iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** IV **Month of publication:** April 2020

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue IV Apr 2020- Available at www.ijraset.com

Interoperability of Cloud using Single API

Hari Kumar. P[1], Sudharshana. J[2], Sunil Kannah M[3]
[1]*Associate Professor,* [2, 3]*UG Students, Department of Computer Science, Easwari Engineering College, Chennai, Tamilnadu, India*

Abstract: ***In the cloud computing model, users access services according to their requirements. Most of the people use cloud since it has low cost, high speed computing, backup and restore, mobility and unlimited flexible storage capacity. Cloud resources are hosted in large data centres operated by companies such as Amazon, Apple, Google and Microsoft. During cloud application deployment, an application is managed over a single service. Such an approach has several short comings. One side-effect of the lack of interoperability among cloud providers is vendor lock in, which means lack of ability to migrate application components from one cloud provider to another cloud provider. If a user finds some required platforms in new service provider but the user cannot leave the current provider as the resources are present with them. This is known as vendor lock in.***
To solve this issue in achieving interoperability several efforts are under-way. Our project is that a user who creates a unique channel that can be used to gain the services provided by different providers. So that the user can use another vendor's service which is not present in present vendor's cloud. That is user can use multiple clouds having different resources using a single API without depending on their own APIs.
Keywords: Interoperability, agent, heterogeneous cloud environment, cloud-cloud migration, hybrid-cloud, semantic interoperability.

I. INTRODUCTION

Cloud computing means storing and accessing data and programs over the Internet instead of your computer's hard drive. Users store and manipulate resources in a cloud. The users are unable to access the other vendors' cloud with the available resources stored in the present cloud as they get lock in with current vendor. So, the user creates new account in new cloud and store the resources. Users must have flexibility to move on to other services which has additional or better features than the previous one. In our project we proposed that using a single API we can be able to access the resources from two servers, since each cloud is a different server. We can store the resources in different servers and can access the resource with this API. In our project we proposed that using a single API we can be able to transfer the data from one cloud to another using agent and modules. We demonstrate the interoperability using storage service provided by the vendor in this project. In future this can be implemented for all other services provided by the cloud vendors to achieve interoperability.

II. RELATED WORK

In the paper "An approach for developing an interoperability mechanism between cloud providers",Meriem Thabet* and Mahmoud Boufaida gave a solution to interoperability. our main concept is using an adapter agent for interoperability of data between clouds. In this case, the company uses the 'customer relationship management' (CRM)software, 'human resources' (RH) software of the cloud provider A and the dashboard software of the cloud provider B for creating its dashboards for sales tracking (CRM) and monitoring of absenteeism (RH). For doing this, the company must have its data in the provider A. to say in detail, this data must be imported by the cloud provider A and integrated in the cloud provider B.
More importantly, the invocation of providers A and B should be in specific order. It must be managed and provided by the coordinator agent. Initially, the provider B must be invoked since it requires data of the provider A which will be invoked afterwards. Hence, the provider B invokes the coordinator agent through a request including the required data to integrate them and provide the necessary services. After receiving the request by the provider B, the coordinator agent activates the mobile agent of the provider A that migrates within the cloud to complete its task. This agent searches, collects, filters the requested data and followed by sending them to the data that is concerning all the products like: product code, description of products and vendor related information like price, sale date. For creating the second dashboard, the mobile agent searches the required data, which are always sought after by the provider B, like the personnel data: age. And also, our mobile agent generates other agents to facilitating the research of data, to process the requests of clients as soon as possible with greater speed. following that, it sends a direct message to the adapter agent of the provider B containing a document of the required data. Now, our aim is to achieve a solution for syntactic heterogeneity. But, in the future it will be fascinating to take into the semantic heterogeneity of the exchanged data into consideration.

