International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

SECURED FILE TRANSFER WITH ADVANCED ENCRYPTION STANDARD

G. Yugeandiran[1], Mrs. R. Vijayalakshmi[2]

[1]*Master of Computer Applications, Krishnasamy College of Engineering and Technology, Cuddalore*
[2]*MCA, MPhil., (Ph.D.), Associate Professor, Department of Computer Applications, Krishnasamy College of Engineering and Technology, Cuddalore*

ABSTRACT

The Advanced Encryption Standard (AES) is an encryption algorithm that is used for securing sensitive unclassified information. AES is proved to be a highly secure, faster, and strong encryption algorithm. AES is used commonly because of its great competence and ease. In reality, the communication channel which is used to transfer data from transmitter to receiver is highly insecure. To resolve this problem the data is being manipulated to another form so that the person with access to the secret key can only read it. This process of manipulation of original data to another form so that an eavesdropper cannot access it is known as encryption. Advanced Encryption Standard (AES) is the most commonly used algorithm for data encryption. This algorithm can be applied to both text and image files that want their file to be secure enough. Cipher Block Chain (CBC) mode achieves this by XOR-ing the first plaintext block with an initialization vector before encrypting it. CBC also involves creating a block as every subsequent plaintext block is XOR-ed with the ciphertext of the previous block. Thus by utilizing CBC user data can be secured by encryption and decryption which eventually makes it non-vulnerable for an eavesdropper.

1. INTRODUCTION

Secure file transfer is a data sharing method that uses secure protocols and encryption to safeguard data in transit. Most secure file sharing solutions use industry-standard protocols that provide encrypted file transfer, such as SFTP, FTPS, HTTPS, and AS2, among others. Here we are using Advanced encryption standard (AES) for encrypting the files.

Advanced Encryption Standard:

AES is iterative rather than Feistel cipher. It is based on a 'substitution–permutation network'. The Advanced Encryption Standard (AES) is an encryption algorithm that was selected by the National Institute of Standards and Technology (NIST) for the United States government, commercial, and private organizations to use for securing sensitive unclassified information. It comprises a series of linked operations, some of which involve replacing inputs with specific outputs (substitutions) and others involve shuffling bits around (permutations). Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix. Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

Encryption and Decryption by using AES:

1. Take the input file
2. Now add the key generated by Elliptical Curve Cryptography (ECC) which is the public key.
3. AES encryption is performed on the input file by using the public key which is generated by ECC.
4. The encrypted file is uploaded on the server after the encryption by AES.
5. Once the file is uploaded then it will be downloaded at the server, and then file is translated by using the public key given by ECC so that the original file is decrypted.
6. The performance of the system depends on the combined effect of ECC and AES, such as the storage space optimization and enhancement of the security services over the cloud server.

Secure Computing:

A security system identifies and mitigates the system vulnerabilities, by either removing them or restricting access to them, to a very small group. The competition between inventing new security measures to protect data and inventing hacking techniques in conjunction with discovering and leveraging pre-existing vulnerabilities is infinite. Therefore, securing data and resources is becoming more and more challenging day by day. Nevertheless, there exist several different techniques to secure the data being transferred over a network and also that on a user machine. Specializes in securing data in motion through the use of the patented REAL-ID based mutual authentication scheme. Secure Sockets Layer (SSL) is one such tool to secure data sent over a network, using ciphertext. Using SSL, data is kept confidential and message integrity is maintained.

However, recently there have been network security breaches, including the famous "heartbleed" bug. But, the question that remains is what if the user machine itself is hacked? it can be used to ensure that the end-user is secured as well as the tunnel. It also uses techniques of authentication to assure each end-user that it is communicating with an authorized user and not a fake one. Such security measures are used to secure data in motion, meaning data that has been shared between computers. They may prove to be of minimum value if the operating system on which it resides is compromised. It is, therefore, crucial to understand and remove the security flaws in the operating system itself. We, on the other hand, are trying to secure data at rest, by coming up with various approaches, one of which is application white listing.

