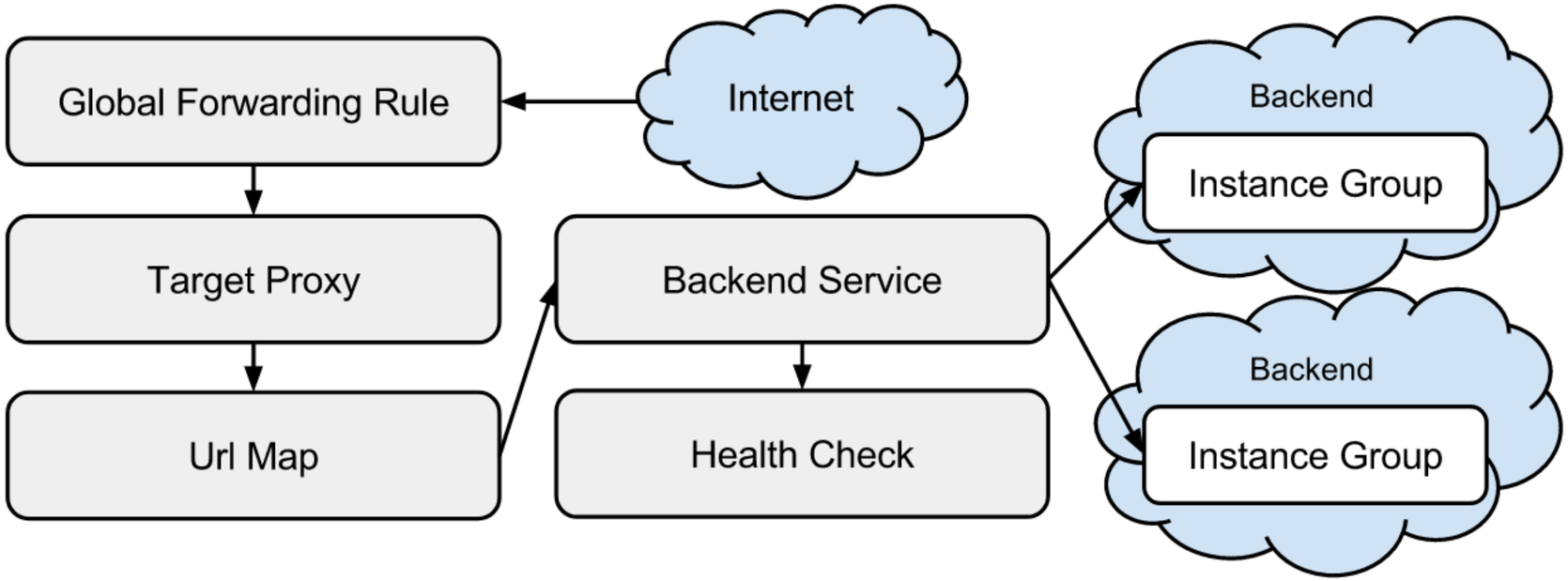
Step 3: (The health of each backend instance is verified using either an HTTP health check or an HTTPS health check - if HTTPS, request is encrypted)

HTTP/HTTPS Load Balancing



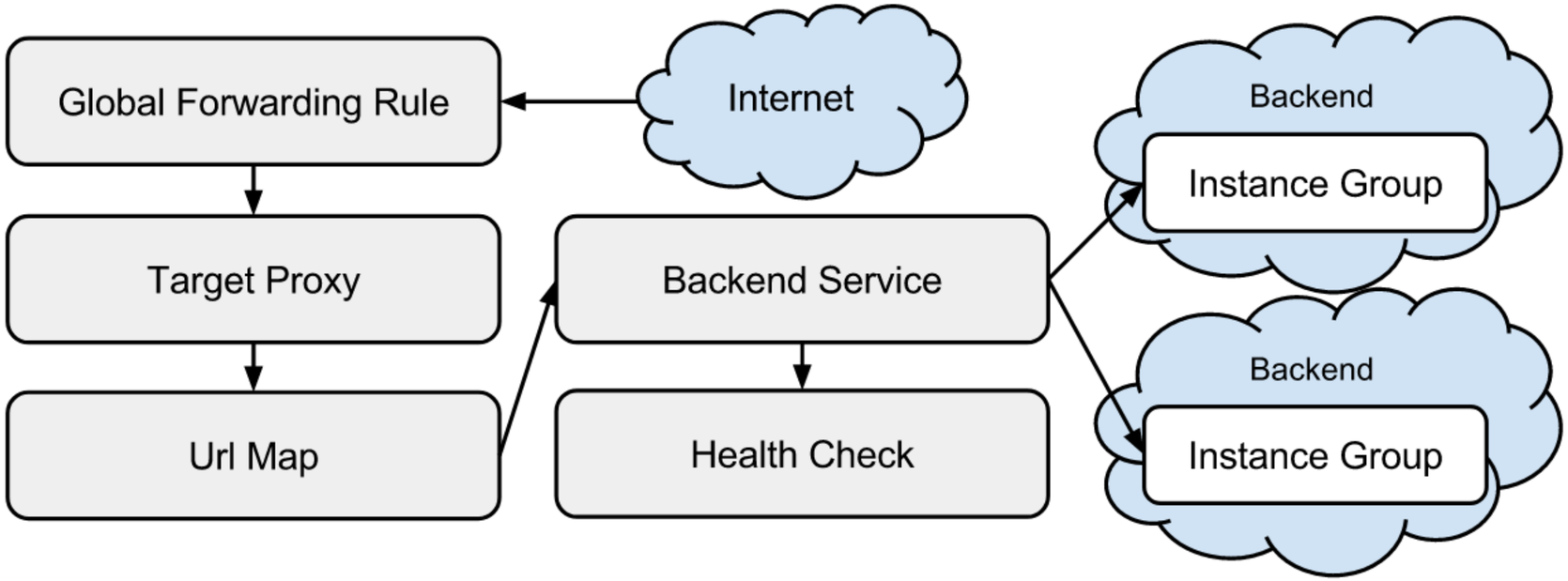
Actual request distribution can happen based on either balancing RPS (requests-per-second) or CPU utilisation - your choice

HTTP/HTTPS Load Balancing



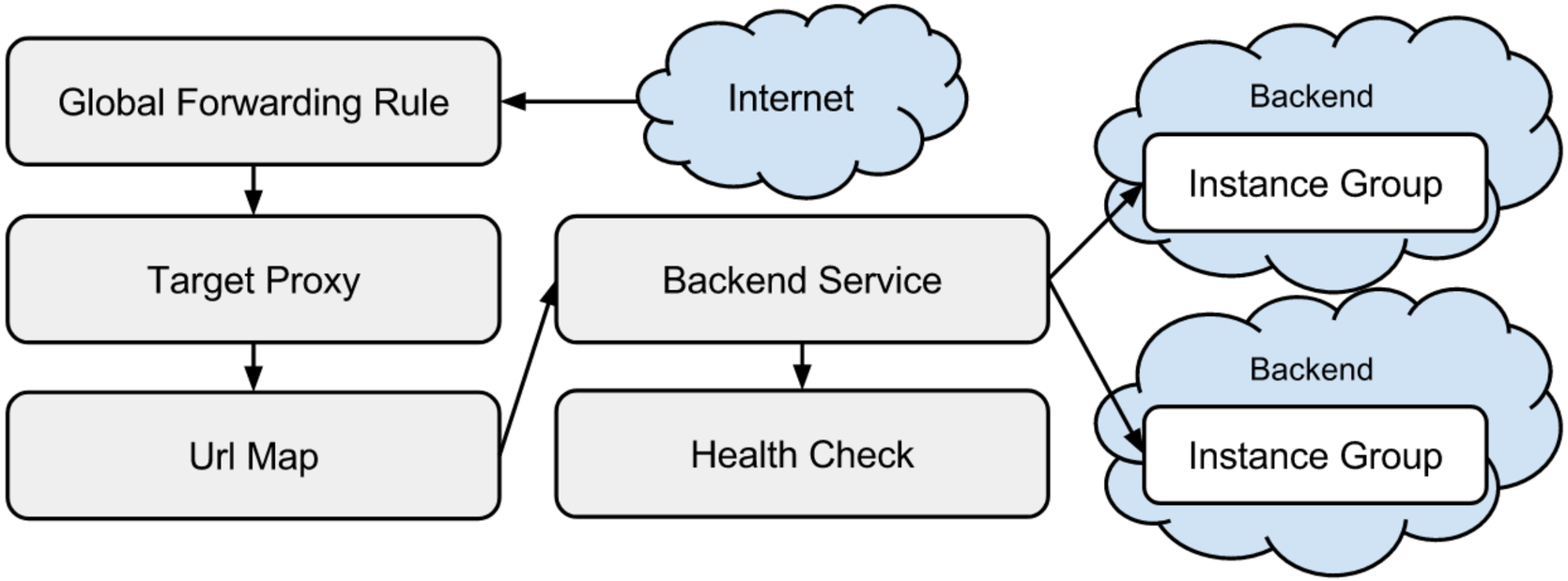
HTTPS load balancing has a signed SSL certificate for load balancer

HTTP/HTTPS Load Balancing



BTW, must create firewall rules to allow requests from load balancer and health checker to get through to the instances

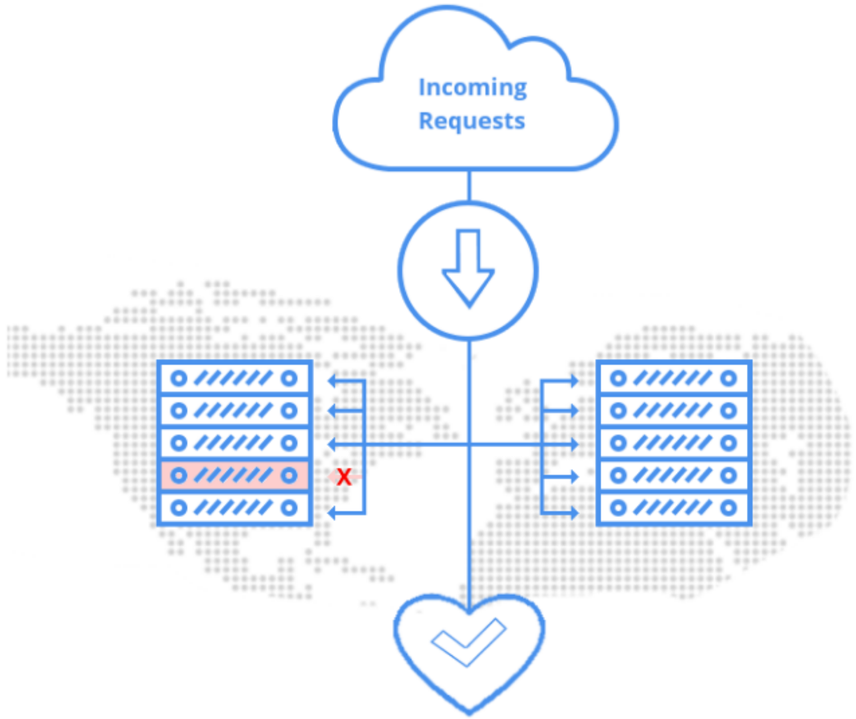
HTTP/HTTPS Load Balancing



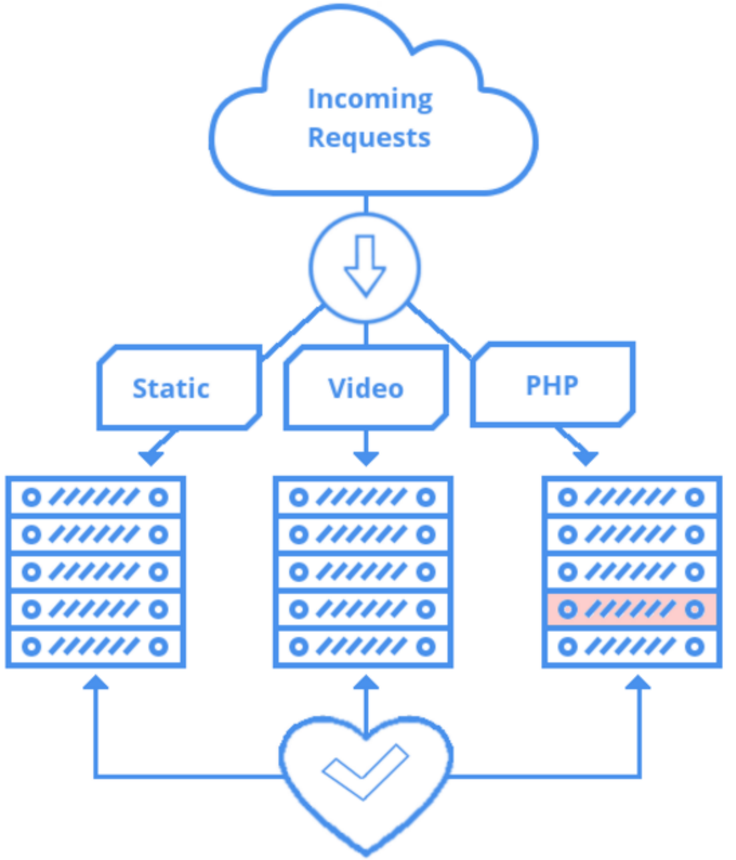
Session affinity: All requests from same client to same server based on either

- client IP
- cookie

HTTP/HTTPS Load Balancing

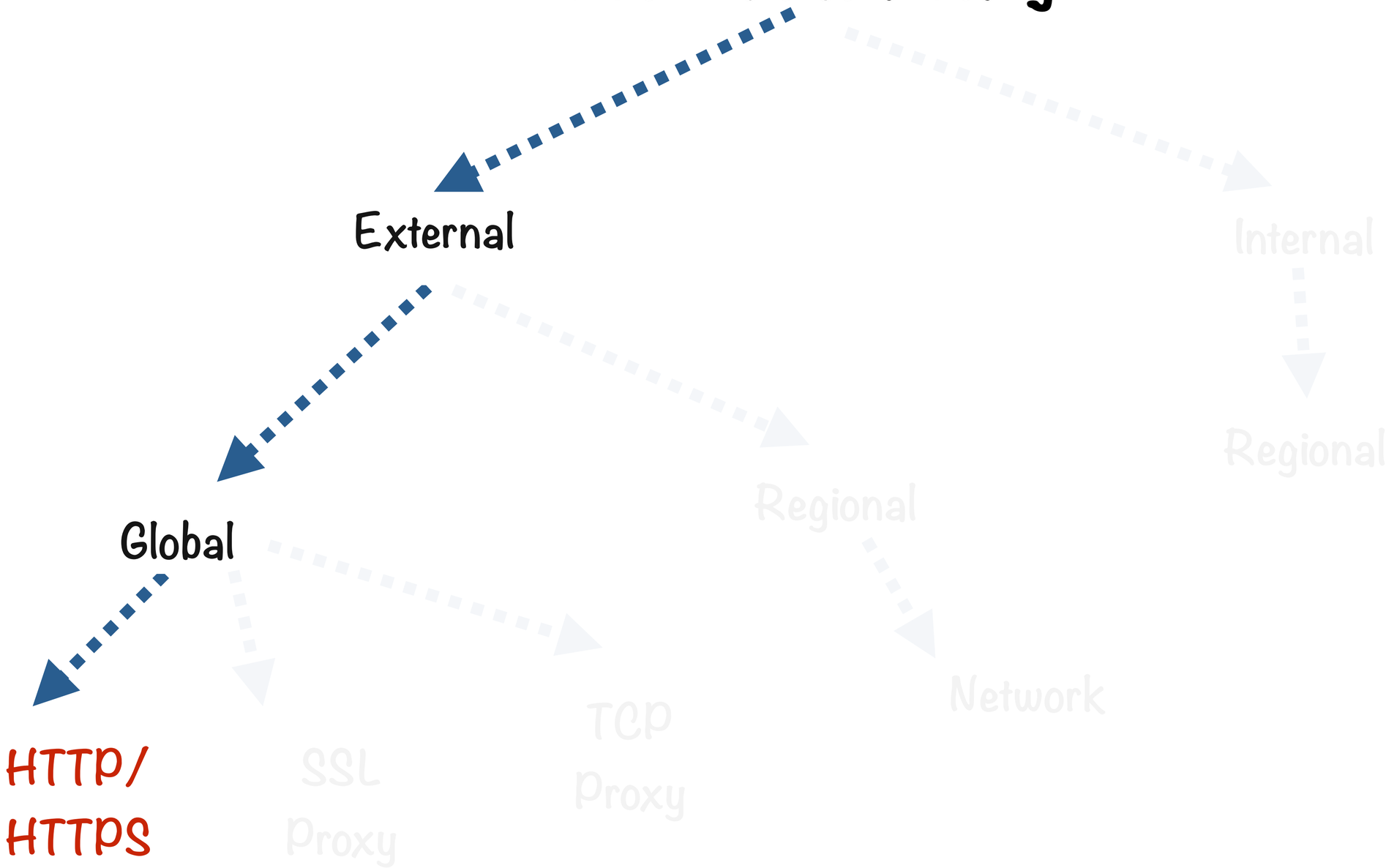


Cross-Regional



Content-based

Load Balancing



Load Balancing

OSI Network Stack

User
Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

HTTP/HTTPS

SSL Proxy

TCP Proxy

Network

HTTP(S) load balancing is the "smartest"

OSI Network Stack

User
Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

HTTP/HTTPS

SSL Proxy

TCP Proxy

Network

SSL operates in the session layer

SSL Proxy Load Balancing

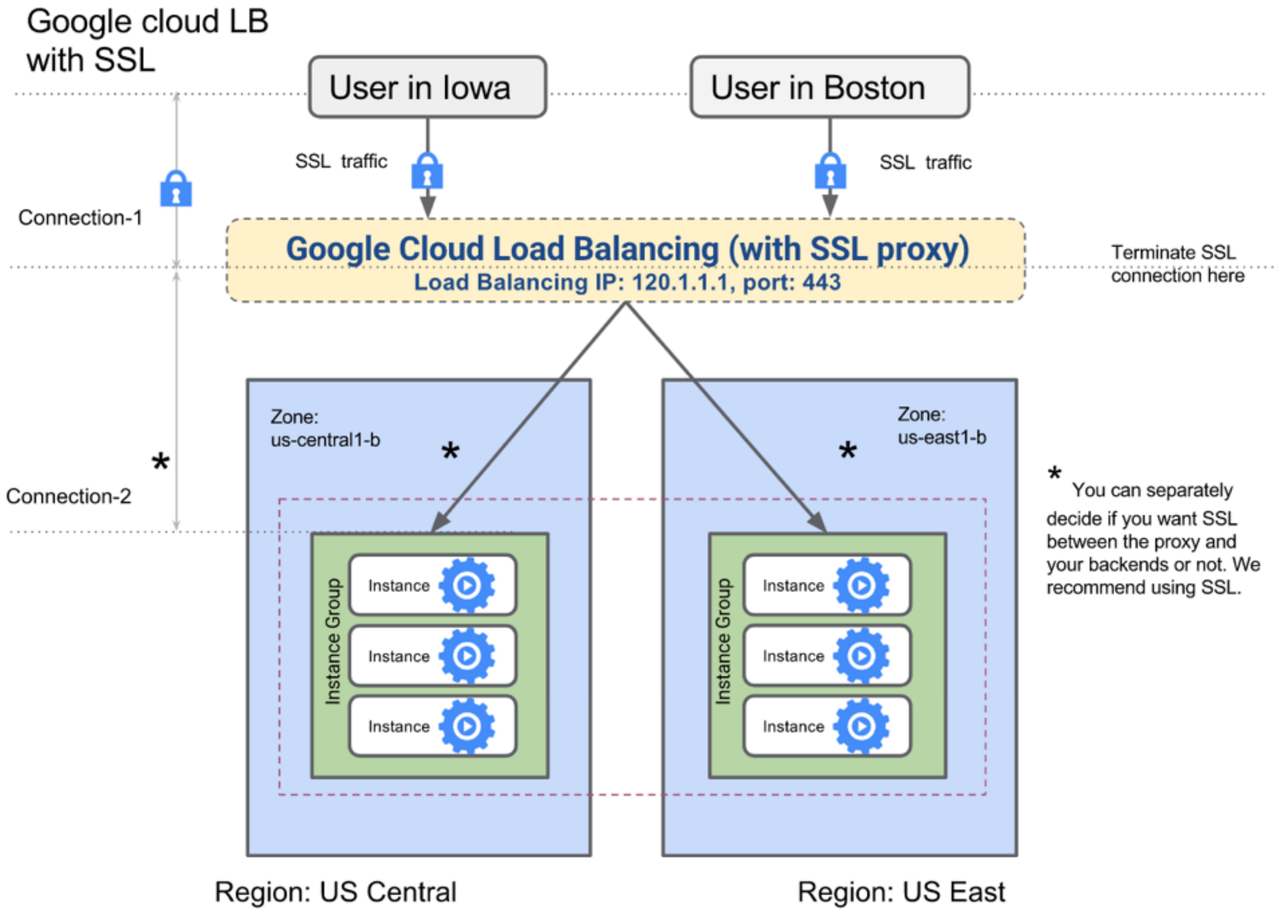
- Remember the OSI network layer stack: physical, data link, network, transport, session, presentation, application?
- The usual combination is TCP/IP: network = IP, transport = TCP, application = HTTP
- For secure traffic: add session layer = SSL (secure socket layer), and application layer = HTTPS

SSL Proxy Load Balancing

- Use only for non-HTTP(S) SSL traffic
- For HTTP(S), just use HTTP(S) load balancing
- SSL connections are terminated at the global layer then proxied to the closest available instance group

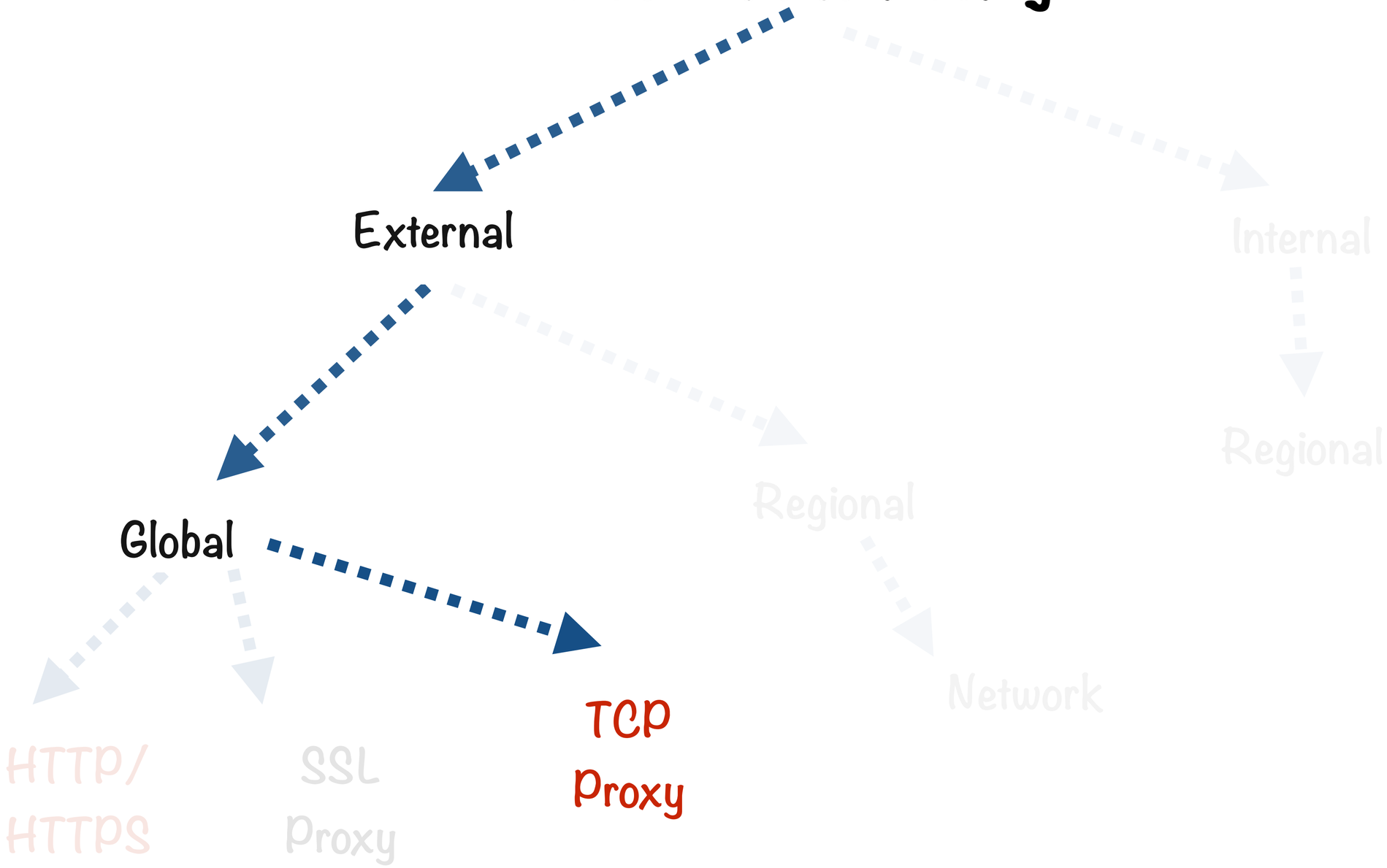
SSL Proxy Load Balancing

- Notice how 2 connections exist per user
- Connection 1 - terminated at global layer
- Connection 2 - separate link to selected backed instance



Load Balancing

Load Balancing



OSI Network Stack

User
Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer

HTTP/HTTPS

SSL Proxy

TCP Proxy

Network

OSI Network Stack

User	
Application Layer	HTTP/HTTPS
Presentation Layer	
Session Layer	SSL Proxy
Transport Layer	TCP Proxy
Network Layer	Network
Data Link Layer	
Physical Layer	

TCP Proxy Load Balancing

- Perform load balancing based on transport layer (TCP)
- Allows you to use a single IP address for all users around the world.
- Automatically routes traffic to the instances that are closest to the user.

