**Overview**

Google Cloud HTTP(S) load balancing is deployed at the edge of Google's network in Google points of presence (POP) around the world. User traffic directed to an HTTP(S) load balancer enters the POP closest to the user and is then load balanced over Google's global network to the closest backend that has sufficient capacity available.

Cloud Armor is Google's distributed denial of service and web application firewall (WAF) detection system. Cloud Armor is tightly coupled with the Google Cloud HTTP Load Balancer and safeguards applications of Google Cloud customers from attacks from the internet.

[reCAPTCHA Enterprise](https://cloud.google.com/recaptcha-enterprise/docs) is a service that builds on the reCAPTCHA API and protects your site from spam and abuse using advanced risk analysis techniques to tell humans and bots apart. Cloud Armor Bot Management provides an end-to-end solution integrating reCAPTCHA Enterprise bot detection and scoring with enforcement by Cloud Armor at the edge of the network to protect downstream applications.

In this lab, you configure an HTTP Load Balancer with a backend, as shown in the diagram below. You set up a reCAPTCHA session token site key and embed it in your website. You also set up redirection to reCAPTCHA Enterprise manual challenge. You then configure a Cloud Armor bot management policy to see how bot detection protects your application from malicious bot traffic.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

What you'll learn

In this lab, you learn how to:

* Set up a HTTP Load Balancer with appropriate health checks
* Create a reCAPTCHA WAF challenge-page site key and associated it with Cloud Armor security policy
* Create a reCAPTCHA session token site key and install it on your web pages
* Create a Cloud Armor bot management policy
* Validate that the bot management policy is handling traffic based on the rules configured

**Setup and requirements**

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

To complete this lab, you need:

* Access to a standard internet browser (Chrome browser recommended).

**Note:** Use an Incognito or private browser window to run this lab. This prevents any conflicts between your personal account and the Student account, which may cause extra charges incurred to your personal account.

* Time to complete the lab---remember, once you start, you cannot pause a lab.

**Note:** If you already have your own personal Google Cloud account or project, do not use it for this lab to avoid extra charges to your account.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is the **Lab Details** panel with the following:
   * The **Open Google Console** button
   * Time remaining
   * The temporary credentials that you must use for this lab
   * Other information, if needed, to step through this lab
2. Click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.

***Tip:*** Arrange the tabs in separate windows, side-by-side.

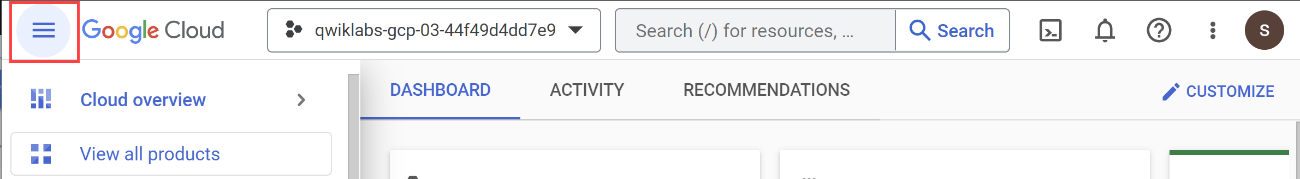
**Note:**If you see the **Choose an account** dialog, click **Use Another Account**.

1. If necessary, copy the **Username** from the **Lab Details** panel and paste it into the **Sign in** dialog. Click **Next**.
2. Copy the **Password** from the **Lab Details** panel and paste it into the **Welcome** dialog. Click **Next**.

**Important:**You must use the credentials from the left panel. Do not use your Google Cloud Skills Boost credentials.**Note:**Using your own Google Cloud account for this lab may incur extra charges.

1. Click through the subsequent pages:
   * Accept the terms and conditions.
   * Do not add recovery options or two-factor authentication (because this is a temporary account).
   * Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

**Note:** You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left. 

Before you begin

* In Cloud Shell, set up your project ID:

export PROJECT\_ID=$(gcloud config get-value project)

echo $PROJECT\_ID

gcloud config set project $PROJECT\_ID

Copied!

content\_copy

Enable APIs

* Enable all necessary services:

gcloud services enable compute.googleapis.com

gcloud services enable logging.googleapis.com

gcloud services enable monitoring.googleapis.com

gcloud services enable recaptchaenterprise.googleapis.com

Copied!

content\_copy

You're now set up to begin the first task.

**Task 1. Configure firewall rules to allow HTTP and SSH traffic to backends**

For this lab, you use the ***default*** VPC network created in your project when the lab spun up.

In this section, you configure firewall rules to allow HTTP traffic to the backends from the Google Cloud health checks and the Load Balancer. You then configure a firewall rule to allow SSH into the instances.

