International Journal of Computer Applications (0975 – 8887)
Volume 94 – No.14, May 2014

Identity based Distributed Data Storage using Digital Signature and Diffie Hellmann Key Exchange Mechanism

R. Allvijay
PG scholar
Sri Venkateswara College of Engineering,
Pennalur, India

V.M.Sivagami
Associate Professor/IT
Sri Venkateswara College of Engineering,
Pennalur, India

ABSTRACT

The widespread use of Internet has increased the demand for computing. Grid computing is a large scale heterogeneous distributed system which helps in sharing of resources such as data storage, computational power, network dynamically. Grid security is an important component of computational grid infrastructure Grid Security solutions addresses critical requirements and are constructed from building blocks comprising cryptographic algorithms and protocols. Choosing the most appropriate algorithm and protocols require an evaluation of performance of various cryptographic technique. Since Grid computing rest on networks, Security issues like privacy, Data Integrity, Authentication and Confidentiality is encountered. Hence, the combination of different mechanisms like encryption, decryption and digital signatures are used to overcome those issues. On these similar terms, a combination of authentication technique, key exchange algorithm blended with an encryption algorithm is employed. RSA Digital Signature algorithm is used to make use of Digital Signature, Diffie- Hellmann Key exchange mechanism for exchanging keys and Data Encryption Standard [DES] for Confidentiality. I propose a three way mechanism for improving the security features of the system. As a part of the work, a performance evaluation of various cryptographic algorithms is made and the results are used for increasing the security features.

General Terms

Grid computing, Security, Algorithm

Keywords

Encryption, Decryption, Digital Signature.

1. INTRODUCTION

The basic principle of data encryption is to transform Plaintext data by some algorithms to some unreadable codes, That is, "cipher text". The cipher text can display the explicit Content only after input the corresponding keys. In this way, the protected data cannot be stolen and distorted [2].

Security approaches has been developed to provide efficient solutions to sophisticated problems such as scientific engineering, business, etc.; this paper describes about different stages of security issues[5]. The basic principle of data encryption is to transform plaintext data by some algorithms to some unreadable codes, that is, "cipher text". The cipher text can display the explicit content only after input the corresponding keys. In this way, the protected data cannot be stolen and distorted [2].

Symmetric key cryptosystem is a traditional password system, and its representative is DES algorithm. The encryption and decryption are using the same keys. For a network with n users, n (n-1) / 2 keys are required. When the user group is not large, the symmetric encryption system is very effective. But for large networks, large amounts of users and widely distributions make the distribution and preservation of keys become the biggest constrains of security problems [3]. In addition, the symmetric encryption system can only be used for encryption and decryption to provide protection of data confidentiality, but it cannot be used for digital signatures. Symmetric algorithm for encryption and decryption functions are shown as follows, respectively:

$$EK(M) = C, DK(C) = M \quad (1)$$

Where M is plaintext, C is cipher text, K is the key, EK is the encryption function, and DK is the decryption function. Non-symmetric key cryptography is also known as public Key cryptosystem, whose representative algorithm is RSA, in, this kind of system, encryption and decryption are usually done using two different keys, known as the "public key" and "Private Key "[4]. Two keys must be used as partnership, or it Cannot decrypt the cipher text. Public key is used to encrypt the plaintext but cannot decrypt cipher text, and private key can decrypt the cipher text but cannot process encryption. The Receiver announce its own public key, anyone in the network Can take advantage of this public key to send encrypted data to the receiver. But the receiver’s own "private key" is not publicly available, and only the private key can decrypt the cipher text. As a result of such asymmetric encryption system, there is no existence of key distribution and preservation problems in symmetric encryption system. For the network with n users, only 2n keys are needed. Asymmetric algorithms for encryption and decryption functions can be expressed as:

$$M = DKB2(C) = DKB2(EKB1(M))$$

The rest of the paper is organized as follows. Section 2 provides an overview on previous researches in security mechanisms. Section 3 describes the system design and implementation details of our grid security mechanism respectively. Section 4 discusses the experimental results and section 5 concludes the paper.

