International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT22828

Reconfigures Protected Protocols for Advanced Cloud Computing

Mohammed Inamur Rahman[1], **Shamimul Qamar** [2] **Abdulilah G.F Saif** [3], **Abdul Azeem** [4]
[1,2,3]Department of Computer Science & Engineering, College of Sciences & Arts, Dhahran Al Janub King Khalid University, Abha, Dharan Al Janub Campus
[4]Research Scholar, Department of Electrical Engineering, Jamia Millia Islamia University, New Delhi, India

ABSTRACT

Article Info
Volume 8, Issue 2
Page Number : 34-43

Publication Issue :
March-April-2022

Article History
Accepted: 05 March 2022
Published: 15 March 2022

Cloud computing is a term which used to depict both a stage and kind of utilization. Cloud computing varies from customary registering ideal models as it is adaptable, can be embodied as a conceptual these elements which gives us diverse level of administrations to the customers, driven by economies of scale and the administrations are progressively configurable. Information put away in outsider stockpiling frameworks like the cloud probably won't be secure since classification and honesty of information are not ensured. In spite of the fact that distributed computing gives savvy stockpiling administrations. Subsequently, numerous associations and clients may not utilize the cloud administrations to store their information in the cloud until the point that specific security ensures are made. In this paper, an answer for the issue of safely putting away the customer's information by keeping up the privacy and trustworthiness of the information inside the cloud is produced. The proposed conventions are created which guarantee that the customer's information is put away just on confided away servers, reproduced just on confided away servers, and certification that the information proprietors and other favoured clients of that information get to the information safely.

Keywords - Cloud Computing, Trusted Storage, and Security.

I. INTRODUCTION

As a stage it supplies, arranges and reconfigures servers, while the servers can be physical machines or virtual machines. Then again, Cloud Computing portrays applications that are stretched out to be available through the web and for this reason expansive server farms and great servers are utilized to have the web applications and web administrations. Decoded information of the customer can't be put away in the cloud on the grounds that the cloud supplier will approach the information and consequently the secrecy of the information will be lost. The encryption and decoding of records is

Copyright: © the author(s), publisher and licensee Technoscience Academy. This is an open-access article distributed under the terms of the Creative Commons Attribution Non-Commercial License, which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited

straightforward to the client and the application. The scrambled key is decoded with the client's private key. This is on the grounds that the encryption keys used to scramble information are put away in the circle and notwithstanding when the encryption keys are encoded, they are scrambled utilizing people in general key for this client of that capacity hub. This client of capacity hub in his cloud is the framework manager who has these greatest benefits on that capacity server. Hence, he could without much of a stretch and getting hold of the encryption / decoding keys for this encoded record framework put away in the circle and in this manner unscramble the customer's information and can be even alter it. Henceforth the secrecy and trustworthiness of the customer's information may be the greatest lost. Loss of information secrecy and respectability is bothersome for a customer. There may two principle security subjects for a customer who, is utilizing this cloud administrations for storing information. To empower a customer to build up trust in this cloud supplier's capacity to safely store his information inside the cloud, we require an answer for accomplish privacy and uprightness of customer's information inside the cloud [1] and [3] and [4]. In this paper, an answer for the issue of safely putting away the customer's information by keeping up the classification and honesty of the information inside the cloud is created. The proposed framework is called 'A Trusted Storage System for Cloud'.

II. Background

Cloud Architecture-

The client cooperates with the front - end boundary as of which he chooses an administration, for instance, to store his records, to get to a report, or to track submission. The client's demand is exchanged to the Structure Running which observes the fitting assets to be allocated, and demands upon the Provisioning Facilities instrument to arrangement the assets to the client. The Provisioning Services component associates the cloud servers and procedures the client's demand. In the wake of preparing the client's demand, the cloud framework screen tracks the use of assets by the client and records it in his profile. Along these lines, the cloud supplier charges the client as per his cloud utilization. These administration errands are mechanized in the distributed computing framework [2].

