National College of Ireland

Configuration Manual

MSc Research Project
MSc in Cloud Computing

Abin Shaji
Student ID: 22190619

School of Computing
National College of Ireland

Supervisor: Prof Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Abin Shaji
Student ID:	22190619
Programme:	MSc in Cloud Computing **Year:** 2024
Module:	MSc research Project
Lecturer:	Prof Vikas Sahni
Submission Due Date:	12/08/2024
Project Title:	Securing Containerized Environments: Implementing Role-Based Access Control with Google Kubernetes Engine
Word Count:	865 **Page Count:** 8

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Abin Shaji

Date: 12/08/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Abin Shaji
Student ID: 22190619

1 Introduction

This document demonstrates the implementation of kubernetes dynamic rbac structure.

2 System configuration

Software Requirements

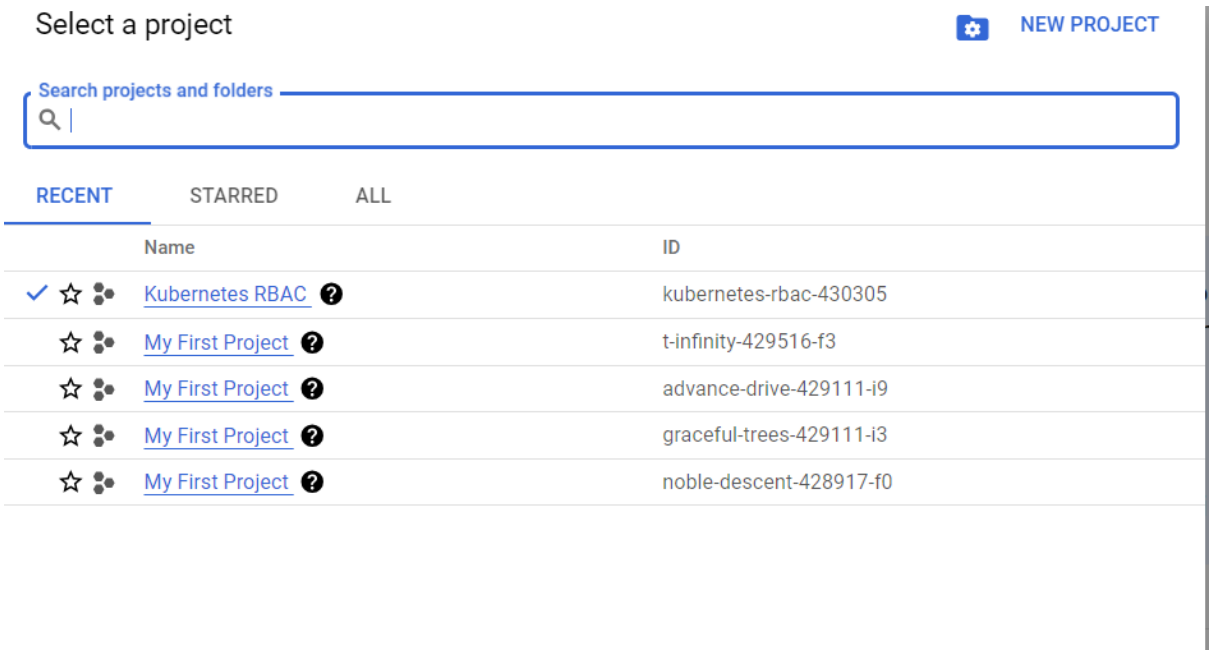
- **GCP Account:** Ensure you have a Google Cloud Platform account.
- **Billing Enabled:** Make sure billing is enabled for your GCP project.
- **gcloud CLI:** Install the gcloud CLI and authenticate it. (Google Cloud, n.d.)

3 Implementation

3.1 Step 1: Set Up GCP Project

1. **Create a GCP Project:** Create a new project.

