**Project-3**

**Integrate Grafana with Linux Server for high cpu utilization and create a graph in Grafana**



Tanuja Tiwari

Table of Contents

* **Introduction**
* **Objective**
* **Abstract of the Project**
* **Services used**
* **Implementation**
* **Task 1: Sign in to AWS Management Console**
* **Task 2: Launching an EC2 Instance**
* **Task 3: Creating IAM Role and Attach this role to EC2 instance**
* **Task 4:** **SSH into EC2 Instance and install necessary Software.**
* **Task 5: Connect Grafana to the Data Source**
* **Task 6: Create a CPU Utilization Graph**
* **Task 7: Testing Grafana Dashboard by Stressing CPU Utilization**
* **Conclusion**
* **Introduction**

**The main purpose of this project is to set up Grafana with a Linux server to track and visualize CPU usage, enabling real-time monitoring and performance analysis using interactive dashboards.** **This project focuses on integrating Grafana with a Linux server to monitor high CPU utilization and visualize system performance through real-time graphs and dashboards.**

* **Objective**

**To visualize high CPU usage on a Linux server by integrating Grafana with system monitoring tools and creating interactive performance graphs.**

* **Abstract of this project**

**The project focuses on creating a cost-effective and user-friendly monitoring framework using open-source technologies. It also aims to support long-term performance analysis through historical data retention. With this integration, organizations can improve system reliability, optimize resource usage, and streamline infrastructure management.**

* **Services used**

1. **AWS cloud**
2. **EC2**
3. **IAM**
4. **Grafana**
5. **Cloud Watch**

* **Implementation**

**Task 1: Sign in to AWS Management Console**

1. **Click on the Sign in to console, and you will get redirected to AWS Console in a new browser tab.**
2. **On the AWS sign-in page,**
   * **Leave the Account ID as default. Never edit/remove the 12-digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.**
   * **Now enter your username and password in AWS Console and click on the Sign in button.**
3. **Once Signed in to the AWS Management Console, Make the default AWS Region as ap south-1.**

**Task 2: Launching an EC2 Instance**

**In this task, we are going to launch an EC2 Instance that will be used for checking various features in CloudWatch.**

1. **Make sure you are in the ap south-1 Region.**
2. **Navigate to EC2 by clicking on the Services menu in the top, then click on EC2 in the Compute section.**
3. **Click on Launch instances button.**

**Name: Enter MyEC2Server.**

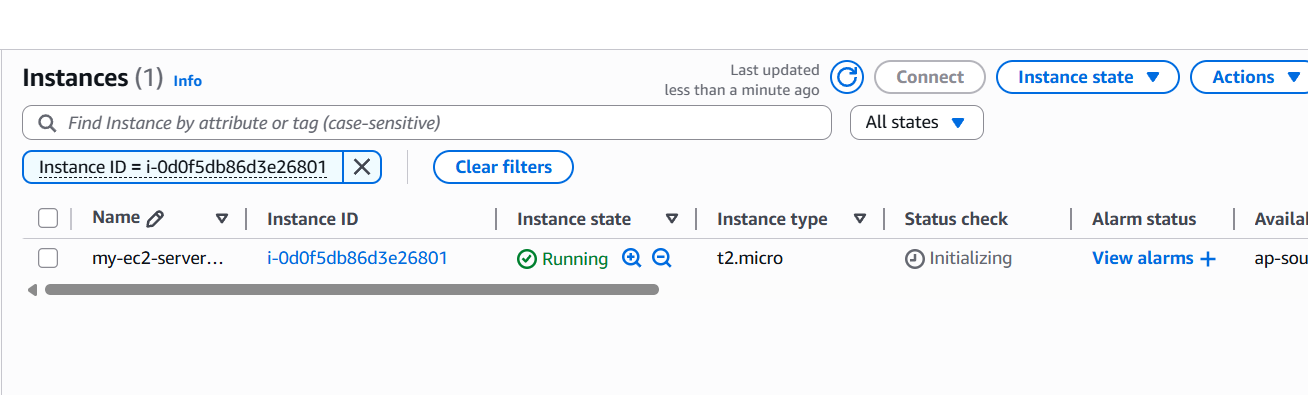
1. **For Amazon Machine Image (AMI): Select Amazon Linux and the select Amazon Linux 2 AMI from the drop-down.**
2. **For Instance, Type: Select t2.micro**
3. **For Key pair: Select Create a new key pair Button**

* **Key pair name: MyEC2Key**
* **Key pair type: RSA**
* **Private key file format: .ppk**

