International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.8, Issue.5, May 2020 E-ISSN: 2347-2693

IoT Data Analytics Pipeline using Elastic Stack and Kafka

Jayashri V[1*], K Badari Nath[2]

[1,2]Dept. of Computer Science and Engineering, R V College of Engineering, Bangalore, India

Corresponding Author: jaya.shri1998@gmail.com, Tel.:9886520107

DOI: https://doi.org/10.26438/ijcse/v8i5.144148 | Available online at: www.ijcseonline.org

Received: 03/May/2020, Accepted: 12/May/2020, Published: 31/May/2020

Abstract— With Internet of Things enabling advanced connectivity of devices and systems, different types of sensors are being used in various use cases. These generate huge volumes of data which can be processed and analysed over distributed systems for the benefit of industries like healthcare, supply chain, agriculture, transportation and so on. Elastic Stack is a set of open source solutions from Elastic. It is designed to help users move data of any kind from any type of source and index, search, analyze, and visualize that data in real-time. It consists of Filebeat for shipping the logs, Logstash for extracting and indexing, Elasticsearch for storing, searching and analysing, and Kibana for visualizing. It can be used along with Apache Kafka which is an open source distributed streaming platform generally used to build data streaming pipelines that move data between systems reliably. This paper explores the fundamentals of Apache Kafka and the Elastic stack and how Kafka can be used in conjunction with the Elastic stack for collecting and analyzing the data generated by networked sensors for IoT applications. It lays out the design for a system based on open source components, that monitors the data from the sensors on devices, performs analytics and alerts the concerned teams when required.

Keywords— Apache Kafka, Elastic Stack, Filebeat, Logstash, Elasticsearch, Kibana, IoT

I. INTRODUCTION

Internet of Things (IoT) is one of the most enabling technologies in the world today. From agriculture and manufacturing to retail, the way many industries do business is being changed by IoT. IoT device installations in agriculture help farmers collect data about their crops and livestock to optimize yields. In the healthcare industry, IoT implies better and faster services for patients and efficient monitoring of healthcare devices. Supply chain logistics connect shipping vehicles with sensors to monitor temperature and other metrics to ensure quality of the goods [1]. Gartner, Inc. estimates that the industrial, commercial and automotive IoT markets together will grow to 5.8 billion endpoints in 2020. This is an increase of 21% from 2019.

IoT devices comprise of various kinds of sensors that are networked and can generate numerous data points at high frequency. A simple temperature sensor can generate few bytes of data per minute while a connected vehicle can generate gigabytes of data per second. These huge data sets indicate specific metrics. They can be ingested, transformed, stored, queried and analysed in real time to gain more value from them [2]. Being able to collect data from various networked sensors at scale and send them to applications at different geographical regions can be used in various industries like supply chain, healthcare, agriculture. This high-volume telemetry data can be streamed and analysed to monitor the IoT device metrics and alert if there is any anomalous behaviour. A system based on open source tools can be setup to perform this at scale. The paper discusses the blueprint for the components used in such a system and how they work in conjunction.
The rest of the paper is organized as follows; Section II contains the proposed system for the use of Internet of Things. Section III discusses the technologies used to implement the architecture proposed. Section IV explains the qualitative results and section V concludes the paper with an idea of the future scope of the proposed architecture.

II. PROPOSED SYSTEM

Different kinds of sensor networks can be used to generate data points that indicate specific metrics. For example, Heating, Ventilation and Air Conditioning (HVAC) refers to different systems in place for providing air circulation, heating and cooling for residential and commercial buildings. They keep reporting the ambient temperature, desired temperature, air quality, humidity and energy consumption metrics using sensors like temperature sensors, air quality sensors, CO_2 sensors, humidity sensors and occupancy sensors [3].

In large organizations, these data points are collected frequently from many HVACs, and can be sent to a central IoT gateway that aggregates the data points as logs and ingests them into the system [4]. The paper discusses the architecture of an IoT data analytics pipeline and the technologies with which the system can be implemented. Fig. 1. shows a technology agnostic block diagram of the pipeline. Fig. 2. goes into the details of the technologies to be used in the implementation of the architecture.