2. RELATED RESEARCH

Secure distributed data storage can shift the burden of maintaining a large number of files from the owner to proxy servers. Proxy servers can convert encrypted files for the owner to encrypted files for the receiver *without* the necessity of knowing the content of the original files. In practice, the original files will be removed by the owner for the sake of space efficiency. Hence, the issues on confidentiality and integrity of the outsourced data must be addressed carefully. In this paper, we propose two identity-based secure distributed data storage (IBSDDS) schemes. Our schemes can capture the following properties: (1) The file owner can decide the access permission independently without the help of the private key generator (PKG); (2) For one query, a receiver can only access one file, instead of all files of the owner; (3) Our schemes are secure against the collusion attacks, namely even if the receiver can compromise the proxy servers, he cannot obtain the owner's secret key. Although the first scheme is only secure against the chosen plaintext attacks (CPA), the second scheme is secure against the chosen cipher text attacks (CCA). To the best of our knowledge, it is the *first* IBSDDS schemes where access permissions is made by the owner for an exact file and collusion attacks can be protected in the standard model [1]. Cluster-based storage is a group of nodes that are linked together where each node is assumed as a server. Each node can work independently without affecting the other nodes, and the same file is present in each node. RSA, ElGamal, and ARIA are well-known encryption techniques, which are used to reduce the server load. Public and private keys are generated by the administrator dynamically and by using public keys. Files are encrypted after, and the same files are uploaded into the three servers. Users can select any files that are present in the database server. After selecting the appropriate file, the system dynamically calculates the minimum response time and starts downloading the file from the server, which has a minimum response time. Comparisons are also done between these encryption techniques in order to show the best encryption technique. This paper investigates compression of data encrypted with block ciphers, such as the Advanced Encryption Standard [6]. It is shown that such data can be feasibly compressed without knowledge of the secret key. Block ciphers operating in various chaining modes are considered and it is shown how compression can be achieved without compromising security of the encryption scheme. Further, it is shown that there exists a fundamental limitation to the practical compressibility of block ciphers when no chaining is used between blocks. Some performance results for practical code constructions used to compress binary sources are presented.

3. BASELINE APPROACHES

In this paper, we describe and simulate other well-known cryptographic algorithms such as: Data Encryption Standard Algorithm (DES), Rivest Shamir Adleman Algorithm (RSA), Advance Encryption Standard Algorithm (AES), and RC4 Algorithm as a baseline to compare the performance of our newly developed encryption algorithm and analyse the results. The performance of Grid security algorithms are based on the execution time to provide the QoS in Grid Environment.

3.1 Advance Encryption Standard (AES)

AES is a Symmetric key algorithm with block length of 128 bits with key lengths of 128,192 and 256 bits. It takes128 bit as 16 byte key expands in to array of 4- 4- 32 bit words. A simple substitution of each byte uses one S-box of 16x16 bytes containing a permutation of all 256 8-bit values. Each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits).S-box constructed using defined transformation of values in GF(256).S-box constructed using a simple math formula using a non-linear Function.

3.2 Data Encryption Standard

DES works on 64bit data at a time. Each 64 bit data is iterated from 1 to 16 times. Each iteration 48 bit subset of 56 bit key is fed in to the encryption block.Fractioning the text in to 64 bit (8 Octet) blocks. Initial permutation of blocks. Breakdown of the blocks in to two parts: L (left) and R (right).Permutation and substitution steps repeated 16 times (called rounds).Re-joining of the left and right paths and inverse initial permutation.

3.3 Rivest Shamir Adleman Algorithm (RSA)

RSA: Public Key Encryption Algorithm with two keys i.e. Public key and Private Key. Public Key-It is used to encrypt messages and verify signatures. Private Key-It is used to decrypt messages and sign signature and those who encrypt messages or verify signatures cannot decrypt messages or create signatures. Public key cryptography is classified in to three categories namely encryption/decryption (provide secrecy), Digital Signatures (provide authentication) and key exchange (of session keys).

3.4 RC4 Algorithm

Many stream ciphers are based on linear feedback shift registers (LFSRs), which, while efficient in hardware, are less so in software. The design of RC4 avoids the use of LFSRs, and is ideal for software implementation, as it requires only byte manipulations. It uses 256 bytes of memory for the state array, S[0] through S[255], k bytes of memory for the key, key[0] through key[k-1], and integer variables, i, j, and K. Performing a modular reduction of some value modulo 256 can be done with a bitwise AND with 255.

