**Monitoring System Performance and Responding to Alerts in AWS**

**Combined SOP and User Guide**

**22 January 2025**

**RECORD OF CHANGES**

|  |  |  |
| --- | --- | --- |
| Date | Description of Change(s) | Author |
| 01/21/2025 | Initial Draft | Jimmie Marshall |
|  | Final revision version |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[1.0 Introduction 4](#_Toc188435561)

[2.0 Purpose 4](#_Toc188435562)

[3.0 Scope 4](#_Toc188435563)

[4.0 Roles and Responsibilities 4](#_Toc188435564)

[5.0 Understanding AWS Storage Classes 5](#_Toc188435565)

[5.1 Amazon S3 Storage Classes 5](#_Toc188435566)

[5.2 Lifecycle Policies 6](#_Toc188435567)

[6.0 Standard Operating Procedures 6](#_Toc188435568)

[6.1 Data Classification and Analysis 6](#_Toc188435569)

[6.3 Utilizing Compression and Deduplication 7](#_Toc188435570)

[6.4 Optimizing Backup Frequency and Types 7](#_Toc188435571)

[6.5 Regular Auditing and Cleanup 8](#_Toc188435572)

[5.6 Selective Cross-Region Replication 8](#_Toc188435573)

[6.7 Monitoring and Alerts 8](#_Toc188435574)

[6.8 Employing S3 Intelligent-Tiering 9](#_Toc188435575)

[6.9 Rightsizing EBS Volumes and Using Data Lifecycle Manager 9](#_Toc188435576)

[6.10 Encryption Key Management 10](#_Toc188435577)

[6.11 Regular Review and Strategy Updates 10](#_Toc188435578)

[7.0 Additional Guidance 10](#_Toc188435579)

[7.1 Training and Awareness 10](#_Toc188435580)

[7.2 Documentation and Reporting 10](#_Toc188435581)

[7.3 Compliance and Security 10](#_Toc188435582)

[8. Conclusion 11](#_Toc188435583)

[Appendices 11](#_Toc188435584)

# 

## 1.0 Introduction

Maintaining optimal system performance in AWS is critical for ensuring application availability, reliability, and user satisfaction. This combined SOP and User Guide aims to equip your team with the necessary procedures and knowledge to monitor system performance effectively and respond to alerts promptly and appropriately.

## 2.0 Purpose

To establish standardized procedures and provide guidance for monitoring AWS system performance and responding to alerts, thereby ensuring system reliability, quick issue resolution, and adherence to service-level agreements (SLAs).

## 3.0 Scope

This guide applies to all team members responsible for system performance monitoring and incident response within the AWS environment, including:

* System Administrators
* DevOps Engineers
* IT Operations Teams
* Site Reliability Engineers (SREs)

## 4.0 Roles and Responsibilities

* **Monitoring Administrator**
  + **Responsibilities:**
* Configure and maintain monitoring tools and dashboards.
* Define performance metrics and thresholds.
* Ensure monitoring systems are operational and up-to-date.

## Actions:

* Set up CloudWatch alarms and notifications.
* Update monitoring configurations as systems evolve.
* **Incident Response Team**
  + **Responsibilities**:
    - Respond to alerts and incidents promptly.
    - Diagnose and resolve performance issues.
    - Document incidents and resolutions.
  + **Actions**:
  + Acknowledge alerts.
  + Follow escalation procedures if necessary.
  + Communicate with stakeholders during incidents.
* **IT Operations Manager**
* **Responsibilities**:
  + Oversee the monitoring and incident response processes.
  + Ensure adherence to SLAs and policies.
* **Actions**:
  + Review incident reports.
  + Facilitate continuous improvement initiatives.

## 5.0 Understanding AWS Monitoring Services

# 5.1 Amazon CloudWatch

Amazon CloudWatch is a monitoring and observability service that provides data and actionable insights for AWS, hybrid, and on-premises applications and infrastructure resources.

* **Features:**
* Collects monitoring and operational data in the form of logs, metrics, and events.
* Provides real-time insights into resource utilization, application performance, and operational health.

# 5.2 AWS CloudTrail

AWS CloudTrail records AWS API calls and events for your account, providing a complete history of user activity and API calls for auditing and troubleshooting purposes.

# 5.3 AWS Systems Manager

AWS Systems Manager provides a unified user interface to view operational data from multiple AWS services and allows you to automate operational tasks across your AWS resources.