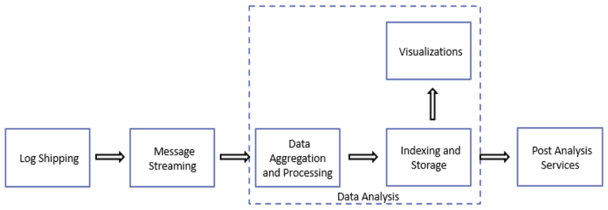

Fig. 1. Architecture diagram for log analytics pipeline

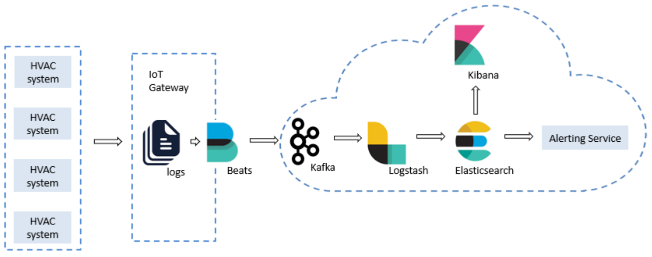

Fig. 2. Technology mapping for architecture components

In the proposed architecture, logs aggregated at the IoT gateway from various HVAC systems are forwarded by a log shipper (Filebeat). The Filebeat agent monitors the log files at regular intervals and forwards it to the output configured (Apache Kafka). Filebeat keeps checking the files for newer log entries whenever the sensor readings update the files. To avoid forwarding unnecessary logs, users can also specify which lines of the files to include or exclude while sending messages to Kafka.

The data is shipped as messages into an Apache Kafka cluster. Kafka acts as a layer that permits huge volumes of sensor data to be ingested with high performance. It enables data to be streamed from the edge(producers) to the consuming applications(subscribers) on the cloud. Kafka provides a pull-based messaging system. This provides the advantage of adding different consumers that consume at their own pace without affecting the performance.

Logstash pulls the data from Kafka, normalizes the logs and sends the data to an analytics data store (Elasticsearch). Logstash provides a broad array of filters that enable users to parse through the sensor logs, remove unnecessary fields and create relevant fields for the JSON output.

Elasticsearch enables the organization to perform search and analytics on the data. By knowing the thresholds of sensor data or data ranges, the management can set maintenance plans on real data instead of having fixed maintenance cycles where components may be forced to be replaced even if they're not nearing end of life. Using Elasticsearch helps generate a specific and useful maintenance plan which is more accurate and reduces production downtime and costs [5]. Appropriate visualizations can also be made to derive more value using Kibana, the visualization tool that works in conjunction with Elasticsearch.

Using the HVAC sensor metrics, the organization can analyze the health of the devices. It can also monitor the performance of the HVAC systems by tracking the metrics and taking corrective actions. This can be used as an early warning mechanism for failures. In case of failures or critical conditions, an alert system can be put in place to inform the concerned authorities about the scenario. For example, an alert can be triggered when the CO_2 levels (indicated by the CO_2 sensor readings) in the building crosses the threshold, so that appropriate action can be taken to increase the ventilation. For this, Kibana's alerting system can be leveraged or a customized solution can be built that observes the data, detects a condition and takes the appropriate action.

The applications processing the data that comes from the IoT edge can be deployed on the cloud. For example, according to the official documentation, Microsoft Azure provides an SLA of 99.9% on Kafka uptime. It allows seamless integration of the Elastic stack on the cloud to format, search, analyze and visualize data reliably and securely in real-time.

III. TECHNOLOGIES USED TO IMPLEMENT THE ARCHITECTURE

A. Filebeat

Filebeat is a lightweight shipper for files located locally. It is used to forward the log data and centralize it. The Filebeat agent can be installed on the server which keeps monitoring the log files or input locations specified and ships them to the outputs which could be Elasticsearch, Logstash, Kafka and a few others. When Filebeat is started, it starts the input configured to look at the locations specified for log data. For each log file located, a harvester is started. It reads the file line by line, sends the new data to libbeat which aggregates the events and forwards the aggregated content to the provided output.

