Load Balancing

Architecting with GCP Fundamentals: Infrastructure

CLOUD LOAD BALANCING

VIRTUAL MACHINE AUTOMATION AND LOAD BALANCING

Last modified 2017-11-27

Use load balancing to distribute user requests among sets of instances

Global	**HTTP(S) load balancing**	Distributes HTTP(S) traffic among groups of instances based on: • Proximity to the user • Requested URL • Both	External
	SSL Proxy load balancing	Distributes SSL traffic among groups of instances based on proximity to the user.	
	TCP Proxy load balancing	Distributes TCP traffic among groups of instances based on proximity to the user.	
Regional	**Network load balancing**	• Distributes traffic among a pool of instances within a region. • Can balance any kind of TCP/UDP traffic.	
	Internal load balancing	Distributes traffic from GCP virtual machine instances to a group of instances in the same region.	Internal

Google Cloud

For more information, see:
Load balancing: https://cloud.google.com/compute/docs/load-balancing/
HTTP(S) load balancing: https://cloud.google.com/compute/docs/load-balancing/http
SSL Proxy load balancing:
https://cloud.google.com/compute/docs/load-balancing/tcp-ssl/
TCP Proxy load balancing:
https://cloud.google.com/compute/docs/load-balancing/tcp-ssl/tcp-proxy
Network load balancing:
https://cloud.google.com/compute/docs/load-balancing/network
Internal load balancing:
https://cloud.google.com/compute/docs/load-balancing/internal/

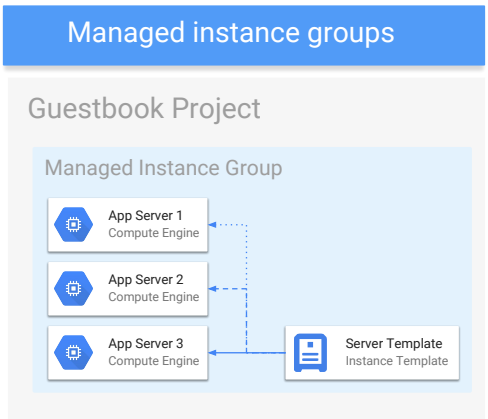
Agenda

- **Managed Instance Groups**
- HTTP(S) load balancing
- Cross-region and content-based load balancing
- SSL proxy/TCP proxy load balancing
- Network load balancing
- Internal load balancing
- Load balancing best practices
- Lab
- Quiz

Google Cloud

Managed Instance Groups

- Deploys identical instances based on instance template
- Instance group can be resized
- Manager ensures all instances are in RUNNING state
- Typically used with autoscaler
- Can be single zone or regional



Google Cloud

Use managed instance groups for most situations. Managed instance groups provide the following benefits:

- Managed instance groups use an instance template to define the properties for every instance in the group. You can easily update all of the instances in the group by specifying a new template in a rolling update.
- When your applications require additional compute resources, managed instance groups can automatically scale the number of instances in the group.
- Managed instance groups can work with load balancing services to distribute network traffic to all of the instances in the group.
- If an instance in the group stops, crashes, or is deleted by an action other than the instance groups commands, the managed instance group automatically recreates the instance so it can resume its processing tasks. The recreated instance uses the same name and the same instance template as the previous instance, even if the group references a different instance template.
- Managed instance groups can automatically identify and recreate unhealthy instances in a group to ensure that all of the instances are running optimally.

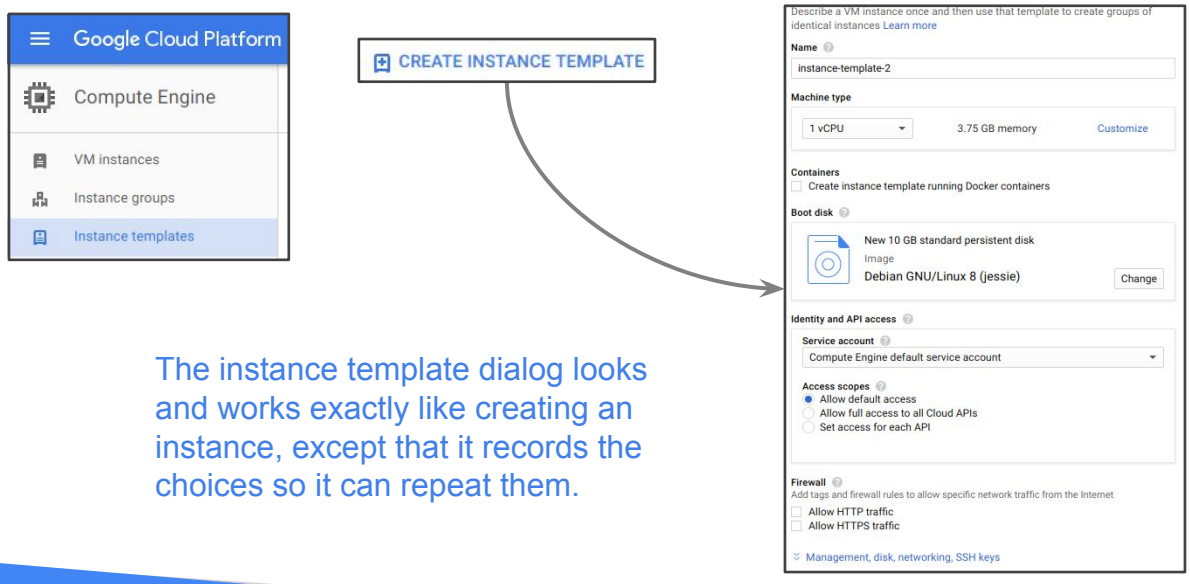
For more information on managed instance groups, see: https://cloud.google.com/compute/docs/instance-groups/#create_managed_group.

In addition to managed instance groups that belong in single zones, you can create regional managed instance groups, which distributes instances across multiple zones in the same region. Regional managed instance groups improve your application availability by spreading your instances across three zones. For example, a regional

managed instance group in us-east1 will have instances in all three zones within us-east1: us-east1-d, us-east1-b, and us-east1-c. Regional managed instance groups also supports autoscaling, network load balancing, and HTTP(S) load balancing.

For more information on distributing instances using regional managed instance groups, see: https://cloud.google.com/compute/docs/instance-groups/distributing-instances-with-regional-instance-groups.

Instance Template



The instance template dialog looks and works exactly like creating an instance, except that it records the choices so it can repeat them.

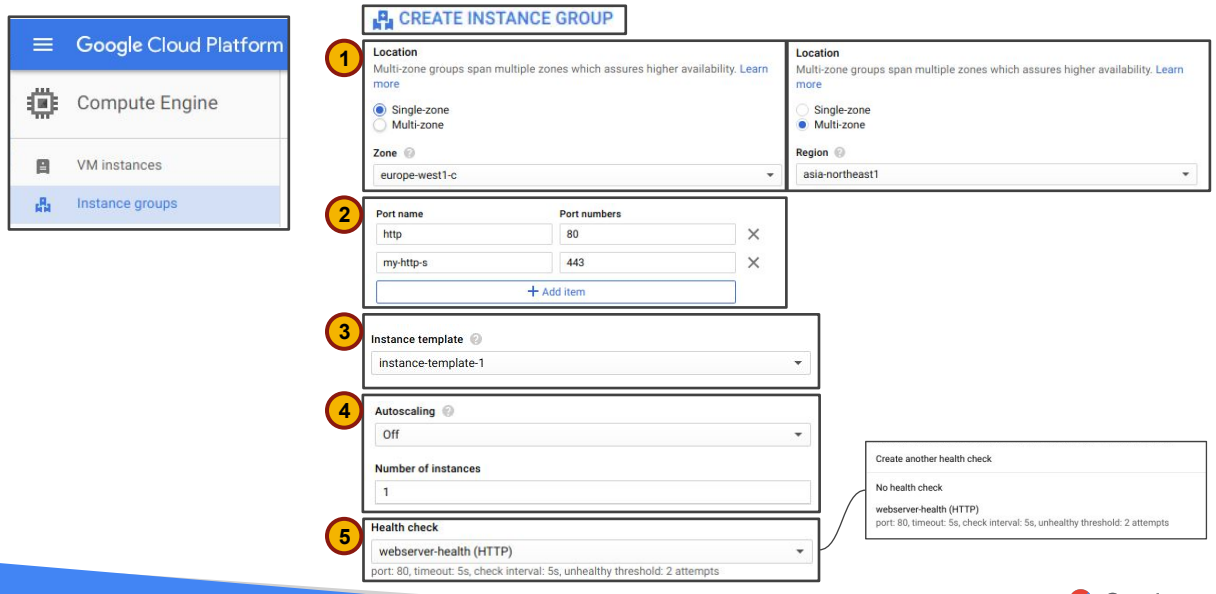
Google Cloud

Selecting **[X]** Allow HTTP traffic or **[X]** Allow HTTPS traffic will create firewall the rules.

