**Lab - Monitoring Infrastructure**

The ability to monitor your applications and infrastructure is critical for delivering reliable, consistent IT services.

Monitoring requirements range from collecting statistics for long-term analysis through to quickly reacting to changes and outages. Monitoring can also support compliance reporting by continuously checking that infrastructure is meeting organizational standards.

This lab shows you how to use Amazon CloudWatch Metrics, Amazon CloudWatch Logs, Amazon CloudWatch Events and AWS Config to monitor your applications and infrastructure.

The lab will demonstrate how to:

* Use AWS Systems Manager Run Command to install the **CloudWatch Agent** on Amazon EC2 instances
* Monitor Application Logs using CloudWatch Agent and **CloudWatch Logs**
* Monitor system metrics using CloudWatch Agent and **CloudWatch Metrics**
* Create real-time notifications using **CloudWatch Events**
* Track infrastructure compliance using **AWS Config**

**Duration**

This lab will require approximately **40 minutes** to complete.

**Accessing the AWS Management Console**

1. At the top of these instructions, click Start Lab to launch your lab.

A Start Lab panel opens displaying the lab status.

1. Wait until you see the message "**Lab status: ready**", then click the **X** to close the Start Lab panel.
2. At the top of these instructions, click AWS

This will open the AWS Management Console in a new browser tab. The system will automatically log you in.

**Tip**: If a new browser tab does not open, there will typically be a banner or icon at the top of your browser indicating that your browser is preventing the site from opening pop-up windows. Click on the banner or icon and choose "Allow pop ups."

1. Arrange the AWS Management Console tab so that it displays along side these instructions. Ideally, you will be able to see both browser tabs at the same time, to make it easier to follow the lab steps.

Please do not change the Region during this lab.

**Task 1: Install the CloudWatch Agent**

The **CloudWatch Agent** can be used to collect metrics from Amazon EC2 instances an on-premises servers including:

* **System-level metrics from Amazon EC2 instances**, such as: CPU allocation, free disk space and memory utilization. These metrics are collected from the machine itself and compliment the standard Amazon CloudWatch metrics collected by CloudWatch.
* **System-level metrics from on-premises servers**, enabling monitoring of hybrid environments and servers not managed by AWS.
* **System and Application Logs** from both Linux and Windows servers.
* **Custom metrics** from applications and services using the [StatsD](https://github.com/etsy/statsd) and [collectd](https://collectd.org/) protocols.

In this task, you will use AWS Systems Manager to install the CloudWatch Agent on an Amazon EC2 instance. You will configure it to collect both application and system metrics.

1. In the **AWS Management Console**, on the **Services** menu, click **Systems Manager**.
2. In the left navigation pane, click **Run Command**.

If there is no visible navigation pane, click the icon in the top-left corner to make it appear.

You will use the Run Command to deploy a pre-written command that installs the CloudWatch Agent.

1. Click **Run a Command**
2. Select the radio button next to **AWS-ConfigureAWSPackage** (typically appears towards the top of the list).
3. Scroll down to the **Command parameters** section and configure:
   * **Action:** *Install*
   * **Name:** AmazonCloudWatchAgent
   * **Version:** *latest*
4. In the **Targets** section, select **Choose instances manually** and then select **Web Server**.

This configuration will install the CloudWatch Agent on the *Web Server*.

1. At the bottom of the page, click **Run**
2. Wait for the **Overall status** to change to *Success*. You can occasionally click refresh towards the top of the page to update the status.

You can view the output from the job to confirm that it ran successfully.

1. Under **Targets and outputs**, click instance name displayed under **Instance ID**
2. Expand **Step 1 - Output**.

You should see the message: *Successfully installed arn:aws:ssm:::package/AmazonCloudWatchAgent*

You will now configure CloudWatch Agent to collect the desired log information. The instance has a web server installed, so you will configure CloudWatch Agent to collect the web server logs and also general system metrics.

You will store the configuration file in AWS Systems Manager Parameter Store, which can then be fetched by the CloudWatch Agent.

