AWS Monitoring, Audit and Performance

Amazon CloudWatch Metrics

- CloudWatch provides metrics for every service in AWS.
- A metric is a variable to monitor, such as `CPUUtilization`, `NetworkIn`, etc.
- Metrics belong to **namespaces**.
- A **dimension** is an attribute of a metric, such as `instance id`, `environment`, etc.
- Each metric can have up to **30 dimensions**.
- Metrics include **timestamps**.
- You can create **CloudWatch Dashboards** to visualize metrics.
- You can also create **CloudWatch Custom Metrics**, for example to monitor RAM usage.

CloudWatch Metric Streams

- Continuously stream CloudWatch metrics to a destination of your choice.
- Offers **near-real-time delivery** with **low latency**.
- Supported destinations:
  - **Amazon Kinesis Data Firehose**, which can forward to:
    - Amazon S3
    - Amazon Redshift
    - Amazon OpenSearch
    - Amazon Athena
  - **Third-party service providers**:
    - Datadog
    - Dynatrace
    - New Relic
    - Splunk
    - Sumo Logic
- You can **filter metrics** to stream only a selected subset.

CloudWatch Logs

- **Log Groups**: Arbitrary name, usually representing an application.
- **Log Streams**: Represent instances within the application, log files, or containers.
- You can define **log expiration policies**, ranging from "never expire" to durations between **1 day and 10 years**.
- CloudWatch Logs can send logs to:
  - **Amazon S3** (exports)
  - **Kinesis Data Streams**
  - **Kinesis Data Firehose**
  - **AWS Lambda**
  - **Amazon OpenSearch**
- Logs are **encrypted by default**.
- You can set up **KMS-based encryption** using your own customer-managed keys.

CloudWatch Logs - Sources

CloudWatch Logs can collect logs from various sources:

- **SDK**, **CloudWatch Logs Agent**, **CloudWatch Unified Agent**
- **Elastic Beanstalk**: Collects logs from applications
- **ECS (Elastic Container Service)**: Collects logs from containers
- **AWS Lambda**: Collects logs from function executions
- **VPC Flow Logs**: Logs specific to Virtual Private Cloud traffic
- **API Gateway**: Logs API requests and responses
- **CloudTrail**: Logs based on filters
- **Route 53**: Logs DNS queries

CloudWatch Logs Insights

- **CloudWatch Logs Insights** is an interactive log analytics tool.
- Allows you to run **queries** on your log data stored in CloudWatch Logs.
- Useful for **real-time application and infrastructure monitoring**.
- You can use a **custom query language** to:
  - Filter log data
  - Aggregate and compute statistics
  - Visualize results directly in the console
- Designed for **fast, interactive exploration** of log data.

CloudWatch Logs Insights

- Search and analyze log data stored in **CloudWatch Logs**.
- Example use cases:
  - Find a specific IP address in logs.
  - Count occurrences of the word “ERROR”.
- Uses a **purpose-built query language**.
- **Automatically discovers fields** from AWS services and JSON log events.
- Capabilities include:
  - Fetching desired event fields.
  - Filtering based on conditions.
  - Calculating aggregate statistics.
  - Sorting events.
  - Limiting the number of returned events.
- You can **save queries** and **add them to CloudWatch Dashboards**.
- Supports querying **multiple Log Groups** across **different AWS accounts**.
- Note: it is a **query engine**, not a **real-time processing engine**.

CloudWatch Logs – S3 Export

- Log data export to **Amazon S3** is supported.
- Exported log data can take up to **12 hours** to become available.
- Export is initiated using the API call `CreateExportTask`.
- This method is **not suitable for near-real-time or real-time use cases**.
- For real-time needs, use **CloudWatch Logs Subscriptions** instead.

CloudWatch Logs Subscriptions

- Enables **real-time delivery** of log events from **CloudWatch Logs** for further processing and analysis.
- You can send log events to:
  - **Kinesis Data Streams**

    - **Kinesis Data Firehose**
    - **AWS Lambda**
- Uses a **Subscription Filter** to control which log events are delivered to the destination.
- Ideal for streaming logs to processing pipelines or storage systems in real-time or near-real-time.

