xgrid.co

COS70008-Technology Innovation Project

# XC3

## Federated IAM User

# User Manual

**Team 12**


**Student Details:**

Pradhumna Dhungana(104491800)

Bhavya(104487966)

Susanta Bhujel(104491527)

Nilanshu Basnet(104346575)

# TABLE OF CONTENT


Contents Page No

1 Introduction .....1

2 System Requirements .....1

3 Installation Instructions .....1

4 Architecture Diagram .....1

5. Setting up alerts for untagged resources.....3

6. Testing Lambda Functions .....5

7. User Interface Overview(Grafana Panel).....6

8. Using the Feature .....12

9. Setting up alerts in Grafana.....12

10. Support.....15

## 1. Introduction

Welcome to a comprehensive guide on the XC3 Federated IAM User Cost Tracking feature. This advanced feature provides detailed insights into cloud expenses for federated IAM users. With this advancement, users can easily monitor, assess, and manage expenses related to resources allocated to individual users. The system also ensures proper tagging of resources, allowing for automated cost metric population in Grafana for in-depth reporting and budgeting decisions. If resources are not tagged, the system will alert users through their preferred communication methods to ensure effective cost control. Explore the following sections to discover how to make the most of this feature and get your cloud framework ready for efficient cost management.

## 2. System Requirements:

In order to access all the features of the XC3 Federated IAM User Cost Tracking, there are certain system requirements that need to be met. These requirements are put in place to make sure that every team member is able to use the system efficiently:

- AWS CLI
- Terraform 1.0+
- Python 3.9
- Grafana and Prometheus
- VS Code connected to WSL

## 3. Installation instructions:

The XC3 Federated IAM User Cost Tracking upgrade is smoothly incorporated into the XC3 platform. There is no separate setup required for this particular feature. To make use of this capability,one can follow these steps:

- Login to your AWS account
- Deploy XC3 in your account by specifying the project names and filling out all necessary fields.

## 4. Architecture Diagram:

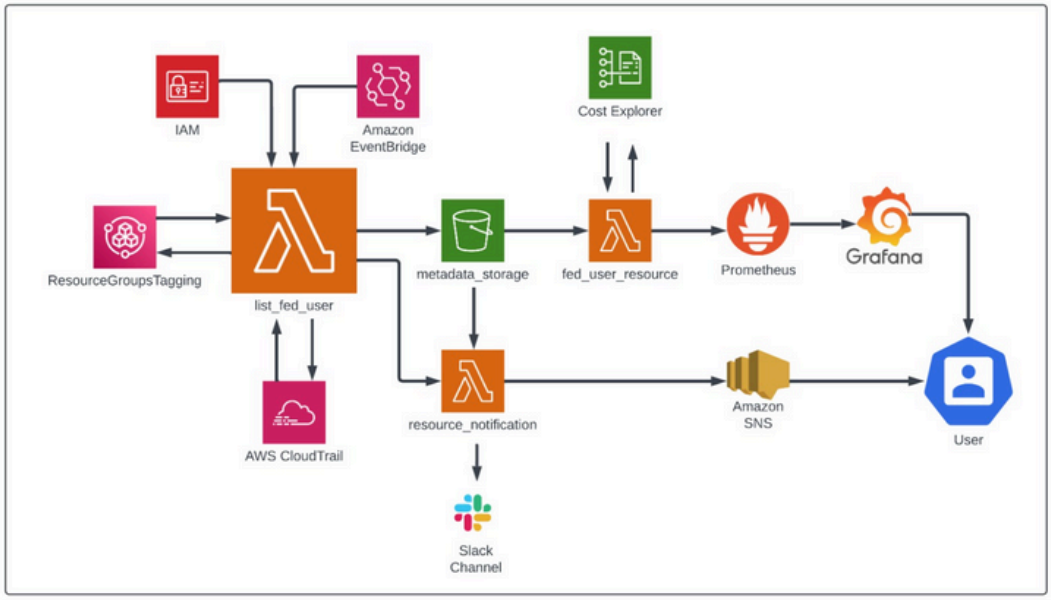
**Event Trigger:** The system is triggered by an EventBridge cron job, which runs every midnight. This triggers the list_fed_user Lambda function.

**list_fed_user Lambda:** This function fetches information about federated users, their provisioned resources, and their tagging compliance status. It then stores this data in an S3 bucket named metadata-storage. Additionally, it triggers the resource_notification Lambda function.

**resource_notification Lambda:** This function is triggered by new objects uploaded to the metadata-storage S3 bucket. It retrieves the data, checks for tagging compliance, and sends notifications to the Slack group and via email if resources are untagged or non-compliant. Emailing is done using SNS.

**fed_user_resources Lambda:** This function is also triggered by new objects uploaded to the metadata-storage S3 bucket. It retrieves the stored data, calculates the cost of compliant resources, and pushes this information to a Prometheus server. Users can view this data through visualizations on a Grafana dashboard.

Below is the architectural diagram illustrating our implemented feature:




In summary, this system automates the monitoring of federated users' resources, ensures tagging compliance, sends notifications for non-compliant resources, and provides cost analysis through visualizations for users.

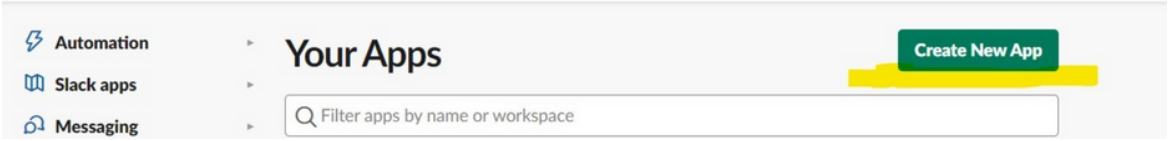
## 5. Setting up alerts for untagged resources

Once you have the application up and running you can access the feature by following these steps:

1. Log into your AWS account (where the installation has taken place)
2. Create webook from slack by following these steps.