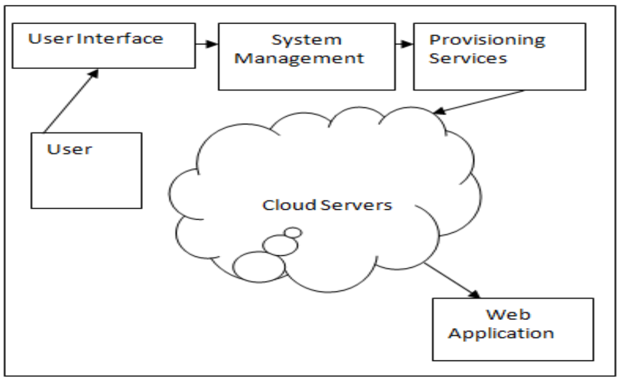

Figure 1 : Architecture of a Cloud Computing System

Cloud Storage-

The client's information is put away in various servers of the cloud as opposed to on a devoted server like in the conventional server farm stockpiling. To the client, it shows up as though his information is put away in a specific area yet actually, the information may move starting with one area then onto the next in time.

The cloud comprises of a few stockpiling servers and a front end server / hub director which deals with the capacity servers inside the cloud. A customer speaks through the cloud concluded the front end server and the other way around. The customer utilizes an online interface to speak with the front_end_server of the cloud. The customer sends his information / documents by means of the web to the cloud for capacity. The front end server in the wake of accepting the customer's records picks a capacity server inside the cloud when sends the customer's documents to it. The Client's information is reproduced inside the distributed storage servers for dependability. A capacity server sends duplicates of its information to other capacity servers for the cloud [1] and [3].

III. Related Works

He and Xu propose a plan to ensure information on a PC stage. They utilize the confided in processing stage created by the confided in figuring gathering (TCG) to build up a protected and dependable model for client verification and information encryption. Their prototypical uses a capacity convention to encode information and utilizations Trusted this Platform Module (T_P_M) to validate distinctive clients for the desired PC. Hosted PC has the confided in figuring stage introduced inside itself, i.e., it has T_P_M introduced in it. Initial, a confided in association is built up between hosted PC and the capacity gadget, i.e., host and the capacity gadget confirm respectively other before any information stockpiling happens inside the capacity gadget and before information is gotten to from the capacity gadget by the host. At that point, security control is given by putting away access control approaches inside the capacity gadget. The entrance control approaches apply to clients, gadgets and applications. Communication control between valuable host and the capacity gadget is given by actualizing session-arranged secure information transmission [4-6].

Mariner, Doorn and Ward depicted a confided in registering stage which expands the equipment established trust assurances of TCG (Trusted Computing Group) advances to the working framework and every one of its applications and enables remote gatherings to check the trust ensures. The TCG characterizes principles for estimating and detailing honesty measurements at the framework boot time. Be that as it may, the TCG does not give data on the dependability of the runtime of the working framework. Mariner et al. give an answer for this issue. They give a methodology that clarifies how a testing gathering can confirm the product heap of an untrusted machine. Trust is set up between two gatherings after shared confirmation happens. The part of the confirmed (framework that should be verified) is instrumented to create estimations of postboot occasions which influence the run-time of the framework. The parts of the product stack that have semantic esteem are estimated. The deliberate parts are piece modules, executables and shared libraries, setup records and other vital information documents that influence trust into the run-time programming stack. The estimations are done when executable substance is stacked into the framework and before it is executed. The verified framework makes and stores an estimation list which is a rundown of hashes of the product stack segments. This rundown is likewise put away in the T_P_M (Trusted Platform Module) for accomplishing respectability. The deliberate rundown and the rundown put away in the T_P_M are sent to the testing party. The testing party looks at both of them and trusts the estimation rundown of the validated framework if both end up being the equivalent. From that point forward, the testing party contrasts the got estimation rundown and its put away expected rundown of estimations. On the off chance that the two end up being the equivalent, the testing party authenticates the framework and consequently the testing party is guaranteed that the framework is running the normal programming stack [7] and [8].

IV. Proposed Techniques

In our research this is consider for privacy and trustworthiness issues of customer's information inside the cloud and give an answer for these topics by offering a structure / framework so-called a Trusted Storage System for Cloud. The framework gives information sanctuary against the cloud supplier, particularly against the framework manager who has the most extreme number of benefits over the capacity hubs in the cloud.

System of Architecture-

The proposed system consists of the following entities:

➔ User/Client
➔ In this_Trusted_Third_Party Node (T_T_P_N)
➔ In this_Front_End_Server (F_E_S)
➔Group _of_Storage_Servers /Nodes

Joining of a storage node in the Trusted Storage System of the Cloud-

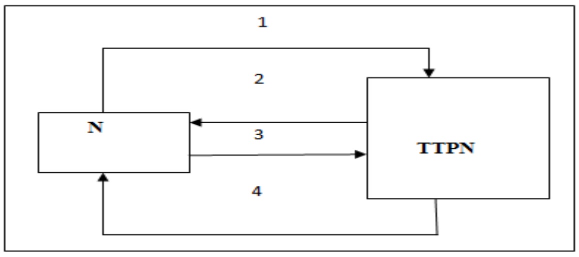

Figure 2: Joining for storage node in the Trusted_Storage_System of the Cloud.

A capacity hub in the cloud must join the Trusted_Storage_System before it can store a customer's information. The T_T_P_N tests if a T_P_M is being introduced on the capacity hub and whenever introduced, it checks for the rightness of the stage of the capacity hub. This procedure is called verification. After fruitful verification of the stage of the capacity hub by the T_T_P_N , the capacity hub is esteemed trusted and can store a customer's information safely.

Notations---
nT_T_P_N : Nonce of the T_T_P_N PCRVN: The P_C_R esteem put away in the T_P_M of hub N (Hash of the bit of the hub and succession of hashes of the product associated with the boot arrangement of the hub bootstrap loader, BIOS).

M_L_N: Measurement rundown of the hub N
pri_(AIK): Private validation character key of the T_P_M of hub N
pub_(AIK): Public verification personality key of the T_P_M of hub N.
C(AIK): Attestation personality key testament of the T_P_M of hub N.
pub(N): Pub_keyof the hub N.
Nid: ID of the hub N.
pub(T_T_P_N): Pub_keyof the T_T_P_N .
pri(T_T_P_N): Pri_keyof the T_T_P_N .

Convention 1
Message_1_one: {"Connect to System", Nid}
Message_2_two: {"Send stage state", nT_T_P_N } pri(T_T_P_N)
Message_3_three: {{PCRVN , nT_T_P_N } pri_(AIK) , M_L_N, pub(N), pub_(AIK), C(AIK)} pub(T_T_P_N)
Message_4_four: {"Joined"}pri(T_T_P_N)

Data Storage in the Trusted Storage System-

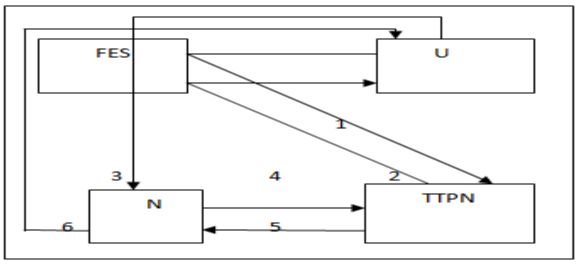

Figure 3 : Data Storage in the Trusted Storage System.

Notations---
U_K: User_Key
Uid: User ID
nU: Nonce of the Hub
nT_T_P_N : Nonce of the T_T_P_N
nN: Nonce of the hub
S_K: One time session key
pub(T_T_P_N): Pub_keyof the T_T_P_N
pri(T_T_P_N): Pri_keyof the T_T_P_N
pub(N): Pub_keyof the hub
pri(N): Pri_keyof the hub
{files}S_K: User's/customer's documents scrambled with the session key
Nid: Node ID

Protocol 2
Message 0: {"Data Send Request", U_K ,Uid, nU } pub(T_T_P_N)
Message 1: {{ S_K, nU, nT_T_P_N }U_K, {H(nU)}pri(T_T_P_N)}

Message 2: {{files}S_K, {H({files} S_K), nT_T_P_N } pub(T_T_P_N)}
Message 3: {Message 3, (H(Message 3))pri(N), {nN, Nid} pub(T_T_P_N)}
Message 4: {{files}S_K, {{nN, U_K , Uid, S_K, H({files}S_K) } pub(N)} pri(T_T_P_N)}
Message 5: {"Information Deposited"}U_K
The customer/client sends an "information send a S_K for" (message 1) to the front_end_server (F_E_S) of the cloud to store his information in the cloud. The front_end_server advances the message to the T_T_P_N. The client creates a symmetric encryption key, U_K, called client key, to utilize it in further correspondences with the cloud. Message 1 incorporates the client key, client ID, Uid, and the client nonce, nU, utilized for avoiding replay assaults. The message is encoded with the general population key of the T_T_P_N so just the T_T_P_N can unscramble and see the message.

