Day 3 Management Tools Week at the AWS Pop-up Loft

Optional EMR with Service Catalog - Lab

- EMR with Service Catalog

Logging and Monitoring - Lab Guide

Introduction

There are two goals of monitoring: - Achieve situational awareness to provide timely and effective responses and - Gain insights for the business, development, and operations that enable proactive courses of action.

In this workshop, we take you through the process of developing and implementing a workload monitoring strategy to achieve these objectives. You will learn how to:

1. Build a monitoring plan
2. Implement the plan using AWS Services:
    - Configure telemetry sources using AWS Systems Manager Run Command and Parameter Store
    - Collect telemetry using Amazon CloudWatch Logs
    - Setup metrics and thresholds using Amazon CloudWatch Log-Filter Metrics
    - Configure alarms to fire when metric thresholds are crossed using Amazon CloudWatch Alarms
    - Setup notifications topics and subscriptions using Amazon Simple Notification Service

- Configure automated responses using AWS Lambda

This lab will demonstrate how to alert on major categories of events, monitor for operational outcomes, trigger responses, and deliver insights.

This is an advanced lab featuring exercises on developing a monitoring plan, and implementing that plan using CloudWatch. It is best to have some familiarity with AWS and specifically CloudWatch before attempting the lab. We will dive deeper into CloudWatch than an introductory session and do hands on configuration in the console. Operational Excellence concepts

Operational Excellence is one of the 5 pillars of the Well-Architected program. Well-Architected was created to provide AWS customers with best practices and guidance on how to operate and design applications to operate in the cloud. You can find out more about the Well-Architected program at this link and the Operational Excellence pillar at this link

In this lab you will apply the concepts of Infrastructure as Code and Operations as Code to the following activities:

1. Deployment of Infrastructure
2. Performing operations activities
3. Event Management and Incident Response

Included in the lab guide are bonus sections that can be completed if you have time, or later if interested.

Note: You will be billed for any applicable AWS resources used in this lab that are not covered in the AWS Free Tier. *AWS Free Tier*

Lab Requirements

You will need the following to be able to perform this lab:

- Your own device for console access

- A personal (not shared) AWS account that you are able to use for testing, that is not used for production or other purposes
- The lab must be performed logged into a user with Administrator permissions and an existing EC2 Key Pair
- An available VPC in the Ireland region EU-West-1

NOTE: You will be billed for any applicable AWS resources used if you complete this lab that are not covered in the AWS Free Tier. *AWS Free Tier*

Exercise 1: Lab Setup

1.1 Deploy the Lab Infrastructure using CloudFormation

There are two options for launching the CloudFormation Stack. The first option will automatically populate your CloudFormation stack variables. The second option will walk through all the steps involved in manually creating a new stack from a template URL.

Option A: Automatically launch and populate stack creation variables

Deploy Lab by clicking here

NOTE: The automatic launch process will only work if you are using the new version of the CloudFormation console.

If the button fails to take you to the create new stack page, try switching to the new CloudFormation console by clicking the New Console link in the CloudFormation drop-down menu.

Option B: Manually walk through the stack creation from a template URL

1. **Copy the CloudFormation script URL** for this lab: **NOTE:** The instructions below are based on the new version of the CloudFormation console.If the button fails to take you to the create new stack page, try switching to the new CloudFormation console

by clicking the New Console link in the CloudFormation drop-down menu.

2. Use your administrator account to access the CloudFormation console at https://console.aws.amazon.com/cloudformation/.
3. Click the **Create Stack** button.
    - On the **Specify Template** screen:
        - Ensure the **Template is ready** radio button is selected under the Prerequisite - Prepare template section.
        - Ensure that the **Amazon S3 URL** radio button is selected under the Specify template section and then paste the copied URL into the Amazon S3 URL textbox. **NOTE:** A CloudFormation template is a JSON or YAML formatted text file that describes your AWS infrastructure containing both optional and required sections. In the next steps, we will provide a name for our stack and parameters that will be passed into the template to help define the resources that will be implemented.
4. On the **Specify Template** step, next to Specify an Amazon S3 template URL, click the link to View in Designer.
    - Take a moment to look at the graphical representation of the environment we are about to create, and the template in the JSON and YAML formats.
    - The template was written in YAML as a nested stack and this is a method to convert it to JSON if that is your preference. **NOTE:** AWS CloudFormation Designer is a graphic tool for creating, viewing, and modifying AWS CloudFormation templates. With Designer, you can diagram your template resources using a drag-and-drop interface, and then edit their details using the integrated JSON and YAML editor. AWS CloudFormation Designer can help you quickly see the interrelationship between a template’s resources and easily modify templates.