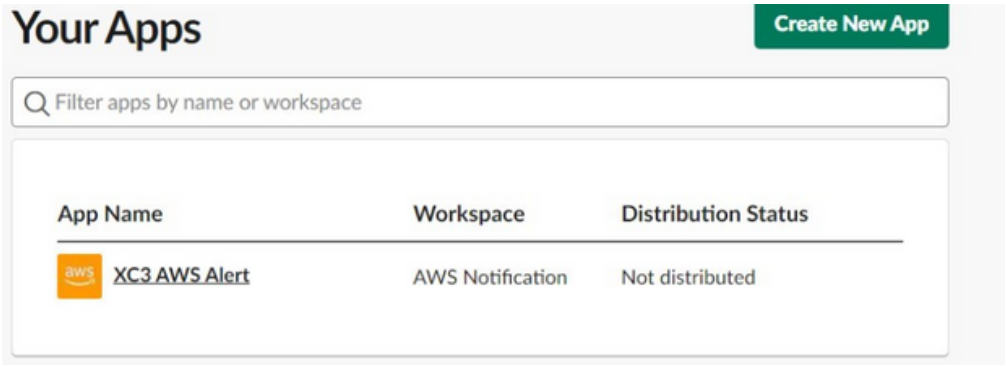
### How to create a webhook using slack

Slack App: If you don't already have a Slack app, create one by visiting Slack API Apps (https://api.slack.com/apps) and clicking "Create New App." Fill in the required details to set up your app.

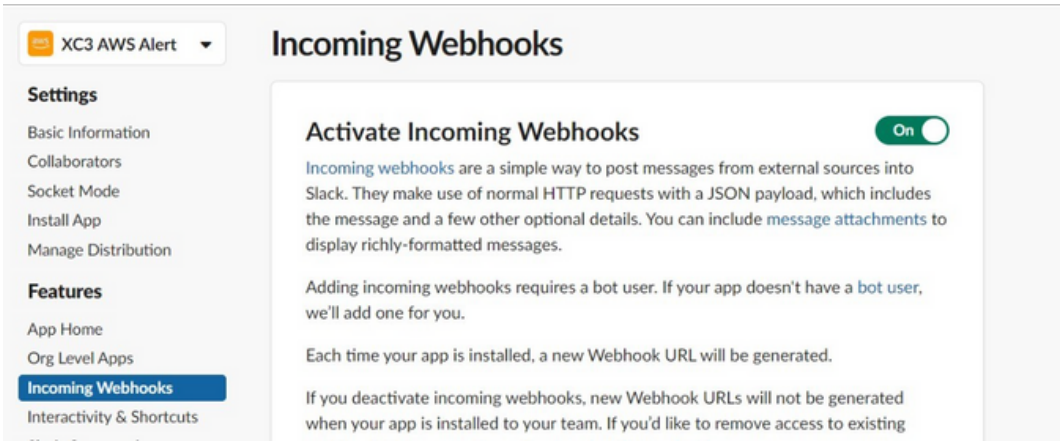



### Step 1: Activate Incoming Webhooks

1. Access your Slack app dashboard by clicking on your app.




2. Navigate to Incoming Webhooks on the left side of the screen.
3. Toggle the switch to activate incoming webhooks.



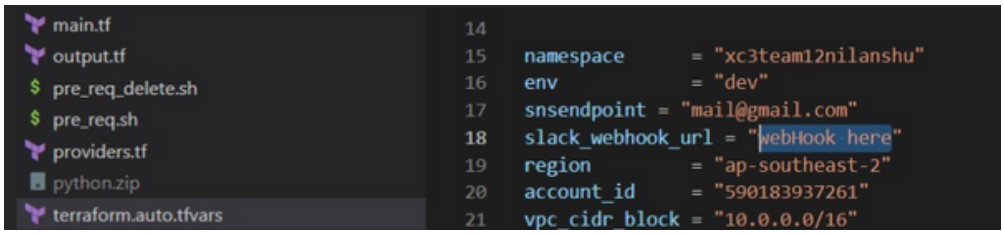

4. Click Add new Webhook to Workspace at the bottom of the page to generate a webhook URL.

## Step 2: Obtain the Webhook URL

Copy the generated webhook URL.

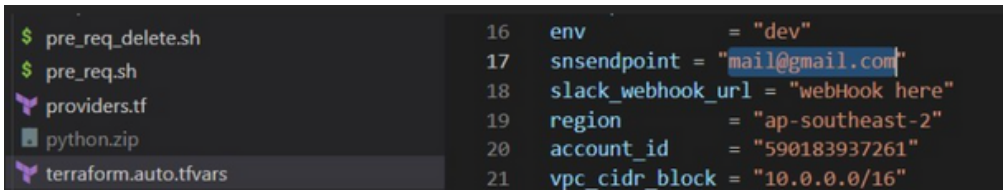

## Step 3: Configure terraform.auto.tfvars

1. Locate the terraform.auto.tfvars file within the infrastructure folder of your project.
2. Set the value of the slack_webhook_url variable to the copied webhook URL.



