

---

# AWS Cloudwatch

RIO CLOUD  
COMPUTING  
MEETUP



Felipe Almeida  
Rio Cloud Computing Meetup October / 2016

---

## Monitoring your AWS Services with Metrics and Alarms

---

# Structure

---

- Intro
- Usage
- Metrics - Builtin
- Metrics - Custom
- Alarms
- Events
- Logs
- Third-party services
- Keep in mind

# Intro

---

- **CloudWatch** is the AWS metric collection service
- **Metrics** are individual *measurements* of some quantity of interest, at a given *point in time*:

Examples:

- At 23:45:29 GMT, on 23 February 2016, CPU usage for instance XPTO was 56%

# Intro

---

- **Nearly all** AWS services can be monitored using CloudWatch
- Metrics provide **visibility** into your applications
- They enable you to make informed decisions such as
  - Whether to downscale a machine that's not being used too much (save \$\$)
  - Whether to scale up a machine that's close to full capacity (avoid failures)
  - Decide what caused an application to fail (debug)

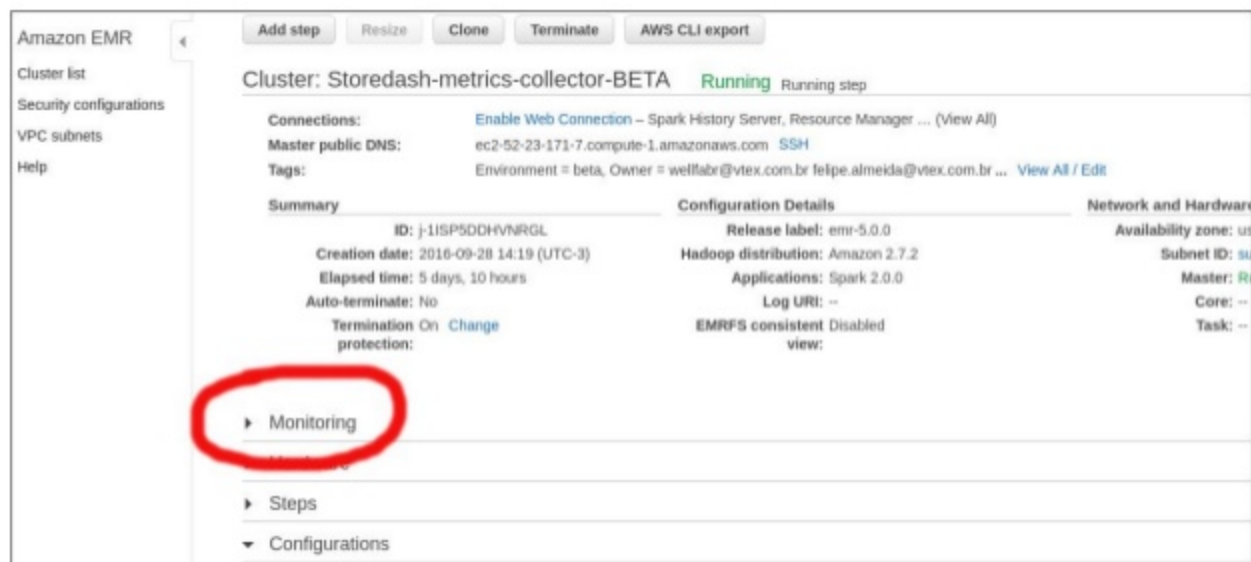
# Usage

---

- You can use CloudWatch in multiple ways:
  - Using the AWS Console
  - Via other services that have monitoring functionality
  - Using the CloudWatch API through the AWS SDK

# Usage - Via other services

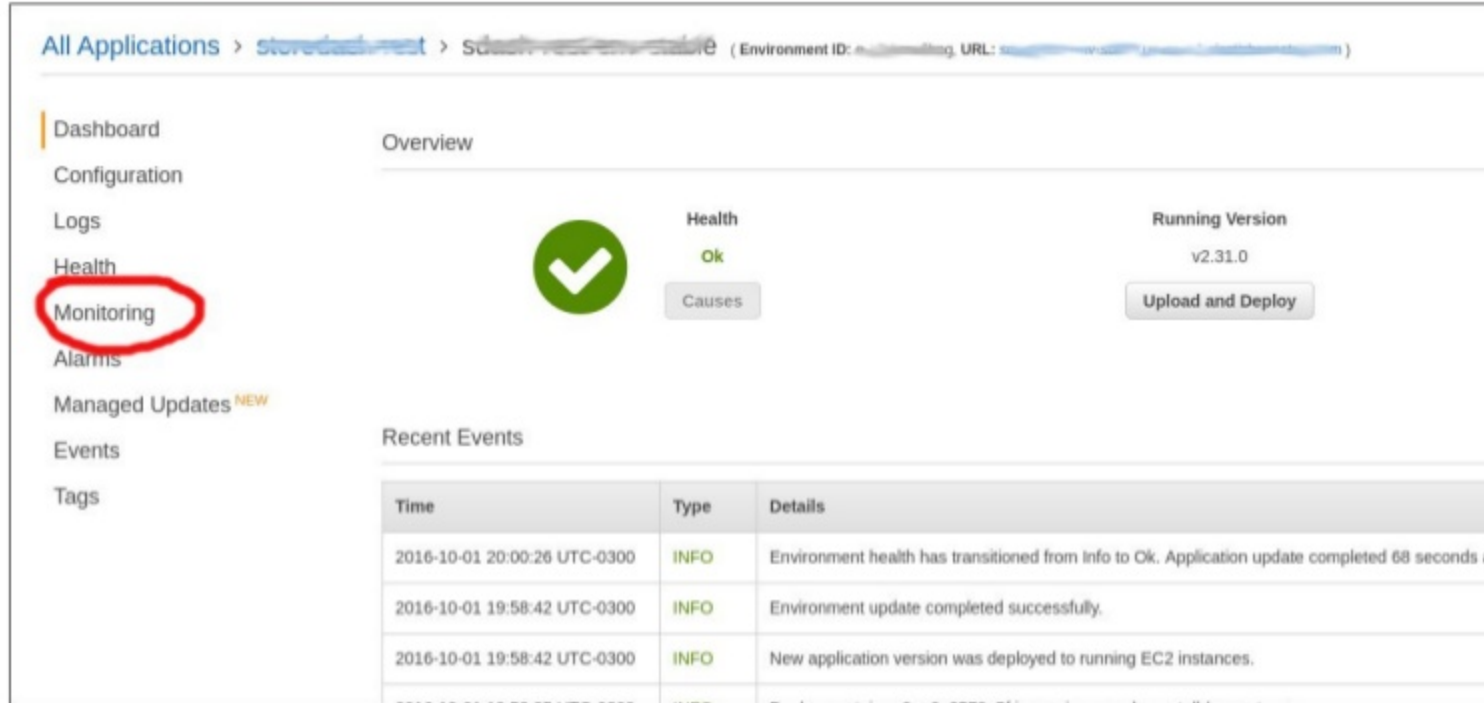
- Via other services
  - Nearly all services have a *monitoring* or *metrics* tab



The screenshot displays the Amazon EMR console interface. On the left, a navigation menu includes 'Cluster list', 'Security configurations', 'VPC subnets', and 'Help'. The main content area shows details for a cluster named 'Storedash-metrics-collector-BETA' in a 'Running' state. At the top of this area are buttons for 'Add step', 'Resize', 'Clone', 'Terminate', and 'AWS CLI export'. Below these, there are sections for 'Connections', 'Master public DNS', and 'Tags'. A 'Summary' section provides details like ID, creation date, and termination protection. To the right, 'Configuration Details' and 'Network and Hardware' are listed. At the bottom, a list of tabs is visible: 'Monitoring' (highlighted with a red circle), 'Steps', and 'Configurations'.

Summary	Configuration Details	Network and Hardware
ID: j-1ISP5DDHVNRLG	Release label: emr-5.0.0	Availability zone: us-east-1a
Creation date: 2016-09-28 14:19 (UTC-3)	Hadoop distribution: Amazon 2.7.2	Subnet ID: subnet-1a
Elapsed time: 5 days, 10 hours	Applications: Spark 2.0.0	Master: Running
Auto-terminate: No	Log URI: --	Core: --
Termination On Change protection: <a href="#">Change</a>	EMRFS consistent view: Disabled	Task: --

# Usage - Via other services



The screenshot shows the Elastic Beanstalk console interface. The left sidebar contains a list of navigation options: Dashboard, Configuration, Logs, Health, Monitoring (highlighted with a red circle), Alarms, Managed Updates <sup>NEW</sup>, Events, and Tags. The main content area is titled 'Overview' and displays the application's status. A large green checkmark icon indicates the application is healthy. The 'Health' status is 'Ok', and the 'Running Version' is 'v2.31.0'. There is a 'Causes' button and an 'Upload and Deploy' button. Below this, the 'Recent Events' section shows a table of events.

