DevOps

[Cloud Watch 2](#_Toc193568844)

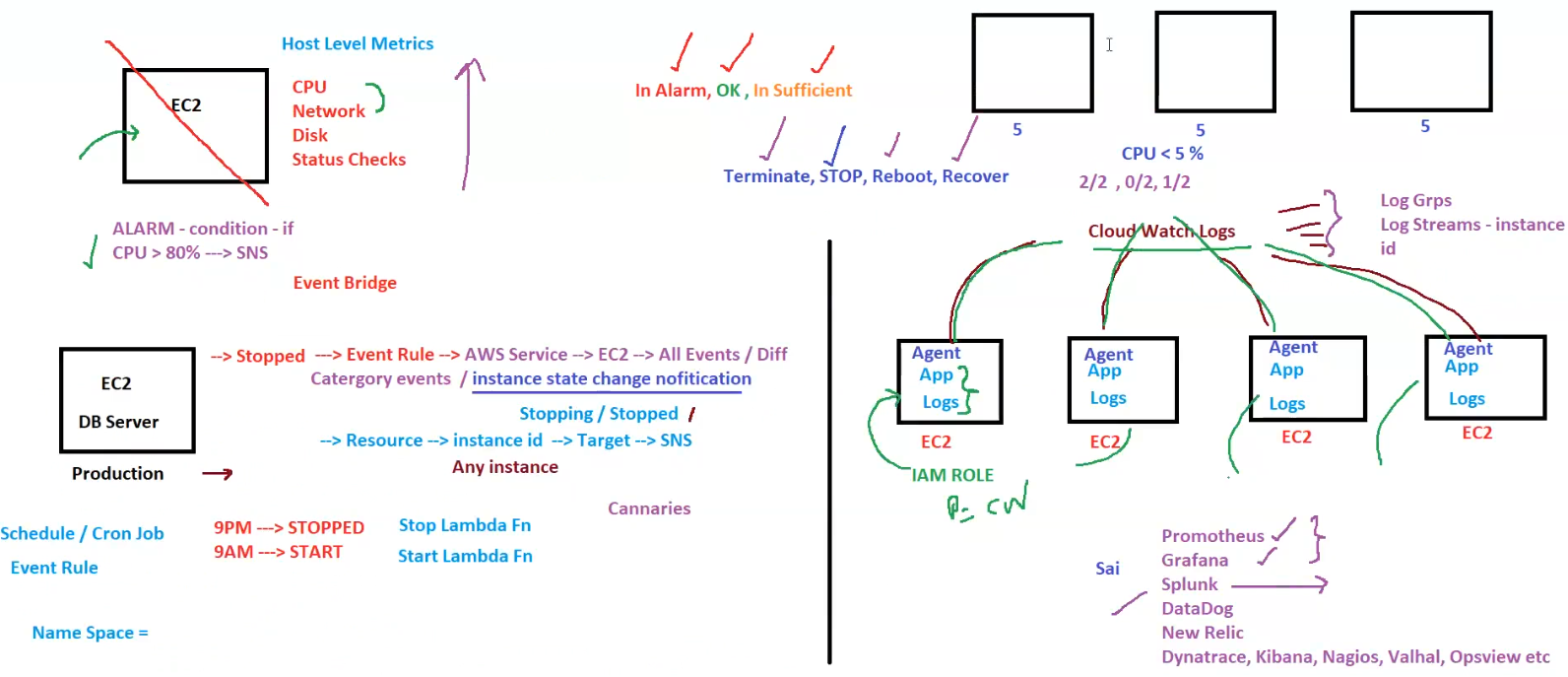
[AWS CloudWatch: Monitoring, Logging & Automation 2](#_Toc193568845)

[CloudWatch 4](#_Toc193568846)

[CloudWatch Logs Setup for EC2 Instances 6](#_Toc193568847)

# Cloud Watch

## AWS CloudWatch: Monitoring, Logging & Automation



**AWS CloudWatch: Monitoring, Logging & Automation**

This diagram provides insights into AWS **CloudWatch Alarms, Logs, EventBridge, and Automated Actions** for EC2 instances.

**1. Key Metrics Monitored by CloudWatch**

✅ **Host-Level Metrics**

* **CPU Utilization**
* **Network**
* **Disk Usage**
* **Status Checks**

✅ **Default Monitoring**

* Every **5 minutes** (Free)

✅ **Detailed Monitoring**

* Every **1 minute** (Billable)

✅ **Custom Metrics**

* Example: **Memory Utilization**

**2. CloudWatch Alarms**

* Example: **Trigger an alarm when CPU > 80% → Send SNS Notification**
* Alarm **Actions**:  
  ✅ Stop, Terminate, Reboot, Recover  
  ❌ Cannot Start EC2
* **Alarm States**:  
  🔴 **In Alarm** (Threshold Exceeded)  
  🟢 **OK** (Normal)  
  🟡 **Insufficient Data**

**3. Automating EC2 Start/Stop with EventBridge & Lambda**

**EventBridge Rules**

* ⏰ **9 PM** → **Stop Lambda Function** (Stops EC2 Instances)
* ⏰ **9 AM** → **Start Lambda Function** (Starts EC2 Instances)