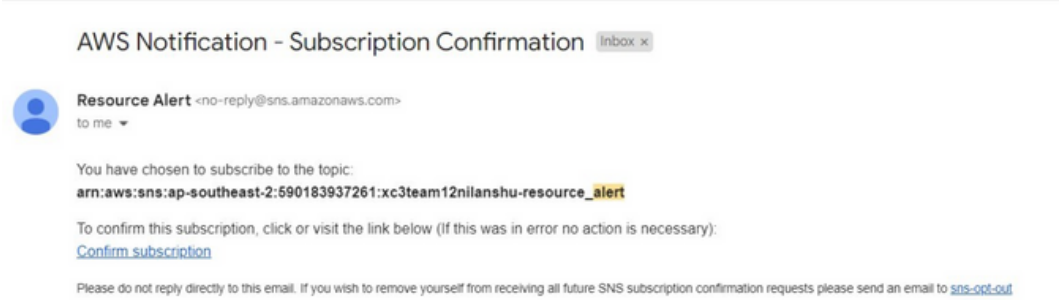

## Step 4: Set Email Notification Endpoint

In the same terraform.auto.tfvars file, update the value of the snsendpoint variable with the desired email address where you want to receive notifications.




## Step 5: Confirm Subscription

1. Deploy your infrastructure.
2. AWS will send a confirmation email to the specified snsendpoint address.
3. Open the email and follow the instructions to confirm the subscription.




By completing these steps, you'll have successfully configured the webhook for notifying untagged resources via Slack and email. Stay informed about your AWS environment effortlessly!

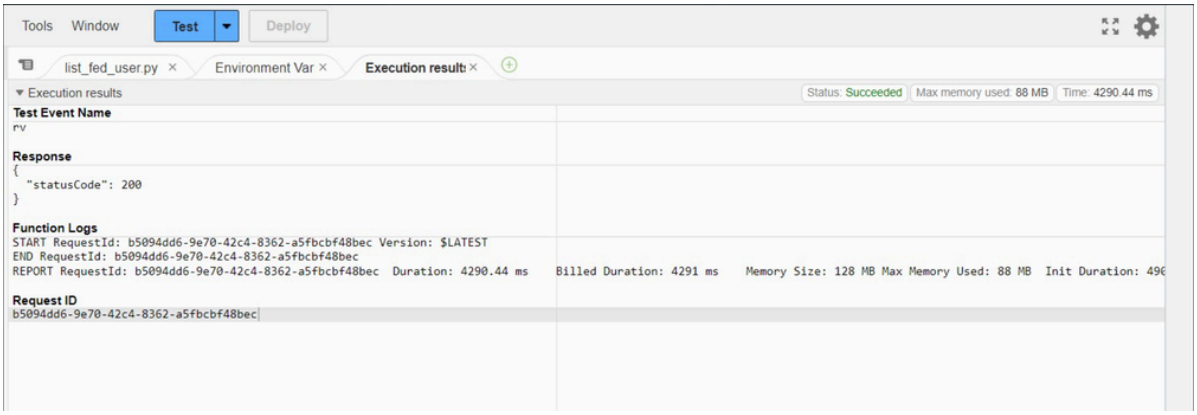
## 6. Testing Lambda Functions:

To view real-time data on a Grafana dashboard, follow these steps:

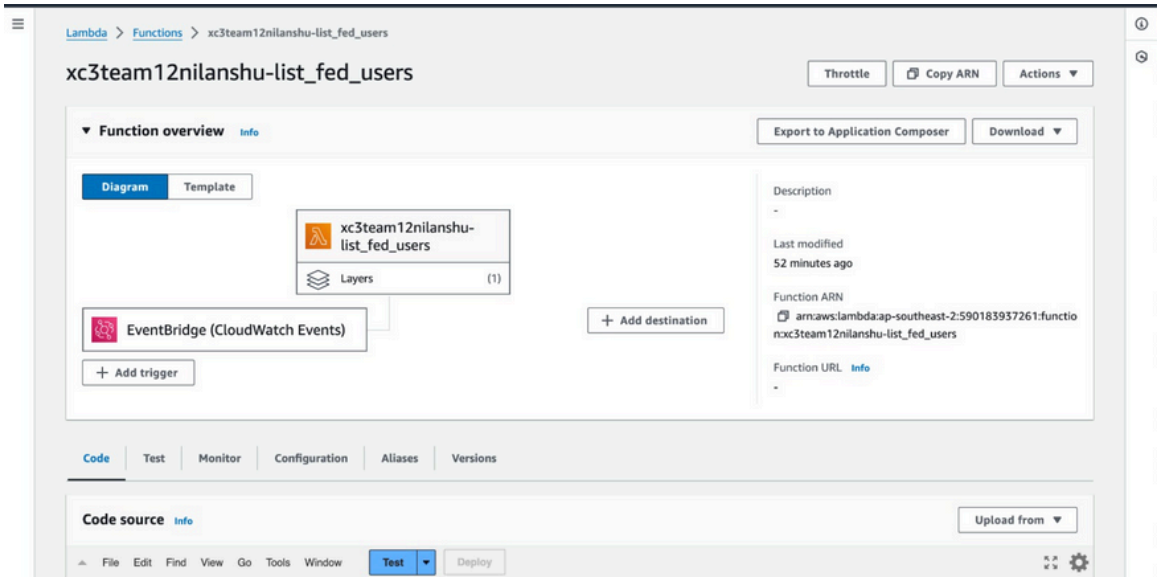
- **list_fed_user**

  Function name: {namespace}-list_fed_user

  Choose this lambda function from the list on the lambda dashboard. Thereafter, test the function by generating an event.



