International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 04 | April - 2022 Impact Factor: 7.185 ISSN: 2582-3930

On-Premise Server Monitoring with Prometheus and Telegram Bot

Favaz Mohammed Rawoof[1], Mohd Tajammul[2], Fakhrun Jamal[3]

[1]*School of CS & IT, Jain University, Bangalore, Karnataka, India*
[2]*School of CS & IT, Jain University, Bangalore, Karnataka, India*
[3]*Department of Computer Science, AMU Aligarh, UP, India*

Abstract: On-premise Servers can be classified as a crucial backbone of a company. Sensitive data like Financial Records, User Information are stored on on-premise servers. This makes it necessary to be kept in a separate room with supervision and security checks, making it difficult to monitor the server periodically. This paper researches on-premise servers which lack remote monitoring and introduces a user-friendly solution. By making monitoring data available remotely via a Telegram bot, monitoring the servers becomes easy on-site or off-site and only requires the administrators to physically inspect the server for initial configuration, periodic checks, and any physical repairs.
Monitoring software like Prometheus can be used to collect server metrics at set intervals. Telegram provides an API to create and control bots that can send messages to its user. By storing the metrics collected by Prometheus in a cloud database, the Telegram Bot can be requested to fetch these data which can be viewed by the server administrator remotely.

Key Words: *Server Monitoring, Telegram Bot, Python Service, Prometheus.*

1. INTRODUCTION

Telegram is a cross-platform, open-source, cloud-based messaging service launched in 2013. The platform provides end-to-end encrypted messaging, voice and video calls, and an API to automate replies to users [8]. Server monitoring metrics can be formatted in a mobile-friendly format and sent as notifications. This eliminates the need to be physically in the server room for monitoring the status of on-premise servers.
Prometheus is a monitoring and alerting open-source toolkit [9]. This software allows to collect monitoring data and storing it in a database, this data can be further pushed to other services to analyze, view, or share it remotely.
Python services can do various tasks like fetching and pushing data to and from databases, interacting with Telegram bots to send and receive messages, etc. [10].
Replit is an online coding platform where users can write and host applications, Build Websites using just a Browser. It also integrates GitHub, a code repository, making it easy to import and export projects to and from Replit. Replit also provides Team Collaboration tools to make working with other developers easier. [25]
Mongo DB is an open-source NoSQL Database Program that stores data in JSON format. MongoDB Atlas provides an easy way to host and manage your data in the cloud. [26]
The aim is to create a python program that can be set up to send Monitoring Metrics selected by the user from Prometheus and other Command-Line Tools at intervals set by the user. When requested by an authorized Telegram user, The bot will provide the latest metrics from the database. This enables the System Administrator to monitor servers remotely and be up to date on the status of the server anytime, anywhere.

2. RELATED WORKS

The paper "Monitoring System of OpenStack Cloud Platform Based on Prometheus" has a similar view on monitoring servers that are on the open stack cloud. The paper designs an efficient way to monitor the open stack cloud platform by using tools like Prometheus and visualize it by using Grafana [3]

The paper "Network's Server Monitoring and Analysis Using Nagios" states the importance of monitoring network topologies and server metrics at regular intervals using NAGIOS, avoiding any system failure by notifying the network administrator via e-mail.[2]

A research "Implementation System Telegram Bot for Monitoring Linux Server", uses a Telegram bot to execute direct terminal commands in a Linux server environment allowing the author to remotely monitor the server. This method requires its users to memorize the commands beforehand and then send them to the bot. [1]

In a research paper titled "Design of Telegram Bots for Campus Information Sharing", the author created a telegram bot for a university to mass circulate upcoming events, classes, timetables, and other information among staff, teachers, and students easily. The author shows how easy it is to share information in a technologically advanced era where most of the users have an online messaging app like Telegram.[4]

A paper titled "Remote Monitoring and Control by Embedded Database Design and Web Server Implementation Using SQL Lite database and boa webserver", ports a boa webserver to an arm Linux platform and uses SQL lite to remotely monitor virtually anything with appropriate sensors by using a portable pocket-sized development board MINI2440. [5]