Managed instance groups

- Create an instance template
- Create a managed instance group of n instances
- The instance group manager automatically populates the instance group based on the instance template

Instance Group

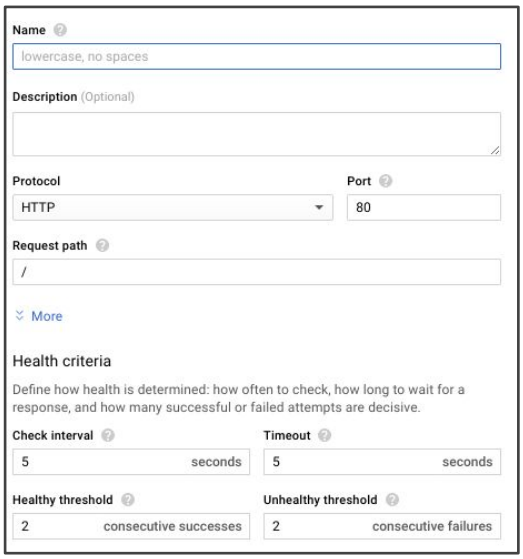


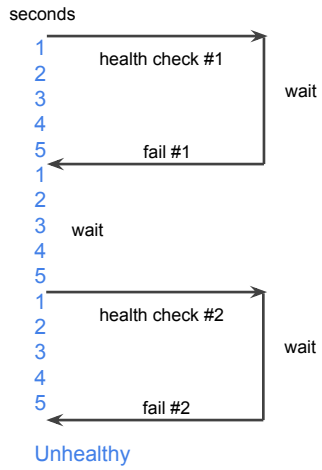
Instance Group configuration.
(1) Location of VMs. Single zone (pick the zone), Multi-zone (pick the region). VMs are evenly distributed in three zones in the region.
(2) Symbolic port name. The name is used by the Forwarding Rules to pass traffic to the instance group.
(3) This is where you select the pre-existing Instance Template that will be used to create VMs in the group.
(4) Autoscaling [Off] and the number of instances in the group. For availability when using a Multi-zone group, 3 minimum (one in each zone) is recommended.
(5) Health check. When the health check is set, if a VM meets the "unhealthy" conditions, it is deleted and a new replacement is created. This process is called **Autohealing**.

Use cases for Managed Instance Groups

- Scale up / scale down
  - Target with autoscaler
- Deploy software / rolling update
- High availability

Create another health check

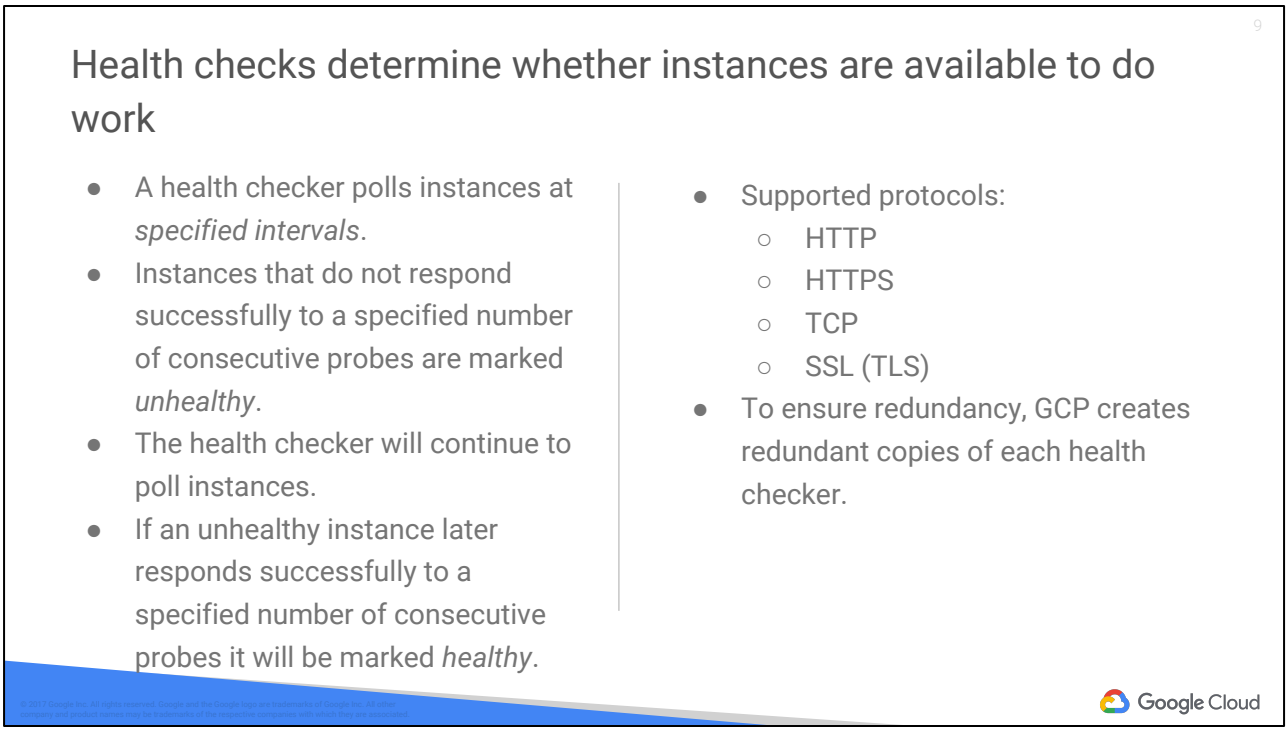




Google Cloud

Protocol: HTTP or HTTPS

Check interval - how long to wait between attempts to check the health of a particular VM
Timeout - how long to wait for a response before declaring that the check attempt has failed
Health threshold - how many consecutive successes indicates that the VM is "healthy" and should receive traffic
Unhealthy threshold - how many consecutive failures indicates that the VM is "unhealthy" and should be replaced



A health checker polls instances at specified intervals. Instances that do not respond successfully to a specified number of consecutive probes are marked as UNHEALTHY. No new connections are sent to such instances, though existing connections are allowed to continue. The health checker continues to poll unhealthy instances. If an instance later responds successfully to a specified number of consecutive probes, it is marked HEALTHY again and can receive new connections.

GCP health checks support HTTP, HTTPS, TCP, and SSL (TLS).

To ensure high availability, GCP creates redundant copies of each health checker. These redundant health checkers also probe your instances. If any health checker fails, a redundant one can take over with no delay. If you examine the logs on your instance, you might see health check polling happening more frequently than than you have configured. This is because the redundant health checkers are also following your settings. These redundant health checkers are created automatically and are not separately user configurable.

For more information, see:
Health Checks: https://cloud.google.com/compute/docs/load-balancing/health-checks
HTTP health checks:
https://cloud.google.com/sdk/gcloud/reference/compute/health-checks/create/http
HTTPS health checks:
https://cloud.google.com/sdk/gcloud/reference/compute/health-checks/create/https
TCP health checks:

https://cloud.google.com/sdk/gcloud/reference/compute/health-checks/create/tcp
SSL (TLS) health checks:
https://cloud.google.com/sdk/gcloud/reference/compute/health-checks/create/ssl

Managed instance groups offer autoscaling capabilities

- Autoscaling allows you to automatically add/remove instances from a managed instance group based on:
  - Increases in load
  - Decreases in load
- Define the autoscaling policy and the autoscaler performs automatic scaling based on the measured load.

Google Cloud

Managed instance groups offer autoscaling capabilities that allow you to automatically add or remove instances from a managed instance group based on increases or decreases in load. Autoscaling helps your applications gracefully handle increases in traffic and reduces cost when the need for resources is lower. You just define the autoscaling policy and the autoscaler performs automatic scaling based on the measured load.

Autoscaling works by scaling up or down your instance group. That is, it adds more instances to your instance group when there is more load (upscaling), and removes instances when the need for instances is lowered (downscaling).