In the paper “Service-Aware Cloud-to-Cloud Migration of Multiple Virtual Machines” by Jargalsaikhan Narantuya, Hannie Zang, and Hyuk Lim, a solution to interoperability is explained. Q-.learning algorithm is used to find an optimal migration policy for each service request. The framework has two main modules: one works at the source which sends VMs to their destination (sender), and the other that works at the destination, receiving inbound VMs (receiver). At the source cloud, initially we obtain the network configuration information of the cloud by using Nova and Neutron APIs. The obtained network configuration information is being transmitted from the sender to the receiver. After transmission of the network configuration information, the sender waits for a response message of the network configuration information from the receiver. The sender, after receiving, the network configuration completion message, the VMs at the source cloud are transmitted to the destination cloud based on the migration strategy. All the VMs from the source are being sent to the destination in the similar way. Following this, the sender transmits a message to the destination in order to signal the receiver that the process of cloud migration is completed. At the destination, the receiver gets ready to receive and configure the inbound Vms. The receiver uses an Inotify event handler which signals the arrival of a new file to receive the Vms. Inotify is a Linux kernel subsystem that acts to extend file systems to notice changes to the file system and report the noted changes to the applications. Initially, the receiver receives the network configuration information from the sender. Next, at the destination, the receiver configures the network based upon the received network configuration information using Neutron APIs, which also includes creating sub-networks and floating Ips. The receiver sends a network configuration completed message to the sender after the completion of network configuration. Then, the sender transmits the VMs to the receiver, and the receiver starts to restart the received VMs with their configurations at the destination cloud. The major issue in this paper is some computing standards can’t be found. In the paper “Towards a Cloud Service Standardization to ensure interoperability in heterogeneous Cloud based environment”, Mazda Elhozmari and Ettalbi gave a solution to cloud interoperability. StoRHm is a protocol adapters which guarantees manual process transitions from SOAP Web services to the REST Web services for enabling existing SOAP clients interact with REST services. The main objective of StoRHm protocol adapter is to permit the existing SOAP clients to make conversations with a REST Web service which appears as same as the original message SOAP Web service. Elements of StoRHm are being highlighted with an overshadow background. A WSDL file, the WSDL schema and a WADL (if needed) file are the wizard inputs. These files are first analyzed and the present information about the user. And then the user matches the SOAP operations with the URIs followed by selecting the HTTP verbs and the response Web services to be used. The output will be the mapping file. The StoRHm integrates in the client side a "SOAP-to-RESTful HTTP protocol adapter". The original SOAP request is being converted to RESTful HTTP and is leaded to the RESTful Web Service. The RESTful Web Service response turns back to the adapter and it is being mapped back to the SOAP so that the SOAP Web service client could easily understand it. And so, the SOAP client is being abstracted from the mappings right from the point of view of the SOAP client. This solution is being used for a particular environment. But it is semi-automatic because the user should undergo a manual process transition from SOAP Web services to REST. REST2SOAP is a framework for integrating SOAP services and RESTful services. REST2SOAP converts the RESTful services into the SOAP services. The document of RESTful services description is being provided by the "mashu service assembler" WADL. The framework converts the RESTful using service description in WADL into a SOAP service in order to create a new composite service so that the "mashup service assembler" can integrate it with other SOAP services. REST2SOAP framework combines RESTful services into SOAP services.

In the paper,”Analysis Of Interoperability In Cloud Computing” Emiroglu Bulent Gursel and Alshiply Tarek gave a solution to interoperability. In this paper, they use two architectures: unified cloud computing architectures and inter cloud architecture. Unified Cloud Computing interface is an interface for customers to meet their needs of services from various cloud other providers. It unifies various cloud api’s so that customers can get services from various cloud providers. The UCI builds an abstract layer which acts as an intermediate layer between the service provider and the client for dealing with any company which wants to join the interoperability environment. The layer first receives the customer's request and then checks for the required settings and data and makes sure that the customer's connection is appropriate with the computing company for the security of the required service. Inter cloud architecture: it works only with the availability of the following components. Governance: It provides standards that is required for each cloud computing provider for making a transaction in the cloud computing environment. For avoiding mismatches and misunderstandings between any companies during the level of service-delivery, it will first determine and provide all the standards to make sure that the sharing process between cloud service providers is implemented in an appropriate manner without any issues. Cloud Computing Catalogue: it includes the standards and rules that has to be available in all the computing companies which want to join cloud environment. Inter Cloud Root: it’s purpose is to host the computing companies that want to join the interoperability environment and checks for the standards that is required from companies and these standards are stored in governance.