Remote monitoring is crucial in areas other than servers, in a paper titled "Design and Implementation of Remote Monitoring System for Welding Machines based on Web", The authors tackle the problem of monitoring clusters of welding machines in their workshop. By creating a web server using HTML, JavaScript, PHP, and Ajax and connecting them to welding machine endpoints, the system helps to monitor welding stations at mass and mitigate any issues.[6]

In a survey paper titled "Network and Server Resource Management Strategies for Data Centre Infrastructures," the author dives deep into various resource management methods data centers choose to efficiently deploy, monitor, and run virtual machines. The paper shows various methods of monitoring virtual machines in data centers.[7]
Authors have discussed various aspects of implementing cloud security and best practices to be followed on cloud platforms like Amazon Web Services and Microsoft Azure, aiming to protect user information. [11-24]

3. RESEARCH GAP AND PROBLEM FORMULATION

From the literature review, it is understood that traditional On-Premise servers lack remote monitoring dashboards, unlike online servers which have Amazon CloudWatch and Azure Insights. Authors have proposed various ways to monitor various types of systems and other IoT devices by installing monitoring software like Prometheus, NAGIOS, etc. The solutions provided require users to have in-depth knowledge of coding and terminal commands.

Monitoring on-premise servers are crucial as modern-day servers are deployed to store and access confidential and sensitive data that can't be stored in the public cloud. This sensitive data is accessed on-premise and is required to be highly available for business continuity. [27] An application is proposed to make these metrics available in Telegram via a bot, this will help system administrators monitor the on-premise server remotely.

4. OVERVIEW OF THE PROPOSED APPLICATION

A. SYSTEM ARCHITECTURE

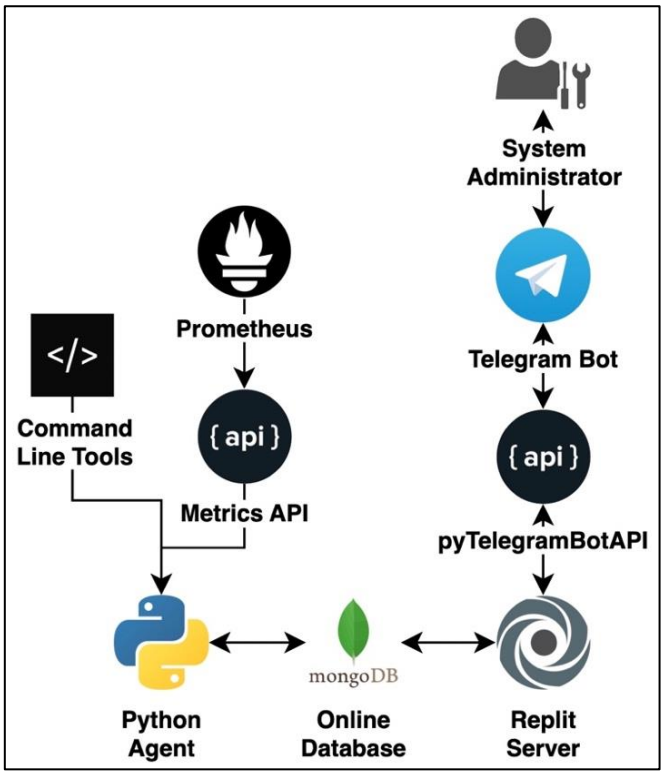

Figure 1: System Architecture

The system architecture as shown in **Figure 1** shows what the proposed application uses in the background.

Open-source monitoring software like Prometheus and other command-line tools are used to collect a wide range of system metrics.

Services like MongoDB, Replit, and Telegram are used to remotely access and view the metrics.

Thus, allowing remote monitoring of an on-premise server.

B. SYSTEM WORKFLOW

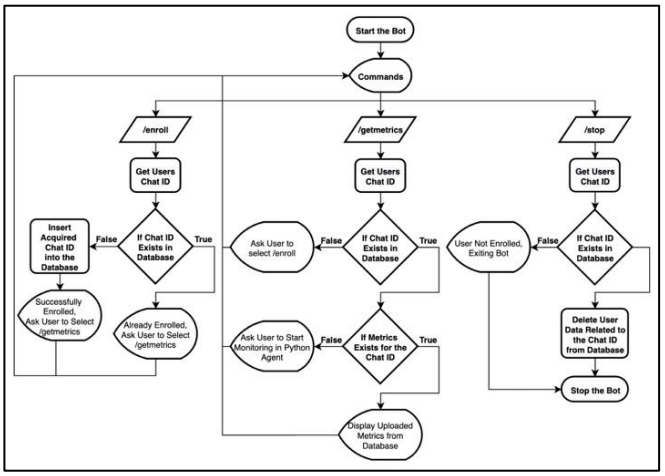

Figure 2: Telegram Bot Work Flow

Figure 2 shows how the Telegram Bot workflow occurs. Users are presented with three options when they first start the bot using /start command as shown in **Figure 5**.

