ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.6

Real-Time Cloud Monitoring Solution Using Prometheus Tool and Predictive Analysis using Arima Model

Monika.K .A[1], Arun.V[2], Deepika Dash[3]
B.E Student[1, 2], Assistant Professor[3]
Department of Computer Science and Engineering
R V College of Engineering, Bangalore, Karnataka, India

Abstract:
Prometheus is a white box monitoring system developed to monitor multiple micro services running in a system. It has a modular architecture with several readily available modules called exporters, which helps to capture metrics from the running applications deployed on AWS instances. Prometheus joined the Cloud Native Computing Foundation in 2016 as the second hosted project, after Kubernetes. The system is scalable and is highly suitable for large distributed environments. Monitoring can also be done using AWS Cloud watch exporter. The flexibility of Prometheus over the Cloudwatch exporter is that it is an open-source tool and supports integration with a large number of application-specific exporters like elasticsearch, postgres, mongodb, redis etc.System metrics like CPU utilization, memory utilization, Disk I/O are scraped into Prometheus using Node-exporters. Application metrics and Business metrics are captured using elastic search exporter, postgres exporter, redis exporter, mongodb exporter, Cloudwatch exporter. All these exporters are deployed in a docker-environment. Required configurations are made in Prometheus set-up. Graphical visualization of all these metrics are done through visualization tool. Grafana which runs on port 3000 by default. The data handled is time-series data. A scrape request is sent by Prometheus for each metric at an interval of 5 seconds. Query results are stored in CSV files for predictive analysis and it is done using ARIMA (Auto-regressive integrated moving average) model. Evaluation metrics used is Mean Square error. Graphical plots are also created using python libraries. Hence, real-time monitoring along with predictive analysis of data using ARIMA model helps in making better decisions on resource usage.

Keywords: ARIMA, Elasticsearch, Grafana, MongoDB, Prometheus, Postgres, Redis

I. INTRODUCTION

Prometheus is an open-source system monitoring and alerting toolkit originally built at Sound Cloud. Since its adoption in 2012, many companies and organizations have adopted Prometheus, and the project has a very active developer and user community. Prometheus joined the Cloud Native Computing Foundation in 2016 as the second hosted project, after Kubernetes. The software was created because of the need to monitor multiple microservices that might be running in the system. Its architecture is modular and comes with several readily available modules called exporters, which helps to capture metrics from the most popular software. Prometheus is written in the Go language, and it ships with easily distributed binaries that can be used to get it running as quickly as possible. Prometheus interacts with the target system and its environment through monitors and actuators that are responsibility of the system designers to build and usually are application specific [1]. Also, it allows users to build interactive applications providing on-the-fly feedback during analysis. A plug-in based architecture is a crucial factor to provide flexibility to the real-world applications [2]. Time series is one of the tools for making predictions. Since Auto Regressive Integrated Moving Average (ARIMA) model was created by George E.P Box and Gwilym M.Jenkins in1970, it has been successfully used in forecasting economic, marketing, industry production, social problems. It is a linear model that fits for dealing with stochastic series [3].It will completely eliminate subjective impact from people and will give out a more objective and accurate result based on the historical time series data. The ARIMA models have also been proved to be excellent short-term forecasting models for a variety of time-series because short-term factors are expected to change slowly [4]. The proposed paper gives a new approach for cloud monitoring solutions. It is a combination of real-time monitoring using Prometheus tool and predictive analysis using ARIMA model. This combined approach helps in making wise decisions on resource usage.

II. LITERATURE SURVEY

Author in [5] describes the usage of 15 monitoring solutions for the distributed environments. The various tools under consideration are Zabbix, Nagios, Ganglia, Hyperic, DataDog, Prometheus and so on. They have discussed basic concepts of monitoring, areas where monitoring provides benefits and key factors that should be considered when choosing a monitoring solution. Authors in [6] have proposed a lightweight, scalable agentless system which can be configured, setup, and begin monitoring network health within minutes. The architecture utilizes a polling script to query Simple Network Monitoring Protocol (SNMP) demon for metrics, Prometheus Time Series Database instances for storage, and a Grafana Dashboard for metric presentation and alerting. The approach demonstrated shows the ease and security with which specific site network monitoring can be deployed in a cloud configuration. The main focus of author in [7] is to describe the process of designing and implementing a software solution that would analyze multi-tenant cloud cluster resource usage metrics on customer level and then use the information to bill customers accordingly. The two main tools under consideration are Kubernetes and Prometheus with Grafana dashboarding solutions. This thesis also presents all the necessary background information to what has gone into building this kind of tool. It also digs into challenges that the process presented. It also evaluates the tool that was generated as a part of the process. Author in [8] describes an application developed for the analysis of monitoring data from computers, servers or cloud infrastructures using ARIMA model. The analysis is based on the extraction of patterns and trends from historical data, using elements of time-series analysis. The tool provides features for identifying network traffic, congestion and peak usage times, and for making memory usage projections. Authors in [9] describe how a shift from desktop applications to cloud-based software as a service (SaaS) applications deployed on public clouds, increases the competition to provide the same quality of service without any compromise. In order to survive in such a competitive market, cloud-based companies must achieve good quality of service (QoS) They deal with the problem by proactive dynamic provisioning of resources based on the prediction from ARIMA model results, which can estimate the future need of applications in terms of resources and allocate them in advance, releasing them once they are not required.

