**Log Analytics Step-By-Step Demo to ingest and analyze apache web log using Beats, Logstash & Amazon Elastic Search:**

**Pre-reqs:**

1. Setup Apache Web Server
2. Setup Logstash on Web server instance

**Setup Apache Web Server:**

Login to your linux server and run this command.

*#!/bin/bash yum update -y yum install -y httpd service httpd start*

Create a couple test pages and script to hit the web server to generate logs continuously.

1. mkdir /var/www/html/adminer; cd /var/www/html/adminer
2. vi landing.txt and add below text, save and quit.

This is landing page

1. vi users.txt and add below text, save and quit.

user1

user2

1. vi pagehit.sh and add below, save and quit.

*#!/bin/bash*

*while true*

*do*

*curl http://localhost:/adminer/landing.txt*

*sleep 30*

*curl http://localhost/adminer/users.txt*

*sleep 30*

*curl http://localhost/adminer/users.txt*

*sleep 15*

*curl http://localhost/adminer/failed.txta*

*sleep 30*

*done*

1. Run the command below so the webpage gets hit continuously and the web log is updated.

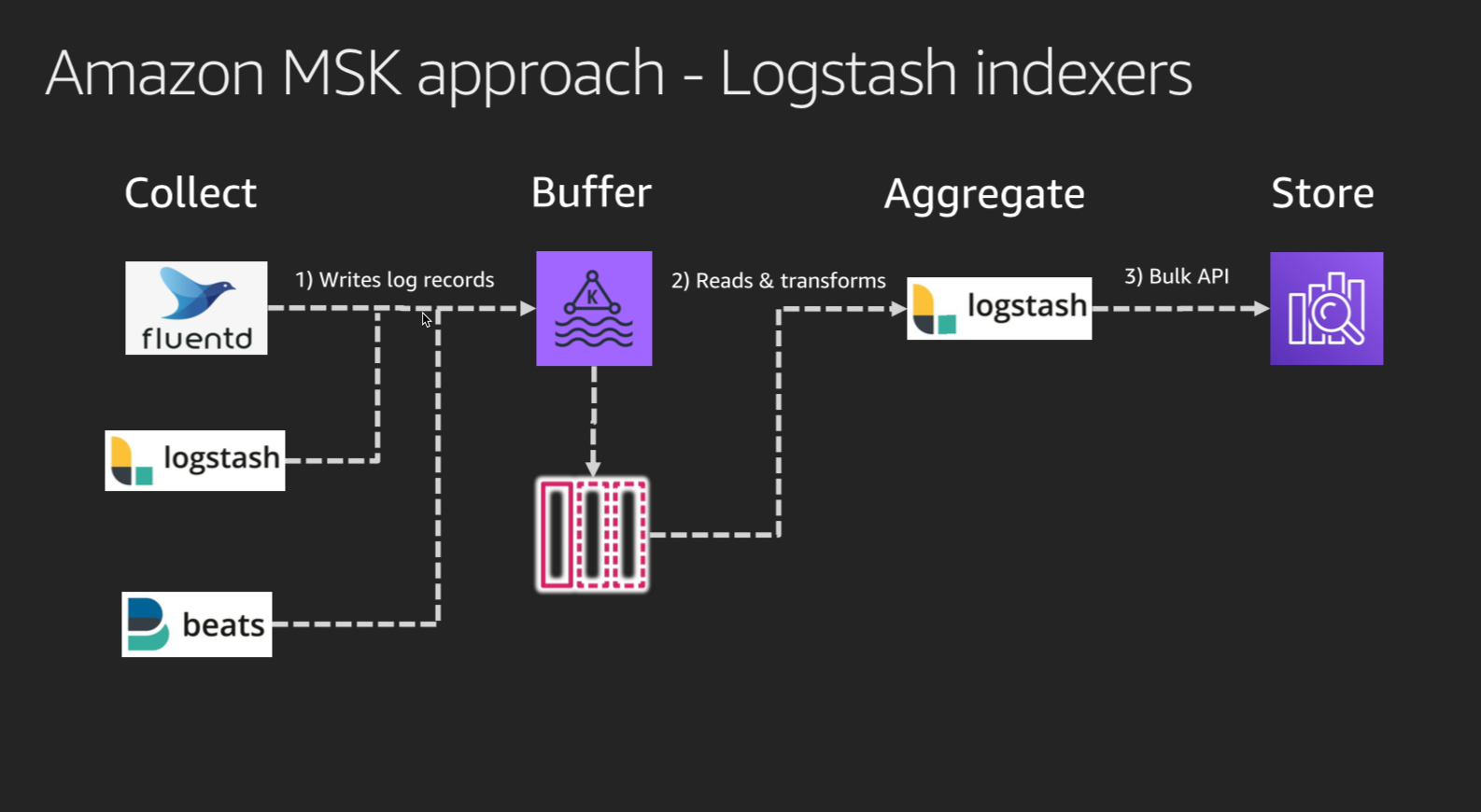
*nohup ./pagehit.sh > /dev/null 2>&1 &*

