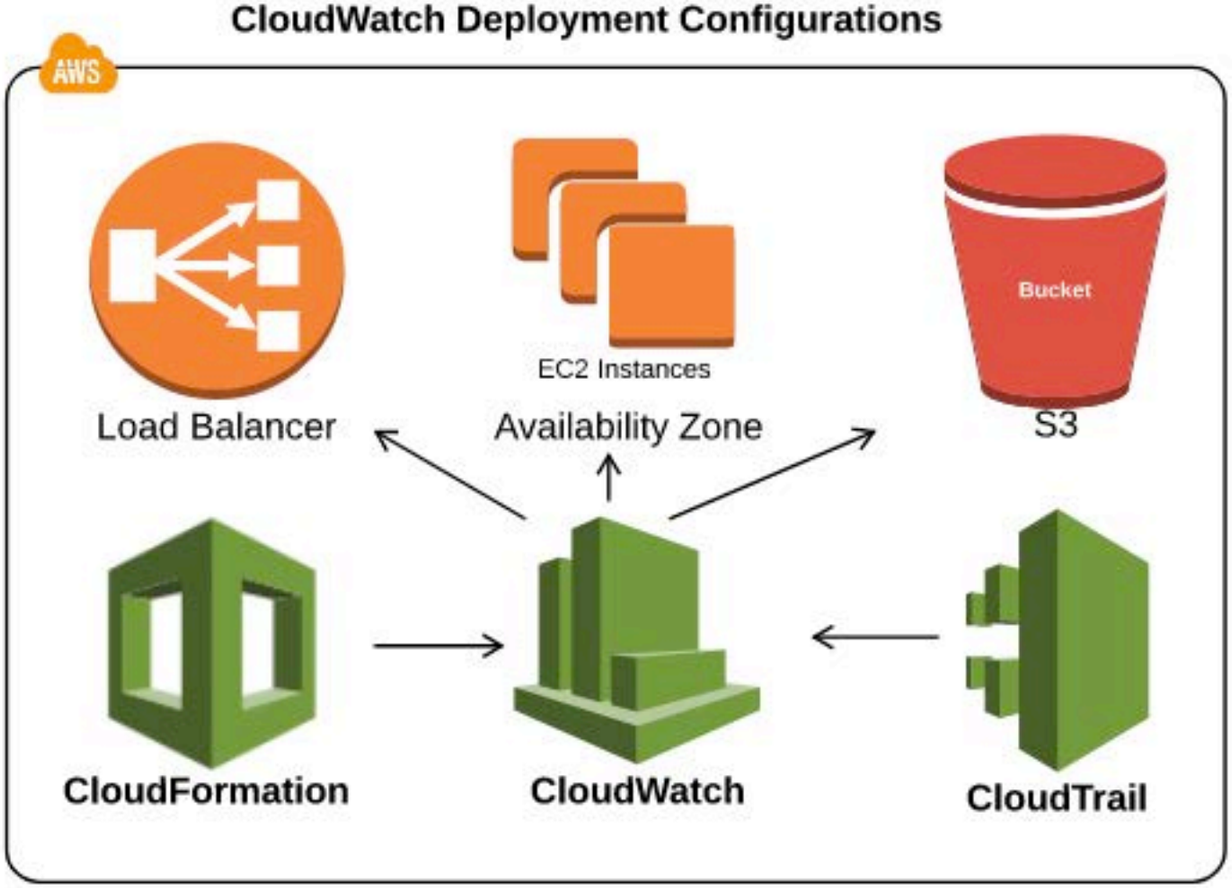
Choose a Topic

CloudWatch For DevOps

Scenario: CloudWatch is an essential tool for the DevOps Engineer. CloudWatch supports the DevOps concepts of automation, communication, and collaboration, by giving access to monitoring and logging. CloudWatch metrics can be used to work with Elastic Load Balancers and determine the scaling actions of Auto Scaling Groups. Custom Metrics are a very powerful tool which allow the DevOps Engineer to leverage CloudWatch monitoring in a wide range of scenarios.

CloudWatch Deployment Configurations



Choose a Topic

CloudWatch For DevOps

Scenario Simulator | Key Concepts | X

AWS

CloudWatch Concepts And Terminology

CloudWatch Concepts

- CloudWatch Metrics:
  - A time-ordered set of datapoints that are published to CloudWatch.
  - Only exist in the region in which they are created.
  - Can't be deleted but expire after 14 days if no new data is published to them.
  - Services can have multiple different metrics.
  - Each metric has data points organized by time:
    - Name
    - Namespace
    - One or more dimensions
  - API actions can be used to publish and retrieve data to and from CloudWatch
- CloudWatch Statistics - There are 5 statistics in CloudWatch: Average, Minimum, Maximum, Sum, SampleCount.
  - Statistics are aggregations of metric data over a period of time.
- CloudWatch Periods - Allows us to control what data is aggregated in statistics and can be used for alarms.