TCP Proxy Load Balancing

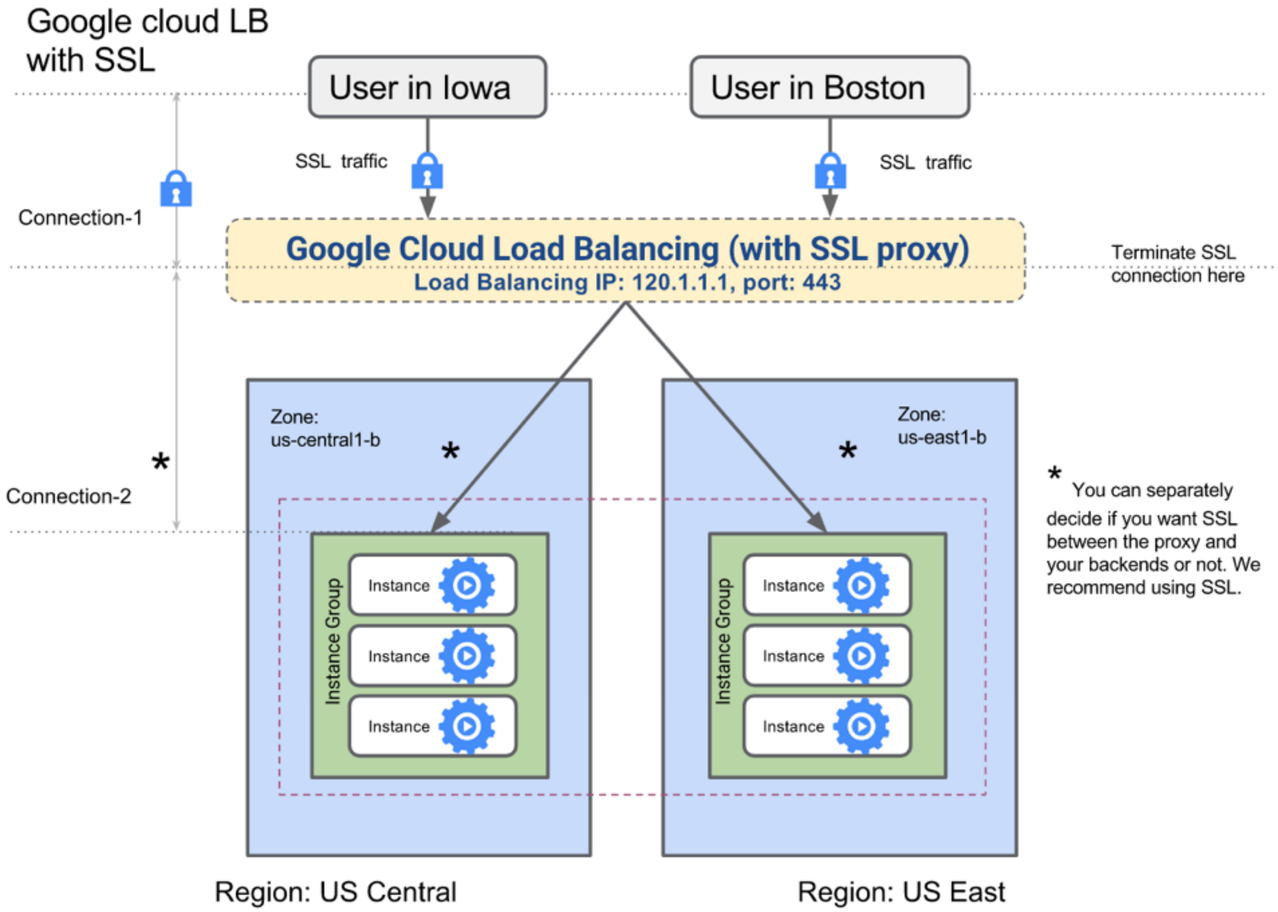
- Advantage of transport layer load balancing:
  - more intelligent routing possible than with network layer load balancing
  - better security - TCP vulnerabilities can be patched at the load balancer

TCP Proxy Load Balancing

- Do NOT use for HTTP(S)
- Use for specific ports only
  - 25, 43, 110, 143, 195, 443, 465, 587, 700, 993, 995

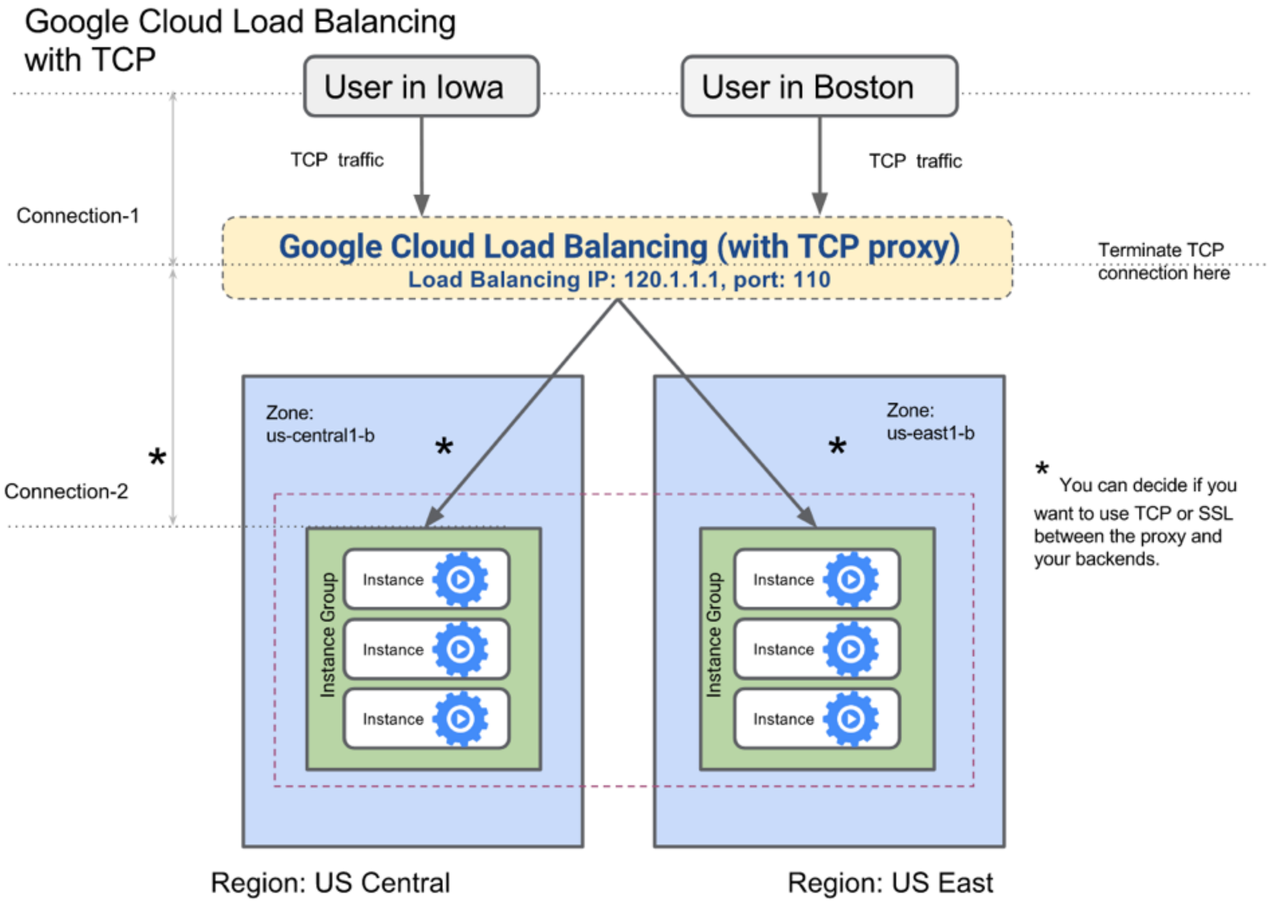
SSL Proxy Load Balancing

- Notice how 2 connections exist per user
- Connection 1 - terminated at global layer
- Connection 2 - separate link to selected backend instance

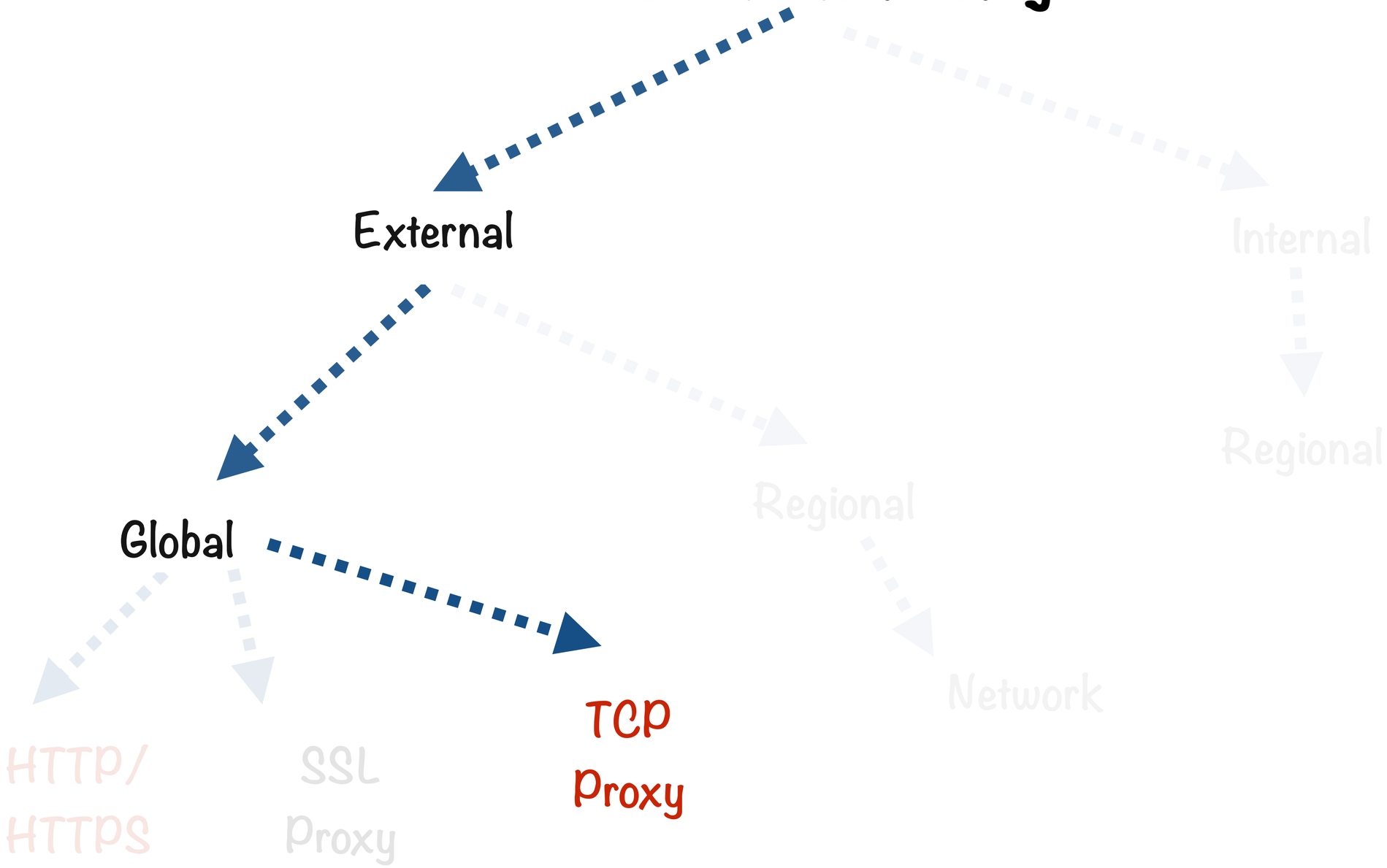


TCP Proxy Load Balancing

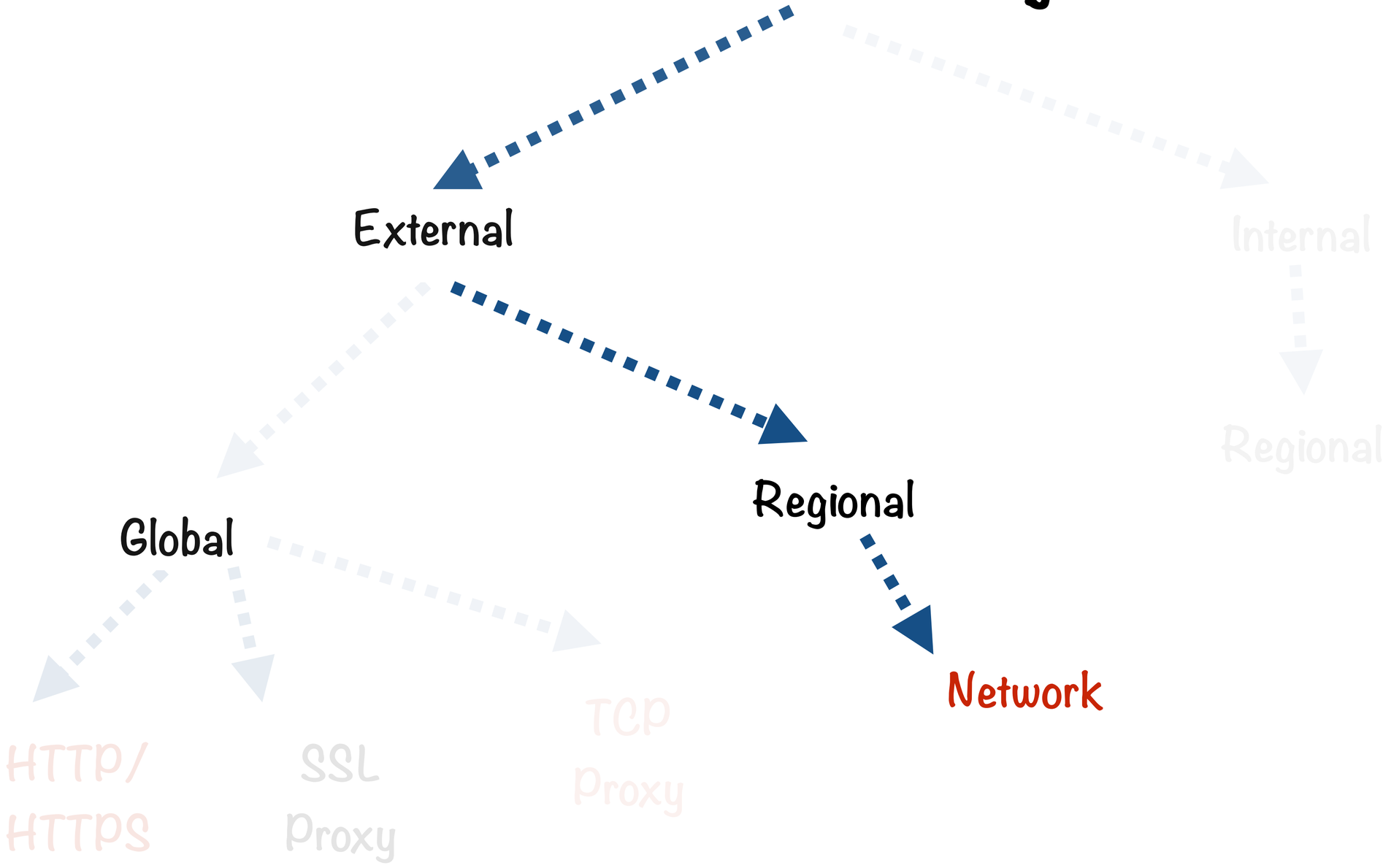
- Notice how 2 connections exist per user
- Connection 1 - terminated at global layer
- Connection 2 - separate link to selected backend instance



Load Balancing



Load Balancing



OSI Network Stack

User	
Application Layer	HTTP/HTTPS
Presentation Layer	
Session Layer	SSL Proxy
Transport Layer	TCP Proxy
Network Layer	Network
Data Link Layer	
Physical Layer	

OSI Network Stack

User	
Application Layer	HTTP/HTTPS
Presentation Layer	
Session Layer	SSL Proxy
Transport Layer	TCP Proxy
Network Layer	Network
Data Link Layer	
Physical Layer	

Network Load Balancing

- Based on incoming IP protocol data, such as address, port, and protocol type
- Pass-through load balancer - does not proxy connections from clients
- Use it to load balance UDP traffic, and TCP and SSL traffic on ports that are not supported by the SSL proxy and TCP proxy load balancers

Load Balancing Algorithm

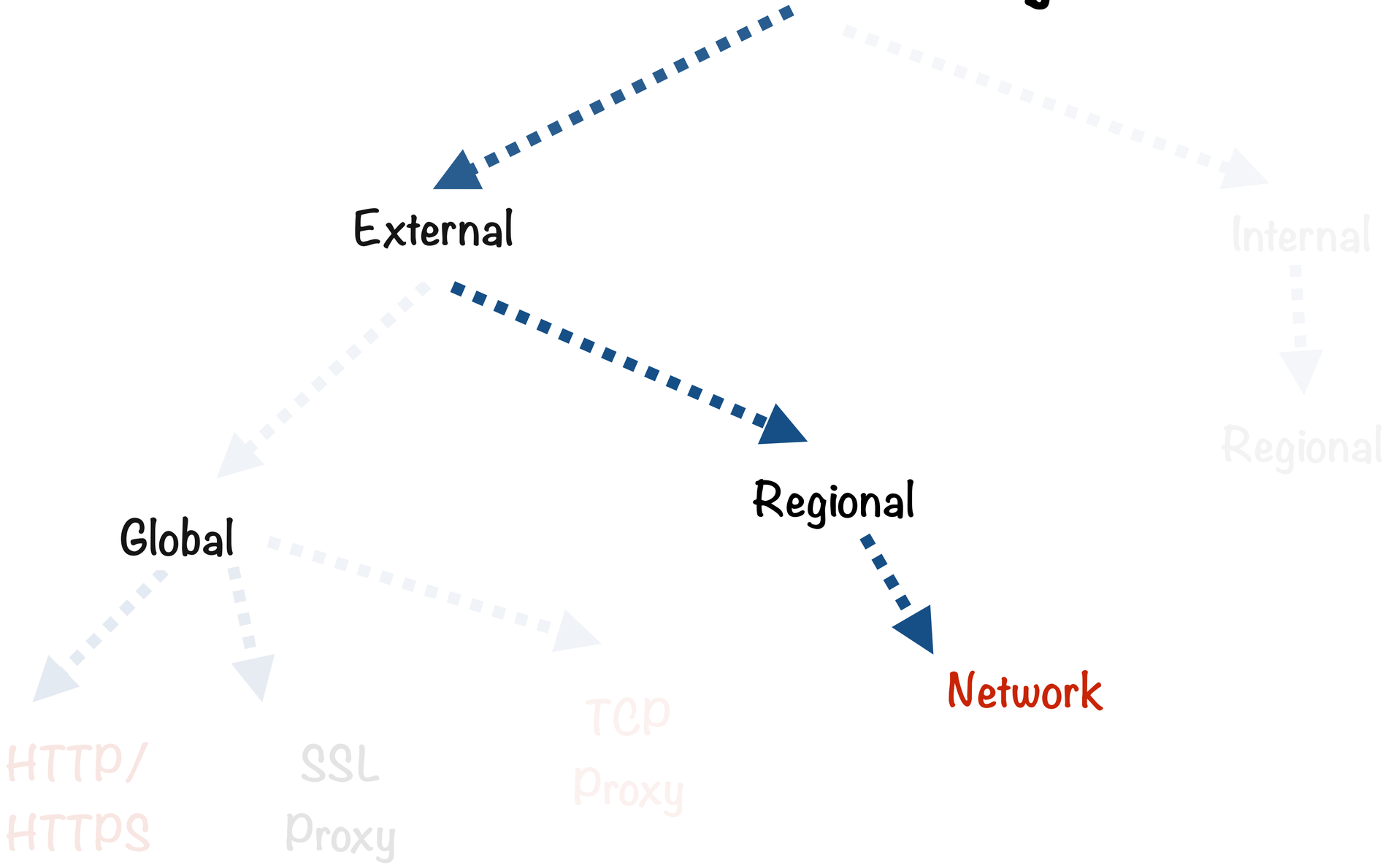
- Picks an instance based on a hash of the source IP and port, destination IP and port, and protocol

- This means that incoming TCP connections are spread across instances and each new connection may go to a different instance.

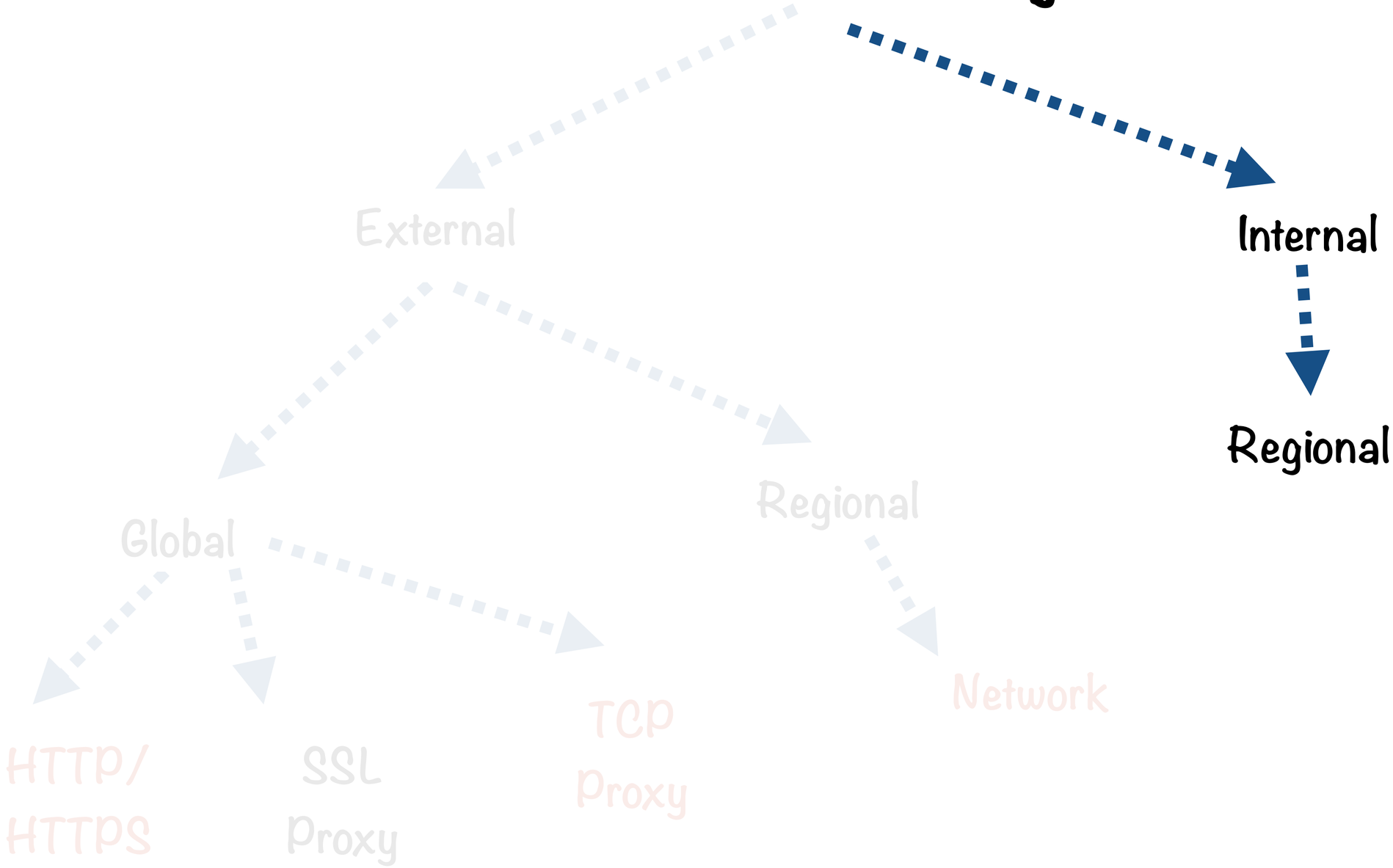
- Regardless of the session affinity setting, all packets for a connection are directed to the chosen instance until the connection is closed and have no impact on load balancing decisions for new incoming connections

- This can result in imbalance between backends if long-lived TCP connections are in use.