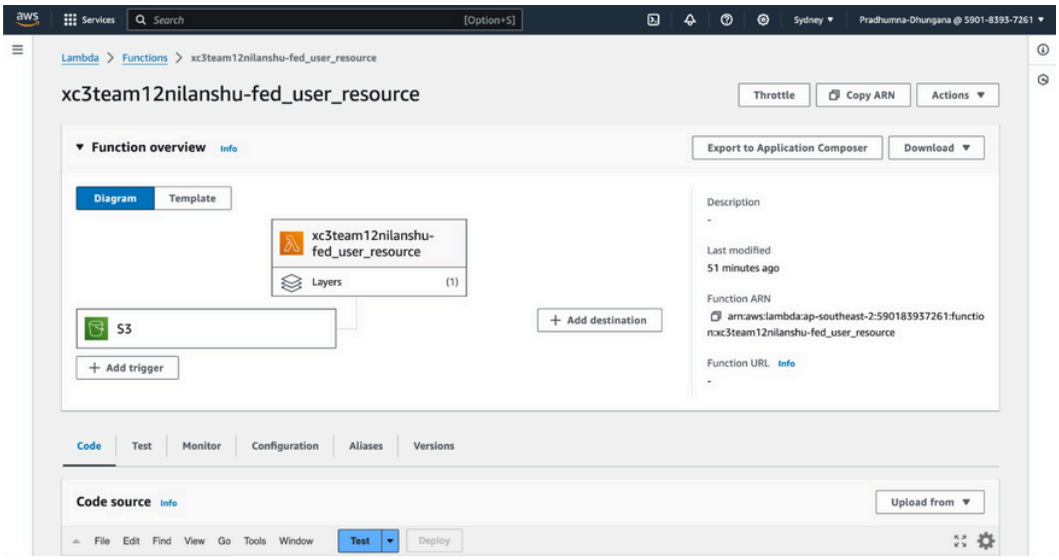

After testing the lambda, the function overview should look like follows:



- **fed_user_resource**

  Function name: {namespace}-fed_user_resource

  Choose this lambda function from the list on the lambda dashboard. Thereafter, test the function by generating an event. The overview then should look something like this. If in case you are not able to see the S3 trigger here, try to run terraform plan and apply again till you see this.



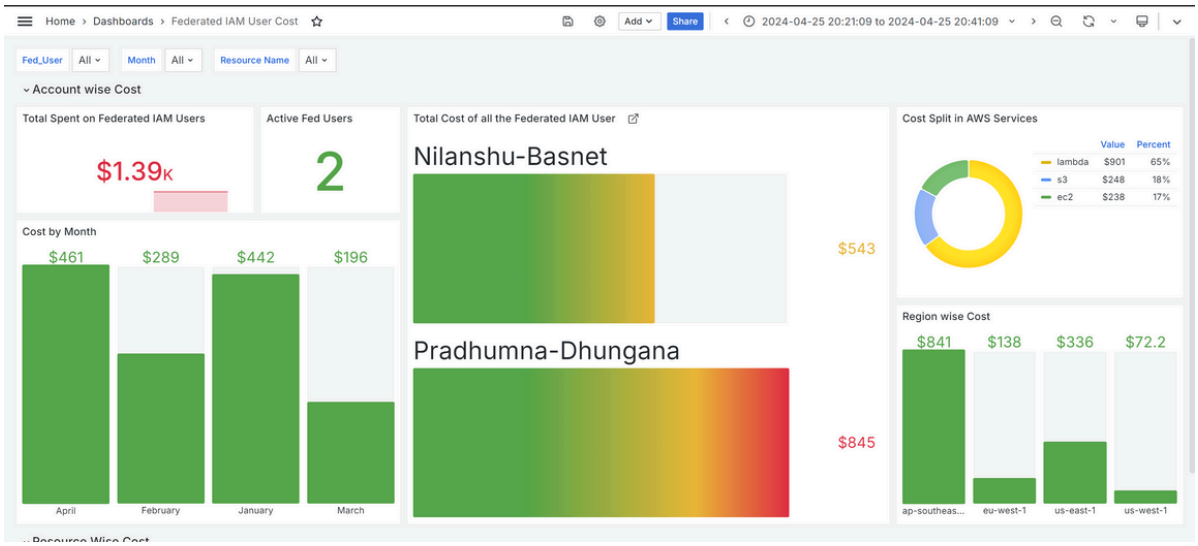

## 7. Grafana Dashboard:

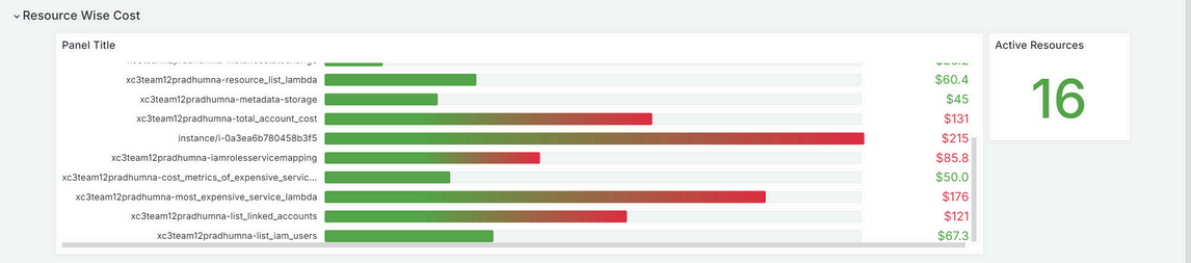
### Login Panel:

- Navigate to the EC2 public IP on port 3000
- Connect using the HTTP protocol
- Login to the dashboard with Grafana Default Credentials
- Enter Username: admin
- Enter Password: admin

### Federated IAM User Cost Dashboard:

The Federated IAM User Cost Analysis Dashboard provides a detailed look at the costs associated with Federated IAM Users on AWS. It is essential for tracking spending and improving the oversight of AWS resources linked to identity and access control. Detailed explanation of the panels are in the paragraphs to follow






## Detailed Dashboard Composition:

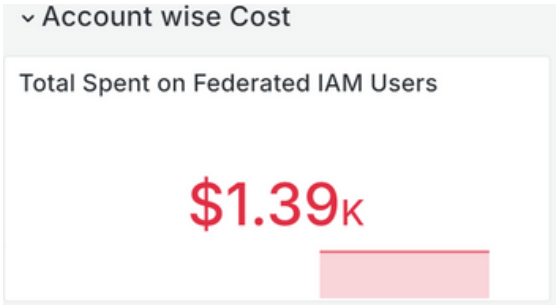
In this dashboard, you will find various important components that offer unique insights into the spending and operations of federated IAM users.