1. In the left navigation pane, click **Parameter Store**.
2. Click **Create parameter** then configure:
   * **Name:** Monitor-Web-Server
   * **Description:** Collect web logs and system metrics
   * **Value:** Paste the configuration shown below:



{

 "logs": {

   "logs\_collected": {

     "files": {

       "collect\_list": [

        {

           "log\_group\_name": "HttpAccessLog",

           "file\_path": "/var/log/httpd/access\_log",

           "log\_stream\_name": "{instance\_id}",

           "timestamp\_format": "%b %d %H:%M:%S"

        },

        {

           "log\_group\_name": "HttpErrorLog",

           "file\_path": "/var/log/httpd/error\_log",

           "log\_stream\_name": "{instance\_id}",

           "timestamp\_format": "%b %d %H:%M:%S"

        }

      ]

    }

  }

},

 "metrics": {

   "metrics\_collected": {

     "cpu": {

       "measurement": [

         "cpu\_usage\_idle",

         "cpu\_usage\_iowait",

         "cpu\_usage\_user",

         "cpu\_usage\_system"

      ],

       "metrics\_collection\_interval": 10,

       "totalcpu": false

    },

     "disk": {

       "measurement": [

         "used\_percent",

         "inodes\_free"

      ],

       "metrics\_collection\_interval": 10,

       "resources": [

         "\*"

      ]

    },

     "diskio": {

       "measurement": [

         "io\_time"

      ],

       "metrics\_collection\_interval": 10,

       "resources": [

         "\*"

      ]

    },

     "mem": {

       "measurement": [

         "mem\_used\_percent"

      ],

       "metrics\_collection\_interval": 10

    },

     "swap": {

       "measurement": [

         "swap\_used\_percent"

      ],

       "metrics\_collection\_interval": 10

    }

  }

}

}

Examine the above configuration. It defines the following items to be monitored:

* + **Logs:** Two web server log files to be collected and sent to Amazon CloudWatch Logs
  + **Metrics:** CPU, disk and memory metrics to send to Amazon CloudWatch Metrics

1. Click **Create parameter**

This parameter will be referenced when starting the CloudWatch Agent.

You will now use another *Run Command* to start the CloudWatch Agent on the Web Server.

1. In the left navigation pane, click **Run Command**.
2. Click **Run command**
3. Click then:
   * *Document name prefix*
   * *Equals*
   * AmazonCloudWatch-ManageAgent
   * Press Enter

Before running the command, you can view the definition of the command.

1. Click **AmazonCloudWatch-ManageAgent** (click on the name itself).

A new web browser tab will open, showing the definition of the command.

Browse through the content of each tab to see how a Command Document is defined.

1. Click the **Content** tab and scroll to the bottom to see the actual script that will run on the target instance.

The script references the AWS Systems Manager Parameter Store because it will retrieve the CloudWatch Agent configuration you defined earlier.

1. Close the current web browser tab, which should return you to the *Run a command* tab you were using earlier.

Verify that you have selected the radio button next to **AmazonCloudWatch-ManageAgent**.

1. In the **Command parameters** section, configure:
   * **Action:** *configure*
   * **Mode:** *ec2*
   * **Optional Configuration Source:** *ssm*
   * **Optional Configuration Location:** Monitor-Web-Server
   * **Optional Restart:** *yes*

This configures the Agent to use the configuration you previously stored in the Parameter Store.

1. In the *Targets* panel below, select **Choose instances manually** and then select **AmazonCloudWatch-ManageAgent** (click the circle).
2. In the **Instances** section, select **Web Server**.
3. Click **Run**
4. Wait for the **Overall status** to change to *Success*. You can occasionally click refresh towards the top of the page to update the status.

The CloudWatch agent is now running on the instance, sending log and metric data to Amazon CloudWatch.

**Task 2: Monitor Application Logs using CloudWatch Logs**

You can use **Amazon CloudWatch Logs** to monitor applications and systems using **log data**. For example, CloudWatch Logs can track the number of errors that occur in your application logs and send you a notification whenever the rate of errors exceeds a threshold you specify.

CloudWatch Logs uses your existing log data for monitoring; so, no code changes are required. For example, you can monitor application logs for specific literal terms (such as "NullReferenceException") or count the number of occurrences of a literal term at a particular position in log data (such as "404" status codes in web server access log). When the term you are searching for is found, CloudWatch Logs reports the data to a CloudWatch metric that you specify. Log data is encrypted while in transit and while it is at rest.