Enroll Command gets the user Chat ID and inserts it into the database as a key with an empty value. The bot asks the user to run /getmetrics command if it already exists.

Get Metrics commands gets the user Chat ID and check if it exists in the database, the bot then gets the value metrics associated with the key Chat ID from the database. The bot asks the user to click start monitoring in the python agent if the value is empty.

Stop command stops the bots custom command keyboard, gets the user Chat ID and checks if exists in the database, and deletes all data related to the key Chat ID.

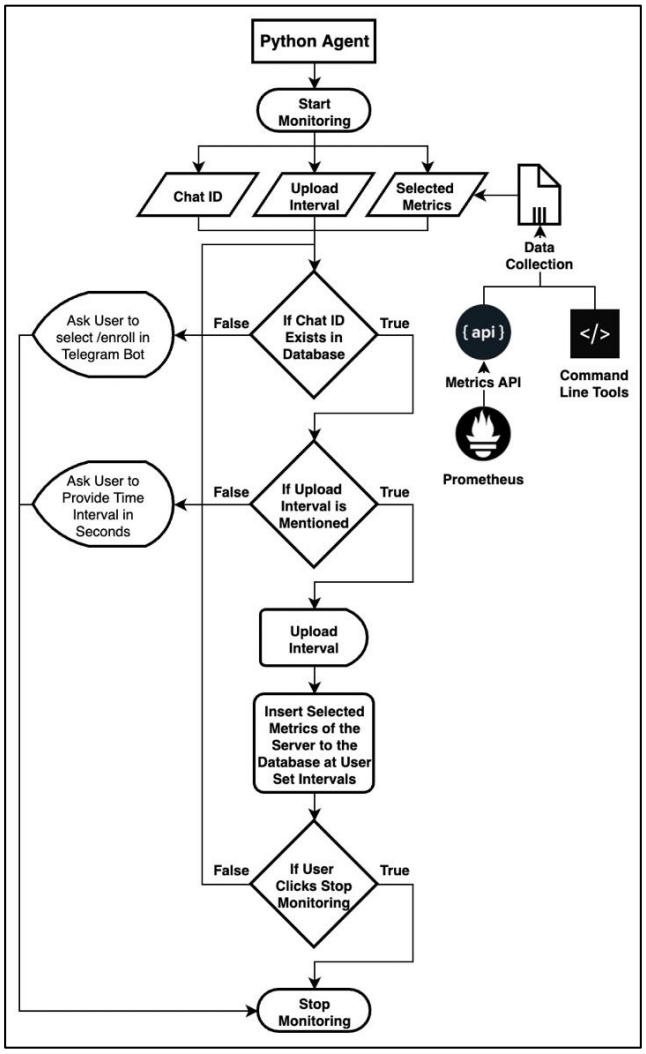

Figure 3: Python Agent Work Flow

Figure 3 Shows the python agent workflow, it takes the Chat ID that the user got from the telegram bots enroll command, and Interval at which the selected metrics will be uploaded to the database.

The python agent then checks if the Chat ID as a key exists in the database, then it inserts the system metrics as value to the Chat ID key.

C. APPLICATION USER INTERFACE

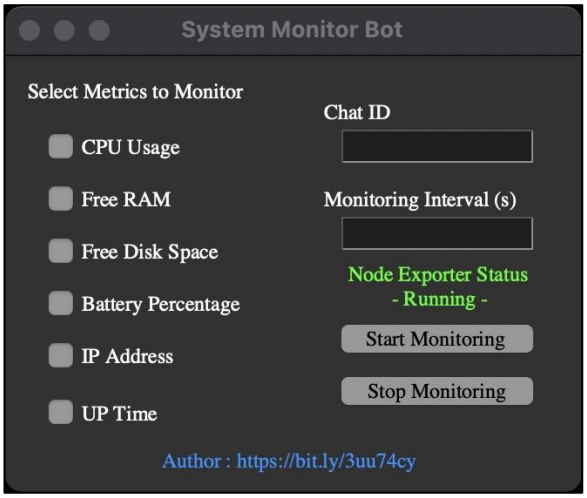

Figure 4: Python Agent UI

Figure 4 Shows the python agent UI which will be running on the on-premise server.
Node Exporter Status is a dynamic label that changes if Prometheus is not running in the system.
The application provides options to select various metrics which will be uploaded to the database when the user provides a Chat ID from the Telegram Bot, Intervals in Seconds at which the metrics will be updated in the database. The agent will stop monitoring once the user clicks the stop monitoring button or if the user commands the Telegram Bot to stop.

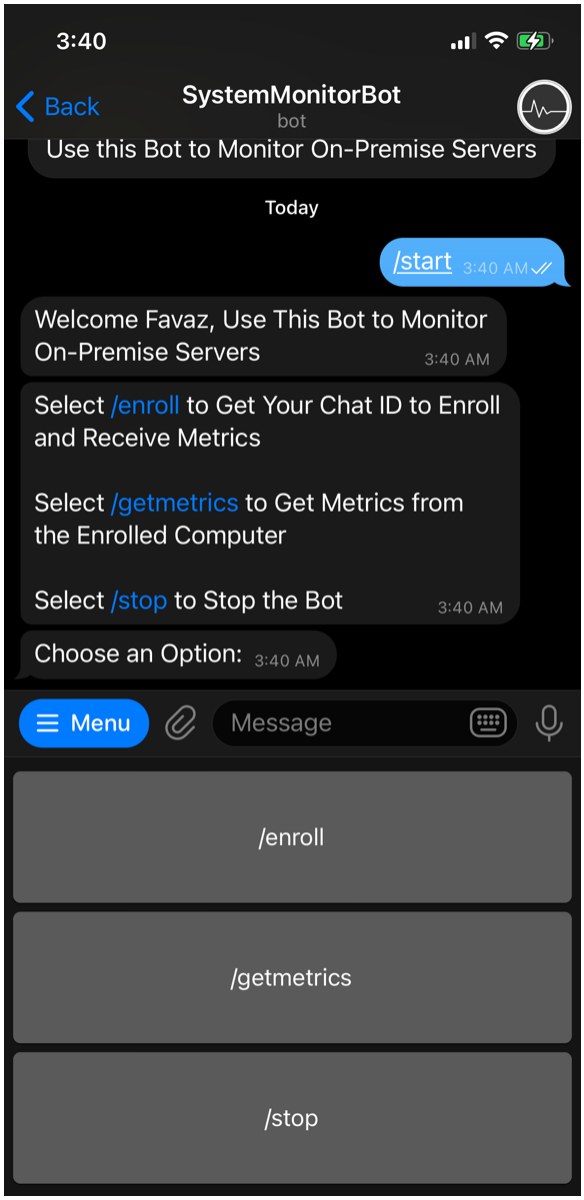

Figure 5: Telegram Bot Welcome Screen

Figure 5 shows the interface of the Telegram Bot when the user first starts it up. The user is presented with three commands and is provided with a proper explanation of what each command does.

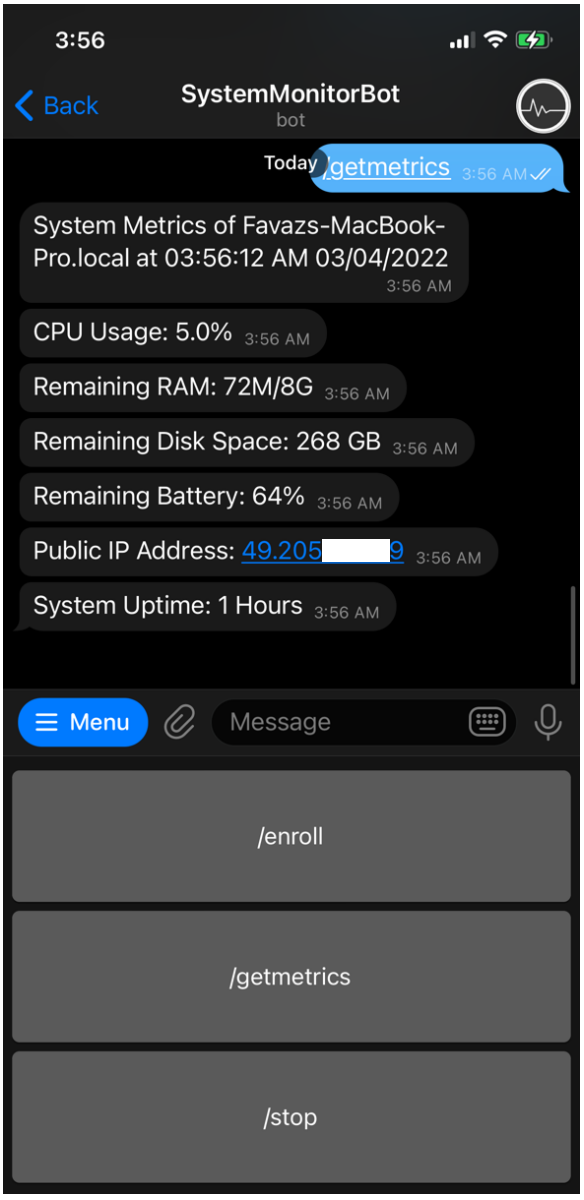

Figure 6: Receiving Metrics from System

Figure 6 shows what the user will receive if he runs the command /getmetrics. Getting these metrics will help the administrator remotely figure out the system's current state.

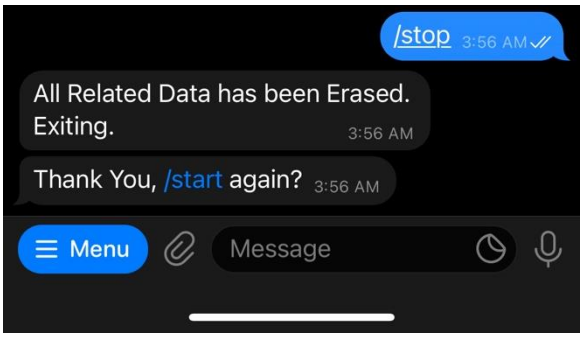

Figure 7: Stopping the Bot

Figure 7 shows the bot's response when the user runs /stop command. Stop command completely erases the user Chat ID and any metrics related to the Chat ID from the database, this will also force the python monitoring agent to stop monitoring.

5. CONCLUSION

This paper proposes a solution that helps mitigate the lack of remote monitoring of on-premise servers.

It is understood that the impact of not monitoring servers can vary from Overloading, Overheating, Data loss to Malicious attacks.

Monitoring data collected by Prometheus and other command-line tools can be extracted and formatted for user-friendly viewing and then pushed to an online database. The system administrator can view this data remotely using the Telegram Bot, Making remotely monitoring on-premise servers an easy task.

6. FUTURE SCOPE

Additional features that can be added to the proposed solution in the future are as follows:

- Introduce compatibility with other monitoring tools.
- Introduce compatibility with other messaging platforms.
- Notify multiple administrators simultaneously.

REFERENCES

[1] Mohammad Idhom, Ronggo Alit, Henni Endah Wahanani, Akhmad Fauzi "Implementation System Telegram Bot for Monitoring Linux Server", 2018

[2] J. Renita, N. Edna Elizabeth, "Network's Server Monitoring and Analysis Using Nagios", 2017

[3] Lei Chen, Ming Xian, Jian Liu, "Monitoring System of OpenStack Cloud Platform Based on Prometheus", 2020

[4] Hari Setiaji, Irving V Paputungan, "Design of Telegram Bots for Campus Information Sharing", 2018

[5] Roopa Bammidi, Deepa Kundala, "Remote Monitoring and Control by Embedded Database Design and Web Server Implementation Using SQL Lite database and boa web server", 2017

[6] Zhang Yu-Long, Cao Heng, Hu Jing-Feng, Wang Jin-Cheng, "Design and implementation of remote monitoring system for welding machine based on web", 2018

[7] Fung Po Tsoa, Simon Jouetb, Dimitrios P. Pezaros, "Network and Server Resource Management Strategies for Data Centre Infrastructures: A Survey", 2017

[8] Telegram, https://core.telegram.org/bots, 2021

[9] Prometheus, https://prometheus.io/docs/introduction, 2021

[10] CoreTechnologies, https://www.coretechnologies.com/products/AlwaysUp/Apps/RunPythonScriptAsAService.html, 2021

[11] Alam T., Tajammul M., Gupta R. (2022) Towards the Sustainable Development of Smart Cities Through Cloud Computing. In: Piuri V., Shaw R.N., Ghosh A., Islam R. (eds) AI and IoT for Smart City Applications. Studies in Computational Intelligence, vol 1002.

[12] Tajammul, M., Shaw R.N., Ghosh A., Parveen R. (2021) Error Detection Algorithm for Cloud Outsourced Big Data. In: Bansal J.C., Fung L.C.C., Simic M., Ghosh A. (eds) Advances in Applications of Data-Driven Computing. Advances in Intelligent Systems and Computing, vol 1319.

[13] Tajammul, M, Parveen, R., "Cloud Storage in Context of Amazon Web Services", International Journal of All Research Education and Scientific Methods, vol. 10, issue 01, pp. 442-446, 2021.

[14] Tajammul, M., Parveen, R., "Auto Encryption Algorithm for Uploading Data on Cloud Storage", BIJIT - BVICAM's International Journal of Information Technology, vol. 12, Issue 3, pp. 831-837, 2020.

[15] Tajammul, M., Parveen, R., "Key Generation Algorithm Coupled with DES for Securing Cloud Storage," International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958, Volume-8 Issue-5, June 2019 no. 5, pp. 1452–1458, 2019.

[16] Tajammul M., Parveen R., "Two-Pass Multidimensional Key Generation and Encryption Algorithm for Data Storage Security in Cloud Computing", International Journal of Recent Technology in Engineering, Vol. 8, Issue-2, pp. 4152–4158, 2019.

[17] Tajammul M., Parveen R., "Algorithm for Document Integrity Testing Pre-Upload and Post- Download from Cloud Storage", International Journal of Recent Technology in Engineering, Vol. 8, Issue-2S6, pp. 973–979, 2019.

[18] Tajammul, M., Parveen, R., "Auto Encryption Algorithm for Uploading Data on Cloud Storage", BIJIT - BVICAM's International Journal of Information Technology, vol. 12, Issue 3, pp. 831-837, 2020.

[19] Tajammul, M., Parveen, R., and M. Shahnawaz, "Cloud Computing Security Issues and Methods to Resolve: Review," Journal of Basic Applied Engineering and Research, vol. 5, no. 7, pp. 545–550, 2018.

[20] Tajammul, M., Parveen, R., Delhi, N. (2018). Comparative Study of Big Ten Information Security Management System Standards, International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 5, Issue 2, pp. 5-14, 2018.

[21] M. Tajammul, R. Parveen, N. K. Gaur and S. D, "Data Sensitive Algorithm Integrated with Compression Technique for Secured and Efficient Utilization of Cloud Storage," 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON), 2021, pp. 1-9, doi: 10.1109/GUCON50781.2021.9573648.

[22] Tajammul, M., Parveen, R., (2017). Comparative Analysis of Big Ten ISMS Standards and Their Effect on Cloud Computing, 978-1-5386-0627 8/17/31:00c2017IEEE; 9001; 362367.

[23] Tajammul, M., and R. Parveen, "To Carve out Private Cloud with Total Functionality," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 831-835, doi: 10.1109/ICACCCN51052.2020.9362826.

[24] M. Tajammul, R. Parveen and I. A. Tayubi, "Comparative Analysis of Security Algorithms used in Cloud Computing," 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), 2021, pp. 875-880, doi: 10.1109/INDIACom51348.2021.00157.

[25] Replit, https://docs.replit.com/, 2022

[26] MongoDB, https://www.mongodb.com/docs/atlas/getting-started/, 2022

[27] Favaz Mohammed Rawoof, Mohd Tajammul, "A Survey on Remote On-Premise Server Monitoring", Journal of Emerging Technologies and Innovative Research, vol. 9, Issue 2, Feb, 2022