For more information, see: https://cloud.google.com/compute/docs/autoscaler/

Connection Draining

- Connection draining delays the termination of an instance until existing connections are closed
  - New connections to the instance are prevented
  - Instance preserves existing sessions until they end OR a designate timeout is reached (1 to 3600 seconds)
  - Minimizes interruption for users
- Connection draining is triggered when an instance is removed from an instance group
  - Manual removal, resizing, autoscaling, etc.

Google Cloud

For more information on connection draining, see
https://cloud.google.com/compute/docs/load-balancing/enabling-connection-draining

Additional features

- Instance Group Updater BETA
  - Zero-downtime and staggered releases
  - Use Instance Group Updater to apply a rolling update
  - Apply canary updates with rollback

- Autohealing BETA
  - Automated server monitoring and restarts
  - If HTTP health check sees a service has failed on an instance, re-create instance where service failed

Google Cloud

Need to update software on running service, you'd kill a server and replace it with another. All new instances are in the new template.
The instance Group Update updates instances by applying a new instance template. Updates can be proactive or opportunistic, and can roll out gradually to all instances or only to a subset of instances (and rolled forward after test period). For more information on rolling updates, see:
https://cloud.google.com/compute/docs/instance-groups/updating-managed-instance-groups

For more information on autohealing, see:
https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-managed-instances#monitoring_groups

For more information on rolling updates, see:
https://cloud.google.com/compute/docs/instance-groups/#starting_an_update. For more information on autohealing, see:
https://cloud.google.com/compute/docs/instance-groups/#monitoring_groups.

Agenda

- Managed Instance Groups
- **HTTP(S) load balancing**
- Cross-region and content-based load balancing
- SSL proxy/TCP proxy load balancing
- Network load balancing
- Internal load balancing
- Load balancing best practices
- Lab
- Quiz

Google Cloud

HTTP(S) load balancing

- HTTP(S) load balancing:
  - Provides global load balancing for HTTP(S) requests destined for your instances.
  - Supports both IPv4 and IPv6 addresses for client traffic.
  - Uses instance groups to organize instances.
- HTTP requests can be load balanced based on ports:
  - 80
  - 8080
- HTTPS requests can be load balanced on port:
  - 443
-

Google Cloud

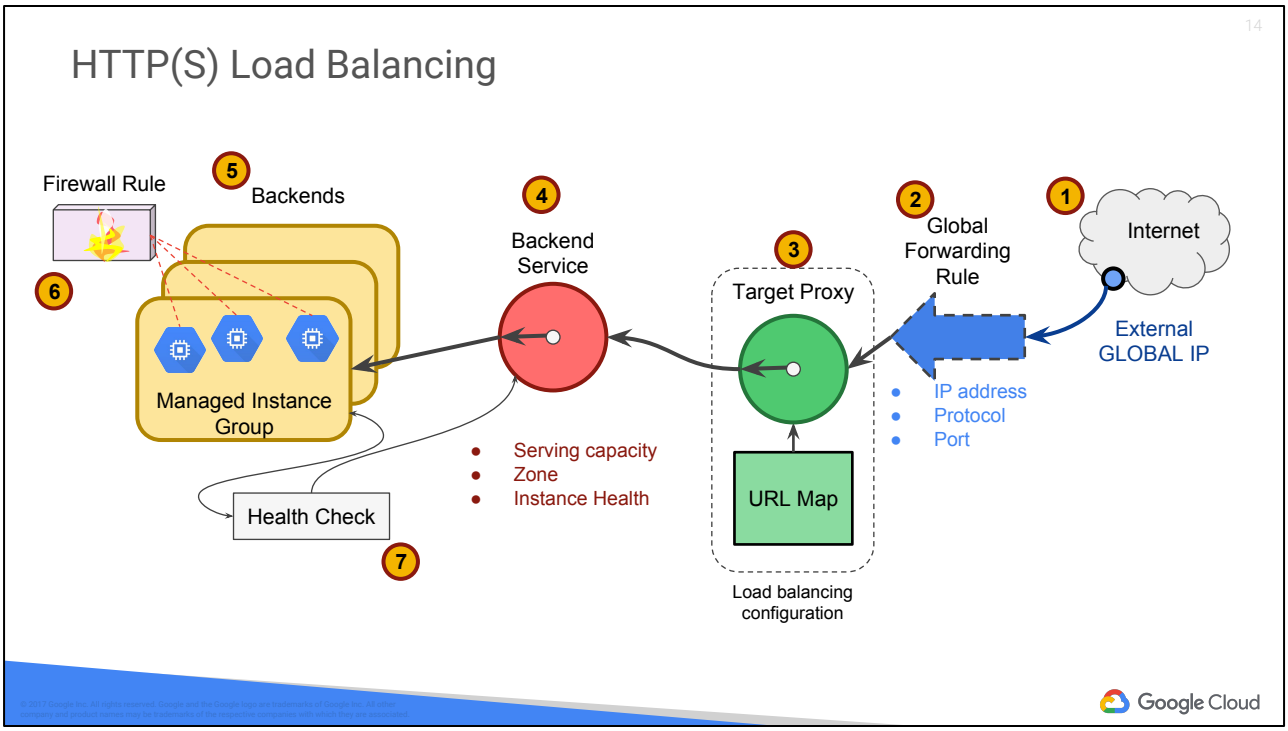
Google Cloud Platform (GCP) HTTP(S) load balancing provides global load balancing for HTTP(S) requests destined for your instances. You can configure URL rules that route some URLs to one set of instances and route other URLs to other instances. Requests are always routed to the instance group that is closest to the user, provided that group has enough capacity and is appropriate for the request. If the closest group does not have enough capacity, the request is sent to the closest group that does have capacity.

HTTP(S) load balancing supports both IPv4 and IPv6 addresses for client traffic. Client IPv6 requests are terminated at the global load balancing layer, then proxied over IPv4 to your backends.

HTTP requests can be load balanced based on port 80 or port 8080. HTTPS requests can be load balanced on port 443.

The load balancer acts as an HTTP/2 to HTTP/1.1 translation layer, which means that the web servers always see and respond to HTTP/1.1 requests, but that requests from the browser can be HTTP/1.0, HTTP/1.1, or HTTP/2. HTTP/2 server push is not supported.

For more information, see
HTTPS load balancing: https://cloud.google.com/compute/docs/load-balancing/http/
Instance groups: https://cloud.google.com/compute/docs/instance-groups



(1) Recall that external IPs are related to internal resources through Network Address Translation (NAT). External REGIONAL IPs can be used by VMs or by Network Load Balancers. External GLOBAL IPs can only be used by GLOBAL forwarding rules. External GLOBAL IPs cannot be assigned to any regional or zonal resource. External IPs can only be assigned to one Global Forwarding Rule.
(2) One Global Forwarding Rule can only forward traffic for a single protocol and port, as follows: **HTTP** { **80** | 8080 }, **HTTPS** { **443** }
SSL { 25 | 43 | 110 | 143 | 105 | **443** | 465 | 587 | 700 | 993 | 995 }, **TCP** { **25** | 43 | 110 | 143 | 105 | 443 | 465 | 587 | 700 | 993 | 995 }
The Global Forwarding Rule requires a Target Proxy to already exist before it can be configured. Support for HTTP and HTTPS requires two rules.
(3) A Target Proxy is specific to a single protocol (HTTP, HTTPS, SSL, TCP)

Backend Services and Backends

- Backend services are comprised of...
  - A health check
  - Session affinity settings
  - One or more backends
- A backend consists of...
  - An instance group (managed or unmanaged)
  - A balancing mode (CPU utilization or Rate in request/second)
  - A capacity scaler (ceiling % of CPU/Rate targets)
- A backend service may have up to 500 endpoints per zone

Google Cloud

The backend service will not send requests to instances reported as unhealthy by the Health Check.

Session affinity can be set so that requests from the same client end up going to the same instance.

Client IP affinity directs requests from the same origin IP to the same server. NAT and other network routing technologies can cause requests from multiple different users to look like they come from the same address causing many users to get routed to the same instance. On the other hand, one user who moves from network to network may be seen as two different users, and not end up directed to the same instance.

Generated cookie affinity causes the load balancer to issue a cookie named GCLB on the first request from a user, then directs subsequent requests with the same cookie to the same instance.