1. **Select Create key pair Button**
2. **In Network Settings Click on Edit button:**

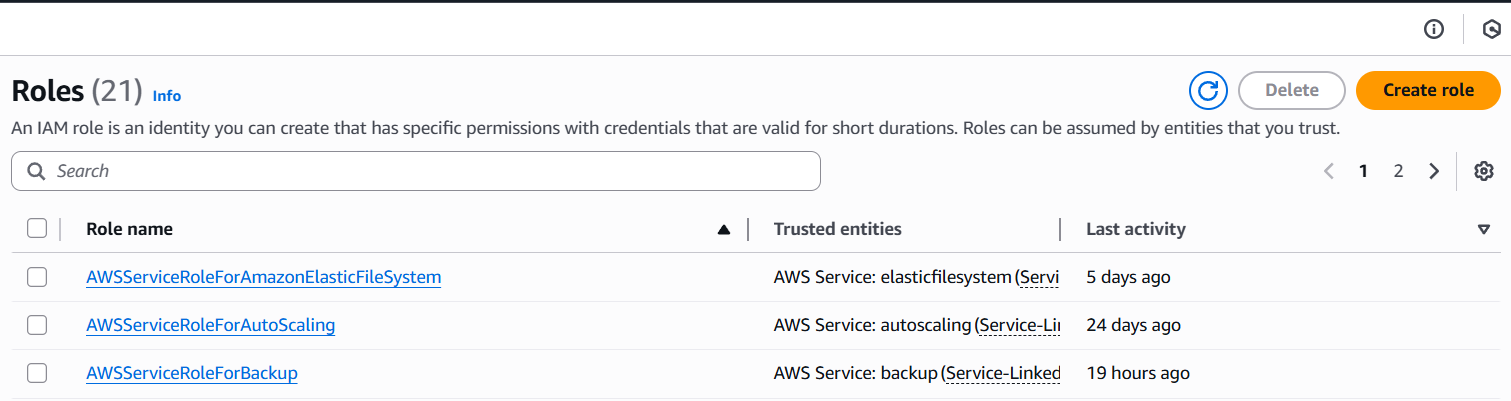
* **Auto-assign public IP: Enable**
* **Select Create new Security group**
* **Security group name: Enter MyEC2Server\_SG**
* **Description: Enter Security Group to allow traffic to EC2**
* **To add SSH:**
* **Choose Type: Select SSH**
* **Source: Select Anywhere**
* **Keep Rest the things as Default and Click on Launch Instance Button.**

1. **Select View all Instances to View the Instance you created.**
2. **Launch Status: Your instance is now launching. Click on the instance ID and wait for complete initialization of the instance (until the status changes to running).**

****

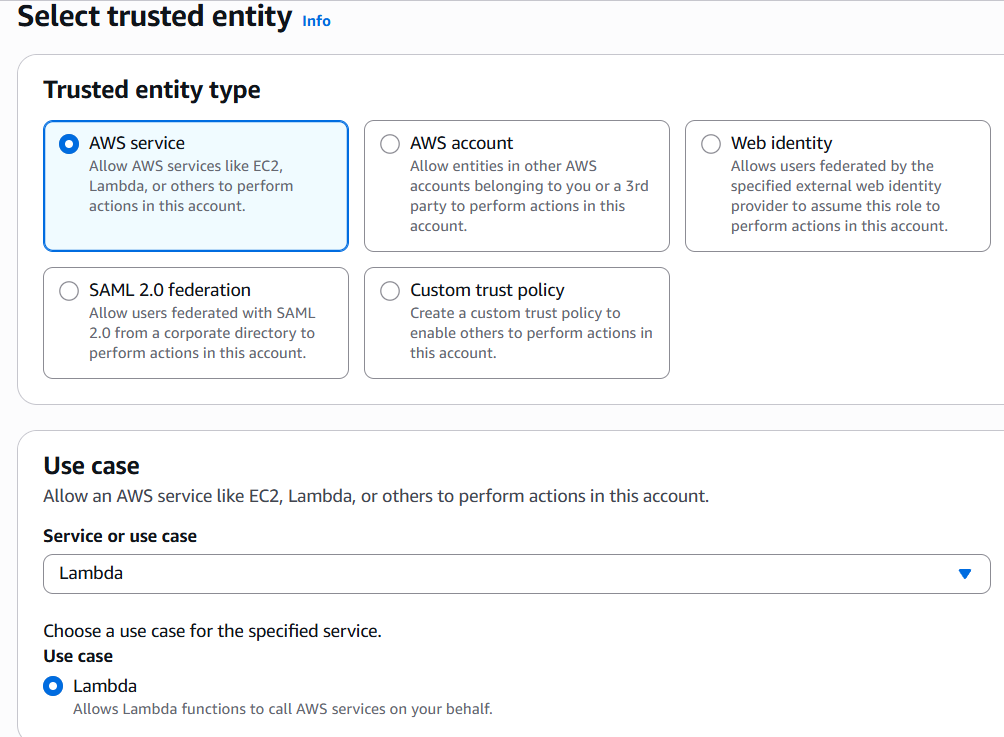
**Task 3: Creating IAM Role and Attach this role to EC2 instance**

1. **Navigate to IAM by clicking on the Services menu in the top, then click on IAM in the Compute section.**
2. **Click on roles and then click on create roles.**

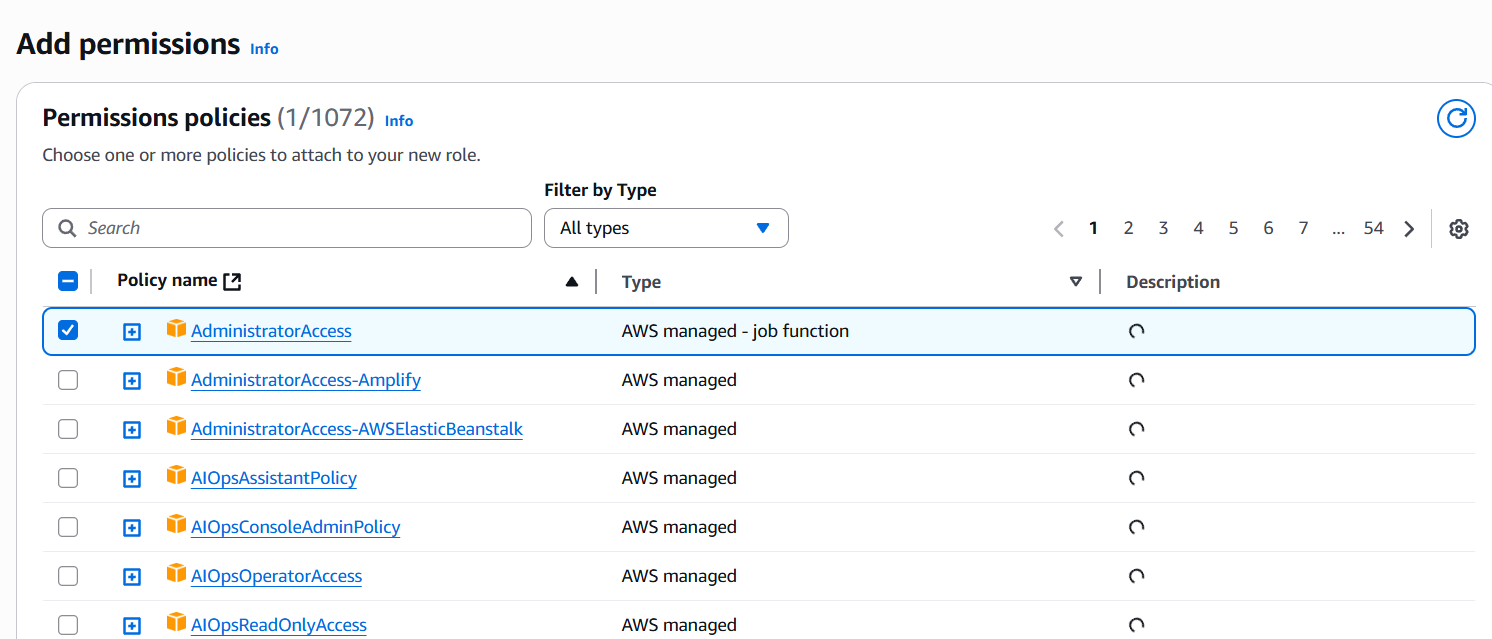
****

1. **Trusted entity type: AWS service**

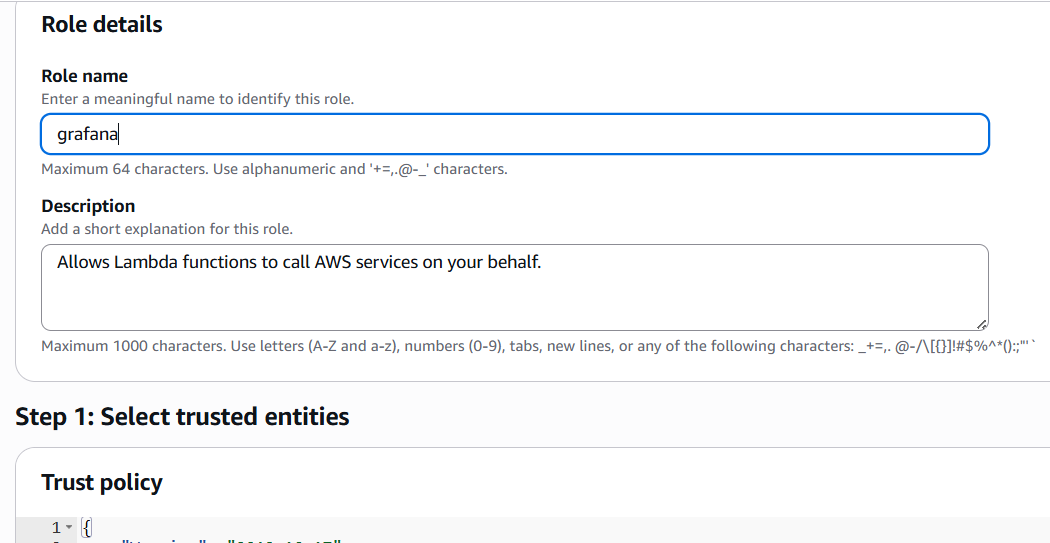
**Chose service or use case: EC2**

****

1. **Click on next.**
2. **Add Permissions: AdministratorAccess and click on next.**

****

1. **Enter Role name: Grafana-Role and then click on create role.**

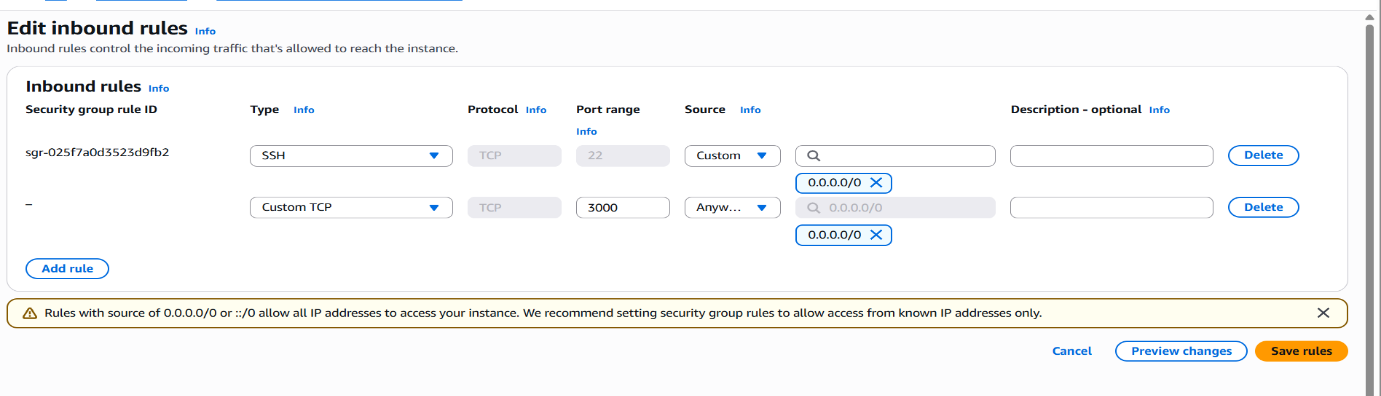
****

1. **Go back to your EC2 instance.**
2. **Select your EC2 instance and then click on Actions🡪 Security🡪Modify IAM roles.**
3. **Chose the IAM role you have created and click on update IAM role.**
4. **Click on Security then click on edit inbound rules.**
5. **Click on Add Rule, then Type: custom TCP**

**Port: 3000**

**Source: 0.0.0.0/0**

1. **Click on save rules.**

****

**Task 4:** **SSH into EC2 Instance and install necessary Software.**

* 1. **Once you are logged into the EC2 instance, switch to root user.**

sudo su

* 1. **Update:**

yum update -y

* 1. **To download Grafana enter this command:**

**yum install -y**[**https://dl.grafana.com/enterprise/release/grafana-enterprise-12.0.2- 1.x86\_64.rpm**](https://dl.grafana.com/enterprise/release/grafana-enterprise-12.0.2-%20%20%20%20%20%20%20%201.x86_64.rpm)