Listing 1. shows a sample filebeat.yml configuration to get prospectors (inputs) from a folder of log files and output to a Kafka cluster. The output can be configured by setting options in the Outputs section of the filebeat.yml configuration file. In the 'prospectors' settings of the filebeat.yml file, the path of the log files that are the input to Beats is provided. Filebeat also provides 'include_lines' and 'exclude_lines' settings. These are to include or exclude respectively, only those lines of the log files that wholly or partly match the regular expression specified. Multiline pattern settings enable users to configure which lines are to be sent as a single event to Kafka.

```
filebeat.prospectors:
- type: log
  enabled: true
  paths:
    - /usr/local/etc/filebeat/sensor/*.log
  exclude_lines: ['ERR', 'WARN']
output.kafka:
 hosts: ["localhost:9092"]
 topic: 'sensorTopic'
 codec.json:
```

```
 pretty: true
```

Listing 1. Sample Filebeat Configuration

B. *Kafka*

Kafka was developed to solve the problem of delivering high volume event data to numerous subscribers. Built originally by the world's largest professional social network LinkedIn, it has been donated to the Apache Software Foundation and hence open sourced. As of October 2019, LinkedIn's Kafka deployments handle over 7 trillion messages every day.

Kafka helps move data to where we need it, in real time, reducing the problems that come with integrations between multiple source and target systems. This is done by allowing data streams to be decoupled from the systems. So, source systems generate events or messages to Apache Kafka and target systems will source their data from Apache Kafka.

Kafka functions as a cluster consisting of one or more servers called Kafka brokers. Producers push records into Kafka topics. A topic in Kafka cluster is a category name to which applications can publish, process and reprocess records. Topics can have zero or more consumers subscribed to the data that is written to them. Data is stored in the topics for a specified retention period. Brokers in a cluster are managed by Zookeeper. A cluster can have multiple zookeepers preferably odd numbered. Zookeepers store the metadata information related to the cluster [6].

Kafka topics can be divided into zero or more partitions. They contain records in a sequence that can't be changed. Each record in a partition is assigned and identified by its unique offset as shown in fig. 3. Order of the messages is maintained within partitions. Partitions let users to parallelize a topic by allowing the data produced to the topic to be split across multiple brokers (servers). Each partition can be placed in different servers to allow multiple consumers to read from a topic simultaneously. Writes to a partition are sequential. Reading messages however can be either from beginning or also from any point in the partition by providing the offset value [6].

Each partition is replicated across a number of servers that can be configured. This provides one of Kafka's important advantages, fault tolerance. Replica is a redundant unit of a topic partition. For each partition, there is a server that behaves as the leader and others (zero or more) acting as followers. The leader is responsible for all reads and writes for the partition and the followers make exact copies of the leader. One of the followers are assigned as the new leader automatically if the leader fails. To balance the load, a server can be a leader for a few of its partitions and follower for some others [6].

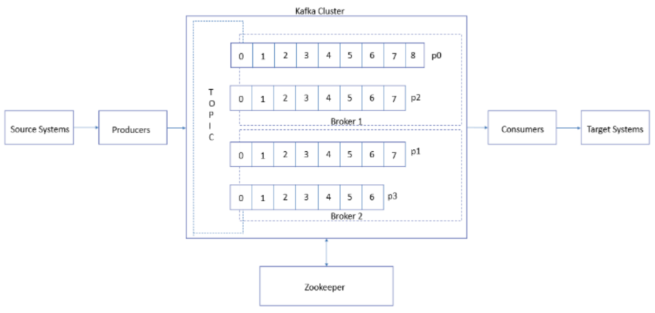

Fig. 3. Apache Kafka architecture

C. *Logstash*

Logstash is an open source data processing pipeline that accepts data from multiple sources, transforms it and forwards it to the configured output. The events are processed in the pipeline: inputs->filters->outputs. The input plugin allows handling of a specific input stream which could be events from Kafka, Filebeat, CSV files, Redis, RabbitMQ, TCP sockets, endpoints of HTTP/HTTPS APIs and others. The inputs can then be mutated to modify it, remove unwanted data, or add extra information. Various plugins are provided for many operations like parsing, extracting, managing and structuring the data [7]. For example, the Grok filter can be used to extract data from a string field that is of a known pattern, or custom patterns specified in a separate folder. Logstash also supports a huge range of destinations like Elasticsearch, CSV files, database, data analytics algorithms or just to be shown on the console. Listing 2 shows a sample 'logstash-sample.conf' file that matches a custom pattern specified in the 'patterns' directory. The index pattern is specified in the output plugin to split the data (HVAC data in Listing 2) into many indices for Elasticsearch. An index pattern can match many Elasticsearch indices. So, each document can be sent to the appropriate index depending on its content or timestamp, automatically [7].

```
input {
   kafka {
      bootstrap_servers => "localhost:9092"
      topics => ["sensors-topic"]
      codec => json
   }
}
filter {
   grok {
      patterns_dir => ['./patterns']
      match => [ "message","%{SENSORSFORMAT:HVAC}
%{OMIT:omitfield}" ]
   }
   mutate {
      remove field => [ "@version" ,"@metadata" ,"host" ,"omitfield" ]
   }
   split { field => "HVAC" }
}
output {
   elasticsearch {
      hosts => ["http://localhost:9200"]
      index => "HVAC-%{+dd.MM.YYYY}"
   }
}
```

Listing 2. Sample Logstash configuration file

D. Elasticsearch

Elasticsearch is an open source, distributed search and analytics engine. It can be used to store, search and analyze big volumes of data that could be logs, numerical, textual, structured and unstructured. For full text querying, it implements inverted indices with finite state transducers and for analytics, it makes use of column stores. Raw data flows from various sources. This raw data is parsed, normalized, transformed and enriched and then ingested to Elasticsearch for indexing. After indexing in Elasticsearch, users can run queries and aggregations on the data to obtain useful data summaries [8].

When a JSON object is sent to Elasticsearch, it automatically creates a document with the right mapping for each field without causing an overhead on performance. Elasticsearch provides dedicated APIs for mapping configuration. To customize the indexing, users can provide their own mapping definition [9]. Users can also prevent indexing unwanted data by disabling dynamic mapping for sections of the document selected. For some use-cases, multiple mappings are possible causing an overhead. For instance, a string field can be indexed for text search as a text field and for sorting and aggregations as a key word field. If the main use of such a field is known, overhead can be avoided by specifying beforehand how it will be indexed.

Listing 3. is a sample Index template that has mappings, index settings and patterns based on which the system determines whether the template pertains to an index that is newly added [7]. The data types for fields like carbon_dioxide, temperature and humidity are specified.

```
PUT 'http ://localhost:9200/_template/hvac'
{
   "template":"HVAC-*",
   "settings":{"number_of_replicas":4},
   "mappings":{
      "logs":{
         "properties":{
            "carbon_dioxide":{"type":"short"},
            "temperature":{"type":"byte"},
            "humidity":{"type":"byte"}
         }
      }
   }
}
```

Listing 3. Sample index template for HVAC-* indices [7]

E. Kibana

Storing a lot of information in a structured format is only useful if we get value out of it. One way to generate value is by getting better insights into what is happening using visualizations. Kibana provides a web-based interface for viewing, searching and analysing the data stored in Elasticsearch. The main components of the Kibana main view are – Discover, Visualize, Dashboards and Management. The management component is to configure index patterns and other details. It can be used to view the index fields and their properties. Formatters for the field can be modified to change how the field is viewed in the Kibana Graphical User Interface. Discover permits users to browse interactively and analyze the data. Visualize section is used to visualize the data using one or more of the provided plugins. It helps create tables, graphs, pie charts and more [10]. Dashboard is used to combine multiple saved visualize and discover outputs into a single view. Users can rearrange and resize the dashboard elements according to their needs [7].

IV. RESULTS AND DISCUSSION

A basic test of this architecture using readings from temperature, pressure and humidity sensors has shown that significant flexibility for configuration changes is present when using elastic stack. The streaming of large volumes of continuous sensor readings happens reliably and seamlessly using Apache Kafka. Elasticsearch not only behaves as a high-performance database but also as an efficient source to visualize data on the Kibana dashboards. The watcher UI, a feature of Elasticsearch is used to set the conditions to generate a threshold alert. All these offer a higher level of customization as they are open source tools.

V. CONCLUSION

The paper throws light on the problem of how HVAC systems' sensor data produced in one location needs to be aggregated and processed for analysis elsewhere for monitoring and alerting. A solution for handling this processing on a pipeline based on open source technologies is presented. The application of Filebeat to forward sensor data to Apache Kafka which ensures guaranteed, low latency and high throughput delivery to the data analytics pipeline is discussed. The consumer end of Kafka shows how Logstash is used for parsing and transforming the data into a structured format for indexing in the search and analytics engine, Elasticsearch. Visualizations are done using Kibana. The paper also provides an in-depth introduction to the technologies (Elastic stack and Apache Kafka) used throughout the proposed IoT data analytics pipeline.

The Elastic stack offers benefits like incredible scalability, reliability, flexibility to work with multiple data sources, support for big data and customizability that comes with being open sourced [11]. With many well-known organizations today like Uber, GitHub and Adobe deploying the Elastic stack, it is rapidly growing as a complete solution for operational and application logging.
Although the analytics pipeline discussed in the paper is from an Internet of Things perspective, its scope can also be extended for other use cases like processing and performing analysis on System logs, Web server logs, logs from database and so on.

REFERENCES

[1] M. Sheik Dawood, M. Jehosheba Margaret, R. Devika, "*Review on Applications of Internet of Things (IoT)*", International Journal of Advanced Research in Computer Engineering & Technology, Vol. 7, Issue 12, pp: 841-845, December 2018

[2] Mantripatjit Kaur, Anjum Mohd Aslam, "*Big Data Analytics on IOT: Challenges, Open Research Issues and Tools*", International Journal of Scientific Research in Computer Science and Engineering, Vol.6, Issue.3, pp.81-85, June 2018.

[3] Jordi Serra, David Pubill, Angelos Antonopoulos, Christos Verikoukis, "*Smart HVAC Control in IoT: Energy Consumption Minimization with User Comfort Constraints*", The Scientific World Journal, 2014. doi:10.1155/2014/161874

[4] Poonam M. Mahajan, "*WSN: Infrastructure and Applications*", International Journal of Scientific Research in Network Security and Communication, Vol.06, Issue.01, pp.6-10, 2018.

[5] Hong-Linh Truong, "*Integrated Analytics for IIoT Predictive Maintenance Using IoT Big Data Cloud Systems*", 2018 IEEE International Conference on Industrial Internet (ICII), pp: 109-118.

[6] Jay Kreps, Neha Narkhede, Jun Rao, "*Kafka : a Distributed Messaging System for Log Processing*", 6th International Workshop on Networking Meets Databases (NetDB 2011), Athens, Greece. Jun. 12, 2011.

[7] Marcin Bajer, "*Building an IoT Data Hub with Elasticsearch, Logstash and Kibana*", 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), Prague, 2017, pp. 63-68.

[8] D. Kalyani, Dr. D. Mehta, "*Paper on Searching and Indexing Using Elasticsearch*", International Journal of Engineering and Computer Science Vol. 6, Issue 6, pp: 21824-21829, June 2017

[9] Paro A., ElasticSearch Cookbook - Second Edition, Packt Publishing
Ltd, Jan 2015

[10] Sunny Advani, Meghna Mridul, Prof. S. R. Vij, Manil Agarwal, Loya Palak A., Kasturkar Sanketa S, "*Log Analytics Using ELK Stack on Cloud Platform*", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 4, pp: 50-52, April 2016

[11] Manuj Aggarwal, TetraTutorials Team, "*ElasticSearch, LogStash, Kibana (the ELK Stack) #3*", Packt Publishing, Jan 2018 [Online]

Authors Profile

Miss. Jayashri V is pursuing Bachelor of Engineering from R.V. College Of Engineering, Bangalore, India from 2016 (curremtly in 8th semester). Her areas of interest are Cloud computing, IoT, Web Development and Big Data Analytics.

Dr. Badari Nath K pursued Bachelor of Engineering from UVCE Bangalore, India in 2006 and Master of Engineering from PSG College Of Technology and his Doctorate from Visvesvaraya Technological University. He is currently working as Assistant Professor in Department of Computer Scientce, RVCE Bangalore, India since 2010. He has published more than 17 research papers in reputed international journals including IJITR and conferences including IEEE and Springer and it's also available online. His main research work focuses on Embedded & Real time systems.Image processing & Graphics Programming, Cloud Computing & IOT based Systems.