INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 2 ISSUE 2 (2016) PAGES 743-748
RECEIVED : 22/03/2016. PUBLISHED : 29/03/2016
March 29, 2016

DELETING SECRET DATA WITH PUBLIC VERIFIABILITY

V.pradeep[1], S.sriganeshkumar[2],C.santhosh[3],S.navin arulappan[4],R.venkatesh[5],
[1,2,3,4]Dept of Information Technology,S.K.P Engineering College, Tiruvannamalai,
[5]Asst prof, Dept of Information Technology,S.K.P Engineering College, Tiruvannamalai.

Abstract:

Existing software-based data erasure programs can be summarized as following the same one-bit-return protocol: the deletion program performs data erasure and returns either success or failure. However, such a one-bit-return protocol turns the data deletion system into a black box – the user has to trust the outcome but cannot easily verify it. However, the return of the"Success" bit can be misleading. Although the link of the file has been removed, the content of the file remains on the disk. An attacker with a forensic tool can easily recover the deleted file by scanning the disk. This is especially problematic when the deletion program is encapsulated within a Trusted Platform Module (TPM), and the user has no access to the code inside.

Keywords – TPM, Black box, One bit.

1. INTRODUCTION

In this paper, we present a cryptographic solution that aims to make the data deletion process more transparent and verifiable. In contrast to the conventional black/white assumptions about TPM (i.e., either completely trust or distrust), we introduce a third assumption that sits in between: namely, "trust-but-verify". Our solution enables a user to verify the correct implementation of two important operations inside a TPM without accessing its source code: i.e., the correct encryption of data and the faithful deletion of the key. Finally, we present a proof-of-concept implementation of the SSE system on a resource-constrained Java card to demonstrate its practical feasibility. To our knowledge, this is the first systematic solution to the secure data deletion problem based on a "trust-but-verify" paradigm, together with a concrete prototype implementation.

2. EXISTING SYSTEM

Deletion methods can be described using essentially the same protocol, which we call the "one-bit-return"protocol. In this protocol, the user sends a command– usually through a host computer – to delete data from a storage system, and receives a one-bit reply indicating the status of the operation. Take the deletion in the Windows operating system as an example. When the user wishes to delete a file (say by hitting the "delete" button), the operating system removes the link of the file from the underlying file system, and returns onebit to the user: Success. The same problem also applies to the default deletion program bundled in other operating systems

3. PROPOSED METHOD

Proposed to first generate a random AES key for encrypting data and then use the password to wrap the AES key and store the wrapped key on the disk. This is essentially equivalent to deriving the key from the password. The wrapped key now becomes an oracle, against which the attacker can run the exhaustive

search. The third method is to store the key in a decentralized network. Along this line, Geambasu et. al. proposea solution called Vanish, which generates arandom key to encrypt the user's data locally andthen distributes shares of the key using Shamir'ssecret sharing scheme to a global-sale, peer-to-peer,distributed hash tables (DHTs). The shares of thekey naturally disappear (vanish), due to the fact thatthe DHT is constantly changing. However, Wochok et. al subsequently show two Sybil attacks that work by continuously crawling the DHT andrecovering the stored key shared before they vanish. They conclude that the original Vanish scheme cannotguarantee the secure deletion of the key.

4. HARDWARE REQUIREMENTS:

Hard Disk	:	500 GB
Monitor	:	15 VGA color
Mouse	:	Logitech.
Keyboard	:	110 keys enhanced.
RAM	:	2 GB

Software Requirement

Operating system	:	Windows 7
Front End	:	JAVA JDK 1.7
Tool	:	Net Beans 7.0 IDE
Database	:	MySQL

5. LITERATURE SURVEY

He Oracle Diffie-Hellman Assumptions and an Analysis of DHIES," Topics in Cryptology

This paper provides security analysis for the public-key encryption scheme DHIES (formerly named DHES and DHAES), which was proposed in [7] and is now in several draft standards. DHIES is a Diffie-Hellman based scheme that combines a symmetric encryption method, a message authentication code, and a hash function, in addition to number-theoretic operations, in a way which is intended to provide security against chosen-ciphertext attacks. In this paper we find natural assumptions.