Choose a Topic

CloudWatch For DevOps

Next

X

AWS

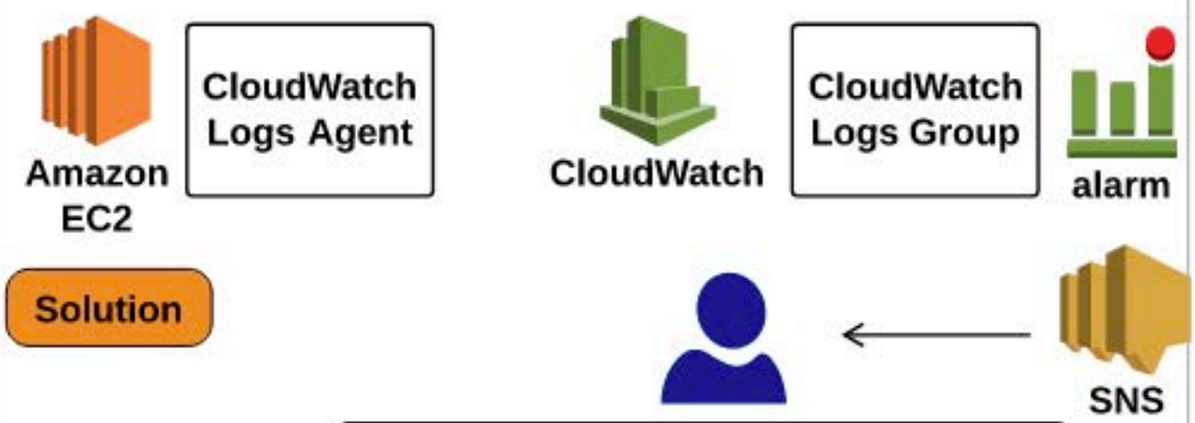
CloudWatch Scenarios

Scenario: Send your EC2 logs to CloudWatch, store your logs durably, and after 60 days send to long term storage.

Solution

Solution: Install CloudWatch Logs Agent on EC2, send log data to CloudWatch, archive data to S3, create S3 Lifecycle policy to send logs to Glacier after 60 days.

Scenario: Using CloudWatch, you need to capture 500 errors from your web server and notify your on-call engineer.



Solution

Solution: Install CloudWatch Logs Agent on EC2, stream log data to CloudWatch, create CloudWatch Log Group to capture 500 errors, set an alarm on those errors. Use SNS to email on-call personnel when alarms occur.

Choose a Topic

CloudWatch For DevOps

Next

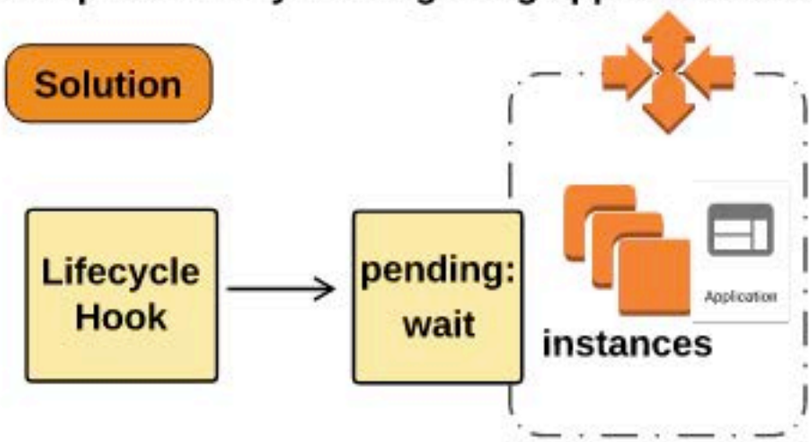
X

AWS

Auto Scaling Scenarios

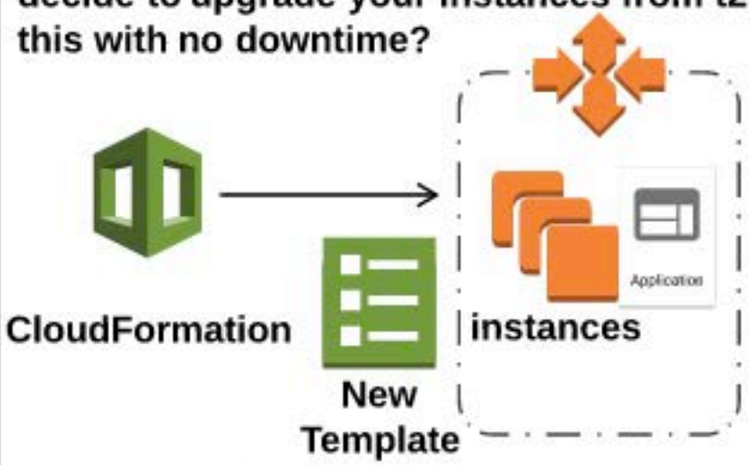
Scenario: Bootstrapping instances in ASG takes 10 minutes. Instances are reported as in-service before bootstrapping completes and you are getting application alarms.