4. PROPOSED IDENTITY BASED ENCRPTION ALGORITHM

Modified Encryption algorithm (MEA) executes the task with a low execution time when compared to other algorithms.

4.1 Secret Key Generation and Encryption

Step 1. Generate a secret key *Kpr* using user identity.
Step 2. Generate another random number *'N'*.
Step 3.Determine location to store secret key in encrypted data file using formula *Loc = |shared key (Kpu) – reverse of shared key (Kpu)| + 'N'.*
Step 4. Convert *‗N'* in to its binary equivalent ‗Bn*'*.
Step 5. Repeat steps 6 and 9 till EOF.
Step 6. Read a character *Pi* from the plain text file.
Step 7. Encrypt it and store its cipher text (*Cp)* value in the encrypted data file. *Cp=E (Kpr(Pi))* where i= 1,2…n
Step 8. Encrypt secret key using a shared key. *Ck =E(Kpu(Kpr))*
Step 9. Increment secret key by one.
Step 10. Write encrypted secret key at position Loc in the encrypted data file.
Step 11. Upload this whole encrypted file to remote server. *C*= Bn *Cp… Cp Ck Cp ….Cp*

4.2 Key Retrieval and Decryption Algorithm

Step 1. Read 4 characters *(Bn)* from encrypted data file and convert into decimal number say*'N'*.
Step 2. Determine location of the secret key in encrypted data file using formula *Loc = |shared key (Kpu) – reverse of shared key (Kpu)| +'N'*
Step 3. Set file pointer=Loc.
Step 4. Repeat step 5 for i=1 to 4.
Step 5. Read 9 characters and decrypts them by shared key. *Kpr* =D(*Kpu* (*Ck))* //secret key has been determined.

Step 6. Set file pointer at 5th position. //First 4 positions are occupied by random number.
Step 7. Repeat steps 8 and 9 till EOF of the encrypted data file.
Step 8. If (file pointer != Loc) then Read 9 characters one by one and decrypts it by secret key. *Pi=D(Kpr(CP))* else file pointer= Loc+36
Step 9. Concatenation of decrypted characters gives plain text *(P) (data).*

4.3 Flowchart of Secured IBDDS

The mechanism uses three different phases of operation which includes the implementation of data Integrity, Authentication and Security using an Encryption Algorithm, Digital Signature and a Key Exchange mechanism. Encryption is implemented using the Identity based Encryption Algorithm, Digital Signature Algorithm for Data integrity and Diffie Hellman Key exchange mechanism. These three different approaches can be combined to provide a greater level of security, Initially User access to the data present will be given only based on the priority of the user and user can access available data on the same terms and User will be examined for authentication using the decryption mechanism after which the data can be made available. Initially, the data will be present in the server in an encrypted format and user should have to enter the decryption key for using the data, keys will be identities of the user sine the encryption mechanism is identity based approach. The diffie Hellman key exchange mechanism will check for the correctness of the key given by the user. After the key validation mechanism, if the key proves to be correct the data will be made available to the user or else the access to the data will be denied. Thus Authentication is achieved.

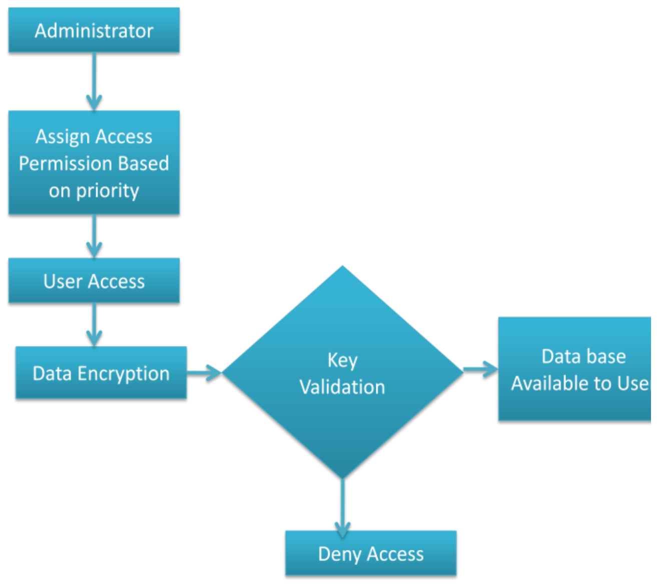
Administrator
Assign Access Permission Based on priority
User Access
Data Encryption
Key Validation
Data base Available to Use
Deny Access

Fig 1: Flowchart of Secured IBDDS

These mechanisms combined together can provide a high security feature over data.