gcloud projects create PROJECT_ID

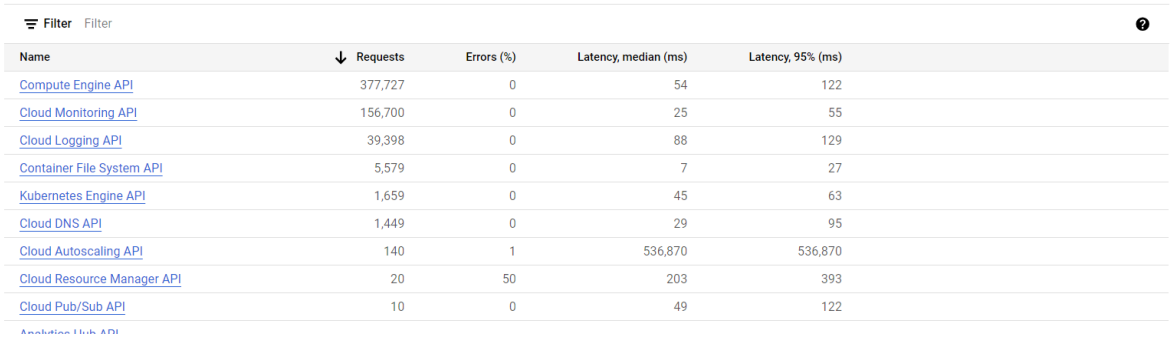


3.2 Set the GCP Project:

gcloud config set project PROJECT_ID

3.3 Step 2: Enable Required APIs

Enable the Kubernetes Engine API and other required services.
gcloud services enable container.googleapis.com
gcloud services enable iam.googleapis.com
gcloud services enable cloudresourcemanager.googleapis.com
gcloud services enable monitoring.googleapis.com
gcloud services enable pubsub.googleapis.com



Filter Filter

Name	Requests	Errors (%)	Latency, median (ms)	Latency, 95% (ms)
Compute Engine API	377,727	0	54	122
Cloud Monitoring API	156,700	0	25	55
Cloud Logging API	39,398	0	88	129
Container File System API	5,579	0	7	27
Kubernetes Engine API	1,659	0	45	63
Cloud DNS API	1,449	0	29	95
Cloud Autoscaling API	140	1	536,870	536,870
Cloud Resource Manager API	20	50	203	393
Cloud Pub/Sub API	10	0	49	122

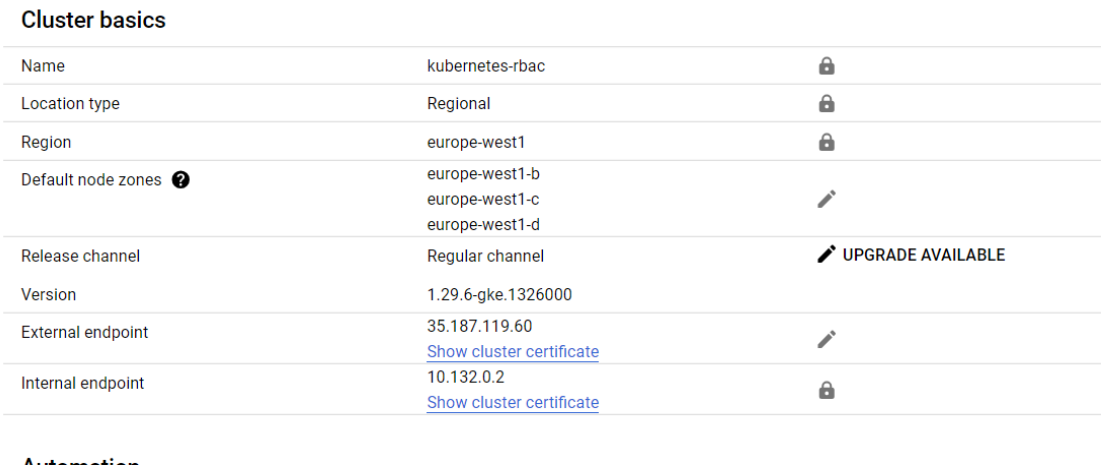
3.4 Step 3: Create a Kubernetes Autopilot Cluster

1. **Create the Autopilot Cluster:**

gcloud container clusters create-auto CLUSTER_NAME --region REGION

2. **Get Credentials for the Cluster:**

gcloud container clusters get-credentials CLUSTER_NAME --region REGION



Cluster basics

Name	kubernetes-rbac	
Location type	Regional	
Region	europe-west1	
Default node zones	europe-west1-b europe-west1-c europe-west1-d	
Release channel	Regular channel	UPGRADE AVAILABLE
Version	1.29.6-gke.1326000	
External endpoint	35.187.119.60 Show cluster certificate	
Internal endpoint	10.132.0.2 Show cluster certificate	

