# CloudWatch Monitoring Dashboard Guide

Log Processor deploys a comprehensive CloudWatch dashboard that provides real-time visibility into every component of the pipeline. The dashboard is automatically created when alarms are enabled.

> To access: AWS Console → CloudWatch → Dashboards → `<stackname>-monitoring`

## Alarm Status

The top row shows the current state of all configured alarms in a single status widget. Green = OK, red = in alarm, gray = insufficient data. Alarms cover:

- Lambda errors and throttles
- Lambda concurrency (80% of reserved)
- Dead letter queue depth (chunk DLQ and S3 event DLQ)
- Firehose delivery freshness
- OpenSearch cluster status (red, yellow for multi-node)
- OpenSearch free storage, JVM pressure, CPU
- OpenSearch Dashboards healthy nodes

All alarms notify the configured SNS email when triggered and when they return to OK.

## Lambda

Three widgets monitoring the log processor Lambda function:

- **Invocations & Errors** – Total invocations vs errors over time. A spike in errors indicates processing failures.
- **Throttles & Concurrency** – Throttle count (left axis) and concurrent executions (right axis). If concurrency approaches the reserved limit, increase `LambdaConcurrencyValidation`.
- **Duration & Memory** – Average and maximum execution duration. If max duration approaches 15 minutes, the function may be timing out on large S3 objects.

## SQS Queues

Four widgets showing queue health:

- **S3 Event Queue** – Messages visible and sent. Spikes indicate new Firehose deliveries arriving.
- **Chunk Queue** – Messages visible and sent. Shows the volume of log chunks being processed.
- **Dead Letter Queues** – Messages in both DLQs. Any non-zero value means messages failed processing 3 times and need investigation.
- **Oldest Message Age** – How long the oldest message has been waiting. High values indicate a processing backlog.

## Firehose

Three widgets monitoring the Kinesis Firehose delivery stream:

- **Delivery Freshness** – Time since the last successful delivery to S3. Should stay below the buffer interval (default 60s). High values mean delivery is backing up.
- **Incoming Records & Bytes** – Volume of data entering Firehose from CloudWatch subscription filters.
- **Delivery Success** – Percentage of successful deliveries to S3. Should be 1.0 (100%).

## Log Processor Metrics

Custom EMF (Embedded Metric Format) metrics published by the Lambda:

- **Events Processed** – S3Events (Firehose objects processed), SqsEnqueued (chunks sent to queue), SqsEvents (chunks consumed). These should track together.
- **Errors & Retries** – Processing errors, warnings, and OpenSearch retry count. OsRetries indicates bulk indexing failures that were re-enqueued.
- **Entitlement & Gauges** – Marketplace entitlement status (invalid/dropped counts) and gauges for active subscriptions and Firehose error objects.
- **Pattern Detection** – Events with detected patterns and events filtered by PII scanning. Only active when pattern mode is enabled on subscriptions.

## OpenSearch

Seven widgets monitoring the OpenSearch domain (only shown when OpenSearch is enabled):

- **Cluster Status** – Green (all shards assigned), yellow (replicas unassigned), red (primary shards unassigned). Single-node clusters are always yellow – this is normal.
- **Storage (MB)** – Free storage space and cluster used space. Monitor free space to avoid running out of disk.
- **JVM Memory Pressure** – Percentage of JVM heap used. Sustained values above 80% indicate the cluster needs larger instance types.
- **CPU Utilization** – Cluster CPU usage. Sustained values above 80% indicate the cluster needs more nodes or larger instances.
- **Indexing Rate** – Documents indexed per second. Correlates with log ingestion volume.
- **Search Rate** – Searches per second. Reflects Dashboards usage and saved search activity.
- **Dashboards Healthy Nodes** – Number of healthy OpenSearch Dashboards instances. Should always be at least 1.

## Logs — Audit & App

Per-index-type breakdown showing events processed, OpenSearch retries, indexing rate, and document count for both `audit` and `app` index types. Use these to compare ingestion volume and health between index types.

## Nginx Proxy

Three widgets monitoring the Nginx reverse proxy EC2 instance (only shown when dashboards are enabled):

- **CPU** – User and system CPU usage on the Nginx instance.
- **Memory** – Memory utilization percentage.
- **Disk** – Disk usage percentage. Should stay well below 100% with log rotation enabled.

## Lambda Logs

Two log query widgets showing recent log entries from the Lambda function:

- **Recent Error(s)** – Last 50 log lines containing ERROR. Check here first when investigating failures.
- **Recent Warning(s)** – Last 50 log lines containing WARNING. Warnings include subscription issues, pattern scan problems, and DLQ messages.

## Estimated Costs (USD)

Six widgets showing AWS billing estimates by service category. These use the `AWS/Billing` namespace and require billing alerts to be enabled in the AWS Billing console (Settings → Billing Preferences → Receive Billing Alerts).

- **Account Total** – Total estimated charges across all AWS services.
- **Compute & Processing** – OpenSearch, Lambda, EC2.
- **Storage & Delivery** – S3, Kinesis Firehose, SQS.
- **Security & Management** – KMS, DynamoDB, Cognito.
- **Networking** – Elastic Load Balancing, VPC.
- **Monitoring & Analytics** – CloudWatch, Athena, Glue.

**Note:** Billing metrics update approximately every 6 hours and are only available in us-east-1. Estimates reflect account-wide charges, not just this stack.

## Tips

- **Set the time range** – Use the time picker in the top-right corner. Start with 1h or 3h for real-time monitoring, 1d or 1w for trends.
- **Auto-refresh** – Enable auto-refresh (top-right) for live monitoring. 1-minute interval recommended.
- **Drill into alarms** – Click any alarm in the status widget to see its history and configuration.
- **DLQ investigation** – If DLQ depth is non-zero, check the Lambda Logs section for ERROR messages. Common causes: OpenSearch cluster full, malformed log data, or Lambda timeout.
- **Scaling indicators** – If Lambda concurrency is consistently above 80%, increase the concurrency limit. If OpenSearch JVM or CPU is consistently above 80%, upgrade the instance type or add nodes.
- **Cost optimization** – Use the billing widgets to identify the most expensive services. OpenSearch instances and EBS volumes are typically the largest cost drivers.