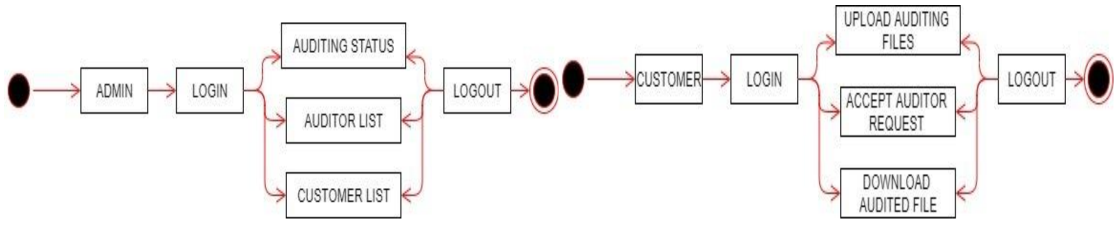
Hardening is a technique to reduce vulnerabilities of the existing operating system. It aims to eliminate security risks in an operating system. This is done by turning off all those services of the operating system which are not used are risky and allowing only those which are secure for user's data. Thus, this environment becomes a kind of locked down or reduced version of a full-fledged operating system. Operating system hardening is a technique that allows us security on the machine level. A hardened operating system can be considered as a smaller version of an otherwise compromised operating system. Secondly, we implement a technique called application white listing. It is the technique of preparing a list of all applications that are safe to execute. All applications that are excluded from this list are disallowed to spawn.

2. LITERATURE SURVEY

AES is most effective algorithm to provide security in message transmission. Singh et al. [1] studied three encryption algorithms like Rivest-Shamir-Adleman, Data Encryption Standard, Triple Data Encryption Standard and Advanced Encryption Standard to analyze effectiveness cryptography based on speed, time, and throughput and avalanche effect.[2] Researchers proved that Advanced Encryption Standard is a better algorithm than Data Encryption Standard, Triple Data Encryption Standard and Rivest-Shamir-Adlel for communication security. Mahajan et al. [3] surveyed the performance of existing encryption techniques like Advanced Encryption Standard, Data Encryption Standard and Rivest-Shamir-Adleman algorithms. Based on the investigation researchers determined that Advanced Encryption Standard algorithm consumes least encryption and Rivest-Shamir-Adleman consumes longest encryption time. Also, Decryption of Advanced Encryption Standard algorithm is better than other algorithms. From the simulation result, it is estimated that Advanced Encryption Standard algorithm is greatly better than Data Encryption Standard and Rivest-Shamir-Adleman algorithm. Padmavathi et al. [4] implemented three encryption algorithms like Data Encryption Standard, Advanced Encryption Standard and Rivest-Shamir-Adleman along with Least Significant Bit Substitution to analyze effectiveness cryptography based on encryption and decryption time and buffer usage. Researchers proved that Advanced Encryption Standard is better algorithm than Data Encryption Standard, Triple Data Encryption Standard and Rivest-Shamir-Adleman for communication security. Musliyana et al. [5] have used Dynamic key generation in AES to solve attack vulnerability. Researchers proposed to use function of time. Key can be generated at random based on the value of the time when sender logs in to the system. The decryption process takes a time value with certain tolerance limits to find the same key pair of the time value generated in the encryption process. It provided stronger cipher key for AES encryption and decryption.

3. PROPOSED SYSTEM

In the proposed system, we have an Auditing System implemented to maintain the process of securing files and time management. Business owners are struggling to get their accounts to be audited by the expertise as there is a chance of data breach which leads to catastrophe. Traditional encryption will inevitably result in multiple different ciphertexts produced from the same plaintext by different users' secret keys, which hinders data duplication. The proposed technique has AES CBC mode has been implemented to encrypt data in files for high-security purposes. When customers upload files, the encryption process is started, here a 128 bits key was used to encrypt data. For converting the access key to a 128-bit key, the base64 encoder was proposed. In that file, data was split into blocks of data, each block is encrypted by a different initialization vector. Auditors request the key from the customer to decrypt the file sent by them, auditors revert files to customers then customers can decrypt the file using the key from their default settings. Our approach has proved that our application effectively secures customer auditors file transactions and creates a secure environment for the highly delicate data transmission between two or more parties.



4. ALGORITHM

According to the scheme construction, convergent keys in our scheme KeyD are encrypted utilizing the Identity-Based Broadcast Encryption (IBBE) keys. The semantic security of convergent keys requires that it is computationally hard for an adversary to distinguish convergent keys according to their corresponding encrypted versions. It also implies that He cannot distinguish the convergent key is encrypted using an IBBE key or a random value in K by its encryption, where K is the key space of an IBBE scheme, namely GT . Formally, The semantic security of convergent keys is defined using the following game between a challenger C and an attacker A, and the security parameter λ in IBBE scheme is given to both players:

1. C runs algorithm SetupIBBE(λ, m) to generate system master key MSK and public key PK, where m is the maximum number of owners of a data block.

2. C randomly chooses a data block B and calls the algorithm KeyGenCE(B) to generate the corresponding convergent key kc. And He also randomly constructs a receiver set S, the number of identities in which is supposed to be s. Then C calls the algorithm EncIBBE(S, PK)to generate an IBBE key kB and randomly selects b ∈ {0,1}. He sets kb = kB and k1−b to a random value in K. Finally, He computes two encryptions ck0 = EncSE(kb, kc), ck1 = Encase(k1−b, kc) and gives B, ck0, ck1 to the attacker.

3. The attacker outputs b′ and wins the game if b = b′ . In other words, the attacker wins if He correctly guesses whether the convergent key kc of B is encrypted under kB or a random value in K. And we define attacker A's advantage in winning the game as AdvA = |P r[b = b′] − 1/2|. Since the IBBE scheme has been demonstrated to be IND − sID − CPA secure in using the General Decisional Diffie-Hellman Exponent (GDDHE) framework, we can easily obtain the following theorem. Theorem 1. Convergent keys in KeyD are semantically secure if the IBBE scheme is secure under the GDDHE assumption. Proof 1. Let the data block is all included in a dictionary S. Suppose we have a semantic security attacker A for which AdvA > ε. We build an attacker B that breaks the IND − sID − CPA security of IBBE where AdvB > ε/(CS'S · |S|). The reduction is immediate: B is given a block B and two encryptions ck0, ck1 (one of them is an encryption under an IBBE key and the other is under a random value in K) of B's convergent key kc, with the IBBE receiver set S of B, where |S| = s. B randomly chooses another s ′ − s identities and combine them with S to construct a "whole" receiver set S ′ . B sends S ′ and S to A who then responds with a data block Bi and a receiver set Si on which A wishes to be challenged. If Bi is not equal to B or Si is not equal to S algorithm B reports failure and aborts. Otherwise, B sends the challenge B, ck0, ck1 to A who then responds with a b ′ ∈ {0, 1}. The algorithm outputs b' as its response to the semantic security challenge. We can observe that b ′ = b if algorithm B did not abort and A's response was correct. This probability is at least 1/2 + ε/(C s s ′ · |S|). Hence AdvB > ε/(C s s ′ · |S|).

5. ADVANTAGES OF PROPOSED WORK

(i) Secure computing creates trust between different entities where trust is either nonexistent or unproven.

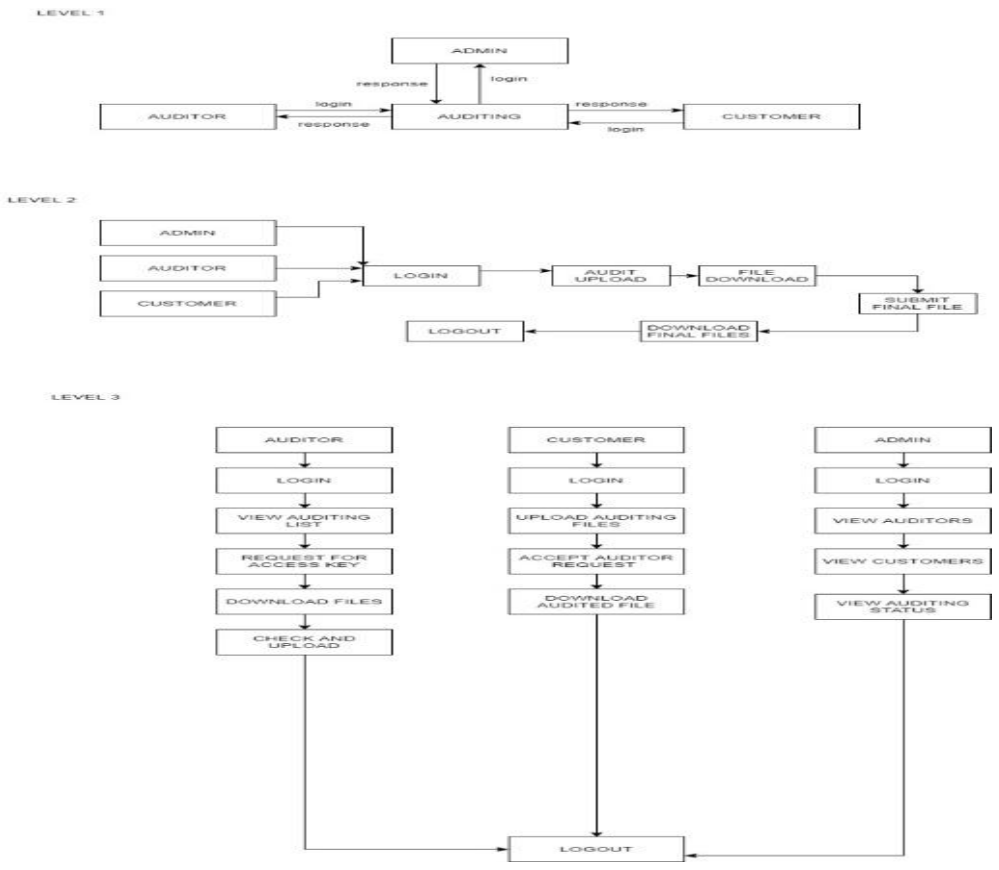
(ii) Our encryption algorithm creates an unalterable record of transactions with end-to-end encryption, which shuts out fraud and unauthorized activity.

(iii) The proposed approaches have a superior speed advantage.

(iv) No practical cryptanalytic attacks against AES have been discovered.

(v) Accuracy has improved compared to the existing traditional encryption method.

LEVELS OF DIAGRAM:



6. RESULT

The proposed approaches show not only good encryption performance but privacy has improved compared to existing. Here the application uses test data. In the future, we will launch the system to real-world applications. We show experimentally that the approach has robust security for highly confidential documents between businesses or any.[6] In future studies, we intend to extend the proposed approaches to a wider range of applications to further demonstrate their feasibility for practical use. With secure computing implementation, sectors like trade finance witnessed reduced processing time, eliminated paperwork, and became cost-efficient while maintaining security and trust.

REFERENCES

[1] Amazon Web Services, [Online]. Available: http-s://aws.amazon.com/cn/.

[2] D.A. Sarma, X. Dong, and A. Halevy, Bootstrapping pay-as-you-go data integration systems[C]. ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, Bc, Canada, June. DBLP, 2008:861-874.

[3] J.R. Douceur, A. Adya, W.J. Bolosky, D. Simon, and M. Theimer, Reclaiming Space from Duplicate Files in a Serverless Distributed File System[C]. Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on. IEEE, 2002: 617-624.

[4] S. Ghemawat, H. Gobioff, and S. Leung, The Google File System[M]. SOSP '03 Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003, 37(5): 29-43.

[5] D. Borthakur, HDFS architecture guide[J]. Hadoop Apache Project, 2008, 53.

[6] J. Li, X. Chen, M. Li, J. Li, P.P.C. Lee, and W. Lou, Secure Deduplication with Efficient and Reliable Convergent Key Management[J]. IEEE transactions on parallel and distributed systems, 2014, 25(6): 1615- 1625.