Session affinity can break…

- If an instance group runs out of capacity and traffic is routed to another zone
- If autoscaling change capacity and load is reallocated
- If the target instance fails health checks

The balancing mode and capacity scaler designate how maximum utilization for a backend. The balancing mode uses one of the following metrics to determine if the backend instance group is at capacity and the load balancer needs to send requests

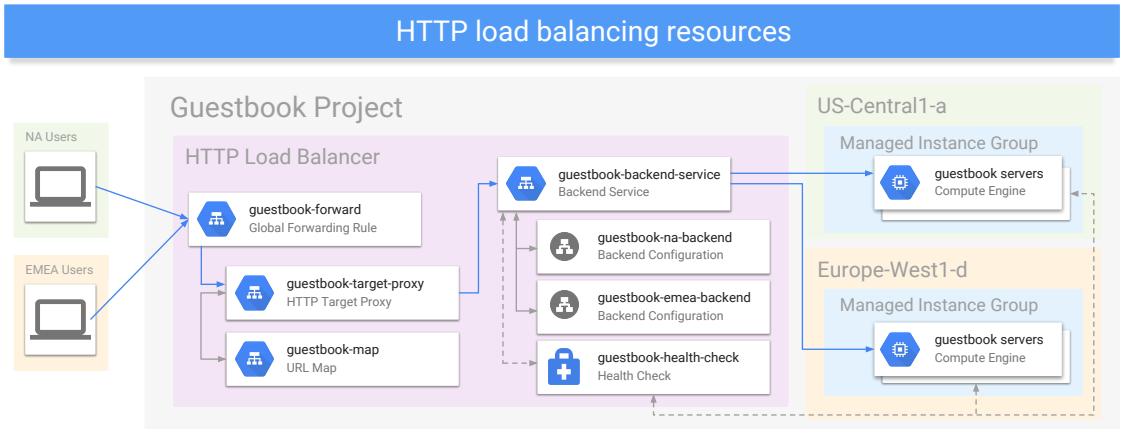
to another backend:

- CPU utilization (average of CPU use across all instances in backend)
- Rate: Maximum Requests/second/instance
- Rate: Maximum Requests/second/group
- CPU utilization and Rate (either RPS/instance or RPS/group) - either condition triggers at-capacity state

The Capacity scaler is an additional setting that directs the load balancer to only direct requests to a given backend instance group as long as utilization is below a % of the balancing mode maximum. For example, if balancing mode is utilization and Max CPU utilization is 80%, setting the capacity scaler to 50% would mean the load balancer would see the backend as being at capacity when CPU utilization is at 40% average across the instance group.

For more information on Backend Services and Backends, see https://cloud.google.com/compute/docs/load-balancing/http/backend-service

HTTP(S) Load Balancing Example



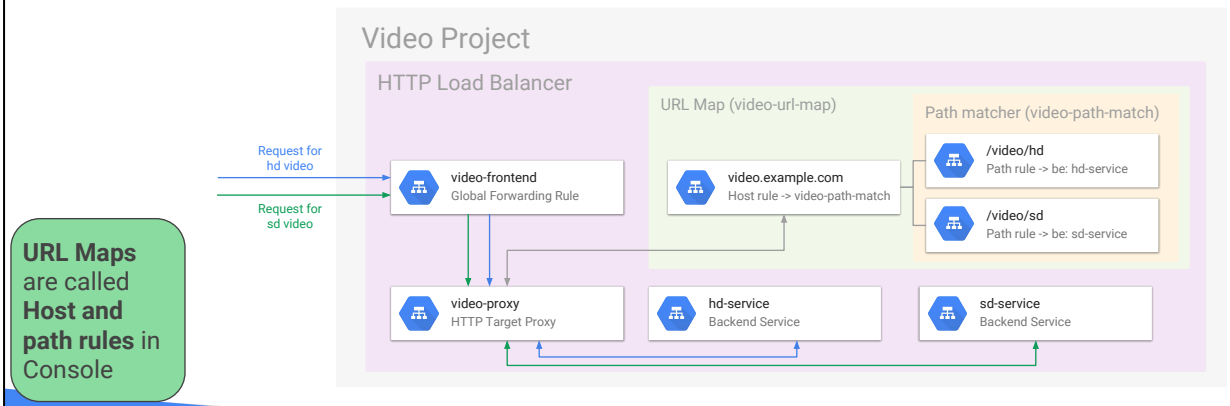
Google Cloud

Distributes HTTP(S) traffic among instance groups based on proximity to user or URL or both

For an overview of setting up HTTP(S) Load Balancing, see https://cloud.google.com/compute/docs/load-balancing/http/#illegal_request_handling

URL Maps

- The load balancer can direct traffic to different instances based on the incoming URL
- The URL map designates which requests to send where



Google Cloud

Notes:

URL Maps are comprised of:

- Host rules
- Path matchers
  - Path rules

Host rules map host names to patch matchers. Patch matchers are comprised of one or more path rules with URL paths mapped to specific backend services.

The default configuration for a URL map is a single patch matcher (/*), which is created automatically and routes all traffic to a the default service.

For more information on URL Maps, see
https://cloud.google.com/compute/docs/load-balancing/http/url-map

Traffic Allocation for Backend Services

- Instances included in backend services may reside across...
  - A zone
  - A region
  - Multiple regions
- Traffic is allocated by location and capacity
  - If the instances nearest the origin of request have capacity, the request is forwarded to that set of instances
  - If there are no healthy instances with capacity in a region, the request is sent to the next closest region with capacity
  - Request to a given region are distributed evenly across backend services in that region

Google Cloud

A backend service can span multiple regions. At the same time, there can be multiple backend services within the same region.

HTTPS Load Balancing

- HTTPS uses SSL (Secure Sockets Layer) to establish encryption for a secure link
- In HTTP(S) Load Balancing the Target Proxy terminates the client-originated SSL session, therefore, HTTP(S) requires the additional configuration of the SSL certificate
  - Provides a single place to maintain and update client certificates
- You can then load balance between the Target Proxy and the VMs using either TCP or a separate internally-originated SSL session

Google Cloud

When using Network Load Balancing with HTTPS, the SSL session terminates on the VMs.

SSL Certificates

- To use HTTPS or SSL load balancing, you must create at least one SSL certificate to be used by the target proxy for the load balancer.
- Each target proxy can be configured with up to 10 SSL certificates.
- For each SSL certificate, you create an SSL certificate resource.

Google Cloud

For more information, see:
SSL Certificates:
https://cloud.google.com/compute/docs/load-balancing/http/ssl-certificates
SSL Certificate Reource:
https://cloud.google.com/compute/docs/reference/v1/sslCertificates

HTTPS additional information

- An HTTPS target proxy accepts only TLS 1.0 and up when terminating client SSL requests
  - Speaks only TLS 1.0 and up when backend protocol is HTTPS
- The load balancer blocks illegal requests
  - See documentation for details

Google Cloud

Load Balancing Logging (Alpha) writes request information into Stackdriver logs:

- Logs HTTPRequest log fields and statusDetails that explains why the load balancer returned a given status
- Check documentation for release status

Agenda

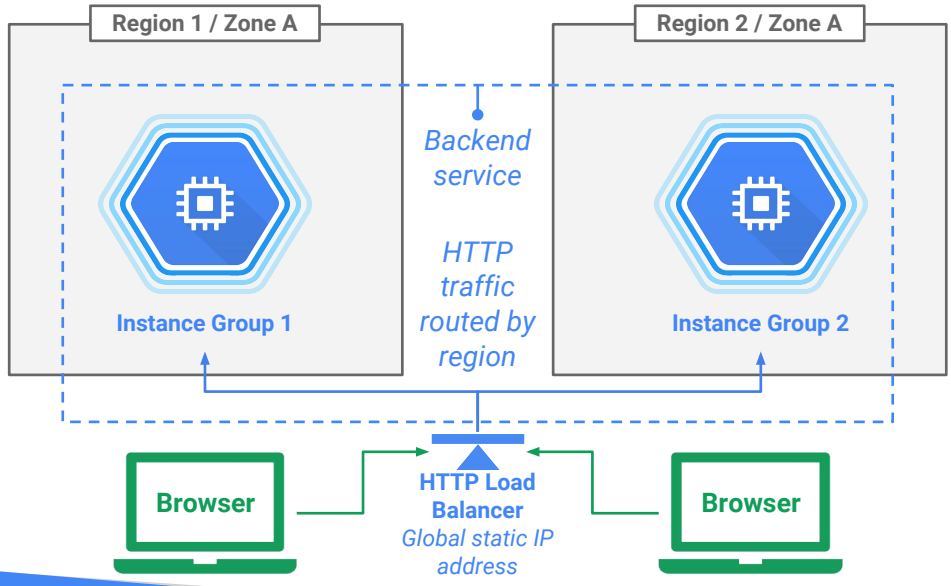
- Managed Instance Groups
- HTTP(S) load balancing
- **Cross-region and content-based load balancing**
- SSL proxy/TCP proxy load balancing
- Network load balancing
- Internal load balancing
- Load balancing best practices
- Lab
- Quiz

