International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-5, March 2019

Metaheuristic Homomorphic FFHE Architecture to Upgrade Encryption in Cloud Communication

Akhilesh Kumar Bhardwaj, Rajiv Mahajan, Surender Kumar

Abstract*: The number of cipher-texts in distributed cloud storage increasing rapidly. An objective function based homomorphic approach is proposed that is reasonable for dynamic cloud storage. The design solves cipher-text class requirements and encryption secret key leakage issue. The goal of this research is to construct and execute an encryption design for open cloud access, based on the metaheuristic objective functional homomorphic viewpoint. Ciphertext classes are taken from 100 to 500. Delegation ratio is 0 to 100 and file uploading range is 0 to 45. The measurements inspected set-up time, encryption time, extract time, verification time, decryption time, and compression ratio over delegation ratio with compression ratio over numbers of uploaded files.

Index Terms: Cloud, Cryptography, Objective function, Homomorphic. FFHE

I. INTRODUCTION

Cloud technology is a course for clients and businesses to get to cutting edge resources over the internet, from basically wherever on the globe that has accessibility. It is normally given as a product service. This computing regularly discards or decreases the prerequisite for on location hardware or software. Outsourcing is particularly ordinary in the client service industry since companies outsource their responsibilities to delegates in various places. Security of data is accomplished by a cryptographic framework. As characterized in RFC 2828 (Thakur & Kumar, 2011), a cryptographic framework is "an arrangement of cryptographic processes with the key managing modules that help utilization of the system in some useful context." In cryptography, the things begin with the decoded information, mentioned as the plaintext. Plaintext is reformed into ciphertext, which in reverse, is decrypted once more into functional plaintext. The encoding or encryption and decoding or decryption follow the kind of cryptography structure being utilized with key arrangements. This procedure is composed of Ct = EnK (Pt) and Pt = DnK (Ct),
Here Pt = Plaintext, Ct = Cipher content, En = Encryption, Dn = Decryption, and K = the Key.
There are five required elements of cryptography:
a. Privacy/secrecy: Ensuring that nobody can read the content with the exception of the recipient.
b. Authentication: The manner in the direction of signifying one's identity.
c. Integrity: Declaring the recipient that the acknowledged content is un-reformed.
d. Non-repudiation: A system to validate that the original node has actually sent this message.
e. Key exchange: The technique by which the keys are shared amongst sender and recipient.
At last, cryptography is most carefully connected with the improvement and making of the scientific calculations used to encode and decode messages, while cryptanalysis is the learning of analyzing and modeling the encryption schemes. Cryptology is the term indicating to the wide investigation of secret writing and covers both cryptography and cryptanalysis (Kessler, 1998). Cryptography is made up of two classes of computations. They are Symmetric-key encryption algorithms (secret key based) and Asymmetric-key encryption algorithms (public key based). The integrity of measurements is guaranteed by hashing functionalities.

Symmetric- Key Encryption

In this sort of encryption, the source (sender) and the destination (receiver) approve a secret shared key. At that point, they utilize this key to encode and decrypt their sent messages. For example, Node X and Y initially drives with the encryption tactic to operate the communicated information. They relate to this undisclosed key that the two will use in this association. After the encryption setup completes, node X begins sending its information coded with the shared key and node Y uses a similar key to decode the encrypted messages.

Asymmetric- Key Encryption

Here two keys are utilized. It is called as Public Key Cryptography (PKC). The clients in PKC tend to utilize two keys namely public key (known to people) and private key (known just to the client). This ability succeeds the symmetric encryption issue of overseeing secret keys. Be that as it may, then again, this remarkable element of public key encryption makes it mathematically more inclined to attacks. Additionally, asymmetric encryption systems are considerably slower than the symmetric methods, as they require more computational handling power (Arbaugh, 2003) (Hardjono, et al., 2005).

Manuscript published on 30 March 2019.
***Correspondence Author(s)
Akhilesh Kumar Bhardwaj, Research Scholar, Department of Computer Science & Engineering, IKG Punjab Technical University, Punjab, India.
Rajiv Mahajan, Department of Computer Science & Engineering, GCET, Gurdaspur, Punjab, India.
Surender Kumar, Department of Computer Science, GTB College, Sangrur, Punjab, India

Cipher: Block and Stream

A cipher is a calculation for performing encryption. Ciphering is the primary arrangement strategies for encryption methods normally established in the form of the input data, they work on. The two cipher kinds are block and stream cipher. In a block cipher, the information is encoded and decoded in the forms of blocks. The stream cipher performs on a stream of information by working on bit by bit framing with two major segments: a mixing function and a keystream generator.

