iJRASET

International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 Issue: XI Month of publication: November 2022

DOI: https://doi.org/10.22214/ijraset.2022.47360

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue XI Nov 2022- Available at www.ijraset.com

System Observability and Monitoring for Multi Cloud Native Network Service based Function

Vijayalakshmi H R[1], Dr. Dakshayini M[2], Sandeep Roa[3]
[1, 2, 3]*B M S College of Engineering*

Abstract: The simplicity and agility of public Cloud infrastructure is driving large service based and application oriented Enterprises to host their business functions as Cloud Native Services and Applications. These Enterprises often host their applications and services in multiple cloud infrastructures either due to business, compliance and functionality requirements. On these cloud infrastructure platforms, applications and services are primarily designed and deployed as microservices which are small and independent services and functions. Each service microservice is typically self contained and interacts with other services using predefined Service based interfaces or APIs. As these services could be hosted in multiple cloud environments on infrastructures such as virtual machines and containers, there's a need to provide a holistic visibility of services across many infrastructures, form factors and service interfaces.

Keywords: Microservice, API, Virtual Machine, Visibility, Multicloud.

I. INTRODUCTION

Observability is the ability to understanding the internal state of any system by just looking at external outputs of all the system. It has been increasingly applied to improving the overall performance of distributed Information technology systems. In this context, observability will use three different types of telemetry information — traces, logs and metrics. It provides a profound perceptibility into distributed systems and also allow teams to know the root cause of a multitude of issues and improve the performance of the system. Due to the much increased number of interconnected pieces in distributed systems, there are more and different causes of failure. Furthermore, distributed systems are continuously modified, and each modification may result in different kinds of failure. Understanding a current issue in a distributed setting is extremely difficult, hence observability is better suited for the unpredictable nature of distributed systems.

Over the last several years, Enterprises have quickly adopted serverless, microservice, and container technologies as well as cloud-native infrastructure services like AWS. In modern distributed systems, it takes thousands of processes running on-premises, in the cloud, or in both to track an event back to its source. However, the numerous communication channels and interdependencies in these distributed systems make it difficult for traditional monitoring techniques and tools to keep track of them. Because it provides more control over complicated systems, observability is significant. Simple systems are simpler to manage since they contain fewer moving elements. In most cases, keeping an eye on the CPU, RAM, databases, and networking settings is sufficient to comprehend these systems and implement the necessary fixes.

In a distributed system, various components are spread over a number of computers (or other computing devices) connected to a network. These devices divided the work up and coordinated their efforts in order to complete the task more quickly than they would have done if only one device had been in control. The utilisation of distributed systems is an important development in IT and computer science as an increasing number of connected tasks become too massive and complex for a single machine to handle on its own. Distributed computing, however, offers more advantages than the traditional computing environment. Distributed systems minimize the threat posed by having a single point of failure.

Distributed computing used to be expensive, challenging to set up, and manage. However, because to the increased capabilities of software as a service (SaaS) platforms, distributed computing is now simpler to use and more accessible to both large and small businesses. The maintenance of databases is one of the many computing tasks that require distributed computing.

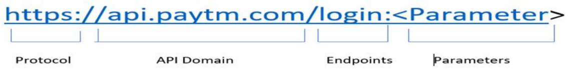


Fig.1 : Application Architecture

Application is built typically using microservices architecture where there is an API for everything. Each API consist of two parts are API path and API endpoints which is basically the action or method and it can have some parameters like login name , amount. An application architecture consist of basically microservices implementing the API with the data stored in a database then the API path consist of protocol and the API domain. API consist of 4 path which is typically https which is secure to ensure that everything was securely then comes API domain to which the API service is delivered then comes to login which is basic in endpoint which is actually implementing that API then some parameters for the API this is how an Application architecture.

II. REVIEW OF LITERATURE

A. Network Function Virtualization

Network Function Virtualization(NFV) is a replacement of network appliances hardware with virtual machine. In the last decade, we saw a significant change in how modern, internet-scale applications are being built. Microservices are a new breed of distributed system designs made possible by cloud computing (infrastructure as a service) and containerization technologies, which were made popular by Docker. successful businesses like Twitter and Netflix have been able to develop extremely scalable, effective, and dependable systems and provide their customers with more services more quickly.