Data Integrity
Amid information stockpiling, the E_F_S processes a hash of the client's information and stores it in the T_P_M. At the point when the information is gotten to by the client, the E_F_S registers the hash of the client's information and contrasts it and the hash put away in the T_P_M. On the off chance that both the hashes are equivalent, the E_F_S is guaranteed that the client's information isn't adjusted. On the off chance that the hashes are not the equivalent, an honesty rupture is distinguished.

User's data is transmitted securely and is stored only on trusted storage nodes:
The client encodes his information with the one-time symmetric key, S_K, or, in other words gave to the client by the T_T_P_N which is trusted by the client. The client sends the encoded information to the cloud. A capacity hub inside the cloud can unscramble the client's information, re-scramble it with the E_F_S keys and store the client's information inside it simply in the wake of getting itself validated by the T_T_P_N .

The capacity hub sends a validation demand to the T_T_P_N by marking the demand with its private key. The T_T_P_N verifies the capacity hub on the off chance that it is an individual from the confided away framework by checking its database on the off chance that it has the general population key of the capacity hub. The T_T_P_N does not validate the capacity hub in the accompanying circumstances:

1. In the event that the capacity hub isn't an individual from the confided away framework, i.e., if general society key of the capacity hub isn't found in the database of the T_T_P_N. Or then again

2. On the off chance that general society enter found in the database does not decode the mark of the hub. This is the situation when the capacity hub is endangered/rebooted.

Data Replication-
Data capacity server duplicates its information onto different servers in the cloud to accomplish information dependability. The information put away in a confided away server ought to be imitated onto other confided away servers as it were. That is, the information ought to be reproduced onto capacity servers which are individuals from the confided away framework.

Packing node organized to agree to take data for imitation-
With the end goal to guarantee that the information for replication is exchanged to confided away servers, the T_T_P_N keeps up a rundown of confided away servers that are prepared to acknowledge information for replication. This rundown is designated "prepared hub list". The prepared hub list is a dynamic rundown which continues changing when new hubs are added to it or existing hubs get erased from it. A capacity

server which is prepared to acknowledge information for replication sends a "prepared for replication" message to the T_T_P_N. The T_T_P_N checks if that server is an individual from the confided away framework. In the event that it is a part, it adds that server to the prepared hub list.

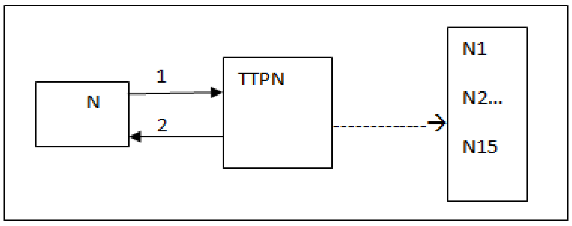

Figure 4: Packing node organized to accept data for imitation.

Notations---

pri(N): Pri_keyof the node N

Nid: ID of the node N

Protocol 3

Message 1: {"Organized for duplication", {"add to prepared hub list"} pri(N), Nid}

Message 2: {"Added/Rejected"}

A capacity server which is prepared to acknowledge information for replication sends "prepared for replication" message which incorporates its ID and "add to prepared hub list" message which is encoded with the Pri_keyof the hub, pri(N), so the T_T_P_N can verify the hub by unscrambling it utilizing people in general key of the hub, pub(N).

At the point when the T_T_P_N gets message 1, checks its database in the event that it has the general population key, pub(N), of that hub put away, i.e., it checks if that hub is a part ("confided in hub") of the Trusted Storage System. In the event that the T_T_P_N finds people in general key, it decodes the "add to prepared hub list" message and hence effectively verifies and adds the capacity hub to its prepared hub list. On the off chance that the TPPN does not locate general society key of that hub or on the off chance that it can't decode the message with the current open key of that hub, it recognizes that the hub isn't an individual from the confided away framework or that the hub has been rebooted and bargained. The T_T_P_N won't add that hub to its prepared hub list. Along these lines, the prepared hub list contains confided away hubs as it were. Because of message 1, the T_T_P_N sends message 2 containing "included/rejected" message. At the point when a hub can't acknowledge any more information for replication, it sends "erase from prepared hub list" message to the T_T_P_N. The T_T_P_N erases that hub from the prepared hub list. Ni.

