ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (51-57), Month: October - December 2014, Available at: www.researchpublish.com

Flaw Indulgence in Cloud Computing For Secure Data Transmission

[1]S. Umadevi, [2]M. Geetha

[1] MCA., Research Scholar, Dept of Computer Science, Kongu Arts and Science College, Erode, Tamil Nadu.

[2] MCA., M.Phil. Assistant Professor, Dept of Computer Science, Kongu Arts and Science College, Erode, Tamil Nadu, India

Abstract: **Distributed cloud servers are composed of processes, located on one or more locations that communicate with one another to offer services to front layer applications. A major difficulty computing paradigm has significantly changed the dimension of risks on user's applications, specifically because the failures like server overload, network congestion, hardware faults that manifest in the data centers and data is difficult to collect. The thesis proposes an Efficient Geographic Multicast Protocol (EGMP) algorithm requires fault tolerance abilities so that they can overcome the impact of failures and perform their functions correctly when failures happen. In these systems when they are prone to process failures, two distributed computing models have received significant attention, namely, a fault tolerance mechanisms and replication mechanism. The Efficient Geographic Multicast Protocol (EGMP) also supports secure and robust data storage and retrieval, and user forward his data in the storage servers to another user without retrieving the data back. The EGMP locates the data in distributed clouds based on the geographic locations and based on the counts of server. So the complete data will not be present in the single server. It analyzes and suggests suitable locations for the number of copies of a message dispatched to storage servers and the number of storage servers queried by a key server.**

Keywords: **Cloud computing, fault tolerance as a service, EGMP, storage servers.**

I. INTRODUCTION

Cloud computing is an internet technology that utilizes both central remote servers and internet to manage the data and applications. This technology allows many businesses and users to use the data and application without an installation. Users and businesses can access the information and files at any computer system having an internet connection. Cloud computing provides much more effective computing by centralized memory, processing, storage and bandwidth. In general, a Cloud computing infrastructure is built by interconnecting large-scale virtualized data centers, and computing resources are delivered to the user over the Internet in the form of an on-demand service by using virtual machines (e.g., [1], [2]). While the benefits are immense, this computing paradigm has significantly changed the dimension of risks on user's applications, specifically because the failures (e.g., server overload, network congestion, hardware faults) that manifest in the data centers are outside the scope of the user's organization [3]. Nevertheless, these failures impose high implications on the applications deployed in virtual machines and, as a result, there is an increasing need to address users' reliability and availability concerns. Cloud computing is a term to describe a technology that distributes computer services away from a local client. For companies, it represents a powerful distribution model, because it shifts the investment away from "just in case" network power, to "pay for usage" and thereby maximizing the ability of their users while avoiding idle network processing. Cloud computing is an internet technology that utilizes both central remote servers and internet to manage the data and applications. This technology allows many businesses and users to use the data and application without an installation. The availability of an extensible pool of resources for the user provides an effective alternative to deploy applications with high scalability and processing requirements [23].Users and businesses can access the information and files at any computer system having an internet connection. Cloud computing provides much more

effective computing by centralized memory, processing, storage and bandwidth. The process of cloud computing technology is broken down into three segments. Platform, Applications and Infrastructure are three segments of this technology. Each part serves many functions and provides applications for both individuals and businesses all around the world

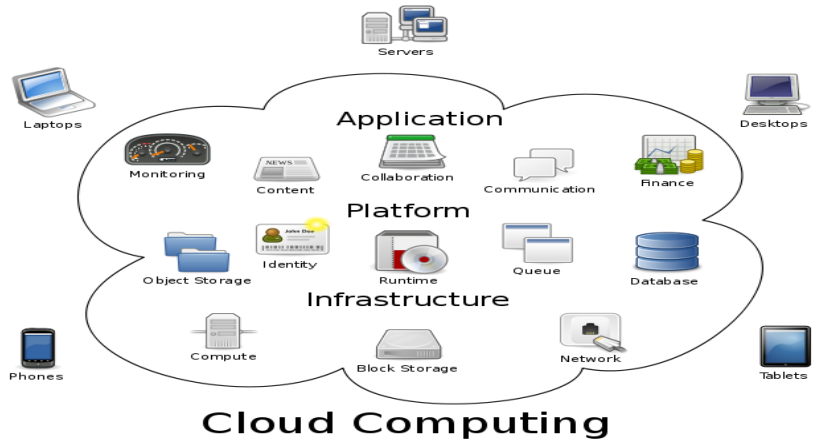

FIG. 1.1 Cloud Computing

EGMP Overview:

EGMP uses a hierarchical structure to implement scalable and efficient group membership management. EGMP can scale to large group size and network size and can efficiently implement multicasting delivery and group membership management. The protocol that allocates the data based on the geographical locations of the nodes. And network-range zone-based bi-directional tree is constructed to achieve a more efficient multicast delivery. The position information is used to guide the hierarchical structure building, multicast tree construction and multicast packet forwarding, which efficiently reduces the overhead for route searching and tree structure maintenance. EGMP does not depend on any specific geographic unicast routing protocol.

EGMP uses a two-tier structure. The whole network is divided into square zones. In each zone, a leader is elected and serves as a representative of its local zone on the upper tier. The leader collects the local zone's group membership information and represents its associated zone to join or leave the multicast sessions as required. As a result, a network-range core-zone-based multicast tree is built on the upper tier to connect the member zones. The source sends the multicast packets directly onto the tree. And then the multicast packets will flow along the multicast tree at the upper tier. When an on tree zone leader receives the packets, it will send the multicast packets to the group members in its local zone.

II. RELATED WORK

Modern day data centers host hundreds of thousands of servers that coordinate tasks in order to deliver highly available cloud computing services [3]. These servers consist of multiple hard disks, memory modules, network cards; processors etc., each of which while carefully engineered are capable of failing. There is no easy way to know when a fault occurred. However, we do track when a repair event takes place and a ticket is filed. The fault tolerance, reliability and resilience in Cloud Computing are of paramount importance to ensure continuous operation and correct results, even in the presence of a given maximum amount of faulty components [21]. Most existing research and implementations focus on architecture-specific solutions to introduce fault tolerance. The Low Latency Fault Tolerance (LLFT) middleware provides fault tolerance for distributed applications deployed within a cloud computing or data center environment, using the leader/follower replication approach [6].

III. SYSTEM DESIGN

A. Existing System

In the existing system, computing paradigm has significantly changed the dimension of risks on user's applications, specifically because the failures like server overload, network congestion, hardware faults that manifest in the data centers. Data is difficult to collect, while ensuring uniform operating conditions. Logs may not be complete or accurate (the embarrassment factor). Even after collection, vendors may not be willing to share data. Impossible to do for one-of-a-kind or very limited systems. Nevertheless, these failures impose high implications on the applications deployed in virtual machines and, there is an increasing as a result. These failures impose high implications on the applications deployed in virtual machines and, as a result, there is an increasing need to address users' reliability and availability concerns. The traditional way of achieving reliable and highly available software is to make use of fault tolerance methods at procurement and development time. This implies that users must understand fault tolerance techniques and tailor their applications by considering environment specific parameters during the design phase. Fault tolerance is a critical issue and comes into picture the moment fault enters system. However, for the applications to be deployed in the Cloud computing environment, it is difficult to design a holistic fault tolerance solution that efficiently combines the failure behavior and system architecture of the application.

B. Drawbacks of Existing System

- High system complexity
- Abstraction layers of Cloud computing that release limited information about the underlying infrastructure to its users.
- Storing the copy of data in term of replica in single node is risky.
- Dynamic faults of a server have not updated.
- Replica allocation is not based on the counts of a cloud server.

C. Proposed System

In propose system, a mechanism Efficient Geographic Multicast Protocol (EGMP) uses a hierarchical structure to implement scalable and efficient group membership management. EGMP does not depend on any specific geographic unicast routing protocol. Several methods are assumed to further make the protocol efficient, for example, introducing the concept of dynamic availability of server for building an optimal tree structure and combining the location service for group members with the hierarchical group membership management. The EGMP allocates the data replica based on the geographic locations. So the complete data will not be present in the single server. Also to enhance the security of the client data, the data's are encrypted and uploaded in the server to provide efficient service to the client as well as forwarding operations over encoded and encrypted messages queried by a key server. EGMP can scale to large group size and network size and can efficiently implement multicasting delivery and group membership management. EGMP uses a hierarchical structure to achieve scalability.