Integrity Verification is done by comparing the Digital Signatures of the data at both sides of the Sender and the receiver. If certain data or users need to be given only read rights then the same can be achieved and verified by using the Digital Signature Mechanism

5. RESULTS AND DISCUSSIONS

Simulation is carried out using gridsim. Various comparisons are made based on the data size using the different algorithms and the same are represented below

5.1.1 Using File size of 1.58MB

Table 1 shows the simulation results for a file size of 1.58MB

Table 1.Comparison of Encryption algorithm

Algorithms	Average Encryption Time (ms)	Average Decryption Time (ms)
RC4	0.504	0.239
DES	0.792	0.399
AES	0.684	0.315
RSA	0.495	0.289
MEA	0.362	0.189

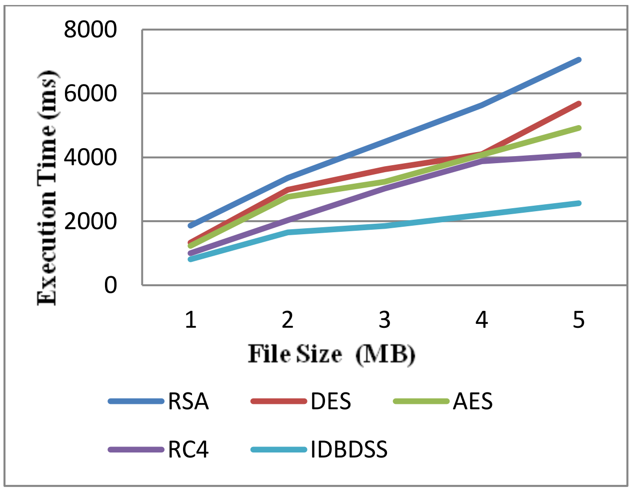

Fig 2: File size vs. Encryption time

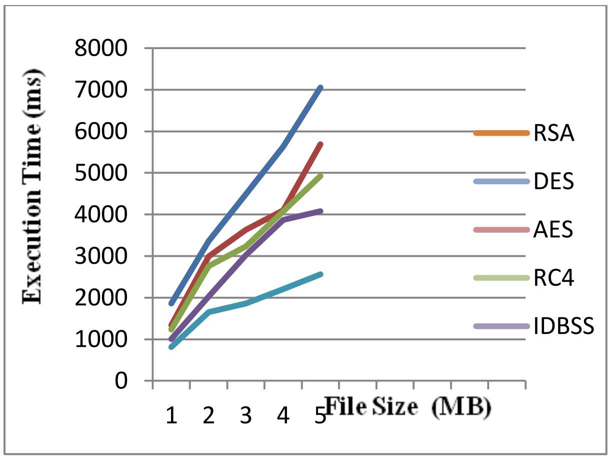

Fig 3: File size vs. Decryption time

5.1.2 Using file size as 10.51 MB

Table 2 shows the simulation results for a file of size 10.51 MB.

Table 2. Comparison of encryption algorithm with respect to execution time

Algorithms	Average Encryption Time (ms)	Average Decryption Time (ms)
RC4	3.453	1.54
DES	5.432	2.66
AES	4.694	2.1
RSA	4.864	1.69
MEA	2.47	1.4

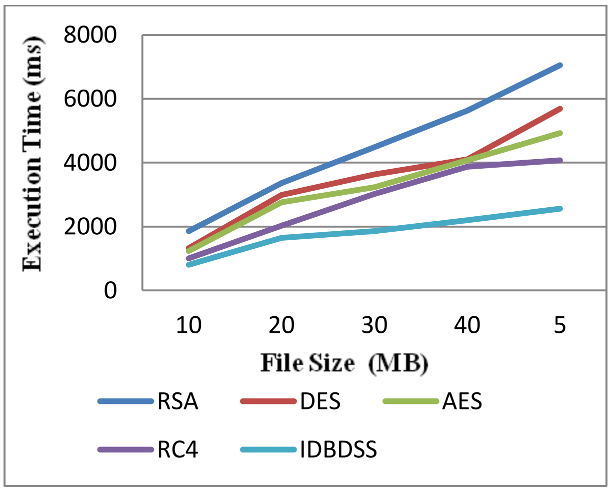

Fig 4: File Size vs Encryption Time

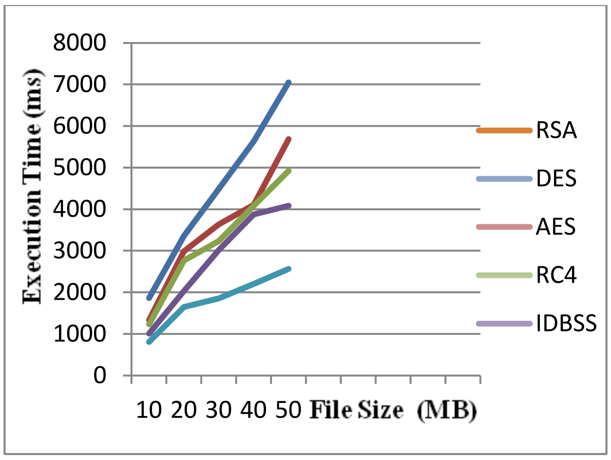

Fig 5: File Size vs. Decryption time

6. CONCLUSION AND FUTURE WORK

The overall comparative performance analysis in Grid simulation has shown that MEA is more efficient in reducing the Execution time than RSA, AES, RC4 and DES algorithms. And the combined features provides a greater performance for Identity Based Encryption Scheme and the added mechanisms like digital Signature and Key exchange mechanism gives more Integrity to the data stored in the distributed environment.

In future, the algorithm can be implemented in Grid Network using Globus Toolkit for adding additional security performance to the system even with increased number of data Sources.

7. REFERENCES

[1] V. Khmer and Y. Kim, “Securing distributed storage: Challenges, techniques, and systems,” in *Proc. ACM Workshop On Storage Security And Survivability - StorageSS’05 (V. Atluri, P. Samarati,* W. Yurcik, L. Brumbaugh, and Y. Zhou, eds.), (Fairfax, VA, USA), pp. 9–25, ACM, Nov. 2005.

[2] A.Sahai and B.Waters,”Fuzzy identity based encryption,”Proc. Acvances in Cryptography –Eurocrypt 2005,vol.3494,LINCS,pp 457-473.

[3] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, “Improved proxy re-encryption schemes with applications to secure distributed storage,” *ACM Transactions on Information and System Security, vol. 9, no. 1, pp. 1–30, 2006.*

[4] M. Green and G. Ateniese, “Identity-based proxy re-encryption,” in *Proc. Applied Cryptography and Network Security - ACNS’07* (J. Katz and M. Yung, eds.), vol. 4521 of *Lecture Notes in Computer Science, (Zhuhai, China), pp. 288–306, Springer, Jun. 2007.*

[5] LiHua, and ZhaoJianPing, “Security Research on P2P Networks” International Conference on Computational Intelligence and Software Engineering, Wuhan, pages 1 – 5, 11-13 Dec. 2009

[6] C. Wang, K. Ren, W. Lou, and J. Li, “Towards Publicly Auditable Secure Cloud Data Storage Services,” IEEE Network Magazine, vol. 24, no. 4, pp. 19-24, July/Aug. 2010.

[7] P. Samarati and S. D. C. di Vimercati, “Data protection in outsourcing scenarios: Issues and directions,” in *Proc. ACM Symposium on Information, Computer and Communications Security - ASIACCS’10 (D. Feng, D. A. Basin, and P. Liu, eds.), (Beijing,* China), pp. 1–14, ACM, Apr. 2010.

[8] H.-Y. Lin and W.-G. Tzeng, “A secure erasure code-based cloud storage system with secure data forwarding,” *IEEE Transactions on Parallel and Distributed Systems,* Digital Object Identifier - 10.1109/TPDS.2011.252, June 2012

[9] Prashant Rewagad and Yogita Pawar, “Use of Digital Signature with Diffie Hellman Key Exchange and AES Encryption Algorithm to Enhance Data Security in Cloud Computing” in International Conference on Communication Systems and Network Technologies, DOI -10.1109/CSNT.2013.97, 2013