CloudWatch Logs Aggregation – Multi-Account & Multi-Region

- CloudWatch Logs can be aggregated across **multiple AWS accounts** and **regions**.
- Use **CloudWatch Logs Subscription Filters** in each account and region to forward logs.
- Logs can be streamed in **near real-time** to centralized destinations like:
    - **Kinesis Data Streams**
    - **Kinesis Data Firehose**
    - **Amazon S3**

Example Setup:

- **Account A - Region 1**
- **Account B - Region 2**
- **Account B - Region 3**

Each region/account sends logs using subscription filters to a centralized logging destination.

CloudWatch Logs Subscriptions – Cross-Account

- CloudWatch Logs supports **cross-account subscriptions**, allowing you to send log events to resources in a **different AWS account**.
- Typically used to send logs to:
    - **Kinesis Data Streams (KDS)**
    - **Kinesis Data Firehose (KDF)**

Example Architecture:

Account – Sender (111111111111)

- Contains the **CloudWatch Logs**.
- A **Subscription Filter** defines which logs to forward.

Account – Recipient (999999999999)

- Hosts the destination, e.g., **Kinesis Data Stream (RecipientStream)**.
- Requires a **Subscription Destination** setup.

Permissions:

- **IAM Role** (Cross-Account) in sender account:
    - Can be assumed by CloudWatch Logs.
    - Must have permission to `PutRecord` to the destination stream.
- **Destination Access Policy** in recipient account:
    - Allows the sender account to put data into the stream.

CloudWatch Logs for EC2

- **By default**, EC2 instances do **not send logs** to CloudWatch Logs.
- To enable logging, you must **run a CloudWatch Logs agent** on the EC2 instance.
- The agent is responsible for **pushing desired log files** to CloudWatch.

- Ensure that the EC2 instance has the **correct IAM permissions** to publish logs.
- The **CloudWatch Logs agent** can also be installed on **on-premise servers** to send logs to AWS CloudWatch.

CloudWatch Logs Agent & Unified Agent

- Designed for **virtual servers**, including **EC2 instances** and **on-premises servers**.

CloudWatch Logs Agent

- **Legacy agent**.
- Can **only send logs** to CloudWatch Logs.

CloudWatch Unified Agent

- **Modern agent** with extended capabilities.
- Collects:
  - **System-level metrics** (e.g., RAM usage, running processes).
  - **Logs** for CloudWatch Logs.
- Supports **centralized configuration** via **SSM Parameter Store**.

CloudWatch Unified Agent – Metrics

- Metrics are collected **directly on your Linux server or EC2 instance**.

Metrics Collected:

- **CPU**: active, guest, idle, system, user, steal
- **Disk**:
  - Usage: free, used, total
  - IO: writes, reads, bytes, IOPS
- **RAM**: free, inactive, used, total, cached
- **Netstat**:
  - Number of TCP and UDP connections
  - Network packets and bytes
- **Processes**: total, dead, blocked, idle, running, sleeping
- **Swap Space**: free, used, used percentage

> *Reminder: EC2 instances already provide **high-level metrics** (disk, CPU, network) out of the box.*

CloudWatch Alarms

- **Alarms** are used to trigger **notifications** based on metric thresholds.
- Support various comparison options: sampling, percentage, max, min, etc.

Alarm States:

- **OK**: Metric is within defined threshold.
- **INSUFFICIENT_DATA**: Not enough data to determine the state.
- **ALARM**: Metric is outside the defined threshold.

Period:

- The **evaluation interval** (in seconds) for the metric.
- For **high-resolution custom metrics**, allowed values are:
  - **10 seconds**

    - **30 seconds**
    - Multiples of **60 seconds**

CloudWatch Alarm Targets

CloudWatch Alarms can trigger various actions when a metric crosses a threshold:

- **Amazon EC2**:
    - **Stop** an instance
    - **Terminate** an instance
    - **Reboot** an instance
    - **Recover** an instance (if supported)
- **EC2 Auto Scaling**:
    - Trigger an **Auto Scaling Action**
- **Amazon SNS**:
    - Send a **notification to an SNS topic**
    - SNS can then trigger actions such as Lambda functions, emails, HTTP/S endpoints, etc.