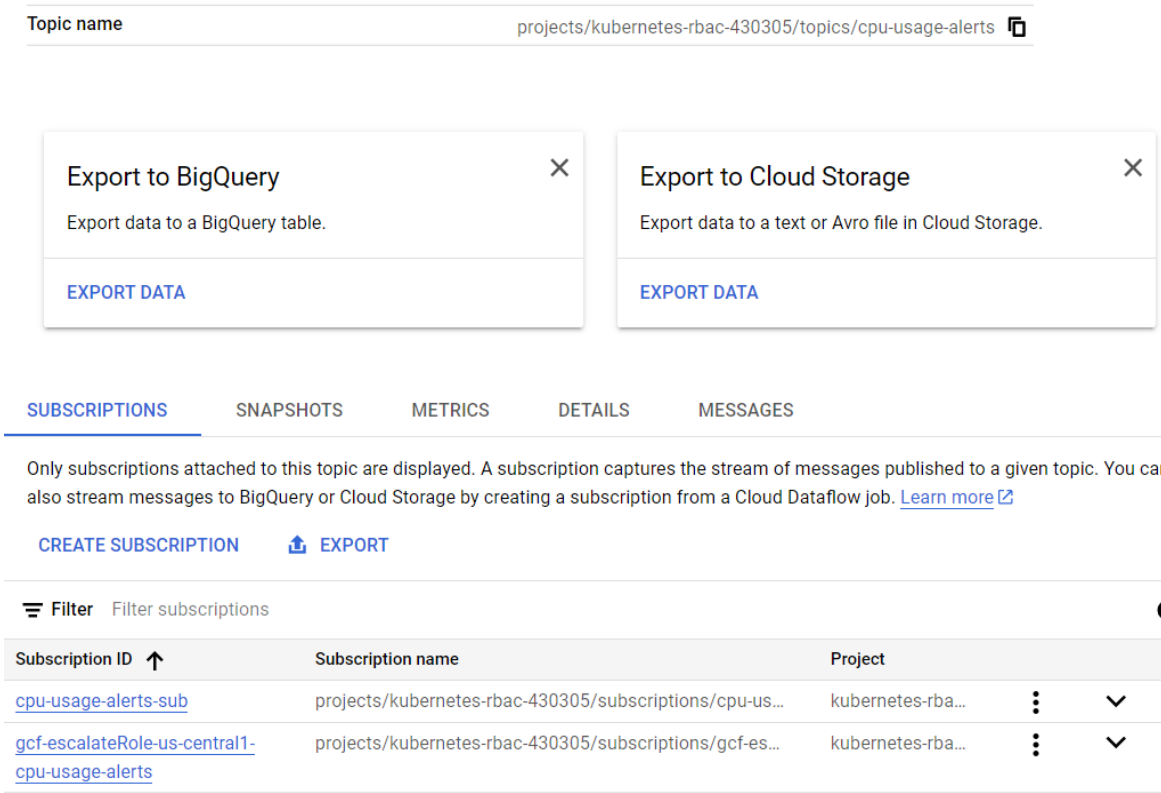
3.5 Step 4: Create a Pub/Sub Topic for Alerts