In this task you will generate log data on the Web Server, then monitor the logs using CloudWatch Logs.

The Web Server generates two types of log data:

* Access Logs
* Error Logs

You will begin by accessing the web server.

1. Click on the Details drop down menu above these instructions you are currently reading, and then click Show. Copy the **WebServerIP** value.
2. Open a new web browser tab, paste the *WebServerIP* you copied, then press Enter.

You should see a web server **Test Page**.

You will now generate log data by attempting to access a page that does not exist.

1. Append /start to the browser URL and press Enter.

You will receive an **error message** because the page is not found. **This is okay!** It will generate data in the access logs that are being sent to CloudWatch Logs.

1. Keep this tab open in your web browser, but return to the browser tab showing the AWS Management Console.
2. From the **Services** menu, choose **CloudWatch**.
3. In the left navigation pane, click **Log groups**.

You should see two logs listed: **HttpAccessLog** and **HttpErrorLog**.

If these logs are not listed, try waiting a minute, then click **Refresh**.

1. Click **HttpAccessLog** (click on the actual name).
2. Click the value displayed under **Logs Streams**.

Log data should be displayed, consisting of **GET** requests that were sent to the web server. You can view further information by expanding the lines. The log data includes information about the computer and browser that made the request.

You should see a line with your **/start** request with a code of 404, which means that the page was not found.

This demonstrates how log files can be automatically shipped from an Amazon EC2 instance, or an on-premises server, to CloudWatch Logs. The log data is accessible without having to log in to each individual server. Log data can also be collected from multiple servers, such as an Auto Scaling fleet of web servers.

**Create a Metric Filter in CloudWatch Logs**

You will now configure a Filter to identify *404 Errors* in the log file. This would normally be an indication that the web server is generating invalid links that users are clicking.

1. In the left navigation pane click **Log groups**.
2. Select **HttpAccessLog** (click the circle, not the link).
3. From the **Actions** drop down select **Create metric filter**.

A **filter pattern** defines the fields in the log file and filters the data for specific values.

1. Paste this line into **Filter pattern:**



[ip, id, user, timestamp, request, status\_code=404, size]

This tells CloudWatch Logs how to interpret the fields in the log data and defines a filter to only find lines with **status\_code=404**, which indicates that a page was not found.

1. Under **Test pattern**, use the drop down menu to select the ec2 instance id. It will be similar to *i-0f07ab62aae4xxxx9*.
2. Choose **Test pattern**
3. In the **Results** section, click **Show test results**.

You should see at least one result with a *$status\_code* of **404**. This indicates that a page was requested that was not found.

1. Choose **Next**
2. For **Filter name** enter 404Errors
3. Configure these **Metric Details:**
   * **Metric Namespace:** LogMetrics
   * **Metric Name:** 404Errors
   * **Metric value:** 1
4. Choose **Next**
5. On the **Review and create** page, choose **Create metric filter**

This metric filter can now be used in an Alarm.

**Create an Alarm using the Filter**

You will now configure an Alarm to send a notification when too many *404 Not Found* errors are received.

1. In the panel titled **404Errors**, choose the check box in the top right corner.
2. Next to **Metric filters**, choose **Create alarm**
3. Configure these settings:
   * **Period:** *1 minute*
   * **Conditions:**
     + Whenever 404Errors is: **Greater/Equal**
     + than: 5
   * Click **Next**
4. For **Notification**, configure:
   * **Select an SNS Topic:** **Create new topic**
   * **Email endpoints that will receive the notification:** Enter an email address that you can access from the classroom
   * Click **Create topic**
   * Click **Next**
5. For **Name and description**, configure:
   * **Alarm name:** 404 Errors
   * **Alarm description:** Alert when too many 404s detected on an instance
   * Click **Next**
6. Click **Create alarm**
7. Go to your email, look for a confirmation message and select the **Confirm subscription** link.
8. Return to the **AWS Management Console**.
9. In the left navigation pane, click **CloudWatch** (at the very top).