5. Choose the Create Stack icon (a cloud with an arrow) to return to the Specify Template step. **NOTE:** By clicking the Create Stack option in the Designer, you will automatically create a new cf-template S3 bucket in your selected region, and the original template will be copied with a unique random name and placed in your new S3 bucket.
6. On the Specify Template step, click the Next button.
7. On the Specify Details page:
    - Define your Stack name: by entering ExampleCorp
    - Provide your email address. This will only be used in the lab, and will only be used to send you SNS notifications of events.
    - Define the HTTPLocation CIDR subnet address to limit who can connect to your resources.
        - You can use to identify your IP address and restrict access to it specifically by entering the IP followed by the CIDR subnet mask of /32 (with no spaces between IP and the CIDR subnet mask).
    - Select an InstanceType. We recommend selecting the default free tier eligible t2.micro option.
    - Select your EC2 KeyPair from the KeyName pull down list.
    - Define your SSHLocation entering your IP address with CIDR subnet mask
    - Define your Workload name as ExampleCorp
8. Choose Next
9. On the Options page optionally specify additional Tags, and then scroll down and choose Next
10. On the Review page:
    - Review your selections
    - Scroll down and check the box I acknowledge that AWS CloudFormation might create IAM resources.
11. Choose Create

1.2 Confirm your subscription to the SNS Notification - After the CloudFormation script has completed, enter the email client for the address you provided earlier. - Locate and email from AWS Notifications with the subject AWS Notifications - Subscription Confirmation. - Click through the Confirm subscription link to confirm your subscription.

Note: In this lab you will interact with the web based application you just created. To generate activity logs you will need to upload pictures to your application server. You can use your own images or download sample images to use here: Sample Photos The sample image file is 76MB. If you plan to use the sample images start the download now so it can complete before you need them.

Exercise 2: Review the ExampleCorp Requirements and Build a Monitoring Plan

2.1 Review the ExampleCorp [fictional] Narrative

In the lab we will use a mock workload for a fictitious company to learn about performing operations on AWS. Details on the application used in this lab can be found on GitHub at this link.

About Us: ExampleCorp was started on a simple idea: “A picture is worth a thousand words”. We help users find out how many words that picture is worth. Our users make thousands of connections every day based on what they discover in the pictures they share. We want to make sharing pictures and identifying what lies within, faster and easier for them.

Our App: ExampleCorp has one primary application – a photo library that tags the contents of images users upload, through machine learning. Through our application users can upload pictures of themselves and friends to see which celebrities they resemble, and find celebrities in photos they upload. We have >5 million users (500,000 active daily).

Our Architecture: The ExampleCorp application is a simple Ruby on Rails app with a MySQL database. It is installed as two Docker containers (application and database) launched in a VPC on EC2 instances using CloudFormation. Users upload images and background jobs analyze them and collect metadata. These jobs use AWS Rekognition to detect labels, text and celebrities, and EXIF metadata from the camera used. Deployment of application code uses update-in-place.

Our Operating Model: The ExampleCorp team are mainly “DevOps” but there’s separation of duties between the cloud infrastructure team and the application team. Application code is pushed via CI/CD pipeline around once a day; the platform engineering team tests and deploys infrastructure changes. The operating model focuses on fast deployment of new features, and rapid feedback on user experience. Application performance and availability are vital to accelerated sign ups, which drive our funding valuation.

Our Team:

Title	Name	Requirements
CEO	Isobel Rose	focused on growing user base to secure funding, building a sustainable revenue stream.
CIO	Henry Pomfret: focused on efficiency and performance goals for IT Operations, release targets and application availability.	
Infrastructure Director	Mel Blank	Primary concern is security and stability of infrastructure.
Application Director	Joanne Groan	making sure production is delighting customers and features are the right features are being delivered to production as quickly as possible
InfoSec Director	Aki French	security threats posed by fast delivery, competing priorities between security and release timelines.
DevOps Engineer	Ryn Brandish	wants insight into platform changes, application issues, and governance requirements affecting feature delivery and application security.
Site Reliability Engineer	Sansa Bailiff	wants to stopped getting paged at all hours and wants to better understand system reliability
Security Engineer	Paco Simpson	Biggest challenge: speed to market sometimes means security issues come up only post-go-live.

