An Asymmetric Key based Disk Encryption Scheme

Dhiman Sarma
Rangamati Science and Technology University
Department of Computer Science and Engineering

ABSTRACT
This paper prompts to understand an asymmetric key based improved disk encryption scheme designed in a tree structure, and analyses the security of encrypted disk data. Conventional disk encryption software use encryption cryptography based on symmetric key, and unable to protect the secret key due to design and implementation weaknesses. Hence, a disk encryption scheme is proposed here which uses asymmetric key cryptography to ensure stronger data security by utilizing the advantage of having two keys over one keyed symmetric key cryptography. The conceptual model includes entangling asymmetric key cryptography with tree structure to provide multiple encrypted layers embedded within the nodes from leaf to root.

General Terms
Information Security, Cryptography

Keywords
Encryption, Asymmetric Key, Disk Security, Disk Encryption Scheme, Message authentication code

1. INTRODUCTION
Disk encryption scheme is not only a software implementation of encryption algorithm [1] but also considered as an interactive protocol that is operated between the design implementation, the encryption module [2] and the physical hard disk. Most of the commercial disk encryption systems use symmetric key algorithm [1,3] like Advanced Encryption Standard (AES)[4]. Encryption systems are useless when the secret key can be recovered. An adversary tries to gain the secret key by either attacking the core encryption system or manipulating the encryption software itself. The primary challenge of designing an improved disk encryption system is to protect the secrete key [5] as well as to preserve data integrity. The proposed focuses to address both the issues scheme in section III.

2. REVIEWING THE THEORY
Encryption [5] is the mechanism of transforming original data, called plaintext, into a form that appears to be unreadable or random, called ciphertext [5]. Decryption [5] is the reverse mechanism of encryption to transform the ciphertext to plaintext. Encryption algorithms [1] are set of mathematical formulas which construct how enciphering and deciphering take place. Algorithms are not themselves secret parts of the encryption process but the way those algorithms works are kept secret from the public. A secret value, called a key [5], is used in conjunction with the encryption algorithm to kept the mechanism secrete. Depending on the key, two types of encryption algorithms are introduced named as symmetric key encryption [3,6] and asymmetric key encryption [3,6]. In symmetric key encryption, same key is used for encryption and decryption on the both sender and receiver sides. In asymmetric key encryption, pair of two different keys is used, one is called Public key [7] and the other is Private Key [7]. Public key is sent and shared by the communicating parties to encrypt the plain text, and corresponding private Key remains secret to the owner computer to decrypt the same cipher text. A hash function [8,9] is a mathematical transformation function which takes a variable-size input and produces output string of a fixed-size called the hash value [6]. A Message Authentication Code (MAC) [10], produced by hash function, is used to ensure integrity of encrypted data. A Tree [11] is a hierarchical data structure where every node has exactly one parent (except the root) and several or no child. An encryption module [2] is a physical encryption device which performs the encryption and decryption according to algorithm.

3. DESIGNING THE ENCRYPTION SCHEME
My scheme is designed to have a number of nodes arranged into a tree structure, and it uses asymmetric key algorithm like RSA [12]. I also assume that the platform also consist of a hardware encryption module with embedded memory. Each leaf nodes of the tree contains the plaintext of a logical sector, and each internal node contains a key which is generated randomly by a random key generator [13]. Except the root node, the content of each node is encrypted by the key which resides in the parent node. At the top of the tree the root key is encrypted by the public key, and the public key is stored in the memory of the encryption module. The private key is encrypted by password and stored on disk. Authentication [8,9] mechanism can be done with Message Authentication Code (MAC) by using hash tree.

3.1 Encryption
- Generate random keys for each node from root node to the leaf node.
- Encrypt the content of each node with the key of their parent node.
- Write the ciphertext on to the disk.
- Generate the MAC of the hash tree with the root key K_root and keep the MAC in the memory of encryption module.
- Encrypt K_root with the public key K_pbc.
- Keep the public key K_pbc in the memory of encryption module.
- Encrypt the private key K_pvt with password and store the K_pvt on to the disk.

3.2 Decryption
- At boot time, use password to decrypt the private key K_pvt.
- Decrypt the K_root with K_pvt.
- Get the MAC on the root of the hash tree.
- Decrypt contents by using K_root and keys from root to the leaf node.
- Erase password and K_pvt.