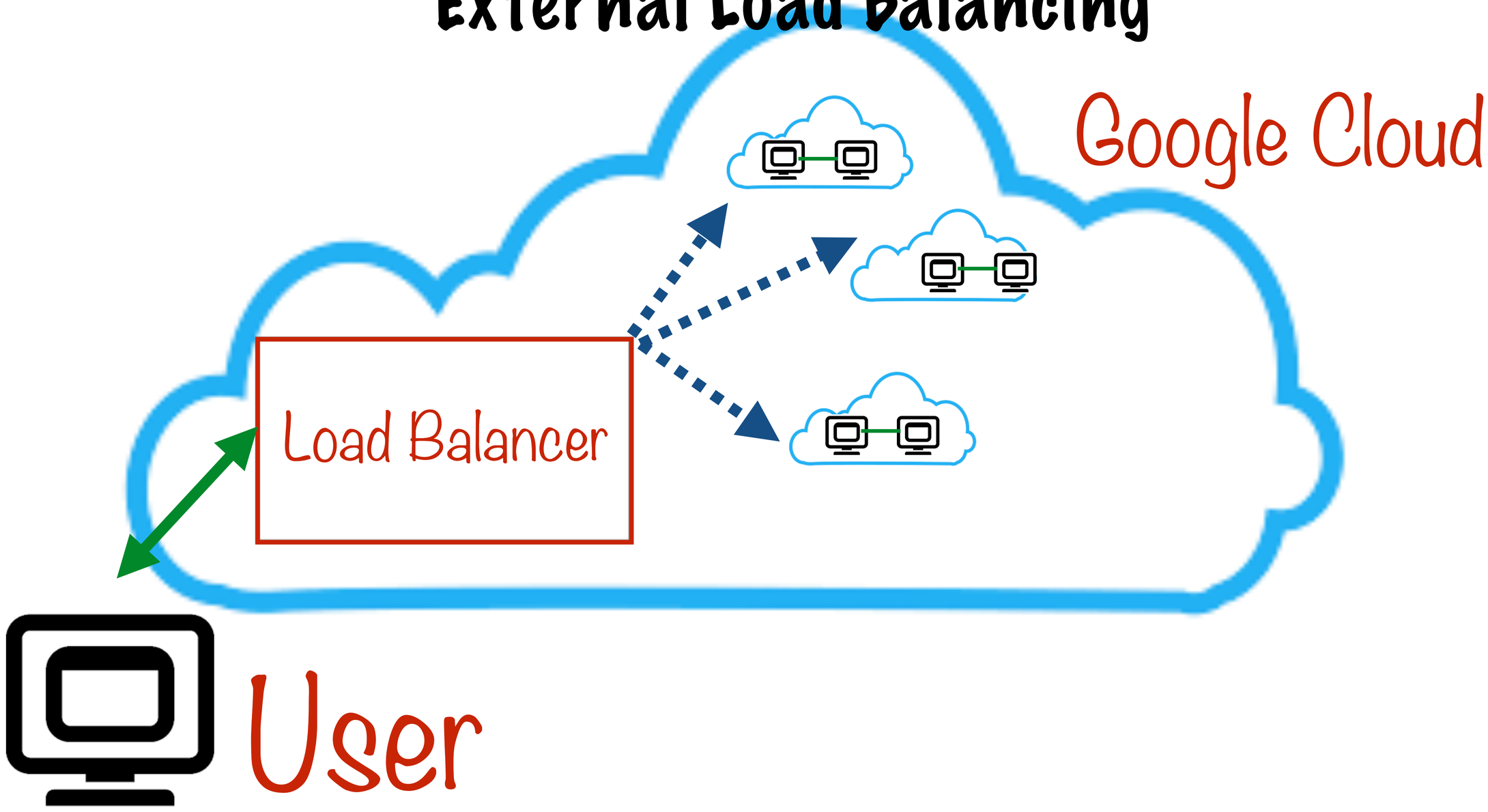
Load Balancing



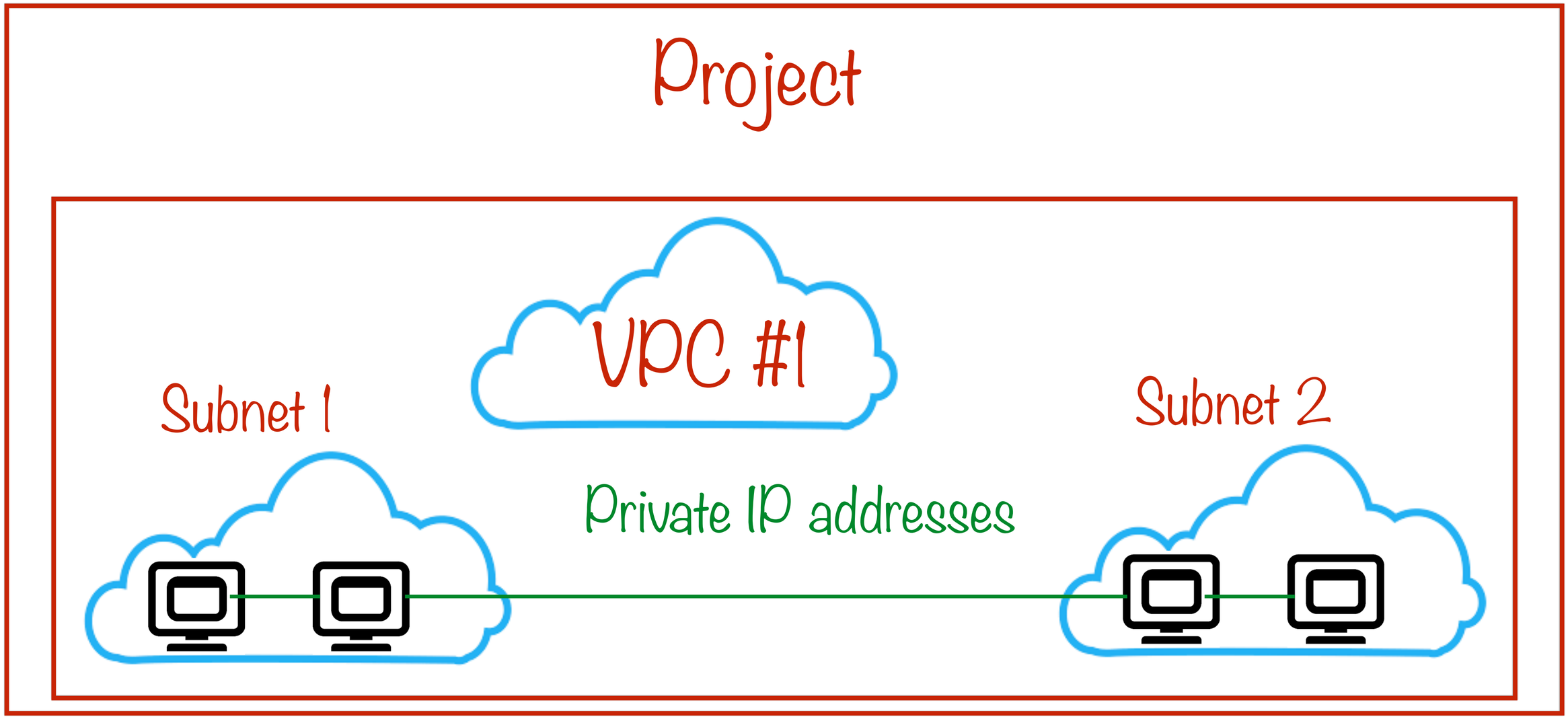
Load Balancing



External Load Balancing



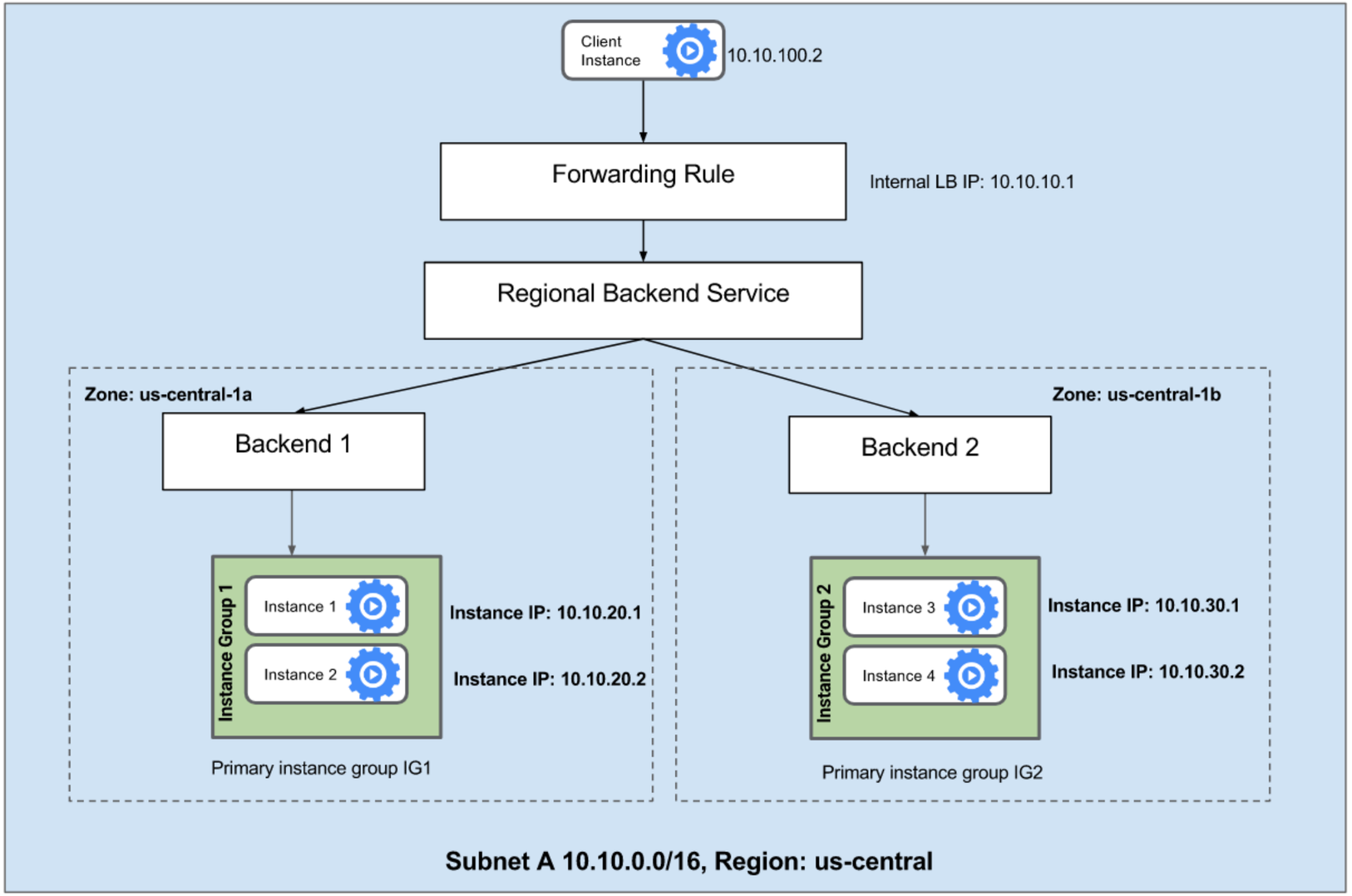
Internal Load Balancing



Internal Load Balancing

- So that your own instances don't need to use external IP addresses to take advantage of load balancing
- Private load balancing IP address that only your VPC instances can access
- VPC traffic stays internal - less latency, more security

Internal Load Balancing



Load Balancing Algorithm

- The backend instance for a client is selected using a hashing algorithm that takes instance health into consideration.

- Using a 5-tuple hash, five parameters for hashing:

    - client source IP
    - client port
    - destination IP (the load balancing IP)
    - destination port
    - protocol (either TCP or UDP)

Load Balancing Algorithm

- Introduce session affinity by hashing on only some of the 5 parameters
  - Hash based on 3-tuple (Client IP, Dest IP, Protocol)
  - Hash based on 2-tuple (Client IP, Dest IP)

Health Checks

- HTTP, HTTPS health checks: If your traffic is HTTP(S), then HTTP(S) health checks provide the highest fidelity check because they verify that the web server is up and serving traffic, not just that the instance is healthy.

- TCP health checks: Configure the SSL health checks if your traffic is not HTTPS but is encrypted via SSL(TLS)

- SSL (TLS) health checks: For all TCP traffic that is not HTTP(S) or SSL(TLS), you can configure a TCP health check

Traditional (Proxy) Internal Load Balancing

- Configure an internal IP on a load balancing device or instance(s) and your client instance connects to this IP
- Traffic coming to the IP is terminated at the load balancer
- The load balancer selects a backend and establishes a new connection to it
- In effect, there are two connections: Client<->Load Balancer and Load Balancer<->Backend.

GCP Internal Load Balancing

- Not proxied - differs from traditional model
- lightweight load-balancing built on top of Andromeda network virtualization stack
- provides software-defined load balancing that directly delivers the traffic from the client instance to a backend instance

Overview

Resources in GCP projects are split across VPCs (Virtual Private Clouds)

VPCs support various networking features such as DNS, Load balancing, direct peering, VPNs and Firewalls

GCP has 3rd party partnerships for CDNs and ISPs

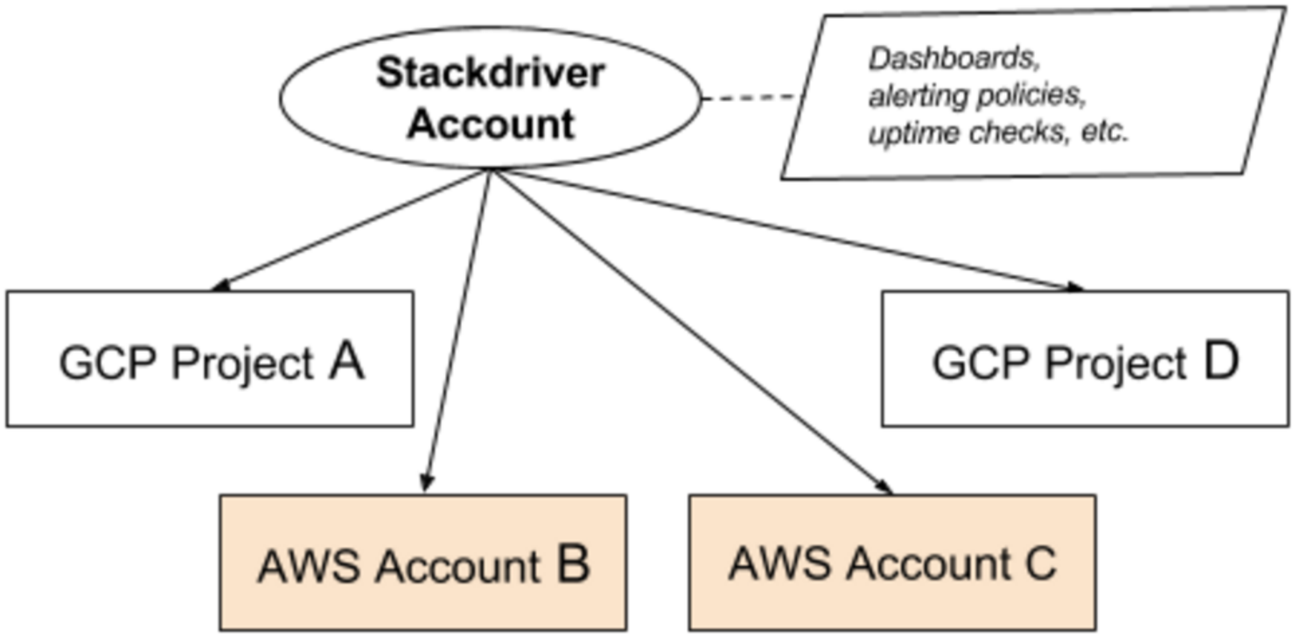
Several complex forms of load balancing are available at different levels of the OSI stack

Module - Operations

Summary

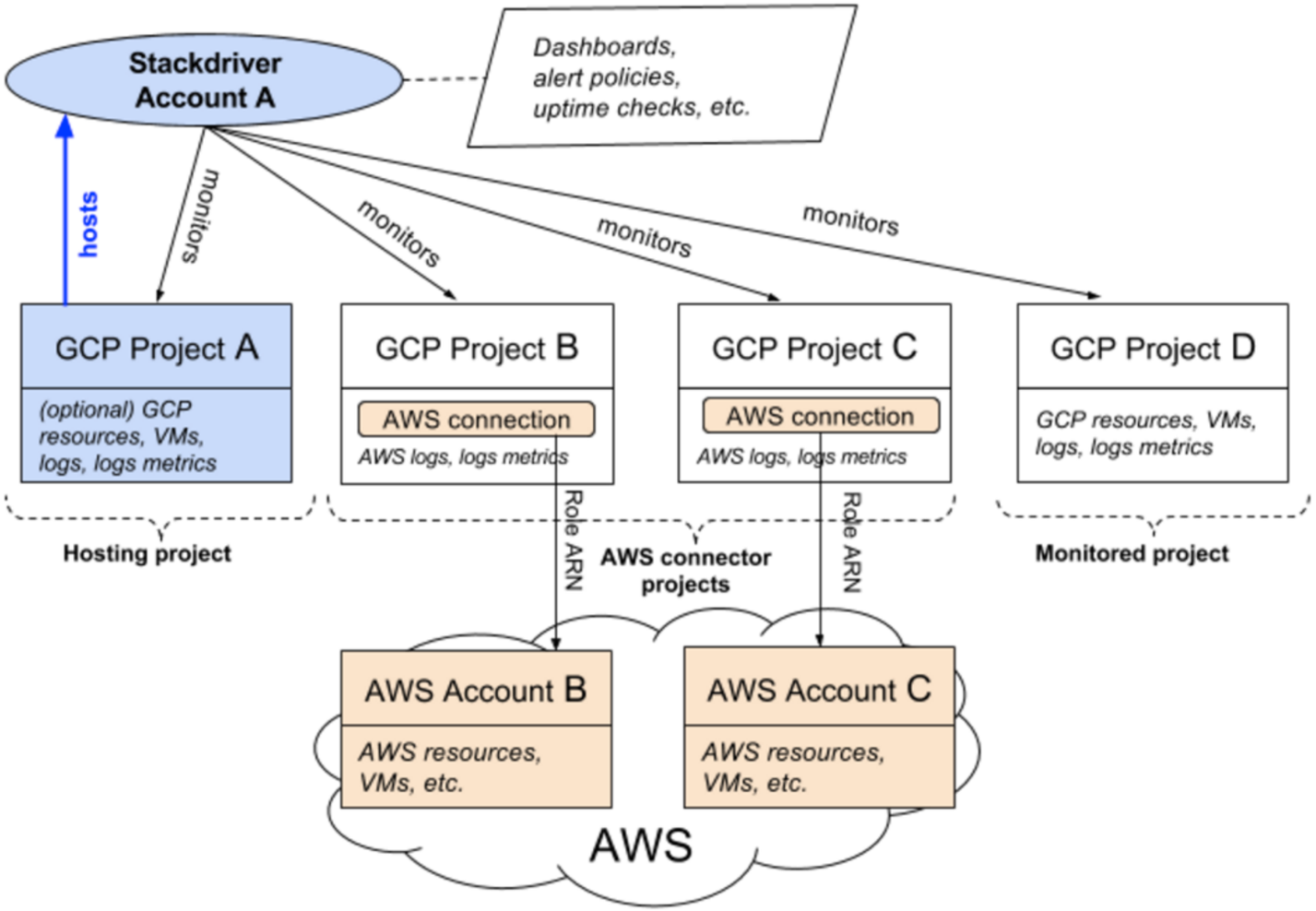
Stackdriver Monitoring

StackDriver Accounts



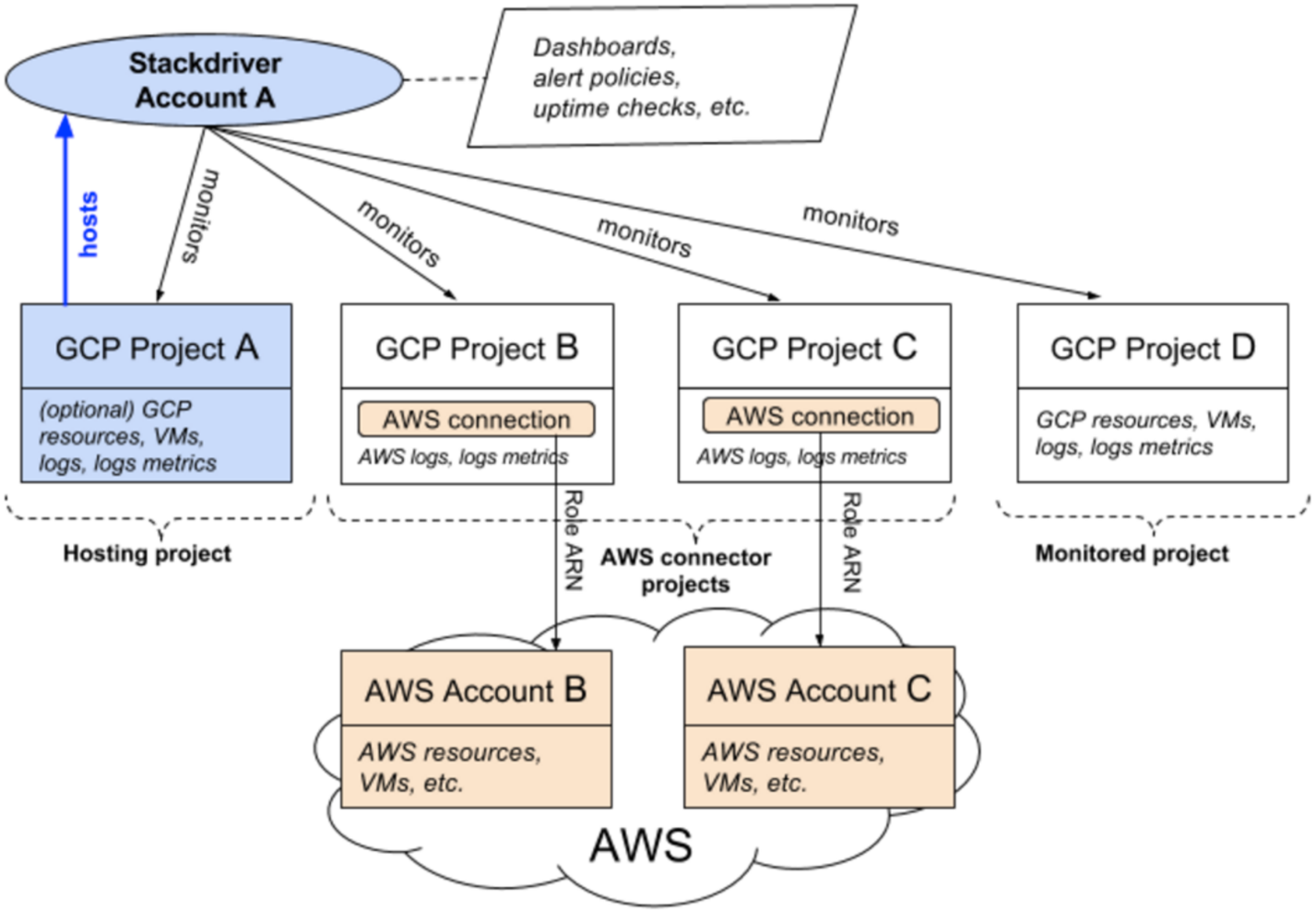
A Stackdriver account holds monitoring and other configuration information for a group of GCP projects and AWS accounts that are monitored together