Time	Type	Details
2016-10-01 20:00:26 UTC-0300	INFO	Environment health has transitioned from info to Ok. Application update completed 68 seconds ago.
2016-10-01 19:58:42 UTC-0300	INFO	Environment update completed successfully.
2016-10-01 19:58:42 UTC-0300	INFO	New application version was deployed to running EC2 instances.

# Usage - Via other services

The screenshot shows the AWS Management Console interface for a DynamoDB table. On the left, the 'DynamoDB' sidebar is visible with options for 'Dashboard', 'Tables', and 'Reserved capacity'. The 'Tables' section is active, showing a list of tables. The table 'Event.ItemDeliver' is selected and highlighted in blue. The main panel displays the 'Metrics' tab for this table, which is circled in red. The 'Metrics' tab shows 'Recent alerts' (No CloudWatch alarms have been triggered for this table) and 'Stream details' (Stream enabled: No, View type: -, Latest stream ARN: -, Manage Stream button). Below this, the 'Table details' section provides information about the table 'Event.ItemDeliver'.

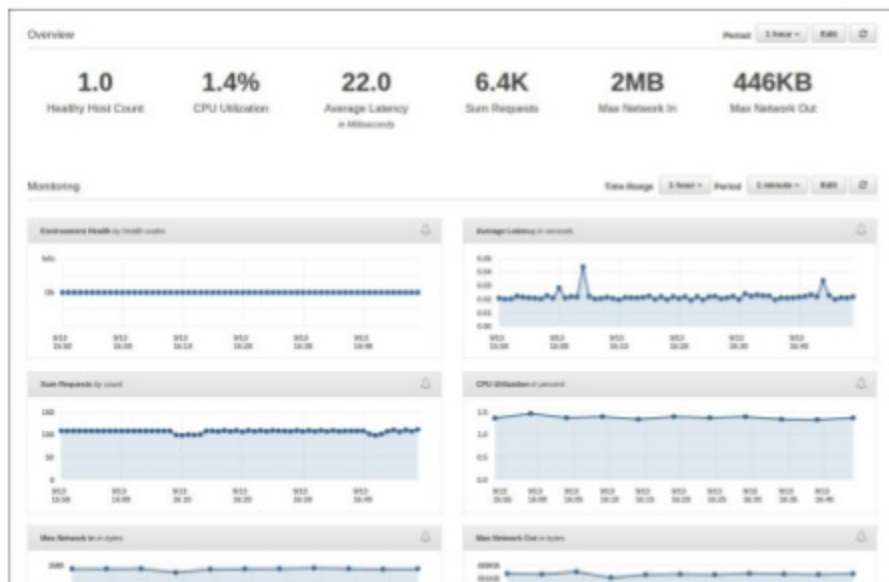
Table details	
Table name	Event.ItemDeliver
Primary partition key	Key (String)
Primary sort key	Range (String)
Table status	Active
Creation date	October 22, 2013 at 6:00 AM
Provisioned read capacity units	1
Provisioned write capacity units	1
Last decrease time	September 30, 2016 at 10:00 AM
Last increase time	September 30, 2016 at 10:00 AM
Storage size (in bytes)	527.72 MB
Item count	2,335,597
Region	US East (N. Virginia)

DynamoDB Table



# Usage - Via other services

- When you open the monitoring tab for a service, what you're looking at are actually CloudWatch Metrics:

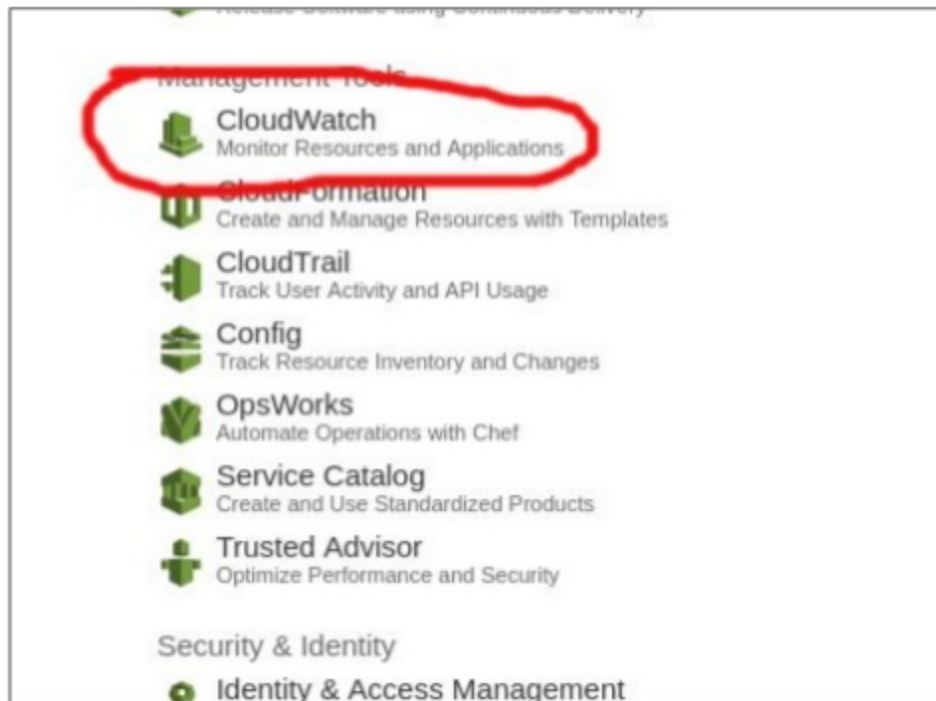


Monitoring Tab for an  
Elastic Beanstalk  
Environment uses  
CloudWatch Metrics

# Usage - Console

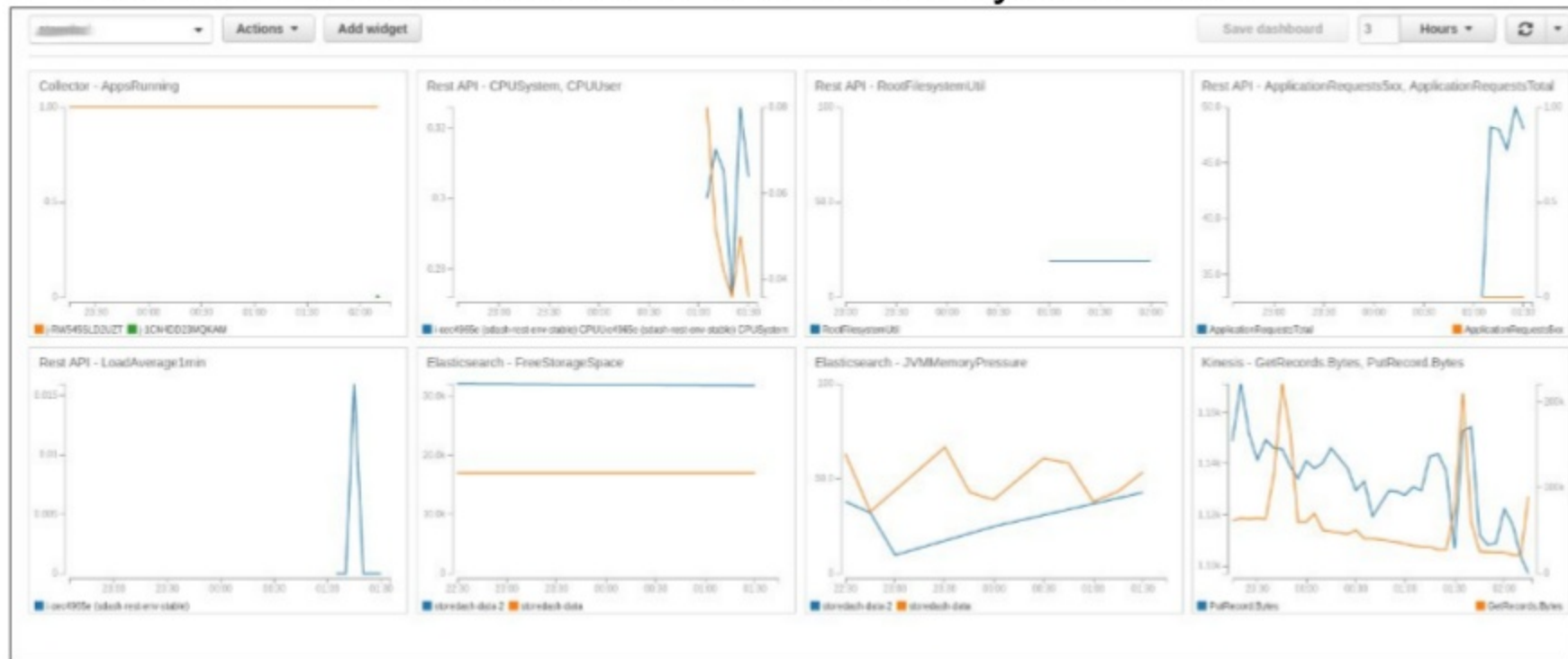
---

- Via cloudwatch itself (on the console)



# Usage - Console

- You can build a Dashboard with metrics you choose



# Usage - API

---

- CloudWatch (like most AWS Services) can also be used **programmatically**
  - I.e. it has an API that you can use via the AWS SDK (library for Java, .NET, Python, etc)

# Metrics - Builtin

---

- Services usually have a set of **builtin metrics**
  - These metrics are collected by default

# Metrics - Builtin

---

- For example, the following metrics (not exhaustive list) are collected by default for EC2 Instances:
  - CPUUtilization (percent)
  - DiskReadBytes (bytes)
  - DiskWriteBytes (bytes)
  - NetworkIn (bytes)
  - NetworkOut (bytes)

# Metrics - Custom

---

- You can also use CloudWatch for your **own** metrics.
- You can then do **anything** you can for regular (builtin) metrics, such as:
  - View them on the AWS console
  - Create Dashboards for them
  - Set up alerts

# Metrics - Custom

---

- For some types of custom metrics, there are **community-provided scripts** to help you publish them to CloudWatch, such as:
  - Memory utilization for EC2 Instances
  - Swap Utilization for EC2 Instances
  - Disk Space Usage for EC2 Instances
- These metrics are controlled by the O.S. and therefore cannot be accessed automatically by AWS.



# Alarms

---

- You can also create **alarms** for any individual metric on CloudWatch, which are triggered if the criteria you have defined are met
- For example, you can have AWS **send you an e-mail** if `FreeStorageSpace` metric for a `ElasticSearch` cluster you have becomes lower than 10GB.

# Alarms

---

- In addition to notifying you when something happens, you can also configure **AutoScaling Actions** to be taken if some criteria are met:
  - For example, if the `CPUUtilization` metric reaches 90% for all machines in a AutoScaling Group, add extra instances to that environment

# Events

---

- CloudWatch Events are a new addition in CloudWatch
- They enable you to execute custom actions in response to State Changes such as
  - An EC2 instance has been started
  - An EC2 instance has been terminated
- Custom actions include running a Lambda Function, publishing the event to Kinesis, etc.

# Logs

---

- CloudWatch Logs refers to Agents you can install on your instances to have them send application logs to CloudWatch
- You can filter your logs and set up Alarms when certain criteria are met, such as:
  - There have been more than 10 Errors in Apache in the last minute
  - There have been 5 log messages matching `IllegalArgumentException` in the last hour

## Related Services (Third-Party)

---

- There are tons of **third-party** services that build on top of or otherwise leverage CloudWatch metrics, such as:
  - **SignalFX** (extends and sends custom metrics to CloudWatch)
  - **AppDynamics** (sends all sorts of low-level and application-level metrics to CloudWatch and displays them in a unified manner; also provides event correlation)
  - **NewRelic** (sends data to CloudWatch)
  - **Grafana** (uses CloudWatch as a Data source)
  - **Logstash** (you can use CloudWatch as an output)
  - **Nagios** (consumes CloudWatch metrics)

## Keep in mind

---

- Metrics last 2 weeks by default
- Metrics are per-region
- Some services (e.g. Elastic BeanStalk) have extra metrics that need to be explicitly enabled before they can be used.
- CloudWatch can also be used to monitor AWS Costs (Billing), even though it's not a service per se

## Keep in mind

---

- In order to publish custom Amazon CloudWatch metrics, the instances in your environment need **permission** to use CloudWatch (see your instance profile for more information).
- CloudWatch is **not free** so you may want to track only relevant metrics
  - In addition, if you use CloudWatch via the AWS SDK, try to collect high-level metrics to keep cost down
    - E.g. track BeanStalk metrics rather than metrics for individual instances

# Links

---

- [Monitoring Scripts for Linux EC2 Instances](#)
- [All Builtin Metrics for all supported AWS services](#)
- [AWS CloudWatch Pricing](#)