CloudWatch Alarms – Composite Alarms

- **Standard CloudWatch Alarms** monitor a **single metric**.
- **Composite Alarms** monitor the **states of multiple other alarms**.
- Use **logical conditions**:
    - **AND**
    - **OR**
- Useful for reducing **alarm noise** by triggering notifications only when **combinations of alarms** are in ALARM state.
- Composite Alarms can trigger actions such as sending notifications via **Amazon SNS**.

EC2 Instance Recovery

- **CloudWatch Alarm** can monitor the status check `StatusCheckFailed_System` for EC2.

Status Checks:

- **Instance Status**: Checks the health of the EC2 virtual machine.
- **System Status**: Checks the health of the underlying **hardware**.
- **Attached EBS Status**: Verifies the health of **attached EBS volumes**.

EC2 Recovery:

- Automatically recovers the EC2 instance if a system failure is detected.
- After recovery, the instance retains:
    - **Same Private IP**
    - **Same Public IP**
    - **Same Elastic IP**
    - **Instance metadata**
    - **Placement group**

- Can trigger a **notification via SNS Topic** when recovery occurs.

CloudWatch Alarm: Good to Know

- Alarms can be created based on **CloudWatch Logs Metric Filters**.
- Useful for triggering alerts from specific log patterns or values.

Testing Alarms via CLI:

You can manually set an alarm to `ALARM` state to test notifications:

```
aws cloudwatch set-alarm-state \
  --alarm-name "myalarm" \
  --state-value ALARM \
  --state-reason "testing purposes"
```

- This allows you to verify alerting mechanisms such as **Amazon SNS** subscriptions.

Amazon EventBridge (formerly CloudWatch Events)

- **EventBridge** allows you to respond to events in AWS services or schedule automated tasks.

Two Main Use Cases:

1. **Schedule (Cron Jobs)**:
    - Define cron expressions to run scheduled scripts or functions.
    - Example: Trigger a Lambda function **every hour**.
2. **Event Pattern**:
    - Define rules to **react to specific AWS service events**.
    - Example: Trigger an alert when the **root user signs in**.

Actions:

- **Trigger Lambda functions**
- **Send messages to SQS queues**
- **Send notifications via SNS topics**

Amazon EventBridge Rules

Example Sources:

EventBridge can capture events from many AWS services and scheduled tasks, such as:

- **EC2 Instance** (e.g., start or stop events)
- **CodeBuild** (e.g., failed build)
- **S3** (e.g., object upload)
- **Trusted Advisor** (e.g., new finding)
- **CloudTrail** (e.g., any API call)
- **Schedule or Cron** (e.g., every 4 hours)

Event Flow:

1. **Event Source** generates a JSON event.

2. EventBridge can **filter events** based on patterns.
3. Matching events are forwarded to **target services**.

Example Destinations:

- **Compute**:
  - AWS Lambda
  - AWS Batch
  - ECS Task
- **Integration**:
  - SQS
  - SNS
  - Kinesis Data Streams
- **Orchestration**:
  - Step Functions
  - CodePipeline
  - CodeBuild
- **Maintenance**:
  - SSM
  - EC2 Actions

Amazon EventBridge

- **Event Buses** are used to receive and route events.
- Types of event buses:
  - **Default Event Bus**: receives events from AWS services.
  - **Partner Event Bus**: receives events from AWS SaaS partners.
  - **Custom Event Bus**: receives events from custom applications.

Key Features:

- **Cross-account access**:
  - Other AWS accounts can send events to your event bus using **resource-based policies**.
- **Event Archiving**:
  - You can archive **all or filtered events** sent to an event bus.
  - Archives can be kept **indefinitely or for a specific retention period**.
- **Event Replay**:
  - Archived events can be **replayed** to help with debugging or reprocessing.

Amazon EventBridge – Schema Registry

- **EventBridge** can analyze the events flowing through your event bus and **infer their schema**.

Schema Registry Features:

- Maintains a **repository of schemas** for events.

- Supports **schema versioning**, allowing you to track changes over time.
- Enables **code generation** for applications, so they can:
  - Understand the **structure of the event data**.
  - Easily deserialize and process events.