Google Cloud

Cross-Region Load Balancing

- HTTP/HTTPS only
- Cross-region using a single global IP address
- Requests routed to the closest region
- Automatically reroutes to next closest once capacity is reached
- Eliminates need for DNS-based load balancing

Example: Cross-Region Load Balancing



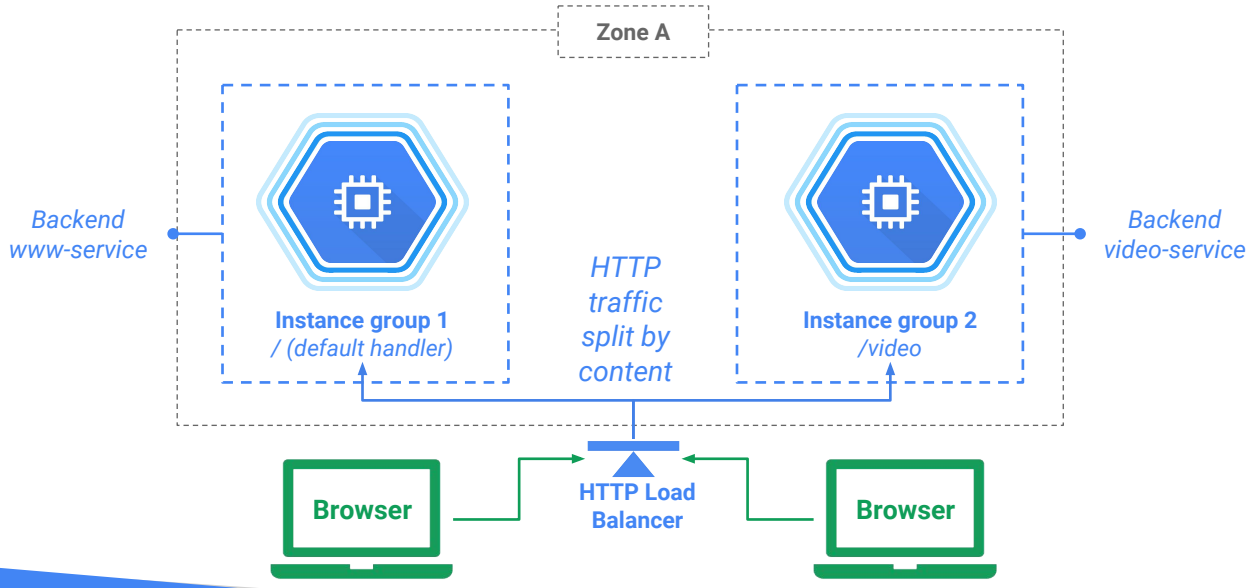
Content-Based Load Balancing

- HTTP/HTTPS only
- Create multiple backend services to handle content types
- Add path rules to backend services
  - `/video` for video services
  - `/static` for static content
- Configure different instance types for different content types

Google Cloud

https://cloud.google.com/compute/docs/load-balancing-and-autoscaling

Example: Content-Based Load Balancing



Agenda

- Managed Instance Groups
- HTTP(S) load balancing
- Cross-region and content-based load balancing
- **SSL proxy/TCP proxy load balancing**
- Network load balancing
- Internal load balancing
- Load balancing best practices
- Lab
- Quiz

Google Cloud

SSL proxy

- Another kind of load balancing
  - Intended for non-HTTP(S) traffic using SSL
  - For HTTP(S) traffic, use HTTP(S) Load Balancing
  - Supports both IPv4 and IPv6 addresses for client traffic
- Benefits:
  - Performs global load balancing of SSL traffic, routing clients to the closest instance with capacity, similar to what HTTP(S) Load Balancing does for HTTP(S) traffic
  - Client IPv6 requests are terminated at the global load balancing layer, then proxied over IPv4 to your backends

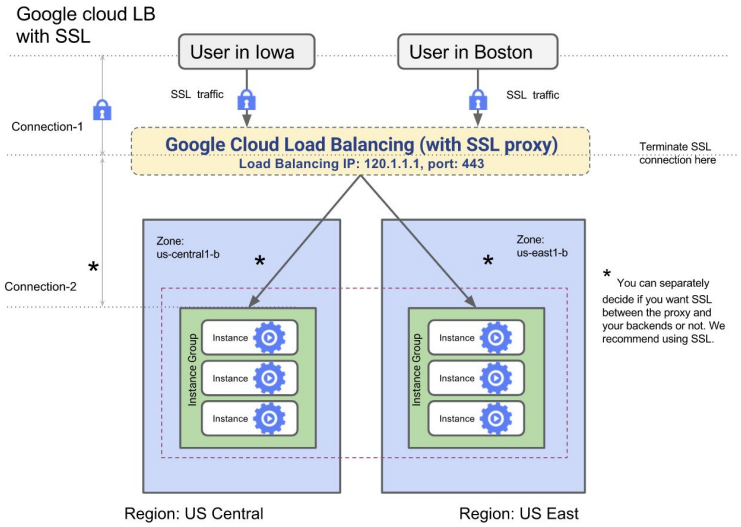
Google Cloud

Advantages of SSL proxy include:

- Intelligent routing
- Better utilization of the virtual machine instances
- Certificate management
- Security patching
- SSL proxy supports the following ports: 25, 43, 110, 143, 195, 443, 465, 587, 700, 993, 995, 1883, 5222

For more information on the Google Cloud SSL proxy, see https://cloud.google.com/compute/docs/load-balancing/tcp-ssl/

Cloud Load Balancing with SSL proxy



Google Cloud

In the example,traffic from users in Iowa and Boston is terminated at the global load balancing layer, and a separate connection is established to the selected backend instance.

TCP proxy

- TCP proxy load balancing:
  - 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
  - Intended for non-HTTP traffic
  - Supports IPv4 and IPv6 addresses for client traffic
    - Client IPv6 requests are terminated at the global load balancing layer, then proxied over IPv4 to your backends

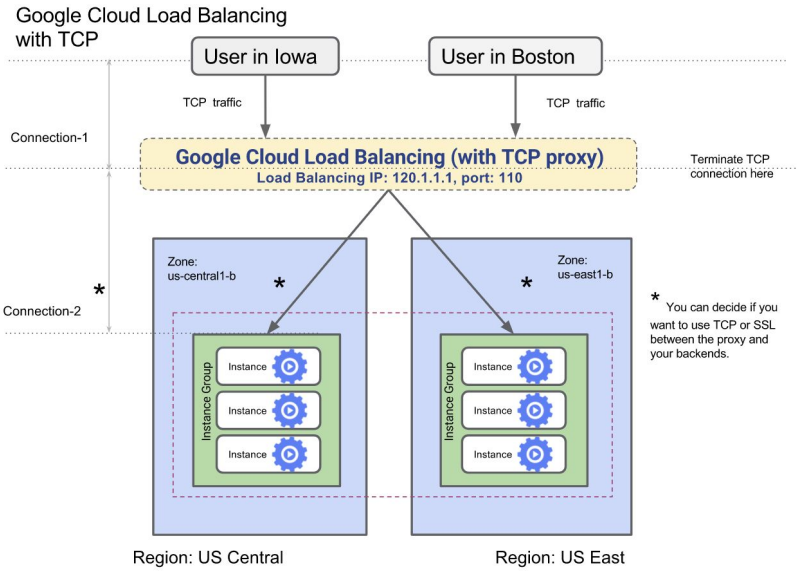
Google Cloud

Advantages of TCP Proxy load balancing include:

- Intelligent routing
- Security patching
- TCP Proxy supports the following ports: 25, 43, 110k, 143, 195, 443, 465, 587, 700, 993, 995, 1883, 5222

For more information, see:
https://cloud.google.com/compute/docs/load-balancing/tcp-ssl/tcp-proxy

Cloud Load Balancing with TCP proxy



Google Cloud

In the example, traffic from the users in Iowa and Boston is terminated at the global load balancing layer, and a separate connection is established to the selected backend instance.