II. RELATED SURVEY

There are various approaches used to actualize security in cloud storage. A portion of the present encryption schemes which were implemented in research work is as per the following. In paper (Natarajan, et al., 2014).have displayed the idea that incorporates encryption of IP address in a network for acquiring privacy in multi-tenant conditions. (Patel, et al., 2015) have suggested a technique to accomplish fine-grained security with the consolidated approach of Kerberos in cloud technology. Kerberos performs secure confirmation of clients and services in light of the indication of a trusted party. (Zhang, et al., 2010) in the paper, have dedicated their work on a few security issues, for example, information security, storage technologies, and information management. (Saravanakumar & Arun, 2014) have performed a thorough learning on security concerns in the cloud. They recommended that a verified client just access the private information, which secures the information from denial of service and check the presence of cloud resources likewise cloud condition repeatedly.

(Agrawal et al., 2004) have projected an order-preserving encryption (OPE) structure for ciphertexts oriented comparable query operations. Unfortunately, if all information is encrypted by means of an OPE, plaintext can be easily derived from cipher-text. Accordingly, it was critical to uplift its security. To address this issue, (Furukawa, et al., 2013) presented a request centered comparable encoding design by using prefix preserving encryption. Although the design upgrades the security of OPE to a specific extent, regardless it produces striking computational and capacity overhead. An efficient confirmation about the correctness of the outsourced information without using a local copy of information files was a major investigation for information security in the cloud. To this end, (Wang, et al., 2010) have projected a strategy known as "privacy-preserving public audit" for information storage privacy in cloud technology. In Wang's protocol, the linear grouping of sample blocks in the cloud server's response was masked with randomness created by a Pseudo-Random Function (PRF). (Liang, et al., 2011) focused on improvement of cipher-text policy attribute-based encryption (CP-ABE). In CP-ABE, if a client has been presented an access to some assets and he is ill-treating it, then no mechanism is there to stop him. On identification, the system supervisory needs to recreate the entire framework. The researchers have utilized binary tree method to refresh data for lessening the computational and communicational cost.

(Liu, et al., 2012) have given another idea of clock-based proxy re-encryption scheme (C-PRE). They have accomplished fine-grained access and scalable client repudiation. In C-PRE, the re-encryption key will be sent in advance of time and it won't have communication delay and workload additionally will be diminished from the data owner side.

(Do, et al., 2011) have determined on classified and fine-grained access in cloud technology with five essential things like flexibility, pay-per-use, self-provisioning of resources, multi-tenancy, and flexibility. (Yu, et al., 2010) have recommended a modification in CP-ABE utilizing proxy re-encryption. They have taken characteristic as index values with occurrences ranking 'positive', 'negative' and 'do not care'.

(C. K. Chu, et al., 2014) proposed a key-aggregation cryptosystem (KAC) Be that as it may, the aggregate-key of KAC isn't precisely constant as the number of encrypted classes is dynamic in cloud storage. This indicates the number of keys addition with the numbers of files in the distributed cloud.

(C. Guo, et al., 2018) recommended key-aggregation authentication cryptosystem (KAAC) creating constantly size keys by supporting flexible delegations rights for cipher content. However, the scheme can be explored more concerning the encryption improvements and data leakage solutions.

III. PROPOSED METAHEURISTIC HOMOMORPHIC ARCHITECTURE

In this segment, initially, we depict the general issues that must be realized. At that point, we introduce the framework of the firefly homomorphic encryption (FFHE) and the relating meanings of the FFHE's functionalities. Further, the points of interest to design are presented. In the key-aggregation authentication cryptosystem, they have utilized key based calculation to encrypt the files. The downside of key based calculation is that information can be decrypted by finding the private key or after the change over the plaintext to cipher-text utilizing key-based algorithm. Additionally, the cipher-text size gets increase more.

To overcome this issue, we have utilized homomorphic encryption structure which is totally key-less based algorithmic approach implies there is no key and it is protected to decrypt by unauthorized clients in the network. Finally, the size of cipher-text is not as much as the key based calculation.

The FFHE Algorithmic Steps

Step 1: The process starts.