Solution



Solution: Create an ASG Lifecycle Hook to hold the instance in a pending:wait state until bootstrapping is complete. Move to pending:complete when bootstrapping is finished.

Scenario: You have used CloudFormation to deploy an application in an Auto Scaling Group. The ASG is at its maximum 6 instances due to high CPU utilization and it is still too high. You decide to upgrade your instances from t2 to C3. How can you do this with no downtime?



Solution: Update the Launch Config in your template (t2 to C3). Add an update policy to your template and specify AutoScalingRollingUpdate.

Perform a stack update with the new template.

Solution

Choose a Topic

CloudWatch For DevOps

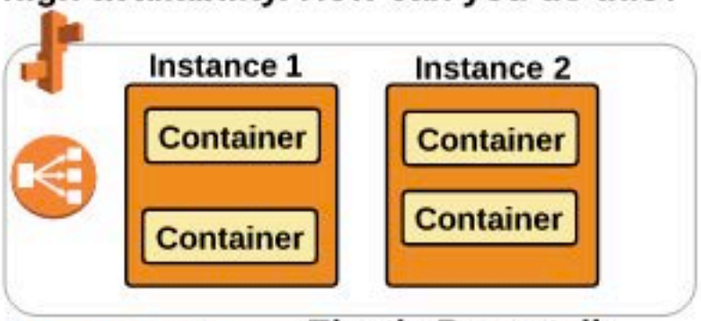
Next

X

AWS

Elastic Beanstalk Scenarios

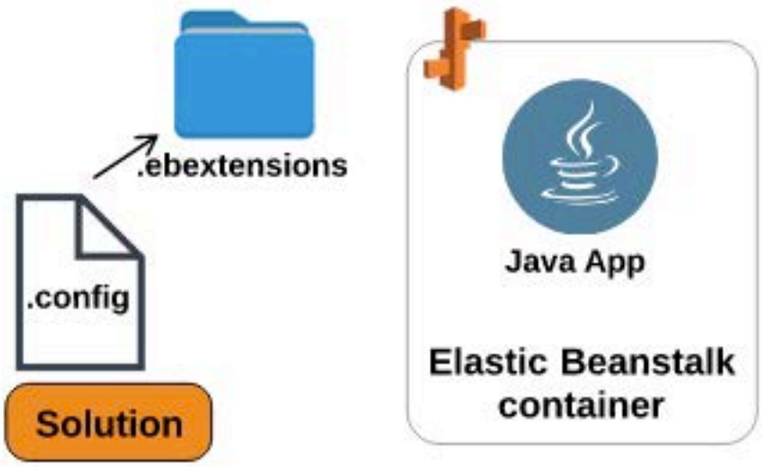
Scenario: You work for a large software company with a very diverse list of programming languages and platforms. The overriding requirement is to be able to deploy all the applications quickly using Elastic Beanstalk and to have high availability. How can you do this?



Solution

Solution: Use Docker containers to develop the apps into Elastic Beanstalk. Specify Elastic Load Balancing and Auto Scaling in your environment.

Scenario: You have deployed a Java application in an Elastic Beanstalk environment. Now you have created a script to force HTTPS on Apache Web Server. What's the best way to deploy this script?



Solution

Solution: Save your script with a config extension and save it in the .ebextensions folder. Elastic Beanstalk will automatically apply the update.