After the operation of checking is over, Inter Cloud Root creates a certificate to the company that is been accepted and the company uses this certificate for verification from their customers and Inter Cloud Exchange uses this certificate for checking other computing companies which shares services of this company. Inter Cloud Exchange: it looks for the most exact cloud computing company for providing the required service. Owl-S ontology: It describes the services provided by each cloud computing provider in order to benefit cloud providers who searches for a required service. Here, reduced resource usage was 93.5% solved but the main issue is missing 6.5% which means that some cloud computing models can't work together.

In the paper "Interoperability and Portability for Cloud Computing: A Guide", a solution to interoperability is explained. Here some standards are used like open visualization format, Cloud Data Management Interface, Open Cloud Computing Interface for achieving interoperability. Open Visualization Format is an open-source standard to perform packaging and distribution of software applications for virtual machines. It can contain multiple files in a single directory. The Cloud Data Management Interface is an interface that applications use to retrieve, update and delete elements from a Cloud. Due to this the client will be able to find the capabilities of the cloud storage. The Open Cloud Computing Interface is a group of specifications delivered through the Open Grid Forum(a community for the users, developers and the vendors) for the cloud service providers. Any two systems can overcome interoperability by using a protocol adapter like ESB. An Enterprise Service Bus (ESB) is an architecture. Which has a set of rules and principles in order to integrate a number of applications together through a bus-like infrastructure. The main concept of the ESB architecture is to integrate various applications by putting a communication bus between them. The main issue here resides in the SaaS applications has the biggest interoperability challenge today. There are very less number of standard APIs for SaaS applications. even after switching from one SaaS application to another SaaS application which has a comparable functionality, it involves a change in the interface. An impact occurs on both users of the cloud service for any of the user interfaces.

In the paper named "Towards Formal-based Semantic Interoperability in Multi-Clouds" Stephanie Challita, Fawaz Paraiso and Phillippe Merle gave a solution to cloud interoperability. Here, the issue of vendor lock-in is solved by using a broker such as Rightscale, Kaavo, etc. These brokers offer a unique interface for handling the heterogeneity property of different APIs. They only mask the problem and wont resolve it. Here, a solution, FCLOUDS is used that is a conceptual formal-based framework for performing semantic interoperability in multiclouds. The FCLOUDS framework is completely based on various cloud formal models that could be composed. It includes formal models for OCCI, GCP, AWS (FAWS), etc. FCLOUDS is the first framework which promotes mathematical reasoning and verification for cloud models. After specifying structure and behaviour semantics of each cloud, we can also define transformation functions, for performing semantic interoperability between them. in the future, we the FCLOUDS framework could be completed fully with a wide range of catalogue of formal cloud models which describes mathematical concepts, constraints and capabilities of cloud providers and also verify the specific properties of clouds.

In the paper "Preventing vendor lock-ins via an interoperable multi-cloud deployment approach", a solution to cloud interoperability is explained by Roland Pellegrini, Patrick Rottmann,Georg Strieder.The main goal is to achieve interoperability by addition of an abstraction layer that is represented through a container orchestration solution as well as through a micro-service approach. Major changes that have to be done in terms of architecture, components and software in order to transform a normal generic service ,into a multi-cloud service. The solution stack for transforming the initial web service into a multi-cloud solution includes a micro-service architecture, a container orchestration layer, a NoSQL database layer and modern DevOps tools. Each of the single Cloud in our multi-cloud approach acts independently, but still it could interoperate with the rest of the Clouds through this solution. This solution does not permit any type of vendor lock-in scenarios. The prevention of a vendor lock-in cannot be performed with an individual solutions. Solutions like as Juju, Kubernetes and Docker supports the portability, deployment process and the interoperable management.