Types of Monitored Projects



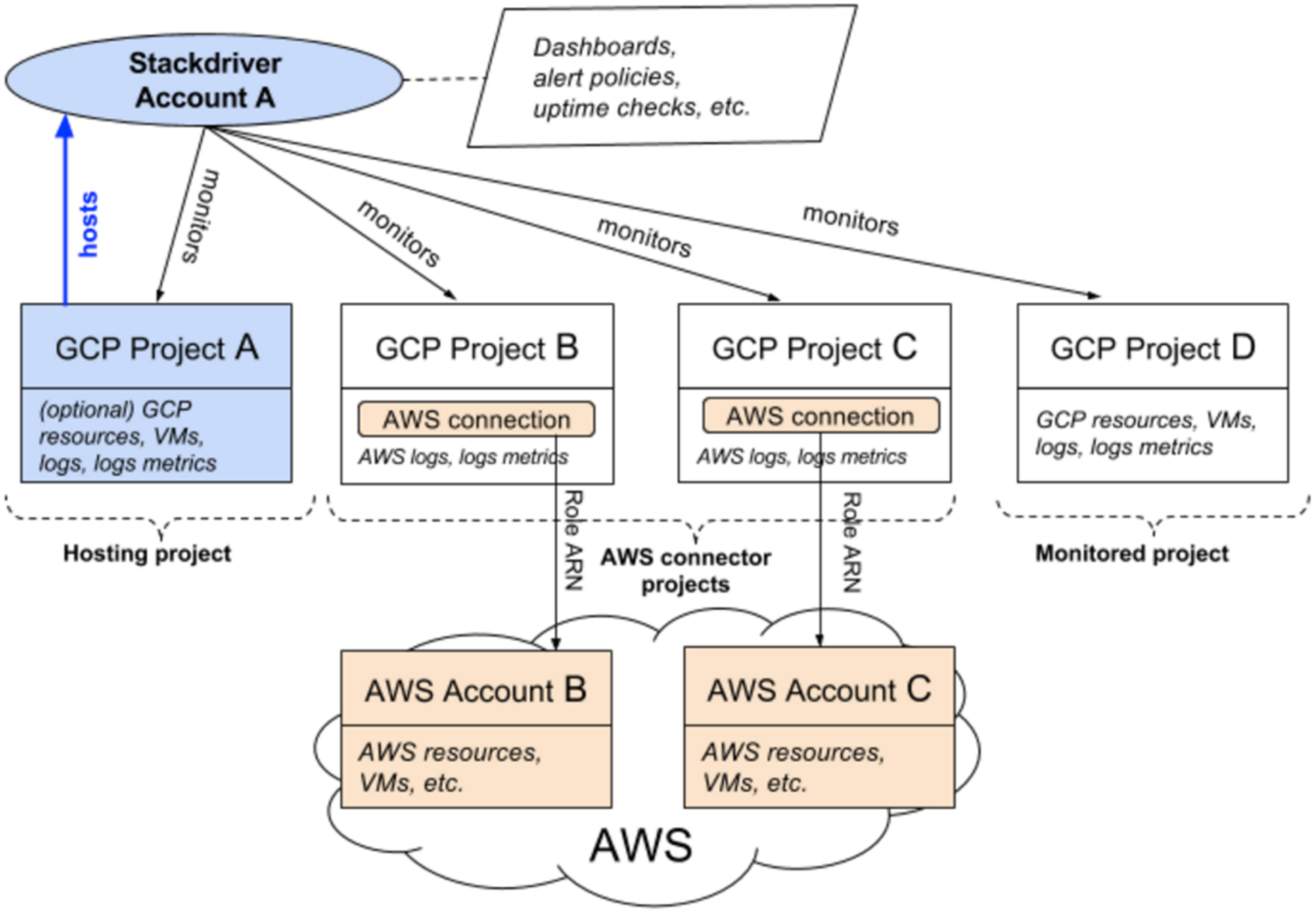
- Hosting Projects: holds the monitoring configuration for the Stackdriver account — the dashboards, alert policies, uptime checks, and so on

Types of Monitored Projects



- To monitor a single GCP project, create new StackDriver account within that 1 project

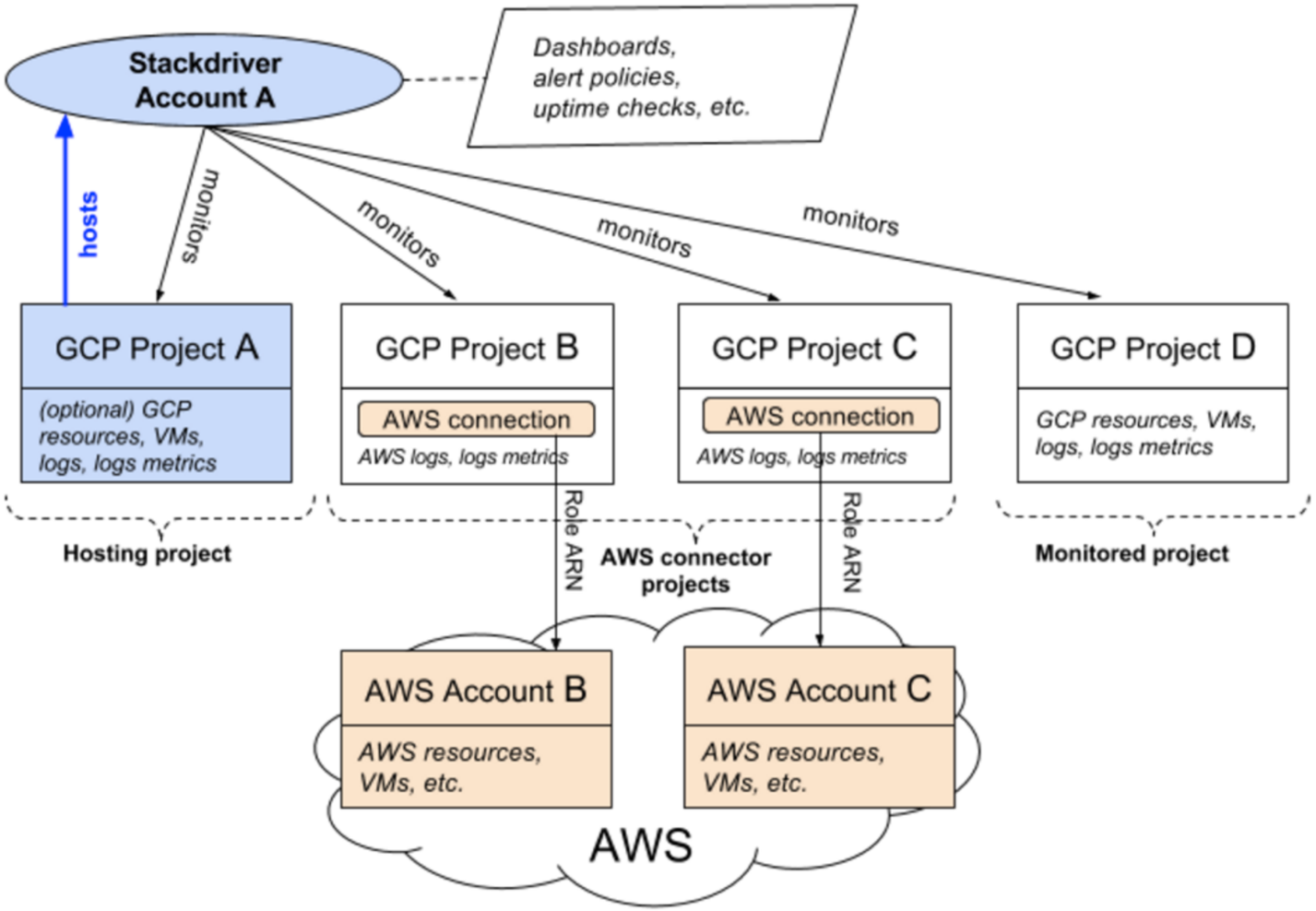
Types of Monitored Projects



- To monitor multiple GCP projects, create new StackDriver account in an otherwise empty hosting project

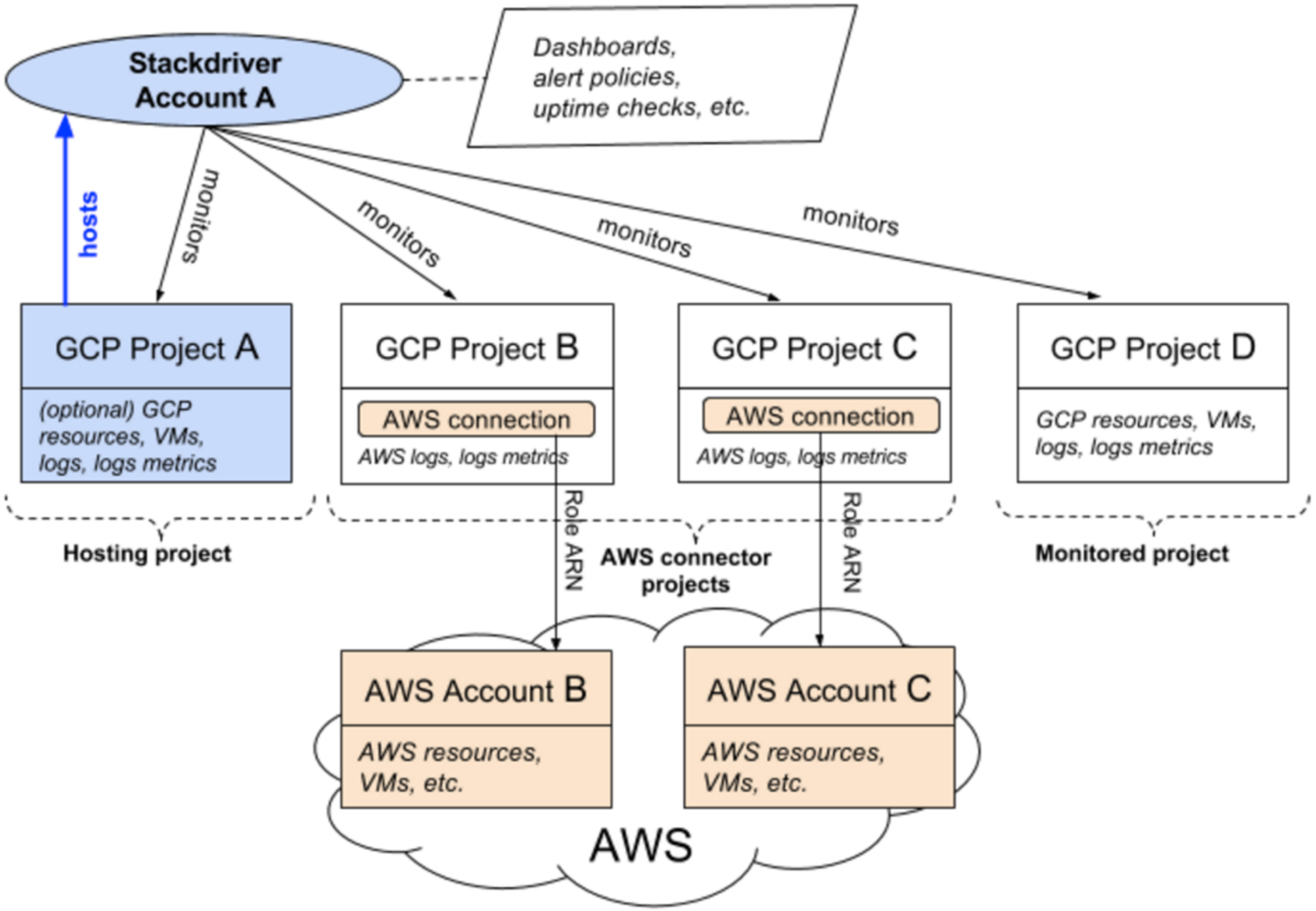
  - Don't use hosting project for any other purpose

Types of Monitored Projects



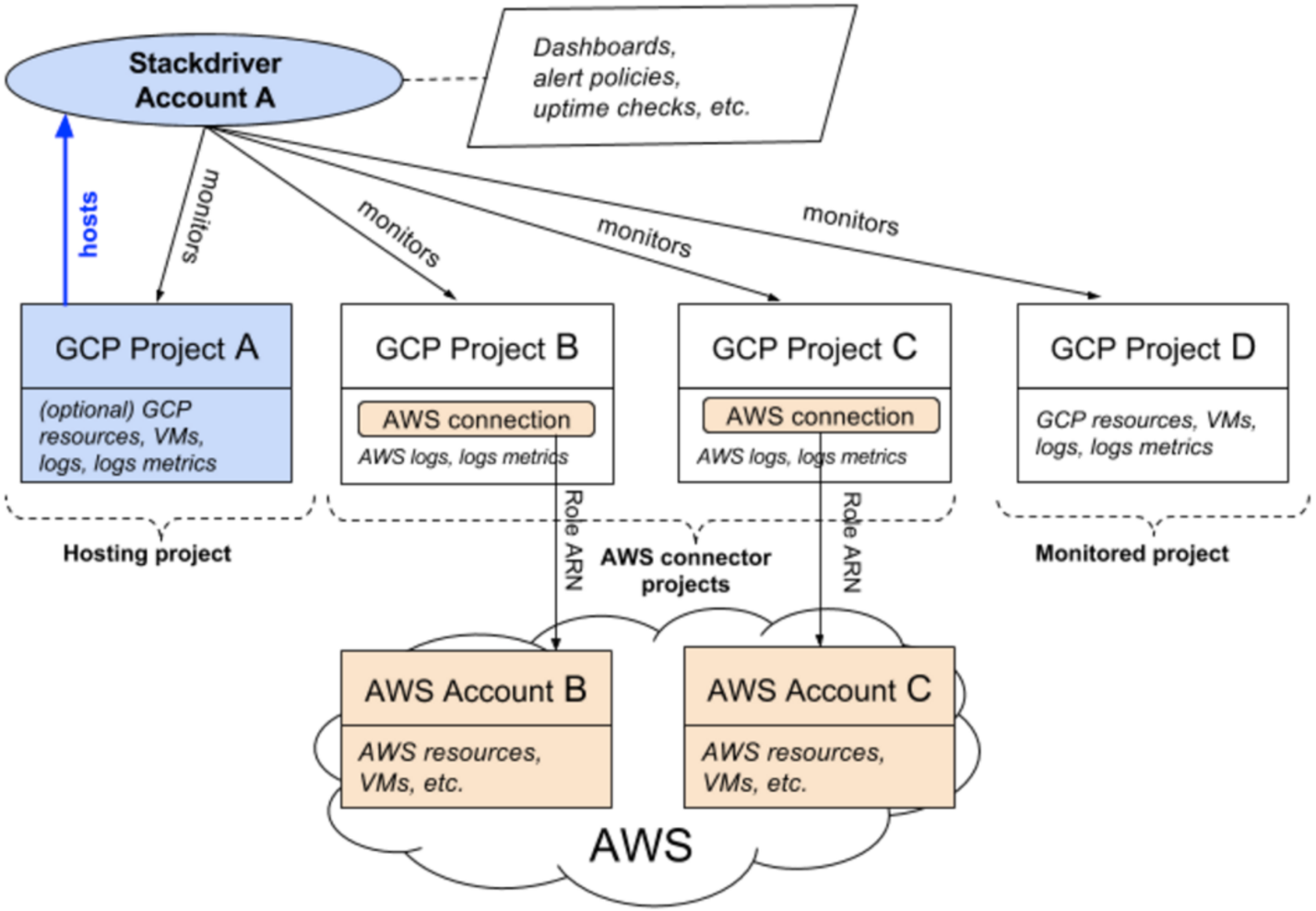
- AWS Connector Projects: When you add an AWS account to a Stackdriver account, Stackdriver Monitoring creates the AWS connector project for you, typically giving it a name beginning AWS Link.

Types of Monitored Projects



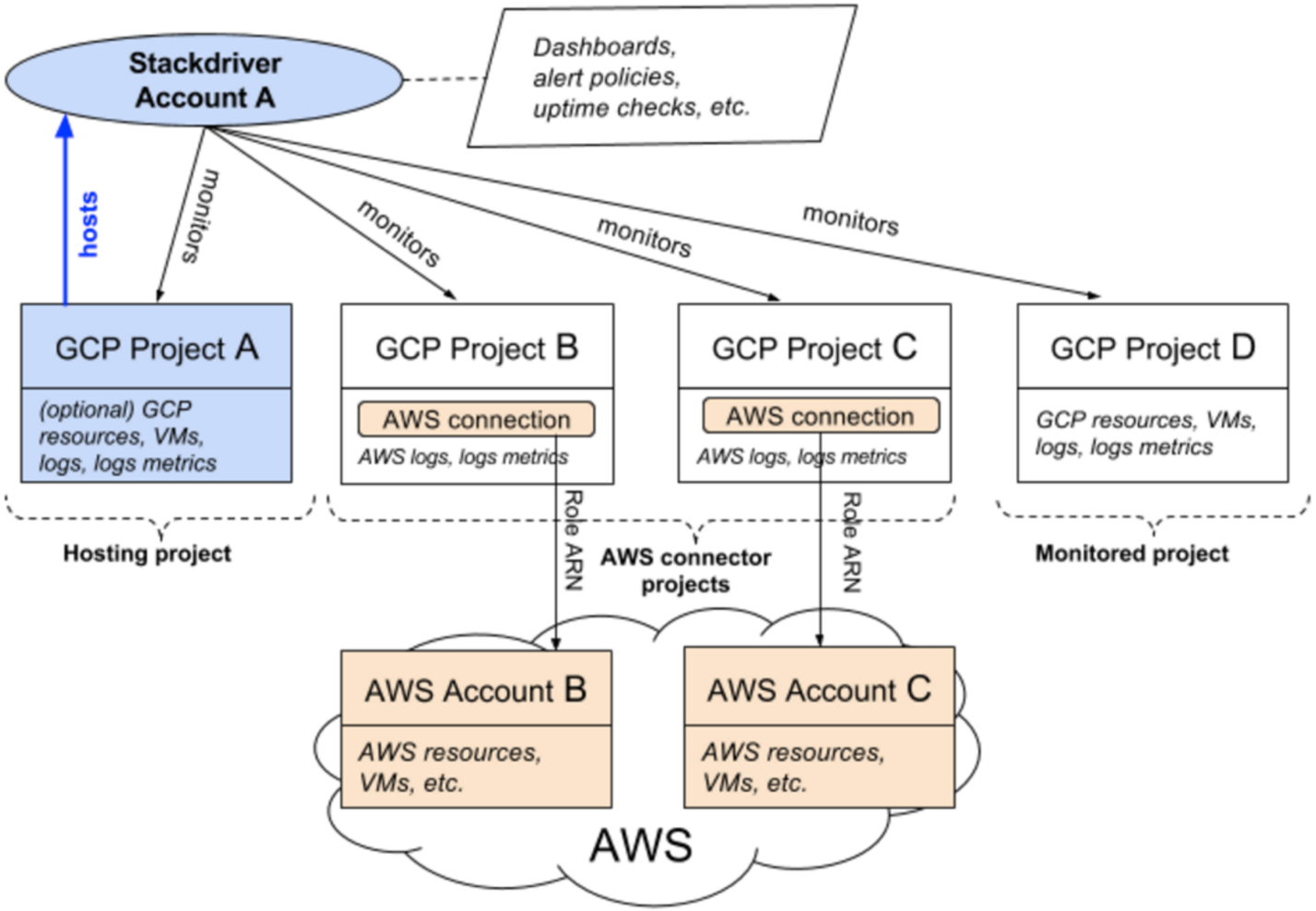
- The Monitoring and Logging agents on your EC2 instances send their metrics and logs to this connector project

Types of Monitored Projects



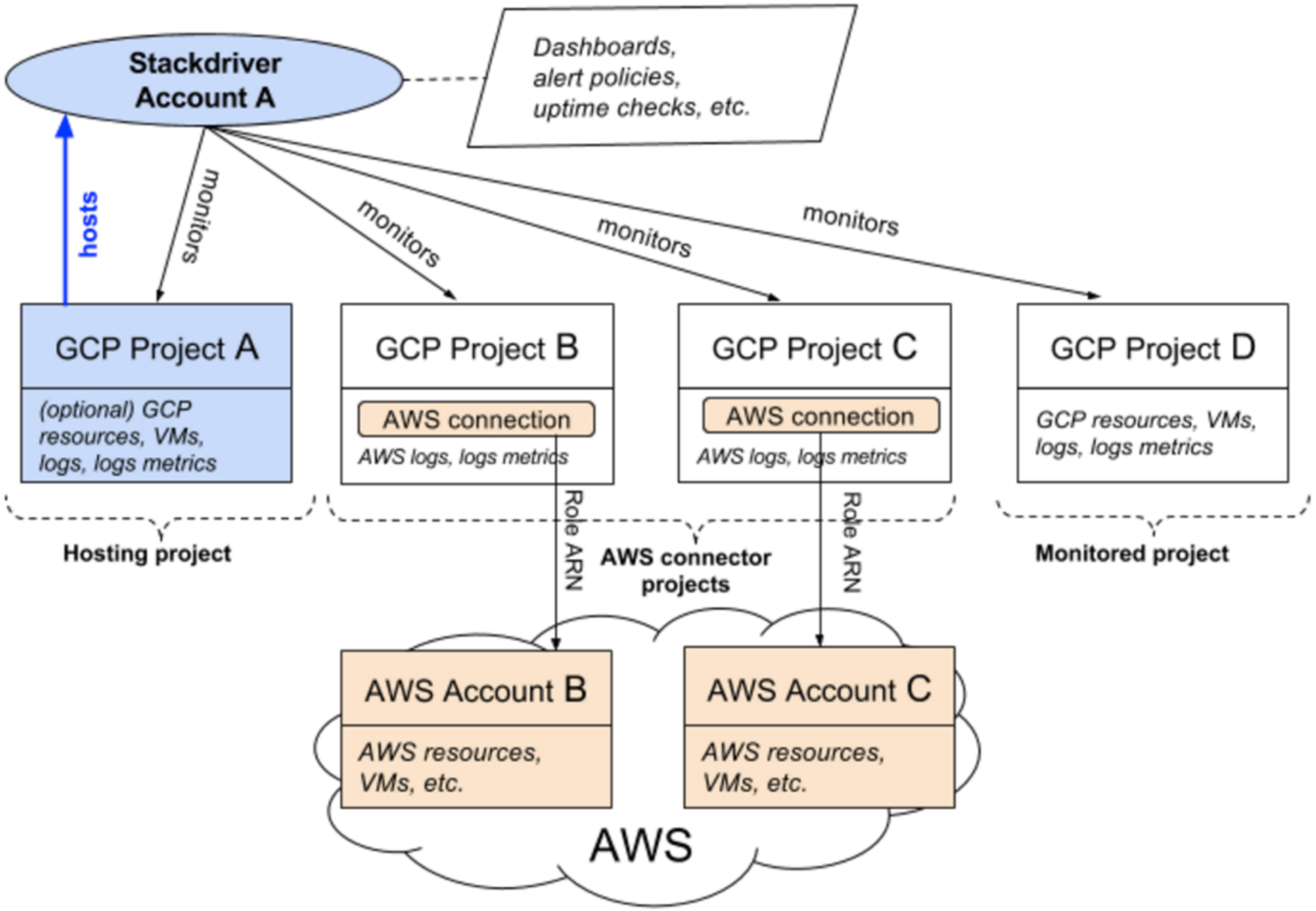
- If you use StackDriver logging from AWS, those logs will be in the AWS connector project (not in the host project of the Stackdriver account)

Types of Monitored Projects



- Don't put any GCP resources in an AWS connector project
  - Will not be monitored!

Types of Monitored Projects



- Monitored Projects: Regular (non-AWS) projects within GCP that are being monitored

Metrics

- Stackdriver Monitoring has metrics for
  - the CPU utilization of your VM instances
  - the number of tables in your SQL databases
  - hundreds more

- Can create custom metrics for StackDriver monitoring to track

- Three types:
  - gauge metrics
  - delta metrics
  - cumulative metrics

- Metric data will be available in StackDriver monitoring for 6 weeks

Metric Latency

- VM CPU utilisation - once a minute, available with 3-4 minutes lag

- If writing data programmatically to metric time series
    - first write takes a few minute to show up
    - subsequent writes visible within seconds

Monitored Resources

- VM instances
- Amazon EC2 instances, RDS databases
- Cloud storage buckets
- ...