### *Row 1: Account Wise Cost:*

It consists of 6 panels with 3 static in regards to dynamic variables and the other three dynamic

Panel 1: Total Spent on Federated IAM Users

This section provides a summary of the total costs incurred by all Federated IAM Users.




In general, this dashboard panel serves as a reporting tool for an administrator or finance team to oversee and control the expenses linked to federated identity users. At present, it indicates that no costs have been accrued. This is because the cost is for the resource at the moment and these resources were just created. In case, anyone has the admin privileges for the dashboard, the cost will be different there.

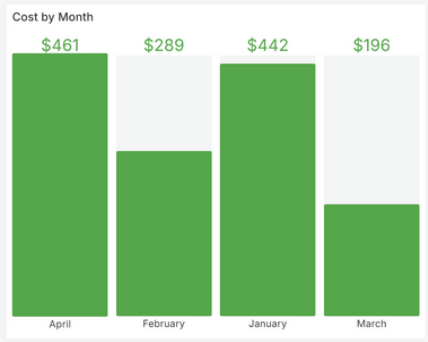
Panel 2: Active Federated Users

This metric shows the number of federated IAM users currently active.


The panel is simple and provides real-time updates on the number of federated users actively using the system, designed for system administrators and security personnel to easily track recent system activity. It is showing 1 as the data was fetched for one single account but with admin privileges, it will show the count of all the Federated IAM User Accounts there.

Panel 3: Cost by Month

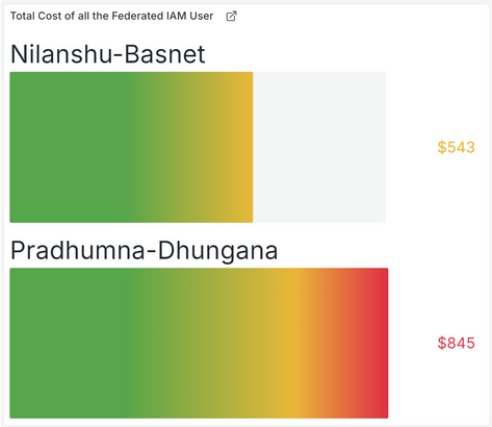
This bar chart displays the costs incurred each month.




Simply put, this panel offers a look at costs throughout history(for every month), which is crucial for keeping track of budgets and analyzing finances in a business.

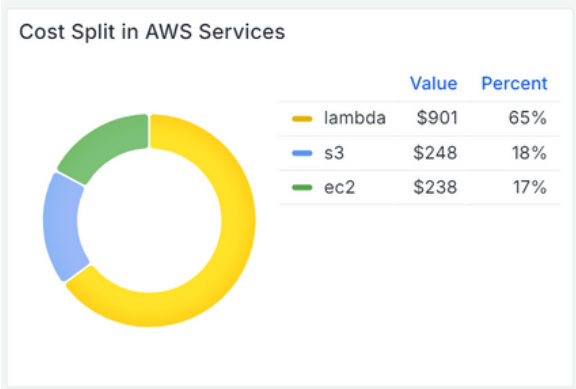
Panel 4: Total Cost of all the Federated IAM User

This bar chart lists individual Federated IAM Users and their associated costs.




Panel 5: Cost Split in AWS Services

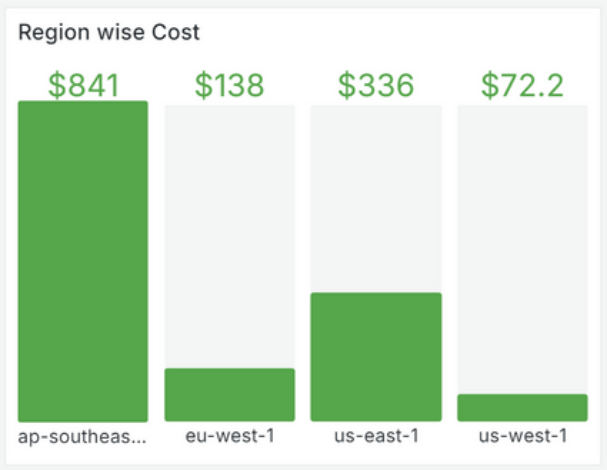
This pie chart breaks down the costs by AWS service.




IThe panel is designed to update with real usage data as time goes on and the services are used more.

Panel 6: Region-wise Cost

This bar chart breaks down costs by AWS region.

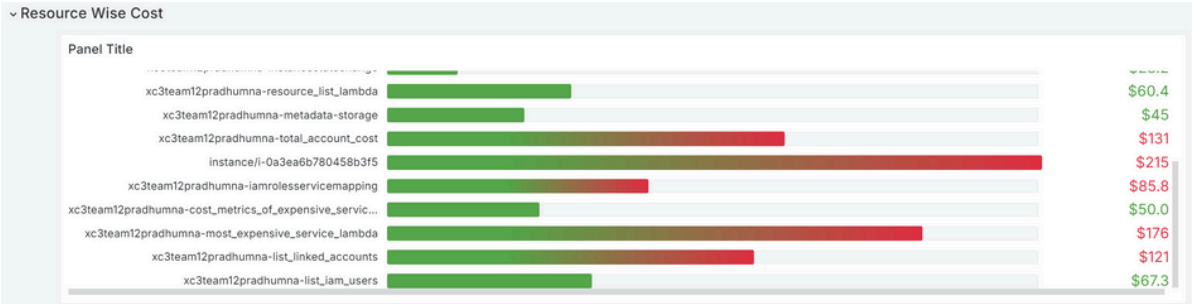



***Row 2: Resource Wise Cost:***

It consists of 2 panels similar to the Account Wise Cost row but with resource cost into consideration.