**Setup LogStash on WebServer:**

1. Use this [link](https://www.elastic.co/guide/en/logstash/7.6/installing-logstash.html#_yum) for yum install or this [link](https://www.elastic.co/downloads/logstash) for other install options.
2. For RHEL6/Centos, use this command. If your OS is different, use this [link](https://www.elastic.co/guide/en/logstash/7.6/running-logstash.html#running-logstash-systemd).

*sudo initctl start logstash*

**Demo Architecture:**



**Collect**

This is the stage where data is collected from the sources where it originates. Some of the commonly used collection agents are Beats, fluentd and logstash. In several production deployments where beats is used, beats become the source of logstash as input.

**Why Beats as a source for Logstash input?**

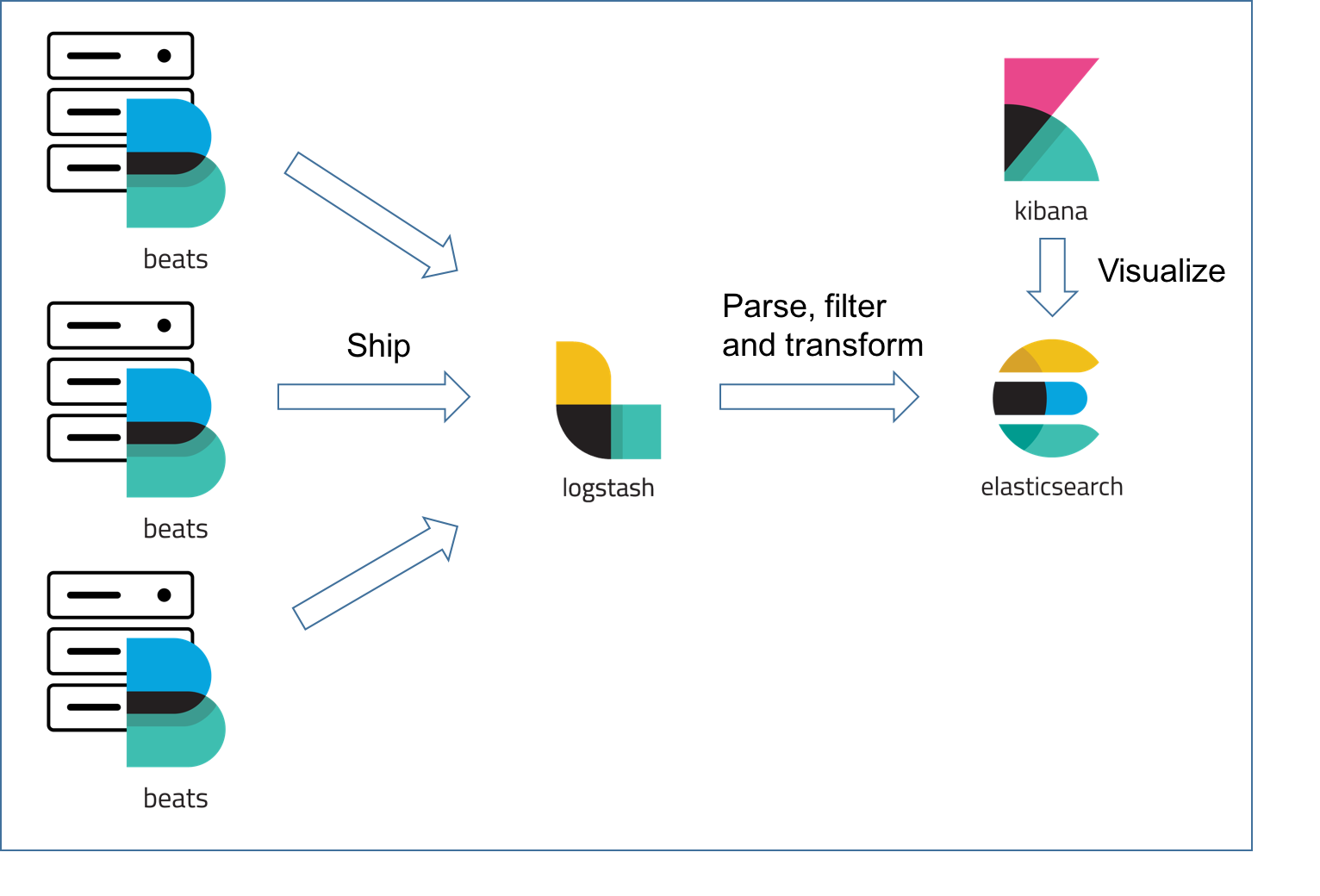
Beats are lightweight data shippers that you install as agents on your servers to send specific types of operational data to Elasticsearch. Beats have a small footprint and use fewer system resources than Logstash. This makes it practical to deploy beats agents on several machines to capture data.