Agenda

- Managed Instance Groups
- HTTP(S) load balancing
- Cross-region and content-based load balancing
- SSL proxy/TCP proxy load balancing
- **Network load balancing**
- Internal load balancing
- Load balancing best practices
- Lab
- Quiz

Network Load Balancing

- Network load balancing allows you to balance load of your systems based on incoming IP protocol data.
  - Uses forwarding rules that point to target pools
  - Target pools
    - List the instances available for load balancing
    - Define which type of health check should be performed on the instances
- Network load balancing is a regional, non-proxied load balancer
- Can be used to load balance the following types of traffic:
  - UPD
  - TCP
  - SSL

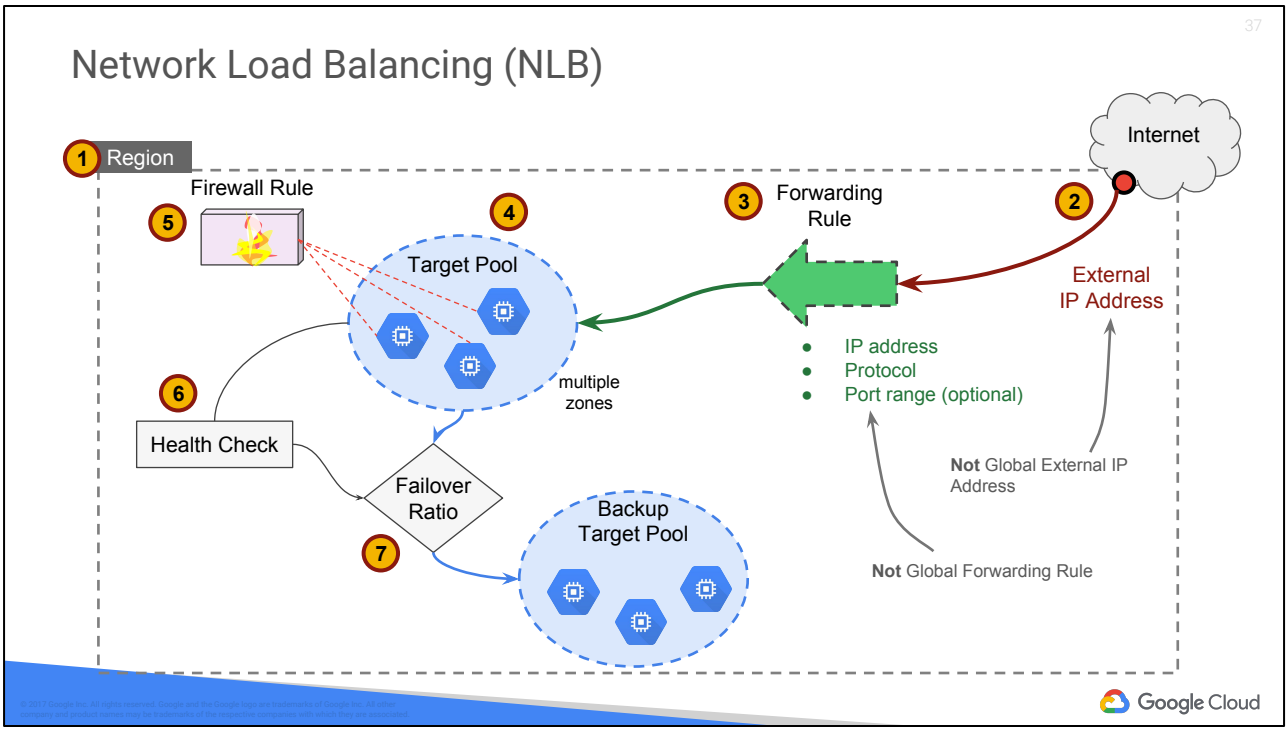
Google Cloud

Network load balancing allows you to balance load of your systems based on incoming IP protocol data, such as address, port, and protocol type.

Network load balancing uses forwarding rules that point to target pools, which list the instances available for load balancing and define which type of health check that should be performed on these instances.

Network load balancing is a regional, non-proxied load balancer. You can 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. A Network load balancer is a pass-through load balancer. It does not proxy connections from clients.

For more information, see:
https://cloud.google.com/compute/docs/load-balancing/network/

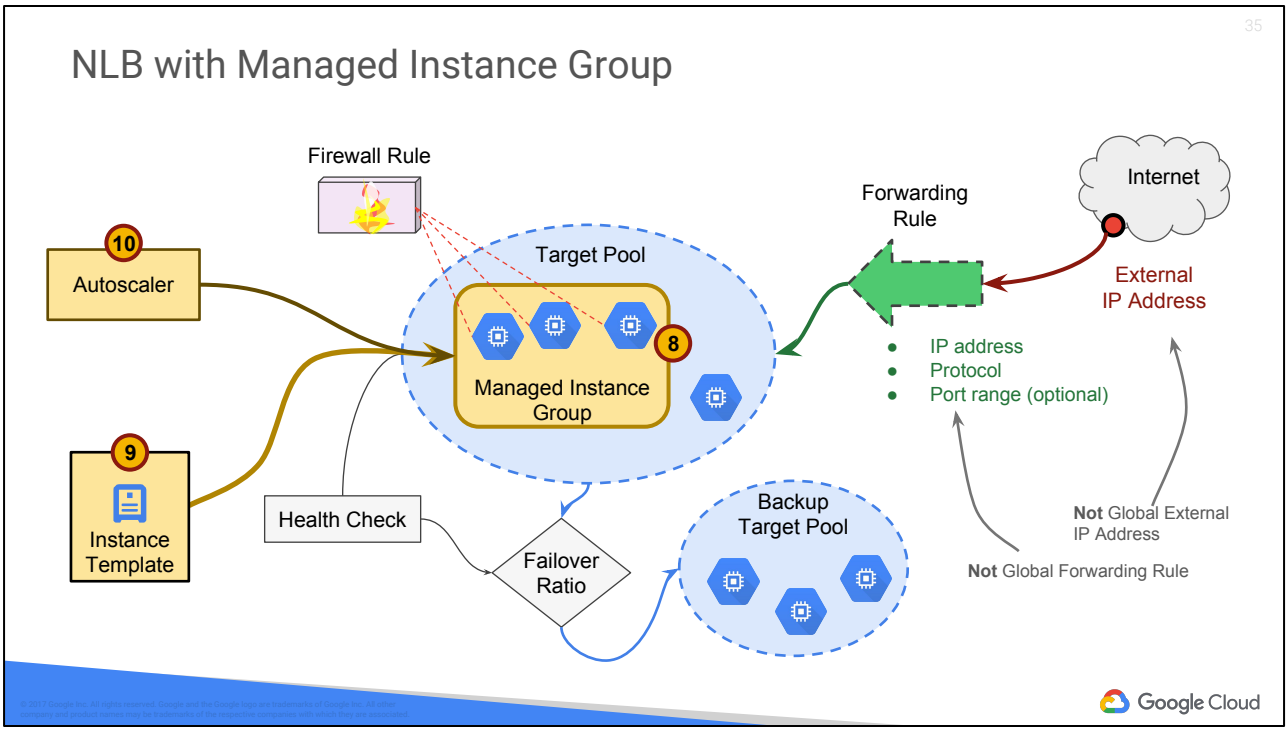


(1) Network Load Balancing occurs within a region. It can cover zones with a region (for high availability) but not multiple regions.
(2) Network Load Balancing (NLB) begins with an External IP Address. You can't use NLB with a Global External IP Address.
(3) A Forwarding Rule is associated with the IP. Traffic arriving on the IP that matches a particular Protocol is directed by the Forwarding Rule. Optionally, you can specify a port range, otherwise all traffic of the specified protocol is forwarded. Forwarding Rules are regional. You can't use a Global Forwarding Rule with NLB.
(4) The Forwarding Rule sends the traffic to a Target Pool. The Target Pool contains instances and it uses a Load Distribution algorithm to distribute traffic to each of the instances in the pool. When the Forwarding Rule is created you need to reference the Target Pool, so that means the Target Pool must be created before the Forwarding Rule. The Target Pool can contain instances from multiple zones in the same region. If a zone is lost, the instances from other zones can continue, providing high availability.
(5) Of course, the instances can't receive traffic without a Firewall Rule to allow the particular protocol to be delivered to them.
(6) A Target Pool has a single Health Check that is used with all instances in the pool. A Health Check is an algorithm that determines whether an instance is "healthy" or not. For example, if the pool contained web servers, the check might be trying to read a particular web page. An error would indicate that the particular server wasn't healthy. An instance that is going out of service might intentionally do something to force the health check to fail, such as removing the web page used by the health check. This causes the target pool to stop sending new traffic to the instance so that it

can "drain" its sessions (complete work in progress) prior to going out of service. So the health check is used for traffic flow control within the pool.
(7) The number of healthy instances over the total number in the pool creates a ratio between 0 and 1. A Failover Ratio can be set such that if there are not enough healthy instances, traffic will be sent to instances in an (optional) backup pool. This service is only one layer deep. That is, you can create a backup pool for a Target Pool, but you can't create a backup for the backup. Note that the Backup Target Pool must be in the same Region as the Target Pool.