Your alarm should appear in orange, indicating that there is *INSUFFICIENT DATA* to trigger the alarm. This is because no data has been received in the past minute.

You will now access the web server to generate log data.

1. Return to the web browser tab with the web server.

If the tab is no longer open, copy the *WebServerIP* shown to the left of the instructions you are current reading, and open a new web page with that IP address.

1. Attempt to go to pages that do not exist by adding a page name after the IP address. Repeat this **at least 5 times**.

For example: [*http://54.11.22.33/start2*](http://54.11.22.33/start2)

Each separate request will generate a separate log entry.

1. Wait 1-2 minutes for the Alarm to trigger. You can occasionally click Refresh to update the status.

The graph shown on the CloudWatch page should turn red to indicate that it is in the *ALARM* state.

1. Check your email. You should have received an email titled *ALARM: "404 Errors"*.

This demonstrates how you can create an Alarm from application log data and receive alerts when unusual behavior is detected in the log file. The log file is easily accessible within Amazon CloudWatch Logs to perform further analysis to diagnose the activities that led to the Alarm being triggered.

**Task 3: Monitor Instance Metrics using CloudWatch**

**Metrics** are data about the performance of your systems. Amazon CloudWatch stores metrics for AWS services you use. You can also publish your own application metrics either via CloudWatch Agent or directly from your application. Amazon CloudWatch can present the metrics for search, graphs, dashboards, and alarms.

In the task, you will use explore metrics provided by Amazon CloudWatch.

1. On the **Services** menu, click **EC2**.
2. In the left navigation pane, click **Instances**.
3. Select **Web Server**.
4. Click the **Monitoring** tab in the lower half of the page.

Examine the metrics presented. You can also **click on a chart** to display more information.

CloudWatch captures metrics about CPU, Disk and Network usage on the instance. These metrics view the instance 'from the outside' as a *virtual machine* but do not give insight into what is running 'inside' the instance, such as measuring free memory or free disk space. Fortunately, you can obtain information about what is happening *inside* the instance by using information captured by **CloudWatch Agent**, because CloudWatch Agent runs *inside* the instance to collect metrics.

1. From the **Services menu**, select **CloudWatch**.
2. In the left navigation pane, choose **Metrics**.

The lower half of the page will display the various metrics that have been collected by CloudWatch. Some are automatically generated by AWS while others were collected by the CloudWatch Agent.

1. Click **CWAgent**, then **device, fstype, host, path**.

You will see the disk space metrics being captures by CloudWatch Agent.

1. Click **CWAgent** (in the line that says *All > CWAgent > device, fstype, host, path*).
2. Click **host**.

You will see metrics relating to system memory.

1. Click **All** (in the line that says *All > CWAgent > device, fstype, host, path*).

Explore the other metrics that are being captured by CloudWatch. These are automatically-generated metrics coming from the AWS services that have been used in this AWS account.

You can select metrics that you wish to appear on the graph.

**Task 4: Create Real-Time Notifications**

**Amazon CloudWatch Events** delivers a near real-time stream of system events that describe changes in AWS resources. Simple rules can match events and route them to one or more target functions or streams. CloudWatch Events becomes aware of operational changes *as they occur*.

CloudWatch Events responds to these operational changes and takes corrective action as necessary, by sending messages to respond to the environment, activating functions, making changes, and capturing state information. You can also use CloudWatch Events to schedule automated actions that self-trigger at certain times using cron or rate expressions.

In this task, you will create a real-time notification that informs you when an instance is Stopped or Terminated.

1. In the left navigation pane, click **Rules**.
2. Click **Create rule**
3. In the **Event Source** section, configure:
   * **Service Name:** *EC2*
   * **Event Type:** *EC2 Instance State-change Notification*
   * Specific state(s)
   * From the drop-down menu, select **stopped** and **terminated**
4. In the **Targets** section on the right, configure:
   * **Add target**
   * Click **Lambda function** and change it to **SNS topic**
   * **Topic:** select the Default\_CloudWatch\_Alarms\_Topic option.
5. Click **Configure details** (at the bottom of the page).
6. In the **Rule definition**, configure:
   * **Name:** Instance\_Stopped\_Terminated
   * Click **Create rule**

**Configure Real-Time Notification**

In addition to receiving an email, you can configure Amazon Simple Notification Service (SNS) to send you a notification to your phone via SMS.