Filebeat is one of the best log file shippers out there today — it’s lightweight, supports SSL and TLS encryption, supports back pressure with a good built-in recovery mechanism, and is extremely reliable

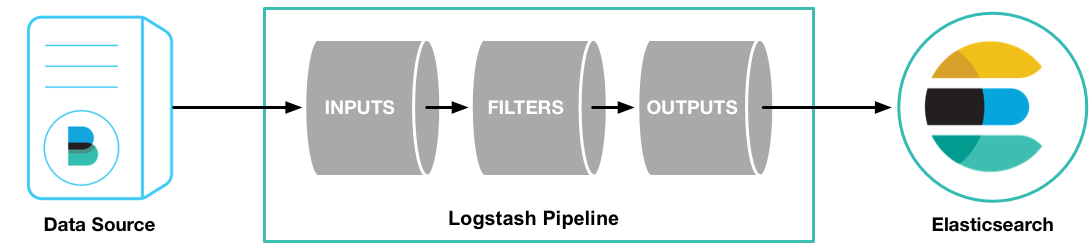
Logstash has a larger footprint, but provides a broad array of input, filter, and output plugins for collecting, enriching, and transforming data from a variety of sources.

Filebeat, and the other members of the Beats family, acts as a lightweight agent deployed on the edge host, pumping data into Logstash for aggregation, filtering and enrichment as seen in diagram below.

**Note:** In a typical use case, Filebeat runs on a separate machine from the machine running your Logstash instance. Many filebeat agents stream data to a single logstash machine.



**Logstash Pipeline**: ALogstash pipeline has two required elements, input and output, and one optional element, filter. The input plugins consume data from a source, the filter plugins modify the data as you specify, and the output plugins write the data to a destination



Lets use the steps below to create a pipeline to stream web server logs. Lets assume for this demo, you want to send existing contents of the apache access\_log file and the new entries in the log file to elastic search .

1. Filebeat setup:
2. Use this [link](https://www.elastic.co/downloads/beats/filebeat) to download the zip file to your server and complete the install steps.
3. In your filebeat.yml file, make following changes

* Under “Filebeat inputs” section, under “paths:” update the apache log file path as /etc/httpd/logs/access\_log
* Comment the lines under “Elasticsearch template setting”, “Kibana”, “Elasticsearch output”
* Uncomment following lines from “Logstash output”

*output.logstash:*

*# The Logstash hosts*

*hosts: ["localhost:5044"]*

* Change false to true in the following line.

