AWS Cloudwatch

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

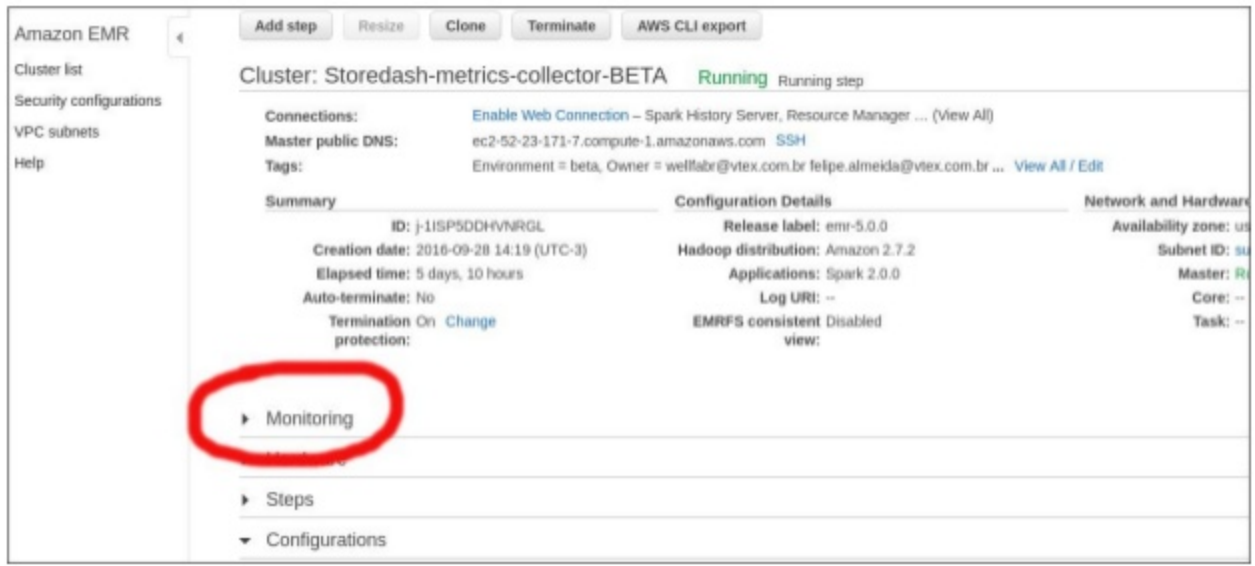
- **Nearly all** AWS services can be monitored using CloudWatch
- Metrics provide **visibility** into your applications
- They enable you to make informed decisions such as
  - Whether to downscale a machine that's not being used too much (save $$)
  - Whether to scale up a machine that's close to full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



EMR

Usage - Via other services

All Applications › [redacted] › [redacted] (Environment ID: [redacted], URL: [redacted])

- Dashboard
- Configuration
- Logs
- Health
- Monitoring
- Alarms
- Managed Updates NEW
- Events
- Tags

Overview

Health
Ok
Causes

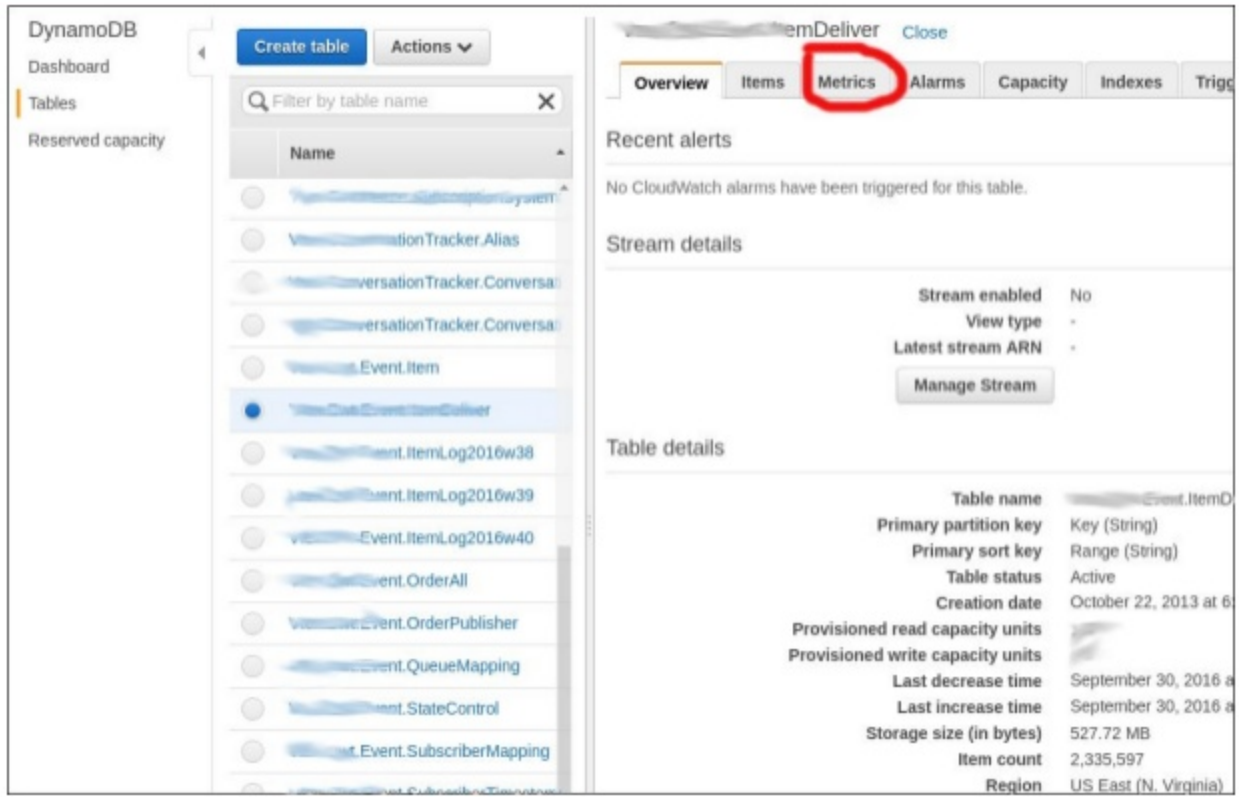
Running Version
v2.31.0
Upload and Deploy

Recent 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 a
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.

Elastic Beanstalk

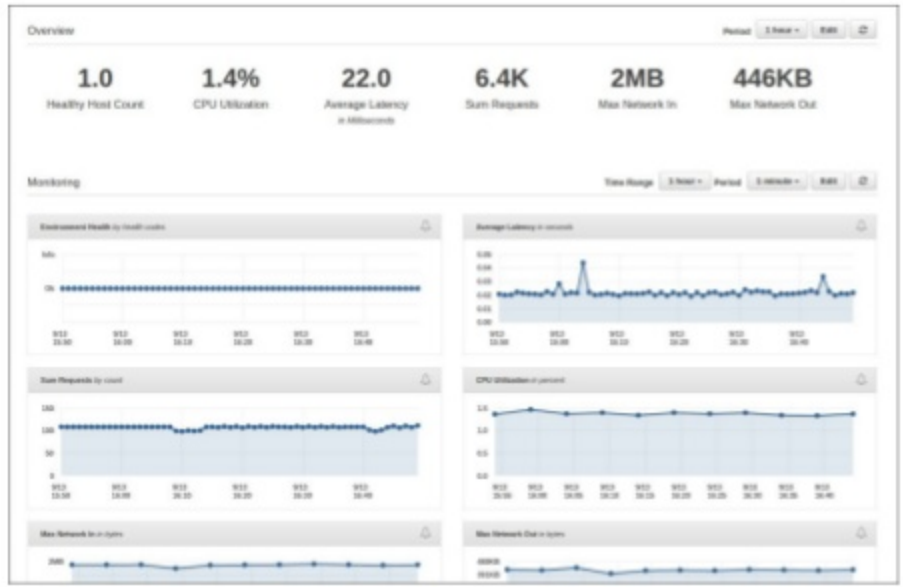
Usage - Via other services



DynamoDB Table

Usage - Via other services

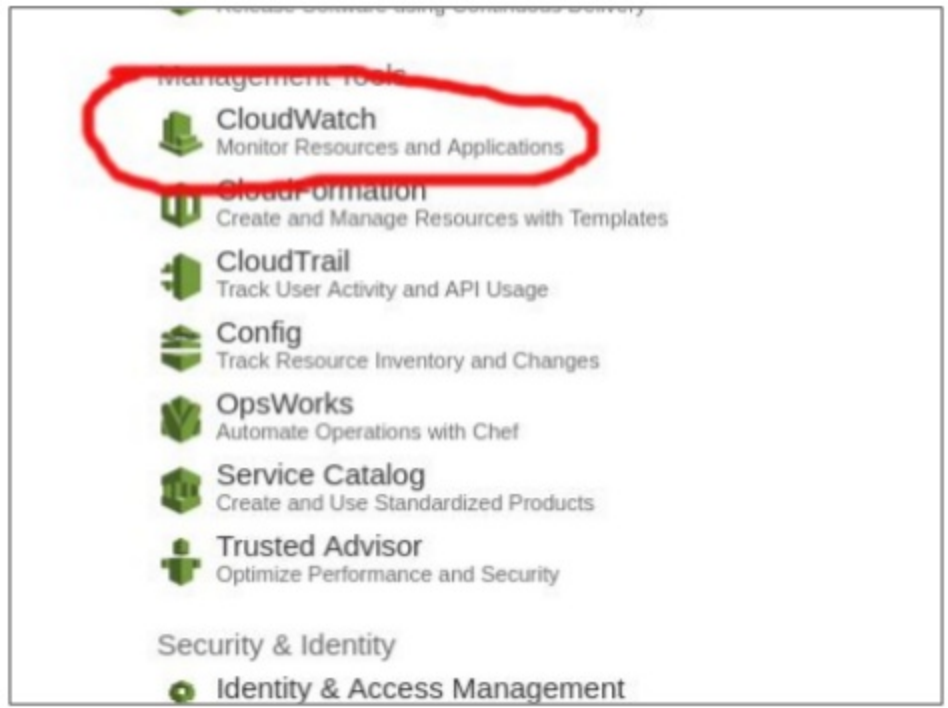
- When you open the monitoring tab for a service, what you're looking at are actually CloudWatch Metrics:



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 **programatically**
  - 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 an 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