Note that instances are added to a Target Pool after they are operational. The Target Pool knows nothing about the origin of the instances.
It also cannot start new instances or shut down instances.

https://cloud.google.com/compute/docs/load-balancing/network/target-pools



(7) When a Managed Instance Group is used with Network Load Balancing, the group adds or removes instances from the Target Pool automatically. You don't tell the Target Pool to use the Managed Instance Group, you tell the Managed Instance Group to use the Target Pool with this command: `gcloud compute instance-groups managed set-target-pools`

(8) The Managed Instance Group creates identical instances from an Instance Template. And it affords other commands for dealing with the entire group of instances so you don't have to perform the same action on each instance individually. For this reason, Managed Instance Groups are useful for software maintenance, because an update can be rolled out to all the instances in a group.

(9) Another benefit of a Managed Instance Group is that it can be used with an Autoscaler. The Autoscaler tells the Managed Instance Group when to create or shutdown instances. There are many modes, settings, and options for configuring the Autoscaler scaling policy. For example, the policy can be set to scale on CPU load or on Stackdriver metrics.

The concept of a Backup Pool is in conflict with the concept of Autoscaling. Therefore, Autoscaling cannot be configured for a Target Pool if it has a Backup Pool defined.

Autoscaling is discussed in detail in a subsequent module.

Forwarding Rules

- Forwarding rules consist of...
  - Name
  - Region
  - IP Address (regional, not global)
  - IP Protocol (TCP, UDP; AH, ESP, ICMP, SCTP)
  - Ports
  - Target-pool or target-instance
- You can manage forwarding rules using...
  - The Google Cloud Platform Console
  - The `gcloud` utility
  - The Compute Engine REST API

Google Cloud

Forwarding rules use for load balancing, thus pointing at a target pool, can only forward TCP & UDP packets. Forwarding rules for AH/ESP/ICMP/SCTP must forward to a target-instance as part of a port-forwarding setup.

For more information on forwarding rules, see https://cloud.google.com/compute/docs/load-balancing/network/forwarding-rules

A Target Pool resource defines a group of instances that should receive incoming traffic from forwarding rules

- Target pools can only be used with forwarding rules that handle:
  - TCP traffic
  - UDP traffic
- You must create a target pool before you can use with with a forwarding rule.
- Each project can have up to 50 target pools.
- A target pool can have only one health check.
  - Network load balancing only supports httpHealthChecks.
- Instances can be in different zones but must be in the same region; add to pool at creation or use Instance Group

Google Cloud

A Target Pool resource defines a group of instances that should receive incoming traffic from forwarding rules. When a forwarding rule directs traffic to a target pool, Google Compute Engine picks an instance from these target pools based on a hash of the source IP and port and the destination IP and port.

Target pools can only be used with forwarding rules that handle TCP and UDP traffic. For all other protocols, you must create a target instance. You must create a target pool before you can use it with a forwarding rule. Each project can have up to 50 target pools. A target pool can have only one health check. Network load balancing only supports httpHealthChecks. Network load balancing supports Compute Engine Autoscaler, which allows users to perform autoscaling on the instance groups in a target pool based on CPU utilization or custom Stackdriver Monitoring metrics.

For more information, see:
httpHealthChecks:
https://cloud.google.com/compute/docs/reference/latest/httpHealthChecks
Compute Engine Autoscaler: https://cloud.google.com/compute/docs/autoscaler
Stackdriver Monitoring: https://cloud.google.com/monitoring/docs

`sessionAffinity` influences load distribution

- The hash method selects a backend based on a subset of:
  - Source/Destination IP
  - Source/Destination Port
  - Layer 4 Protocol (TCP, UDP)
- Possible hashes:
  - NONE
  - CLIENT_IP_PROT
  - CLIENT_IP

Caution: If a large portion of your clients are behind a proxy server, you should not use CLIENT_IP_PROTO or CLIENT_IP. Using them would end up sending all the traffic from those clients to the same instance.

NONE (i.e., no hash specified) (default)
5-tuple hashing, which uses the source and destination IPs, source and destination ports, and protocol. Each new connection can end up on any instance, but all traffic for a given connection will stay on the same instance if the instance stays healthy.

CLIENT_IP_PROTO
3-tuple hashing, which uses the source and destination IPs and the protocol. All connections from a client will end up on the same instance as long as they use the same protocol and the instance stays healthy.

CLIENT_IP
2-tuple hashing, which uses the source and destination IPs. All connections from a client will end up on the same instance regardless of protocol as long as the instance stays healthy.

5-tuple hashing provides a good distribution of traffic across many virtual machines. However, a second session from the same client may arrive on a different instance because the source port may change. If you want all sessions from the same client to reach the same backend, as long as the backend stays healthy, you can specify CLIENT_IP_PROTO or CLIENT_IP options.

In general, if you select a 3-tuple or 2-tuple method, it will provide for better session affinity than the default 5-tuple method, but the overall traffic may not be as evenly distributed.

Fragmented UDP packets: If you are load balancing UDP traffic that is likely to be fragmented, set session affinity to CLIENT_IP_PROTO or CLIENT_IP. Do not use NONE (5-tuple hashing). This is because UDP fragments other than the first one do not carry the port number, and the load balancer may drop the fragments without the port.

Caution: If a large portion of your clients are behind a proxy server, you should not use CLIENT_IP_PROTO or CLIENT_IP. Using them would end up sending all the traffic from those clients to the same instance.

For more information, see:
Target pools:
https://cloud.google.com/compute/docs/load-balancing/network/target-pools
Load balancing and fragmented UDP packets:
https://cloud.google.com/compute/docs/load-balancing/network/#load_balancing_and_fragmented_udp_packets

Agenda

- Managed Instance Groups
- HTTP(S) load balancing
- Cross-Region and Content-Based Load Balancing
- SSL Proxy/TCP Proxy load balancing
- Network load balancing
- **Internal load balancing**
- Load Balancing Best Practices
- Lab
- Quiz

Internal Load Balancing

Internal load balancing allows you to:

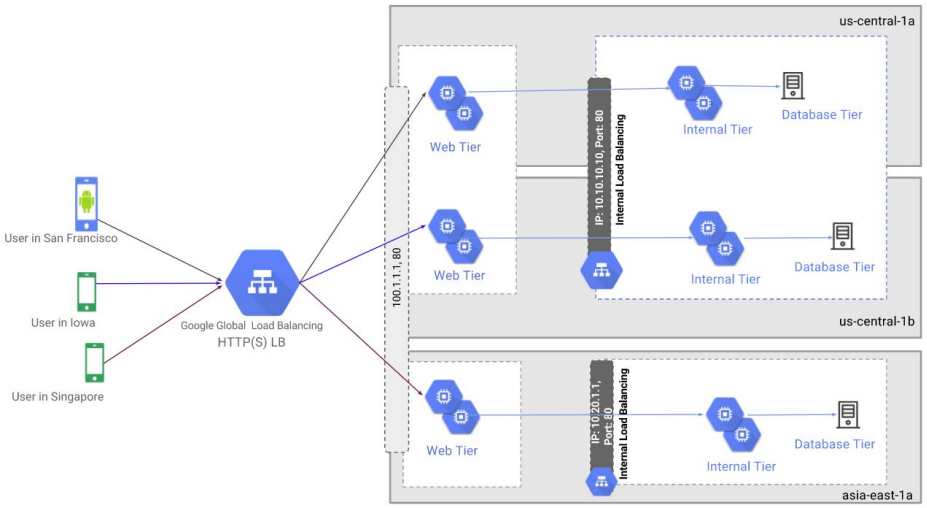
- Load balance TCP/UDP traffic using a private frontend IP.
- Load balance across instances in a region.
- Configure health checking for you backends.
- Get the benefits of a fully managed load balancing service that scales as you need to handle client traffic.