Traditionally, for every component and function of the network, network operators were deploying physical, proprietary equipment and devices. The classical network appliances includes message router, session border controller, fixed access network node is dedicated to hardware. It is a slow process and it takes times to restart. This results in expensive installation costs and limitations on network modification and improvement. After that, it will transition to network function virtualization (VNF).

Network Function Virtualization is having a firewall with the hardware technology such as virtual machine. NFV environment are more dynamic than traditional ones, which might requires scaling up with additional features to cope. NFV also demands for process realignment that traditional and virtual infrastructure can be managed simulataneously so then it is transition into the Cloud Native Network Functions(CNF).

B. On observability and monitoring distributed system

The availability and functionality of client apps have a significant impact on a company's ability to succeed commercially. Modern software development paradigms like DevOps and microservice architectural styles have caused applications to be decoupled into services with complex interactions and dependencies. It is a significant problem to observe and monitor such distributed systems. The writers of the research article offer a qualitative investigation to comprehend the difficulties and best practises in the area of observability and monitoring of distributed systems.

In this paper, authors tells that observability becomes a prerequisite to ensure stable services and maintain the development of client applications.

In recent years, Many information technology teams have successfully transitioned to cloud computing for their services. Even yet, there are still issues with how these services are run and thoroughly monitored on the cloud. Despite the fact that conventional monitoring solutions can be used to keep track on conventional IT infrastructure, the complexity of distributed systems exceeds the capabilities of monitoring tools to handle the complexity. As a result the cloud environments are more complex and dynamic. Monitoring is a key challenge to the adoption of these technologies since emerging developments like the Internet of Things (IoT) and microservices add to the complexity.

In order to ensure the companies must be able to connect emerging technology and methodologies to the problems they face. To integrate new solutions into both established enterprise architectures and developing cloud architectures, processes and best practises are required.

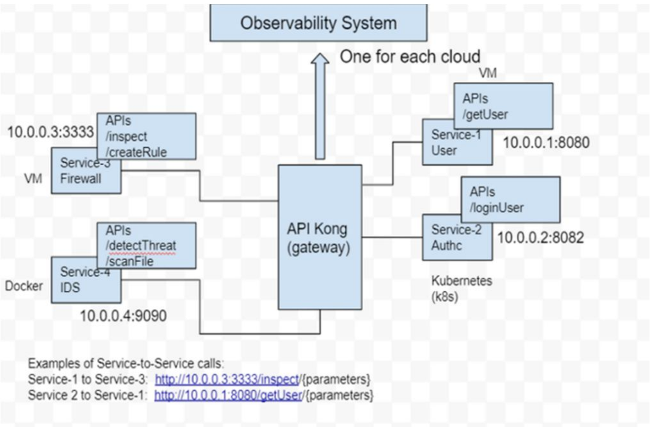
C. Disaster Recovery in Single and Multi Cloud Environment

Many medium-sized and large-sized businesses have significantly increased their use of cloud computing during the past few decades. Numerous advantages of cloud computing include lower costs and easier access to data. Cloud computing services offer quick access to the applications and a decrease in the infrastructure expenses, small- and medium-sized businesses employ them for a variety of functions. Due to the possibility that the data saved with the cloud storage providers may be extremely sensitive and the providers may not be trustworthy, integrity and privacy of cloud-based data services must be protected, which is a major challenge for cloud computing. Most contemporary businesses and organizations use cloud computing primarily to lower the overall cost of infrastructure ownership and to benefit from Information technology.

The capacity to store data while guaranteeing their availability is a distinctive feature of the cloud, which is crucial when storing sensitive data Due to the construction of numerous copies in various data centres, current methods for data backup and recovery for Single-Cloud setups require enormous amounts of storage space. The usage of a Single-Cloud paradigm has hazards that include harm caused by humans, natural calamities, and also software errors and hardware faults. This may result in issues with data recovery as they relate to Single-Cloud and Multi-Cloud systems have been discussed in this work. The paper's goal was to discuss important issues that cloud-based data recovery experts have solved.