Create a firewall rule to allow HTTP traffic to the backends.

Health checks determine which instances of a load balancer can receive new connections. For HTTP load balancing, the health check probes to your load balanced instances come from addresses in the ranges **130.211.0.0/22 and 35.191.0.0/16**. Your VPC firewall rules must allow these connections. Also, the load balancers talk to the backend on the same IP range.

To create a firewall rule to allow HTTP traffic to the backends:

1. In the console, navigate to **Navigation menu (Navigation menu icon) > VPC network > Firewall**.
2. Notice the existing **ICMP**, **internal**, **RDP**, and **SSH** firewall rules. Each Google Cloud project starts with the default network and these firewall rules.
3. Click **Create Firewall Rule**.
4. Set the following values, leave all other values at their defaults:

|  |  |
| --- | --- |
| **Property** | **Value (type value or select option as specified)** |
| **Name** | default-allow-health-check |
| **Network** | default |
| **Targets** | Specified target tags |
| **Target tags** | allow-health-check |
| **Source filter** | IP Ranges |
| **Source IP ranges** | 130.211.0.0/22, 35.191.0.0/16 |
| **Protocols and ports** | Specified protocols and ports, and then check tcp. Type 80 for the port number |

**Note:**Be sure to enter the two **Source IP ranges** one-by-one and press SPACE in between them.

1. Click **Create**.

Alternatively, in the gCloud command line, use the following command:

gcloud compute firewall-rules create default-allow-health-check --direction=INGRESS --priority=1000 --network=default --action=ALLOW --rules=tcp:80 --source-ranges=130.211.0.0/22,35.191.0.0/16 --target-tags=allow-health-check

Copied!

content\_copy

1. Similarly, create a firewall rule to allow SSH-ing into the instances:

gcloud compute firewall-rules create allow-ssh --direction=INGRESS --priority=1000 --network=default --action=ALLOW --rules=tcp:22 --source-ranges=0.0.0.0/0 --target-tags=allow-health-check

Copied!

content\_copy

Click **Check my progress** to verify the objective.

Successfully configured the firewall rules. Assessment completed!

Configure firewall rules to allow HTTP and SSH traffic to backends

Check my progress

*Successfully configured the firewall rules. Assessment completed!*

**Task 2. Configure instance templates and create managed instance groups**

A managed instance group uses an instance template to create a group of identical instances. Use these to create the backend of the HTTP Load Balancer.

Configure the instance templates

An instance template is a resource that you use to create VM instances and managed instance groups. Instance templates define the machine type, boot disk image, subnet, labels, and other instance properties.

To create an instance template:

1. In the console, navigate to **Navigation menu (Navigation menu icon) > Compute Engine > Instance templates**, and then click **Create instance template**.
2. For **Name**, type **lb-backend-template**.
3. For Series, select **N1**.
4. Click **Networking, Disks, Security, Management, Sole-Tenancy** under **Advanced options**.
5. Go to the **Management** section and insert the following script into the **Startup script** field:

#! /bin/bash

sudo apt-get update

sudo apt-get install apache2 -y

sudo a2ensite default-ssl

sudo a2enmod ssl

sudo su

vm\_hostname="$(curl -H "Metadata-Flavor:Google" \

http://metadata.google.internal/computeMetadata/v1/instance/name)"

echo "Page served from: $vm\_hostname" | \

tee /var/www/html/index.html

Copied!

content\_copy

1. Click on the **Networking** tab and add the network tag: **allow-health-check**
2. Set the following values and leave all other values at their defaults:

|  |  |
| --- | --- |
| **Property** | **Value (type value or select option as specified)** |
| **Network (Under Network Interfaces)** | default |
| **Subnet (Under Network Interfaces)** | default ( us-east1 ) |
| **Network tags** | allow-health-check |

**Note:**The network tag **allow-health-check** ensures that the HTTP Health Check and SSH firewall rules apply to these instances.

1. Click **Create**.

Wait for the instance template to be created.

Create the managed instance group

To Create the managed instance group:

1. Still in **Compute Engine** page, click **Instance groups** in the left pane.
2. Click **Create instance group**.
3. Select **New managed instance group (stateless)**.
4. Set the following values, leave all other values at their defaults:

|  |  |
| --- | --- |
| **Property** | **Value (type value or select option as specified)** |
| **Name** | lb-backend-example |
| **Location** | Single zone |
| **Region** | us-east1 |
| **Zone** | us-east1-d |
| **Instance template** | lb-backend-template |
| **Autoscaling** | Set **Autoscaling mode** to **Off: do not autoscale** |
| **Number of instances** | 1 |

1. Click **Create**.

Add a named port to the instance group

* For your instance group, in Cloud Shell, define an HTTP service and map a port name to the relevant port:

gcloud compute instance-groups set-named-ports lb-backend-example \

--named-ports http:80 \

--zone us-east1-d

Copied!

content\_copy

The load balancing service forwards traffic to the named port.