**Lambda Function Example**

import boto3

ec2 = boto3.client('ec2')

def lambda\_handler(event, context):

instances = ['i-0123456789abcdef0'] # Replace with your EC2 instance ID

action = event.get('action', 'stop') # Default action is 'stop'

if action == 'stop':

ec2.stop\_instances(InstanceIds=instances)

return f"EC2 Instances {instances} Stopped"

elif action == 'start':

ec2.start\_instances(InstanceIds=instances)

return f"EC2 Instances {instances} Started"

**4. CloudWatch Logs for Centralized Logging**

* **Log Streams**: Each EC2 instance logs data separately
* **Log Groups**: Group logs based on applications/services
* **IAM Role**: EC2 must have **CloudWatch permissions** to send logs
* **Tools for Log Analysis**:
  + 📊 **Grafana, Splunk, DataDog, New Relic, Prometheus**

**5. Canary Monitoring & AWS X-Ray**

| **Feature** | **Description** |
| --- | --- |
| **AWS X-Ray** | Traces application requests for debugging |
| **Canary Monitoring** | Synthetic testing of URLs |

**6. Summary**

✅ **Monitor EC2, Apps, and Infrastructure**  
✅ **Use EventBridge & Lambda for automation**  
✅ **CloudWatch Logs centralizes log data**  
✅ **Integrate with Grafana, Splunk, DataDog for insights**

## CloudWatch

**AWS CloudWatch Overview**

This diagram provides an overview of **AWS CloudWatch**, focusing on **monitoring, alarms, logging, and automation** using EventBridge and Lambda.

**1. Key CloudWatch Features**

| **Feature** | **Description** |
| --- | --- |
| **Host-Level Metrics** | CPU, Network, Disk, Status Check |
| **Basic Monitoring** | Every 5 min (Default, Free) |
| **Detailed Monitoring** | Every 1 min (Billable) |
| **Alarms** | Trigger notifications (SNS) or actions (Stop, Terminate, Reboot) |
| **Composite Alarms** | Combine multiple alarms with AND/OR conditions |
| **CloudWatch Logs** | Aggregate logs from EC2, Apps, and Services |
| **EventBridge** | Automates EC2 Start/Stop via Lambda |

**2. CloudWatch Alarms**

* **Example:** EC2 **CPU > 90%** → Send **SNS Notification**
* **Actions:**
  + ✅ Stop, Terminate, Reboot, Recover
  + ❌ Cannot Start an EC2 Instance
* **States:**
  + 🔴 **In Alarm** (Threshold Exceeded)
  + 🟢 **OK** (Normal)
  + 🟡 **Insufficient Data**

**3. Automating EC2 Start/Stop Using EventBridge & Lambda**

* **EventBridge Rules:**
  + ⏰ **9 PM** → **Stop Lambda Function** (Stops EC2 Instances)
  + ⏰ **6 AM** → **Start Lambda Function** (Starts EC2 Instances)

**Lambda Function Example**

import boto3

ec2 = boto3.client('ec2')

def lambda\_handler(event, context):

instances = ['i-0123456789abcdef0'] *# Replace with your EC2 instance ID*

action = event.get('action', 'stop') *# Default action is 'stop'*

if action == 'stop':

ec2.stop\_instances(InstanceIds=instances)

return f"EC2 Instances {instances} Stopped"

elif action == 'start':

ec2.start\_instances(InstanceIds=instances)

return f"EC2 Instances {instances} Started"