Choose a Topic

CloudWatch For DevOps

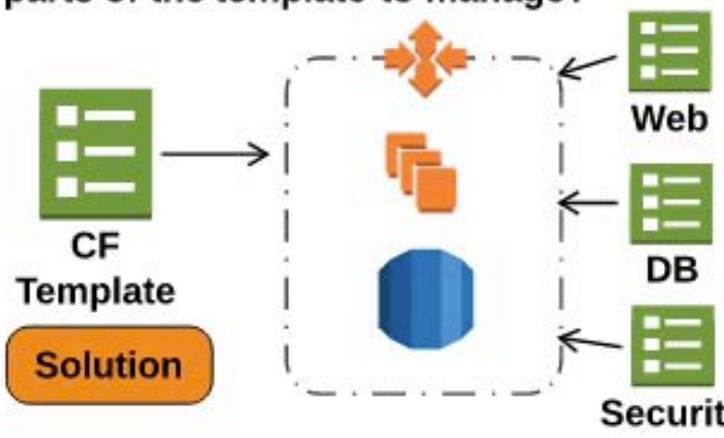
Next

X

AWS

CloudFormation Scenarios

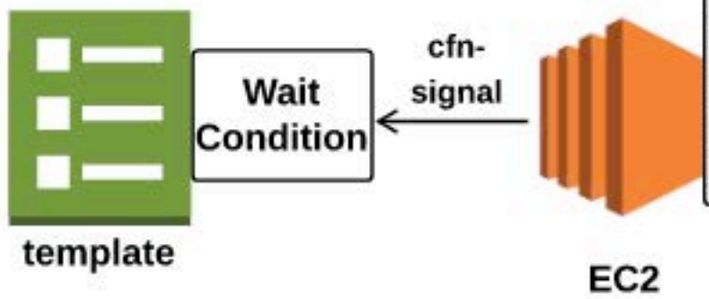
Scenario: You have a single CloudFormation template for your company infrastructure, a multi tier web application. Your web, database, and security teams are in conflict over editing the template. How can you give each team separate parts of the template to manage?



Solution

Solution: Create a nested structure for your template. Create a separate template for each group, allowing them to manage their own resources.

Scenario: You are building a Web Server using a CloudFormation template. You add a long running script to the user data. What can you do to insure that the script has finished and the server is up and running before it is added to the Load Balancer?



Solution: Add a CloudFormation wait condition. Use cfn-signal to signal when the script is complete and the web server is ready.

Solution

CloudWatch For DevOps

Next

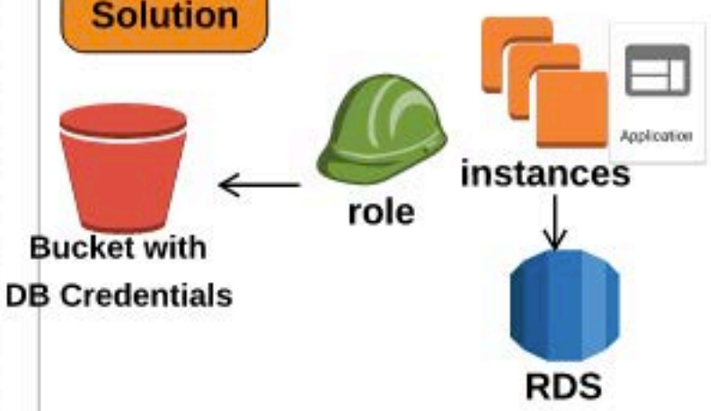
X

AWS

EC2 Scenarios

Scenario: You have an application on EC2 instances. You need to securely store DB connection information (not hard code in your app).

Solution



Solution: Create an IAM Role for the EC2 instances giving read access to an S3 Bucket which stores the DB connection credentials. Retrieve the credentials in the app on the EC2 instances.

Scenario: You have an Auto Scaling Group of EC2 instances that you need to bootstrap. You need a highly durable, secure storage for your bootstrapping files and choose S3. How do you retrieve this information?

Solution: Pre-bake the AMI which creates the EC2 instances with an IAM Role which allows read access to the bucket. Retrieve the bootstrapping file programmatically from the instances.

Solution

Choose a Topic

CloudWatch For DevOps

Next

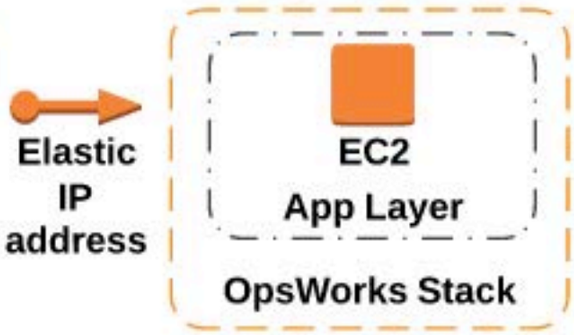
X

AWS

OpsWorks Scenarios

Scenario: You are creating an OpsWorks Stack to host your application. You have created the stack, added Layers including an Application Layer, and add an instance to the Layer. But the instance never reaches a ready state and deployment fails. What could be wrong?

Solution



Solution: The EC2 instance does not have a public ip address (or elastic ip address) and is failing bootstrapping.

Make sure instances are assigned a public ip address (or elastic ip address)

Scenario: You are managing an OpsWorks Stack. You have a new requirement to perform Blue/Green Deployments to greatly minimize downtime. How can you implement this in OpsWorks?

Green Stack with app layer and instance. Update application on the instance.

Solution: Clone your OpsWorks Stack. When you are ready to deploy, update the application on your new (green) stack, then swap urls.

