INTEGRATED MONITORING SOLUTION FOR CLOUD PLATFORMS (PaaS)

[1]Raghavendra P. Nakod, [2]Rashmi R.
[1]Student, [2]Assistant Professor
[1]Information Science,
[1]RV College of Engineering, Bengaluru, India

Abstract : Every business organization spend a lot of money and time developing and installing software to improve their operations. Cloud computing enables every business to access software on the internet as a service. Cloud computing is also a safe way of storing and sharing data. With this demand for cloud computing as every organization in the world is trying to move to the cloud, comes several security and monitoring concerns. Every application including cloud and standalone application requires performance monitoring. Monitoring plays a vital role in smooth running of the applications. Although some cloud services and enterprises provide monitoring services, they are neither full-fledged nor integrated completely. In this paper, an integrated monitoring solution is designed and implemented to provide an easy, simplified experience for monitoring for various cloud services. Providing an integrated system helps reduce the need for unnecessary redundant monitoring across various sub-systems.

IndexTerms **– Monitoring, Cloud, Integration, Systems, Design.**

I. INTRODUCTION

Monitoring is the process of managing activities that are currently running to verify that they are on-course and on-schedule in achieving the objectives and performance targets. It involves periodically obtaining performance related information from each of the services. Monitoring is a continuous evaluation that aims at providing all stakeholders with early detailed information on the progress or delay of the ongoing assessed activities. The aim of monitoring is to ascertain if the outputs and schedules framed have been reached so that necessary action can be taken to rectify the issues as early as possible.

Every standalone application or cloud application requires performance monitoring. Monitoring plays a vital role in smooth running of the application and the organization as it can detect underlying problems before they have an adverse effect. It can also detect problems that affect a user's productivity. Each cloud platform which is presently available provides several services like java applications, html5 applications, databases and schemas. Each of these services can be unique depending on the business context. This Monitoring solution provides readily-available integration tool for the components and it assesses the availability, quality and health status of these services by collecting different metrics from various data sources to provide an out-of-the-box solution-specific integrated dashboard to perform monitoring. It also triggers alerts for critical events so that necessary action can be taken by the users. The monitoring solution proposed in this paper uses Java applications that are deployed on the cloud platform for implementation.

II. OVERVIEW OF TECHNOLOGY USED

Cloud Platform can be of three types. Namely, Infrastructure as a Service, Platform as a Service and Software as a Service. The Cloud Platform includes a range of hosted services for security, storage and application development that run on of the vendors hardware. Each of the Cloud Platform services can be accessed by software developers, cloud administrators and other enterprise IT professionals over the public internet or through a dedicated network connection.

The various Cloud Platform offers services for developing, storage, block chain, big data, machine learning and the internet of things (IoT), as well as cloud management, security and developer tools.

The Cloud Platform used in this implementation is a Platform as a Service (PaaS) that provides comprehensive application development services and capabilities, which lets you build, extend and integrate business applications in the cloud. We can also develop and deploy Java and HTML5 applications in the cloud.

III. SOFTWARE DESCRIPTION

3.1 Metric Collection Tools

Metric collection can be done in various ways depending on the services we want to monitor. Metric collection tools are used for collecting the various metrics from the cloud platform currently in use. Telegraf is one such example. Metrices of the various applications that are presently running on that particular cloud platform can be collected by writing the necessary plugins to teach the metric collection tools what as to be collected. Metric collection tools are connected to one of the various time-series databases which are available. The different application performance metrics that are being collected in this implementation are availability of the application, no. of processes running in each app, average response time, CPU load, no. of busy threads, disk I/O, heap memory usage, OS memory usage, no. of requests per minute and used disc space.

Table 1. shows the different metrics that are collected for this implementation along with their meanings.

Table 1. Metrics and their meanings

Metrics	Meaning
Availability	Availability of the Java application (Started/Stopped)
No. of processes	No. of processes running in each application
Average Response Time	Time taken by the application server to return the results of a request to the user
CPU Load	Measure of amount of computation work that a CPU performs
No. of busy threads	No. of busy threads waiting to perform an operation
Heap memory usage	Heap memory used for dynamic memory allocation in percentage
OS memory usage	OS memory usage provides the percentage of physical and virtual memory
No. of requests per minute	No. of requests received for the application per minute
Used disc space	Amount of data that is stored in disc by the application

3.2 Time-series Database

Time series databases are databases which are ideal for regular time stamped data which are provided from various sources. These databases usually get the data to be stored from metric collection tools which collect the application metrics from the cloud platform which is in use. To continuously monitor any service, we must collect metrics at equally spaced points in time creating a series of measurements collected over time. We use time-series database instead of relational databases because of two reasons mainly. Scale and usability. TSDB can handle up to 4000 GB of data per day whereas relational databases are not designed to handle that scale. It has many other features that can provide a better user experience and make your life easier. Some of the TSDB that can be used are InfluxDB, Prometheus, Open TSDB, kairosDB, Elasticsearch, DalmatinerDB, Druid and Riak TS.

3.3 Visualization Tools

The data which is received from the time-series databases should be processed and modelled in a simple fashion. The data is usually provided by using visualization tools which provide easy user interfaces for understanding the data provided. The various visualization tools available can provide data to the user in different forms from tables and graphs to single stats and simple status bars.

The visualization tools have the added advantage of being able to provide the user notifications at the requisite team so that the retrospective action can take place. The visualization tools being used provide contextual information which is highly important in the field where the amount of metrics stored is massive. Some of the examples of visualization tools are Grafana, Kibana, Logstash, Kafka FluentD, weave scope and Dynatrace.

IV. SUGGESTED SOLUTION

4.1 Methodology

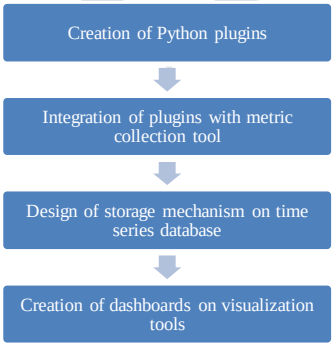

Fig. 1. Methodology

Fig. 1 describes the step-by-step methodology for design of the monitoring solution proposed by the paper. The existing standard monitoring solutions don't provide readily-available integration tools for various cloud components. As each cloud platform provides several services, the process of monitoring all the services can be difficult as each service can be unique depending on the business context and not all the metrics offered by individual monitoring services are relevant to a given context, or application. Metric collection can and has to be done from a multitude of data sources and solution-specific integrated dashboards are not currently present as part of cloud services of the various cloud providers. All these complexities are taken into consideration while developing the monitoring solution.

4.2 System Architecture

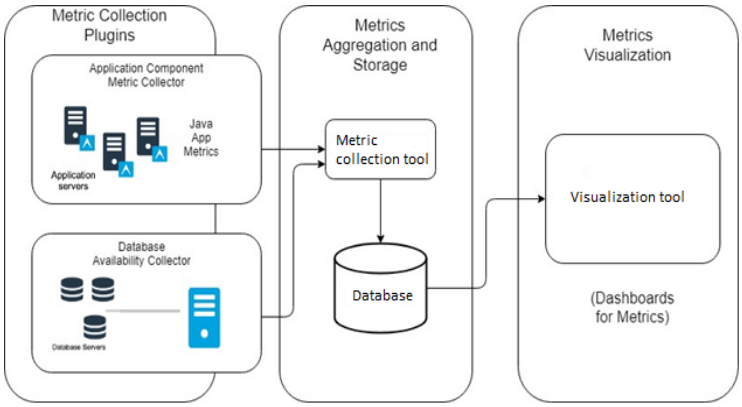

Fig. 2. System Architecture

Fig. 2 represents the system architecture of the monitoring solution developed. It uses one of the many popular stacks as its base on which everything is built upon. The architecture consists of mainly three components. They are: Metric Collection Plugins, Metrics Aggregation & Storage and Metrics Visualization. The entire metrics collection plugin architecture for the metric collection tool has been written in Python. The plugins are written for collecting metrics of Java applications, any database system and their availability from various data sources. These plugins contain many more modules within them to overcome all the complexities that arise when collecting the metrics as each service can be unique depending on the business context and not all the metrics offered by the monitoring services are relevant to a given context or application.

Metric collection depends on the services we want to monitor. There are several ways in which we can collect metrics depending on whether the application is a standalone application, or the application is deployed on vendor's cloud platform. If the application is a standalone application, then there are some specific ways to collect different performance metrics of the application. Similarly, for applications deployed on vendor's cloud platform, there are unique ways to collect application metrics and sometimes vendors themselves provide the application metrics. In the proposed implementation, Telegraf is used as a metric collection tool.

The second component of the architecture is Metrics Aggregation and Storage. It explains how the metrics of different services are collected from plugins continuously and stored in the database. The database used for storing these metrics is a Time Series Database (TSDB). TSDB are mainly used for monitoring, analysis and using data over long periods of time. Most of the times, the metrics collected from various data sources are JSON format or CSV file. As the databases have their own format for storing data, the data of these metrics should be converted to any of the TSDB format such as InfluxDB, Prometheus, etc.


```
"series": [
  {
    "name": "Availability_Check",
    "columns": [
      "time",
      "app",
      "host",
      "response_time",
      "running_processes",
      "state"
    ],
    "values": [
      [
        "2019-03-26T10:47:56Z",
        "helloworld",
        "INLN50942771A",
        0.023,
        1,
        1
      ],
      [
        "2019-03-26T10:48:14Z",
        "helloworld",
```


Fig. 3. Sample screenshot of the data collected

Fig. 3 shows the sample screenshot of the data collected in JSON format. The data shows some of the metrics of the Java application deployed on the vendor's cloud platform. It includes application name, host, response time, no. of processes running in that application and the state of the application in binary values. 1 indicates that the application has started and 0 represents that the application has stopped.

The final component of the architecture is Metrics Visualization. Visualization tools have various ways in which data can be visualized. For the solution proposed in this paper, Grafana was used for visualization of metrics. It is an open source visualization tool that can be used with a variety of data sources but most commonly used with InfluxDB and Prometheus. It helps users to easily create and edit dashboards It also has many features including the alert notifications via email whenever a critical event occurs.

V. RESULTS

The results of the proposed system are visualized using Grafana. It can also be visualized with any of the several visualization tools currently available in the market. Dashboards are created in the visualization tools for clear and concise understanding of the various metrics present in the time series database in use. Some of the dashboards which are usually created are:

5.1 Java Application Health

The dashboard shows the health of various Java applications running in the cloud platform for an account or sub-account as required by the user. The dashboard measures various application metrics such as the number of processes, average response time and CPU load.

The Java metrics can be viewed over different time periods as shown in the screenshots. The proposed system also has set thresholds above which necessary actions are taken by the system to inform the user. This information is sent to the user via various notification channels such as Slack, Email and Google Hangouts.

5.2 Database System Health

The dashboard shows the health of various database systems running in the cloud platform for an account or sub-account as required by the user. The dashboard measures various application metrics such as the number of processes, average response time., etc.

The database system metrics can be viewed over different time periods. The proposed system also has thresholds above which necessary actions are taken by the system to inform the user. This information is sent to the user via various notification channels such as Slack, Email and Google Hangouts.

5.3 Availability

The dashboard can also show the availability of various systems in the cloud platform for an account or sub-account as required by the user.

The availability can be viewed over different time periods as shown in the screenshots. The proposed system also has set thresholds above which necessary actions are taken by the system to inform the user. This information is sent to the user via various notification channels such as Slack, Email and Google Hangouts.

V. CONCLUSION

Cloud is the future as the world knows it. People and organizations want to move to the cloud for its ease of functionality, simplicity and lower cost. Year-on-year the number of functionalities provided by the cloud has regularly increased.

The cloud is usually accessed by the various platforms provided by companies such as Microsoft, Google, SAP and Amazon. These platforms were a major development in the cloud infrastructure and played a major role in bringing together the various users into the cloud ecosystem. With any major development in software, security and monitoring plays a very important role.

The various results in this report show how an integrated monitoring system can be used in the various cloud platform. With such a system, it can be seen in the report how the problems which arise can be effectively tackled and any future problems can be prevented.

VI. ACKNOWLEDGMENT

We want to thank SAP Labs India Pvt. Ltd as this proposed solution was carried out on SAP Cloud Platform.

REFERENCES

[1] "Reviewing Cloud Monitoring: Towards Cloud Resource Profiling," From the proceedings of 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, 2018, pp. 678-685.

[2] "Varanus: In Situ Monitoring for Large Scale Cloud Systems," From the proceedings of 2013 IEEE 5th International Conference on Cloud Computing Technology and Science, Bristol, 2013, pp. 341-344.

[3] "Towards an Out-of-the-Box Cloud Application Monitoring Framework," From the proceedings of 2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing (CS Cloud), Beijing, 2016, pp. 46-49.

[4] "A Context Based Scheduling Approach for Adaptive Business Process in the Cloud," From the proceedings of 2014 IEEE 7th International Conference on Cloud Computing, Anchorage, AK, 2014, pp. 948-951.

[5] "A Theoretical Approach to the Data-Oriented Scheduling Strategies across Multiple Clouds," From the proceedings of 2013 IEEE Sixth International Conference on Cloud Computing, Santa Clara, CA, 2013, pp. 942-943.

[6] "A Cloud Computing Platform for FY-4 Based on Resource Scheduling Technology," From the proceedings of 2016 International Conference on Advanced Cloud and Big Data (CBD), Chengdu, 2016, pp. 122-126.

[7] "Scheduling algorithm with plotter in cloud environment," From the proceedings of 2016 International Conference System Modeling & Advancement in Research Trends (SMART), Moradabad, 2016, pp. 68-70.

[8] "Cloud Monitor: Monitoring Applications in Cloud," From the proceedings of 2012 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, 2012, pp. 1-4.

[9] "A knowledge Base Driven Solution for Smart Cloud Management," From the proceedings of 2015 IEEE 8th International Conference on Cloud Computing, New York, NY, 2015, pp. 1069-1072.

[10] "Monitoring and Capacity Planning of Private Clouds: The Challenges and the Solution," From the proceedings of 2012 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, 2012, pp. 1-3.