In the paper "Hybrid cloud: A solution to cloud interoperability" by Mrs.Shweta M.Barhate and Dr.M.P.Dhore gave a technical solution to cloud interoperability. In this, CloudSim simulator is for analyzation of the implementation of Hybrid Cloud Architecture with respect to interoperability between clouds. Hybrid Cloud involves sharing of resources from combinations of public and also private clouds which therefore gives an easy access and a good utilization of resources to the users. Inter cloud is one of the existing technologies that gives a complete solution to cloud interoperability. Intercloud: The inter cloud is a "cloud of clouds" which means that various individual different clouds are combined into a single mass. Standardization is a better solution to solve the interoperability issue.but in the beginning of cloud computing, interoperability did not arise as a major issue. Microsoft and Amazon acts as a proof to this statement since both of them do not support unified cloud interface(UCI) that was established by the cloud computing interoperability forum(CCIF). Multi cloud approach is not considered for interoperability since it is normally imbibed by big companies like google or amazon clouds who do not like sharing their resources and choose exclusiveness of their respective customers.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue IV Apr 2020- Available at www.ijraset.com

Hybrid Cloud which is other wise known as cloud federation is the union of different cloud service providers which is either from public or private clouds which agree to follow a standardized approach for providing the service of interoperability to the customers while they move from one cloud service provider to another. A CSP has a few resources for the clients. Here, a client request is sent to a particular CSP. If the CSP is not able to respond to the request from the client, then it could affect various clients who are waiting to receive the required service from the cloud service provider which severely affects the quality of service of the client. This means that unavailability of resources may result in loosing the important client of the CSP which is a major problematic case. In order to solve this problem various cloud service providers with mutual consent would get ready for resource sharing. The biggest advantage of a hybrid cloud is resource sharing in cloud environment. Hybrid Cloud or cloud federation shares infrastructure for providing a good quality of service like interoperability, availability and resource sharing and a better resource sharing. The hybrid cloud or cloud federation is of two types: vertical and horizontal federation. Horizontal federation is a collection of resources at a particular service model like IAAS, PAAS and SAAS. In vertical federation, one service model makes use of another service model from other cloud service providers. Hybrid cloud is a combination of public and private clouds which is a boosting solution for cloud interoperability.

In a paper "Software Design Patterns to Develop an Interoperable Cloud Environment" Elena Markosa, Nevena Ackocska, Sasko Ristov, Marjan Gusev and Magdalena Kostoska gave a solution to cloud interoperability. Here, our main objective is to implement an adapter which connects with two important open source cloud frameworks-Eucalyptus and Open Stack and also work with their APIs to permit the creation of a newer instance, authentication of the users, as well as setting up the user privileges for a particular user. Various design patterns are used for the development of an interoperable adapter. The adapter component in this implementation involves the development of a web service which communicates with both the clouds which in turns creates a heterogeneous environment which is also interoperable. Open Stack provides an exclusive and extensive API that unifies numerable services and offers the client with certain methods which can manage them. Eventhough Eucalyptus is initially a private cloud framework, the API offers the implementation of the Amazon Web Services API which is implemented on top of the Eucalyptus. This provides the ability for tools in cloud ecosystem along with the possibility to communicate with the AWS, using the same API as the Eucalyptus. This leads to enable users who are already on the AWS Public Cloud easily migrate to a private cloud which also works in vice versa depending upon the needs of the customers, which increases the performance of the cloud application portability. the future work is to define a prolicy or a standard for an interoperable cloud interface.

III. PROPOSED SYSTEM

In our project, we proposed that using a single API we will be able to access the services from two vendors. We can store the resources in different clouds and can access the resource with this API. Adding to this, we proposed that using a single API we can be able to migrate the data from one cloud to another using agent and modules. We demonstrate the interoperability using storage service provided by the vendor in this project.