Step 2: Check all IPs from its information base every 45 minutes to dynamically interface with all file servers and in addition with trusted party server (TPS) having database of all authenticated IP addresses.

Step 3: Whenever any client will first login from its machine, it will enter its email and secret key or password. Our program will get MAC address on click of login button.

Step 4: Then email and secret key alongside MAC address is initially encrypted utilizing AES 16 round calculation by TPS i.e. all strings get concatenated and then passed to AES. AES algo returns one cipher-text message.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Step 5: The message is now passed to main server with login keyword. Main server (MS) receives the keyword with cipher-text and then passes the message to TPS.
Step 6: TPS receives the cipher-text message and decrypt it. Further, compares in its database. If it is true, then it returns status TRUE to the main server (MS).
Step 7: MS messages back to its client with status TRUE.
Step 8: After successful login, the client can select file for uploading.
Step 9: This file is encrypted utilizing AES in FIREFLY objective function domain and processed to fully homorphic scheme with the developed equation
Ck = ENCpk (K) = 2 (R + subset – sum (Xi's)) + b mod N = Kp + 2Ri + K and sent to the MS.
Step 10: MS gets file. It initially processes value, if found genuine, then Compression takes place. Likewise, verification procedure of each running file server and database refreshing takes place simultaneously.
Step 11: For decryption, the value is computed on client's machine and sent to the MS. On the off chance that value matches, at that point file will be decrypted else not.
Step 12: The process ends.

FFHE Configuration

Table 1. Symbols Representation

Symbols	Definitions
OBJ	Firefly objective function
Array	An array of the 64-bit block
IP_F	The initial population of firefly
D	Length of array
I	Outer loop
K	Variable value
R	Constant value
ENCpk	Encryption function for plain text
Ri	Lower of the length of the array
Kp	Length of all arrays
Ck	Ciphertext
Xi	Variable containing ciphertext file and a string variable
STR	String variable
DECs	Decryption function

IV. THE ENCRYPTION PROCESS ALGORITHM

Begin
(a) In the first step of firefly algorithm, using the objective function 'OBJ', we divide the whole data in 64 bit and store in different array variables (array1, array2, array3……arrayN)
(b) Generate initial population of fireflies IP_F
D= size of class firefly bean
I=0;
While (I<D)
For (firefly bean class)
K = fireflies [j];
Ck = ENCpk (K) = 2 (R + subset – sum (Xi's)) + b mod N = Kp + 2Ri + K……(1)
(Fully homomorphic encryption scheme)
End for
End while
STR= "upload files"
(c) Rank the fireflies and find the current best
Kh = (Ck + STR) …… (2)
Now, the key packet is sent to the main server and then the main server extracts the header and read string based it the server which is running least process and update the "Ck" file on to there
(d) Now when delegate (user) requests for downloading file, it shares their session key with server which is verified by public cloud via trusted third party server
(e) If matches
Then file get decrypt first as below
DECs (Ck) = (Ck mod p) mod 2 …(3)
End if
Else
Reject session key.

V. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, the performance measurements of FFHE are evaluated first and then compared with the key-aggregation authentication based approach. The tests were performed in real time environment using JDK 1.8, Net Beans 8.1, My SQL 5.5, Java Swing and Network Programming with RAM 2 GB and HDD 50 GB.

Table 2. Results for Set-up Time (in milliseconds)

Max. no. of ciphertext classes(n)	KAAC	FFHE
100	1115	930
200	2119	1850
300	3061	2947
400	3931	3570
500	4971	4679

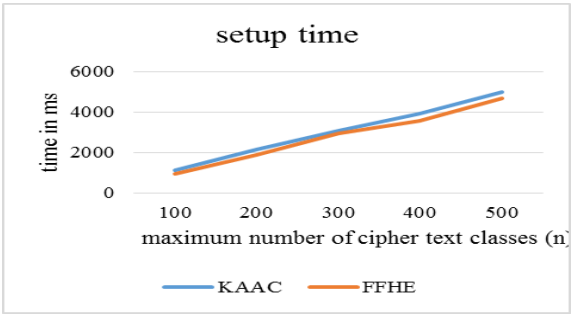

Figure 1. Comparison of KAAC and Proposed FFHE for Set-up Time

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Table 3. Results for Encryption Time (in milliseconds)

Max. no. of ciphertext classes(n)	KAAC	FFHE
100	115	107
200	115	109
300	115	109
400	115	111
500	115	111