# 5.4 Amazon SNS (Simple Notification Service)

Amazon SNS is a fully managed messaging service for both application-to-application (A2A) and application-to-person (A2P) communication, used here for sending alert notifications.

## 6.0 Standard Operating Procedures

# 6.1 Setting Up Monitoring Metrics

**Procedure**:

1. **Identify Key Performance Indicators (KPIs)**
   * **Action:** Determine which metrics are critical for your application's performance and reliability.
   * **Common Metrics:**
     + **Compute:** CPU utilization, memory usage, disk I/O.
     + **Networking:** Network in/out, latency.
     + **Storage:** Disk space utilization, read/write operations.
     + **Application:** Error rates, response times, throughput.

**Explanation:**

Identifying the right KPIs ensures that you monitor what's most important for your application's health and performance.

1. **Enable Detailed Monitoring**
   * **Action:** For EC2 instances and other services, enable detailed monitoring to get data at a 1-minute granularity.
   * **Steps:**
     + Navigate to the EC2 Dashboard.
     + Select the instance(s).
     + Click on **Actions > Monitor and troubleshoot > Modify monitoring**.
     + Choose **Enable detailed monitoring**.

**Explanation:**

Detailed monitoring provides more granular data, allowing for better detection of performance issues.

1. **Collect Custom Metrics**
   * **Action:** Use the CloudWatch agent and AWS SDKs to collect application-specific or system-level custom metrics.
   * **Implementation:**
     + Install and configure the CloudWatch agent on your instances.
     + Define custom metrics in the agent configuration file.

**Explanation:**

Custom metrics allow you to monitor aspects that are not covered by default, offering deeper insights into your system's performance.

# 6.2 Configuring CloudWatch Alarms

**Procedure**:

1. **Define Thresholds for Alerts**

* **Action:** Set threshold values for each metric that, when crossed, should trigger an alarm.
* **Example:**
* CPU Utilization > 80% for 5 consecutive minutes.
* Memory Usage > 75%.
* Disk Space Utilization > 85%.

**Explanation:**

Proper thresholds help in early detection of potential issues before they impact users.

1. **Create CloudWatch Alarms**

* **Steps:**
* Go to the **CloudWatch Console.**
* Click on **Alarms > All alarms > Create alarm.**
* Select the metric and define the threshold and period.
* Configure notification options.

**Explanation:**

CloudWatch alarms trigger notifications or automated actions when thresholds are breached, enabling quick response.

1. **Set Up Notifications**

* **Action:** Use Amazon SNS to send notifications to email, SMS, or other endpoints.
* **Steps:**
* **Create an SNS Topic:**
* Navigate to the SNS console.
* Click on **Topics > Create topic**.
* **Subscribe Team Members:**
* Add email addresses or phone numbers as subscribers.
* **Configure Alarms to Use SNS:**
* In the alarm creation, specify the SNS topic under Notification.

**Explanation:**

Notifications ensure that the relevant personnel are alerted promptly to take corrective action.

# 6.3 Implementing Dashboards for Real-Time Monitoring

**Procedure:**

1. **Create CloudWatch Dashboards**

* **Action:** Build custom dashboards to visualize key metrics in one place.
* **Steps:**
* In the CloudWatch console, select Dashboards > Create dashboard.
* Choose a name for the dashboard.
* Add widgets for relevant metrics by selecting Add widget.

**Explanation:**

Dashboards provide at-a-glance visibility into system performance, aiding in quick assessment.

# Customize Dashboard Views

* **Action:** Organize widgets logically, group related metrics, and use appropriate visualization types (line charts, gauges).
* **Examples:**
* Group compute metrics together.
* Use heatmaps for latency visualization.
* Color-code widgets based on severity levels.

**Explanation:**

Well-organized dashboards enhance readability and make it easier to spot anomalies.

# 6.4 Responding to Alerts and Incidents

**Procedure:**

1. **Alert Acknowledgement**

* **Action:** Upon receiving an alert, acknowledge it to prevent duplicate efforts.
* **Steps:**
* Update the incident management tool (e.g., Jira, ServiceNow) with the alert details.
* Assign the incident to a team member or yourself.

**Explanation:**

Acknowledging alerts ensures that incidents are tracked and managed efficiently.

1. **Incident Assessment**

* Determine the severity and potential impact of the issue.
* **Considerations:**
* Affected systems or services.
* Number of users impacted.
* SLA commitments.

**Explanation:**

Assessing the impact helps prioritize response efforts and resources.