III. PROPOSED WORK

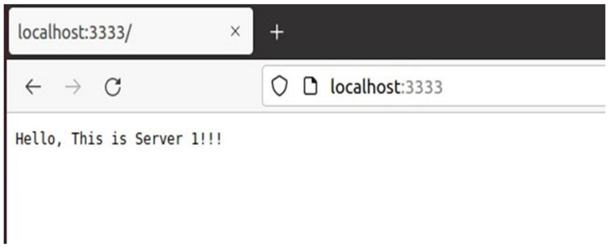
The proposed solution is to build a framework to provide a holistic multi-cloud observability and monitoring of services and applications, its telemetry and service interfaces. We explore ways to monitor, collect and share telemetry that includes, but not limited to Service Inventory - Inventory of services in each cloud are Tenancy of the service and Form factor of the service. Tenancy of the service - in which cloud or platform is the service is running Form factor of the service - how is the service deployed in virtual machine, Docker, Kubernetes Pod.



Examples of Service-to-Service calls:
Service-1 to Service-3: http://10.0.0.3:3333/inspect/{parameters}
Service 2 to Service-1: http://10.0.0.1:8080/getUser/{parameters}

Fig. 2. Observability System Architecture

There is a sample services which is called echo server, The first service is running at a local server 1 which is starting at the port number 3333. Similarly the second and third server is running at the port number 4444 and 5555 respectively.



```
Hello, This is Server 1!!!
```

Fig.3 : Server 1 is running on the Localhost at port no. 3333

An application with all the necessary functionality is deployed as one package using the open source technique called as Docker. The process of dockerizing involves using a docker container to build, deploy, and execute applications. To deploy a dockerized application on a remote server is to transfer the application's image with docker pull and then use docker run. This runs the application in a container similar to do it the development environment.

Docker reduces the need for additional infrastructure resources for development and allows for the sharing of containers created for individual processes with other applications. While using less memory than virtual machines, instances of these containerized applications are much less expensive to develop and deploy. So all the three services are dockerized so that it can run anywhere in the cloud through the Kong gateway.

With Kong Gateway, Service and Route objects expose the services to customers. The process of configuring API access will begins with the specification of a Service. A service in Kong Gateway is a representation of an external upstream API or microservice — for example, a billing API, a data transformation microservice, and so forth. A service's URL, where it watches for requests, is its most important property.

The protocol, host, port, and path can all be specified separately or in a single string to specify the URL. You must add a Route to the Service before you can start making requests against it. After reaching Kong Gateway, routes control how (and if) requests are forwarded to their Services. A Service may have a large number of Routes.

Making requests using Kong Gateway has become possible after configuring the Service and the Route.

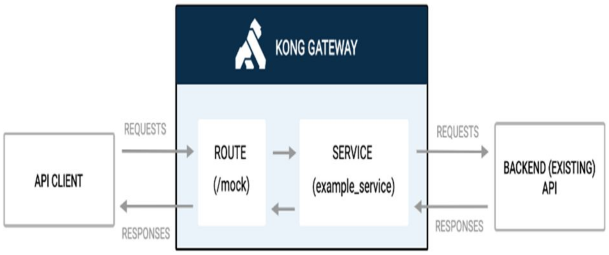
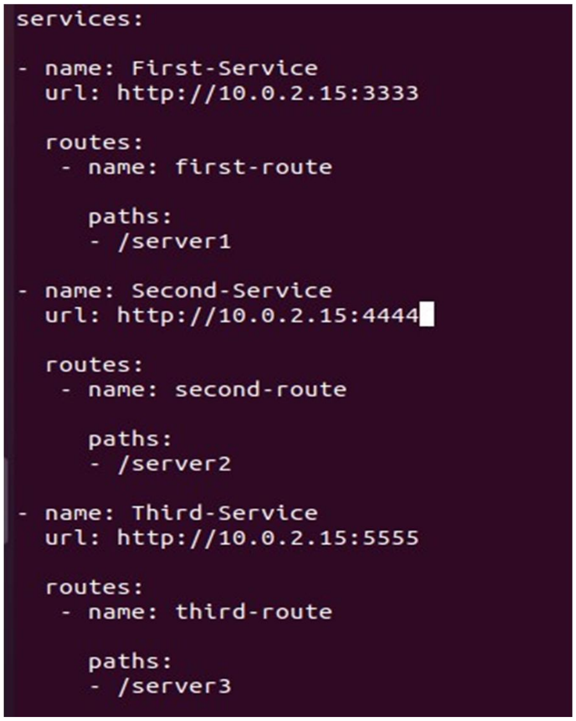


