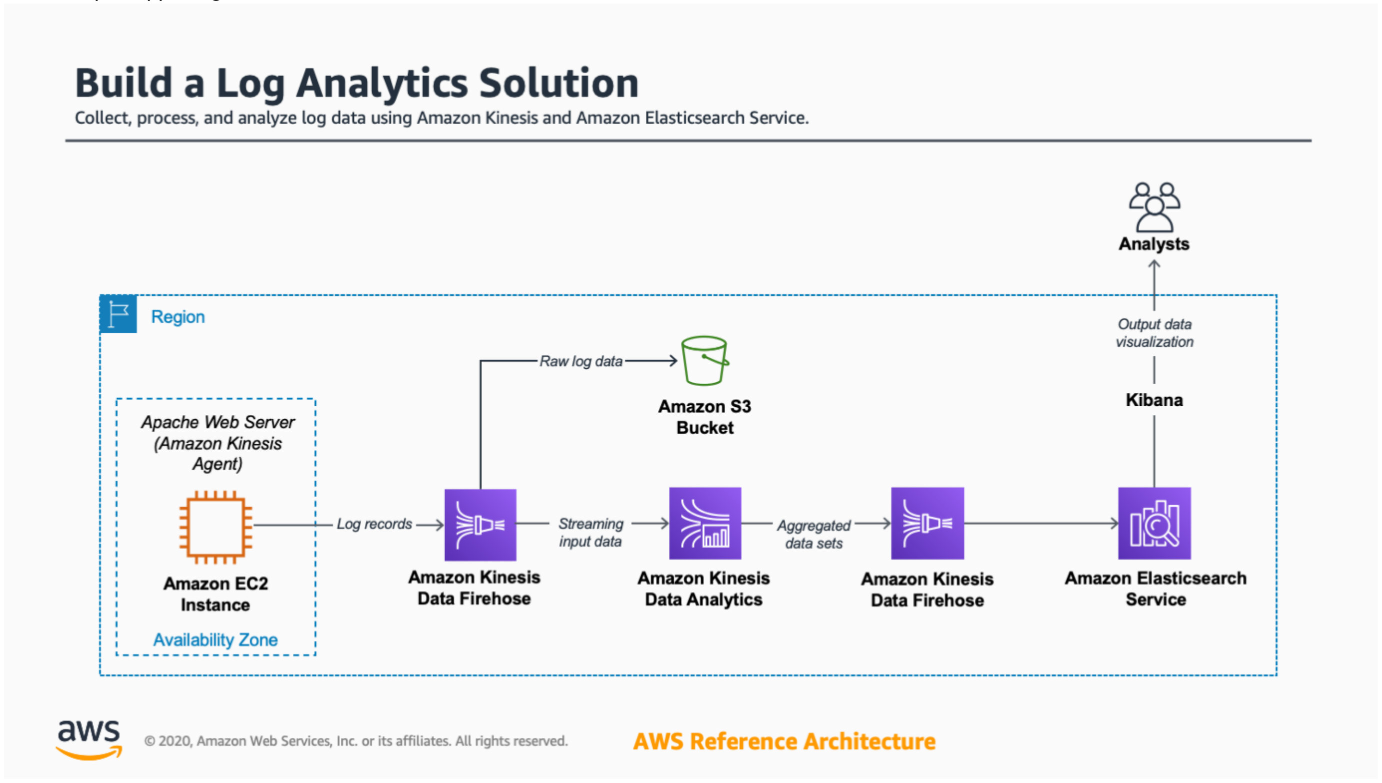
Log Analytics Solution Architecture Using Kinesis

From the Build a Log Analytics Solution on AWS tutorial: *click to go into tutorial*

This is a tutorial that helps us create the foundation for one way of using our data lake which is through log streaming. The logged data streams into the data lake and gets processed and aggregated in order to be ready to be analyzed. This process is an automated way of using our data lake and helps us track data lineage as part of data governance in a data lake. The use case diagram is a perfect way of recording this, as well as the steps happening in the back end.