1. **Troubleshooting**

* **Steps:**
* Review relevant metrics and logs in CloudWatch and CloudTrail.
* Identify any recent changes that could have caused the issue.
* Use AWS Systems Manager to run commands or scripts if needed.
* Check application logs for errors or exceptions.

**Explanation:**

Systematic troubleshooting aids in identifying root causes quickly.

1. **Implementing a Fix**

* Apply the necessary fix to resolve the issue.
* **Examples:**
* **Resource Scaling:** Increase instance size or add instances to handle load.
* **Configuration Changes:** Update misconfigured settings.
* **Service Restarts: Restart** services or instances.
* **Rollback Deployments:** Revert to a previous stable version.

**Explanation:**

Promptly addressing the issue minimizes downtime and user impact.

1. **Communication**

* **Action:** Keep stakeholders informed about the incident status.
* **Steps:**
* Send updates via established communication channels (e.g., email, Slack).
* Provide estimated resolution times if possible.
* Notify users if the incident affects them directly.

**Explanation:**

Transparent communication maintains trust and ensures alignment among teams.

1. **Documenting the Incident**

* **Action:** Record the incident details, actions taken, and resolution outcome.
* **Include:**
* Time of occurrence and resolution.
* Root cause analysis.
* Steps for preventing recurrence.
* Impact assessment.

**Explanation:**

Documentation enables learning from incidents and improving future responses.

# Escalation Procedures

**Procedure:**

* + - 1. **Determine Escalation Criteria**
* **Action:** Define when an incident should be escalated based on severity and impact.
* **Criteria:**
* **Severity Levels:**
* **Critical:** Complete outage or severe degradation.
* **High:** Significant impact on functionality.
* **Medium:** Moderate impact with workaround available.
* **Low:** Minor issues with little to no user impact.
* **Duration:** If issue remains unresolved beyond a certain timeframe.

**Explanation:**

Clear criteria ensure critical issues receive the appropriate attention quickly.

# Escalate to Senior Engineers or Management

* **Steps:**
* Notify designated senior personnel as per the escalation matrix.
* Provide detailed information about the incident, including impact and actions taken.

**Explanation:**

Escalation brings in additional expertise to resolve complex issues.

# Engage AWS Support if Necessary

* **Action:** Open a support case with AWS if the issue involves AWS infrastructure.
* **Steps:**
* Log in to AWS Support Center.
* Click on Create case.
* Select the Service and Category related to the issue.
* Provide a detailed description and urgency level.
* Set the contact method (email, phone).

**Explanation:**

AWS Support can assist with underlying platform issues beyond your control.

# 5.6 Post-Incident Review

**Procedure:**

1. **Conduct a Post-Mortem**

* **Action:** Hold a meeting with involved team members to discuss the incident.
* **Agenda:**
* **Timeline Review:** Chronological account of events.
* **Root Cause Analysis:** Identify the underlying cause.
* **Lessons Learned:** What worked well, what didn't.
* **Action Items:** Steps to prevent recurrence.

**Explanation:**

Post-mortems facilitate learning and process enhancements to prevent future incidents.

# Update Documentation

* **Action:** Revise SOPs, runbooks, and documentation based on insights gained.
* **Examples:**
* Adjust alert thresholds if they were too sensitive or not sensitive enough.
* Update troubleshooting guides with new scenarios.
* Incorporate new steps in response procedures.

**Explanation:**

Keeping documentation current ensures that future responses are more effective.

# 6.7 Continuous Improvement

**Procedure:**

1. **Regularly Review Monitoring Systems**

* **Action:** Assess the effectiveness of current monitoring and alerts.
* **Frequency:** Monthly or after significant changes to the system.

**Explanation:**

Regular reviews help adapt the monitoring system to evolving application needs.

# Automate Responses Where Possible

* **Action:** Implement automated actions in response to specific alarms (e.g., auto-scaling).
* **Implementation:**
* Use CloudWatch alarm actions to trigger AWS Lambda functions or auto-scaling policies.
* Set up AWS Systems Manager Automation documents.

**Explanation:**

Automation reduces response times and minimizes human intervention for routine **issues.**

# Stay Updated with AWS Services

* **Action:** Keep abreast of new AWS features and services that can enhance monitoring and alerting.
* **Resources:**
* **AWS What's New:** Latest updates on AWS services.
* **AWS Blogs and Webinars:** In-depth articles and presentations.
* **AWS re:Invent Sessions:** Annual conference sessions available online.

**Explanation:**

Leveraging new tools and features can improve efficiency and capabilities.

## 7.0 Additional Guidance

# 7.1 Security and Compliance

* **Access Control:**
* Ensure that only authorized personnel have access to monitoring and alerting configurations.
* Use IAM roles and policies to enforce the principle of least privilege.
* **Data Privacy:**
* Be mindful of sensitive data in logs and metrics.
* Implement encryption and secure storage practices for logs (e.g., encrypt logs at rest using KMS keys).
* **Compliance Standards:**
* Adhere to industry-specific regulations (e.g., HIPAA, GDPR) when handling monitoring data.
* Regularly review compliance requirements and update procedures accordingly.

# 7.2 Training and Awareness

* **Team Training:**
* Conduct regular training sessions on monitoring tools and incident response procedures.
* Encourage knowledge sharing and cross-training among team members.
* **Simulated Drills:**
* Perform simulated incident response exercises to test and improve readiness.
* Analyze drill outcomes to refine procedures.

# 7.3 Tools Integration

* **Third-Party Monitoring Tools:**
* Integrate tools like Datadog, New Relic, or Prometheus if they provide additional insights.
* Ensure that integrations comply with security policies.
* **Communication Platforms:**
* Integrate alert notifications with platforms like Slack, Microsoft Teams, or PagerDuty for efficient communication.

## 8. Conclusion

By following this SOP and understanding the underlying processes, your team can effectively monitor AWS system performance and respond to alerts swiftly, ensuring high availability and reliability of your applications. Continuous improvement and proactive management are key to maintaining optimal operations.

# Appendices

**A. Useful AWS Services and Tools**

* **AWS CloudWatch**
  + **Documentation:** [**CloudWatch Documentation**](https://docs.aws.amazon.com/cloudwatch/)
  + **Features:** Metrics, alarms, dashboards, logs.
* **AWS CloudTrail**
  + **Documentation:** [**CloudTrail Documentation**](https://docs.aws.amazon.com/cloudtrail/)
  + **Uses:** Auditing, compliance, and troubleshooting AWS account activity.
* **AWS Systems Manager**
  + **Documentation:** [**Systems Manager Documentation**](https://docs.aws.amazon.com/systems-manager/)
  + **Functions:** Automation, Run Command, Parameter Store.
* **Amazon SNS**
  + **Documentation:** [**SNS Documentation**](https://docs.aws.amazon.com/sns/)
  + **Uses:** Sending notifications to various endpoints.
* **AWS Trusted Advisor**
  + **Features: Provides best practice recommendations.**
  + **Documentation:** [**Trusted Advisor Documentation**](https://docs.aws.amazon.com/general/latest/gr/Welcome.html)

**B. Sample Commands and Scripts**

* **Install CloudWatch Agent on Linux**

wget https://s3.amazonaws.com/amazoncloudwatch-agent/amazon\_linux/amd64/latest/amazon-cloudwatch-agent.rpm

sudo rpm -Uvh amazon-cloudwatch-agent.rpm

* **Configure CloudWatch Agent**

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard

* **Starting the CloudWatch Agent**

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl \

-a start \

-m ec2 \

-c file:/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.json \

-s

* **Creating a CloudWatch Alarm via AWS CLI**

aws cloudwatch put-metric-alarm \

--alarm-name "High\_CPU\_Utilization" \

--alarm-description "Alarm when CPU exceeds 80%" \

--metric-name CPUUtilization \

--namespace AWS/EC2 \

--statistic Average \

--period 300 \

--threshold 80 \

--comparison-operator GreaterThanThreshold \

--dimensions Name=InstanceId,Value=i-1234567890abcdef0 \

--evaluation-periods 2 \

--alarm-actions arn:aws:sns:us-east-1:123456789012:OperationsAlerts

**Next Steps**

* **Implementation Plan:**
  + **Assign Responsibilities:** Allocate tasks to team members for setting up monitoring and incident response.
  + **Set Timelines:** Establish deadlines for initial setup and routine maintenance.
  + **Resource Allocation:** Ensure necessary tools and permissions are in place.
* **Monitoring and Review:**
  + **Regular Meetings:** Schedule weekly or bi-weekly meetings to review system performance and incidents.
  + **Metrics Evaluation:** Continuously assess the relevance and effectiveness of monitored metrics.
* **Feedback Loop:**
  + **Encourage Continuous Improvement:** Invite team members to suggest enhancements.
  + **Implement Suggestions:** Act on viable recommendations promptly.