True Erase: Per-File Secure Deletion for the Storage Data Path

The ability to securely delete sensitive data from electronic storage is becoming important. However, current per-file deletion solutions tend to be limited to a segment of the operating system's storage data path or specific to particular file systems or storage media. This paper introduces TrueErase, a holistic secure-deletion framework. Through its design, implementation, verification, and evaluation, TrueErase shows that it is possible to build a legacy-compatible full-storage-data-path framework that stems or

storage media. This paper introduces TrueErase, a holistic secure-deletion framework. Through its design, implementation, verification, and evaluation, TrueErase shows that it is possible to build a legacy-compatible full-storage-data-path framework that performs per-file secure deletion and works with common file systems and solid-state storage, while handling common system failures. In addition, this framework can serve as a building block for encryption- and tainting-based secure-deletion systems.

Data Remanence in Flash Memory Devices

Data remanence is the residual physical representation of data that has been erased or overwritten. In non-volatile programmable devices, such as UV EPROM, EEPROM or Flash, bits are stored as charge in the floating gate of a transistor. After each erase operation, some of this charge remains. Security protection in microcontrollers and smartcards with EEPROM/Flash memories is based on the assumption that information from the memory disappears completely after erasing.While microcontroller manufacturers successfully hardened already their designs against a range of attacks, they still have a common problem with data remanence in floating-gate transistors. Even after an erase operation, the transistor does not return fully to its initial state, thereby allowing the attacker to distinguish between previously programmed and not programmed transistors, and thus restore information from erased memory. The research in this direction is summarised here and it is shown how much information can be extracted from some microcontrollers after their memory has been ‘erased’.

Scalable Security for Petascale Parallel File Systems

Petascale, high-performance file systems often hold sensitive data and thus require security, but authentication and au-thorization can dramatically reduce erformance. Existing security solutions perform poorly in these environments because they cannot scale with the number of nodes, highly distributed data, and demanding workloads. To address these issues, we developed Maat, a security protocol de-signed to provide strong, scalable security to these systems. Maat introduces three new techniques. Extended capabilities limit the number of capabilities needed by allowing a capability to authorize I/O for any number of client-file pairs.Automatic Revocation uses short capability lifetimes to al-low capability expiration to act as global revocation, while supporting non-revoked capability renewal. Secure Delegation allows clients to securely act on behalf of a group toopen files and distribute access, facilitating secure joint computations. Experiments on the Maat prototype in the Cephpetascale file system show an overhead as little as 6-7%.

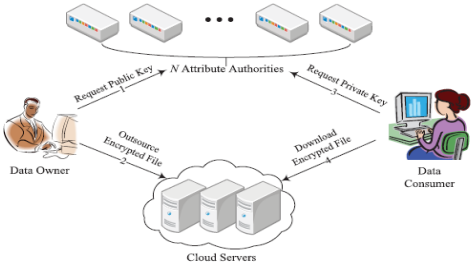

Fig.1.Architecture diagram

Public cloud owner:

In this module main work are based on admin approval system. If admin can approve to deleting the data from admin side to cloud server this server can take some extra backup about public verifiable data with the verification on secrete key method.

User request:

In this module are have request role to admin side and it can do this work based on user public data.

Make secrete data:

In this module are used to make some data from user side to request side it may occur some useful data, secret data and useless data they always manipulated from public cloud.

Key distributer:

In this project main role are key distributer they can used to delete some secrete data from public cloud from cloud server with the help of authorized mobile number. They need some basic information about current user.

CONCLUSION

While the "trust-but-verify" paradigm has been well studied and established in some fields (e.g., e-voting), it has been almost entirely neglected in the field of secure data deletion. In this paper, we initiate an investigation on how to apply the "trust-but-verify" paradigm to make the data deletion process more transparent and verifiable.We present a concrete cryptographic solution, called Secure Storage and Erasure (SSE), which enables a user to verify the correct implementation of cryptographic operations inside a TPM without having to access its internal source code. The practical feasibility of our solution is validated by a proof-of-concept implementation using a resourcecontained Java card as the TPM. Future work includes extending the "trust-butverify" paradigm to other crypto primitives, in particular, the secure random number generator. The problem of permitting end users to audit if a random number has been generated correctly in a TPM as part
of the encryption process (or a cryptographic protocol) is still largely unsolved and deserves further research.