Alerts and Notifications

- Lots of options
  - Email
  - Phone
  - SMS
  - Slack
  - WebHooks
  - ...
- (depends on service tier)

Error Reporting

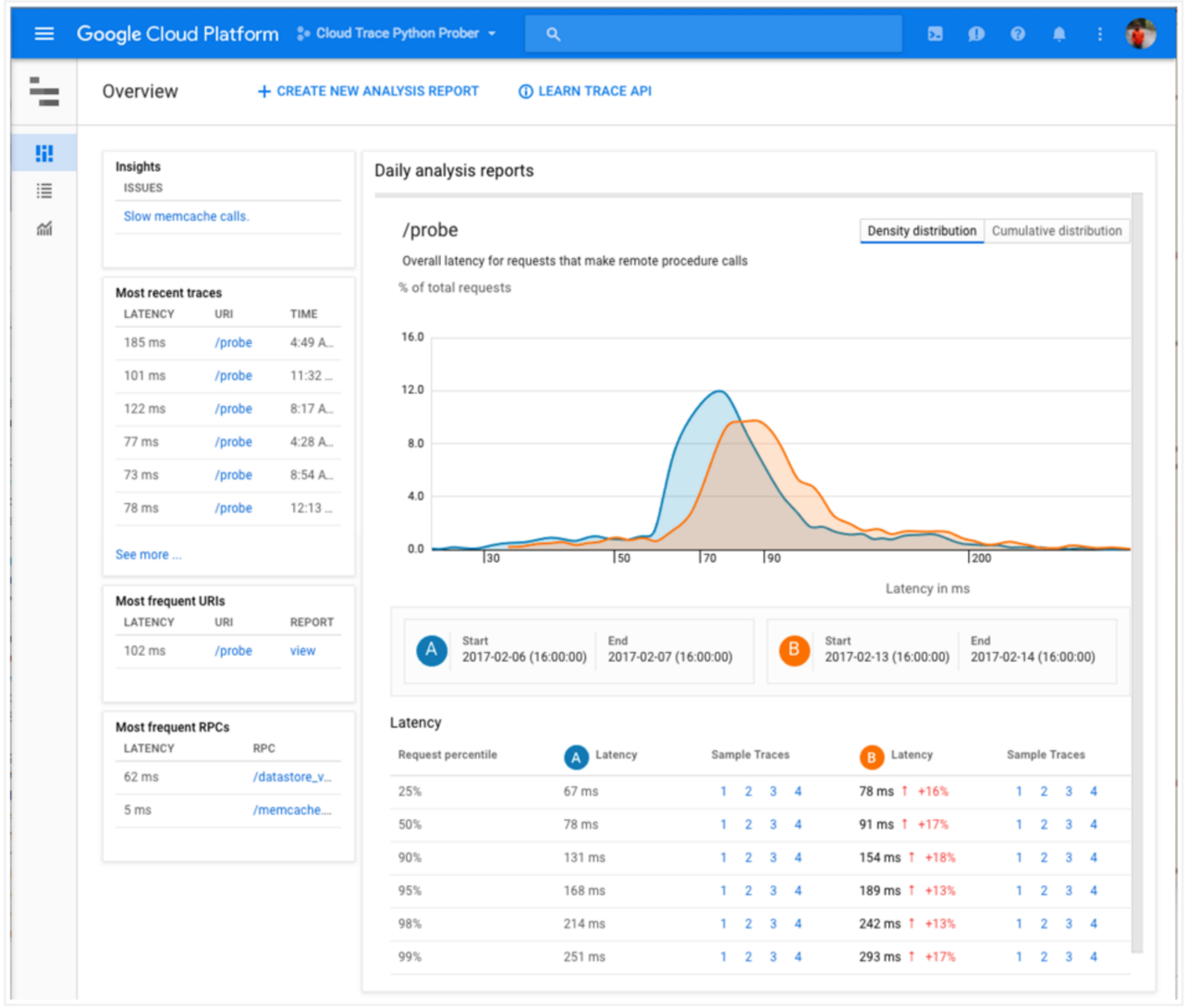
- StackDriver error reporting works on
  - AppEngine Standard Environment - log entries with a stack trace and severity of ERROR or higher automatically show up
  - AppEngine Flexible Environment - anything written to stderr automatically shows up
  - Compute Engine - instrument - throw error in exception catch block
  - Amazon EC2: Enable StackDriver logging

Trace

- Distributed tracing system that collects latency data from Google App Engine, Google HTTP(S) load balancers, and applications instrumented with the Stackdriver Trace SDKs

- Think TensorBoard for Google Cloud Apps

Trace


Google Cloud Platform
Cloud Trace Python Prober
Overview
+ CREATE NEW ANALYSIS REPORT
LEARN TRACE API
Insights
ISSUES
Slow memcache calls.
Most recent traces
LATENCY URI TIME
185 ms /probe 4:49 A...
101 ms /probe 11:32 ...
122 ms /probe 8:17 A...
77 ms /probe 4:28 A...
73 ms /probe 8:54 A...
78 ms /probe 12:13 ...
See more ...
Most frequent URIs
LATENCY URI REPORT
102 ms /probe view
Most frequent RPCs
LATENCY RPC
62 ms /datastore_v...
5 ms /memcache...
Daily analysis reports
/probe
Density distribution
Cumulative distribution
Overall latency for requests that make remote procedure calls
% of total requests
16.0
12.0
8.0
4.0
0.0
30
50
70
90
200
Latency in ms
A Start 2017-02-06 (16:00:00) End 2017-02-07 (16:00:00)
B Start 2017-02-13 (16:00:00) End 2017-02-14 (16:00:00)
Latency
Request percentile A Latency Sample Traces B Latency Sample Traces
25% 67 ms 1 2 3 4 78 ms ↑ +16% 1 2 3 4
50% 78 ms 1 2 3 4 91 ms ↑ +17% 1 2 3 4
90% 131 ms 1 2 3 4 154 ms ↑ +18% 1 2 3 4
95% 168 ms 1 2 3 4 189 ms ↑ +13% 1 2 3 4
98% 214 ms 1 2 3 4 242 ms ↑ +13% 1 2 3 4
99% 251 ms 1 2 3 4 293 ms ↑ +17% 1 2 3 4

Trace

- how long it takes your application to handle incoming requests from users or other applications
- how long it takes to complete operations like RPC calls performed when handling the requests
- round-trip RPC calls to App Engine services like Datastore, URL Fetch, and Memcache.

Stackdriver Logging

Logging with StackDriver

- Stackdriver Logging includes storage for logs, a user interface (the Logs Viewer), and an API to manage logs programmatically

- Stackdriver Logging lets you
  - read and write log entries
  - search and filter your logs
  - export your logs
  - create logs-based metrics.

Types of Logs

- Audit logs: permanent GCP logs (no retention period)
- Admin activity logs: for actions that modify config or metadata
- Data access logs: API calls that create modify or read user-provided data
- Admin activity logs are always on; data access logs need to be enabled (can be big)
- BigQuery data access logs are always on by default

Service Tiers and Retention

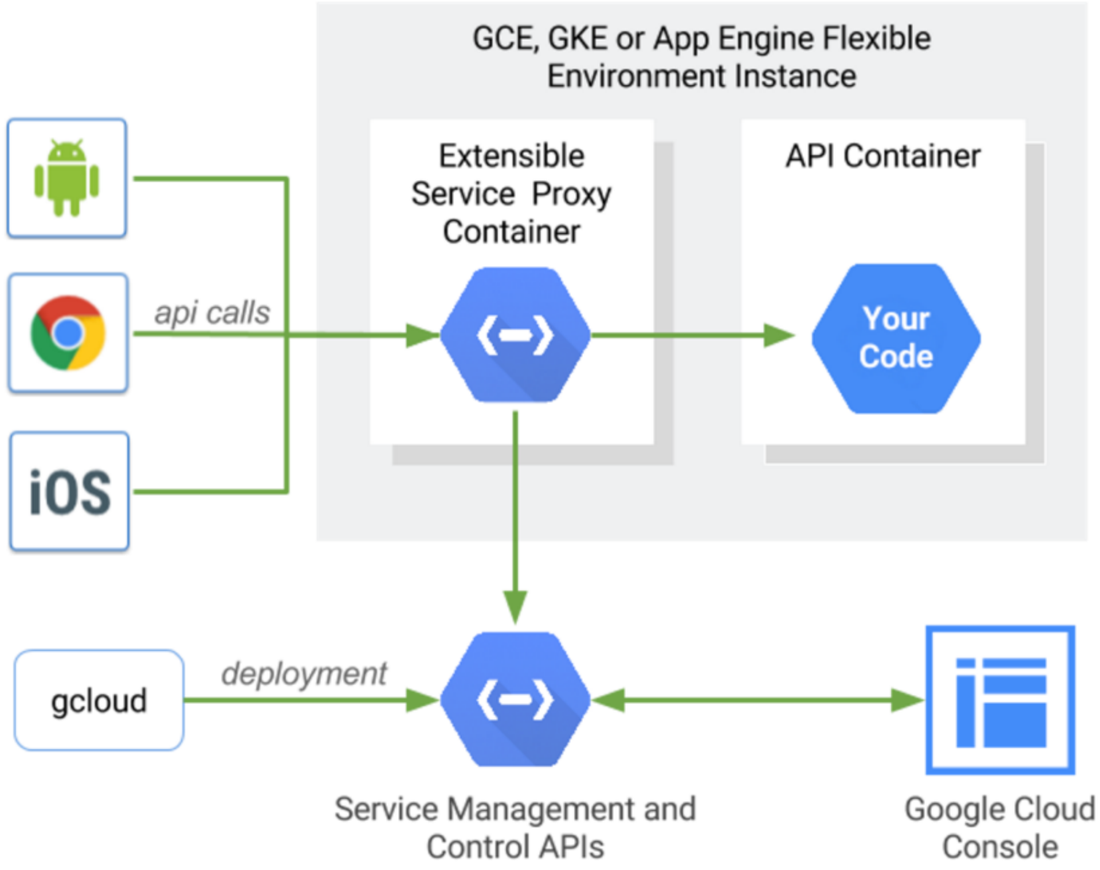
- Basic - no StackDriver account - free and 5 GB cap
- Retention period of log data depends on service tier

Using Logs

- Monitor virtually anything - VM instances, AWS EC2 instances, database instances ...
- Exporting to sinks: Cloud Storage, BigQuery datasets, Pub/Sub topics
- Create metrics to view in StackDriver monitoring

Cloud Endpoints

Cloud Endpoints



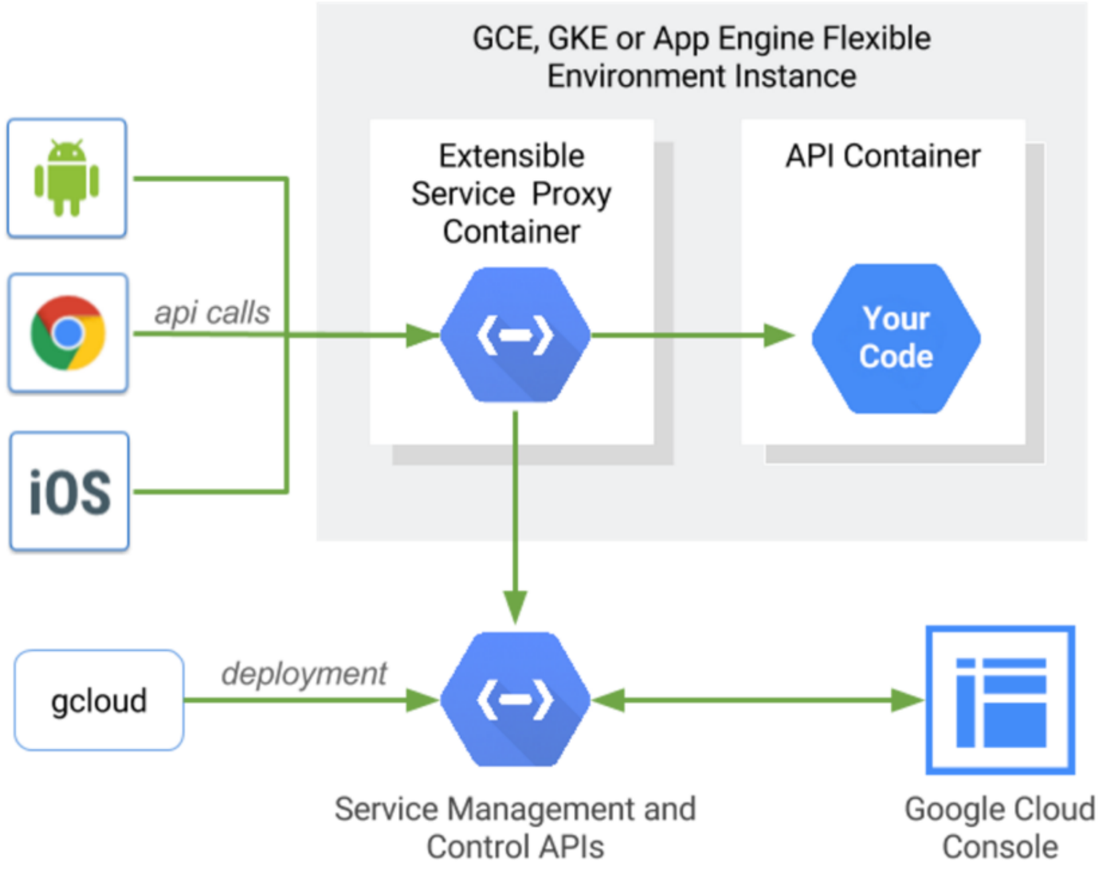
Helps create, share, maintain, and secure your APIs

Uses the distributed Extensible Service Proxy to provide low latency and high performance

Provides authentication, logging, monitoring

Host your API anywhere Docker is supported so long as it has Internet access to GCP

Cloud Endpoints



Host your API anywhere Docker is supported so long as it has Internet access to GCP

But, ideally, use with

- App Engine (flexible or some types of standard)
- Container Engine instance
- Compute Engine instance

Note - proxy and API containers must be on same instance to avoid network access

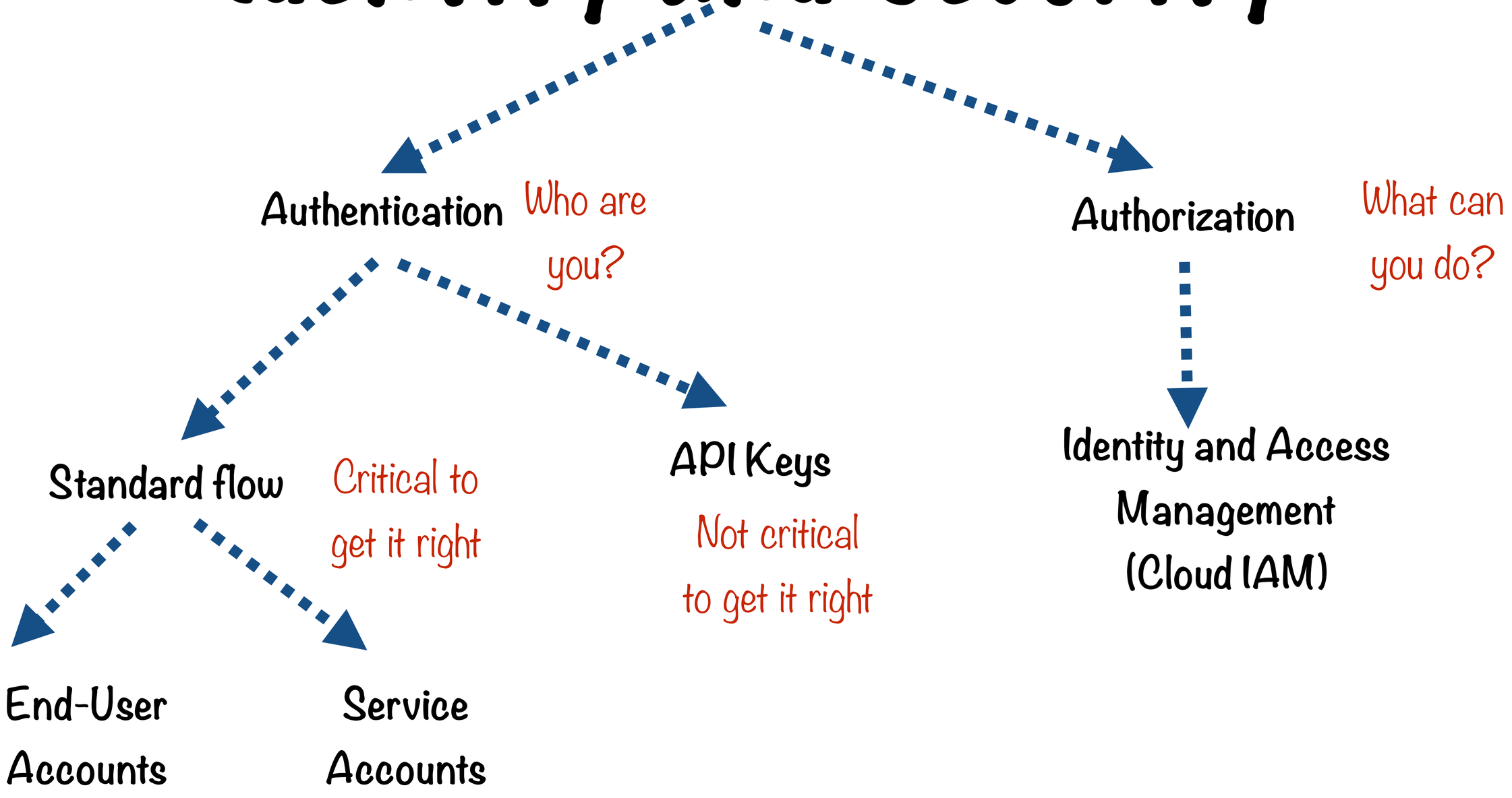
Summary

Module - Identity and Security

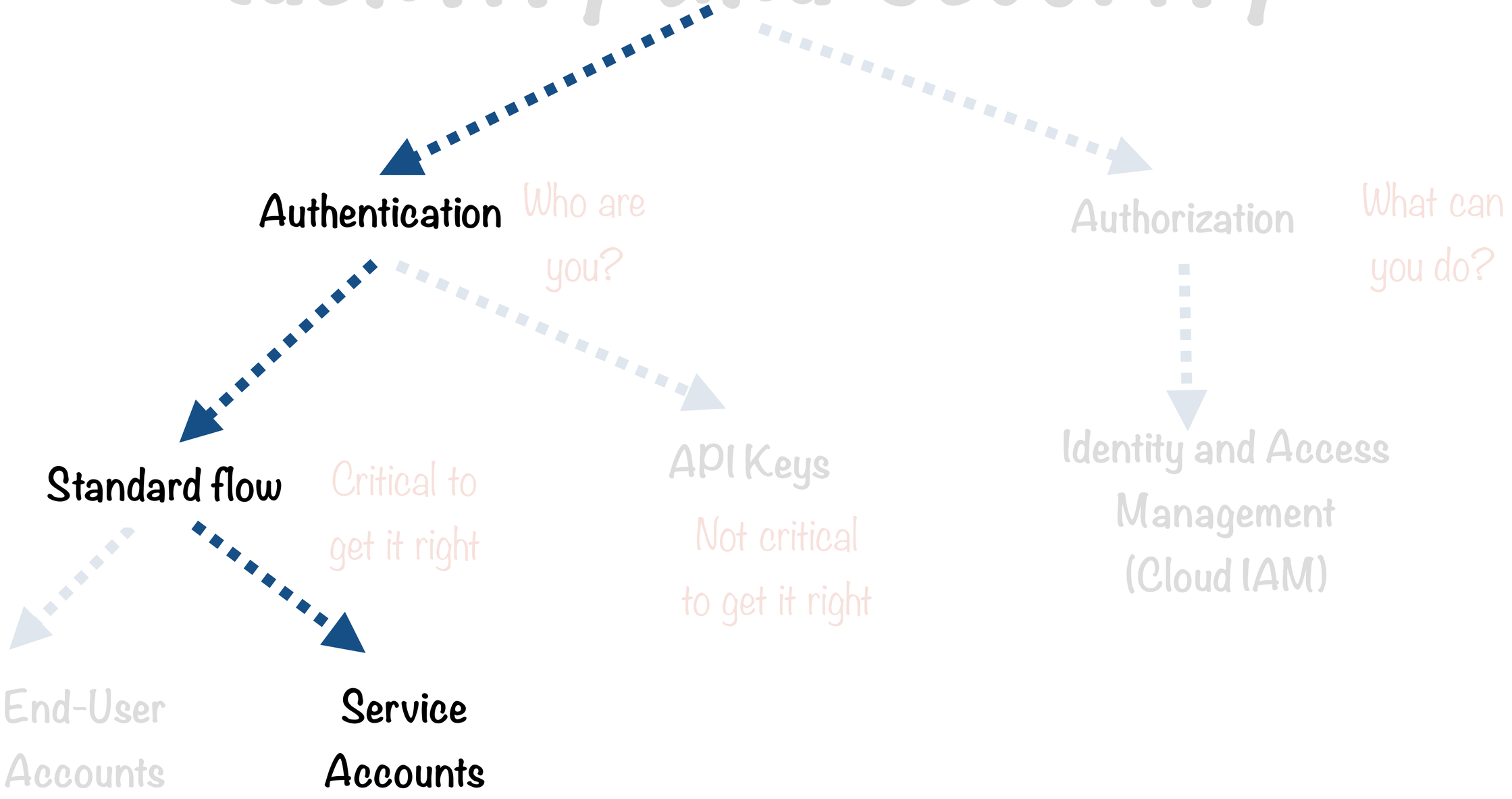
Overview

Authentication

Identity and Security



Identity and Security



Service Accounts

- Most flexible and widely supported authentication method
- Different GCP APIs support different credential types, but all GCP APIs support service accounts
- For most applications that run on a server and need to communicate with GCP APIs, use service accounts

Why use them?

- Service accounts are associated with a project, not a user
- So, any project user gets access to all required resources at one go
- Btw, can also assign roles to service accounts
- Only use end-user accounts if you'd like to differentiate even between different end-users on the same project

Application Credentials

- A service account is a Google account that is associated with your GCP project, as opposed to a specific user.

- Create from
  - GCP Console
  - Programmatically

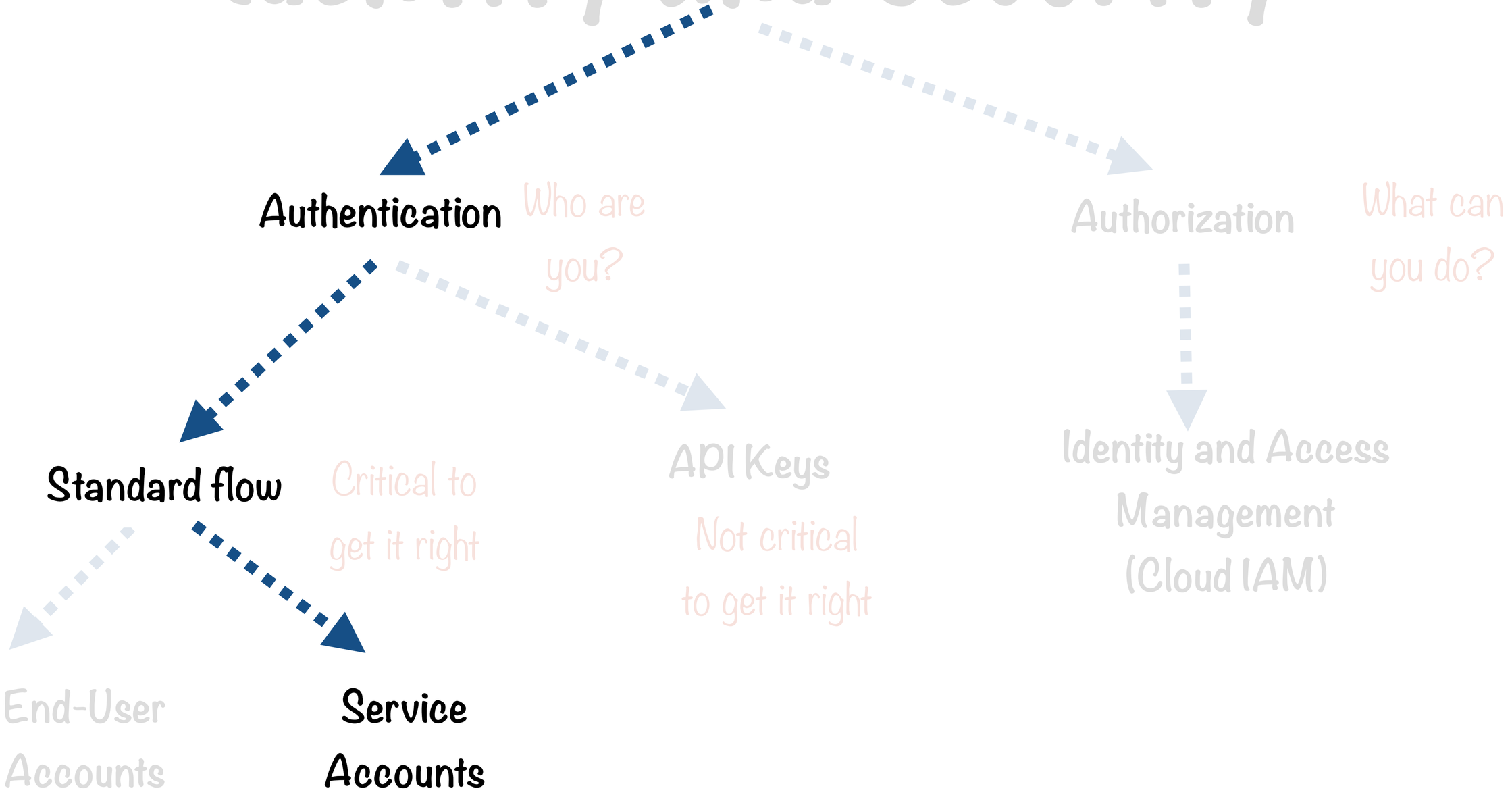
- Service account is associated with credentials via environment variable GOOGLE_APPLICATION_CREDENTIALS

- At any point, one set of credentials is 'active', called **Application Default Credentials**

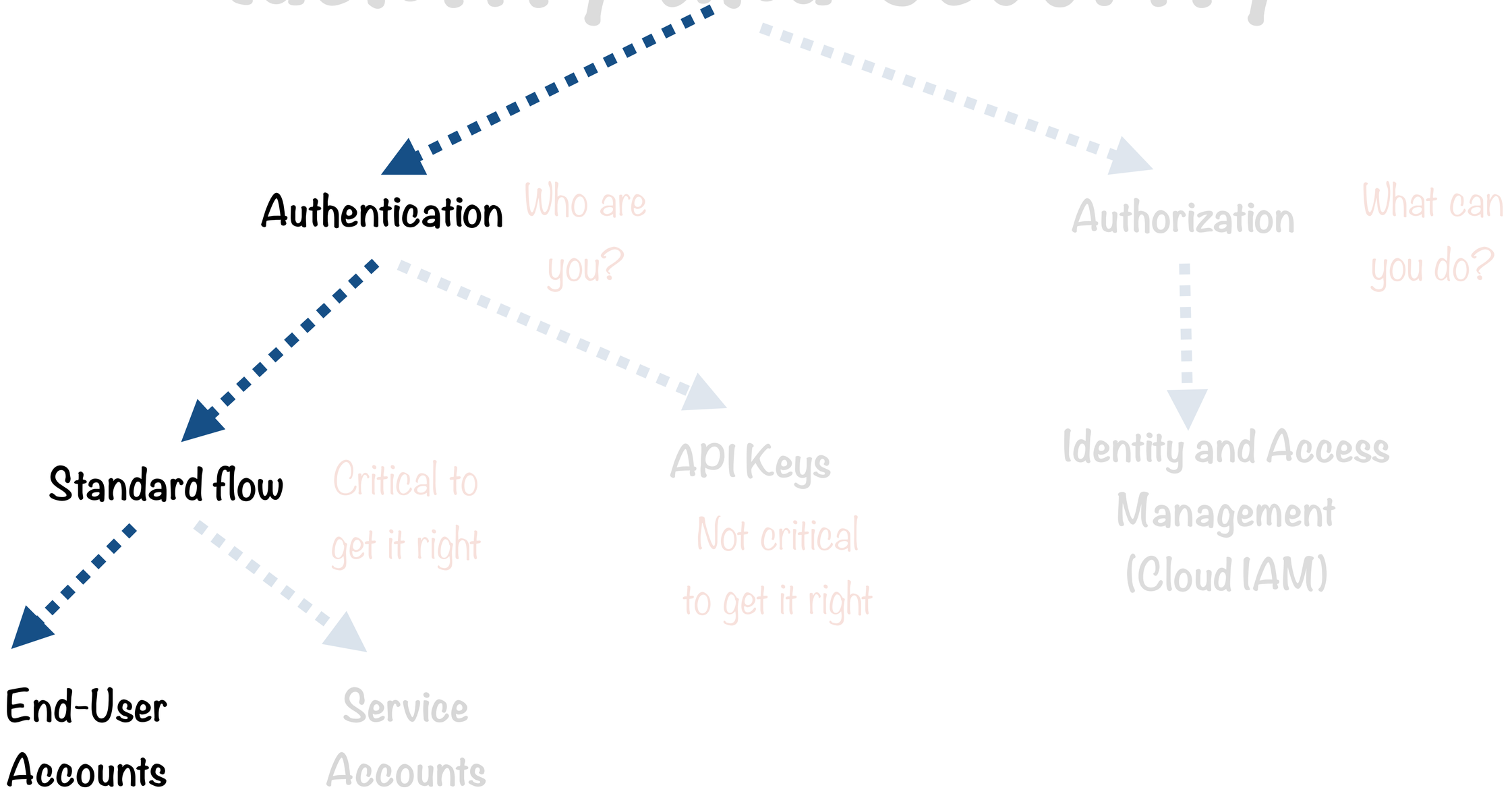
Application Default Credentials

- When your code uses a client library, the strategy checks for your credentials in the following order:
  - First, ADC checks to see if the environment variable GOOGLE_APPLICATION_CREDENTIALS is set. If the variable is set, ADC uses the service account file that the variable points to.
  - If the environment variable isn't set, ADC uses the default service account that Compute Engine, Container Engine, App Engine, and Cloud Functions provide, for applications that run on those services.
  - If ADC can't use either of the above credentials, an error occurs.

Identity and Security



Identity and Security



End-user Authentication

- Use service accounts wherever possible
- In certain specific cases however, end-user authentication its unavoidable

End-user Authentication

- You need to access resources on behalf of an end user of your application
    - For example, your application needs to access Google BigQuery datasets that belong to users of your application.
- You need to authenticate as yourself (not as your application)
    - For example, because the Cloud Resource Manager API can create and manage projects owned by a specific user, you would need to authenticate as a user to create projects on their behalf.

"Sign in to Quora using Google"

- User navigates to quora.com
- Quora needs to access resources on behalf of user
- Quora presents Google sign-in screen to user; user signs in
- Quora requests Google to authenticate user
- Quora has authenticated user, now releases resource

“Sign in to Quora using Google”

- Resource owner: Quora guarding access to your account
- Resource server: Quora granting access to your account
- Client: Quora talking to Google
- Authorisation server: Google

OAuth 2.0

- Application needs to access resources on behalf of a specific user
- Application presents consent screen to user; user consents
- Application requests credentials from some authorisation server
- Application then uses these credentials to access resources

Creation

- GCP Console => API Manager => Credentials => Create
- Select “OAuth client ID”
- Will create OAuth client secret

"Access API via GCP Project"

- User wants to access some API
- Project needs to access that API on behalf of user
- Project requests GCP API Manager to authenticate user by passing client secret; API manager responds
- Project has authenticated user, now gives API access

"Access API via GCP Project"

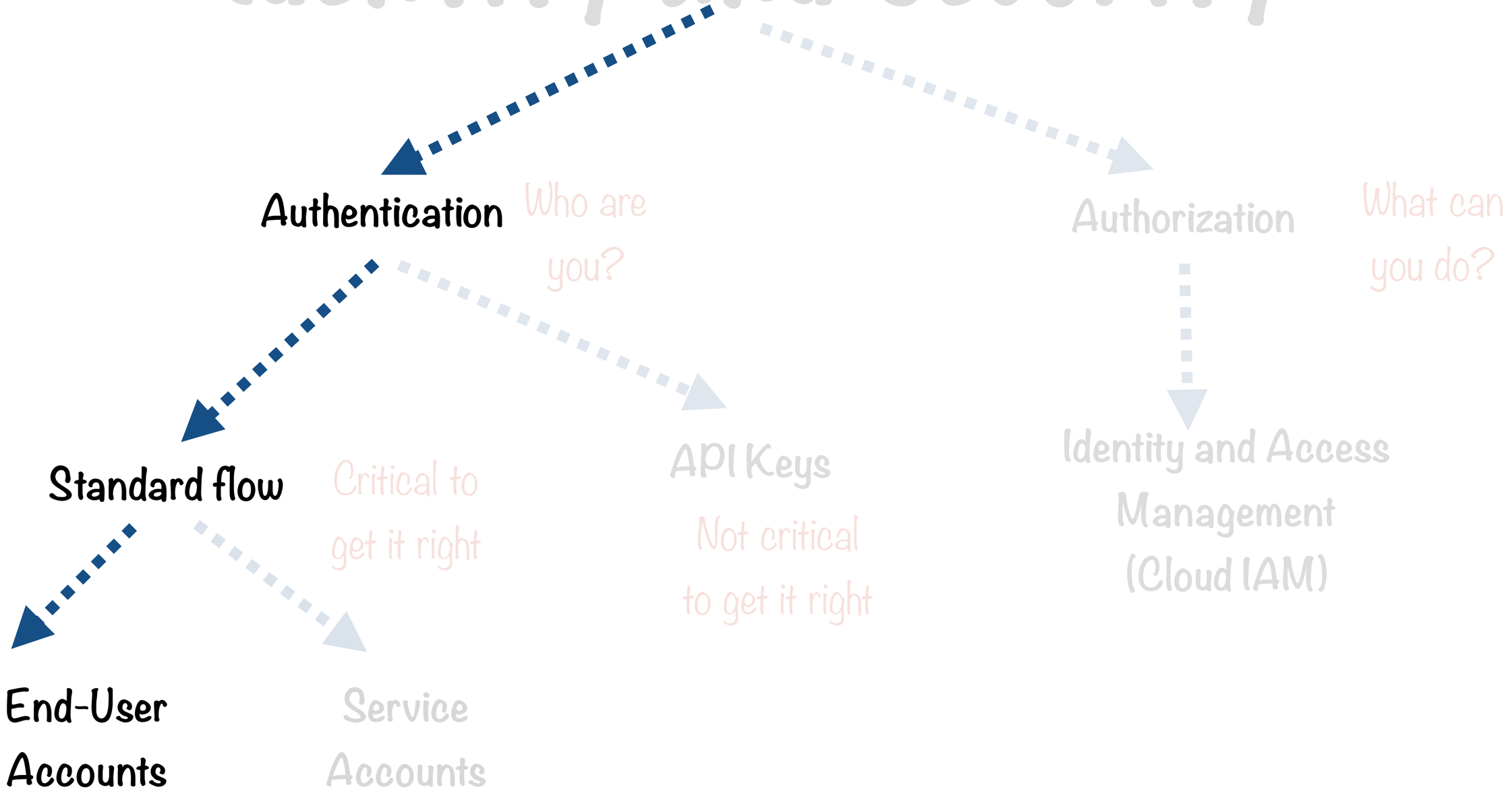
- Resource owner: Project guarding access to your account
- Resource server: Project granting access to your account
- Client: Project talking to API manager
- Authorisation server: API manager

Caution

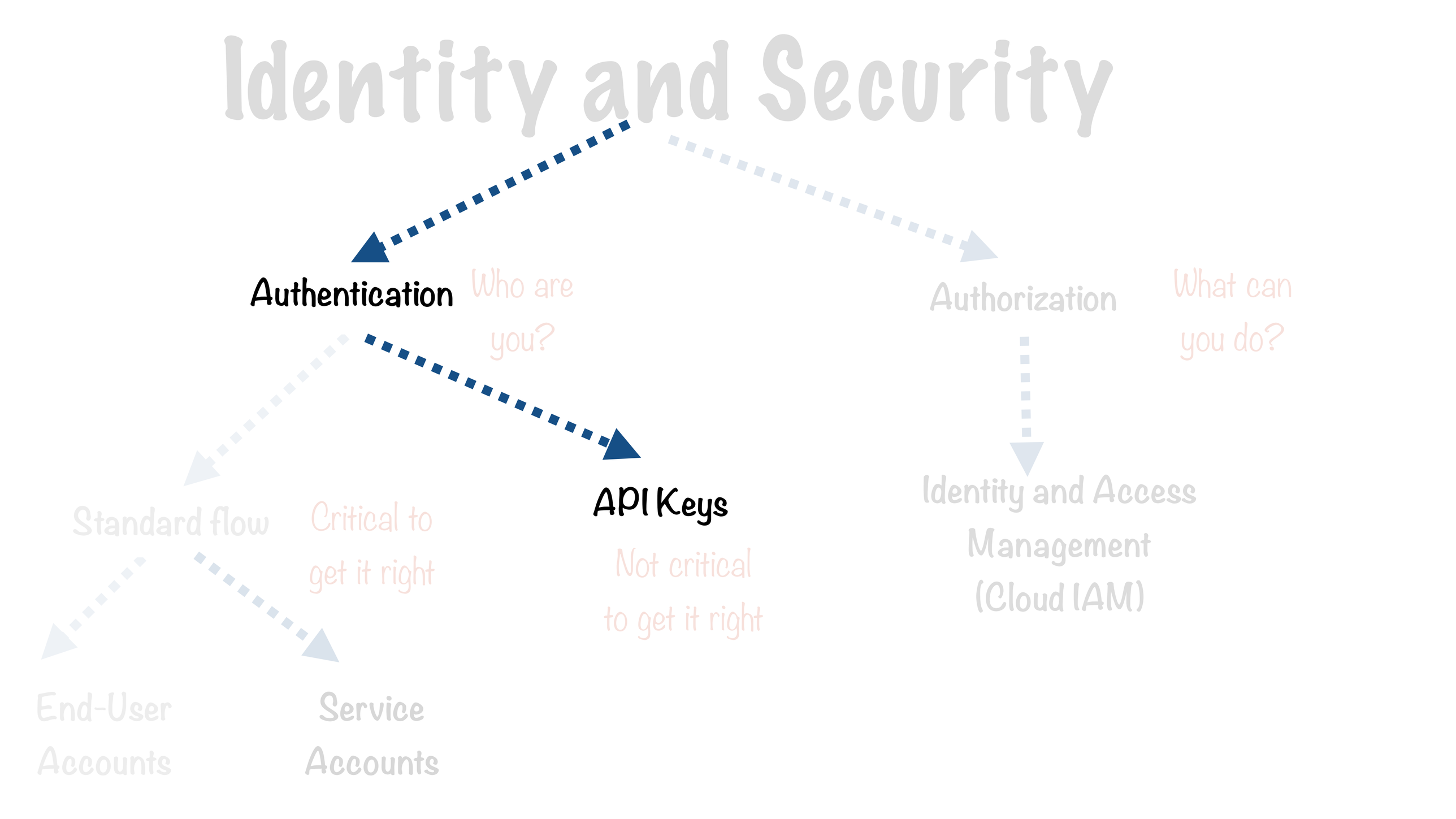
- OAuth client ID secrets are viewable by all project owners and editors, but not readers

- If you revoke access to some user, remember to reset these secrets to prevent data exfiltration

Identity and Security



Identity and Security



API Keys

- Simple encrypted string

- Can be used when calling certain APIs that **don't need to access private user data**

API Keys

- Useful in clients such as browser and mobile applications that don't have a backend server

- The API key is used to track API requests associated with your project for quota and billing.

Creation

- GCP Console => API Manager => Credentials => Create
- Select “API Key”

Beware

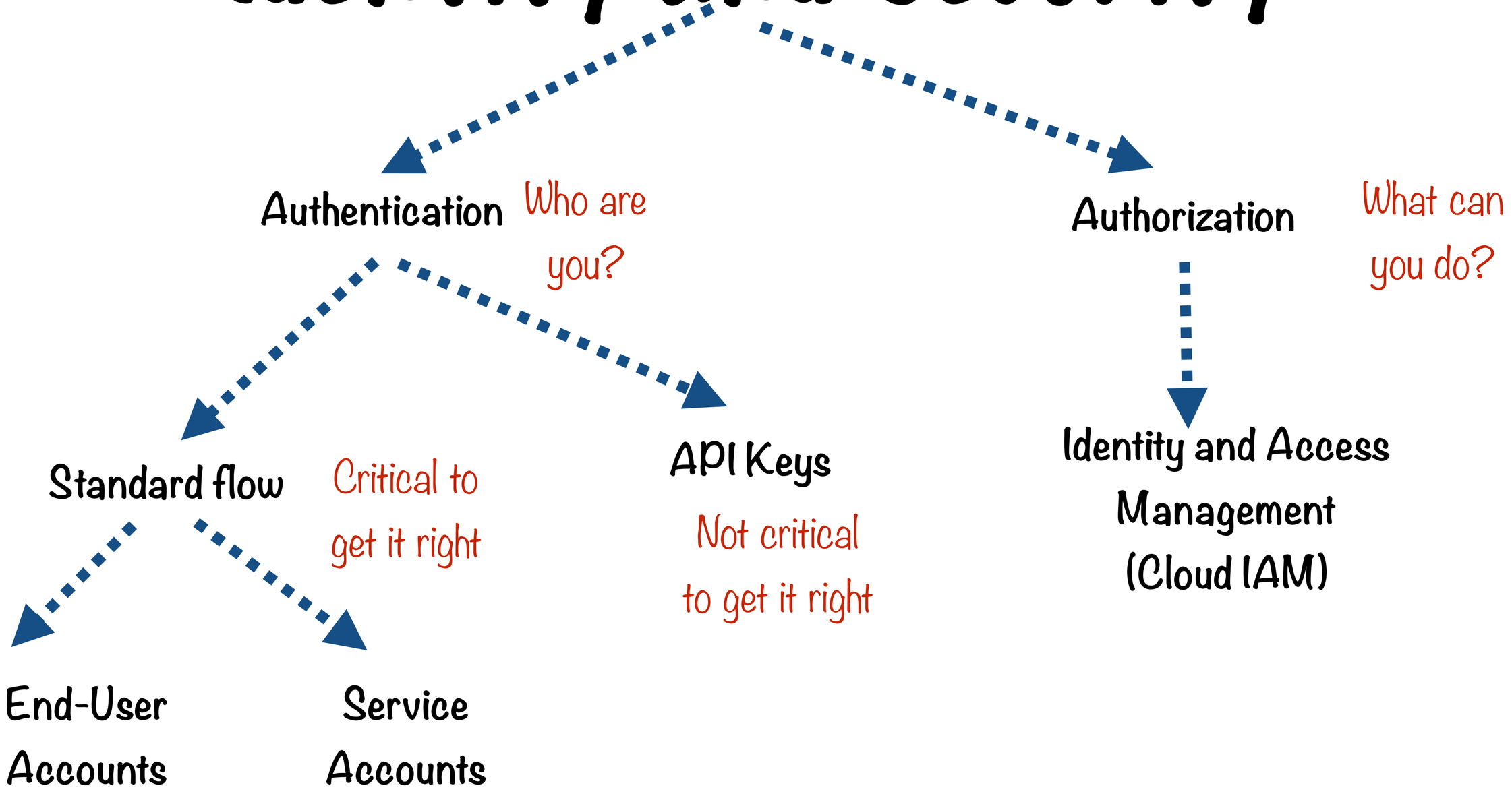
- Can be used by anyone - Man-in-the-Middle
- Do not identify user or application making request

ML APIs

- Used by a small number of GCP APIs
  - Natural language processing
  - Translation
  - Speech
  - Vision

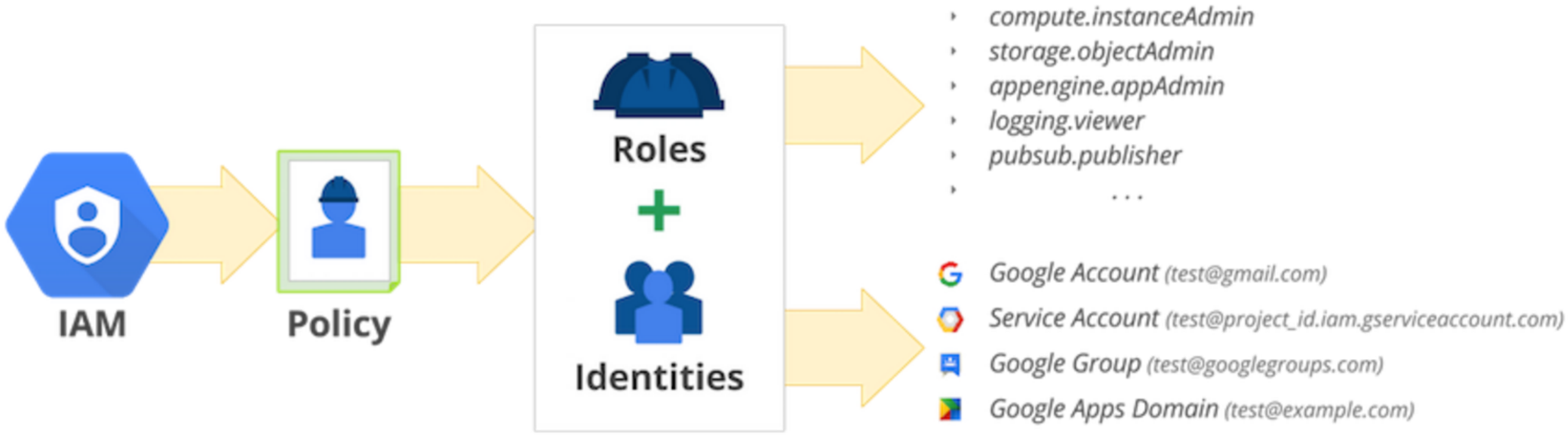
- API key can be used for any/all of these - can't restrict which is used

Identity and Security

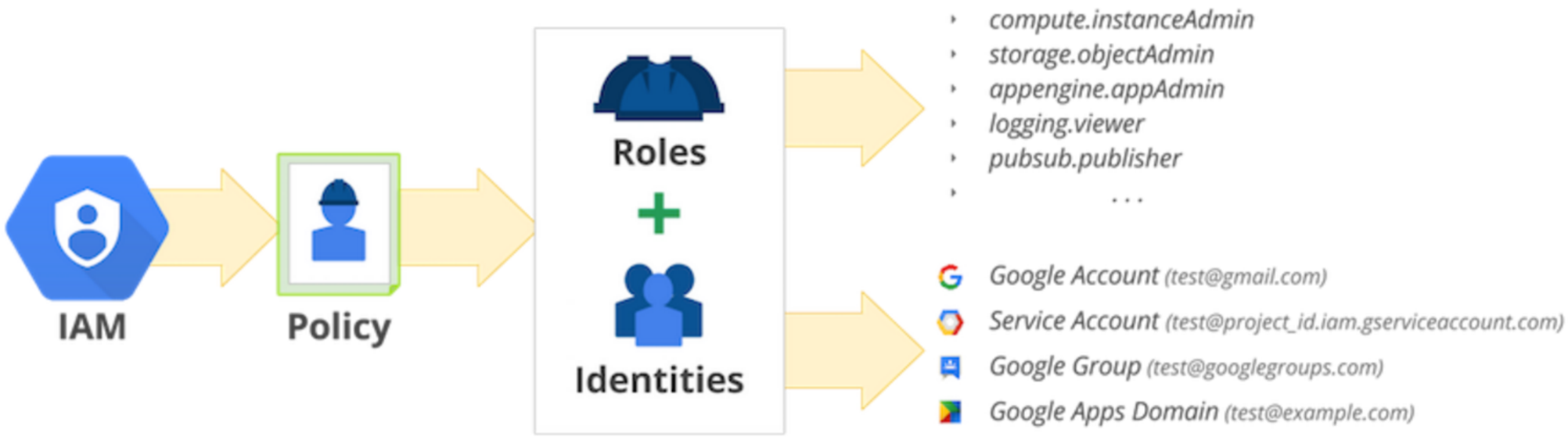


Identity and Security

Identity and Access Management (IAM)



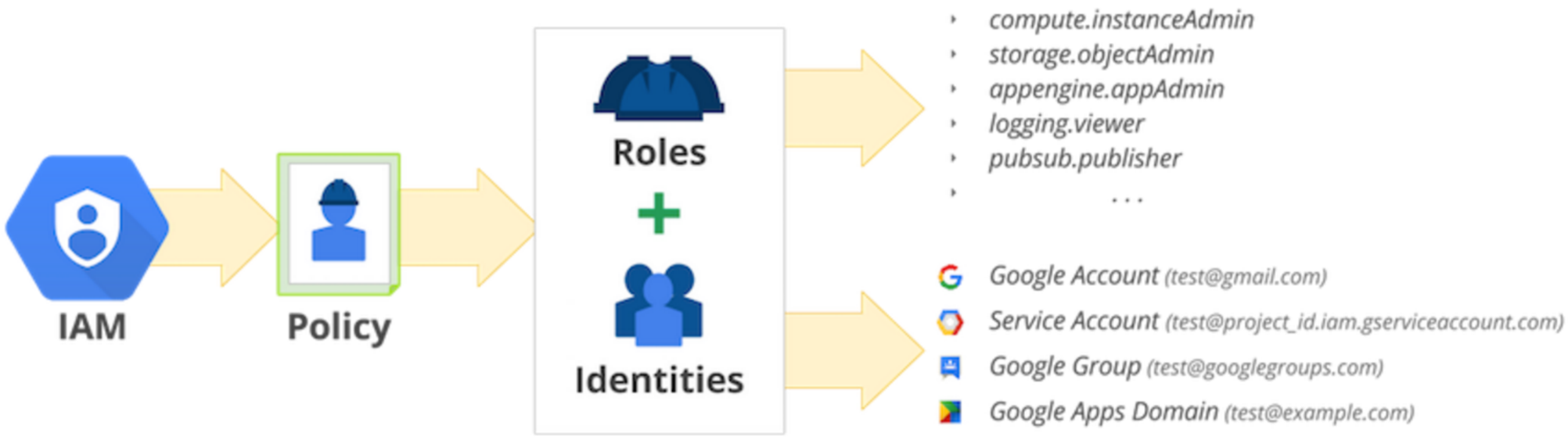
Identity and Access Management (IAM)



Identities

- End-user (Google) account
- Service account
- Google group
- G-Suite domain
- Cloud Identity domain
- allUsers, allAuthenticatedUsers

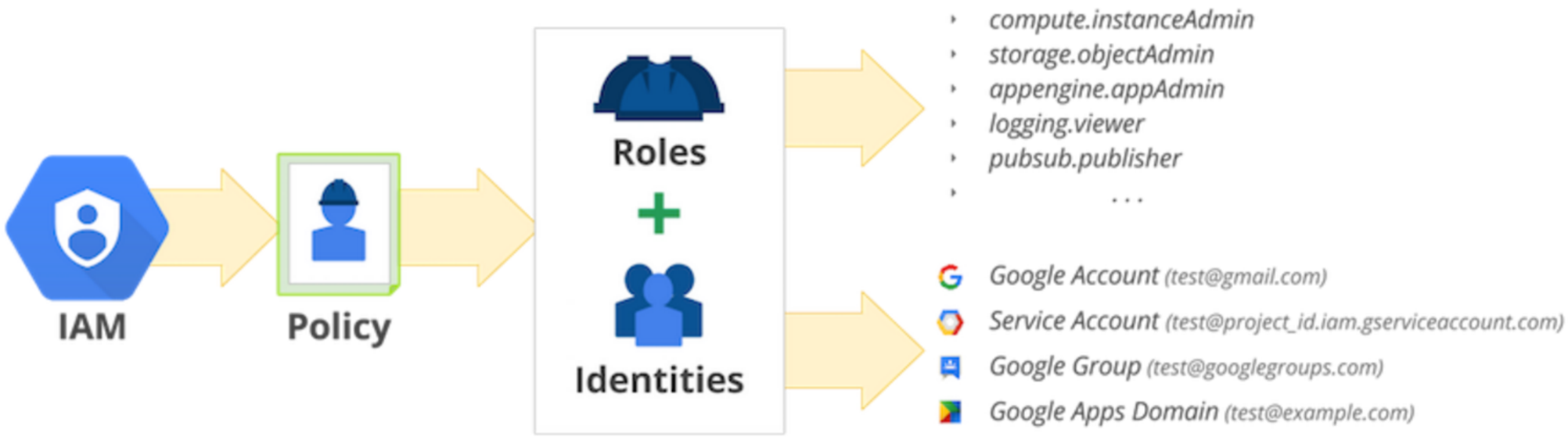
Identity and Access Management (IAM)



Roles

- lots of granular roles
- per resource

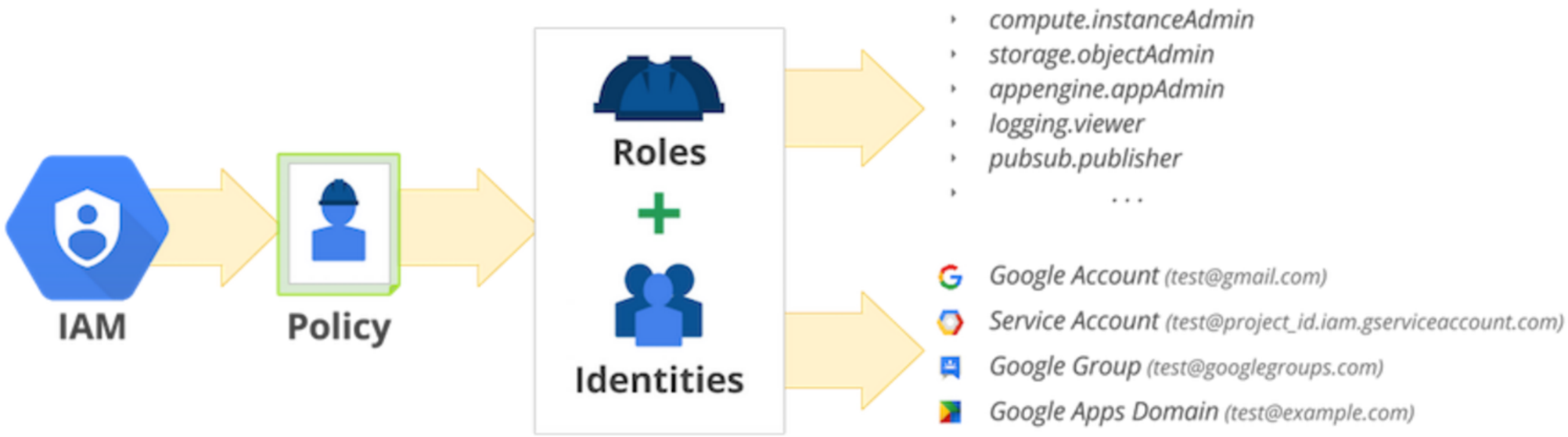
Identity and Access Management (IAM)



Resources

- Projects
- Compute Engine instances
- Cloud Storage buckets
- ...

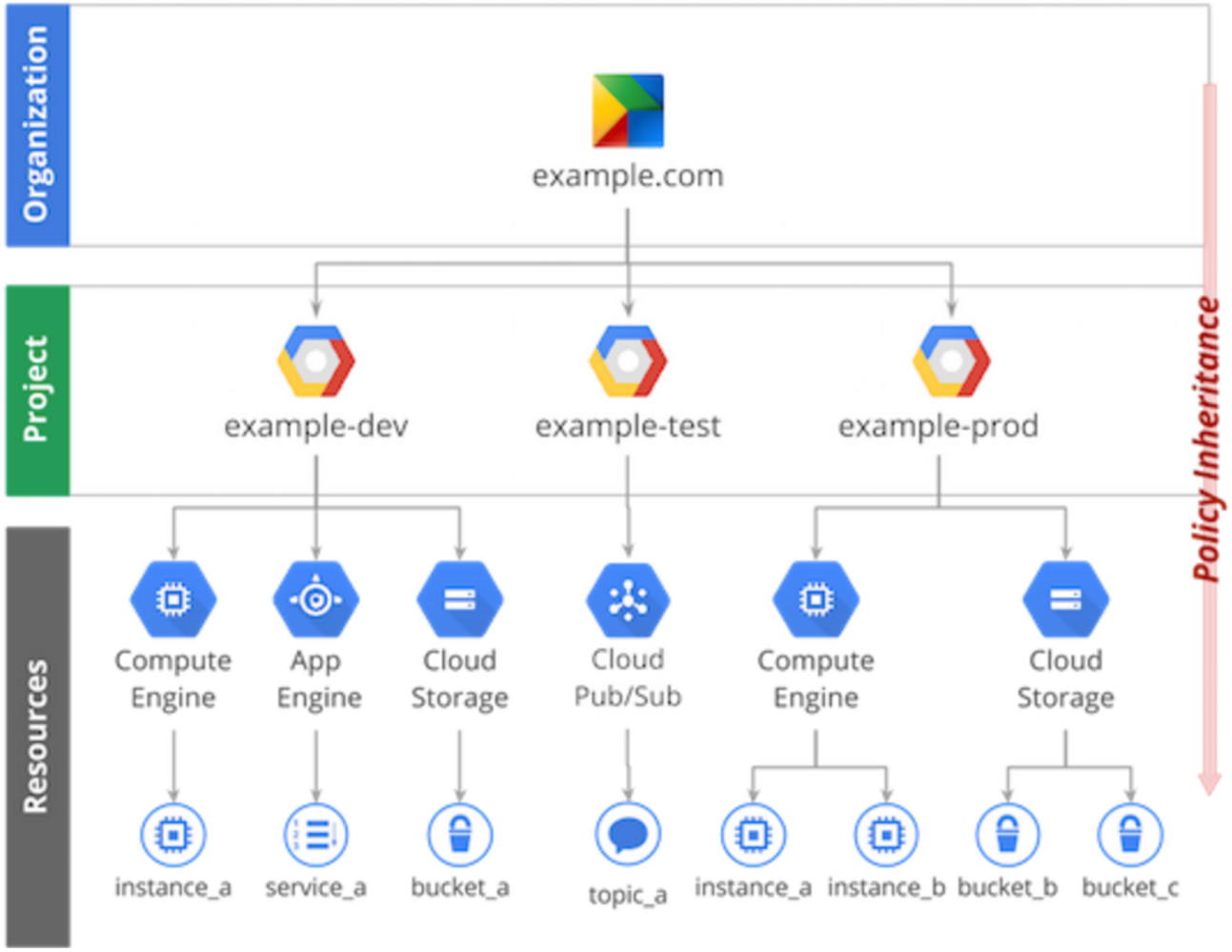
Identity and Access Management (IAM)



Policy

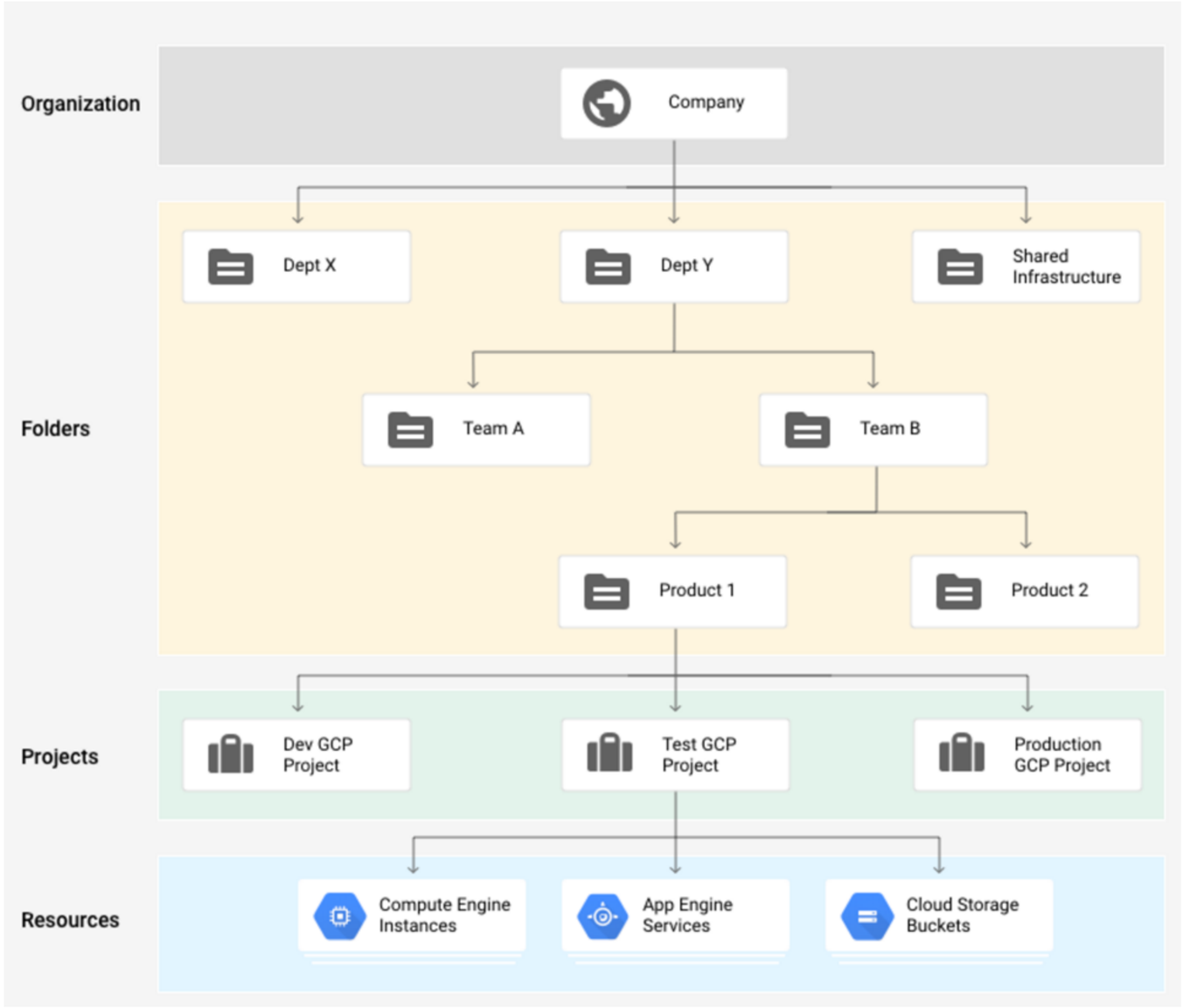
- Associate identities with roles

Resource Hierarchy



- **Organization >> project >> resource**
- Can set an IAM access control policy at any level in the resource hierarchy
- Resources inherit the policies of the parent resource

Resource Hierarchy



Organization

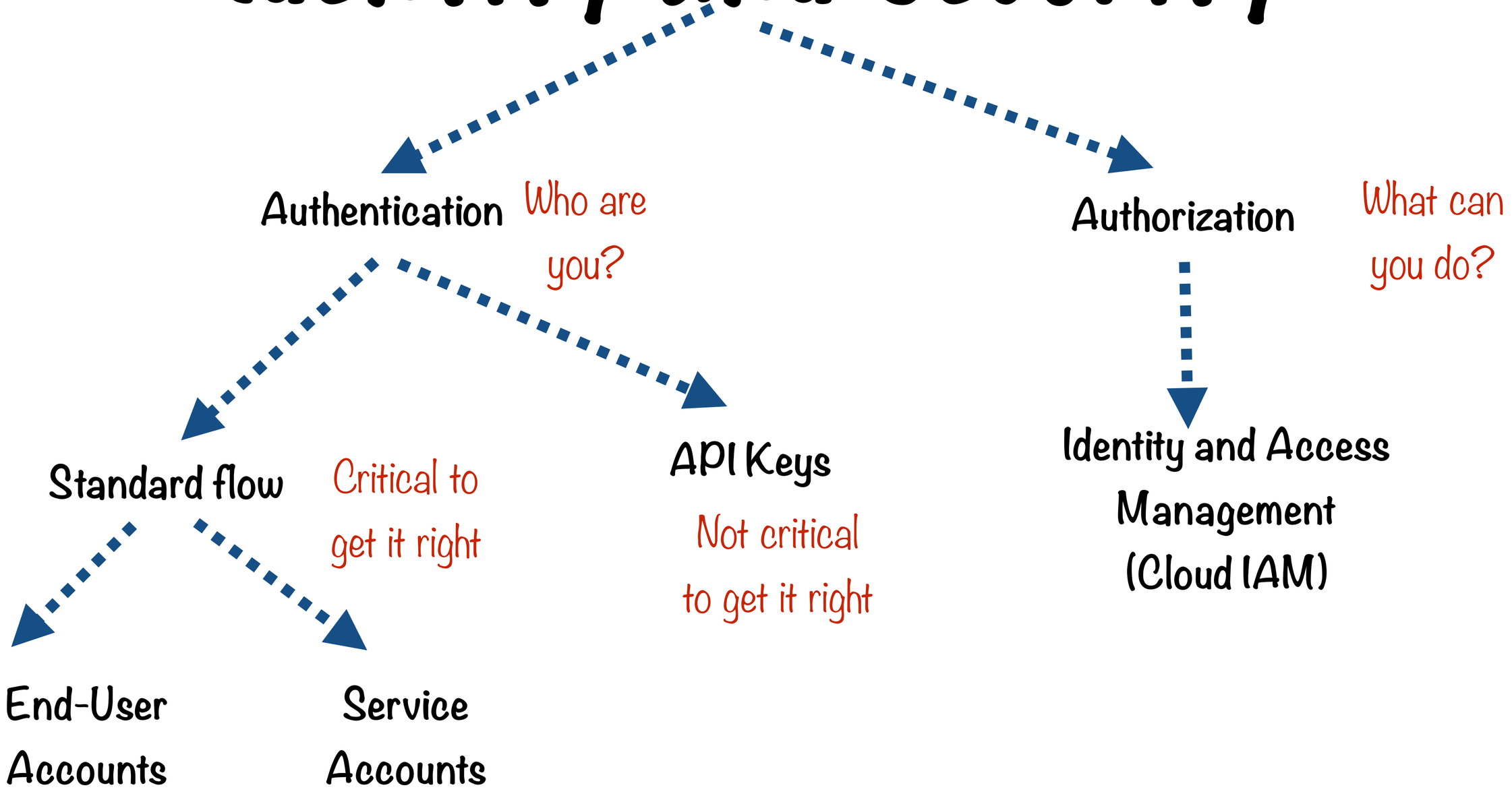
- Not required, but helps separate projects from individual users
- Link with G-suite super admin
- Root of hierarchy, rules cascade down

Folders

- Logical groupings of projects

Identity-Aware Proxy

Identity and Security



Identity and Security

Authentication

Who are you?

Authorization

What can you do?

- **Identity-Aware Proxy** (IAP) is an HTTPS-based, i.e. web-based, way to combine all of this

Standard flow

Critical to get it right

API Keys

Not critical to get it right

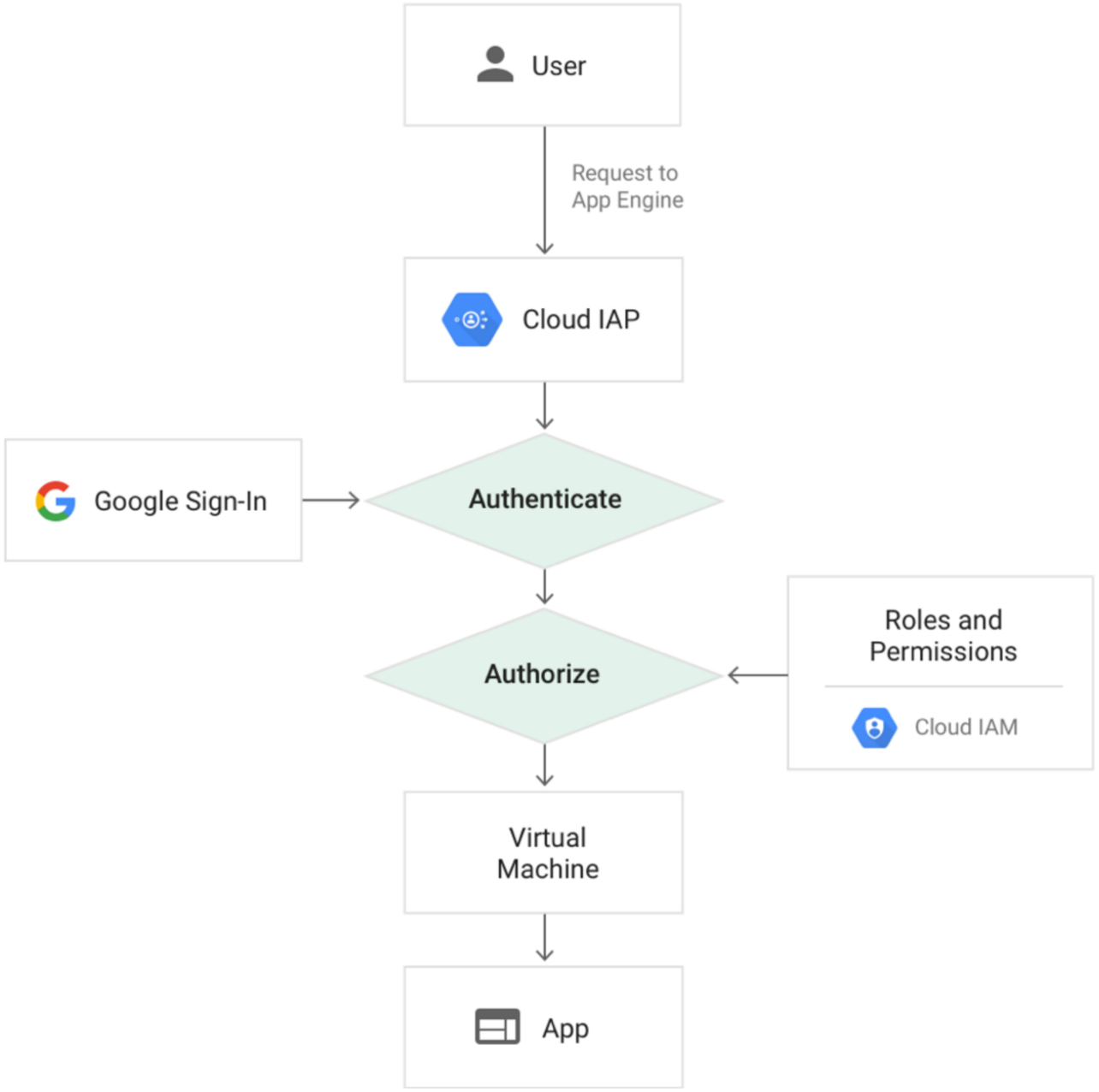
Identity and Access Management (Cloud IAM)

End-User Accounts

Service Accounts

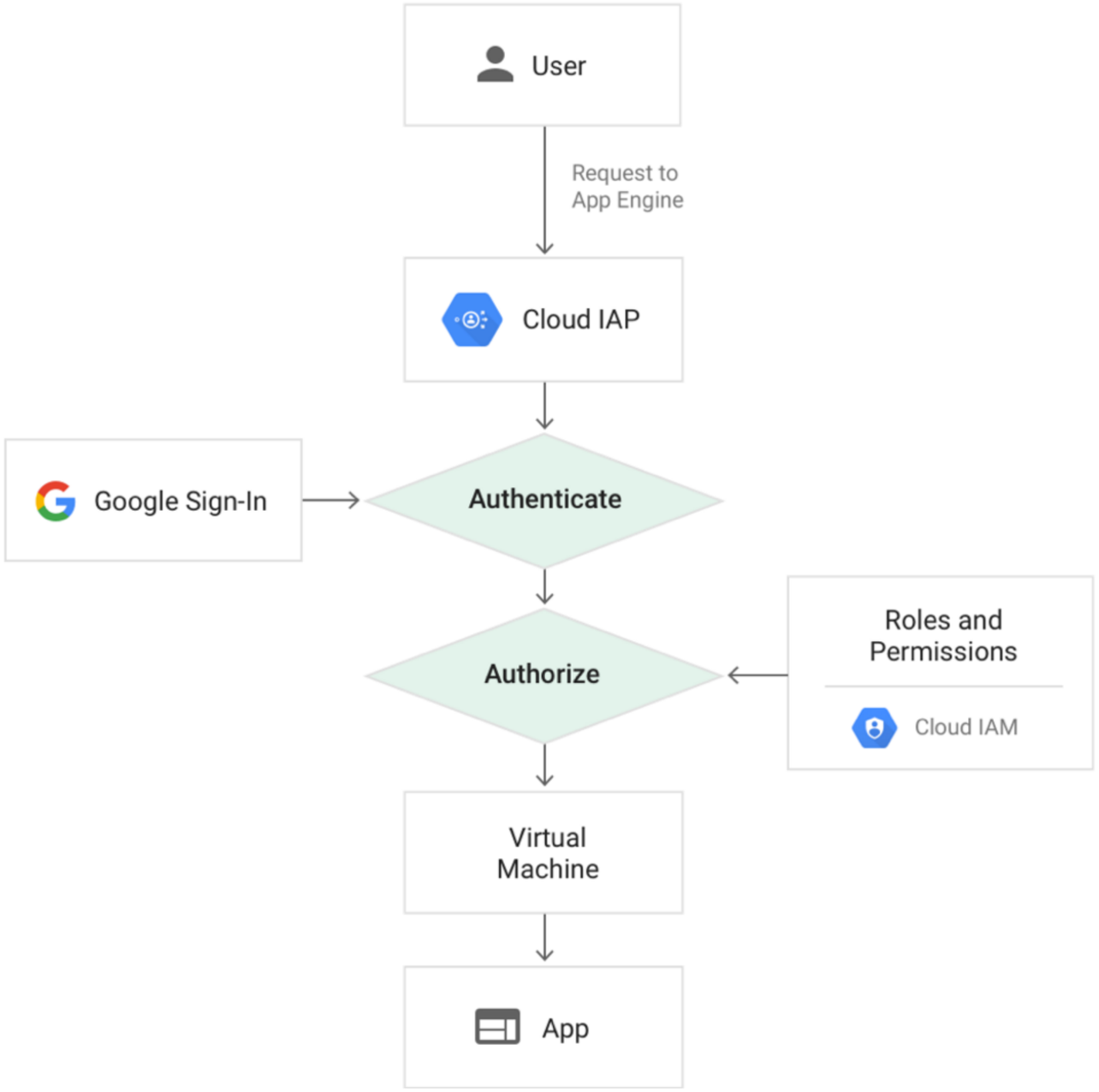
IAP

- IAP acts as an additional safeguard on a particular resource
- Turning on IAP for a resource causes creation of an OAuth 2.0 Client ID & secret (per resource)
    - Don't go in and delete any of these! IAP will stop working



Identity Aware Proxy

- central Authorization layer for applications accessed by HTTPS
- Application-level access control model instead of relying on network-level firewalls

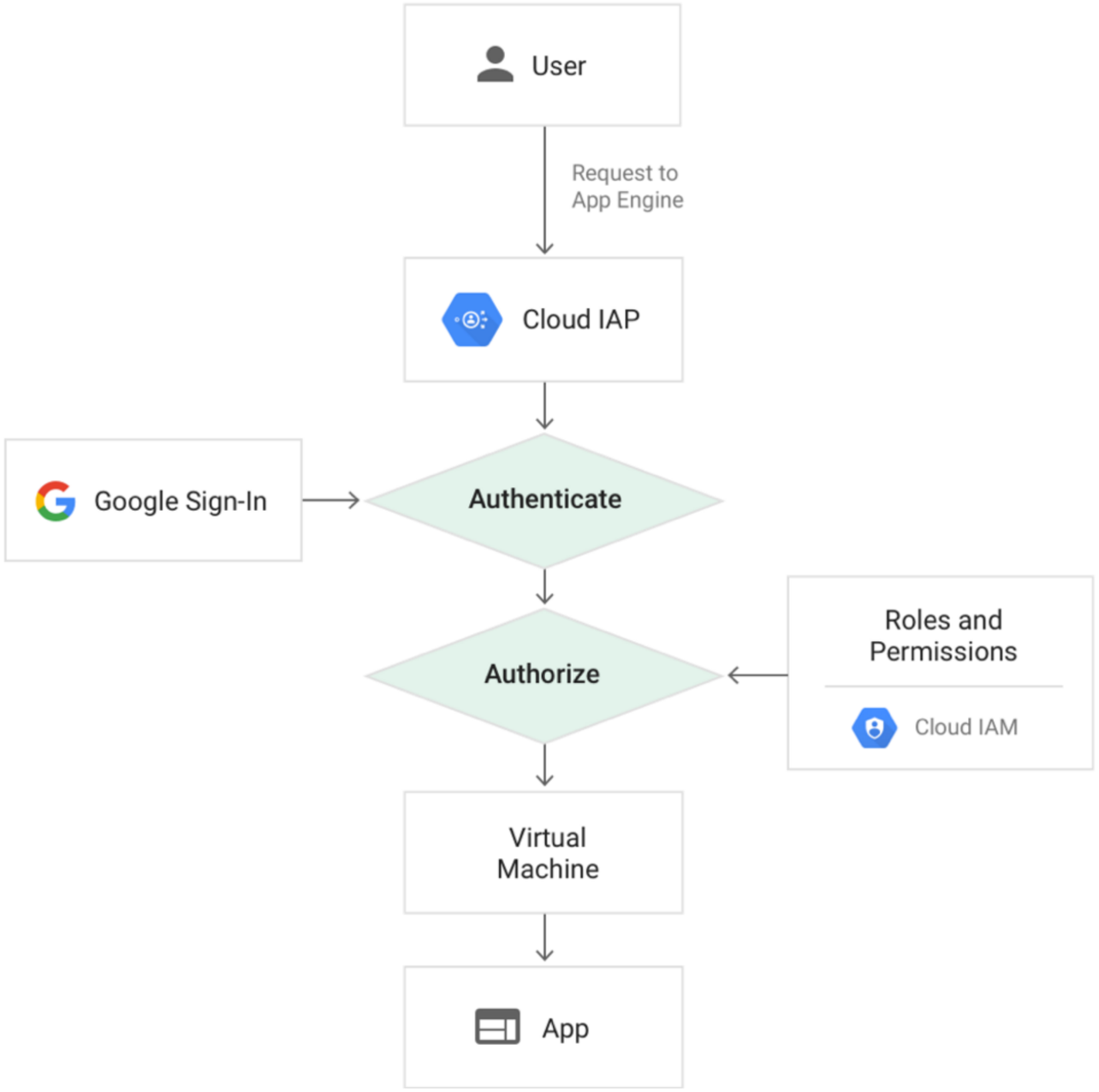


Identity Aware Proxy

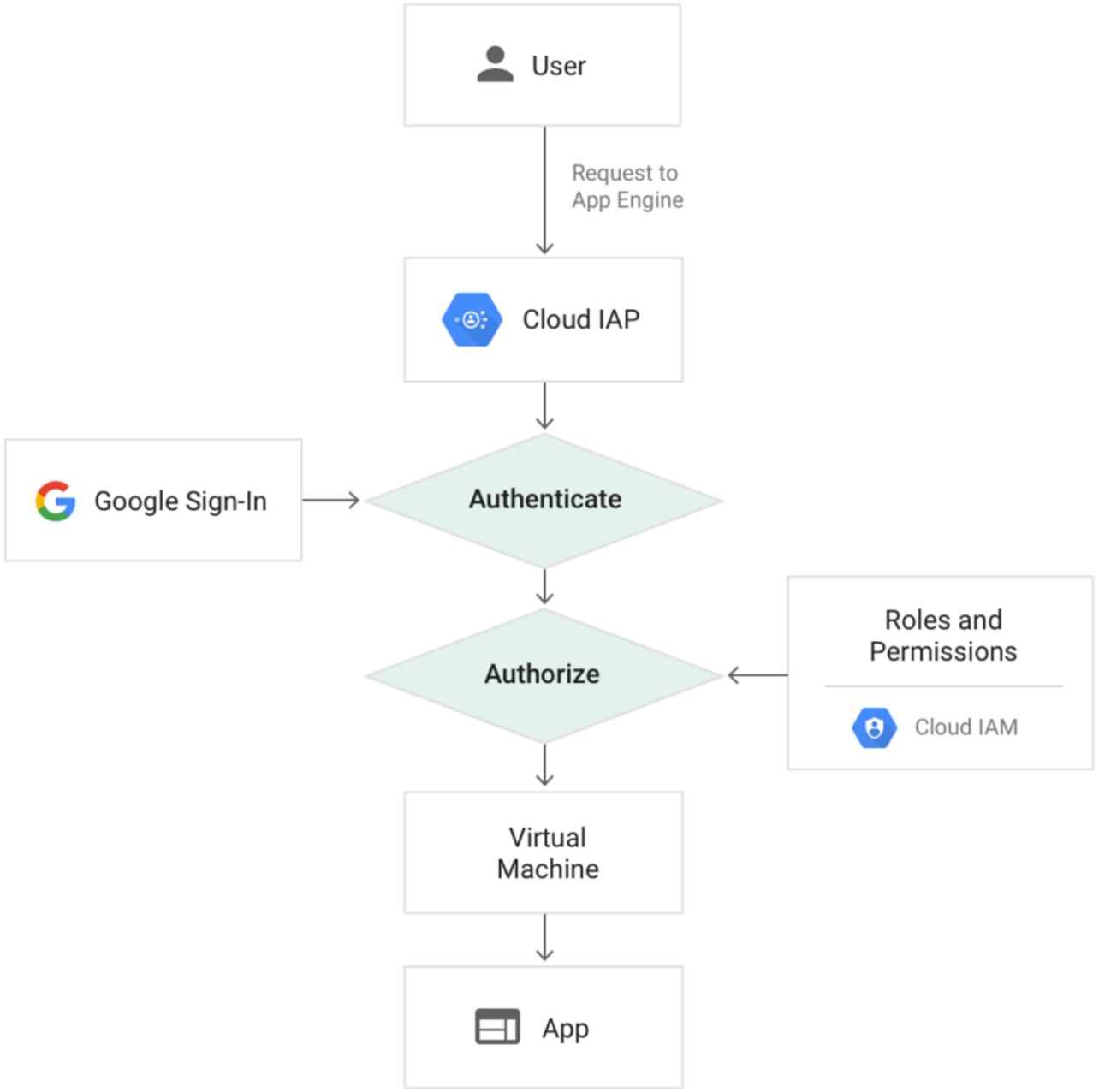
- With Cloud IAP, you can set up group-based application access:

- a resource could be accessible for employees and inacccessible for contractors, or only accessible to a specific department.

IAP and IAM



- IAP is an additional step, not a bypassing of IAM
- So, users and groups still need correct Cloud Identity Access Management (Cloud IAM) role



Authentication

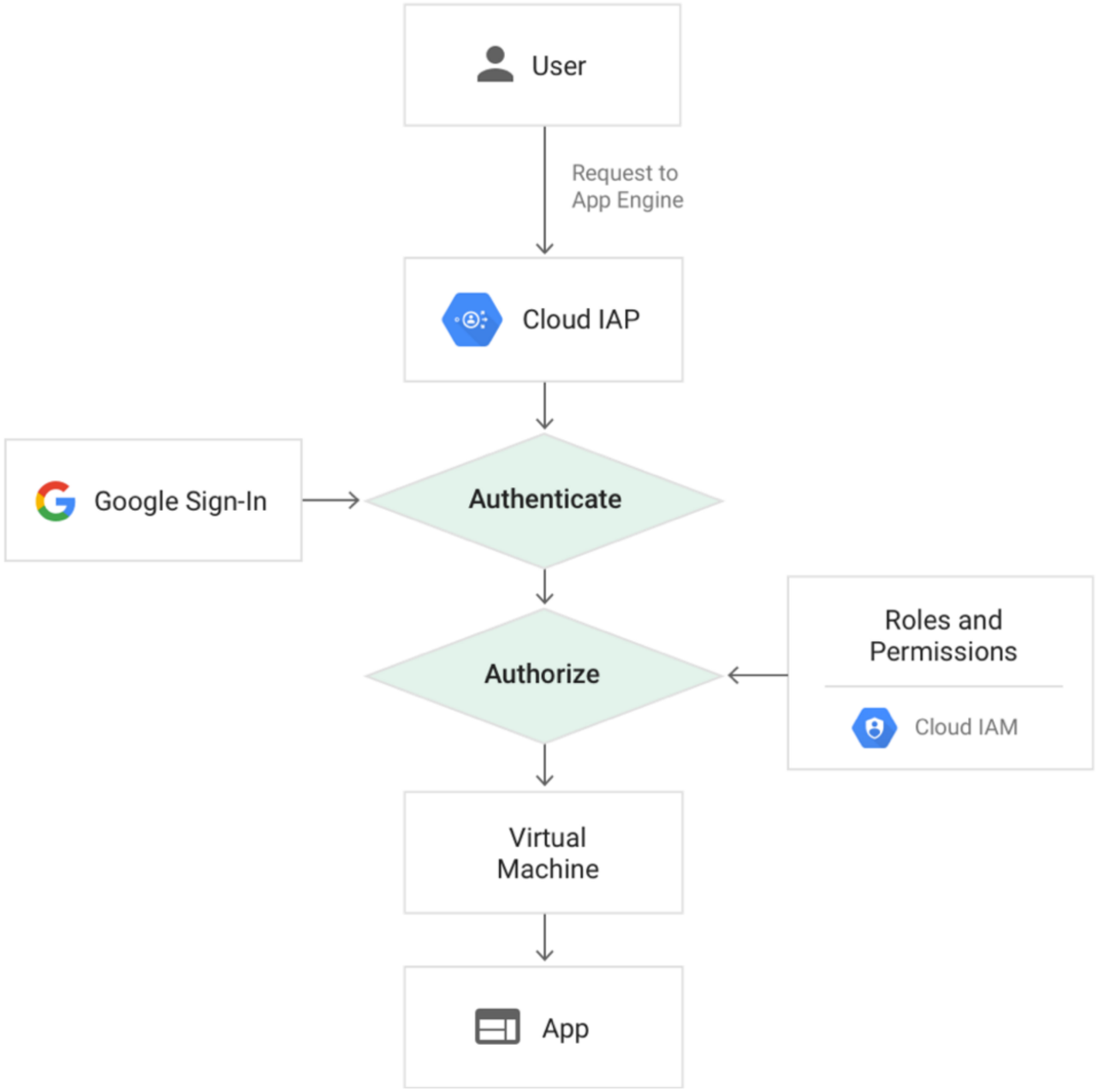
- Requests come from 2 sources:
  - App Engine
  - Cloud Load Balancing (HTTPS)

- Cloud IAP checks the user's browser credentials

- If none exist, the user is redirected to an OAuth 2.0 Google Account sign-in

- Those credentials sent to IAM for authorisation

Authorisation



- As before using IAM

IAP Limitations

- Will not protect against activity inside VM, e.g. someone SSH-ing into a VM or AppEngine flexible environment
- Need to configure firewall and load balancer to disallow traffic not from serving infrastructure
- Need to turn on HTTP signed headers

Data Loss Prevention

Data Loss Prevention API

- Understand and manage sensitive data in Cloud Storage or Cloud Datastore

- Easily classify and redact sensitive data
  - Classify textual and image-based information
  - Redact sensitive data from text files, and classify

Classification

- Input

```
Please update my records with the following information:
Email address: foo@example.com

National Provider Identifier: 1245319599

Driver's license: AC333991
```

- Output

  - Information type - specific types of sensitive information
  - Likelihood
  - Offset - location where that sensitive information occurs

Classification

- Input

```
Please update my records with the following information:
Email address: foo@example.com

National Provider Identifier: 1245319599

Driver's license: AC333991
```

- Output

InfoType	Likelihood	Offset
US_HEALTHCARE_NPI	VERY_LIKELY	122
EMAIL_ADDRESS	LIKELY	72
US_DRIVERS_LICENSE_NUMBER	LIKELY	155
CANADA_BC_PHN	VERY_UNLIKELY	122
UK_TAXPAYER_REFERENCE	VERY_UNLIKELY	122
CANADA_PASSPORT	VERY_UNLIKELY	155

Redaction

- Input

```
Please update my records with the following information:
Email address: foo@example.com

National Provider Identifier: 1245319599

Driver's license: AC333991
```

- Output

```
Please update my records with the following information:
Email address: ***

National Provider Identifier: ***

Driver's license: ***
```

Wow. How?

- ML-based:
  - Contextual Analysis
  - Pattern Matching
- Rule-based:
  - Checksum
  - Word and phrase list

Deployment Manager

https://cloud.google.com/dlp/docs/

Configuration

- It describes all the resources you want for a single deployment and this file written in YAML syntax.
- This lists each of the resources you want to create and it's respective resource properties.
- A configuration must contain a resource. Resource must contain three components :-
    - a)Name-user-defined string for identification.
      b)Type-Type of resource being deployed
      c)Properties-Parameters of the resource type

Templates

- Parts of the configuration and abstracted into individual building blocks. This file is written in python or jinja2.
- They are much more flexible than individual configuration files and intended to support easy portability across deployments.
- The interpretation of each template eventually must result in the same YAML syntax.

Resource

- Which represents a single API resource and provided by Google-managed base type.

- API resource provided by a Type Provider.

- To specify a resource- provide a Type for that resource.

Types

- Which represents a single API resource or set of resources and more important for resource creation.

- Base type- Creates single primitive resource and type provider used to create additional base types.

- Composite base types contains one or more templates - preconfigured to work together.

Manifest

- It is a read only object contains original configuration.
- At the time of updation Deployment manager generates manifest.
- Manifest is useful for solving troubleshooting issue.

Deployment

- Deployment is a collection of resources,deployed and managed together.

Access Control Options

Access control for users

- If the users have access permission to our project then they can create configurations and deployments.
- IAM(Identity and Access Management)-Support predefined and primitive roles.
- Primitive roles-map directly to the legacy project owner, editor, and viewer roles.
- eg:-

Role	Includes Permission (s)	For resource type
`roles/deploymentmanager.viewer`	`deploymentmanager.deployments.get`	Deployment
	`deploymentmanager.manifests.get`	Manifest

Access control for Deployment Manager

- Deployment Manager uses the credentials of Google API service account for create Google Cloud Platform resources.
- The Google APIs service account is automatically granted editor permissions on the project.
- The service account exists indefinitely with the project and is only deleted when the project is deleted.

Runtime Configurator Fundamentals

What is Runtime Configurator?

- Which lets you define and store and store as hierarchy of key value pairs in the google cloud.

- These key value pairs are used for Dynamically configure services,Communicate service states,Send notification of changes to data and Share information between multiple tiers of services.

- Runtime configurator also offers watcher and waiter service.

Concepts

- Config resource-Which contains a hierarchical list of variables.
- Variables are the key value pairs belongs to RuntimeConfig resource.
- Watchers can use the watch() method to watch a variable and return when the variable changes, and finally waiters which have a cardinality condition.
- eg:-

```
/foo/variable1 = "value1"
/foo/variable2 = "value2"
/bar/variable3 = "value3" # Not /foo path
```

Cloud Key Management

Object hierarchy

Project

- Cloud KMS resources belong to a project.

- Resources have permission when the account with primitive IAM roles on any project with cloud KMS resources.

Location

- Which represents geographical data centre location of where requests to Cloud KMS regarding the given resources.
- If locations are close to you,which is more fast and reliable.
- Global- If we using this resource, KMS resources are available from multiple data centers.

KeyRing

- KeyRing is a grouping of CryptoKeys for organisational purpose.

- Combination of CryptoKey and KeyRing-No need act individually.

CryptoKey

- Cryptographic key used for special purpose.
- CryptoKey is used to protect some corpus of data.
- Can encrypt and decrypt by users with the permissions of CryptoKey.

CryptoKey Version

- Represents the key materials, Which have many versions and starting from 1.
- Which have states like enabled,disabled and scheduled.
- Primary version will use for the encryption of data.

Key States

CryptoKey Version state

- CryptoKeyVersion has a state:-
  a)Enabled(ENABLED)-Used for encryption and decryption of cryptokey requests.
  b)Disabled(DISABLED)-May not be used,placedback to enabled state.
  c)Scheduled for destruction(DESTROY-SCHEDULED)-For destruction and destroyed soon.
  d)Destroyed(DESTROYED)-Key material no longer stored in cloud KMS.

Primary CryptoKeyVersion

- Used for the time of encryption.
- At any given point of time,One version of the cryptography cab be primary.
- If primary CryptoKeyVersion is disabledCryptoKey cannot encrypt the data.

CryptoKey and CryptoKeyVersion states

- CryptoKeyVersion Contains the states.
- If Primary CryptoKeyVersion is enabled, Then only CryptoKey to encrypt the data.
- At the time of decryption, no need for the primary version.

Key Rotation

Rotation in cloud KMS

- At the time of generating crypto key version of crypto key marking that is primary key.

- Each Crypto key version rotated to primary key, at the point to encrypt the data .

Frequency of key rotation

- Regular rotation and Irregular rotation are the two rotations of Encryption keys.
- Regular rotation take the time for data encrypted with single key.
- Irregular rotation disable the to the restrict access of data.

Automatic rotation

- CryptoKey rotation schedule d using cloud command or via Google Cloud Platform Console.

- Rotation schedule is scheduled by rotation period and next rotation time.

Manual Rotation

- Used for irregular key rotation
- Manually rotated using cloud command line or via Cloud Platform Console.
-

Separation of duties

Setting up Cloud KMS in separate project

- The user and owner can access and manage the project at the time of run.
  a)Create the key project without an owner-recommended.
  b) Grant an owner role for your key project-Not recommended.

Choosing the right IAM roles

- In smaller organisation-Owner,editor and viewer provide sufficient granularity for key management.

- In large organisation-Separation of duties required.
- The roles they recommend are:-
- a)For the business owners whose application requires encryption.
- b)For the user managing cloud
- c)For the user or service using keys for encryption and decryption operations.

Secret Management with Cloud KMS

Overview of secret management

- Common ways to storing secrets are Code of binaries, Deployment Management etc...

- Authorisation, Verification of usage,Encryption at rest,RotationAnd Isolation are the security concerns.

- Consistency and Version Management describes the functionality concerns of secret management .

Choosing a secret management solution

- Storing secrets in code,encrypted with a key from Cloud KMS and Storing secrets in storage bucket in Google cloud storage are some example for approaches.

- Rotating secrets, Cache secrets locally and using a separate solution or problem are some of the changing secrets.

- Encryption options are Use application layer encryption using a key in Cloud KMS and aUse the default encryption built into the Cloud Storage bucket.

- Managing access to secrets are Access controls on the bucket in which the secret is stored and Access controls on the key which encrypts the bucket in which the secret is stored.
- Key rotation and Secret rotation are some example for secret management .
- Permission management without a service account requires several users: An organizational-level administrator,A second user that has the a storage,A third user with the cloudkms.admin role and A fourth user that has both the storage.objectAdmin and cloudkms.cryptoKeyEncrypterDecrypter roles.

Envelope encryption

- The key used to encrypt data itself is called a data encryption key (DEK).

-The DEK is encrypted (or wrapped) by a key encryption key (KEK)

Exfiltration Prevention

Data Exfiltration

An authorized person extracts data from the secured systems where it belongs, and either shares it with unauthorized third parties or moves it to insecure systems. Data exfiltration can occur due to the actions of malicious or compromised actors, or accidentally.

Types

- Outbound email
- Downloads to insecure devices
- Uploads to external services
- Insecure cloud behaviour
- Rogue admins, pending employee terminations

Don'ts for VMs

- Don't allow outgoing connections to unknown addresses
- Don't make IP addresses public
- Don't allow remote connection software e.g. RDP
- Don't give SSH access unless absolutely necessary

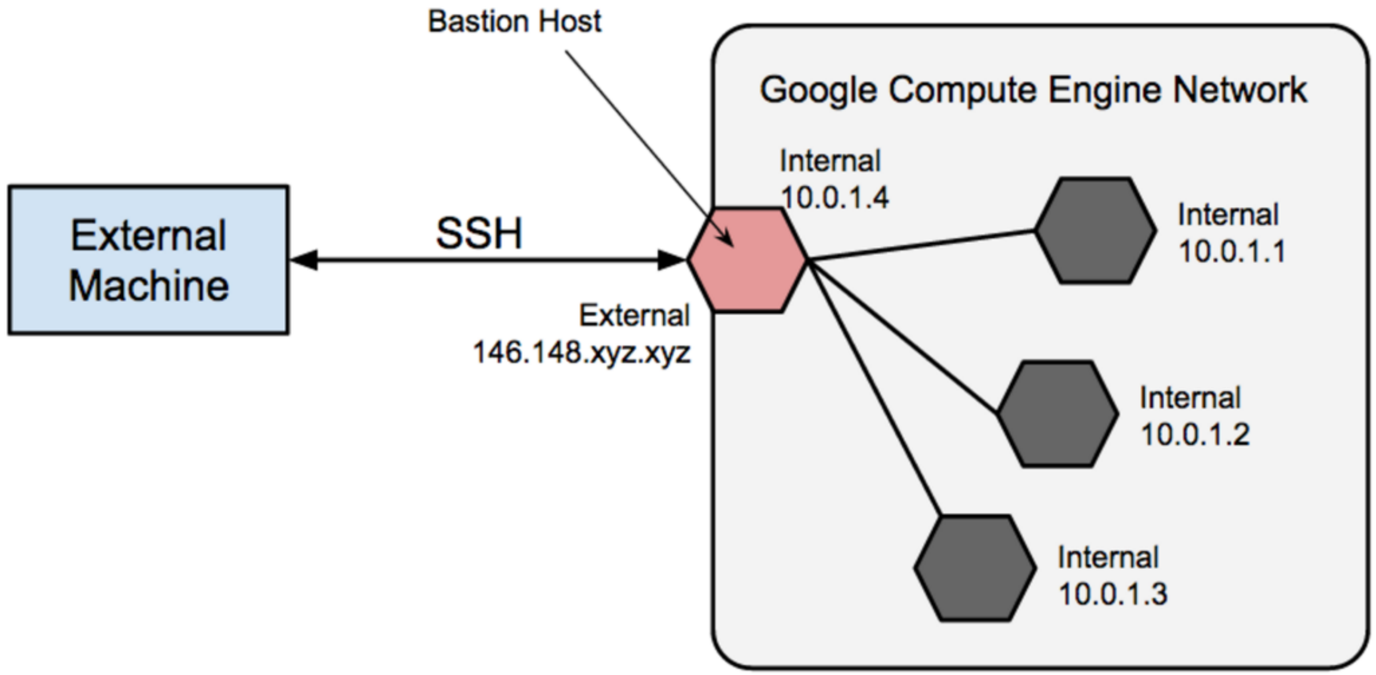
Dos for VMs

- Use VPCs and firewalls between them
- Use a bastion host as a chokepoint for access
- Use Private Google Access
- Use Shared VPC, aka Cross-Project Networking

Dos for VMs

- Use VPCs and firewalls between them
- Use a bastion host as a chokepoint for access
- Use Private Google Access
- Use Shared VPC, aka Cross-Project Networking

Bastion Hosts



- Limit source IPs that can communicate with the bastion

- Configure firewall rules to allow SSH traffic to private instances from only the bastion host.