Fig. 4: The route of requests and responses from the Service to the backend API.



```
services:

- name: First-Service
  url: http://10.0.2.15:3333

  routes:
   - name: first-route

     paths:
     - /server1

- name: Second-Service
  url: http://10.0.2.15:4444

  routes:
   - name: second-route

     paths:
     - /server2

- name: Third-Service
  url: http://10.0.2.15:5555

  routes:
   - name: third-route

     paths:
     - /server3
```

Fig.5 : Kong Configuration

This paper describes the solution in two layers are Visibility Dashboard and Doc abstraction layer. In the Visibility dashboard which will aggregate all the data coming from different cloud data and visualize the data in the from graph, charts. In DOC Abstraction layer which is dockerized it will intern go to every cloud and get the data and keep the data with the front end middle access.

As a result, Prometheus and Grafana are the visibility dashboard which will aggregate all the data in the form of JSON format put the data to Grafana it gives the data to visualize in the from of graph and charts. That graph are shown in figure 4, 5, and 6 of Bandwidth, Caching, and Latency respectively.

IV. RESULTS AND DISCUSSION

A cloud-based computing and the modern Internet has been built using native technologies. Users and corporations often employ cloud-based applications. On the other hand, service failures and decreases in quality could have disastrous consequences for our society. In addition, web applications have evolved into complex distributed systems that are difficult to understand and utilize, increasing their mistake potential if not managed effectively. As a result, Understanding, observing, stopping, spotting, and fixing any issues that can lead to failures.

The mechanism for gaining Observability in a cloud-native environment is proposed in this thesis. Observability aims to provide a deeper understanding of the complex distributed system that web applications have evolved. A proof of concept is used to show how the proposed observability framework works, and it is .then deployed in real world environment,

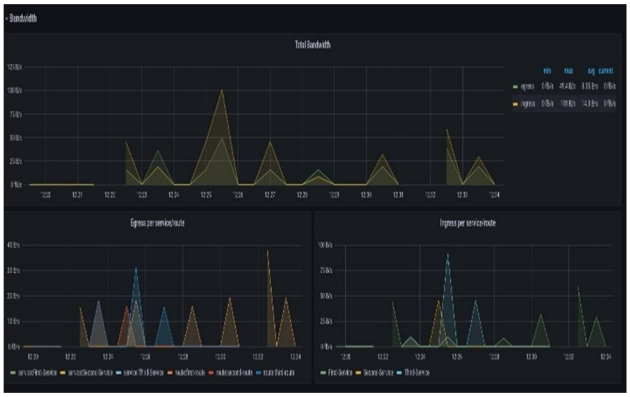


Fig.6 : Total Bandwidth per Service/Route

In the figure 4, represents the total bandwidth of services/route and Egress and Ingress of three service/route.
Total Bandwidth measures the data transfer rate, capacity and quality of a services or route. Bandwidth is commonly measured in bits per second(bps). Egress traffic can be defined as packets that originated from a pod inside the network and travel out through switches and routers to an external destination. Ingress traffic that originates outside of a given network and travel to a pod inside the network.

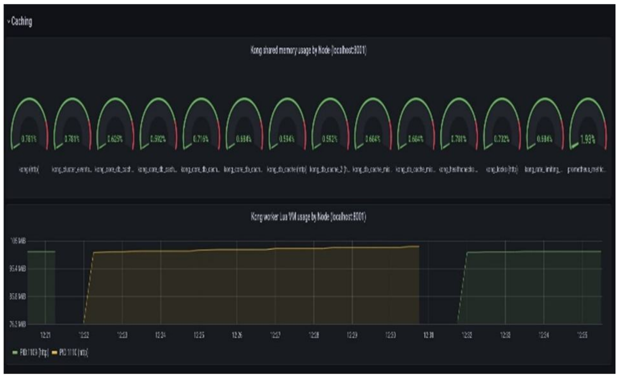

Caching
Kong shared memory usage by Node (localhost:8001)
Kong worker Lua VM usage by Node (localhost:8001)

Fig.7 : Kong shared memory and Kong worker Lua VM usage by Node (localhost:8001)

A cache is a high-speed data storage layer used in computing that holds a part of the data that is often temporary in nature. By doing this, It is possible to respond to future requests for that data more quickly than by directly accessing the main storage location. Data that has already been retrieved or calculated can be efficiently reused due to caching.
In the figure 5, represents the Kong shared memory usage by node running at the localhost:8001 and Kong worker Lua Virtual machine usage by Node running at the localhost:8001.