Click **Check my progress** to verify the objective.

Successfully created instance template and managed instance group. Assessment completed!

Configure instance templates and create managed instance groups

Check my progress

*Successfully created instance template and managed instance group. Assessment completed!*

**Task 3. Configure the HTTP Load Balancer**

In this section, you configure the HTTP Load Balancer to send traffic to your backend **lb-backend-example**.

Start the configuration

1. In the console, click **Navigation menu** (Navigation menu icon) > click **Network Services** > **Load balancing**, and then click **Create load balancer**.
2. Under **HTTP(S) Load Balancing**, click **Start configuration**.
3. Select **From Internet to my VMs**, **Global HTTP(S) Load Balancer (classic)** and click **Continue**.
4. Set the **Name** to http-lb.

Configure the frontend

1. Click **Frontend configuration**.
2. Specify the following, leaving all other values at their defaults:

|  |  |
| --- | --- |
| **Property** | **Value (type value or select option as specified)** |
| **Protocol** | HTTP |
| **IP version** | IPv4 |
| **IP address** | Ephemeral |
| **Port** | 80 |

1. Click **Done**.

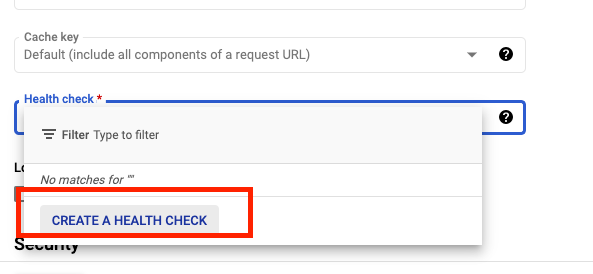
Configure the backend

Backend services direct incoming traffic to one or more attached backends. Each backend is composed of an instance group and additional serving capacity metadata.

1. Click on **Backend configuration**.
2. For **Backend services & backend buckets**, click **Create a backend service**.
3. Set the following values, leave all other values at their defaults:

|  |  |
| --- | --- |
| **Property** | **Value (select option as specified)** |
| **Name** | http-backend |
| **Protocol** | HTTP |
| **Named Port** | http |
| **Instance group** | lb-backend-example |
| **Port numbers** | 80 |

1. Click **Done**.
2. For **Cloud CDN**, set the **Cache mode** to **Use origin settings based on Cache-Control headers**.
3. For **Health Check**, select **Create a health check**.



1. Set the following values, leave all other values at their defaults:

|  |  |
| --- | --- |
| **Property** | **Value (select option as specified)** |
| **Name** | http-health-check |
| **Protocol** | TCP |
| **Port** | 80 |

**Note:**Health checks determine which instances receive new connections.

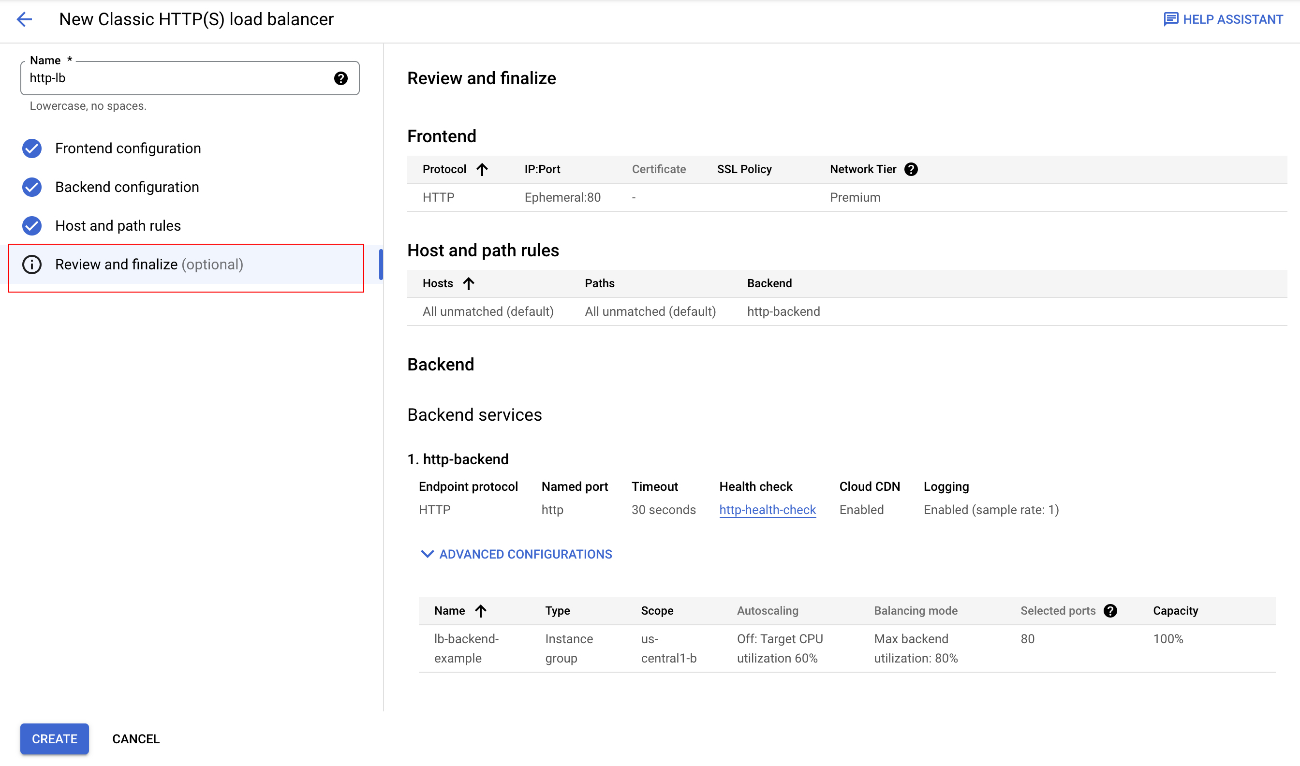
This HTTP health check polls instances every 5 seconds, waits up to 5 seconds for a response and treats 2 successful or 2 failed attempts as healthy or unhealthy, respectively.

1. Click **Save**.
2. Check the **Enable Logging box**.
3. Set the **Sample rate** to 1.
4. Click **Create** to create the backend service.

Review and create the HTTP Load Balancer

Host and path rules determine how your traffic is directed. For example, you could direct video traffic to one backend and static traffic to another backend. However, you are not configuring the Host and path rules in this lab.

1. Click on **Review and finalize**.



1. Review the **Backend services** and **Frontend**.
2. Click on **Create**.

Wait for the load balancer to be created.

1. Click on the name of the load balancer (**http-lb**).
2. Note the IPv4 address of the load balancer for the next task. In this lab, refer to it as [LB\_IP\_v4].

Click **Check my progress** to verify the objective.

Successfully created HTTP Load balancer. Assessment completed!

Configure the HTTP Load Balancer

Check my progress

*Successfully created HTTP Load balancer. Assessment completed!*

Test the HTTP Load Balancer

Now that you created the HTTP Load Balancer for your backends, verify that traffic is forwarded to the backend service.

* To test IPv4 access to the HTTP Load Balancer, open a new tab in your browser and navigate to **http://[LB\_IP\_v4]**. Be sure to replace [LB\_IP\_v4] with the IPv4 address of the load balancer.

**Note:**It might take up to 5 minutes to access the HTTP Load Balancer. In the meantime, you might get a 404 or 502 error. Keep trying until you see the page load.

**Task 5. Create and deploy reCAPTCHA session token and challenge-page site key**

reCAPTCHA Enterprise for WAF and Google Cloud Armor integration offers the following features: [reCAPTCHA challenge page](https://cloud.google.com/recaptcha-enterprise/docs/usecase#redirect), [reCAPTCHA action-tokens](https://cloud.google.com/recaptcha-enterprise/docs/usecase#action-token), and [reCAPTCHA session-tokens](https://cloud.google.com/recaptcha-enterprise/docs/usecase#session-token). In this task, you implement the reCATCHA session token site key and reCAPTCHA WAF challenge-page site.

Create reCAPTCHA session token and WAF challenge-page site key

Before you create the session token site key and challenge page site key, double-check that you've enabled the reCAPTCHA Enterprise API as indicated in the previous *Enable API* section.

The reCAPTCHA JavaScript sets a reCAPTCHA session-token as a cookie on the end-user's browser after the assessment. The end-user's browser attaches the cookie and refreshes the cookie as long as the reCAPTCHA JavaScript remains active.

1. Create the reCAPTCHA session token site key and enable the WAF feature for the key:

gcloud recaptcha keys create --display-name=test-key-name \

--web --allow-all-domains --integration-type=score --testing-score=0.5 \

--waf-feature=session-token --waf-service=ca

Copied!

content\_copy

The output of the above command gives you the session token site key you created. Record it to use later in this task.

You also set the WAF service to Cloud Armor to enable the Cloud Armor integration.

**Note:**You are using the integration type **score** which is leveraged in the Cloud Armor policy. You can alternatively use **checkbox** and **invisible**.

You also set a **testing score** when you create the key to validate that the bot management policies created with Cloud Armor are working as intended. Replicating bot traffic is not easy, so this is a good way to test the feature.

1. Create the reCAPTCHA WAF challenge-page site key and enable the WAF feature for the key. You can use the reCAPTCHA challenge page feature to redirect incoming requests to reCAPTCHA Enterprise to determine whether each request is potentially fraudulent or legitimate. Later, you associate this key with the Cloud Armor security policy to enable the manual challenge. This lab refers to this key as CHALLENGE-PAGE-KEY in the later steps.

gcloud recaptcha keys create --display-name=challenge-page-key \

--web --allow-all-domains --integration-type=INVISIBLE \

--waf-feature=challenge-page --waf-service=ca

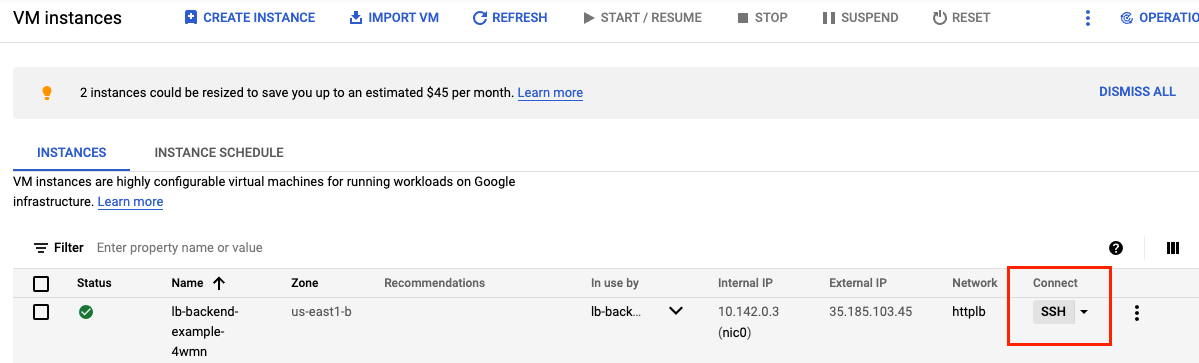
Copied!

content\_copy

1. Navigate to **Navigation menu (Navigation menu icon) > Security > reCAPTCHA Enterprise**. You should see the keys you created in the **Enterprise Keys** list:

Implement reCAPTCHA session token site key

1. Navigate to **Navigation menu (Navigation menu icon) > Compute Engine > VM Instances**. Locate the VM in your instance group and SSH to it.



1. Go to the webserver root directory and change user to root:

cd /var/www/html/

sudo su

Copied!

content\_copy

1. Update the landing index.html page and embed the reCAPTCHA session token site key. The session token site key (that you recorded earlier) is set in the head section of your landing page as below:

src="https://www.google.com/recaptcha/enterprise.js?render=<SESSION\_TOKEN\_SITE\_KEY>&waf=session" async defer>

Remember to replace <SESSION\_TOKEN\_SITE\_KEY> with the site token before you run the following command:

echo '<!doctype html><html><head><title>ReCAPTCHA Session Token</title><script src="https://www.google.com/recaptcha/enterprise.js?render=<SESSION\_TOKEN\_SITE\_KEY>&waf=session" async defer></script></head><body><h1>Main Page</h1><p><a href="/good-score.html">Visit allowed link</a></p><p><a href="/bad-score.html">Visit blocked link</a></p><p><a href="/median-score.html">Visit redirect link</a></p></body></html>' > index.html

Copied!

content\_copy

1. Create three other sample pages to test out the bot management policies:

* **good-score.html**

echo '<!DOCTYPE html><html><head><meta http-equiv="Content-Type" content="text/html; charset=windows-1252"></head><body><h1>Congrats! You have a good score!!</h1></body></html>' > good-score.html

Copied!

content\_copy

* **bad-score.html**

echo '<!DOCTYPE html><html><head><meta http-equiv="Content-Type" content="text/html; charset=windows-1252"></head><body><h1>Sorry, You have a bad score!</h1></body></html>' > bad-score.html

Copied!

content\_copy

* **median-score.html**

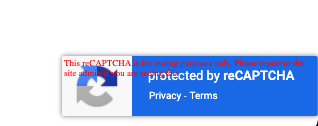
echo '<!DOCTYPE html><html><head><meta http-equiv="Content-Type" content="text/html; charset=windows-1252"></head><body><h1>You have a median score that we need a second verification.</h1></body></html>' > median-score.html

Copied!

content\_copy

Validate that you are able to access all the webpages by opening them in your browser. Be sure to replace [LB\_IP\_v4] with the IPv4 address of the load balancer:

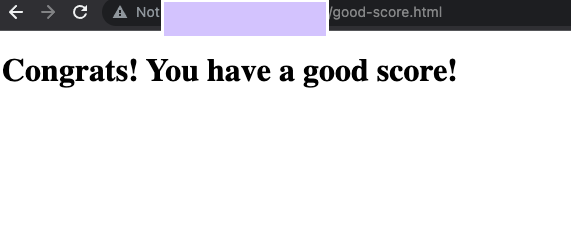
1. Open http://[LB\_IP\_v4]/index.html. You verify that the reCAPTCHA implementation is working when you see "protected by reCAPTCHA" at the bottom right corner of the page:



1. Click into each of the links.



1. Validate you are able to access all the pages.



Click **Check my progress** to verify the objective.

Successfully created Cloud Armor security policy rules for Bot Management. Assessment completed!

Deploy reCAPTCHA session token and challenge-page site key

Check my progress

*Successfully created Cloud Armor security policy rules for Bot Management. Assessment completed!*

**Task 6. Create Cloud Armor security policy rules for Bot Management**

In this section, you use Cloud Armor bot management rules to allow, deny and redirect requests based on the reCAPTCHA score. Remember that when you created the session token site key, you set a testing score of 0.5.

1. In Cloud Shell, create security policy via gcloud:

gcloud compute security-policies create recaptcha-policy \

--description "policy for bot management"

Copied!

content\_copy

1. To use reCAPTCHA Enterprise manual challenge to distinguish between human and automated clients, associate the reCAPTCHA WAF challenge site key (CHALLENGE-PAGE-KEY) you previously created for a manual challenge with the security policy. In the following script, remember to replace "CHALLENGE-PAGE-KEY" with the key you previously created:

gcloud compute security-policies update recaptcha-policy \

--recaptcha-redirect-site-key "CHALLENGE-PAGE-KEY"

Copied!

content\_copy

1. Add a bot management rule to allow traffic if the url path matches good-score.html and has a score greater than 0.4:

gcloud compute security-policies rules create 2000 \

--security-policy recaptcha-policy\

--expression "request.path.matches('good-score.html') && token.recaptcha\_session.score > 0.4"\

--action allow

Copied!

content\_copy

1. Add a bot management rule to deny traffic if the url path matches bad-score.html and has a score less than 0.6:

gcloud compute security-policies rules create 3000 \

--security-policy recaptcha-policy\

--expression "request.path.matches('bad-score.html') && token.recaptcha\_session.score < 0.6"\

--action "deny-403"

Copied!

content\_copy

1. Add a bot management rule to redirect traffic to Google reCAPTCHA if the url path matches median-score.html and has a score equal to 0.5:

gcloud compute security-policies rules create 1000 \

--security-policy recaptcha-policy\

--expression "request.path.matches('median-score.html') && token.recaptcha\_session.score == 0.5"\

--action redirect \

--redirect-type google-recaptcha

Copied!

content\_copy

1. Attach the security policy to the backend service http-backend:

gcloud compute backend-services update http-backend \

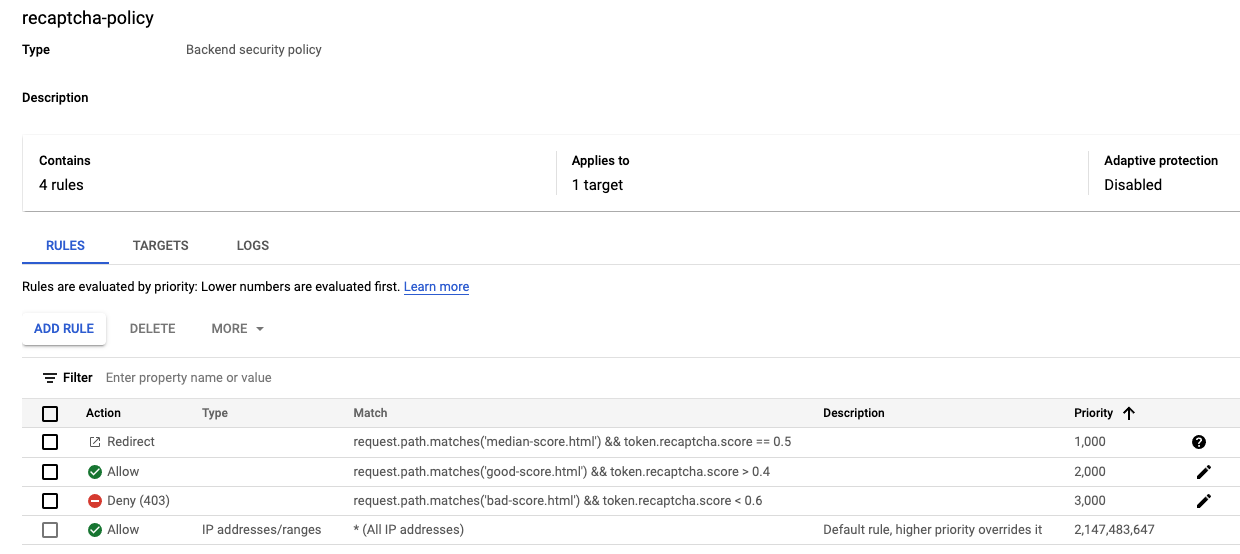
--security-policy recaptcha-policy --global

Copied!

content\_copy

1. In the console, navigate to **Navigation menu > Network Security > Cloud Armor**.
2. Click recaptcha-policy.

Your policy should resemble the following:



Click **Check my progress** to verify the objective.

Security policy rules successfully applied to the webpage. Assessment completed!

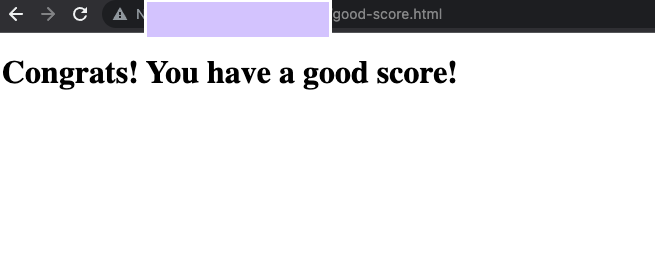
Create Cloud Armor security policy rules for Bot Management

Check my progress

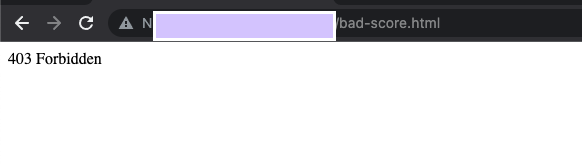
*Security policy rules successfully applied to the webpage. Assessment completed!*

**Task 7. Validate Bot Management with Cloud Armor**

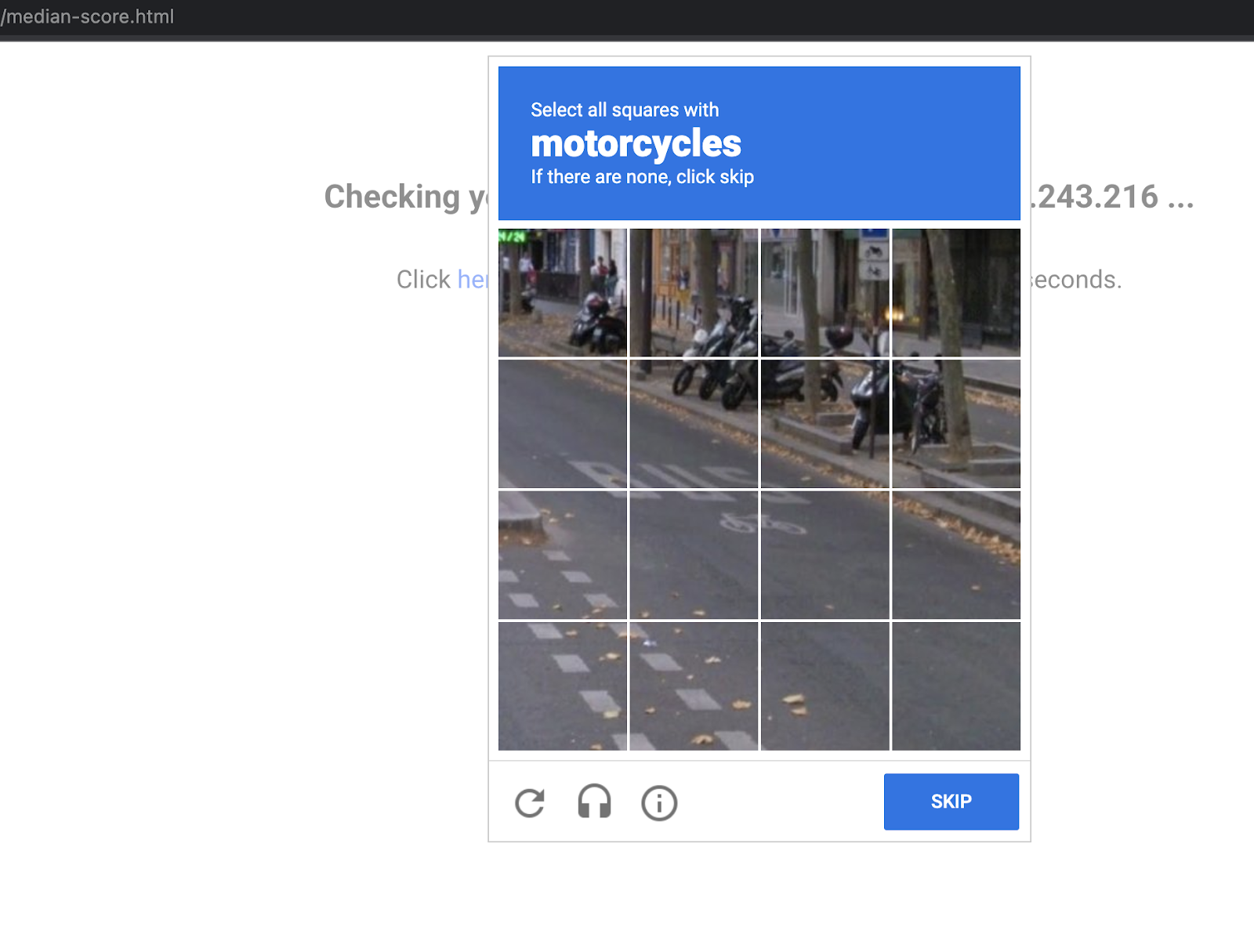
1. Open up a browser and enter the url http://[LB\_IP\_v4]/index.html. Navigate to **"Visit allow link"**.