1. On the **Services** menu, click **Simple Notification Service**.
2. In the left navigation pane, click **Topics**.
3. Click the link in the **Name** column.

You should see a single subscription associated with your email address. You will now add an SMS notification.

If you do not have a phone that can receive SMS messages, you can skip this step.

1. Click **Create subscription** then configure:
   * **Protocol:** *SMS*
   * **Endpoint:** Enter your cell phone number in International format (eg +14155557000 or +917513200000)
   * Click **Create subscription**

You are now ready to trigger a real-time alert!

1. On the **Services** menu, click **EC2**.
2. In the left navigation pane, click **Instances**.
3. Select **Web Server**.
4. Click **Instance state** then **Stop instance**, then **Stop**

The *Web Server* instance will enter the *stopping* state. After a minute it will enter the *stopped* state.

You should then receive an SMS message with details of the instance that was stopped.

The message is formatted in JSON. To receive an easier-to-read message, you could create an AWS Lambda function that is triggered by CloudWatch Events. The Lambda function could then format a more friendly message and send it via Amazon SNS.

This demonstrates how easy it is to receive real-time notifications when infrastructure changes.

**Task 5: Monitor for Infrastructure Compliance**

**AWS Config** is a service that enables you to assess, audit, and evaluate the configurations of your AWS resources. Config continuously monitors and records your AWS resource configurations and allows you to automate the evaluation of *recorded configurations* against *desired configurations*.

With Config, you can review changes in configurations and relationships between AWS resources, dive into detailed resource configuration histories, and determine your overall compliance against the configurations specified in your internal guidelines. This enables you to simplify compliance auditing, security analysis, change management, and operational troubleshooting.

In this task, you will activate AWS Config Rules to ensure compliance of tagging and EBS Volumes.

1. On the **Services** menu, click **Config**.
2. If a **Get started** button appears, do the following:
   * Click **Get started**
   * Click **Next**. If you receive an error, then under **AWS Config role** at the bottom of the page, select **Use an existing AWS Config service-linked role**.
   * Click **Skip**
   * Click **Confirm**

This will configure AWS Config for initial use.

1. In the left navigation pane, click **Rules** (the one towards the top).
2. Click **Add rule**
3. In the search field, enter: required-tags
4. Click the **required-tags** box that appears.

You will configure the rule to require a **project** code for each resource.

1. Scroll down to **Rule parameters** and configure:
   * To the right of **tag1Key**, enter: project (replace any existing value)
   * Click **Save** (at the bottom of the page)

This rule will now look for resources that do not have a *project* tag. This will take a few minutes to complete, so please continue with the next steps. There is no need to wait.

You will now add a rule that looks for Amazon EBS volumes that are not attached to Amazon EC2 instances.

1. Click **Add rule**
2. In the search field, enter: ec2-volume-inuse-check
3. Click the **ec2-volume-inuse-check** box that appears.
4. Click **Save**
5. Wait until at least one of the rules has completed evaluation. Click **Refresh** in the top-right every 60 seconds to update the status.

If you receive a message that there are *No resources in scope*, please wait a few minutes longer. This message is an indication that AWS Config is still scanning available resources. The message will eventually disappear.

1. Click each of the rules to view the result of the audits.

Amongst the results should be:

* + **required-tags:** A compliant EC2 Instance (because the Web Server has a *project* tag) and many non-compliant resources that do not have a *project* tag
  + **ec2-volume-inuse-check:** One compliant volume (attached to an instance) and one non-compliant volume (*not* attached to an instance)

AWS Config has a large library of pre-defined compliance checks and you can create additional checks by writing your own AWS Config Rule using AWS Lambda.

**Lab Complete**

Congratulations! You have completed the lab.

1. Click End Lab at the top of this page and then click **Yes** to confirm that you want to end the lab.

A panel will appear, indicating that "DELETE has been initiated... You may close this message box now."

1. Click the **X** in the top right corner to close the panel.