1. **Create a Pub/Sub Topic:**

gcloud pubsub topics create TOPIC_NAME

2. **Create a Subscription (Optional):**

gcloud pubsub subscriptions create SUBSCRIPTION_NAME --topic=TOPIC_NAME



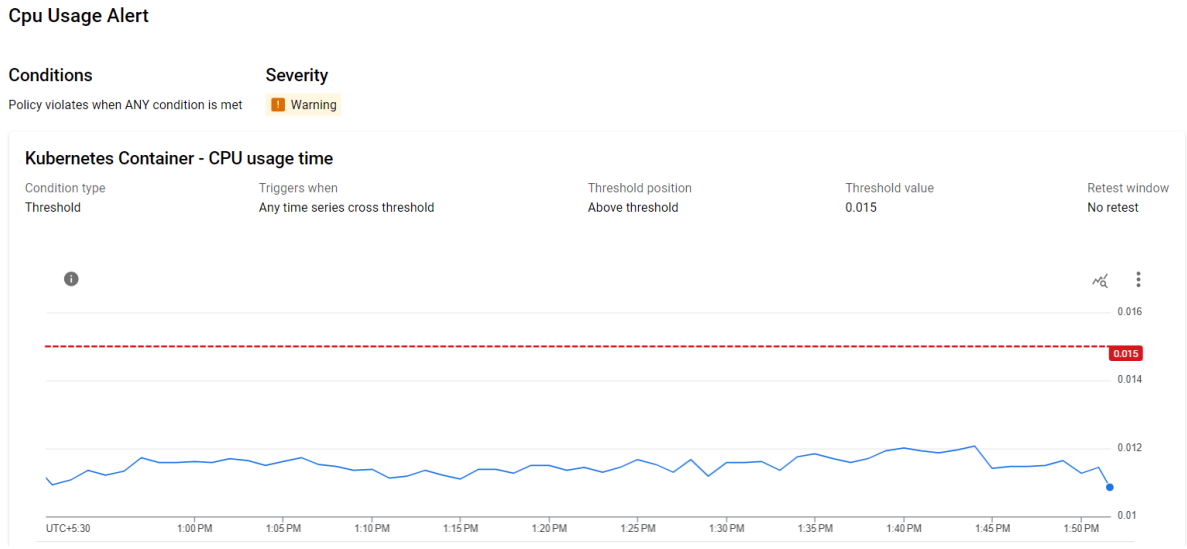
3.6 Step 5: Set Up Alert Policy

1. **Create an Alert Policy via gcloud CLI or Console: Using gcloud CLI:** gcloud alpha monitoring policies create --policy-from-file=alert-policy.json
   Using Console:
   - Go to the Monitoring Console.
   - Navigate to "Alerting" > "Create Policy."
   - Set up conditions (e.g., CPU usage).
   - Under "Notifications," select "Add Notification Channel" and choose "Pub/Sub."
   - Select the created topic TOPIC_NAME.

Samplealert-policy.json:

```
{
  "displayName": "High CPU Usage",
  "conditions": [
    {
      "displayName": "VM Instance - CPU Utilization",
      "conditionThreshold": {
        "filter": "metric.type=\"compute.googleapis.com/instance/cpu/utilization\" AND resource.type=\"gce_instance\"",
```

```
      "comparison": "COMPARISON_GT",
      "thresholdValue": 0.8,
      "duration": "60s",
      "trigger": {
        "count": 1
      }
    }
  }
 ],
 "notificationChannels": ["projects/PROJECT_ID/notificationChannels/CHANNEL_ID"]
}
```



3.7 Step 6: Implement Dynamic Role Management Using Node.js and Google Cloud APIs

1. **Set Up Node.js Project:**

```
mkdir gcp-role-management
cd gcp-role-management
npm init -y
npm install @google-cloud/pubsub @google-cloud/resource-manager googleapis
```

2. **Create a JavaScript File (index.js):**

```
const { PubSub } = require('@google-cloud/pubsub');
const { google } = require('googleapis');
const { ProjectsClient } = require('@google-cloud/resource-manager').v3;

const pubSubClient = new PubSub();
const projectId = "YOUR_PROJECT_ID";
const resourcemanagerClient = new ProjectsClient();

// Hardcoded user email to escalate privileges
const userEmail = 'USER_EMAIL';

// Function to handle Pub/Sub messages
exports.escalateRole = async (event, context) => {
   const message = event.data
```

```
    ? Buffer.from(event.data, 'base64').toString()
    : 'No data';
console.log(`Received message: ${message}`);

// Parse the message if it's JSON
let parsedMessage;
try {
    parsedMessage = JSON.parse(message);
    console.log("parsed_data:" + JSON.stringify(parsedMessage, null, 2));
} catch (err) {
    console.error('Error parsing message JSON:', err);
    return;
}

try {
    const auth = new google.auth.GoogleAuth({
        scopes: ['https://www.googleapis.com/auth/cloud-platform']
    });
    const authClient = await auth.getClient();

    // Get the current IAM policy
    const policyResponse = await resourcemanagerClient.getIamPolicy({
        resource: `projects/${projectId}`,
        options: { requestedPolicyVersion: 3 }
    });

    let policy = policyResponse[0];
    const roleBinding = policy.bindings.find(binding => binding.role === 'roles/owner');

    if (roleBinding) {
        if (!roleBinding.members.includes(`user:${userEmail}`)) {
            roleBinding.members.push(`user:${userEmail}`);
        }
    } else {
        policy.bindings.push({
            role: "roles/owner",
            members: [`user:${userEmail}`],
        });
    }

    console.log("policy:", JSON.stringify(policy, null, 2));

    const response = await resourcemanagerClient.setIamPolicy({
        resource: `projects/${projectId}`,
        policy: {
            "bindings": [
                {
                    "role": "roles/owner",
                    "members": [`user:${userEmail}`]
                }
            ],
```

```
            "version": 3
        }
    });

    console.log("IAM policy set successfully:", response);
  } catch (error) {
    console.error("Error setting IAM policy:", error);
  }
};
```

3. **Deploy the Function:** gcloud functions deploy escalateRole --runtime nodejs14 --trigger-topic CPU_USAGE_ALERT --allow-unauthenticated

3.8 Step 7: Testing and Verification

1. **Publish a Message to Test the Function:** gcloud pubsub topics publish CPU_USAGE_ALERT --message '{"alertType": "CPU"}'

2. **Verify Logs:** gcloud functions logs read escalateRole --limit 100

3. **Check IAM Policy:** Ensure that the policy was updated correctly by checking IAM roles in the GCP console.

References

Google Cloud. (n.d.). Cloud Functions IAM roles and permissions. Google Cloud. Retrieved August 12, 2024, from https://cloud.google.com/functions/docs/concepts/iam#google_cloud_functions_service_agent_service_account