Amazon EventBridge – Resource-based Policy

- Use **resource-based policies** to **manage permissions** for a specific **EventBridge event bus**.

Key Capabilities:

- **Allow or deny** events from:
  - Other **AWS accounts**
  - Other **AWS regions**

Use Case:

- **Centralize event processing** by aggregating all events from your **AWS Organization** into a **single account or region**.

Example:

- AWS Account `123456789012` sends events to the **central event bus** in Account `111122223333`.
- The **PutEvents** action must be permitted by the **resource-based policy** on the central bus.

CloudWatch Container Insights

- **CloudWatch Container Insights** helps you **collect, aggregate, and summarize** metrics and logs from containers.

Supported Platforms:

- **Amazon ECS** (Elastic Container Service)
- **Amazon EKS** (Elastic Kubernetes Service)
- **Kubernetes platforms on EC2**
- **AWS Fargate** (for both ECS and EKS)

How It Works:

- In **Amazon EKS** and **Kubernetes**, CloudWatch Insights uses a **containerized CloudWatch Agent** to automatically discover containers and collect telemetry data.

CloudWatch Lambda Insights

- A **monitoring and troubleshooting solution** for **serverless applications** running on **AWS Lambda**.

Features:

- Collects, aggregates, and summarizes **system-level metrics**, including:
  - CPU time
  - Memory usage
  - Disk I/O
  - Network usage
- Collects, aggregates, and summarizes **diagnostic information**, such as:
  - **Cold starts**

    - **Lambda worker shutdowns**
- **Lambda Insights** is delivered as a **Lambda Layer** that must be enabled for your functions.

CloudWatch Contributor Insights

- Allows you to **analyze log data** and create **time series** that display contributor data.
- Helps identify:
    - **Top-N contributors**
    - **Total number of unique contributors**
    - Their **usage patterns**

Benefits:

- Understand **who or what is impacting system performance**.
- Detect:
    - **Bad hosts**
    - **Heaviest network users**
    - **URLs causing the most errors**

Supported Log Sources:

- Works with **any AWS-generated logs**, including:
    - **VPC Flow Logs**
    - **DNS logs**

Configuration:

- You can:
    - Build **custom rules**
    - Use **AWS-provided sample rules**
    - Use **built-in rules** for analyzing metrics from other AWS services
- Leverages existing **CloudWatch Logs** to generate insights.

CloudWatch Application Insights

- Provides **automated dashboards** highlighting **potential problems** with monitored applications.
- Helps **isolate and troubleshoot ongoing issues** faster.

Supported Application Environments:

- Runs on **Amazon EC2 instances** with specific technologies:
    - **Java**
    - **.NET**
    - **Microsoft IIS Web Server**
    - **Databases**
- Can monitor a wide range of **AWS resources**, including:
    - **Amazon EBS**
    - **Amazon RDS**
    - **Elastic Load Balancing (ELB)**
    - **Auto Scaling Groups (ASG)**

    - **Lambda**
    - **SQS**
    - **DynamoDB**
    - **S3 buckets**
    - **ECS**
    - **EKS**
    - **SNS**
    - **API Gateway**

Additional Features:

- Powered by **Amazon SageMaker** for intelligent insights.
- Improves visibility into application health.
- Reduces time to **troubleshoot and repair** applications.
- Sends findings and alerts to:
    - **Amazon EventBridge**
    - **AWS Systems Manager OpsCenter**

CloudWatch Insights and Operational Visibility

A set of CloudWatch tools providing deep visibility and operational insights into various AWS environments:

CloudWatch Container Insights

- Supported on:
    - **ECS**
    - **EKS**
    - **Kubernetes on EC2**
    - **Fargate**
- Requires an **agent** for Kubernetes.
- Collects **metrics and logs** from containers.

CloudWatch Lambda Insights

- Provides **detailed metrics** to help **troubleshoot serverless applications**.

CloudWatch Contributor Insights

- Identifies **Top-N contributors** by analyzing **CloudWatch Logs**.
- Useful for spotting heavy resource users and system performance issues.

CloudWatch Application Insights

- Delivers **automated dashboards** to help **troubleshoot applications** and associated **AWS services**.

AWS CloudTrail

- Provides **governance**, **compliance**, and **audit capabilities** for your AWS account.
- **CloudTrail is enabled by default**.

Features:

- Records the **history of events and API calls** made in your AWS account via:
    - AWS **Management Console**
    - AWS **SDKs**

    - AWS **CLI**
    - **AWS services** themselves
- Logs can be sent to:
    - **CloudWatch Logs**
    - **Amazon S3**

Trail Scope:

- A trail can be created for:
    - **All Regions** (default)
    - A **single Region**

> *Tip: If a resource is unexpectedly deleted, check **CloudTrail** to investigate the cause.*

CloudTrail – Architecture Overview

Sources of Events:

- AWS **SDK**
- AWS **CLI**
- AWS **Management Console**

These interactions generate events that are recorded by **CloudTrail**.

Destinations:

- **CloudWatch Logs**: for real-time monitoring and alerting.
- **Amazon S3 Bucket**: for long-term storage and auditing.

Users and Roles:

- Actions by **IAM Users** and **IAM Roles** are captured for auditing purposes.

Usage:

- Use the **CloudTrail Console** to **inspect and audit** recorded API activity across your AWS environment.

CloudTrail Events

1. Management Events

- Represent **operations on AWS resources**.
- Examples:
    - **IAM**: AttachRolePolicy
    - **EC2**: CreateSubnet
    - **CloudTrail**: CreateTrail
- By default, **trails are configured to log management events**.
- You can separate:
    - **Read events** (non-modifying)
    - **Write events** (modifying)

2. Data Events

- **Not logged by default** due to potentially high volume.
- Examples:

    - **Amazon S3**: Object-level activity (e.g., GetObject, PutObject, DeleteObject)
    - **AWS Lambda**: Function execution activity (Invoke API)
- Can also separate **Read** and **Write** operations.

3. CloudTrail Insights Events

- Special events for identifying **unusual operational activity** (see next slide).

CloudTrail Insights

- **CloudTrail Insights** helps detect **unusual activity** in your AWS account.

Examples of Unusual Activity:

- Inaccurate **resource provisioning**
- **Hitting service limits**
- **Bursts of AWS IAM actions**
- Gaps in **periodic maintenance activity**

How It Works:

- Analyzes **normal management events** to establish a **baseline**.
- Continuously monitors **write events** to detect **anomalous behavior**.

When Anomalies Are Detected:

- They are **visible in the CloudTrail console**.
- An **event is sent to Amazon S3**.
- An **Amazon EventBridge event is generated**, which can be used to trigger **automated responses**.

CloudTrail Events Retention

- **CloudTrail stores events for 90 days** by default.
- To **retain events longer**, you should:
    - Configure CloudTrail to **log events to an Amazon S3 bucket**.
    - Use **Amazon Athena** to analyze the logs stored in S3.

Event Types:

- **Management Events**
- **Data Events**
- **Insights Events**

All event types can be archived for **long-term retention** in S3 and queried with Athena.

Amazon EventBridge – Intercept API Calls

- **Amazon EventBridge** can be used in combination with **CloudTrail** to intercept and respond to **specific API calls**.

Example Use Case:

- A **DeleteTable API call** on **DynamoDB** is detected by **CloudTrail**.
- CloudTrail sends the event to **EventBridge**.
- EventBridge can then:
    - **Trigger an alert** via **SNS**.
    - **Log the event** in **DynamoDB** or another destination.

Benefits:

- Enables real-time monitoring and automated responses to sensitive or critical operations in your AWS environment.

Amazon EventBridge + CloudTrail

- **CloudTrail** captures API calls made by IAM users or roles.
- **EventBridge** consumes these events to automate responses.

Example Workflow:

1. **IAM Role Assumption**:
    - A user assumes an **IAM role**.
    - CloudTrail logs the **AssumeRole** API call.
    - Event is passed to **EventBridge**, which can trigger **SNS notifications**.
2. **EC2 Security Group Change**:
    - A user modifies a security group using the **AuthorizeSecurityGroupIngress** API call.
    - CloudTrail logs the API call.
    - EventBridge captures the event and can:
        - Notify via **SNS**
        - Log the event
        - Trigger automated remediation actions

> *This integration is powerful for enforcing **security policies** and **automating incident response**.*

AWS Config

- Helps with **auditing**, **compliance**, and **recording configuration changes** of your AWS resources over time.