REFERENCES

[1] M. Abdalla, M. Bellare and P. Rogaway, "The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES," Topics in Cryptology - CT-RSA'01, LNCS Vol. 2020, 2001.

[2] B. Adida, "Helios: Web-Based Open-Audit Voting," Proceedings of the 17th USENIX Security Symposium, pp. 335-348,2008.

[3] R.J. Anderson, Security Engineering : A Guide to Building Dependable Distributed Systems, 2nd edition, New York, Wiley 2008.

[4] A. Antipa, D. Brown, A., Menezes, R. Struik, S. Vanstone, "Validation of Elliptic Curve Public Keys," Proceedings of the 6th International Workshop on Practice and Theory in Public Key Cryptography Public Key Cryptography (PKC'03), LNCS 2567, pp. 211-223, 2003.

5] S. Bauer, N.B. Priyantha, "Secure Data Deletion for Linux File Systems," Proceedings of the 10th USENIX Security, 2001.

[6] C, Burton, C. Culnane, J.A. Heather, P.Y.A. Ryan, S. Schneider,
T. Srinivasan, V. Teague, R. Wen, Z. Xia, "Using Pret a Voter in Victorian State Elections," Proceedings of the 2012 Electronic Voting Technology/Workshop on Electronic Voting (EVT/WOTE'12), 2012.

[7] D. Boneh, R. Lipton, "A Revocable Backup System," Proceedings 6th USENIX Security Conference, pp. 91-96, 1996.

[8] C. Cachin, K. Haralambiev, H.C. Hsiao, A. Sorniotti, "Policy- Based Secure Deletion," Proceedings of the 2013 ACM Conferenceon Computer and Communications Security (CCS'13),pp. 259-270, 2013.

[9] D. Chaum and T.P. Pedersen, "Transferred Cash Grows in Size," Proceedings of EUROCRYPT, pp. 390-407, 1993.

[10] A. Fiat, A. Shamir, "How to Prove Yourself: Practical Solution to Identication and Signature Problems," Proceedings of CRYPTO, pp. 186-189, 1987.
Increasing Data Privacy with Self-Destructing Data," Proceedings of the USENIX Security Symposium, 2009.

[12] S. Garfinkel, A. Shelat, "Remembrance of Data Passed: A Study of Disk Sanitization Practices," IEEE Security & Privacy, Vol. 1, No. 1, pp. 17-27, 2003.

[13] P. Gutmann, "Secure Deletion of Data from Magnetic and Solid-State Memory," Proceedings of the Sixth USENIX Security Symposium, pp. 22-25, 1996.

[14] P. Gutmann, "Data Remanence in Semiconductor Devices," Proceedings of the 10th conference on USENIX Security Symposium, 2001.

[15] F. Hao, M. Kreeger, B. Randell, D. Clarke, S. Shahandashti, P. Lee, "Every Vote Counts: Ensuring Integrity in Large-Scale Electronic Voting," USENIX Journal of Election Technology and Systems (JETS), Vol. 2, No. 3, 2014.

[16] N. Joukov, H. Papaxenopoulos, E. Zadok, "Secure Deletion Myths, Issues, and Solutions," Proceedings of the second ACM workshop on Storage Security and Survivability (StorageSS), pp. 61-66, 2006.

[17] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, K. Fu, "Plutus: Scalable Secure File Sharing on Untrusted Storage," Proceedings of the 2nd USENIX Conference on File and Storage Technologies (FAST'03), pp. 29-41, 2003.

[18] R. Kissel, M. Scholl, S. Skolochenko, X. Li, "Guidelines for Media Sanitization," NIST Special Publication 800-88, 2006.

[19] T. Kohno, A. Stubblefield, A.D. Rubin, and D.S. Wallach, "Analysis of an Electronic Voting System," Proceedings of the 25th IEEE Symposium on Security and Privacy, May, 2004.

[20] J. Lee, S. Yi, J.Y. Heo, H. Park, S.Y. Shin and Y.K. Cho, "An Efficient Secure Deletion Scheme for Flash File Systems," Journal of Information Science and Engineering, Vol. 26, pp. 27-38,2010.