III. PROBLEM STATEMENT

World has adapted itself to the use of micro services and development operations (DevOps). This shift adds a great deal of complexity. Instead of having to monitor one system, there is a challenge to oversee system manifold services. Monitoring includes collecting, processing, aggregating, and displaying real-time quantitative data about a system. Though numerous monitoring systems available, not all of them are fit for monitoring large and distributed systems, hence there is a need to develop monitoring systems that are flexible and scalable. With Prometheus, all these questions can be solved, by exposing the internal state of applications. By monitoring this internal state, alerts can be thrown and action can be taken upon certain events.

IV. OBJECTIVES

A number of objectives are set, to be attained for a successful implementation of the project. The objectives of the cloud monitoring system are listed below.

- To design a monitoring system that continuously checks the health of all the servers deployed in a cloud environment.

- Metrics for the whole system can be classified as System metrics which includes CPU Utilization, Memory Utilization and Disk Input/Output of all the servers, Application metrics customized with respect to per-specific applications deployed on each server. Applications can include various database servers like Postgres, Elastic-search, MongodB,Redis. Business metrics include User metrics to monitor the rate of usage of the software at various intervals of time.

- To create dashboards for each f these metrics using Grafana tool that runs on port 3000. This helps in analysis of the entire system from time to time.

- To perform predictive analysis of data using ARIMA model.

V. IMPLEMENTATION AND METHODOLOGY

Initially, an AWS machine named “mt_monitor” was created which was responsible for the monitoring of the entire system architecture. AWS configuration was done on this machine to extract information about all the servers through AWS Command Line Interface (CLI). All the production and staging machines required monitoring. Hence those machines were extracted. Prometheus was installed on ”mt_monitor‟ machine. Various exporters like Node exporters (to scrape System metrics and runs on port 9100), Postgres exporter which runs on port 9187, Redis exporter which runs on port 9121, MongoDB exporter which runs on port 9104, Elasticsearch exporter (Elastic search runs on port 9200), CloudWatch exporter which runs on port 9106 were installed based on the specific application running on the AWS instance. Prometheus Query Language (PromQL) was used as the data Query language to analyse various metrics and display them on Hyper Text Transfer Protocol (HTTP) web page. To enhance the visualization, a data Visualization tool called Grafana which runs on port 3000. Prometheus (which runs on port 9090) was imported into Grafana tool, which was used to create dashboards for the various metrics. Predictive analysis of data was done using ARIMA model for the real time time-series data collected on a daily basis from the real time Prometheus monitoring system. As ARIMA model works on time series data it is not possible to analyze non-stationary data, so the data should be converted to stationary data. The process of converting non-stationary to stationary depends on various parameters like lag, difference and order of moving average. ARIMA model uses two intervals to find the moving average, which is based on the window size used to calculate the overall usage average The general definition of ARIMA model is given as ARIMA(p,d,q). *p* is the order also known as number of time lags of the AR model, *d* is the degree of differencing or generally, the number of times the data have had past values subtracted, and *q* is the order of the MA model. The values chosen for ARIMA(p,d,q) in this paper are ARIMA(5,1,1) for all the system metrics. Stationary of time-series data is achieved through Box-Jenkins approach. Differencing is done between two consecutive observations. Mathematically, it is

$$y'_t = y_t - y_{(t-m)}, \qquad (1)$$

where m in (1) is the number of seasons.

VI. EXPERIMENTAL ANALYSIS AND RESULTS

Experimental Analysis of software deployed involves strategies and procedures followed to verify the successful implementation of the software. All the functional and non-functional requirements of the software have to be achieved before delivery of the software.