D. Advantages of Proposed System

- Hacking of a single server, the data will not be lost.
- EGMP does not maintain the complete data in a single server.
- Replica allocation process increase finer granularity of archives and resources is obtained.
- It does not require more time for Transactions and it can be done in efficient manner.
- Within a specified time, the server can response for client request.
- Replica allocation is based on the geographic location and number of the servers.
- It also provides forwarding operations over encoded and encrypted messages queried by a key server.

IV. MODULES

A. User Authentication

User Authentication is the process of identity verification you are trying to prove a user is who they say they are. For a user to prove their identity, a user needs to provide some sort of proof of identity that your system understands and trust. The authentication process starts with creating an instance of the Login Context. Various constructors are available; the example uses the Login Context variety. The first parameter is the name (which acts as the index to the login module stack configured in the configuration file), and the second parameter is a callback handler used for passing login information to the Log server. Callback Handler has a handle method which transfers the required information to the Log server. The example uses a very simple handler which saves the username and password in an instance variable, so that it can be passed on during the invocation of the handle method from the Log server. It's also possible to create callbacks that interact with the user to obtain user credentials, and transfer that information to the log server for authentication.

B. Construction of Cloud Data Storage

In the Construction of Cloud Data Storage, the file to be secured by the user into the cloud storage is allocated. The cloud user browses and fetches the information which the user wants to safeguard onto the cloud. The cloud data storage gets the user name that has saved their file into cloud and the file names belonging to particular users. The cloud storage stores this information regarding the user and file. Then the cloud data storage initiates the work of fragmentation to be carried out at each node in order the keep the file secured and available all the time.

C. Replica Allocation and Data Fragmentation

In the replica allocation phase, the replica is allocated based on the availability of node. Initially the user has to configure to allocate the replica in the selected node. The copy of data will be store in the respective node. The user can login the select the file to be uploaded in the server. Once the file has been selected the data is fragmented and encrypted and uploaded in the distributed server. In the File fragmentation the single file has been broken into multiple pieces and stored into the replica.

In data encryption, the data being uploaded onto the cloud storage by the cloud user and it is saved. The AES algorithm is used to encrypt the data loaded by the user with various permutations and rounds specifies in the algorithm. The key is maintained for each encrypted data and it will be used during the time of decryption. Then the server from the cloud can give the encrypted form of the uploaded file. Thus, the saved content is fragmented and gets distributed to various locations onto the cloud storage. Then the cloud storage keeps track of information where the fragmented data gets secured onto the cloud. So even if any one of the fragment is damaged, the content can be retrieved through other fragments, thus providing robustness of data.

D. Fault Tolerance

The task of offering fault tolerance as a service requires the service provider to realize generic fault tolerance mechanisms such that the client's applications deployed in virtual machine instances can transparently obtain fault tolerance properties.

To this aim, define *ft−unit* as the fundamental module that applies a coherent fault tolerance mechanism to a recurrent system failure at the granularity of a VM instance. The notion of ft−unit is based on the observation that the impact of hardware failures on client's applications can be handled by applying fault tolerance mechanisms directly at the virtualization layer than the application itself.

For instance, fault tolerance of the banking service can be increased by replicating the entire VM instance in which its application tier is deployed on multiple physical nodes, and server crashes can be detected using well-known failure detection algorithms such as the heartbeat protocol. Where the primary and backup components are run in VM instances independent of the banking service's application tier. The design stage starts when a client requests the service provider to offer fault tolerance support to its applications. In this stage, the service provider must first analyze the client's requirements, match them with available ft−units, and form a complete fault tolerance solution using appropriate ft−units.

Note that each ft−unit offers a unique set of fault tolerance properties that can be characterized using its functional, operational, and structural attributes.

E. Data Forwarding

In forwarding, the storage details of the uploaded files on to cloud. If the client request for a particular file, then the cloud user generates a key for the particular file which the client requested for. Then the cloud user hands over the key to the requester, so using that key the client can view the information alone. The key code being given to the client is destroyed after its being used by the client once.

F. Data Retrieval

In Data Retrieval, the receiver logins in order the view the file being stored onto the cloud meanwhile the server process must be run, which means the server can be connected with its particular user with the key code. Now the receiver or client has to give the key given by the cloud user in order to view the file which he requested for. Then using that key the client can view the file and use that file appropriately. If once the key has been used by client then it cannot be reused, where the cloud storage deletes that key being used by client in order to provide security for the files on the cloud.

V. EGMP ALGORITHM

```
Procedure Leader Join(me, pkt)
        me:  the leader  itself
        pkt:  the JOIN_REQ message the leader received
        BEGIN
        If(pkt.srcZone==me.zoneID)  then
        /* the join request is from a zNode*/
        /* add the node into the downstream node list of the multicast table */
                AddNodetoMcast Table(pkt.groupID, pkt.nodeID);
        else
        /* the join request is from another zone */
              If(d_me < d_pkt) then
                /* add this zone to the downstream zone list of the multicast table*/
                ddZonetoMcastTable(pkt.groupID, pkt.zoneID);
                else
                        ForwardPacket(pkt);
                        return;
                end if
        end if
if(!LookupMcast|TableforCore(pkt.groupID)) then
        /* there is no core-zone information */
        SendCoreZoneRequest(pkt.groupID);
        else if(!LookupMcastTableforUpstream(pkt.groupID)) then
    /* there is no upstream zone information */
                SendJoinRequest(pkt.groupID);
        else
                SendReply;
        end if
        END
```

VI. SYSTEM IMPLEMENTATION

Implementation is the process that actually yields the lowest-level system elements in the system hierarchy (system breakdown structure). The system elements are made, bought, or reused. Production involves the hardware fabrication processes of forming, removing, joining, and finishing; or the software realization processes of coding and testing; or the operational procedures development processes for operators' roles. If implementation involves a production process, a manufacturing system which uses the established technical and management processes may be required.

The purpose of the implementation process is to design and create (or fabricate) a system element conforming to that element's design properties and/or requirements. The element is constructed employing appropriate technologies and industry practices. This process bridges the system definition processes and the integration process. System Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The existing system was long time process. The proposed system was developed using Java Swing. The existing system caused long time transmission process but the system developed now has a very good user-friendly tool, which has a menu-based interface, graphical interface for the end user. After coding and testing, the project is to be installed on the necessary system. The executable file is to be created and loaded in the system. Again the code is tested in the installed system. Installing the developed code in system in the form of executable file is implementation.

VII. CONCLUSION

The existing system presented an approach toward transparently delivering fault tolerance on the applications deployed in virtual machine instances and realizing generic fault tolerance mechanisms as independent modules. Validating fault tolerance properties of each mechanism, and matching user's requirements with available fault tolerance modules to obtain a comprehensive solution with desired properties. The proposed approach provides efficient geographic multicast protocol (EGMP) algorithm requires fault tolerance abilities so that they can overcome the impact of failures and perform their functions correctly when failures happen.

The efficient geographic multicast protocol (EGMP) algorithm provides the security of the client data, the data's are encrypted and uploaded in the server to provide efficient service to the client and as well as forwarding operations over encoded and encrypted messages. Furthermore, proposed designed that allows suitable locations for the number of copies of a message dispatched to storage servers and the number of storage servers queried by a key server. The components of proposed framework locations allow more flexible adjustment between the number of storage servers and robustness.

VIII. FUTURE ENCHANCEMENT

In future work mainly be driven toward the implementation of the framework to measure the strength of fault tolerance service and forward the cloud data to end user with secure file transfer. Key servers act as access nodes for providing a front-end layer such as a traditional file system interface. When the number users and data upload by user in server increases, then the additional server should be added to the network.

REFERENCES

[1] Amazon Elastic Compute Cloud [Online]. Available: http://aws.amazon.com/ec2/

[2] Eucalyptus Systems [Online]. Available: http://www.eucalyptus.com/

[3] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in Proc. 1st ACM Symp. Cloud Comput., 2010.

[4] C. A. Ardagna, E. Damiani, R. Jhawar, and V. Piuri, "A model-based approach to reliability certification of services,"in Proc. 6th IEEE Int.Conf. Digit. Ecosyst. Technol., Jun. 2012 pp. 1–6.

[5] Heddaya and A. Helal, Reliability, Availability, Dependability and Performability: A User-centered View, Boston University, Boston, MA, 1997.

[6] W. Zhao, P. M. Melliar-Smith, and L. E. Moser, "Fault tolerance middleware for cloud computing," in Proc. 3rd Int. Conf. Cloud Comput., Jul. 2010.

[7] Y. Tamura, K. Sato, S. Kihara, and S. Moriai, "Kemari: Virtual machine synchronization for fault tolerance," in Proc. USENIX Annu. Tech. Conf.(Poster Session), 2008.

[8] S. S. Kulkarni and K. N. Biyani and U. Arumugam, "Composing distributed fault-tolerance components," in Proc. Int. Conf. Dependable Syst. Netw., Supplemental Volume, Workshop Principles Dependable Syst., Jun. 2003, pp. W127–W136.

[9] Y. Mao, C. Liu, J. E. van der Merwe, and M. Fernandez, "Cloud resource orchestration: A data-centric approach," in Proc. 5th Biennial Conf. Innovative Data Syst. Res., 2011.

[10] G. Koslovski, W.-L. Yeow, C. Westphal, T. T. Huu, J. Montagnat, and P. Vicat-Blanc, "Reliability support in virtual infrastructures," in Proc.IEEE 2nd Int. Conf. Cloud Comput. Technol. Sci., Nov. 2010.

[11] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, G.Pelosi, and P. Samarati, "Encryption-based policy enforcement for cloud storage," in Proc. 30th Int. Conf. Distributed Comput. Syst. Workshop,2010.

[12] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu,"Plutus: Scalable Secure File Sharing on Untrusted Storage," Proc.Second USENIX Conf. File and Storage Technologies (FAST), pp. 29-42, 2003.

[13] P. Samarati and S. De Capitani di Vimercati, "Data protection in outsourcing scenarios: Issues and directions," in Proc. 5th ACM Symp.Inform. Comput. Commun. Security, 2010.

[14] C. A. Ardagna, E. Damiani, R. Jhawar, and V. Piuri, "A model-based approach to reliability certification of services," in Proc. 6th IEEE Int. Conf. Digit. Ecosyst. Technol., Jun. 2012.

[15] Hsiao-Ying Lin and Wen-Guey Tzeng "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding" in IEEE transactions on parallel and distributed systems, VOL. 23, NO. 6, JUNE 2012.

[16] Xiaojing Xiang and Xin Wangy Zehua Zhou "An Efficient Geographic Multicast Protocol for Mobile Ad Hoc Networks" State University of New York at Buffalo, Buffalo, NY, USA fxxiang, zzhou5g@cse.buffalo.edu State University of New York at Stony Brook, Stony Brook, NY, USA.

[17] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, G.Pelosi, and P. Samarati, "Encryption-based policy enforcement for cloud storage," in Proc. 30th Int. Conf. Distributed Comput. Syst. Workshop,2010, pp. 42–51.

[18] L. L. Pullum, "Software Fault Tolerance Techniques and Implementation"Norwood, MA: Artech House, 2001.

[19] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in Proc.3rd Symp. Operating Syst. Design Implementation, 1999, pp. 173–186.

[20] B. Cully, G. Lefebvre, D. Meyer, M. Feeley, N. Hutchinson, and A. Warfield, "Remus: High availability via asynchronous virtual machine replication," in Proc. 5th USENIX Symp . Networked Syst. DesignImplementation, 2008, pp. 161–174.

[21] R. Jhawar, V. Piuri, and M. D. Santambrogio, "A comprehensive conceptual system-level approach to fault tolerance in cloud computing," in Proc. IEEE Int. Syst. Conf., Mar. 2012.

[22] R. Guerraoui and M. Yabandeh, "Independent faults in the cloud," in Proc. 4th Int. Workshop Large Scale Distributed Syst, Middleware, no.6. 2010, pp. 12–17.

[23] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia,"Above the clouds: A Berkeley view of cloud computing," EECS Dept., Univ. California, Berkeley, UCB/EECS-2009-28, Feb. 2009.

[24] N. Ayari, D. Barbaron, L. Lefevre, and P. Primet, "Fault tolerance for highly available internet services: Concepts, approaches, and issues,"IEEE Commun. Surveys Tutorials, vol. 10, no. 2, pp. 34–46, Apr.–Jun.2008.

[25] M. Hiltunen and R. Schlichting, "An approach to constructing modular fault-tolerant protocols," in Proc. 12th Symp. Reliable Distributed Syst.,1993, pp. 105–114.

[26] L. A. Barroso and U. Hölzle, "The datacenter as a computer: An introduction to the design of warehouse-scale machines," Synthesis Lectures Comput. Architecture, vol. 4, no. 1, pp. 1–108, 2009.