**ECS-Metering Solution Deployment**

Table of Contents

[1. Introduction 1](#_Toc71559943)

[2. Solution Architecture 2](#_Toc71559944)

[3. Prerequisites 5](#_Toc71559945)

[4. Steps to deploy 5](#_Toc71559946)

[5. Limitations 8](#_Toc71559947)

# Introduction

AWS provides a wide list of cloud components as services, some of these services are physical allocations (EC2, RDS, etc..) and others are logical groupings (such as ECS).

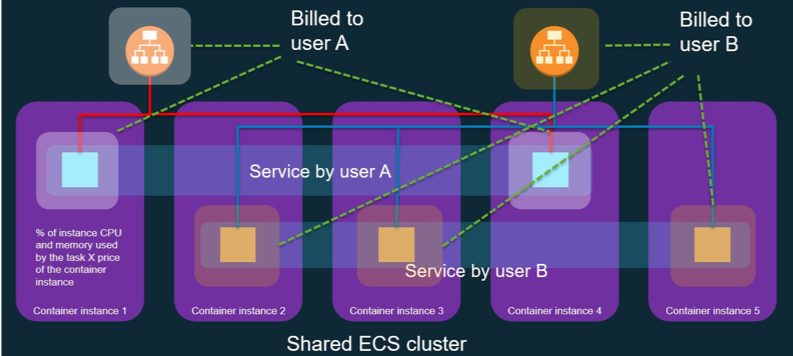
Unfortunately, in the case of ECS services, AWS doesn’t provide any cost details on services or the tasks running in an ECS service (even using “Cost Explorer”).

**The ultimate goal of this sidecar solution is to provide ECS service cost (based on resource consumption) to the LeanIX CI workspace.**

AWS propose a set of alternative solutions to measure ECS services chargeback.

([Measuring service chargeback in Amazon ECS](https://aws.amazon.com/blogs/compute/measuring-service-chargeback-in-amazon-ecs/))

This document detail the second solution in the aforementioned article (B. Billing based on resource usage at the task level.)



# Solution Architecture

An ECS environment composed of an ECS cluster with multiple ECS services and each service have multiple ECS tasks running on different ECS instances. Each ECS instance could be running one or more ECS task.

ECS Task

ECS Task

Task Running

Task Stopped

Running on

**ECS Cluster**

ECS service

ECS service

ECS service

ECS Task

ECS Task

ECS Task

ECS Task

ECS Task

ECS Task

**Step 1:**

DynamoDB

Task runs/stops

Inserting records

Running in the background

Lambda Function

(ecs-metering)

Each time a task starts or stops the ecs-metering lambda function is triggered.

This function will update a table in DynamoDB with this information, keeping track of the memory, vCPU and for how long this task was running on which ECS instance.

Request list of records

Runs daily

Leanix Workspace

**ECS Cluster**

Request list services and tasks

Compute cost

&

Export Data

DynamoDB

Lambda Function

(compute-ecs-cost)

**Step 2:**

The “compute-ecs-cost” lambda function is triggered on daily basis.

To compute the cost of ECS services this function will:

1. Self configurate (region, cluster, etc..)
2. Request list of services and tasks from AWS
3. Request list of records that correspond to the services and tasks
4. Compute the cost of each service
5. Create an LDIF
6. Upload to an S3 bucket (for history and trail)
7. Export data to LeanIX workspace Business Context

# Prerequisites

* Git
* Terraform
* AWS credentials
* AWS console
* Tag all relevant ECS services with the Business Context tag “**info:componentId**”  
  (e.g. Key: **info:componentId**, value: 5ce0fa9f-698e-4622-bccb-b887cbad23ad)

# Steps to deploy

This solution has been implemented and could be cloned from the following repository:

[ecs-metering](https://github.com/vg-leanix/ecs-metering)

* 1. Start by cloning the repository to a local folder.  
     git clone [git@github.com:vg-leanix/ecs-metering.git](mailto:git@github.com:vg-leanix/ecs-metering.git)
  2. Make sure you have AWS console installed and configured.
  3. Navigate to the cloned repository and run the initialization command for Terraform to recognize all the modules within:  
     terraform init
  4. (Optional) Run the Terraform planning command to verify which components will be created:  
     terraform plan
  5. To create all components, you can run Terraform execution command:  
     terraform apply  
     This will prompt a question on which region you’d like to deploy the script  
     Text

     Description automatically generated  
     Before execution you will be presented with the list of changes that will be made:  
     Text

     Description automatically generated  
     finally, you will get the list of changes:  
     Text

     Description automatically generated

Last thing rest to do is to configure the environment:

* In your AWS environment, navigate to “Secrets”  
  Graphical user interface, application

  Description automatically generated
* Access the secret “leanixdatasecret” and edit the values by clicking on “Retrieve Secret Value”  
  Graphical user interface, application, Teams

  Description automatically generated  
    
  Enter the secret value:  
  - businesscontext: Is the tag used to create the Business Contexts in the CI workspace (in this case: **info:componentId**)  
  - token: Is a token generated from the LeanIX CI workspace  
  - host: Is the host of the LeanIX CI workspace (in this case: libertexgroup.leanix.net)  
  Graphical user interface, application

  Description automatically generated with medium confidence

Reminder that the ecs-metering function will be triggered first to populate the database and keep track of any changes, and the compute-ecs-cost function will be triggered daily to compute ECS services cost and export the data to LeanIX.

N.B: In the case of any problem, you can destroy all the component previously created using Terraform by running the following command:

terraform destroy

Running this command will only destroy components that are managed by terraform and exist in the script.

**Nevertheless, running this command will cause an error!**  
Error deleting CloudWatch Event Rule: ValidationException: Rule can't be deleted since it has targets.

status code: 400  
Due to limitation from Terraform, it will try to delete the BridgeWatch Rule before deleting the target Lambda Function.  
To solve this problem, in your AWS environment, navigate to EventBridge and delete both rules: ComputeECSCostDaily and ECSTaskStatusRuleGraphical user interface, text, application, email

Description automatically generated

After that it is safe to run terraform destroy and terraform apply again.

# Limitations

1. Idle time is not accounted for 🡪 Hence potentially not all EC2 costs captured (see figure below)
2. Pricing based on pricing list (EC2); May vary from actual cost

Chart, treemap chart

Description automatically generated

# Integration API Processors:

In your LeanIX CI workspace, please add one Integration API processor like the following:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Connector Type** | **Connector Id** | **Connector Version** | **Processing Direction** | **Processing Mode** |
| leanix-custom | ecs-cost-distribution | 1.0.0 | Inbound | partial |

Append the following processor function to the newly created processor:

{

    "processors": [

        {

            "processorType": "inboundMetrics",

            "processorDescription": "Store the tag-level costs for all tags configured as a source for the business context for all billing entries with dimension TAG read from the master account's LDIF in the measurement 'Cloud Costs (Business Context)'",

            "filter": {},

            "updates": [

                {

                    "key": {

                        "expr": "measurement"

                    },

                    "values": [

                        {

                            "expr": "Cloud Costs (Business Context)"

                        }

                    ]

                },

                {

                    "key": {

                        "expr": "time"

                    },

                    "values": [

                        {

                            "expr": "${content.data.datetime}.000000Z"

                        }

                    ]

                },

                {

                    "key": {

                        "expr": "fieldKey"

                    },

                    "values": [

                        {

                            "expr": "other"

                        }

                    ]

                },

                {

                    "key": {

                        "expr": "fieldValueNumber"

                    },

                    "values": [

                        {

                            "expr": "${content.data.totalCloudCostsYesterday}"

                        }

                    ]

                },

                {

                    "key": {

                        "expr": "tags"

                    },

                    "values": [

                        {

                            "map": [

                                {

                                    "key": "key",

                                    "value": "External ID (Business Context)"

                                },

                                {

                                    "key": "value",

                                    "value": "${content.data.application}"

                                }

                            ]

                        },

                        {

                            "map": [

                                {

                                    "key": "key",

                                    "value": "Connector Instance"

                                },

                                {

                                    "key": "value",

                                    "value": "${content.data.serviceId}"

                                }

                            ]

                        }

                    ]

                }

            ],

            "enabled": true

        }

    ]

}