Solution

Choose a Topic

CloudWatch For DevOps

X

AWS

CloudWatch Key Concepts

- **CloudWatch Dimensions** - represent a name/value pair that uniquely identifies a metric.
- Example EC2 metric dimensions:
  - InstanceId
  - InstanceType
  - ImageId
  - AutoScalingGroupName
- Example ELB metric dimensions:
  - AvailabilityZone
  - LoadBalancerName
- **CloudWatch Namespaces** - used to isolate different application and service metrics
- Example namespaces:
  - EBS - AWS/EBS
  - ELB - AWS/ELB
  - EC2 - AWS/EC2
- We can create custom namespaces for custom metrics.
- **Logging Terminology**:
  - Log Event - the activity being reported.
  - Log Stream - represents a sequence of Log Events from the same source.
  - Log Group - A grouping of Log Events that have the same properties, policies, and access controls.
  - Metric Filters - allow us to define which metrics to extract and publish to CloudWatch.
  - Retention Policies - Dictate how long the data is kept.
  - Log Agent - This is the agent that we can install on EC2 instances to automatically publish log events to CloudWatch.

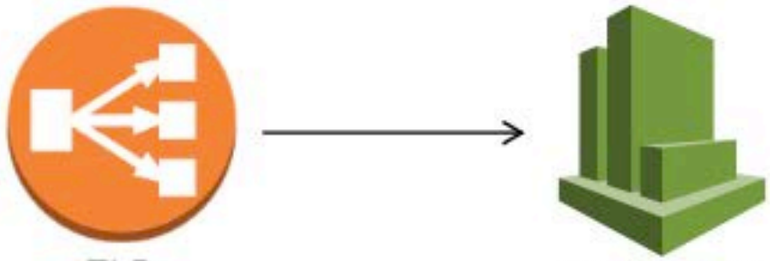
Choose a Topic

CloudWatch For DevOps

X

AWS

Elastic Load Balancer Metrics



ELB Metrics

- ELB Metrics vs Logs
  - Metrics: The ELB can publish data points about the Load Balancer and back-end instances .
  - We can retrieve statistics about those data points as an ordered set of time-series data.
  - Logging : Logs give information regarding requests made to the Load Balancer.
  - Each log contains: when a request was received, the client's ip address, latency information, the request path and parameters, and server responses.
- ELB Metrics
  - BackendConnectionErrors - number of unsuccessful connections between ELB and its instances.
  - HealthyHostCount, UnhealthyHostCount - # of healthy or unhealthy registered instances.
  - HTTPCode_Backend_XXX - # of Http response codes from backend instances.
  - HTTPCode_ELB_4XX and HTTPCode_ELB_5XX
  - Latency - time elapsed after request leaves ELB to receipt at back end.
  - RequestCount - # of requests completed during a specific interval (1 or 5 minutes).
  - SurgeQueueLength - # of requests that are pending routing.
  - SpillOverCount - # of requests rejected due to SurgeQueueLength being full.
- ELB Statistics - Metric data aggregations over a specified period of time.