Storage node sends data for replication-

A storage node N in the cloud which is going to replicate its data on other storage servers should ensure that they are trusted, i.e., they are members of the trusted storage system, before storing data in them. The storage node N requests the T_T_P_N to send a list of ready nodes that accept data for replication. Since, the ready node list consists of storage nodes that are trusted, the storage node N is assured that the data will be replicated securely. After receiving the ready node list from the T_T_P_N , the storage node N sends its data to the storage nodes, (Ni, Ni+1, Ni+2,…), in the received list.

Notations--

Nid: ID of hub N

{(Ni, pub(Ni)), (N(i+1), pub(N(i+1))), (N(i+2), pub(N(i+2))),… … ..)}: Pairs of IDs and open keys of prepared hubs.

Pri(T_T_P_N): Pri_keyof the T_T_P_N

EFS_K: Symmetric key complete and applied by the twisted document preparation of the capacity hub to encode/decrypt the customer's information

{files}EFS_K: Client's documents/information twisted with the encryption record framework key

Uid: User ID

U_K: User_Key.

Convention 4

Message 1: {"Data Replication", Nid, nN}
Message 2: {nN, (Ni, pub(Ni)), (N(i+1), pub(N(i+1))), (N(i+2), pub(N(i+2))),… … ..}pri(T_T_P_N)
Message 3: {{files}EFS_K, {H({files}EFS_K), EFS_K, Uid, U_K} pub(Ni)}/

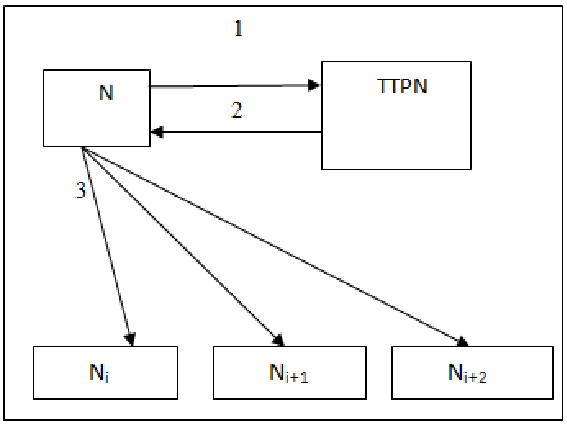

Figure 5: A reliable storing node sends its data for imitation to other reliable storing nodes.

Data Access-

Convention 5 clarifies how the client gets to his information from the cloud safely. The client contacts the front_end_server, F_E_S, to get to his information. The F_E_S sends the client's demand to a capacity hub which has the client's records and the capacity hub safely sends the client documents to the client.

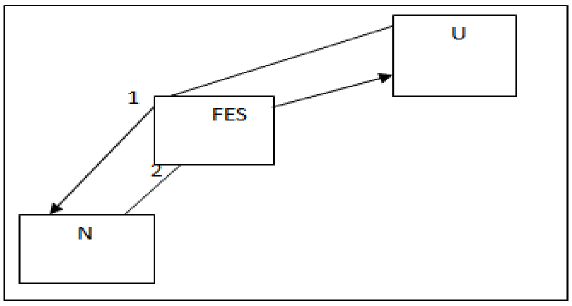

Figure 6: User accesses data.

Notations---

Uid: User ID
nU: nonce of the user
files: Client's files
pub(N): Pub_keyof the storage node N
pri(N): Pri_keyof the storage node N
U_K: User_Key

Protocol 5

Message 1: {"Access Data", {nU, file name/set of file names} U_K, Uid}
Message 2: {{files, nU, pub(N)} U_K , (nU+1) pri(N)}

The client gets to his information put away in the cloud by sending the document name or an arrangement of record names it needs to get to and the client's nonce, nU, and the client ID to the front_end_server (F_E_S) in message 1. The client encodes the nonce and the record names with its key, U_K, to get itself validated by the cloud so the cloud does not send the aS_Ked for documents to another person.