Use Cases:

- Detect **unrestricted SSH access** in security groups.
- Check if **S3 buckets have public access**.
- Track **changes to ALB (Application Load Balancer) configuration**.

Features:

- Records **configurations and changes** of AWS resources.
- Sends **SNS notifications** for any detected changes.
- **Per-region service**, but can be **aggregated across regions and accounts**.
- Configuration data can be stored in **Amazon S3** and queried with **Amazon Athena**.

AWS Config Rules

- Used to evaluate the **compliance** of AWS resource configurations.

Types of Rules:

- **AWS Managed Rules**: Over 75 pre-defined rules provided by AWS.
- **Custom Rules**: Defined via **AWS Lambda** functions.

Example Use Cases:

- Check if all **EBS volumes** are of type `gp2`.

- Verify that all **EC2 instances** are of type `t2.micro`.

Evaluation Triggers:

- **On every configuration change**
- **At regular time intervals**

> *Note: AWS Config Rules **do not prevent actions** from happening. They only **report compliance**.*

Pricing:

- No free tier.
- **$0.003** per configuration item recorded (per region).
- **$0.001** per config rule evaluation (per region).

AWS Config Resource

For each AWS resource, AWS Config allows you to:

- **View compliance status over time**
  - Track whether the resource was compliant or non-compliant with your config rules.
- **View configuration history over time**
  - See how the configuration of the resource has changed.
- **View associated CloudTrail API calls**
  - Understand what actions were performed on the resource and when.

Config Rules – Remediations

- Allows **automated remediation** of **non-compliant resources** using **SSM Automation Documents**.

Options:

- Use **AWS-Managed Automation Documents**.
- Create **Custom Automation Documents**.
  - Example: documents that invoke a **Lambda function** to perform the remediation.

Features:

- **Remediation Retries** can be configured to retry the action if the resource remains non-compliant.
- Helps enforce compliance automatically by taking **corrective actions** when violations are detected.

Example:

- If an **IAM Access Key** is **expired** (non-compliant), AWS Config can:
  - Trigger an **SSM Document** (e.g., `AWSConfigRemediation-RevokeUnusedIAMUserCredentials`)
  - Automatically **deactivate** the access key
  - Retry remediation up to a specified number of times (e.g., 5 retries)

Config Rules – Notifications

- Use **Amazon EventBridge** to trigger **notifications** when AWS resources become **non-compliant**.

Notification Capabilities:

- **Send notifications** about:
  - **Configuration changes**
  - **Compliance state changes**
- Notifications can be routed to:
  - **SNS**
  - **SQS**
  - **Lambda**

Filtering:

- All events can be sent to **SNS**.
- Use **SNS filtering** or perform **filtering client-side** to manage which events trigger actions.

Example:

- When a **security group** becomes **NON_COMPLIANT**, AWS Config:
  - Sends the event to **EventBridge**
  - EventBridge triggers a **Lambda**, **SNS**, or **SQS** action
  - Admin is notified of the issue

CloudWatch vs CloudTrail vs Config

CloudWatch

- **Performance monitoring**:
  - Metrics: CPU, network, etc.
  - Dashboards for visualization
- **Events & alerting**
- **Log aggregation & analysis**

CloudTrail

- **Records API calls** made within your AWS account
- Logs activity from:
  - Console
  - SDK
  - CLI
  - AWS Services
- You can define **trails for specific resources**
- **Global service**

AWS Config

- **Records configuration changes** of AWS resources
- **Evaluates compliance** using rules
- Provides a **timeline of changes and compliance history**

For an Elastic Load Balancer

CloudWatch

- **Monitor metrics** like incoming connections.
- **Visualize error codes** over time (e.g., as a percentage).
- Create **dashboards** to assess load balancer performance.

AWS Config

- Track **security group rules** associated with the Load Balancer.
- Monitor **configuration changes** to the Load Balancer.
- Ensure **compliance** by verifying that an **SSL certificate** is always assigned.

CloudTrail

- Track **who made changes** to the Load Balancer using **API call logs**.