Choose a Topic

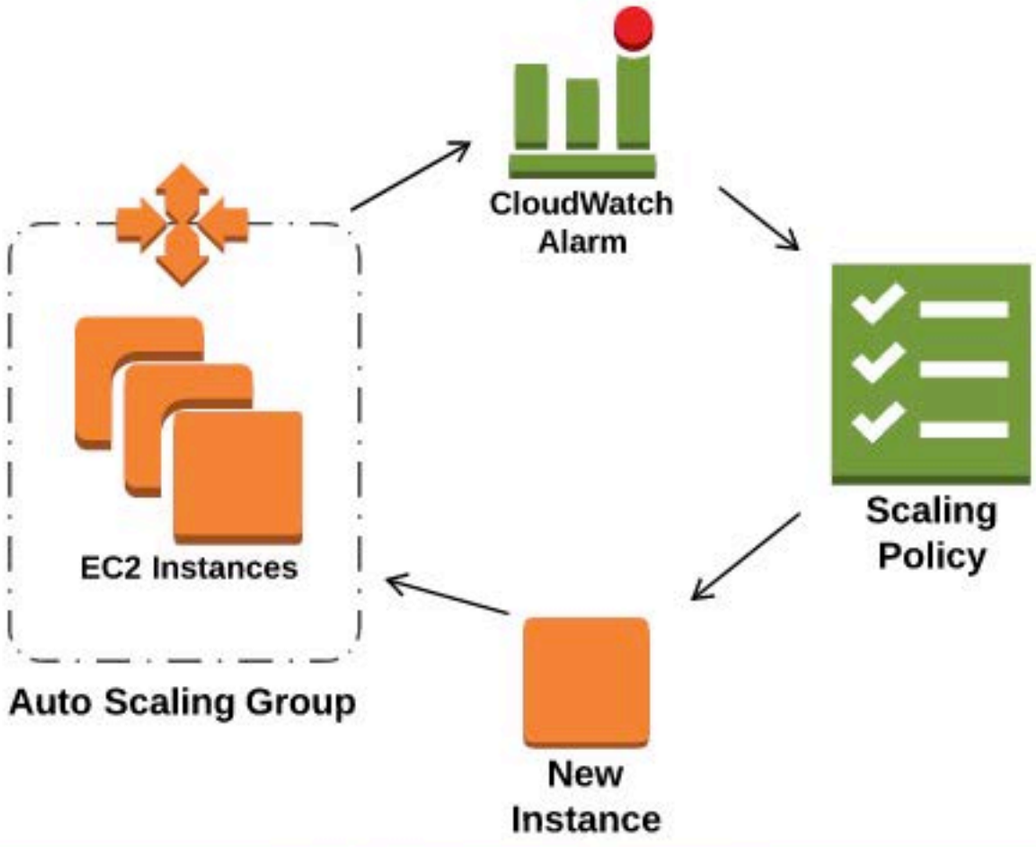
CloudWatch For DevOps

Key Concepts

X

AWS

Auto Scaling And EC2 Metrics



Auto Scaling And EC2 Metrics

- Auto Scaling Metrics:
  - GroupMinSize, GroupMaxSize, GroupDesiredCapacity, GroupInServiceInstances, GroupPendingInstances, GroupStandbyInstances, GroupTerminatingInstances, GroupTotalInstances
- EC2 Metrics:
  - CPUUtilization - % of allocated EC2 compute units currently in use.
  - DiskReadOps - completed read operations from all instance store volumes available to an instance.
  - DiskWriteOps - completed write operations from all instance store volumes available to an instance.

Choose a Topic

CloudWatch For DevOps

X

AWS

CloudWatch ASG And EC2 Key Concepts

EC2 Metrics continued:

- **DiskReadBytes** - Bytes read from all instance store volumes available to the instance.
- **DiskWriteBytes** - Bytes written from all instance store volumes available to the instance.
- **NetworkIn** - # of bytes received on all network interfaces of an instance.
- **NetworkOut** - # of bytes sent on all network interfaces of an instance.
- **NetworkPacketsIn, NetworkPacketsOut** - Monitors the number of packets sent and received on all network interfaces of an instance.
- **StatusCheckFailed_Instance** - Keeps track of whether the instance passed the status check within the last minute.
- **StatusCheckFailed_System** - Keeps track of whether an instance passed the system status check within the last minute.
- **StatusCheckFailed** - Combines both of the status checks and reports whether they passed. 0 = passed, 1 = failed.

Scaling Policies:

- ChangeInCapacity
- ExactCapacity
- PercentChangeInCapacity

Scaling Policy Type:

- Simple Scaling - increases or decreases the current capacity of the group from a single scaling adjustment.
- Step Scaling - increases or decreases the capacity from a set of scaling adjustments instead of just one.