V. Results Analysis

In convention 1, we demonstrate that just capacity hubs with the normal stage state join the confided away framework. A capacity hub readies an estimation list, M_L_N, which means the estimation of the current condition of the framework at boot time. The deliberate condition of the framework is safely put away in the T_P_M of the capacity hub for keeping up the honesty of the deliberate state. The T_T_P_N has the normal condition of the stage put away in it. At the point when a T_T_P_N gets a "join framework" aS_K for from the capacity hub, it sends a "send stage state" demand to the capacity hub. Consequently, the capacity hub sends the estimation rundown, M_L_N, and the stage state, PCRVN, put away in the P_C_R registers of the T_P_M which is marked by the T_P_M. The T_T_P_N confides in the marked substance of the T_P_M in the wake of survey the authentication character key endorsement which is marked by an affirmation specialist. There is a probability that a vindictive/bargained hub can alter the estimation list, M_L_N, to speak to a bona fide stage state. The T_T_P_N analyzes the marked P_C_R esteem, PCRVN, with the estimation list, M_L_N, of the hub. In the event that they are unique, the T_T_P_N does not verify the capacity hub. The authentication falls flat. On the off chance that they are equivalent, it confides in the estimation list, M_L_N, of the hub. The T_T_P_N currently looks at the estimation list, M_L_N, with the normal state put away inside it. On the off chance that they are extraordinary, the verification comes up short. On the off chance that they are equivalent, the T_T_P_N confides in the stage running on the capacity hub and henceforth validates the hub and sends a "joined" message to the hub. In this way, just hubs with a normal/authentic stage are permitted to join the confided away framework.

In convention 2, we demonstrate that a client's information is safely transmitted to the cloud and is put away on confided away hubs just, accordingly, accomplishing classification and respectability.

In convention 3 and 4, We demonstrate that a client's information is imitated just on confided away hubs. The T_T_P_N readies a rundown of capacity hubs which are prepared to acknowledge information for replication. This prepared hub list contains confided in hubs as it were. A capacity hub sends a demand to the T_T_P_N to send the prepared hub rundown to it for reproducing its information. At the point when the T_T_P_N gets the demand, it sends the prepared hub rundown to the capacity hub. Quite possibly an assailant adjusts the prepared hub list arranged by the T_T_P_N or makes his own rundown of pernicious hubs and sends it to the capacity hub N. Since message 2 is marked with the Pri_keyof the T_T_P_N , the rundown can't be altered or changed by any aggressor. Henceforth, the capacity hub is guaranteed that the prepared hub list that it gets is set up by the T_T_P_N as it were. The capacity hub repeats its information onto the hubs in the got rundown. Consequently, information is imitated on confided away hubs just and is secure.

In convention 5 we demonstrate that the convention permits secure information access by a client. A client U aS_Ks for the cloud to get to his information by scrambling the required document name/set of record names with his client key, U_K. At the point when a client U sends a demand to get to his information, the capacity hub scrambles the client's information with the client's vital, U_K, and sends the encoded information to the client. U_K is known just to the client, T_T_P_N and the confided away hubs. So the information is safely gotten to by the client. The mark, (nU+1)pri(N), guarantees the client that the message is set up by the capacity hub as it were. The client nonce in the messages guarantees that the message gotten by the client is new.

VI. Conclusion

The proposed structure gives information security against the framework director who has the most extreme number of benefits on a capacity hub. The framework utilizes a Trusted Platform Module (T_P_M) contribute every capacity hub which gives ensured capacity of encryption/unscrambling keys of customer's information and remote validation of every capacity hub. Every capacity server has a scrambled document framework which encodes the customer's information and the relating keys are put away in the T_P_M. Cryptographic methods are utilized to give secure correspondence between the customer and the cloud. The framework guarantees that the customer's information is put away just on confided away servers and it can't be exchanged by vindictive framework executives to some degenerate hub. The framework engineering and the proposed conventions together shape a Trusted Storage System for Cloud. The framework accomplishes privacy and respectability of the customer's information put away in the cloud.

Acknowledgements:

The authors would like to submit their gratefulness to Deanship of Scientific Research, King Khalid University, Abha, Saudi Arabia for providing administrative, financial and technical support under Grant Number G.R.P 90/1443.

VII. REFERENCES

[1]. Muhammad Aufeef Chauhan and Muhammad Ali Babar, "Migrating Service-Oriented System to Cloud Computing: An Experience Report", 2011 IEEE 4th International Conference on Cloud Computing, pp 404-411.

[2]. Louridas, P., Up in the Air: Moving Your Applications to the Cloud. Software, IEEE, 2010. 27(4): p. 6-11.