2.2 Building the Monitoring Plan

There are two goals of monitoring: - Achieve situational awareness to provide timely and effective responses and - Gain insights for the business, development, and operations that enable proactive courses of action.

Understanding needs requires engagement. Users frequently ask for what they think they need or what they understand is possible. Apply the “5 Whys” technique, and do not assume that what they are asking for is what they actually need. Operations' engagement with teams enables understanding what they are trying to achieve and help determine options for achieving their monitoring goals.

Consider:

- Personas and requirements: Who will consume the outputs from monitoring? What are their goals?
- System architecture and insights: What are the components? What telemetry should be collected?
- Thresholds for testing: What does good look like? How will it be measured?
- Reporting: What information should be provided via "pull" on-demand in Dashboards? Who are the stakeholders?
- Alerts and actions: What are the alerts that align to business outcomes? What actions should be taken when alerts occur?

3.Implementing the Monitoring Plan: Telemetry

Consider: What telemetry can we leverage to achieve the goals? Is there desirable telemetry we do not have? If so, how can we acquire it? Can we further instrument our application to emit the information we need? Can we configure out infrastructure to provide the information we need?

Our workload must emit the data that allows operations to understand workload health and the achievement of business outcomes. We have to collect that information, analyze it, and then act on it to maximize the benefits for our organizations. If we are collecting without analysis or action we are paying to store records that will at best only be used after an event in a forensic capacity. Our goal is to achieve insight.

Categories of insight:

- Faults
- Configuration
- Accounting
- Performance
- Security
- Outcomes
- User Behavior
- Workload Behavior

We will iterate: engaging our stakeholders, determining requirements and priorities, implementing improvements to monitoring, evaluating their success, and repeating. Or put more simply: create a plan, implement the plan, check to see if the plan works, adjust the plan and repeat.

Exercise 3: Acquire Telemetry

We are going to start by focusing on 3 goals

- Titus Grone, our application director, wants to know that our production application is delighting our customers.
- Ryn Brandish, our DevOps Engineer, wants to understand when there are application issue and is concerned about security.
- Sansa Bailish, our SRE, is focused on outages, reliability, and getting better sleep.

3.1 Install and configure the CloudWatch agent

We will collect metrics and logs from our Amazon EC2 instances using the CloudWatch agent. If we had on-prem servers we could also use the CloudWatch agent to capture metrics and logs from those.

The CloudWatch Agent requires and IAM role that was created with the execution of the CloudFormation script. We will install the agent, and then configure it using a predefined configuration placed in System Manager Parameter Store by our CloudFormation script. With our configuration in parameter store we can easily apply it as a standard across our fleet. You can learn more by following this *Getting Started* link.

3.2 Review the Parameter Store configuration

The configuration file we have in Parameter Store follows a required naming convention (it must start with AmazonCloudWatch-) that allows us to take advantage of the CloudWatchAgentServerPolicy IAM role. To simplify our management. You can learn more at this link..

1. Open **Parameter Store** in the System Manager console by clicking on this link.
2. Choose **AmazonCloudWatch-ExampleCorpConfig**
3. Review the logs that will be ingested into CloudWatch Logs

3. Install and Configure the CloudWatch agent Systems Manager Run Command

Using System Manager's Run Command we can install and configure the CloudWatch agent on an entire fleet as easily as we install it on a single instance. By using a command document we ensure consistent execution and limit the errors that could be introduced by a manual process.

1. Navigate to the Systems Manager console **Run Command** page by clicking on this link.

2. Choose **Run command**.
    - In the **Command document** list, select **AWS-ConfigureAWSPackage** by selecting the radio button next to it.
    - In the **Action** list, choose **Install**.
    - In the **Name** field, type **AmazonCloudWatchAgent**
    - In the **Version** field, type *latest*. Command Parameters cwagenttargets
    - In the **Targets** area, Manually select the ExampleCorp instances. You could also target instances by tag. Command Parameters
    - In the **Output options** area uncheck the box next to **Enable writing to an S3 bucket**.
    - Choose **Run**.
3. Optionally, in the **Targets and outputs** areas, select the button next to an instance name and choose **View output**. Systems Manager should show that the agent was successfully installed.