1. Open a new window in Incognito mode to ensure you have a new session.
2. Enter the url http://[LB\_IP\_v4]/index.html and navigate to **"Visit blocked link"**. You should receive a **HTTP 403** error:



1. Open a new window in Incognito mode to ensure you have a new session.
2. Enter the url http://[LB\_IP\_v4]/index.html and navigate to **"Visit redirect link"**. You should see the redirection to Google reCAPTCHA and the manual challenge page as below:



Verify Cloud Armor logs

Explore the security policy logs to validate bot management worked as expected.

1. In the console, navigate to **Navigation menu > Network Security > Cloud Armor**.
2. Click **recaptcha-policy**.
3. Click **Logs**.
4. Click **View policy logs**.
5. Below is the MQL(monitoring query language) query, copy and paste into the query editor:

resource.type:(http\_load\_balancer) AND jsonPayload.enforcedSecurityPolicy.name:(recaptcha-policy)

Copied!

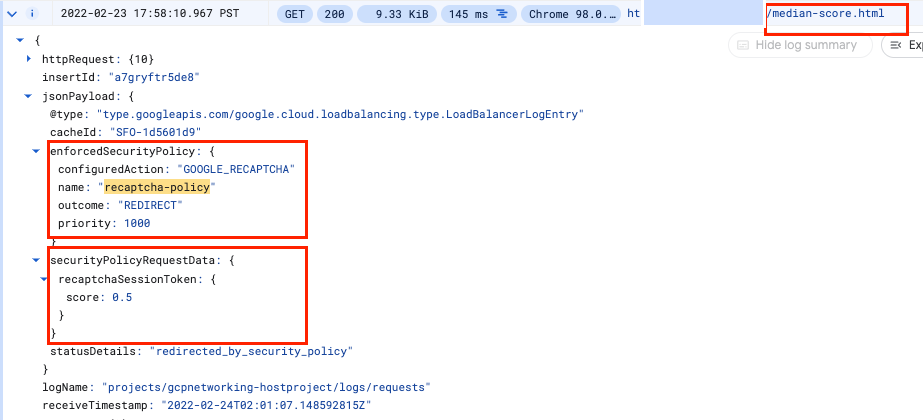
content\_copy

1. Now click **Run Query**.
2. Look for a log entry in Query results where the request is for http://[LB\_IP\_v4]/good-score.html. Expand jsonPayload.Expand enforcedSecurityPolicy.



1. Repeat the same for http://[LB\_IP\_v4]/bad-score.html and http://[LB\_IP\_v4]/median-score.html.





Notice that the configuredAction is set to **ALLOW, DENY or GOOGLE\_RECAPTCHA** with the name **recaptcha-policy**.

**Note:**Cloud Armor security policies create logs that can be explored to determine when traffic is denied and when it is allowed, along with the source of the traffic.

Click **Check my progress** to verify the objective.

We're having trouble checking your progress. Try again later.

Validate bot management with Cloud Armor

Check my progress

*We're having trouble checking your progress. Try again later.*

**Congratulations!**

You successfully implemented bot management with Cloud Armor. You configured an HTTP Load Balancer. Then, you created and implemented reCAPTCHA session token site key on a webpage. You also created a challenge-page site key. You set up Cloud Armor Bot management policy and validated how they handle requests based on the rules. You explored the security policy logs to identify why the traffic was allowed, blocked or redirected.