The model comprises of the proposed system is a simple layout with a user application interacting with the cloud service providers through a third party trusted agent. User stores the data temporarily in the third party trusted agent for undergoing data migration. In the third party agent, user authentication and cloud authentication processes are undergone.

The proposed system consists of the following modules:

A. File Uploading To Specified Cloud

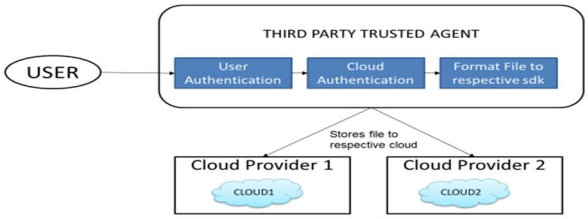


Figure 1 File Uploading

User sends the files to the third party trusted agent. The processes of user authentication, cloud authentication and formatting file to respective sdk takes place there. The third party agent sends the files to the respective clouds of the cloud service providers the user chooses.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue IV Apr 2020- Available at www.ijraset.com

B. Data Migration Using Temporary Storage

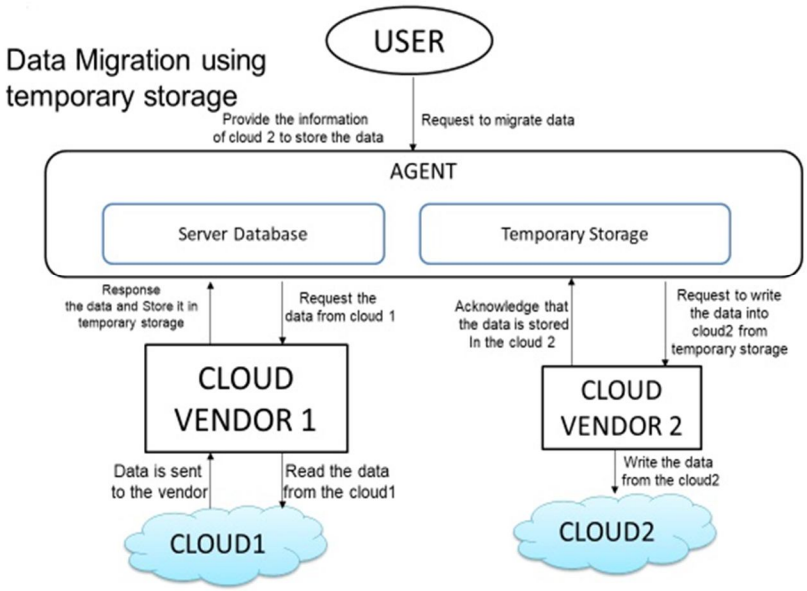


Figure 2: Data Migration Using Temporary Storage

In this module the process of data migration from cloud 1 to cloud 2 using temporary storage will be done. User requests the agent for migrating the data by providing the information about the cloud 2. the agent has two components. They are the server database and the temporary storage. The agent requests the data from cloud 1 by sending a request to the cloud vendor 1. the cloud vendor 1 reads the data from cloud vendor 1 and the data is sent from cloud 1 to cloud vendor 1. the cloud vendor responses by sending the data to the agent and the agent stores it in the temporary storage.the agent then requests to cloud vendor 2 to store the data in cloud 2 from the temporary storage. The cloud vendor 2 stores the data from temporary storage to the cloud 2.the cloud 2 acknowledges the cloud vendor 2 that the data is written into cloud 2. the cloud vendor 2 sends an acknowledgement to the agent that the data is written into cloud 2.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

The files get stored in the cloud which the user chooses. Since there is transparency for the user in using the clouds, the user can view the location of clouds where the files are stored later, whenever in need, in the view files option of the browser. This helps the user flexible in interoperability of clouds.