[3]. Ali Babar, M., Chauhan, M. A., A Tale of Migration to Cloud Computing for Sharing Experiences and Observations, SECLOUD workshop, Collocated with ICSE 2011, Hawaii, USA.

[4]. Rajkumar Buyyaa, Chee Shin Yeoa, Srikumar Venugopala, James Broberga, and Ivona Brandicc, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility", Future Generation Computer Systems, Volume 25, Issue 6, June2009, Pp 599-616.

[5]. K. Keahey and T. Freeman, "Science Clouds: Early Experiences in Cloud Computing for Scientific Applications," in proceedings of Cloud Computing and Its Applications 2008, Chicago, IL. 2008.

[6]. Junjie Peng, Xuejun Zhang, and Zhou Lei, Bofeng Zhang, Wu Zhang, Qing Li, "Comparison of Several Cloud Computing Platforms", Second International Symposium on Information Science and Engineering, IEEE 2009, pp 23-27.

[7]. Ommeren, E. V., Duivestein, S., deVadoss, J,Reijnen, C. & Gunvaldson, E. Collaboration in the Cloud. Microsoft and Sogeti, Bariet, Ruinen, theNetherlands, 2009.

[8]. Oram, A. "Cloud computing perspectives and questions at the World Economic Forum," WikiContent, 2009. Accessed January 20, 2011

Authors:

Dr. MOHAMMED INAMUR RAHMAN, is an eminent scholar in the field of Computer Science & Engg.. He had done his B.C.A. from SRT University, Nanded, M.Sc. from SRT University, Nanded, and earned his Ph.D in Computer Science & Engg. with highly honorable grade. He was selected as a Head in RG College of Science and Management SRTMUN on 14-07-2006. DR. MOHAMMED INAMUR RAHMAN has a wide teaching experience in various Computer Science and Engineering colleges. He has research interests in Computer Science & Multimedia , Artificial Neural Network, Networking & Security , Artificial Intelligence . He has published several research papers in reputed national/ international Journals and conference. He served as a PROCTOR UNIVERSITY OF IDAHO, 709 S DEAKIN ST, MOSCOW, ID 83844. He is also member in IEEE since 2011. His technical depth and interest resulted in setting up a research lab according to latest technical innovations. Along with this, he has actively participated in various technical courses workshops, seminar etc.

Dr. Shamimul Qamar, is a Professor & an eminent scholar in the field of Computer Science & Engg.. He had done his B.Tech from MMMUT Gorakhpur, M.Tech from AMU, Aligarh and earned his Ph.D in Computer Science & Engg. degree from IIT Roorkee with highly honorable grade. He was selected as a full professor & Head in UPTU Lucknow on 01-12-2006. Prof. Qamar has a wide teaching experience in various Engineering colleges. He has research interests in Computer network & Multimedia , Networking & Security , Artificial Intelligence . He has published several research papers in reputed national/international Journals and conference. He served as Consultant in Jackson state university, USA. He is a reviewer of IEEE/Elsvier/MDPI/EUROSIP/ IJCSIS, USA. He has written some text books and chapters in the field of Electronics & Computer Engineering. He is also a technical programme committee member in various international conference. He is a life time member of international association of Engineers and a life member of Indian Society of technical educational. His technical depth and interest resulted in setting up a research lab according to latest technical innovations. Along with this, he has actively participated in various technical courses workshops, seminar etc. at the IITS.

ABDULILAH G.F SAIF is an eminent scholar in the field of Computer Science & Engg.. He had done his B.Sc. M.Sc. and earned his Ph.D in Computer Science & Engg from Egypt.

ABDUL AZEEM is an eminent scholar in the field of Electrical & Computer Engg.. He had done his B.Tech from HNB Gharwal central University and . M.Tech. in Electrical Engg and pursuing his his Ph.D in Electrical Engg with specialization and applications of Artificial Intelligence.

Cite this article as :

Mohammed Inamur Rahman, Shamimul Qamar, Abdulilah G.F Saif, Abdul Azeem, "Reconfigures Protected Protocols for Advanced Cloud Computing", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 8 Issue 2, pp. 34-43, March-April 2022. Available at doi : https://doi.org/10.32628/CSEIT22828
Journal URL : https://ijsrcseit.com/CSEIT22828