3.3 Configure and (re)start the CloudWatch agent using Systems Manager Run Command

1. Open Run Command in the Systems Manager console by clicking on this link.
2. Choose **Run command**.
    - In the Command document list, select **AmazonCloudWatch-ManageAgent** by selecting the radio button next to it.
    - In the **Command Parameters** area
        - In the **Action** list, choose the default **configure**.
        - In the **Mode** list, choose the default **ec2**
        - In the **Optional Configuration Source** list, choose the default ssm.

        - In the **Optional Configuration Location** box, type the name of the agent configuration file that we created and saved to Systems Manager Parameter Store *AmazonCloudWatch-ExampleCorpConfig* Command Parameters
    - In the **Optional Restart** list, choose **yes** to start the agent after you have finished these steps.
    - In the **Targets** area, choose the instance where you installed the CloudWatch agent.
    - In the **Output options** area uncheck the box next to **Enable writing to an S3 bucket**.
3. Choose **Run**.
4. Optionally, in the Targets and outputs areas, select the button next to an instance name and choose View output. Systems Manager should show that the agent was successfully started

Want to learn more about collecting logs and metrics using the unified CloudWatch Agent? Follow this link.

(Optional Exercise) Review the Collected Application Logs [with the development team]

The more integrated the business, development, and operations teams are, the more successful they will be. The custom developed application emits a variety of logs. Engaging the developers to help gain insight to their contents, meaning, and their purpose will shorten the amount of time it takes to get value out of monitoring.

1. Navigate to the CloudWatch Logs dashboard at this link.
2. Choose application.log from the Log Groups list
3. Choose the Log Stream of your instance
4. Review the available logs

3.4 Generate logs through user activity

1. In a web browser navigate to the URL Output of the Example Corp CloudFormation Stack to reach the ExampleCorp application CloudFormation Output
2. Create an account by choosing login and then choosing Sign up at the Log in page.
   - Enter an Email address, a Username, and a Password and choose Sign up
   - There is no validation on the email address you provide so you may use a fake address
3. Upload some photos
   - you can either use your own photos or
   - **download sample photos here** **Note:** There are 76MB of images
4. Navigate to the CloudWatch Logs dashboard at this link.
   - Choose application.log from the Log Groups list
   - Choose the Log Stream of an instance
   - At the top right of the grey field, in the time window definition box, choose the pull down.
   - Choose to Relative, by clicking on the term.
   - Choose 2 Hours CloudFormation Output
   - Review the recent logs. Enter a Tag value identified by ExampleCorp in the Filter Events text box and review the associated logs.

3.5 Publish VPC Flow Logs to CloudWatch Logs

When publishing VPC flow logs to CloudWatch Logs, flow log data is published to a log group, and each network interface has a unique log stream in the log group. Log streams contain flow log records. You can create multiple flow logs that publish data to the same log group. If the same network interface is present in one or more flow logs in the same log group, it has one combined log stream. If you've specified that one flow log should capture rejected traffic, and the other flow log should capture accepted traffic, then the combined log stream captures all traffic. For more information, see Flow Log Records.

Consider how you will use the collected VPC Flow Log data. On internal networks visibility of intended and unintended traffic can facilitate diagnosing issues. Capturing rejected traffic on your Internet Gateway by default may result in storing and processing a lot of data with limited value to you. Remember you can always enable the capability when needed and then disable it when no longer needed to help optimize the value your get from CloudWatch.

(Optional) Create a flow log for your VPC

Note: The following steps have been completed for you by the CloudFormation script.

1. Open the Amazon VPC console and navigate to Your VPCs by clicking on this link.
2. Select your ExampleCorp VPCs and then choose Actions, Create flow log.
3. For Filter, choose All from the pull down list to log accepted and rejected traffic.
4. For Destination, choose the default Send to CloudWatch Logs.
5. For Destination log group, type the name of a log group in CloudWatch Logs to which the flow logs are to be published. If you specify the name of a log group that does not exist, it will attempt to create the log group for you. Enter ExampleCorpVPC in the text box next to Destination log group*.
6. For IAM role select the role with name in the form (your stack name)-FlowLogsRole-(random string). It was created by the CloudFormation script with permissions to publish logs to CloudWatch Logs.
7. Choose Create.

8. Open the CloudWatch console and navigate to Logs by clicking on this link.

What have we accomplished?

All of our instances are now logging to CloudWatch Logs and, our VPC is logging to CloudWatch Logs as well. Our instances and AWS services are publishing metrics to CloudWatch. Our application and workload are providing traces though integration with AWS X-Ray. We are collecting the available telemetry, and now it is time to analyze it and take advantage of it.

Exercise 4: Generate Metrics from Logs

Traditional application development emits events in the form of logs. Use CloudWatch we can generate metrics from our logs using pattern matching. By generating metrics based on observed log messages we can increase the value of our CloudWatch logs by providing visualizations of the metric data through dashboard, and providing alerts when metrics breach baseline thresholds. Using the AWS CLI or API you can publish your own custom metrics.

Metric filters are created using the same filter and pattern syntax that is used when browsing log streams in the console.

Create a Confidence metric

Monitoring for Business Outcomes Titus Grone wants to know that ExampleCorp is delighting our customers. Feedback from the customer indicates that accuracy of items identified in the upload images is the greatest source of satisfaction when it works well, and frustration when it does not. Focus groups indicate that it is better to not have misidentified (low confidence) objects.

He wants to track the image recognition confidence levels as a measure of how accurate the ExampleCorp application is performing. He will use this information to help determine where to focus development efforts.

4.1 Create the Log Metric

1. Navigate to the CloudWatch **Logs** dashboard at this link.
2. In the contents pane, select the **application.log** group by clicking on the radio button next to it, and then choose Create Metric Filter. Create Metric Filter
   - On the **Define Logs Metric Filter** screen, for **Filter Pattern**, type:

```
[logType, myTimestamp, severity, delim1, delim2, type, action,
for, Image, imgNum, Name, imgTags, Confidence, cValue]
```

```
- To test your filter pattern, for **Select Log Data to Test**,
select the log group to test the metric filter against, and then
choose Test Pattern.
- Under **Results**, CloudWatch Logs displays a message showing
how many occurrences of the filter pattern were found in the log
file. To see detailed results, click Show test results.
- Choose **Assign Metric**
```

Define Log Metric 3. On the **Create Metric Filter and Assign a Metric** screen, - For Filter Name type *confidenceLevels* - Under **Metric Details**, for **Metric Namespace**, type *ApplicationLogMetrics* - For **Metric Name**, type *cValue* - Choose **Show advanced metric settings** - For **Metric Value** choose *$cValue*. - Leave the **Default Value** undefined, and then choose **Create Filter**. Assign Metric

4.2 Review the resulting metrics

1. Navigate to metrics in the left side navigation bar of the CloudWatch console or by clicking this link.
2. Under **Custom Namespaces** you will see your **ApplicationLogMetrics** namespace.
   - Chose **Metrics with no dimensions**
   - Then choose **Metrics with no dimensions**
   - Then select **cValue**

3. Choose the **Graphed Metrics** tab and examine the differences in reported values when you change the **Statistic** in use by selecting alternatives from the pull down list beneath it.
4. Change the **Period** to **10 Seconds** and the **Statistic** to **Average** and examine the differences in reported values.

4.3 Create a dashboard

1. Choose Actions in to top right corner of the page and choose Add to dashboard
2. In the Add to dashboard dialog
   - Under Select a dashboard choose Create new and enter ExampleCorp in the Dashboard name text box, and then choose the check mark icon next to the text box to confirm your choice.
   - Under Select a widget type choose Stacked area
   - Under Customize widget title replace the prepopulated value cValue with confidence
   - Choose Add to dashboard
3. On the Dashboards page, choose Save dashboard

Exercise 5: Create Alarms for Known Issues from Metrics

Monitoring for Application Outcomes Ryn Brandish wants to understand when there are application issue and is concerned about security. He would like metrics for errors and warnings. ExampleCorp has limitations on the image size that it can successfully process. While this issue does not happen frequently it does not make customers happy. Being aware of the error rate for submitted images will allow the Business and Development team to determine if increase the image size should be a priority. Currently most warnings are related to security issues. Ryn would like visibility on how often they happen.

5.1 Create a log filter metric based on the defined threshold for the error

1. Click on Log Groups in the breadcrumb trail at the top of the screen
2. Select the application.log log group
3. Click on the Create Metric Filter button
4. Enter "ActiveStorage::InvariableError" (include quotes) for Filter Pattern. This is a known error that we will cause later in the lab.
5. Click on the Assign Metric button
6. Enter ImageError as the Metric Name
7. Click on the Create Filter button Image Error Metric

5.2 Create an alarm for the error when the metric threshold is crossed

1. Click on Create Alarm
2. Click Edit by Metric Edit Metric
3. Change Period to 10 seconds
4. Click on Select metric button
    - Enter ImageErrorAlarm for Name
    - After is >=, change the 0 to 1
    - Select good (not breaching threshold) for Treat missing data as
    - Under Actions Select from the Drop Down the ExampleCorpErrorNotify* list to Send notifications to
    - Enter [your email address] for Email list Create Alarm
5. Click on the Create Alarm button
6. Click on the I will do it later button when asked to verify the email address

NOTE: You should see that the alarm has insufficient data with a 1 next to INSUFFICIENT in the navigation menu. This will change to OK within a few seconds.

5.3 Generate error logs through user activity

1. Navigate to the ExampleCorp using the URL that you made a note of earlier (CloudFormation Output).
2. Enter admin@admin.com for email
3. Enter Password123 for Password
4. Click Login
5. Click Upload Image Click Select Image
6. Find break_app.jpg in your filesystem
7. Click Upload

NOTE: The application will start exhibiting issues, and the application will eventually fail to load.

5.4 Review logs to identify error

1. Navigate to the CloudWatch Console
2. Click Logs in the navigation menu
3. Click on application.log log group - this is from /opt/ExampleCorp/log/application.log as configured earlier in the CloudWatch agent configuration.
4. Click on the Search Log Group button
5. Enter ERROR (case sensitive) and hit enter.
6. You will see errors ending ActiveStorage::InvariableError. This is an error that is generated when a user uploads a non-image file. This is a known issue, but the Dev team has not had cycles to adress it.

Build a Monitoring Plan: Alerting and Response

Monitoring for Operational Outcomes Sansa Bailish is focused on outages, reliability, and getting better sleep. There is a known issue with image trends; when the instances are rebooted the application doesn't restart. Too frequently Sansa has received a late night call to get online and restart the application. The user experience lead is very frustrated with the downtime associated to these incidents. They need to be detected sooner and resolved faster.

Sansa is looking for a monitoring solution to detect the incident (the reboot event), and a way to trigger an automated recovery.

Exercise 6: Create and Automate Responses

Create a Runbook to Resolve the Error

6.1 Resolve the error manually

1. Depending on your browser, you will see an error showing that the page cannot be loaded.
2. Add /admin to the end of the URL from the CloudFormation output.
3. Click on images in the menu to see the images with the error.
4. Delete the image by pressing the x next to the image with no tags or errors in the tags.
5. Click Home on the navigation bar at the top **NOTE:** The application should be working again.

6.2 Automate Responses using Lambda

1. Navigate to the Lambda Console
2. Click on the ExampleCorpErrorEvent Lambda function
3. Under Basic settings
    - Set Timeout to 30 seconds

    - Under Network
        - Select the VPC created earlier with a 10.180.0.0/16 CIDR block for VPC
        - Select ...App Subnet (AZ1) under Subnets
        - Select ...App Subnet (AZ2) under Subnets
        - Select ...AppHostSecurityGroup... under Security groups

4. Click on the Save button

6.3 Validate - Ensure the response has the desired outcome

1. Log in to ExampleCorp using the URL that you made a note of earlier (CloudFormation Output) in a new browser tab.
2. Enter admin@admin.com for email
3. Enter Password123for Password
4. Click Login
5. Click Upload Image
6. Click Select Image
7. Find break_app.jpg in your filesystem
8. Click Upload
9. It will break again, but it should recover in a few seconds
10. Keep an eye on the Lambda monitoring page

6.4: Bonus Content: Review the outcomes in CloudWatch

1. Click on View logs in CloudWatch
2. Expand parts of the logs to see what happened.
3. The Lambda got invoked
4. You can see the SNS Message
5. Then you see that 2 rows were deleted
6. Finally the Lambda execution finishes with some summary statistics.

7. Click on Alarms
8. Select ImageErrorAlarm
9. View the history see the Alarm State update and the Action to notify SNS which then calls the Lambda function.

What have we achieved? - We have answered three monitoring needs, one for each of our teams. - We have provided insight to the quality of the end user experience by creating a image tag confidence metric for the business team. - We have provided insight to issues and potential development needs by providing the development team (DevOps) with error and warning metrics. - We have provided a mechanism for the operations team (SRE) to use an event trigger to automatically remediate an outage causing issue in the environment.

How did we create an automatic remediation? - Knowing the log entry that indicates that the application has failed and is in a state from which it can be recovered we created a metric. - Using that metric we created an alarm that notifies via an SNS topic. - We have a Lambda function that is subscribed to that SNS topic that creates a CloudWatch Event in response. - We created a CloudWatch rule that triggers on our Lambda initiated CloudWatch event and invokes run command. - Our run command invocation uses a command document we created to execute the start up script on our server restoring it to an operating state.

We have created something of a Rube Goldberg machine to achieve that outcome. In doing so we have demonstrated the use of logs and how to get more value out of them, the use of events, and the triggering of actions in response to events; all enabled by CloudWatch.

Exercise 7: CloudWatch Logs Insight (CWL-I)

CWL-I allows you to rapidly query across a log group using a powerful query language. To extract data from a log field, creating one or more ephemeral fields that can be further processed by the CWL-I queries perform the following.

7.1 Play Around with application.log in CloudWatch Logs Insight

1. Navigate to the CWL-I console
2. Choose the Log Group you want to work with from the pull down, at the top right of the grey field. Choose application.log
3. Choose the pull down in the time window definition box, at the top right of the grey field.
4. Choose to define a Relative or Absolute time window, by clicking on the term
5. Select a period overlapping the log activity of interest.
6. Leave the default Query.

```
fields @timestamp, @message
| sort @timestamp desc
| limit 20
```

1. Run Query and Observe and play around with the Data and Graph. CW Insights application.log

7.2 Play Around with ManagementToolsWeek/CloudTrail in CloudWatch Logs Insight

1. Navigate to the CloudWatch Logs Console, on the left side Click on **Insight**
2. Choose the **Log Group** you want to work with from the pull down, at the top right of the grey field. Choose *ManagementToolsWeek/CloudTrail*
3. Choose the pull down in the time window definition box, at the top right of the grey field.
4. Choose to define a Relative time for the past two days.
5. Select Sample Queries --> CloudTrail --> Number of log entries by service, event type, and region

```
stats count(*) by eventSource, eventName, awsRegion
```

1. Run Query and Observe and play around with the Data and Graph. CW Insights ManagementToolsWeek/CloudTrail

Exercise 8: View X-Ray Traces

1. Navigate to the X-Ray Console
2. Let the Service Map Refresh
3. View the Traces, and notice the information collected.

X-Ray Service Map for Example Corp App

Exercise 9 Lab tear down

To remove all the resources configured and deployed as part of this lab perform the following:

1. Navigate to the CloudFormation console at this link
   - Select your stack, choose Actions, and choose Delete stack
   - This will delete all resources created by the CloudFormation template
2. Navigate to the Logs page on the CloudWatch console at this link
   - Select /aws/lambda/ExampleCorpRebootedEvent, then choose Actions, then choose Delete log group, and finally choose Yes, Delete to delete the Log Group
   - Select application.log, then choose Actions, then choose Delete log group, and finally choose Yes, Delete to delete the Log Group
   - Select boot.log, then choose Actions, then choose Delete log group, and finally choose Yes, Delete to delete the Log Group
   - Select messages, then choose Actions, then choose Delete log group, and finally choose Yes, Delete to delete the Log Group

    - Select production.log, then choose Actions, then choose Delete log group, and finally choose Yes, Delete to delete the Log Group

3. Navigate to the Metrics page on the CloudWatch console at this link
    - CloudWatch does not support metric deletion. Metrics expire based on the retention schedules which can be found at this link
4. Navigate to the Alarms page on the CloudWatch console at this link
    - Select ExampleCorpRebootedAlarm by checking the box on the left in its row.
    - Choose Actions, choose Delete, and finally choose Yes, Delete to delete the Alarm
5. Navigate to the Dashboards page of the CloudWatch console at this link
    - Choose ExampleCorp
    - Choose Actions, choose Delete dashboard, and finally when prompted select Delete dashboard to confirm that you wish to delete your dashboard.

References: ExampleCorp

A mock workload for a fictitious company.

- Source code: https://github.com/horsfieldsa/example-corp-app
- Admin interface: http:///admin
- Admin user: admin@admin.com
- Admin password: Password123

End of Lab Exercises

Thank you for using this lab.