Build a Log Analytics Solution
Collect, process, and analyze log data using Amazon Kinesis and Amazon Elasticsearch Service.
Analysts
Region
Output data visualization
Raw log data
Amazon S3 Bucket
Kibana
Apache Web Server (Amazon Kinesis Agent)
Log records
Streaming input data
Aggregated data sets
Amazon EC2 Instance
Amazon Kinesis Data Firehose
Amazon Kinesis Data Analytics
Amazon Kinesis Data Firehose
Amazon Elasticsearch Service
Availability Zone
aws
© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
AWS Reference Architecture

Steps of what is happening in the back end

1. IoT's create logs from users
2. The logs stream into our EC2 instance (virtual server)
3. The kinesis agent takes the data logs and processes them into JSON
4. The processed data ingests into our AWS data lake using our 1st firehose stream
    a. failed logs go into the S3 bucket
5. The successful logs go into kinesis data analytics and gets aggregated with pre-made SQL queries
6. The aggregated data is sent to Elasticsearch via a 2nd firehose stream
7. That data can now be analyzed through Kibana
8. The process is automated so now analysts can discover insights without worrying about the incoming data

---

Creating the Kinesis log streaming architecture process:

Step 1: Set Up Prerequisites

- Get AWS credentials/membership
- AWS resources have to be created in the US East AWS Region (us-east-1)
- Set up an EC2 instance to be your web server and log producer
- Prepare log files (we used a fake apache log generator, typically it'll generate automatically from IoT's)
- Connect to your EC2 instance (using terminal) and generate the log files into that instance for storage

Step 2: Create an Amazon Kinesis Data Firehose Delivery Stream

- This will load the log data into AWS typically in 5mb or 300s batches
- Amazon Kinesis Data Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon Simple Storage Service (Amazon S3), Amazon Redshift, or Amazon Elasticsearch Service

Step 3: Install and Configure the Amazon Kinesis Agent on the EC2 Instance

- The EC2 instance uses the agent to send data to Firehose to be ingested into AWS
- The agent continuously monitors a set of files and sends new data to your delivery stream
- It also emits Amazon CloudWatch metrics to help you better monitor and troubleshoot the streaming process
- It also has preprocessing power and will convert the Common Log Format into JSON before sending

Step 4: Create an Amazon Elasticsearch Service Domain

- This domain is used to store the data produced for later visualization and analysis

Step 5: Create a Second Amazon Kinesis Data Firehose Delivery Stream

- Now that we have somewhere to persist the output of our Amazon Kinesis Data Analytics application, we need a simple way to get our data into our Amazon Elasticsearch Service domain
- Amazon Kinesis Data Firehose supports Amazon ES as a destination

Step 6: Create an Amazon Kinesis Data Analytics Application

- We create the application in order to aggregate data from our streaming web log data and store it in our Amazon ES domain
- Within the application we:
    - Discover our schema
    - Create the SQL that analyzes the streaming data
    - Save the running queries into the second firehouse delivery stream (step 5)

Step 7: View the Aggregated Streaming Data

- After a few miniutes the aggregated data will move into ElasticSearch and can be explored using Kibana
- Here we can create new visualizations and analyze our data

Issues:

1. After connecting to the instance you must create tmp/logs directories and then copy the raw version of the apache-fake-log-gen.py file into tmp/logs using
    a. *curl -o apache-fake-log-gen.py https://raw.githubusercontent.com/kiritbasu/Fake-Apache-Log-Generator/master/apache-fake-log-gen.py*
        i. UPDATE: problem may be fixed in step 9 of ‘start an EC2 an instance’
2. When running the fake logs, they will show up in the base of the instance, not in tmp/logs, doesn’t seem to work when running in tmp /logs, so you must move the files after creating them
3. sudo python /tmp/logs/apache-fake-log-gen.py -n 0 -o LOG & must be changed to sudo python tmp/logs/apache-fake-log-gen.py -n 0 -o LOG &. (take out / in front of tmp)
4. /home/ec2-user/tmp/logs (is the root to use when setting up your agent configuration, instructions weren’t totally clear with setting up)
    a. Or just pwd at the root to know actual root
    b. use code - sudo nano {*configuration-agent-name*}
5. Waiting for data to stream came to a stand still and even in free tier it costs money to wait, so had to terminate
6. It is recommended to terminate (see billing estimates) everything when not using (steps are in the tutorial) because running these services cost roughly $0.57 just for the tutorial in the free tier but typically cost just under $400/month when in use.