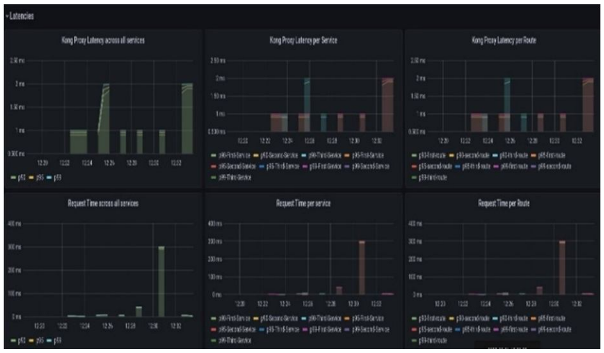

Latencies
Kong Proxy Latency across all services
Kong Proxy Latency per Service
Kong Proxy Latency per Route
Request Time across all services
Request Time per service
Request Time per Route

Fig.8 : Latencies of all services and route

In computer networking, latency is refers to the amount of time it takes for a data packet to move between two specified points

Applications and data used by enterprises are now processed globally due to the growth of cloud computing services. The architecture of a Single-Cloud environment is an integrated environment with a cloud platform, storage, infrastructure, and security challenges that arise when applications and data are handled by a Single-Cloud provider. Sensitive information, like medical records, are uploaded to the cloud for security, but consumers have no control over the information and cannot detect whether it is being used inappropriately. Data security cannot be guaranteed by the owners of the data that is stored in a Single-Cloud environment.

The cloud service provider has complete control over it. Building a relationship of trust with the cloud service provider is crucial when determining whether to go to the cloud. Employees who may be trustworthy but interested as well as employees who might compete with or threaten the data owner are other aspects that may create uncertainty about the security of the data in the cloud. Sensitive data stored in the cloud has been tampered with it by some cloud service providers and their users. A single-cloud environment may also encounter the problems with data loss and/or high management costs for handling large amounts of data. Data security is maintained by distributing data and the applications across a variety of different clouds. High data availability is achieved by this multi-cloud architecture, which also provides the ability to manage resources, balance tasks, and store data securely. In this system, no Single-Cloud architecture has enough information to allow data to be decrypted without access to other parts of the data in other clouds, the security of a service provider may be less significant in this system. The owner of the data may have assurance in the cloud provider to protect it.

If sensitive data is to be secured against misuse or other types of interference, security is essential in cloud computing. Data owners who store sensitive data in the cloud must make use of its scalability, device independence, and remote access capabilities. The cloud service providers, who may be sincere yet hesitant, have complete control over the data of their clients (HBC). Data security in a Single-examining the Cloud environment is necessary because it is easy for data to be stolen when any part of the cloud is compromised. Because a Single-Cloud environment is insufficient for a malicious influence to access all massive data, data theft does not occur when encrypted data are dispersed among the multiple clouds. Therefore, a multi-cloud architecture offers the necessary data protection for the cloud.

V. CONCLUSION

Organizations require strategies and tools that may improve the QoS, reliability, and accessibility of their cloud-native applications as cloud-native adoption has become the standard for new information technology and development.

Now so many API, so many gateway, so many cloud the question is we don’t know where what is running, where it is running so that why the visibility. Orchestration –This will help to identify which API is running where to do the patch management. If an API is having issue, if it is broken you release a patch which is basically fix, that fix deploy for the API whenever it is running so we don’t know where the API is running are how many instances of API are running will not be able to patch it also. Lack of security posture – some API’s may not be secure or protected some API’s may be public or private. API may helps which you want to private. Private API’s may be public which you want to private suddenly is publicly access so how to find those issues. API protection – Protect API from external attacks, abuse fraud.