V. CONCLUSION

The Multi-Cloud architecture provides an environment where the businesses can build powerful and secure cloud environments outside the traditional cloud infrastructure. If multi-cloud usage is maximized, it however leads to tackling the challenges of unique portals, app sprawls, compliance, migration and also security head-on. The main goal of multi-cloud strategy is to use as many clouds as possible according to the requirements in order to find out the limitations of using one cloud from a single cloud provider. Even though toggling between cloud service providers to perform certain tasks could be complicated, cloud service providers are working on to make the toggling between clouds efficient increasingly. If it becomes more efficient the multi-cloud computing will evolve more. In this project we have proposed the migration of the files between multi clouds using single API. The main objective of the project is to save the memory of the cloud storage service by migrating files between multiple clouds using single API. The users can also store and migrate the files in cloud service in which the user wishes to upload. This work describes the overview of Multi-Cloud storage approaches which defines the techniques, challenges and also their corresponding limitations. Many more similar approaches have been proposed but there is a lack of improvement in the technical approach. In real time it is a difficult process for individuals, and the cloud service providers to fulfil their demands.

VI. APPENDIX

File Upload Syntax
void cloudapi(String cloud_name, String file_name, String user_id, String user_name);
This is the function which is called from the application.
In this function the cloud name and user id are provided from which the cloud authentication details are retrieved from the database.

A. Two Ways of Authentication

Different cloud has different authentication types such as

1) User Name and Password

void upload (String file_name, String user_name, String password);
In this function the file name and user-name & password of the cloud is given so that the given file stores in the respective cloud.

2) API Key

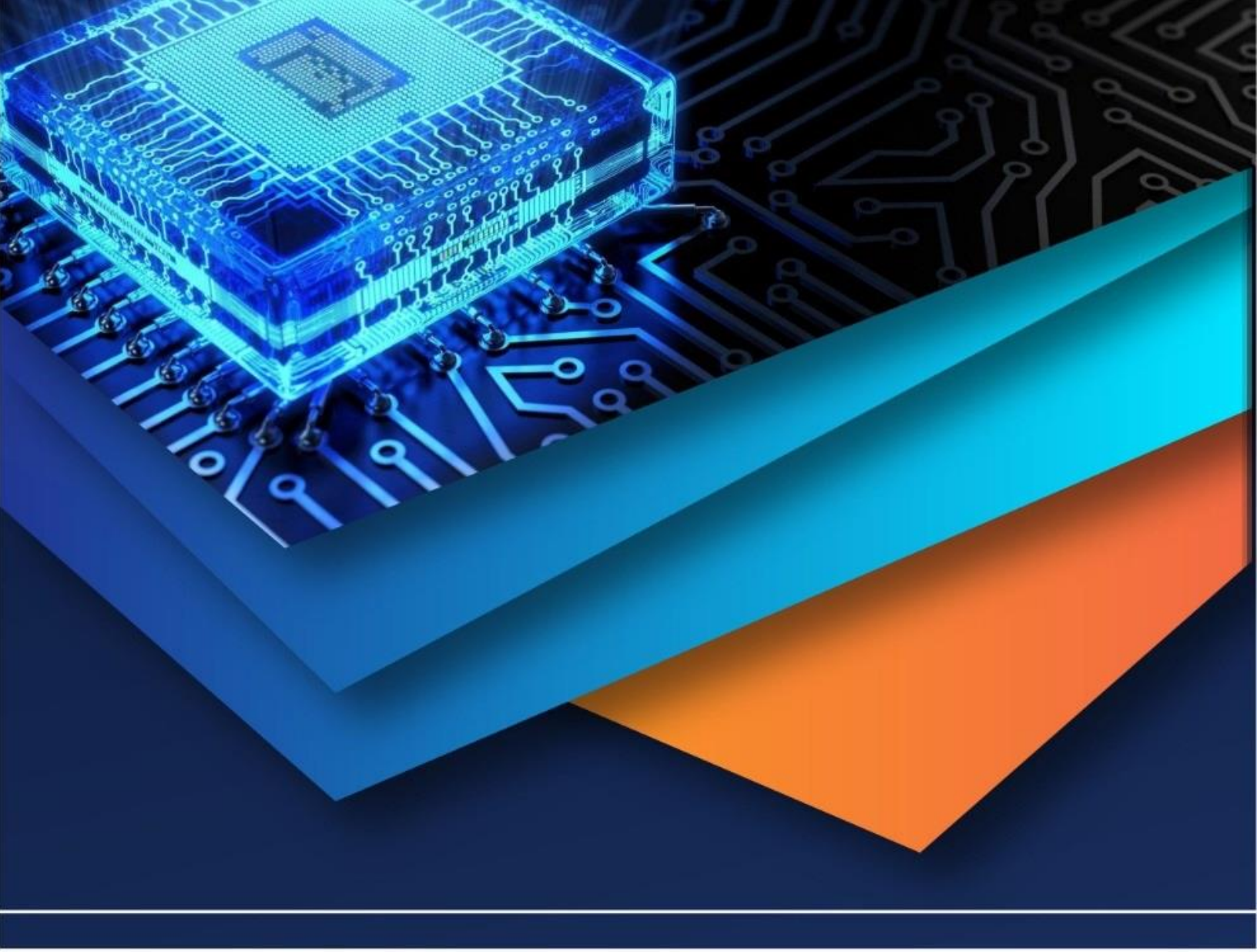
void upload (String API_key, String file_name);
In this function the file name and API key of the cloud is provided so that the given file is uploaded in the cloud.