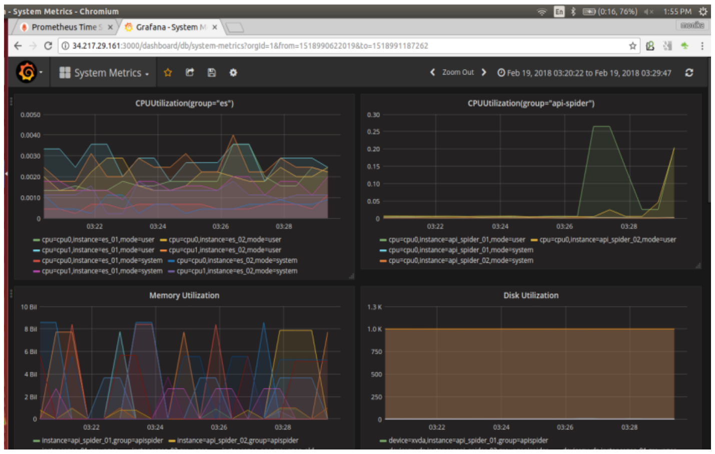
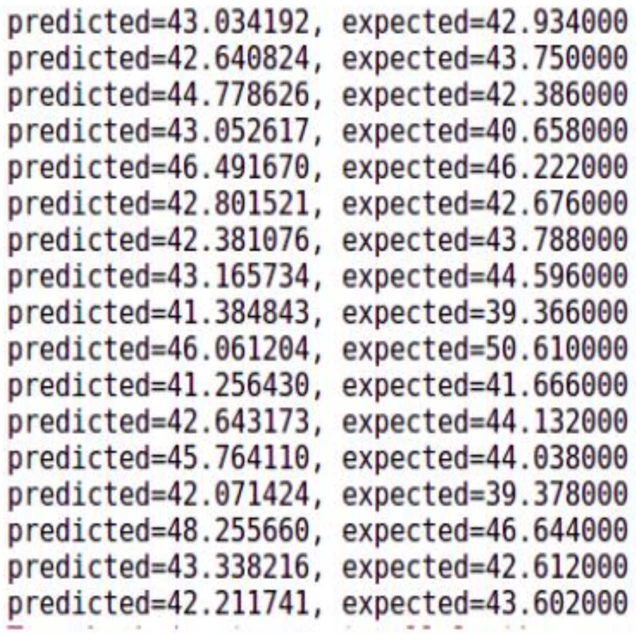

System Metrics
CPUUtilization(group="es")
CPUUtilization(group="api-spider")
Memory Utilization
Disk Utilization

Figure.1. System metrics (cpu utilization, memory utilization, disk utilization)

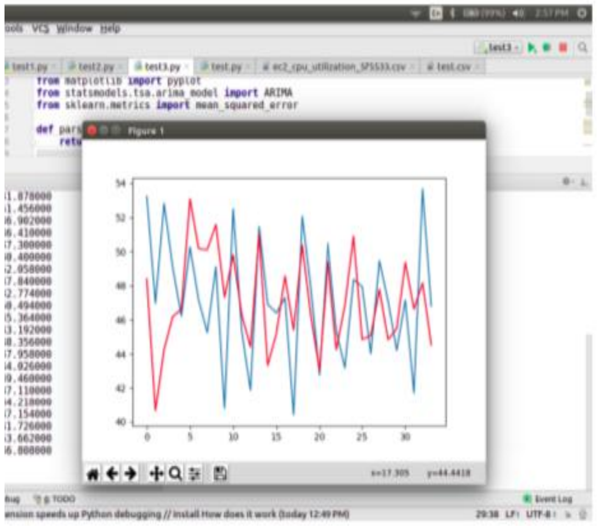
The Figure 1 describes graphs for various system metrics collected for the system monitoring. These include CPU utilization, Memory utilization and Disk input/output operations. These statistics are collected for a period of 1 month at once. The data generated can be used as input to predictive analysis of modeling. ARIMA model is used as the predictive forecast model for statistical analysis of time-series data generated by the Prometheus monitoring system. The evaluation metric used for the experimental analysis is Mean Square Error (MSE). Mean Square Error is defined as the average of the squares of the errors or deviations between the predicted and expected values of time series data. Lower the mean squared error values; greater is the efficiency of the model. Hence MSE is inversely proportional to efficiency of the model. In the ARIMA model deployed for predictive analysis of time-series data for system metrics, the MSE value obtained is 11.81. The MSE estimators are obtained by defining the unknown target components such as the trend.



```
predicted=43.034192, expected=42.934000
predicted=42.640824, expected=43.750000
predicted=44.778626, expected=42.386000
predicted=43.052617, expected=40.658000
predicted=46.491670, expected=46.222000
predicted=42.801521, expected=42.676000
predicted=42.381076, expected=43.788000
predicted=43.165734, expected=44.596000
predicted=41.384843, expected=39.366000
predicted=46.061204, expected=50.610000
predicted=41.256430, expected=41.666000
predicted=42.643173, expected=44.132000
predicted=45.764110, expected=44.038000
predicted=42.071424, expected=39.378000
predicted=48.255660, expected=46.644000
predicted=43.338216, expected=42.612000
predicted=42.211741, expected=43.602000
```

Figure.2. Test results of arima model for a sample test data set of 1 month interval

The Figure 2 shows the predicted and expected values for test data set ran on the trained ARIMA model for CPU Utilization.



```
from matplotlib import pyplot
from statsmodels.tsa.arima_model import ARIMA
from sklearn.metrics import mean_squared_error
```

Figure .3. Evaluation metric (mean squared error)

The above Figure 3 shows the Mean Squared Error (MSE) which is taken as the evaluation metric for the predictive analysis of system metrics. It includes CPU utilization, memory utilization and disk I/O.

VII. CONCLUSION

Prometheus is a scalable open-source software tool that provides a monitoring solution to deal with time-series numerical data. To monitor services using Prometheus, an HTTP endpoint for Prometheus runs on port 9090. All the metrics collected by Prometheus are listed and can be visualized with simple graphs. Along with real-time analysis, predictive analysis of data is done using the stored data. The time-series data model used is ARIMA model. The achieved MSE is 11.81. Hence the system deployed is beneficial to monitor all the micro services and DevOps and hence manage the resources efficiently and make better decisions on resource usage.

VIII. FUTURE WORK

Though Prometheus is one of the top cloud monitoring tools used for real-time applications, it has a few limitations. It has no support for log storage and it does not support a long term durable storage of data as well. Since it does not have a dash boarding solution of its own, it depends on Grafana dashboards, adding some additional setup complexity. Secondary storage devices are used to store the data generated from Prometheus queries. All the limitations of the Prometheus tool are addressed in this paper. To enhance it further, alert managers can be configured which can send alerts through emails or slack messages. This functionality is supported as a feature by the Prometheus tool. These features are a part of real-time monitoring solutions .Instead, more importance has to be given to the predictive analysis of data, which gives a better insight of the performance of system for any load. It helps in analyzing various situations in advance and make rightful decisions.

IX. REFERENCES

[1] Konstantinos Angelopoulos, Fatma Basak Aydemir, paolo Giorgini, John Mylopoulos, “Solving the Next Adaptation problem with Prometheus”, IEEE Xplore-2016, Volume 3, No.5, 2016, pp. 1-5

[2] Lucantonio Ghionna, Luigi Granata, Gianluigi Greco, Massimo Guarascio,“Prometheus: A suite for Process Mining Applications”, Conference Paper, ResearchGate, 2012.pp. 6-11

[3] Penf Chen,Hongyong Yuan, Xueming Shu, “Forcasting Crime Using The ARIMA Model”, IEEE Xplore ,Volume 3, No. 4, 2008, pp. 627-629

[4] Xu Ye, “The Application of ARIMA model in Chinese Mobile User Prediction”, IEEE International Conference on Granular Computing, 2010, pp. 111-117

[5] Łukasz Kufel,Tools For Distributed Systems Monitoring, Foundations Of Computing And Decis Ion Sciences,Vol 41,No.4,2016

[6] Morgan Brattstrom, Patricia Morreale,”Scalable Agentless Cloud Network Monitoring“, IEEE Xplore,2017

[7] Mikko Korhonen,Analyzing Resource Usage On Multi Tenant Cloud Cluster For Invoicing, University Of Oulu,Faculty Of Information Technology And Electrical Engineering, 2017, pp.1-50

[8]. Nicolas Gutierrez and Manuela Wiesinger-Widi, “ AUGURY: A time-series based application for the analysis and forecasting of system and network performance metrics”, arXiv:1607.08344v1 , 28 jul 2016

[9]. Rodrigo N. Calheiros, Enayat Masoumi, Rajiv Ranjan, and Rajkumar Buyya ,”Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications QoS”,IEEE Transactions On Cloud Computing, Vol. 3, No. 4, October-December 2015

X. ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this project work. We would like to thank them all. We would like to thank Dr. G Shobha, Head of Department, Computer Science & Engineering, RVCE, Bengaluru, for her valuable suggestions and expert advice. We would also like to thank Dr. K. N. Subramanya, Principal, RVCE, Bengaluru, for his moral support towards completing our project work.