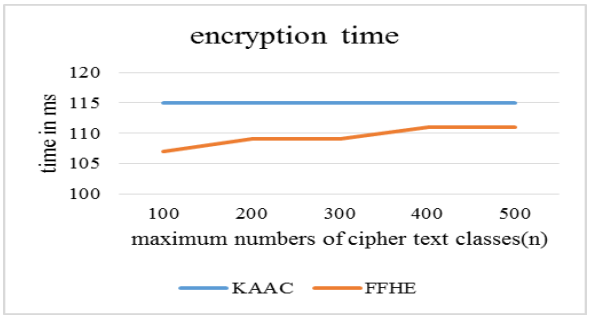

Figure 2. Comparison of KAAC and Proposed FFHE for Encryption Time

Table 4. Results for Extract Time (in milliseconds)

Max. no. of ciphertext classes(n)	KAAC	FFHE
100	3363	3000
200	7069	6566
300	10251	9250
400	13321	11560
500	16477	14478

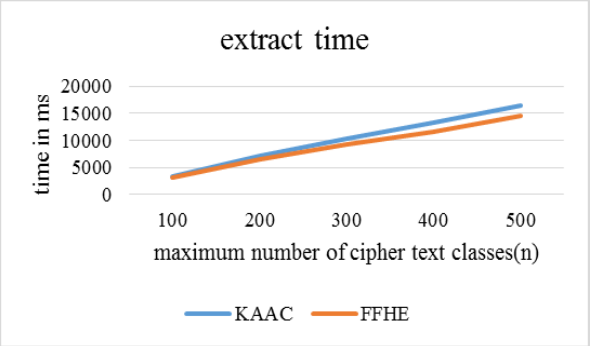

Figure 3. Comparison of KAAC and Proposed FFHE for Extract Time

Table 5. Results for Verification Time (in milliseconds)

Max. no. of ciphertext classes(n)	KAAC	FFHE
100	75	55
200	75	58
300	75	60
400	75	62
500	75	66

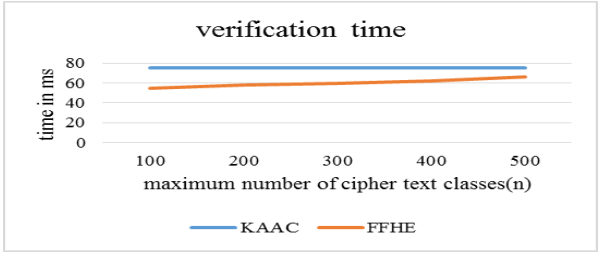

Figure 4. Comparison of KAAC and Proposed FFHE for Verification Time

Table 6. Results for Decryption Time (in milliseconds)

Max. no. of ciphertext classes(n)	KAAC	FFHE
100	76	65
200	76	65
300	76	68
400	76	70
500	76	70

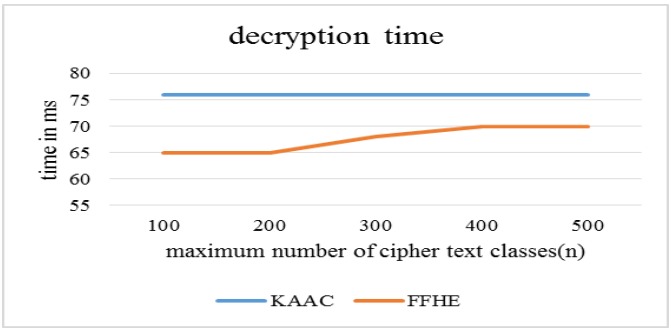

Figure 5. Comparison of KAAC and Proposed FFHE for Decryption Time

Table 7: Comparison of Compression Ratio for Delegation Ratio

Delegation Ratio	KAAC	FFHE
0	1.0	0.78
10	1.0	0.78
20	1.0	0.78
30	1.0	0.78
40	1.0	0.78
50	1.0	0.78
60	1.0	0.78
70	1.0	0.78
80	1.0	0.78
90	1.0	0.78
100	1.0	0.78