* 1. **Enter the following Commands:**
* **systemctl start grafana-server**
* **systemctl enable grafana-server**
* **systemctl status grafana-server**

**Task 5: Connect Grafana to the Data Source**

* + 1. **Go to browser, Enter: http://<your\_server\_ip>:3000**

**(Default login: admin / admin)**

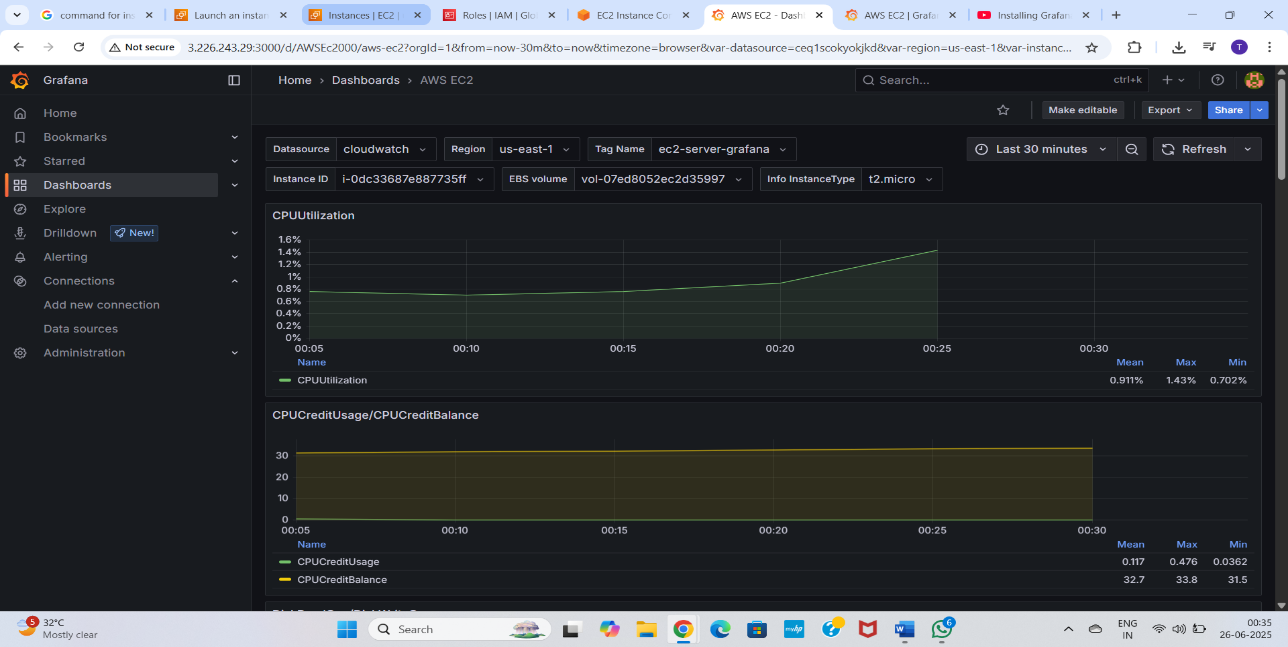
**Task 6: Create a CPU Utilization Graph**

1. **Go to Dashboards > New > New Dashboard.**
2. **Click on Import Dashboard.**
3. **Click on the link: grafana.com/dashboards.**
4. **In search bar, search for AWS EC2 and then click on it.**
5. **Copy the ID and Paste it in the import dashboard next, click on load**
6. **It will show a dashboard of CPU utilization.**

* **Select:**
* **Datasource: Cloudwatch**
* **Region: ap-south-1**
* **Tag name: MyEC2Server (name of your EC2 instance)**
* **Instance id: Select your EC2 instance ID**

**Task 7: Testing Grafana Dashboard by Stressing CPU Utilization**

1. **Go back to your EC2 instance terminal and enter the following command:**
   * **sudo dnf update**
   * **sudo yum install stress-ng -y**
   * **stress**
   * **stress --cpu 8 --timeout 60**
2. **By running this stress command, we have manually increased the CPU utilization of the EC2 Instance.**
3. **On the Graph, you can see places where CPUUtilization has gone above the 25% threshold.**

****

* **Conclusion**

**Overall, the project highlights the value of using open-source tools for building a scalable, customizable, and cost-effective monitoring solution for Linux servers. It provides system administrators with greater visibility into server performance and contributes to maintaining high system availability and reliability.**