**4. CloudWatch Logs**

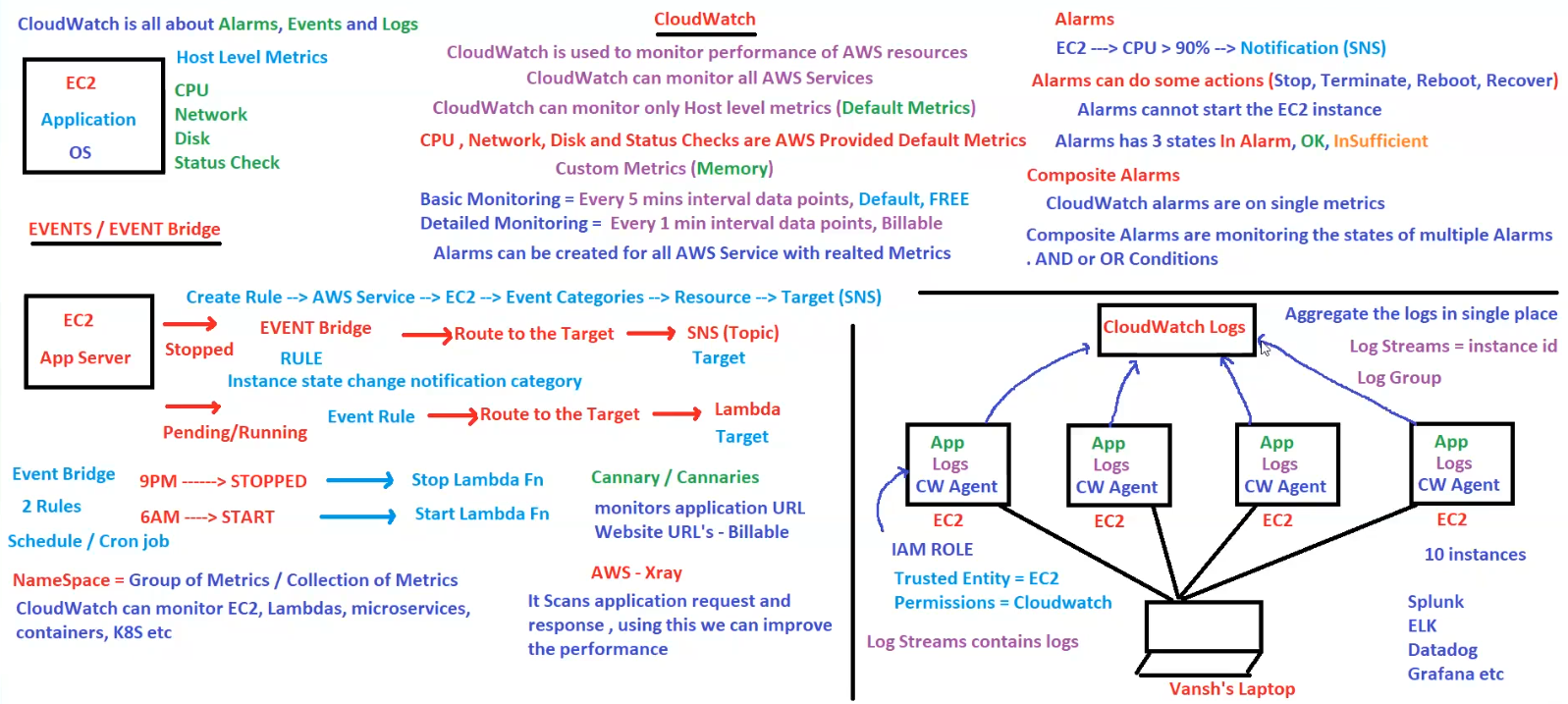
* **Log Streams:** Store logs from EC2, Applications, Lambda
* **Use Case:** Centralized logging for troubleshooting
* **Log Analysis Tools:**
  + 📊 **Splunk, ELK, Datadog, Grafana**
* **IAM Role:** EC2 must have permissions to write logs to CloudWatch.

**5. AWS X-Ray & Canary Monitoring**

| **Feature** | **Description** |
| --- | --- |
| **AWS X-Ray** | Traces application requests for performance tuning |
| **Canary Monitoring** | Synthetic monitoring of application URLs |

**6. Summary**

✅ **CloudWatch provides monitoring, alarms, logs & automation**  
✅ **Use EventBridge + Lambda to automate EC2 lifecycle**  
✅ **CloudWatch Logs centralize log management**  
✅ **Integrate with X-Ray, Splunk, ELK, Grafana for insights**



## CloudWatch Logs Setup for EC2 Instances

**CloudWatch Logs Setup for EC2 Instances (Amazon Linux 2)**

This diagram explains how to **send EC2 instance logs to AWS CloudWatch Logs** using the **CloudWatch (CW) Agent**.

**1. Steps to Configure CloudWatch Logs**

**Step 1: Launch EC2 Instances**

✅ Launch **two EC2 instances** with **Amazon Linux 2**.

**Step 2: IAM Role Setup**

✅ **Create an IAM Role** with the following:

* **Trusted Entity** = EC2
* **Permissions** = CloudWatchAgentServerPolicy

✅ Attach the **IAM Role** to both EC2 instances.

**2. Install CloudWatch Agent**

**Step 3: Install AWS Logs Agent on EC2**

Run the following commands:

sudo -s

yum install -y awslogs

cd /etc/awslogs/

ls

cat awscli.conf

**Step 4: Configure AWS Logs Agent**

✅ Open AWS CLI configuration file:

vi awscli.conf

* Press **i** to enter insert mode.
* Change the **region** to match your AWS region.
* Press **ESC**, type **:wq!**, and press **Enter** to save.

✅ Verify AWS logs configuration:

cat awslogs.conf # No modification required

**Step 5: Start AWS Logs Service**

✅ Start the CloudWatch logs agent service:

systemctl start awslogsd

✅ Enable the service on system reboot:

systemctl enable awslogsd

**3. How Logs Are Structured in CloudWatch**

✅ **Log Group:** Group name for related logs.  
✅ **Log Stream:** Instance ID (logs are stored per instance).

**4. Key Takeaways**

✔️ **EC2 logs can be streamed to CloudWatch Logs.**  
✔️ **IAM Role is mandatory to allow EC2 to write logs.**  
✔️ **Region configuration is required in awscli.conf.**  
✔️ **AWS Logs Agent (awslogs) is needed to send logs to CloudWatch.**  
✔️ **Logs can be monitored in CloudWatch Console under Log Groups.**