3.3 Changing password

- Decrypt the K_root with K_pvt.
- Generate MAC on the root of the hash tree with K_root.
- Generate a new asymmetric key pair and a new password.
- Finally, encrypt the private key with new password and keep K_pvt in to the disk.

4. SECURITY ANALYSIS AND CONCLUSION

The security strength of this scheme is as follows: Assume that the adversary gets either old or current password. Any data which are written beyond the life time of the captured password cannot be accessed because those data was written using a set of keys that are independent from those keys which are used during the captured password. Note that, the keys in the tree are not changed when password is changed. When new data are written to the disk, only then new keys are generated and old keys are erased. Therefore, the adversary can get only the snapshots of the current state during the boot time which is similar to leaking the password. Otherwise, the adversary can learn the current root but not the private key. This means that data cannot be decrypted which are written after when the snapshot was taken. Because, a write operation always update all the keys (along with the root key) on the path of the data written, and the adversary cannot decrypt those keys since he has no private key. This scheme utilizes the advantage of asymmetric key cryptography of having two keys over symmetric key cryptography; otherwise, root key can be decrypted with one symmetric key which can be obtained from the snapshot. Thus the adversary could continue to decrypt data which are written later on. If the adversary has the password or a snapshot, he can get the key which are currently used for generating the MAC on the root of the hash tree.

Therefore, he can give any root to the encryption module for the hash tree together with a valid MAC during boot time. As a result, the adversary can reach to a complete disk content which is either existed earlier or fabricated one by that the adversary. But, the adversary cannot modify the contents after booting the system since the encryption module keeps the current hash value in the memory. Hence, the authenticity is preserved by this scheme.

I believe that this paper emphasizes a new direction for designing and analyzing disk encryption systems for protecting digital information from the external environment.

5. FUTURE SCOPE

The further scope remains in incorporating a smart card[1] where we can securely store the private key [15]. Activating the smart key requires biometrics[2] of the user along with the password. The model can protect the private key which reside inside the card and the key remains safe even if the card is lost.

[1] A Smart card is a card with embedded integrated circuit which provides identification, authentication, data storage and application processing.

[2] Biometrics are unique biological measurements like fingerprints, facial measurements, iris, retina, the patterns that ones veins make and even the way one walk

6. ACKNOWLEDGMENTS

Thanks to Professor Dr. Pradanendu Bikash Chakma, Vice Chancellor of Rangamati Science and Technology University. Also special gratitude to my MSc thesis supervisor Sead Muftic (Ph.D), Professor Emeritus, School of Information & Communication Technology, Royal Institute of Technology (KTH), Sweden.

7. REFERENCES

[1] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73.

[2] N. Goots, B. Izotov, A. Moldovyan, and N. Moldovyan, Modern Cryptography: Protect Your Data with Fast Block Ciphers, ed 2003.

[3] Nedjah, N., and Mourelle, L.d.M. 2005. Embedded Cryptographic Hardware: Design & Security.

[4] Gupta, P.C. 2014. Cryptography and Network Security.

[5] National Institute of Standards and Technology (NIST). 2001. Federal Information Processing Standards Publication 197 Advanced Encryption Standard (AES).

[6] Mollin, R.A. 2007. An Introduction to Cryptography.

[7] Menezes, A.J., Vanstone, S.A., and Oorschot, P.C.v. 2001. Handbook of Applied Cryptography.

[8] Dent, A.W., and Mitchell, C.J. 2005. User's Guide to Cryptography and Standards.

[9] Thorsteinson, P., and Ganesh, G.G.A. 2004. Net Security and Cryptography.

[10] Patel, D. 2008. Information Security: Theory and Practice.

[11] Stinson, D.R. 2005. Cryptography: Theory and Practice.

[12] Lipschutz, S. 2010. Data Structure with C.

[13] Rivest, R., Shamir, A., and Adleman, L. 1978. A Method for Obtaining Digital Signatures and Public Key Cryptosystem.

[14] Schmeh, K. 2006. Cryptography and Public Key Infrastructure on the Internet.

[15] Sarma, D., 2012. Security of Hard Disk Encryption. Masters Thesis. Identifiers: urn:nbn:se:kth:diva-98673(URN), Royal Institute of Technology.