Prometheus and Grafana are the visibility dashboard which will aggregate all the data in the form of JSON format put the data to Grafana it gives the data to visualize in the from of graph and charts. Visualization of data in the form of graphs enable us to recognize emerging trends, gives a better understanding of statistics and creates a better scaling strategy. The ability to process information quickly and easily, compare data, draw inferences, and make sense of complex data is improved. It makes it simpler for us to discover strongly linked parameters.

Users of data visualization software for businesses can learn more about their vast data. Recognize new patterns and errors in the data is advantageous to them. Humans process images more quickly than any time-consuming tabular forms or reports. With the continuous increase of data, data visualization is essential for businesses to rapidly identify data trends and new patterns.

Multicloud solutions are already the standard for the majority of businesses, and barring a significant amount of industry consolidation, this trend is often likely to be uninterrupted for the foreseeable future. However, cloud computing platforms are evolving, and industry standardisation is growing. In the future, it should be simpler to move workloads between services and move services from one cloud to another. Use a multicloud monitoring solution to keep track on the multicloud environment up to that time. In terms of security, this is extremely critical because an expanded attack surface can make it challenging to effectively manage or protect against privacy issues.

REFERENCES

[1] What is observability. [online]. Available: https://www.splunk.com/en_us/data-insider/what-is-observability.html

[2] What is Multicloud [online]. Available: https://www.splunk.com/en_us/data-insider/what-is-multicloud.html

[3] Pareek, P. Cloud Computing Security from Single to Multi-Clouds using Secret Sharing Algorithm. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) (2013).

[4] Pokharel, M., Lee, S., & Park, J. S. (2010). Disaster Recovery for System Architecture Using Cloud Computing. 2010 10th IEEE/IPSJ International Symposium on Applications and the Internet, 304–307.

[5] C.-C. Chang, S.-R. Yang, E.-H. Yeh, P. Lin, et J.-Y. Jeng, " A Kubernetes-Based Monitoring Platform for Dynamic Cloud Resource Provisioning ", in GLOBECOM 2017 - 2017 IEEE Global Communications Conference, Singapore, Dec. 2017, p. 1-6, doi: 10.1109/GLOCOM.2017.8254046.

[6] Sample cloud-native application with 10 microservices showcasing Kubernetes, Istio, gRPC and OpenCensus. https://github.com/GoogleCloudPlatform/microservices demo.

[7] Cloud Native Transformation Pini Reznik, Jamie Dobson, Michelle Gienow O'Reilly Media, Inc. ISBN: 9781492048909

[8] Distributed Systems Observability, Cindy Sridharan, O'Reilly Media, Inc. ISBN: 9781492033424.

[9] Chaos Engineering Observability, Russ Miles, O'Reilly Media, Inc. ISBN: 9781492051039.

[10] N. Marie-Magdelaine, T. Ahmed, et G. Astruc-Amato, "Demonstration of an Observability Framework for Cloud Native Microservices ", in 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), avr. 2019, p. 722-724.

[11] R. Picoreti, A. P. do Carmo, F. M. de Queiroz, A. S. Garcia, R. F. Vassallo, et D. Simeonidou, Multilevel Observability in Cloud Orchestration in 2018

[12] Sulochana, M., & Dubey, O. Preserving Data Confidentiality Using Multi-Cloud Architecture. Procedia Computer Science (2015).

[13] Suguna, S., & Suhasini, A. (2014). Overview of data backup and disaster recovery in cloud. International Conference on Information Communication and Embedded Systems (ICICES2014), (978), 1–7

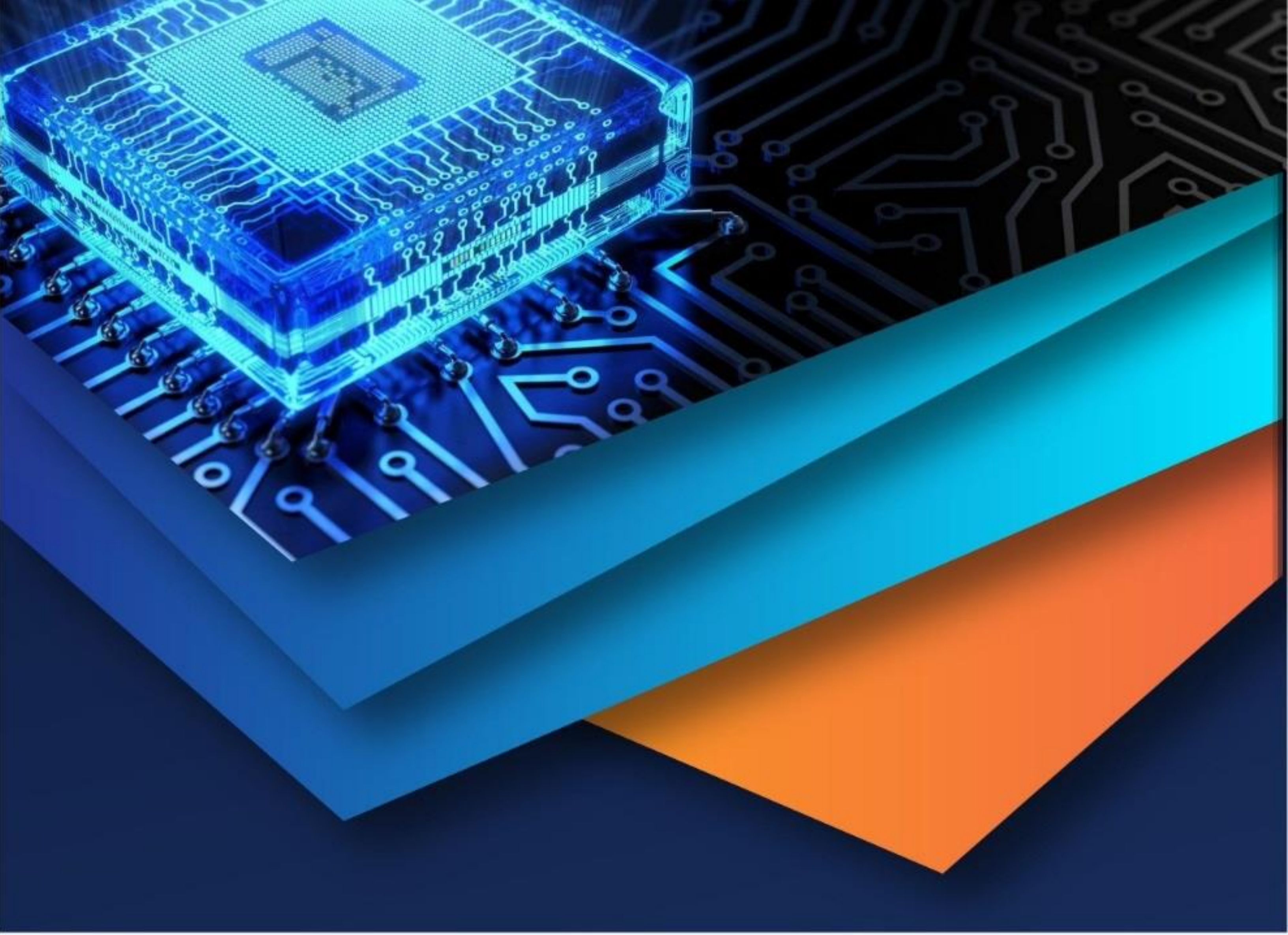
[14] https://docs.konghq.com/gateway/latest/get-started/comprehensive/expose-services/

[15] Wood, T., Cecchet, E., Ramakrishnan, K., Shenoy, P., Van Der Merwe, J., & Venkataramani, A. (2010). Disaster recovery as a cloud service: Economic benefits & deployment challenges. 2nd USENIX Workshop on Hot Topics in Cloud Computing. Boston, MA, 1–7.

[16] Gu, Y., Wang, D., & Liu, C. DR-Cloud: Multi-Cloud based disaster recovery service. Tsinghua Science and Technology (2014).

[17] Sulochana, M., & Dubey, O. Preserving Data Confidentiality Using Multi-Cloud Architecture. Procedia Computer Science (2015).

[18] Prathyakshini, M., & Ankitha, K. Data Storage and Retrieval using Multiple Cloud Interfaces, 5(4), 936–939 (2016).



10.22214/IJRASET

45.98

IMPACT FACTOR:
7.129

IMPACT FACTOR:
7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY