INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 3 ISSUE 6 (2017) PAGES 2650 - 2654
RECEIVED : 05.11.2017 PUBLISHED : 25.12.2017

Dec 25, 2017

Improve The Network Security with Secret Sharing

[1]R. Ganeshan, Assistant Professor, Dept of CSE, St. Joseph College of Engineering Sriperumbudur.

[2]Dr. Paul Rodrigues, Professor, Dept of CSE, Indra Gandhi College of Engineering and Technology Chengalpattu.

Abstract:

In today's high tech technology modern world everyone mostly store their Personal data in the Cloud system which may has passwords, bank details and other important information that may be used for bad purpose by a miscreant, an opponent, or a court of law. These data are read, copied, and archived by Cloud Service Providers (CSPs), often without author's permission and control. Self-destructing data plays a very important role in protecting the user data's privacy. All the data stored at servers and their copies destroyed after a user-specified time, and also any user cannot read this data. The decryption key is destructed after the user-specified time that is ttl (time-to-live) field. In proposed system, by presenting self-destructing data system that meets upto this challenge through a novel integration of secure cryptography techniques with active storage techniques that are based on 'hadoop'. We implement a proof-of- concept SeDas prototype. By functionality and security properties evaluates the SeDas prototype, the results conclude that SeDas is practical to use and achieve all the privacy-preserving goals described. Compared to the system without self-destructing data mechanism, efficiency of uploading and downloading with the proposed SeDas acceptably decreases, while latency for uploading and downloading operations with self-destructing data mechanism increases effectively.

Keywords: Active Storage, Cloud computing, Data Privacy, Self-Destructing Data.

1. INTRODUCTION

With the fast development of Cloud computing and popularization of mobile Internet, cloud services are becoming much important for peoples as a part of life. Peoples are more or less desired to store or post some personal private information on the Cloud using Internet. When people doing this, they hope that service providers will provide security policy to secure their data from hacking, so others people will not lose their privacy. Scheme starts with a secret and then assume from it certain shares which are distributed among users (i.e., participants). The secret may be variously resolve (i.e., recovered) only by certain predetermined subgroups of users which constitute the access structure. The important category of access structure is the (w,N)-threshold access structure in which, given N shareholders, an authorized group consist of any w or more participants and any group of at most w−1 participants is an illegal group. As peoples attracted more and more to the Internet and Cloud, privacy of personal data is at more risk. On the one hand, when database is being in operation by the current system or network, systems or network must cached, copy or archive it. These copies are essential for computer system and the network. However, people don't have knowledge about these copies and user cannot control them, so these copies may leak their confidentiality. On the other hand, their privacy also can be leaked by Cloud Service Providers (CSP's), hacker's intrusion or some fair actions. These problems present dangerous challenges to protect people's privacy. Secret sharing has broad applications in this real world and can be used for situations in which access to important resources to be protected. There is old story which have motivated the secret sharing principle [8]: a group of pirates accidentally detected a map that would lead them to an enclave full of treasure. Today database and networking era, sharing data secretly is also a fundamental issue in network security and can be used in key administration and multi-party secure computation.

2. RELATED WORK

Safe Vanish: An Improved Data Self-Destruction for Protecting Data Privacy: In the background of cloud, self-destructing data mainly aims at protecting the data privacy. All the data and its copies will become destructed or unreadable after a user-specified period, without any user intervention. Besides, anyone cannot get the decryption key after timeout, neither the sender nor the receiver. The Washington's Vanish system is a system for self-destructing data under cloud computing, and it is vulnerable to “hopping attack” and “sniffer attack”. We propose a new scheme in this paper, called Safe Vanish, to prevent hopping attacks by way of extending the length range of the key shares to increase the attack cost substantially, and do some improvement on the Shamir Secret Sharing algorithm implemented in the Original Vanish system. We present an improved approach against sniffing attacks by using the public key cryptosystem to protect from sniffing operations. In addition, we evaluate analytically the functionality of the proposed Safe Vanish system.

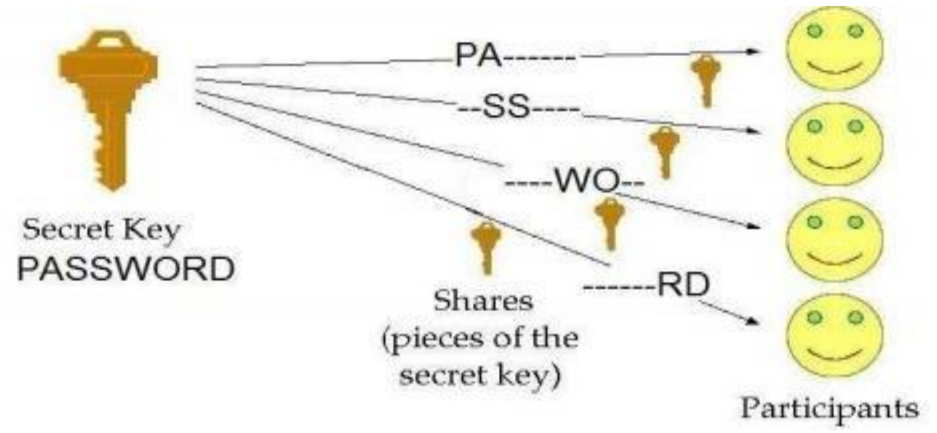

Fig.1.System

Energy-driven methodology for node self-destruction in wireless sensor networks: Wireless sensor networks technology is a rapidly growing domain, getting more and more credit in the area of civilian and military applications. In the same time with technological advancement, new and dangerous information security threats have emerged. In this paper we considered that a node self-destruction procedure must be performed as a final stage in the sensor node lifecycle in order to assure the confidentiality regarding information like: network topology, type of measurement data gathered by sensors, encryption/authentication algorithms and key-exchange mechanisms, etc. that can be unveiled otherwise through reverse engineering methods. Our methodology relies on an efficient power monitoring scheme, based on combined in-network and predictive data, which discover the low battery nodes and initiate a self-destruction procedure for that nodes. In the existing system there are multiple disadvantages are available. In this Hacker can attack the confidential data and gain all the information from the database.

3. PROPOSED SYSTEM

This is big disadvantages of this system. Because client want to security of the data which is confidential from other’s. In this hacking process the sensitive data can be modified by anyone, or if anyone can do changes in this client database then it is very harmfull for client. When the map function completes execution and starts producing output, it is not only written to disk. Theprocess is more involved, and takes advantage of buffering writes in memory and doing some presorting for efficiency reasons. The map output file is sitting on the local disk of the tasktracker that ran the map task, but now it is needed by the tasktracker that is about to run the reduce task for the partition. Furthermore, the reduce task needs the output of map task for its particular partition from several map tasks among the cluster. The map tasks may finish at different times, so the reduce task copies outputs as soon as each completes. This is known as the copy phase of the reduce task. The reduce task has a small number of copier threads so that this can fetch outputs of map in parallel. The default is five threads, but this number are changed by setting the mapred.reduce parallel copiesproperty. scheme involves a Key Server (KS) to generate and distribute shared key to all group members via a secure channel. A decentralized key management divides the whole group into smaller subgroups. Each subgroup is controlled by a single or several KS. A Distributed scheme allows each member to take part in a group key generation collaboratively. Each of the three schemes has its own advantages and disadvantages. Centralized scheme is the simplest one but has the risk of single-point-failure. Decentralized scheme adds some communication complexity between two members within different subgroups. Distributed scheme is somehow more complex than the other two, but it doesn't involve KS. This feature is very useful in the case of no one can play the role of KS, e.g. a sensor Ad-hoc network application.

In a key pre-distribution scheme, a trusted authority generates and distributes secret pieces of information to all users offline. At the beginning of a conference, users belonging to a privileged subset can compute individually a secret key common to this subset. A family of forbidden subsets of users must have no information about the value of the secret. The main disadvantage of this approach is to require every user to store a large size of secrets. Group key transfer protocol relies on one trusted entity, KGC, to choose the key, which is then transported to each member involved. Each user is required to register at KGC for subscribing the key distribution service. The KGC keeps tracking all registered users and removing any unsubscribed users. During registration, KGC shares a secret with each user. In most key transfer protocol, KGC encrypts the randomly selected group key under the secret shared with each user during registration and sends the ciphertext to each group member separately. An authenticated message checksum is attached with the ciphertext to provide group key authenticity. In this approach, the confidentiality of group key is ensured using any encryption algorithm which is computationally secure. Our protocol uses secret sharing scheme to replace the encryption algorithm. A broadcast message is sent to all group members at once. The confidentiality of group key is information theoretically secure. In addition, the authentication of broadcasting message can be provided as a group authentication. This feature provides efficiency of our proposed protocol.

4. ANALYSIS

Adversaries can be categorized into two types. The first type of adversaries are outsiders of a particular group. The outside attacker can try to recover the secret group key belonging to a group that the outsider is unauthorized to know. This attack is related to the confidentiality of group key. In our proposed protocol, anyone can send a request to KGC for requesting a group key service. The outside attacker may also impersonate a group user to request a group key service. In security analysis, we will show that the outside attacker gains nothing from this attack since the attacker cannot recover the group key. The second type of adversaries are insiders of a group who are authorized to know the secret group key; but inside attacker attempts to recover other member's secret shared with KGC. Since any insider of a group is able to recover the same group key, we need to prevent inside attacker knowing other member's secret shared with KGC.

Assume that an attacker who impersonates a group member for requesting a group key service, then the attacker can neither obtain the group key nor share a group key with any group member Although any attacker can impersonate a group member to issue a service request to KGC without being detected and KGC will respond by sending group key information accordingly; however, the group key can only be recovered by any group member who shares a secret with KGC. This security feature is information theoretically secure. If the attacker tries to reuse a compromised group key by replaying previously recorded key information from KGC, this attack cannot succeed in sharing this compromised group key with any group member since the group key is a function of each member's random challenge and the secret shared between group member and KGC. A compromised group key cannot be reused if each member selects a random challenge for every conference.

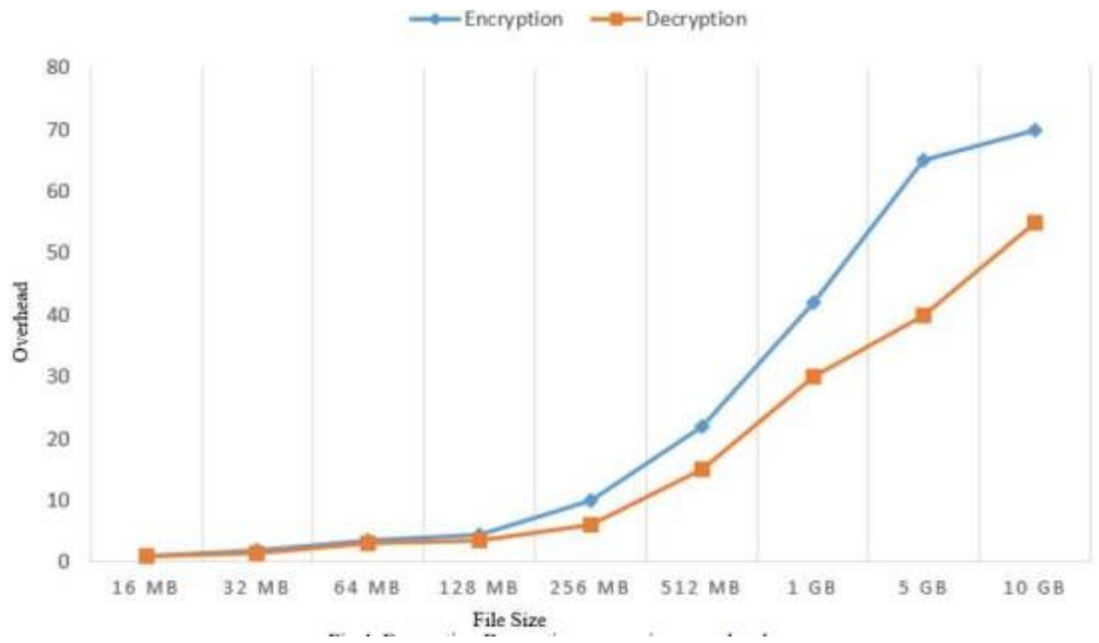

Fig.2.Analysis

Multicast key management, which is much different from unicast key management , is one of the most attractive area of cryptography. For an unicast application, the Diffie-Hellman key exchange protocol can be employed to establish a KEK (Key Encryption Key) between two entities. Then use this KEK to dispatch or update a session key. In contrast, the situation is much more complicated for a multicast application. A multicast application must dynamically handle multi- entities. For example, in a dynamic multicast group, the membership is changeable all the time due to frequently users' addition and eviction. Therefore, the key materials will probably be revealed if no security policies are adopted. For instance, if the key is not updated after the membership change, a new comer is able to read the contents before his coming, or a evictor is capable of reading the content after his leaving. In this case, multicast key management scheme should provide forward secrecy and backward secrecy for security reasons in some special applications.

INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 3 ISSUE 6 (2017) PAGES 2650 - 2654
RECEIVED : 05.11.2017 PUBLISHED : 25.12.2017 Dec 25, 2017

CONCLUSION

In this paper we proposed a novel mechanism for group key transfer protocol based on secret sharing with the help of trusted KGC and preshare a secret with KGC and it broadcasts all the information slimantaniously.It provides the security measures like confidentiality and authentication. We provide group key authentication. Security analysis for possible attacks is included.

REFERENCES

[1] G.R. Blakley, "Safeguarding Cryptographic Keys," Proc. Am. Federation of Information Processing Soc. (AFIPS '79) Nat'l Computer Conf., vol. 48, pp. 313- 317, 1979.

[2] S. Berkovits, "How to Broadcast a Secret," Proc. Eurocrypt '91 Workshop Advances in Cryptology, pp. 536- 541, 1991.

[3] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," Proc. Eurocrypt '84 Workshop Advances in Cryptology, pp. 335-338, 1984.

[4] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly Secure Key Distribution for Dynamic Conferences," Information and Computation, vol. 146, no. 1, pp. 1-23, Oct. 1998.

[5] C. Boyd, "On Key Agreement and Conference Key Agreement," Proc. Second Australasian Conf. Information Security and Privacy (ACISP '97), pp. 294-302, 1997.

[6] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater, "Provably Authenticated Group Diffie-Hellman Key Exchange," Proc. ACM Conf. Computer and Comm. Security (CCS '01), pp. 255-264, 2001.

[7] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably- Secure Authenticated Group Diffie-Hellman Key Exchange," ACM Trans. Information and System Security, vol. 10, no. 3, pp. 255-264, Aug. 2007.