Day 11 Internship Report

Contact Manager Application (Alerting)

Intern Name: Arsalan Sharief

Company: Signiance Technology

Role: DevOps Intern

Day: 11

Repository Link:

https://github.com/arsalan-signiance/internship-day10-11-12

(Pending, will be available on 13-02-2026)

Objective of the Day

To design and begin implementing a scalable and cost-efficient alerting system using Amazon Managed Service for Prometheus (AMP) integrated with the ECS-based backend.

1. Application Metrics Integration

Before configuring AMP, I prioritized preparing the backend to expose Prometheus-compatible metrics.

Added Prometheus Metrics to Backend

- Integrated `prometheus_flask_exporter` into the Flask application.
- Enabled automatic metrics collection.
- Verified `/metrics` endpoint locally.
- Confirmed default Python, process, and HTTP request metrics were being generated.

Rebuilt & Redeployed ECS Service

- Rebuilt the Docker image locally.
- Pushed the updated image to Docker Hub.
- Updated ECS task definition.
- Redeployed the ECS service.

Verification

Accessed ALB endpoint to verify live metrics:

```
http://contact-manager-alb-1595083403.us-east-1.elb.amazonaws.com/metrics
```

Successfully verified live Prometheus metrics output from ECS.

2. Research & Learning – Amazon Managed Prometheus (AMP)

I dedicated time to understanding the fundamentals of AWS managed observability services.

Key Concepts Learned:

- Prometheus architecture fundamentals
- Remote write mechanism
- Amazon Managed Service for Prometheus (AMP)
- AMP alerting flow & SNS integration

Resources Used:

- AWS Official Documentation
- YouTube tutorials (ECS + AMP integrations)

3. Alerting Strategy Decision

After evaluating three distinct alerting approaches:

1. Self-managed Prometheus + Alertmanager on EC2
2. Amazon Managed Prometheus (AMP) + AMP Alerting + SNS (Selected)
3. CloudWatch Alarms + SNS

Reasoning for Selection

Scalability & Production Readiness

AMP removes the operational overhead of maintaining monitoring infrastructure, making it more scalable than self-managed EC2 solutions.

No Infrastructure Overhead

Eliminated the need to handle EC2 provisioning, storage management, backups, and upgrades associated with self-hosted Prometheus.

Future Expansion

The project may evolve into microservices; AMP is optimized for dynamic, cloud-native workloads and auto-scaling environments.

DevOps Learning Goals

Selected to gain hands-on experience with AWS managed observability services, complementing my previous experience with CloudWatch and self-hosted tools.

Cost Consideration

Avoids the cost of running a dedicated EC2 instance 24/7 solely for monitoring purposes.

4. Infrastructure Configured Today

I successfully provisioned the necessary AWS resources for the alerting pipeline:

- Created AMP Workspace (us-east-1): Verified ARN and region alignment.
- Created SNS Topic for Alerts: Name: `c-m-alerts` | Region: `us-east-1`.
- Configured Email Subscription: Added email endpoint and confirmed subscription.
- Updated SNS Access Policy: Modified policy to allow `aps.amazonaws.com` to publish to the topic.

5. Current Monitoring Status

Backend exposes /metrics

Metrics verified on ECS

AMP workspace created

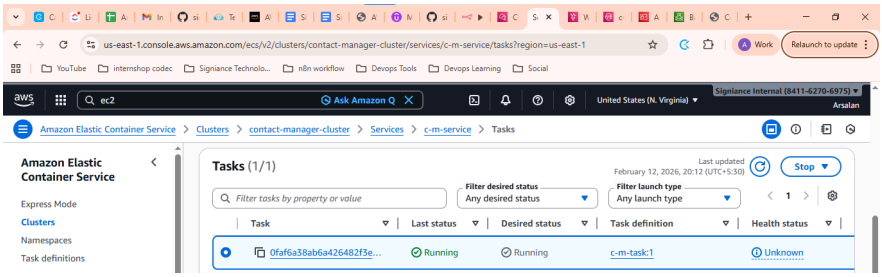
SNS topic configured

Email subscription confirmed

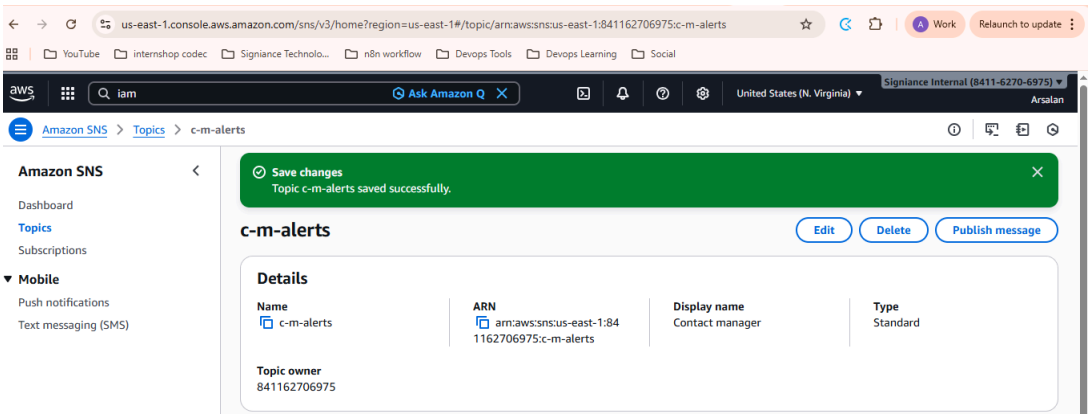
SNS access policy updated

Metrics ingestion (ECS → AMP)

6. Screenshots

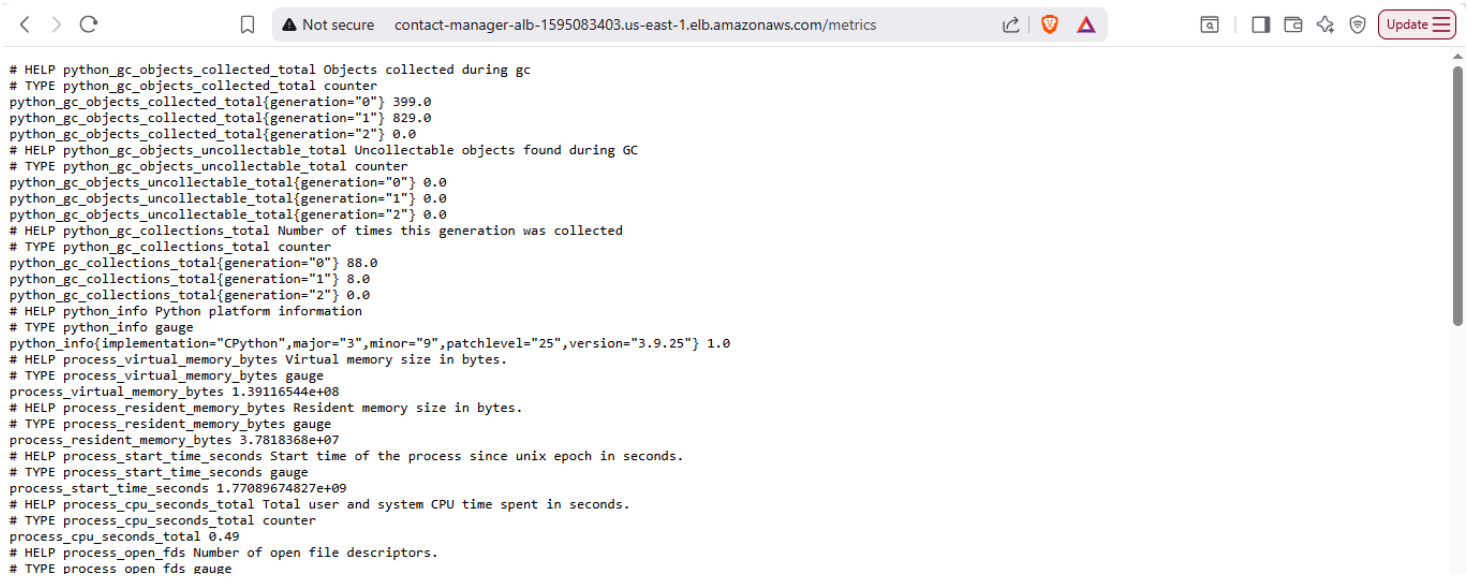


ECS Screenshot: Amazon ECS showing a running task (`c-m-service`) in the `contact-manager-cluster` with health status marked as unknown.



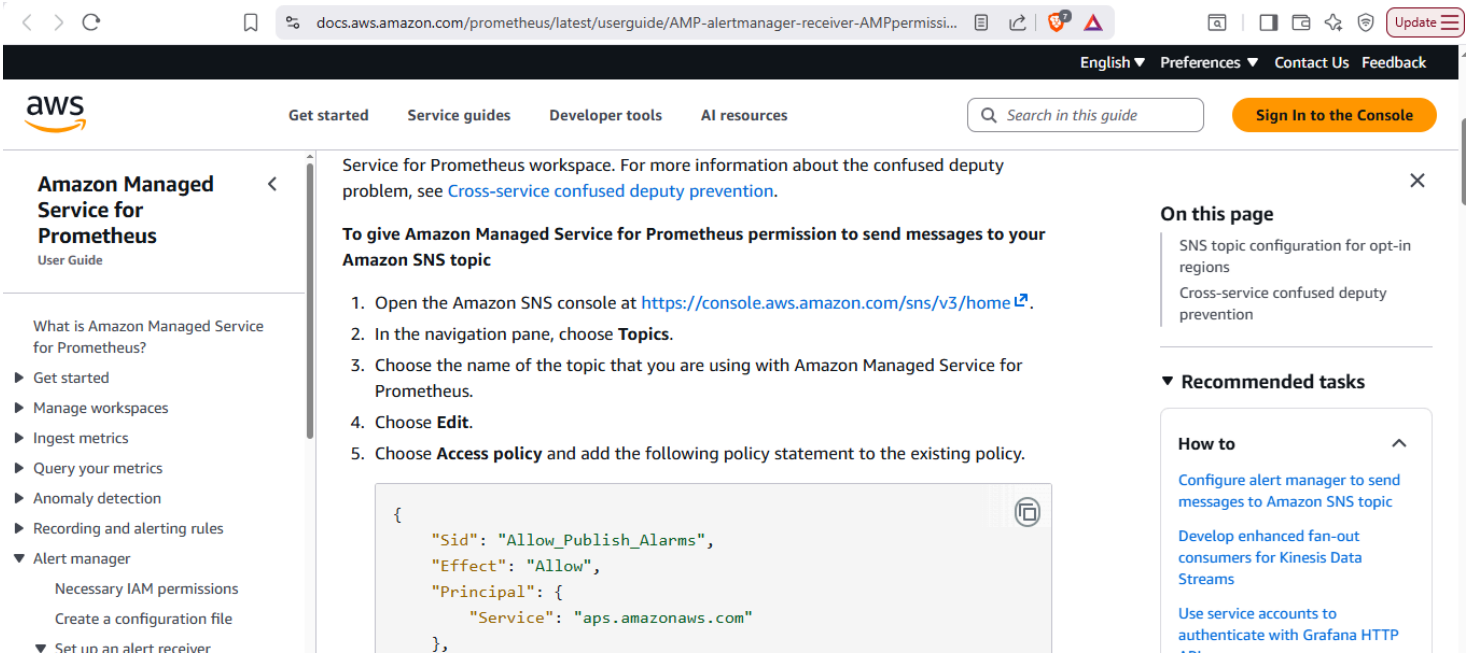
SNS Topic

Amazon SNS console displaying the `c-m-alerts` topic successfully saved, configured as a Standard type for Contact Manager alerts.



Metrics Endpoint Screenshot

Prometheus-style `/metrics` endpoint output from the contact-manager ALB, exposing Python GC, memory, CPU, and process metrics.



AWS Documentation Screenshot: AWS documentation page that I used as a reference to configure permissions for Amazon Managed Service for Prometheus to publish alerts to an SNS topic.

Prepared by: ArsalanSharief
DevOps Intern – Signiance Technology