*# Change to true to enable this input configuration.*

*enabled: true*

1. Now start the filebeat agent using command below. It will start running in background until next OS reboot.

*nohup sudo ./filebeat -e -c filebeat.yml run > /dev/null 2>&1 &*

1. Next, lets start by creating logstash input/output configuration file. Inputs are Logstash plugins responsible for ingesting data. Output is the destination to send the log data to. (In this section, we will write output to standard out on your screen so it is easy to troubleshoot which we will later change to ElasticSearch when we get to the section named “store”.) Filter is the section that helps with transformations/aggregations and is optional.

**accesslog-Ingest-pipeline.json**

*input {*

*beats {*

*port => "5044"*

*}*

*}*

*filter {*

*grok {*

*match => { "message" => "%{COMBINEDAPACHELOG}"}*

*}*

*geoip {*

*source => "clientip"*

*}*

*}*

*output {*

*stdout { codec => rubydebug }*

*}*

a. Lets test the configs before starting

*sudo /usr/share/logstash/bin/logstash -f /workspace/code/elasticsearch/aas/accesslog-ingest-pipeline.json --config.test\_and\_exit --path.settings /etc/logstash*

b. Now lets run logstash pipeline

*sudo /usr/share/logstash/bin/logstash -f /workspace/code/elasticsearch/aas/accesslog-ingest-pipeline.json --config.reload.automatic --path.settings /etc/logstash*

You will see this message if everything went well.

*[2020-04-12T22:36:29,718][INFO ][org.logstash.beats.Server][main] Starting server on port: 5044*

*[2020-04-12T22:36:31,127][INFO ][logstash.agent ] Successfully started Logstash API endpoint {:port=>9600}*

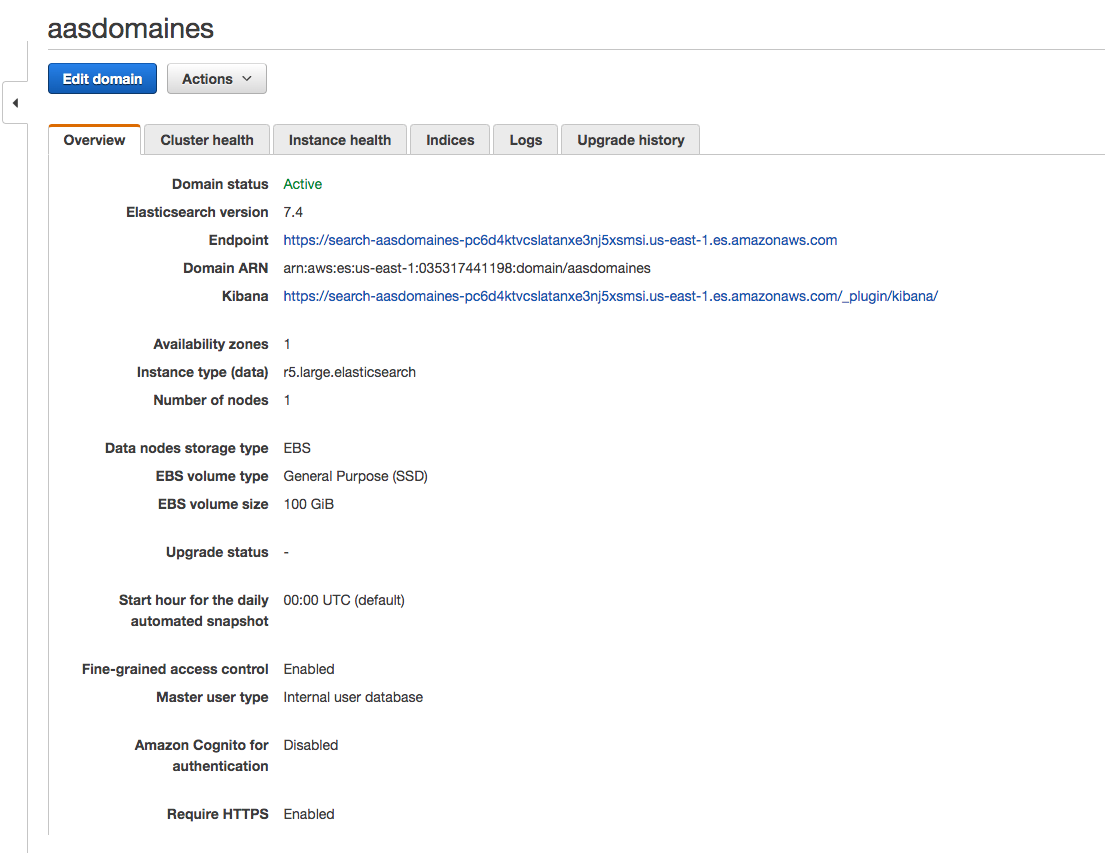
**Note:** The --config.reload.automatic option enables automatic config reloading so that you don’t have to stop and restart Logstash every time you modify the configuration file.