Panel 1: Resource Wise Cost

This bar chart breaks down costs by resources used.




The visualization is designed to display the cost of each resource as horizontal bars of different lengths, based on cost. This chart helps identify which resources are using the budget when costs are incurred, leading to improved cost management and optimization.

Panel 2: Active Resources

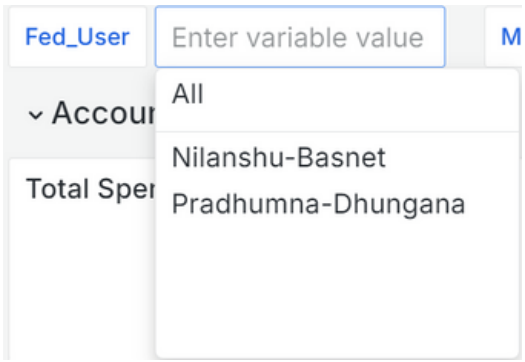
This metric shows the number of resources that are currently active.


The visualization is clear and helpful, providing a concise overview of resource usage that is important for monitoring the system, analyzing costs, and planning for capacity. In an AWS environment, the term 'active' may indicate that resources are currently running and possibly incurring costs, making this metric a useful indicator of potential expenses.

## *Variables set in the Dashboard*

Variable 1: Fed_User

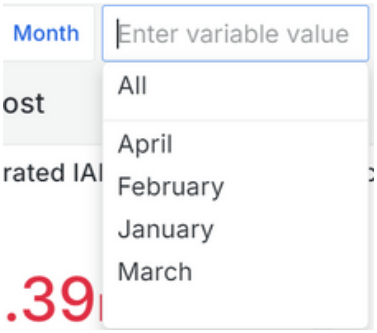
Filter the dashboard view to display cost data for a specific Federated IAM User or all users.




Grafana uses variables to enhance dashboard interactivity. You can easily switch between data displayed on the dashboard by selecting options from dropdown menus, without having to edit the queries manually. This feature allows for a more dynamic and user-friendly dashboard experience. Integrating variables in Grafana simplifies the user interface, enabling users to efficiently handle and analyze a wide range of data points in various dimensions.

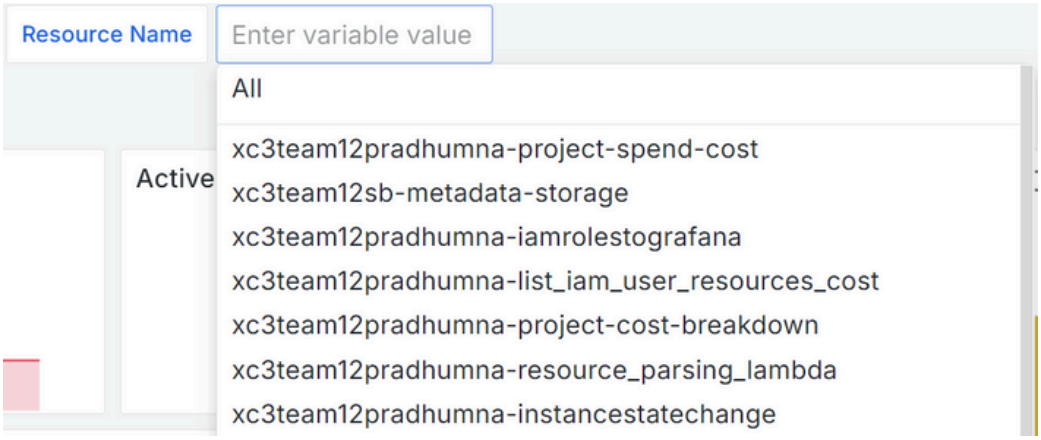
Variable 2: Month

Adjust the dashboard to show cost data for a specific month or for all time periods available.

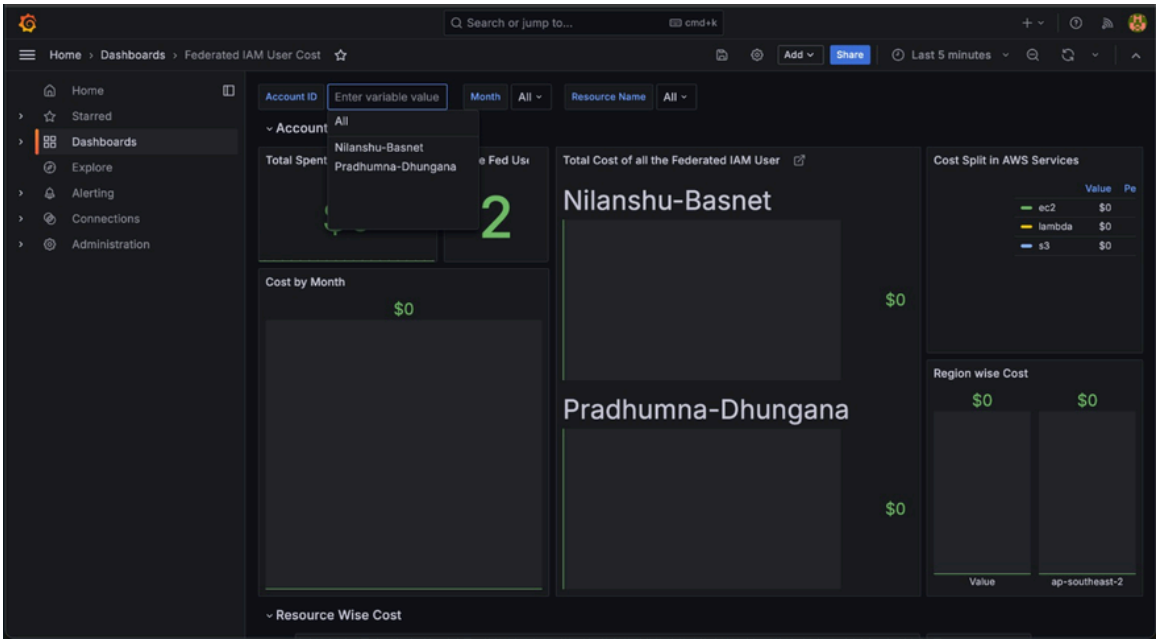



Variable 3: Resource Name

Adjust the dashboard to show cost data for different resources that are being used.