Google Cloud

Internal load balancing enables you to run and scale your services behind a private load balancing IP address which is accessible only to instances internal to your Virtual Private Cloud (VPC).

For more information, see:
Internal Load Balancing:
https://cloud.google.com/compute/docs/load-balancing/internal/
Internal Load Balancing in 5 minutes:
https://cloud.google.com/files/internal-load-balancing-tutorial-slides.pdf

HTTP(S) and Internal Load Balancing Example



Google Cloud

The example shows a 3-tier web application with HTTP(S) load balancing and Internal load balancing.

Agenda

- Managed Instance Groups
- HTTP(S) load balancing
- Cross-region and content-based load balancing
- SSL proxy/TCP proxy load balancing
- Network load balancing
- Internal load balancing
- **Load balancing best practices**
- Lab
- Quiz

Google Cloud

Regional MIG Best Practices

- Spreads and balances across three zones in a region
  - Zones chosen at random
  - If a zone becomes unhealthy, healthy zones take over
  - Rebalances when zone returns
- Overprovisioning
  - Provision at 100% for 2/3rd service during an outage
  - Provision at 150% for full service during an outage
- Testing
  - Tag VMs with "failure_zone"
  - Run script with `sudo shutdown` to simulate outage

Google Cloud

https://cloud.google.com/compute/docs/instance-groups/distributing-instances-with-regional-instance-groups

Keep the following guidance in mind when creating instance groups for use with load balancing

- Do not put a VM in more than one instance group.
- Do not delete an instance group if it is being used by a backend.
- Avoid adding the same instance group to two different backends.
- All instances in a managed or unmanaged instance group must be in the same VPC network.
- If using a MIG with autoscaling, don't use `maxRate` balancing mode in the backend service.
- Don't make an autoscaled MIG the target of two different load balancers.
- Max size of a MIG should be <= size of the subnet.

Google Cloud

Because Compute Engine offers a great deal of flexibility in how you configure load balancing, it is possible to create configurations that do not behave well. Please keep the following restrictions and guidance in mind when creating instance groups for use with load balancing.

- Do not put a virtual machine instance in more than one instance group.
- Do not delete an instance group if it is being used by a backend.
- Your configuration will be simpler if you do not add the same instance group to two different backends. If you do add the same instance group to two backends:
  - Both backends must use the same balancing mode, either UTILIZATION or RATE.
  - You can use maxRatePerInstance and maxRatePerGroup together. It is acceptable to set one backend to use maxRatePerInstance and the other to maxRatePerGroup.
  - If your instance group serves two or more ports for several backends respectively, you have to specify different port names in the instance group.
- All instances in a managed or unmanaged instance group must be in the same VPC network and, if applicable, the same subnet.
- If you are using a managed instance group with autoscaling, do not use the maxRate balancing mode in the backend service. You may use either the

- maxUtilization or maxRatePerInstance mode.
- Do not make an autoscaled managed instance group the target of two different load balancers.
- When resizing a managed instance group, the maximum size of the group should be smaller than or equal to the size of subnet.

For more information, see:
https://cloud.google.com/compute/docs/load-balancing/http/backend-service#restrictions_and_guidance

Securing Load Balanced Servers

- To access load-balanced servers, a firewall rule must be created for the appropriate networks
  - Allows HTTP(S) traffic to Compute Engine instances
- Best practice is to create firewall rules allowing traffic only from GCP Load Balancer networks
  - `130.211.0.0/22`
- Instances can be further secured by having no external IP addresses
  - Load balancing uses internal IP addresses
  - Leave a bastion host by which you can manage these instances

Agenda

- Managed Instance Groups
- HTTP(S) load balancing
- Cross-region and content-based load balancing
- SSL proxy/TCP proxy load balancing
- Network load balancing
- Internal load balancing
- Load balancing best practices
- **Lab**
- Quiz

Google Cloud

Lab: Virtual Machine Automation and Load Balancing

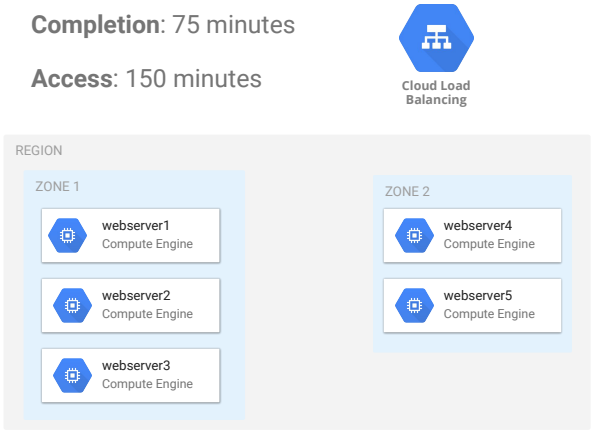
Objectives

In this lab, you learn how to perform the following tasks:

- Create a pool of VMs
- Configure an external load balancer to use the pool
- Place a load on the service and stop a VM to simulate an outage
- Launch two more VMs in a secondary zone
- Configure an internal load balancer
- Test the internal load balancer

Completion: 75 minutes

Access: 150 minutes



Lab Review

In this lab you:

- Created a pool of web servers
- Directed traffic to the web servers through an external network load balancer
- Tested for high availability by shutting down one of the servers
- Launched two additional web servers in a secondary zone
- Configured an internal load balancer for work distribution and availability

Agenda

- Managed Instance Groups
- HTTP(S) load balancing
- Cross-region and content-based load balancing
- SSL proxy/TCP proxy load balancing
- Network load balancing
- Internal load balancing
- Load balancing best practices
- Lab
- **Quiz**

Google Cloud

Quiz

What are the three categories of GCP load balancing as described in the course materials?

1. Local Area load balancing, HTTP(S) load balancing, and auto scaling load balancing
2. Network load balancing, local area load balancing, and unmanaged load balancing
3. HTTP(S) load balancing, SSL proxy load balancing, and HAProxy load balancing
4. Global external load balancing, regional external load balancing, and regional internal load balancing

Quiz

What are the three categories of GCP load balancing as described in the course materials?

1. Local Area load balancing, HTTP(S) load balancing, and auto scaling load balancing
2. Network load balancing, local area load balancing, and unmanaged load balancing
3. HTTP(S) load balancing, SSL proxy load balancing, and HAProxy load balancing
4. Global external load balancing, regional external load balancing, and regional internal load balancing *

Explanation:
The correct answer is global external load balancing, regional external load balancing, and regional internal load balancing.

Quiz

Which form of load balancing distributes traffic among a pool of instances within a region?

1. SSL Proxy load balancing
2. HTTP(S) load balancing
3. Network load balancing
4. TCP Proxy load balancing

Quiz

Which form of load balancing distributes traffic among a pool of instances within a region?

1. SSL Proxy load balancing
2. HTTP(S) load balancing
3. Network load balancing *
4. TCP Proxy load balancing

Explanation:
Network load balancing distributes traffic among a pool of instances within a region. Network load balancing can balance any kind of TCP/UDP traffic.

Quiz

Which form of load balancing uses path rules to send traffic to backend services based on type?

1. Any Load Balancer that uses a Managed Instance Group
2. UDP Network Load Balancer
3. Content-based Load Balancing
4. SSL Proxy

Quiz

Which form of load balancing uses path rules to send traffic to backend services based on type?

1. Any Load Balancer that uses a Managed Instance Group
2. UDP Network Load Balancer
3. Content-based Load Balancing *
4. SSL Proxy

Explanation:
Content-based Load Balancing routes traffic to servers that specialize in different kinds of content. This is the opposite of a Managed Instance Group that uses server templates to ensure that all VMs are the same.

More...

Load Balancing

- https://cloud.google.com/compute/docs/load-balancing/http/

Managed Instance Groups

- https://cloud.google.com/compute/docs/instance-groups/

Forwarding Rules

- https://cloud.google.com/compute/docs/load-balancing/network/forwarding-rules

Google Cloud

More to learn on this subject. Here are some suggestions and links.

More...

Target Pools

- https://cloud.google.com/compute/docs/reference/latest/targetPools

Global Forwarding Rules

- https://cloud.google.com/compute/docs/load-balancing/http/global-forwarding-rules

Target Proxies

- https://cloud.google.com/compute/docs/load-balancing/http/target-proxies