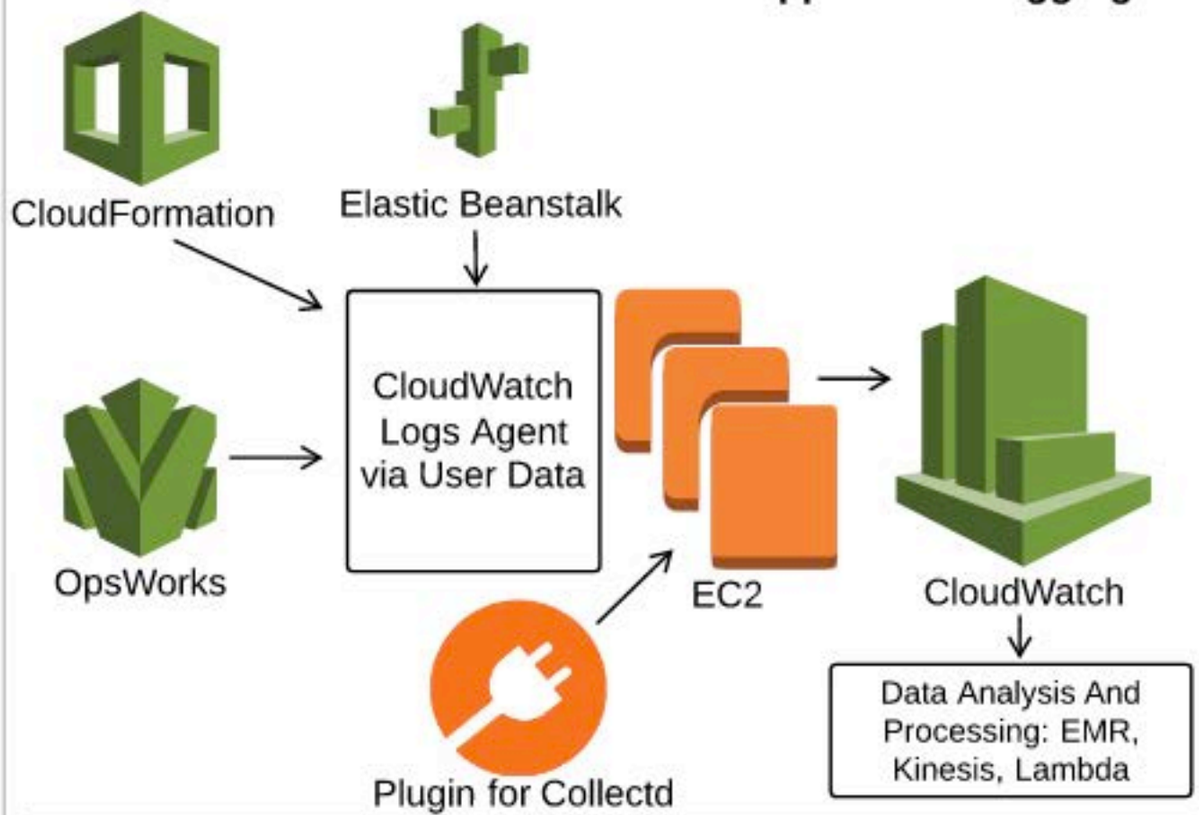
Choose a Topic

CloudWatch For DevOps

X

AWS

CloudWatch Custom Metrics And Application Logging



Custom Metrics And Logging

- Provide the flexibility to publish custom data to CloudWatch.
- Beneficial for troubleshooting and creating alarms.
- Options to install and configure custom metrics:
  - Install the CloudWatch Logs Agent on existing instances. Install the agent using OpsWorks, CloudFormation, or Elastic Beanstalk.
  - Use the API, CLI, or SDKs to install the Logs Agent or Collectd.
- OpsWorks as an example, steps:
  - 1) Install the agent 2) Configure the agent (specify which log file to monitor on each EC2 instance, specify where to send logs) 3) Make sure the agent is running.
- Search and Filter Metric Data with Metric Filters which have 4 key elements:
  - Filter Pattern, Metric Name, Metric Namespace, Metric Value (example: to count 404s, we could use a value of 1 for each 404 found).

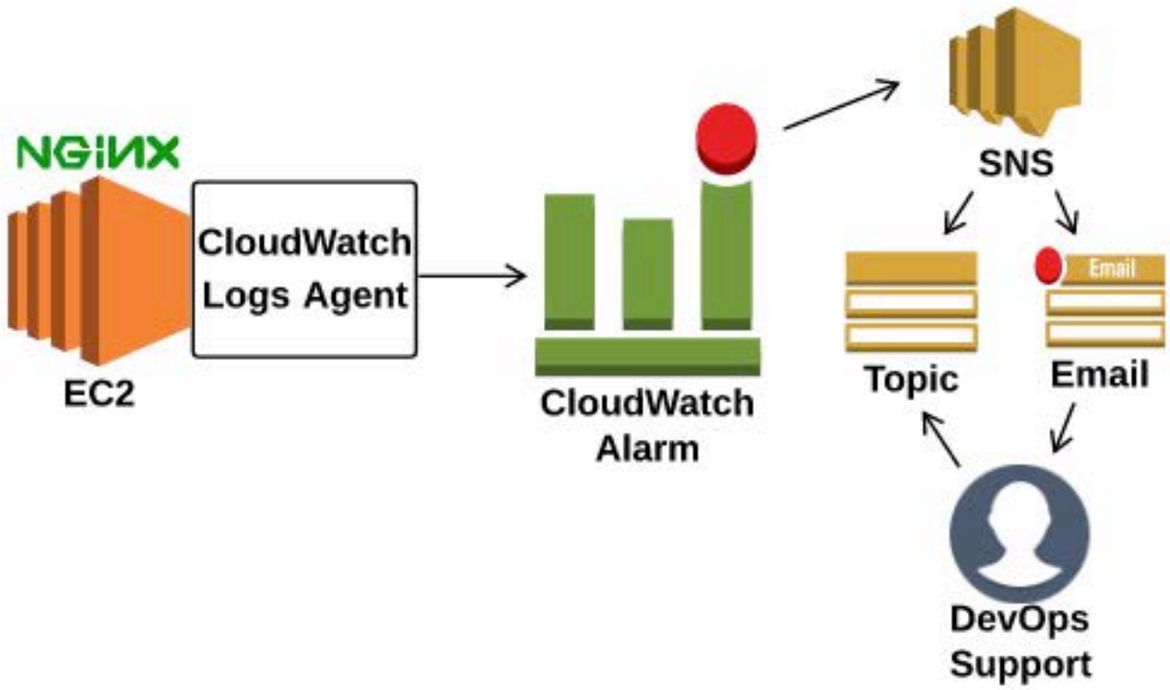
Choose a Topic

CloudWatch For DevOps

X

AWS

Using the CloudWatch Logs Agent, Log Groups, and SNS notifications



Using the CloudWatch Logs Agent With SNS notifications

Step 1 - SSH into the provided instance (CloudWatch Logs Agent already installed).
Step 2 - Start nginx: sudo service nginx start.
Step 3 - Go to /etc/awslogs and modify awslogs.conf.
Step 4 - Restart nginx: sudo service awslogs restart.
Step 5 - Create an SNS Topic and Subscription.
Step 6 - Subscribe to the topic (email address).
Step 7 - Create a CloudWatch metric filter.
Step 8 - Create CloudWatch Alarm.
Step 9 - Verify the CloudWatch Alarm.

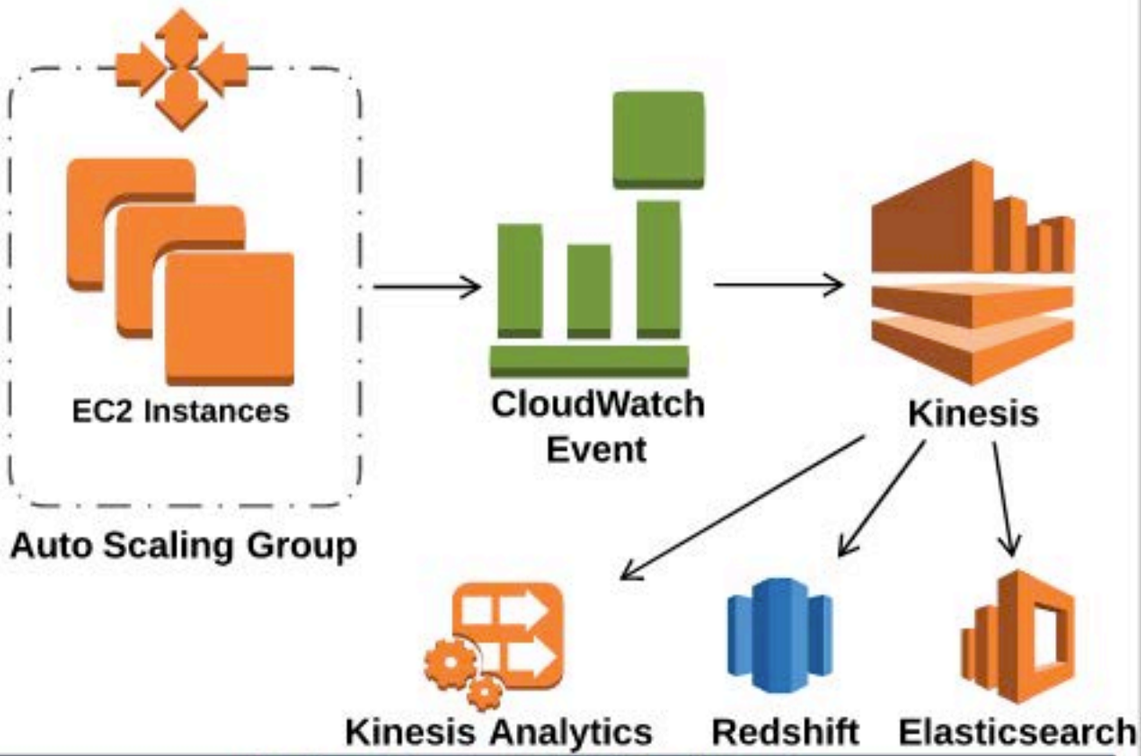
Choose a Topic

CloudWatch For DevOps

X

AWS

Using Kinesis to Process CloudWatch Data



Using Kinesis With CloudWatch

1) Prerequisite - From the AWS Management Console launch two EC2 instances (name them mgmt and demo).
2) SSH into the mgmt instance, run ***aws configure*** and enter the access key, secret access key and region.
3) enter: **aws kinesis create-stream --stream-name *testKinesis* --shard-count 1**
4) enter: **aws kinesis describe-stream --stream-name testKinesis**
5) Open the CloudWatch console, go to events, and create a rule for EC2 instance state-change notification. Choose the Kinesis stream (testKinesis) as a target.
6) To test the rule, go to the EC2 console and stop the demo instance.
7) Go to the CloudWatch console, choose Events, Rules, click on the name of the rule, and choose 'show metrics for the rule'.
8) At the command line enter: **aws kinesis get-shard-iterator --shard-id shardId-000000000000 --shard-iterator-type TRIM_HORIZON --stream-name test**
9) then enter: **aws kinesis get-records --shard-iterator "*this is the shard iterator output from the previous step*"**.