c. Notice the filter section in the access-log-input-pipeline.json file. The grok filter plugin enables you to parse the unstructured log data into something structured and queryable. More information here on [grok filter.](https://www.elastic.co/guide/en/logstash/current/advanced-pipeline.html#configuring-grok-filter)

**Store**

In this section, we will store the logstash filtered/transformed data to Amazon ElasticSearch. Lets create an Amazon ElasticSearch cluster in your account and use steps in this [link](https://docs.aws.amazon.com/elasticsearch-service/latest/developerguide/es-gsg-create-domain.html) to create your ES domain which will act as an output.

1. Once your elastic search domain is up, you should see the domain status as Active



2. Edit the pipeline config json file and replace the “output” section with below changing the “localhost” to your elastic cluster endpoint seein above screenshot.

*vi accesslog-Ingest-pipeline.json*

*output {*

*elasticsearch {*

*hosts => [ "https://search-aasdomaines-pc6d4ktvcslatanxe3nj5xsmsi.us-east-1.es.amazonaws.com:443"]*

*user => "userid"*

*password => "XXXX"*

*ilm\_enabled => false*

*ssl => true*

*}*

*}*

1. From your Elastic Search domain page on AWS console, click on “Actions🡪Modify access policy” from the drop down and add the json below replacing the IP address with the IP address of the server hosting Logstash instance.

*{*

*"Version": "2012-10-17",*

*"Statement": [*

*{*

*"Effect": "Allow",*

*"Principal": {*

*"AWS": "\*"*

*},*

*"Action": "es:\*",*

*"Resource": "\*",*

*"Condition": {*

*"IpAddress": {*

*"aws:SourceIp": "10.2.85.196"*

*}*

*}*

*}*

*]*

*}*

1. To verify if you are able to connect to logstash and view the log indexes from logstash along with rest of the indexes, run the command below from the logstash linux server.

*curl -XGET -u userid:password 'https://search-aasdomaines-pc6d4ktvcslatanxe3nj5xsmsi.us-east-1.es.amazonaws.com/\_cat/indices?v'*

1. Finally, lets query the logs to get all the log lines that has “404” dated on 04-13-2020.

*curl -XGET -u master:Master@123 'https://search-aasdomaines-pc6d4ktvcslatanxe3nj5xsmsi.us-east-1.es.amazonaws.com/logstash-2020.04.13/\_search?q=response:404'*

1. Similar to above query, find all the lines in log with exact match to response 404.

*curl -XGET --header 'Content-Type: application/json' -u master:Master@123 https://search-aasdomaines-pc6d4ktvcslatanxe3nj5xsmsi.us-east-1.es.amazonaws.com/logstash-2020.04.13/\_search -d '{"query" : {"match" : { "response": "404" }}}'*

Look for the first line to read ho many lines were matched in the response. It looks something like this.

*{"took":2,"timed\_out":false,"\_shards":{"total":1,"successful":1,"skipped":0,"failed":0},"****hits":{"total":{"value":791****,"relation":"eq"}*