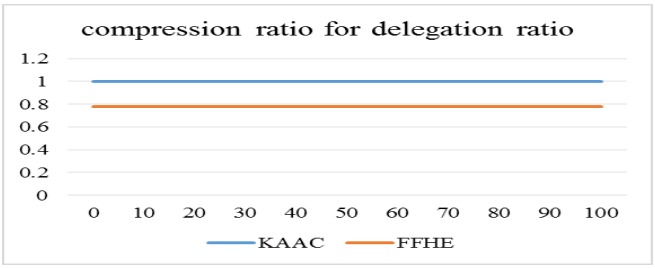

Figure 6. Comparison of KAAC and Proposed FFHE for Compression Ratio on Delegation Ratio

Table 8. Comparison of Compression Ratio for Uploaded Files

No. of Uploaded Files	KAAC	FFHE
0	1.0	1.0
5	0.25	0.40
10	0.10	0.15
15	0.08	0.12
20	0.06	0.10
25	0.05	0.07

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

30	0.05	0.07
35	0.05	0.07
40	0.05	0.07
45	0.05	0.07

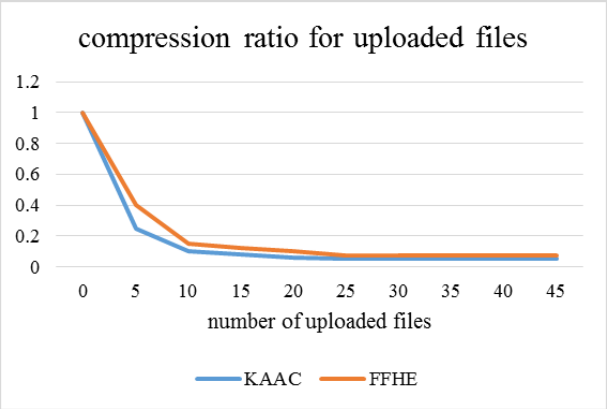

Figure 7. Comparison of KAAC and Proposed FFHE for Compression Ratio on Numbers of Uploaded Files

From the outcomes obtained, the encryption execution is improved essentially showing the sign of improvements than existing on selected metrics.

For cipher-text classes 100, 200, 300, 400, and 500, set-up time was evaluated with the existing scheme. It is improved as 16.59%, 12.69%, 3.72%, 9.18%, and 5.87% respectively. (Table 2 and Figure 1). The encryption time was analyzed with the existing scheme. It is improved as 6.96%, 5.22%, 5.22%, 3.48%, and 3.48% respectively. (Table 3 and Figure 2). The extracted time was evaluated with the existing scheme. It is improved as 10.79%, 7.21%, 9.76%, 13.22%, and 12.13% respectively. (Table 4 and Figure 3). The verification time was analyzed with the existing scheme. It is improved as 26.67%, 22.67%, 20%, 17.33%, and 12% respectively. (Table 5 and Figure 4). The decryption time was evaluated with the existing scheme. It is improved as 14.47%, 14.47%, 10.53%, 8%, and 8% respectively. (Table 6 and Figure 5). For the delegation ratio 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100, the compression ratio was analyzed with the existing scheme. It is improved at an average of 22% respectively. (Table 7 and Figure 6). For the uploaded numbers of files 5, 10, 15, 20, 25, 30, 35, 40, and 45, the compression ratio was evaluated with the existing scheme. It is improved as 60%, 50%, 50%, 66.67%, 40%, 40%, 40%, 40%, and 40% respectively. (Table 8 and Figure 7).

VI. CONCLUSIONS AND FUTURE SCOPE

This key-less FFHE architecture strengthens secure, operative, and flexible information sharing in the distributed storage network. Set-up time is improved to 16.59%. Encryption time is controlled to 6.96%. Extract time is secured up to 13.22%. Verification time controls to 26.67%. Decryption time is decreased to 14.47%. Compression ratio is supported up to 22% against delegation ratio and 66.67% against numbers of uploaded files. Hence this approach can be a superior option for driving a secure and prompt encryption classification on the cloud. The scheme can be additionally enriched to synchronize with multi-dimensional cloud services advancements with time-specific requirements in future.

ACKNOWLEDGMENT

The authors would like to thank reviewers for their helpful comments and for their valuable recommendations towards the enhancement of the paper.

REFERENCES

1. Agrawal, R., Kiernan, J., Srikant, R., & Xu, Y. (2004, June). Order preserving encryption for numeric data. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data* (pp. 563-574). ACM.
2. Arbaugh, W. A. (2003). *Real 802.11 security: Wi-Fi protected access and 802.11 i*. Addison-Wesley Longman Publishing Co., Inc.
3. Chu, C. K., Chow, S. S., Tzeng, W. G., Zhou, J., & Deng, R. H. (2014). Key-aggregate cryptosystem for scalable data sharing in cloud storage. *IEEE transactions on parallel and distributed systems*, *25*(2), 468-477.
4. Do, J. M., Song, Y. J., & Park, N. (2011, May). Attribute based proxy re-encryption for data confidentiality in cloud computing environments. In *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on* (pp. 248-251). IEEE.
5. Furukawa, J. (2013, September). Request-based comparable encryption. In *European Symposium on Research in Computer Security* (pp. 129-146). Springer, Berlin, Heidelberg.
6. Guo, C., Luo, N., Bhuiyan, M. Z. A., Jie, Y., Chen, Y., Feng, B., & Alam, M. (2018). Key-aggregate authentication cryptosystem for data sharing in dynamic cloud storage. *Future Generation Computer Systems*, *84*, 190-199.
7. Hardjono, T., & Dondeti, L. R. (2005). Security in Wireless LANS and MANS (Artech House Computer Security). *Artech House Inc*.
8. Kessler, G. C. (1998). An overview of cryptography. *Published by Auerbach*, *22*.
9. Liu, Q., Wang, G., & Wu, J. (2012, September). Clock-based proxy re-encryption scheme in unreliable clouds. In *Parallel Processing Workshops (ICPPW), 2012 41st International Conference on* (pp. 304-305). IEEE.
10. Lu, R., Liang, X., & Lin, X. (2011). Ciphertext policy attribute based encryption with efficient revocation. *Waterloo, ON, Canada: BBCR, University of Waterloo*.
11. Natarajan, S., & Wolf, T. (2014, December). Network-level privacy for hosted cloud services. In *Network of the Future (NOF), 2014 International Conference and Workshop on* (pp. 1-8). IEEE.
12. Patel, S. C., Singh, R. S., & Jaiswal, S. (2015, February). Secure and privacy enhanced authentication framework for cloud computing. In *Electronics and Communication Systems (ICECS), 2015 2nd International Conference on* (pp. 1631-1634). IEEE.
13. Saravanakumar, C., & Arun, C. (2014, November). Survey on interoperability, security, trust, privacy standardization of cloud computing. In *Contemporary Computing and Informatics (IC3I), 2014 International Conference on* (pp. 977-982). IEEE.
14. Thakur, J., & Kumar, N. (2011). DES, AES and Blowfish: Symmetric key cryptography algorithms simulation based performance analysis. *International journal of emerging technology and advanced engineering*, *1*(2), 6-12.
15. Wang, C., Wang, Q., Ren, K., & Lou, W. (2010, March). Privacy-preserving public auditing for data storage security in cloud computing. In *Infocom, 2010 proceedings ieee* (pp. 1-9). Ieee.
16. Yu, S., Wang, C., Ren, K., & Lou, W. (2010, April). Attribute based data sharing with attribute revocation. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security* (pp. 261-270).
17. Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*, *1*(1), 7-18.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

AUTHORS PROFILE

[1] Akhilesh Kumar Bhardwaj got his B.Tech Degree in Computer Science and Engineering from Kurukshetra University and M.Tech degree in Computer Science from Rajasthan Technical University. At present, he is a Ph.D. research scholar in the Department of Computer Engineering at IKG Punjab Technical University, Punjab.

[2] Rajiv Mahajan got his Ph.D. degree in Computer Engineering with Specialization in Wireless Communication and Security. His exploration zone is security in the ad-hoc system. As of now, he is currently working as Director-Principal at GCET Gurdaspur, Punjab, India. He has 15 Years of Teaching and Research Experience. He has supervised more than 20 Ph.D. and M.Tech. Scholars. He has conducted different sessions at National and International level Conferences. Rajiv Mahajan has published more than 130 research papers in reputed Journals and Conferences. He is board member and reviewer for different International Journals.

[3] Surender Kumar has done an M.Tech degree in Computer Science and Engineering from CDLU and M.Phil. in Computer Science from Alagappa University. He has received Ph.D. in Computer Science and Application from Kurukshetra University. He has more than 10 years teaching experience. He is currently working as an Assistant Professor, Department of Computer Science, GTB College, Bhawanigarh (Sangrur), Punjab, India. He has published over 60 publications in reputed Journals and Conferences. His research interests include Fault Tolerance in Mobile Distributed Systems, Adhoc N/W, System Security, and Cryptography.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)