## How it looks in XC3



## 8. Operating the Feature

The "Federated IAM User Dashboard" allows you to: -

- Easily see how much Federated IAM Users are spending in your AWS environment.
- Analyze the different costs for each Federated IAM User, giving you valuable information on spending habits.
- Customize the display for certain timeframes, services, or AWS regions to better comprehend your AWS spending.

## 9. Setting up alerts in Grafana:

In order to respond quickly to incidents, it is important to ensure that your systems are reliable and to integrate real-time notifications into your team's workflow. This guide will help you set up Grafana alerts to notify your team in a dedicated Slack channel, allowing them to react promptly.

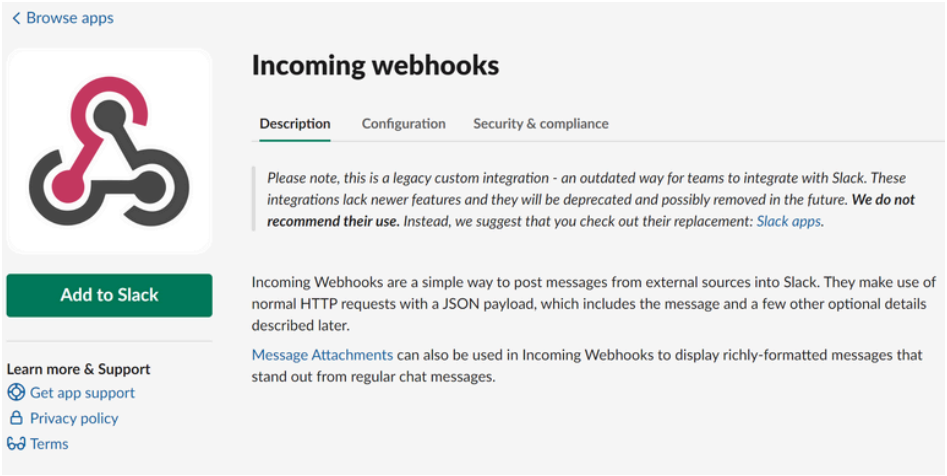
**Prerequisites**

- Admin access to a Grafana server instance.
- Permissions to manage apps and create webhooks in a Slack workspace.

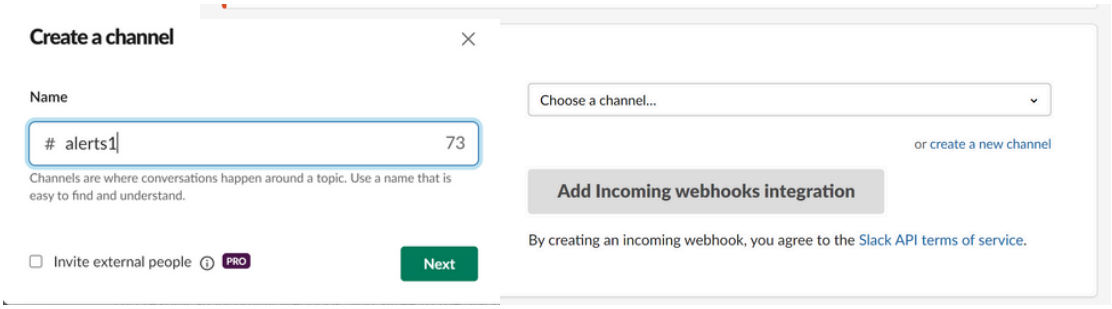
**Configuring Grafana Alerts for Slack**

1. Create a Slack Webhook

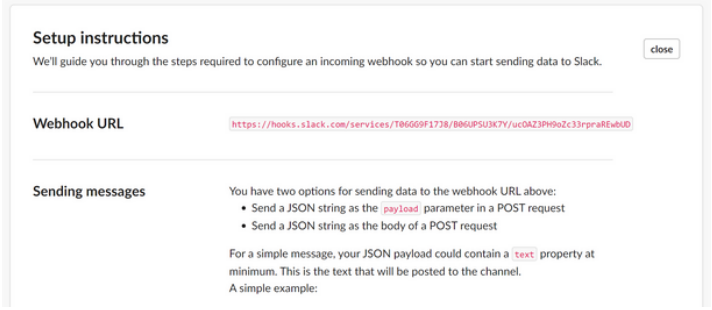
- Go to your Slack workspace and set up an incoming webhook:
  - Visit https://<your-workspace>.slack.com/apps.
  - Find Incoming Webhooks and choose to "Add Configuration".




  - Select the Slack channel for message delivery.

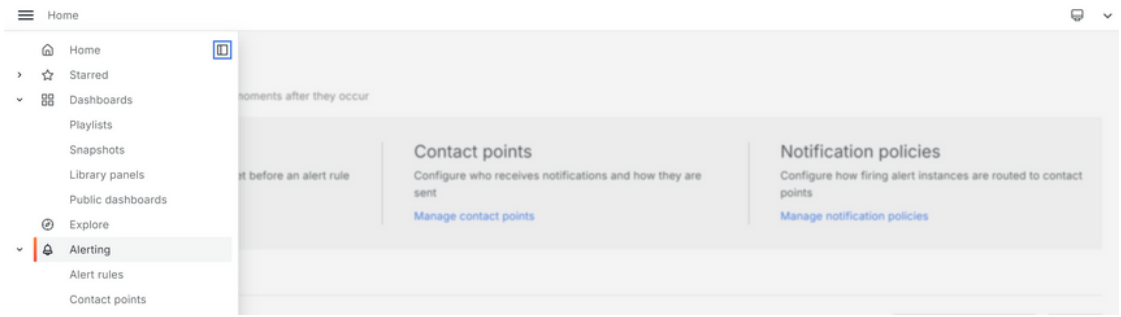


- Copy the provided Webhook URL.

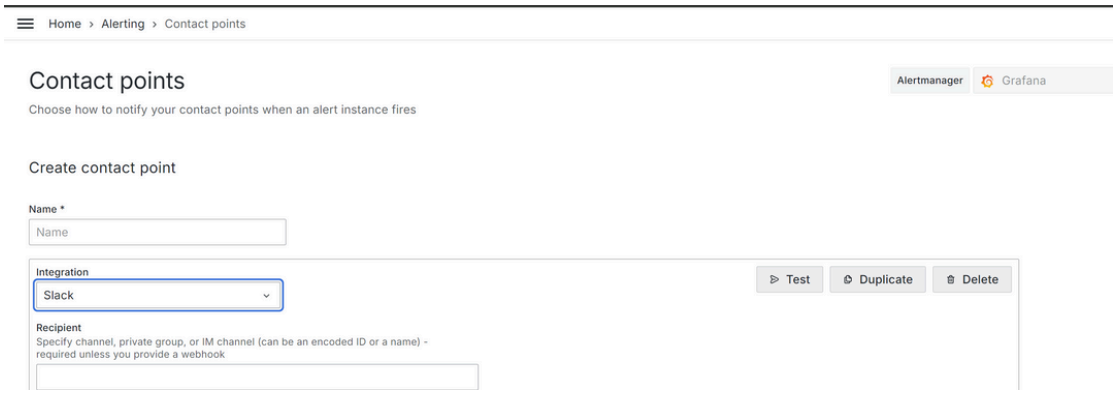



2. Add Slack Notification Channel in Grafana

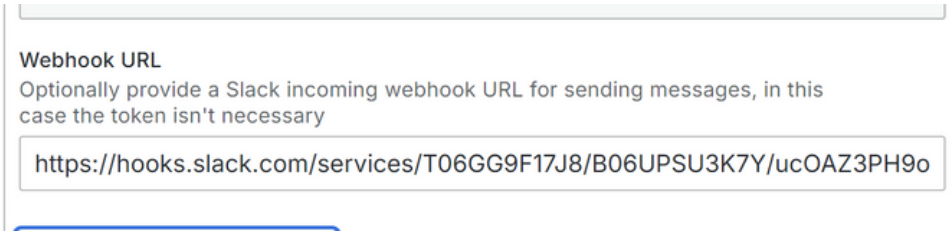
- As an administrator, log into Grafana.
- Navigate to "Alerting" and select "Manage Contact Points".




- Click "Add Contact Point".
- Set "Integration" to "Slack".



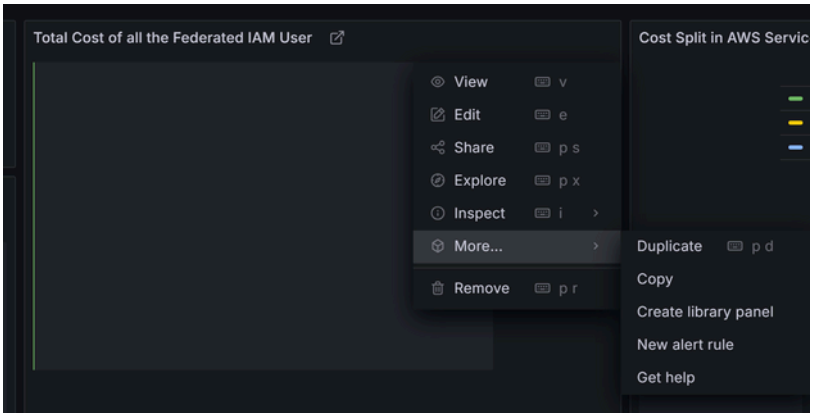

- Input the Slack Webhook URL.




- Assign a name to the Contact point and change the optional Slack Settings as per requirement. .
- Confirm by saving the contact point.

3. Configure Alert Rules

- Access the Grafana dashboard and pick the panel for alert setup.
- Click on the panel title and click on the three dots.
- Go to “More” and select “New Alert Rule”.




- Define a new alert rule with your specified conditions.
- In the "Notifications" section, link the alert to your Slack notification channel.
- Save the panel with the alert configurations.

4. Test Your Notification

- Trigger a test alert or wait for the alert condition to occur.
- Verify that the alert message is received in the designated Slack channel.

Automating Alert Notifications

To have Grafana monitor and notify automatically:

- Open the alert rule's settings.
- Set the "Evaluate every" field to determine the evaluation frequency.
- Check that the "Notifications" section is set up with the correct channel.

Conclusion:

Monitor the effectiveness of these alerts and adjust the conditions as needed to balance awareness and notification frequency.

## 10. Contacting Support

Should you require additional assistance or if you run into any ongoing issues, do not hesitate to reach out to our support team at 104481900@student.swin.edu.au. In any correspondence to ensure that our dedicated team can provide you with tailored assistance and address any questions or concerns you might have with precision.