B. Cloud Migration Syntax

void migrate (String source_cloud, String destination_cloud);
In this function the source and destination cloud are given so that the files in the source cloud is retrieved and store in the temporary storage. The files are upload to the destination cloud. At last the files in the source cloud is deleted.

REFERENCES

[1] Thabet, M., Boufaida, M. and Kordon, F. (2014) 'An approach for developing an interoperability mechanism between cloud providers', *Int. J. Space-Based and Situated Computing*, Vol. 4, No. 2, pp.88–99.
[2] Jargalsaikhan Narantuya, Hannie Zang, and Hyuk Lim 2018 "Service-Aware Cloud-to-Cloud Migration of Multiple Virtual Machines" IEEE Access, Vol no: 6.
[3] Majda Elhozmari and Ahmed Ettalbi 2016 "Towards a Cloud Service Standardization to ensure interoperability in heterogeneous Cloud based environment", IJCSNS International Journal of Computer Science and Network Security, VOL.16 No.7
[4] Emirolu Bulent Gursel, Alshiply Tarek 2019 "Analysis of Interoperability in Cloud Computing", 5th International Conference on Computer and Technology Applications(ICCTA 2019).
[5] Claude Baudoin, Eliezer Deket 2017 "Interoperability and Portability for Cloud Computing: A Guide Vesion 2.0", Cloud Standard Customer Council.
[6] Stephanie Challita, Fawaz Paraiso and Phillippe Merle 2017 "Towards Formal-based Semantic Interoperability in Multi-Clouds", IEEE 10th International Conference on Cloud Computing.
[7] Roland Pellegrini, Patric Rottman and Geog Strieder 2017 "Preventing vendor lock-ins via an interoperable multi-cloud deplo yment approach", 12th International Conference for Internet Technology and Secured Transactions(ICITST).
[8] Mrs.Shweta M.Barhate and Dr.M.P.Dhore 2018 "Hybrid cloud: A solution to cloud interoperability" 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018).
[9] https://docs.pcloud.com/
[10] https://www.drivehq.com/file/DFPublishFile.aspx/FileID1252023466/Keyw1t2o9x5oqoj/MyFTPClient.java
[11] https://cloud.google.com/docs
[12] https://docs.aws.amazon.com
[13] https://docs.microsoft.com/en-us/azure/?product=featured



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY