SELF-ENCRYPTION SCHEME FOR DATA SECURITY IN MOBILE DEVICES

[1]SRINIVASAPPA K V, [2]SUDHARANI B S

[1]Lecturer, [2] Lecturer

[1]Department of Computer Science and Engineering, [2]Department of Computer Science and Engineering

[1] Government Polytechnic KGF, Kolar, Karnataka, India, [2]Government Polytechnic Karkala, Udupi,Karnataka, India

Abstract: The pervasive use of wireless networks and mobile devices has been changing our living style significantly. Along with great convenience and efficiency, there are new challenges in protecting sensitive and/or private data carried in these devices. The most challenging part lies in a dilemma: while it should be computationally infeasible for adversaries to decrypt the data, the cryptographic operation should be efficient for legitimate users and minimize battery drain. This paper proposes a novel data encryption and storage scheme to address this challenge. Treating the data as a binary bit stream, our self-encryption (SE) scheme generates a keystream by randomly extracting bits from the stream. The length of the keystream depends on the user's security requirements. The bit stream is encrypted and the ciphertext is stored on the mobile device, whereas the keystream is stored separately. This makes it computationally not feasible to recover the original data stream from the ciphertext alone.

Index Terms - **Data security, mobile devices, stream cipher.**

I. INTRODUCTION

The pervasive use of wireless networks and mobile devices has been changing our living style significantly [5], [25]. Along with great convenience and efficiency, the progress of technology also brings new challenges in protecting sensitive and/or private information carried in these devices [14]. New vulnerability results from unique characteristics of mobile devices. For instance, due to constraints imposed by limited computing power, storage space, and battery lifetime, a light-weight rather than computing intensive and complex encryption algorithm is desired in the mobile devices [1].

In addition, portability makes mobile devices prone to being stolen or lost. It is very challenging to protect the weakly encrypted information on a mobile device, which might end up in the hands of an adversary, who could then use powerful cryptanalysis tools to break the encryption [21]. Therefore, security solutions developed for general distributed data storage systems cannot be adopted directly for this new frontier.

Statistics show that 22% of PDA owners have lost their devices, and 81% of those lost devices had no protection. Even worse, 37% of PDAs have sensitive information on them, such as bank account information, corporate data, passwords, and more [28]. For this reason, some companies do not allow employees to use PDAs or similar mobile devices to store company data [26]. However, effective protection that would enable the full and convenient use of these devices without the fear of losing or compromising data would be a much better scenario.

The most challenging part of mobile device data protection lies in the conflicting requirements for the data encryption scheme. While it should be computationally infeasible for adversaries to decrypt the data in captured mobile devices, the encryption/decryption operation should be reasonably efficient for legitimate users. Furthermore, the required computations should not consume too much energy so as to minimize battery drain.

This research proposes a novel stream cipher scheme called self-encryption (SE) to address this dilemma. Treating the data set as a binary bit stream, we generate the keystream by extracting n bits in a pseudorandom manner based on a user's unique PIN and a nonce. The length of the keystream n is flexible and depends on the security requirements. Then we encrypt the remaining bit stream using this keystream.

The encrypted remainder is stored in the mobile device, whereas the keystream is stored separately. It is very difficult to recover the original data stream from the ciphertext even if an adversary has the knowledge of the encryption algorithm. The variable length

keystream makes brute force attacks infeasible, and the decrypted data stream is still unrecognizable unless the keystream bits are inserted to the original position.

.

II. RELATED WORK

Securing sensitive and/or private data in mobile communication has been an important topic in security research community [8], [19], [25]. Our research is relative to two main areas: modern stream cipher design and distributed data security.

2.1 Modern Stream Cipher Design

The author Stream ciphers are widely used to protect sensitive data at fast speeds [3], [27]. Although block ciphers have been attracting more and more attention, stream ciphers still are very important, particularly for military applications and to the academic research community. Stream ciphers are more suitable in environments where tight resource constraints are applied, i.e. in wireless mobile devices [4], [27], or wireless sensor networks [8]. When there is a need to encrypt large amount of streaming data, a stream cipher is preferred [3].

In recent years, a lot of efforts have been reported in this area and many interesting new stream ciphers have been proposed and analyzed. A popular trend in stream cipher design is to turn to block-wise stream ciphers like RC4, SNOW 2.0, and SCREAM [16]. In order to improve the time-data-memory tradeoff for stream cipher, a concept of Hellman's time-memory tradeoff [4] has been applied and it achieved obvious improvements [12]. The Goldreich-Levin [11] one-way function hard-core bit construction has been enhanced into a more efficient pseudo-random number generator BMGL [15] with a proof of security.

Efficient hardware implementations of stream ciphers are important in both high-performance and low-power applications [16]. This is the main trend of the stream cipher development in the future. Researchers have pointed out that RFID (Radio Frequency Identification) could be one of the next killer applications for hardware-oriented stream ciphers [27]. The second phase of the eSTREAM project in particular focused stream ciphers suited toward hardware implementation and currently there are eight families of hardware-oriented stream ciphers [7].

Normally there are two input parameters to a stream cipher, the password and an initialization vector (IV). In contrast with the user password being kept secret, the IV is public. As a consequence, attacks against the IV setup of stream cipher have been very successful [29]. Due to the weakness with the IV setup, more than 25% of the stream ciphers submitted to the eSTREAM project in May 2005 have been broken [2]. Some seemingly robust academic designs were broken also due to problems with the IV setup [29]. In this paper, we will investigate an alternative design approach for the self-encryption stream cipher scheme to avoid the shortcomings incurred by using public IV. Also, the robustness of a fixed length keystream has been weakened as the computing power which an adversary possesses has been growing. Instead, a variant length keystream will make brute force attacks computationally infeasible. To reach this goal, this paper will also introduce a novel keystream generation scheme.

2.2 Distributed Data Security

In the Effective data protection solution is one of the essential security requirements that affect the acceptance of next generation pervasive computing [10] and the mobile device utilization in enterprise networks [5]. The rapid increase of sensitive data and the growing number of government regulations require long-term data retention to storage security [26]. During the data's life cycle, there are a lot of potential attack points. In past decades, many researchers have contributed in this area. Among the reported works, here we skip the great achievements in network file system since they are not very relative to the proposed project. Instead, we will briefly introduce the recent progress in security services for distributed data storage protection.

Data should be protected during the whole life cycle. Authentication and authorization are the preliminary requirements in most data security systems [23]. In general, authentication can be implemented using techniques such as passwords, digital signatures, or MAC (Message Authentication Code). Authorization can be performed by certificates, access control, etc. Considering the risks of system crash or denial-of-service, availability is required in most commercial systems. Typical solution is to make duplicated backup. However, replication increases the cost of consistency maintenance.

The essential task of data security is to prevent any unauthorized third party from revealing or modifying the data. Confidentiality can be achieved by using encryption, while data integrity can be achieved by using digital signatures and/or MAC. During transmit the data can be protected by using protocols such as SSL [9] and IPSec [18]. Meanwhile, at the storage the data confidentiality can be achieved using user encryption schemes. Variant cipher schemes are proposed for this purpose including the new designs we mentioned in previous section, eSTREAM project [7].

To be robust against cryptanalysis, the key sharing [17] and key management [24] are also critical part in the context. Special care has to be taken while storing, archiving, and deleting key materials. Another important research area is the key recovery system [6], which helps the users to decrypt the ciphertext under certain conditions. Considering the constraints in mobile devices and the asymmetric power available to the adversary, there is no existing solution can be adopted directly to address the data security question in mobile devices. The proposed project is to investigate more robust stream cipher scheme by exploring more flexible keystream generation methods and more secure keystream management approach. The detailed discussion of our proposed research works is presented in the next section.

III. PROPOSED SYTEM Self-Encryption Scheme

Clustering This section consists of two parts. First of all, we will introduce a framework under which the sensitive and/or private data are separated and stored in a distributed manner. Secondly, we will specify the detailed design of our SE scheme.

3.1 Framework of SE Scheme

Considering the fact that generally mobile devices do not possess as many resources as normal computers, it is very challenging to prevent an adversary from breaking the embedded cryptographic algorithm when the mobile devices are captured. It is also not desirable to implement a complex computing intensive encryption/decryption scheme in a mobile device. Therefore, the rationale of this project is to investigate a novel light-weight approach to protect the information effectively even if an adversary has good knowledge of the encryption algorithm and many more resources to break the cryptography.

To reach this goal, our essential idea is that an adversary can only obtain part of the data from the mobile device alone, which is not enough to reveal any useful information. As illustrated by a scenario shown in Figure 1, the sensitive data is broken into two parts using our self-encryption stream cipher scheme. The major part (Part A: ciphertext) is stored in the mobile device carried by the company employee, and the minor part (Part B: keystream + other parameters) is protected in the secure server of the company. Part A is encrypted using part B. When the user needs to access the data, he or she has to input a correct PIN to pass the authentication procedure. Then the server will send part B to decrypt part A and merge them together to recover the original plaintext. When a mobile device is lost, at most the adversary can access the part A, from which it is computationally infeasible to get meaningful information..
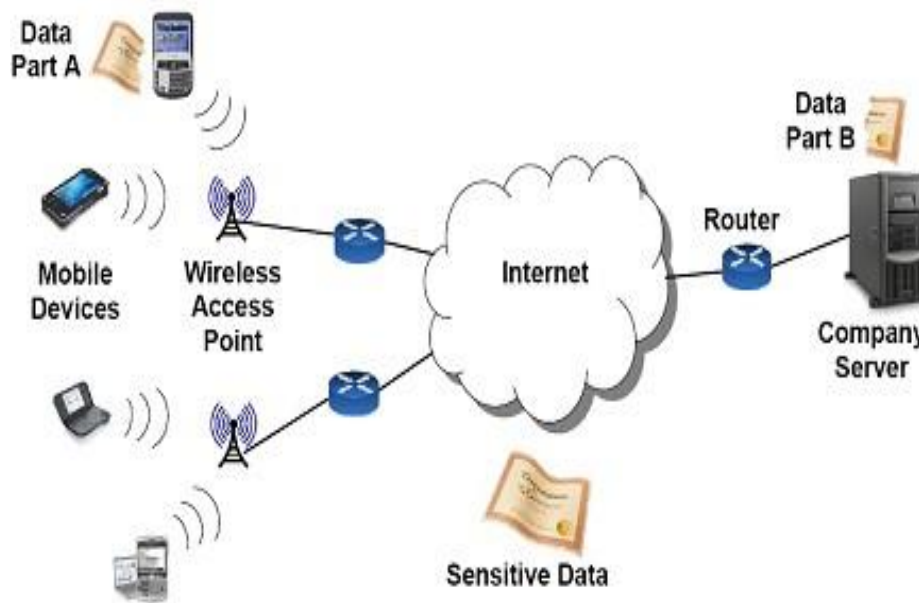

Figure 1. Overview of the Self-Encryption framework.

3.2 Self-Encryption Scheme

Similar as general stream cipher, the proposed SE stream cipher also encrypts the plaintext and decrypts the ciphertext by adding bitwise a keystream:

$$Ciphertext = Plaintext \oplus Keystream \qquad (1)$$

The keystream generator consists of two parts, a hash function H and a random number generator G. The hash function takes the user's PIN and a nonce as input and the output is an integer seed, which is used as the seed of the random number generator G. The output random number sequence $\{r_0, r_1, \ldots r_{n-1}\}$ indicates which bits are selected and abstracted from the message (plaintext) to form the keystream. Therefore, we have:

$$seed = H(PIN, nonce) \qquad (2)$$

$$\{r_0, r_1, \ldots r_{n-1}\} = G(seed) \qquad (3)$$

where $\{r_0, r_1, \ldots r_{n-1}\}$ is a random number sequence generated by the random number generator G. Since the random numbers could beyond the length of the message, and the length of the message body decreases as bits are abstracted, the pointers to the keystream bits need to be normalized following the changing message size. Hence $\{r_0, r_1, \ldots r_{n-1}\}$, the position of among the n abstracted bits the k-th bit is:

$$r'_k = r_k \bmod (m-k) \qquad (4)$$

The length of the random number sequence n, which is also the length of keystream, is determined by the size (number of bits) of the message m and the security requirement. A longer the keystream provides more robust cipher to protect a larger size message. To support this flexibility, we define parameter security as the security level SL and Δ as the minimum length level unit difference between two consecutive security levels. Δ is a percentage instead of a fixed bit number. This design leads to a unique length of each keystream depending on the concrete message size. It makes the brute force attacks much difficult as the working load for keystream guess is increased exponentially. The keystream length n is calculated as:

$$n = \begin{cases} m \times S_L \times \Delta, & if \quad S_L \neq 0 \\ 256, & if \quad S_L = 0 \end{cases} \quad (5)$$

To illustrate the use of equation (5), assume Δ= 5%, for example, then the length of the keystream can be 5% of the original message size when SL =1, 10% when SL=2, 15% when SL=3 and so on. When SL = 0, a default fixed keystream length is adopted, where n = 256 bits. Actually, further experimental and theoretical analysis will be conducted to set the optimal value of Δ. The use of security should be specified in more detail in the design of level SL the SE protocol in the future.

Figure 2 presents the working flow of the proposed SE stream cipher. When the user has finished editing or reading the document, the following works are performed. The seed of the random number generator is calculated by the hash function taking the user's PIN and a nonce as the input. Then, according to the size of the sensitive document and the security level, a sequence of random numbers is generated with length n. By treating the file as a binary stream, this random number sequence indicates which bits in the data file are abstracted to form the keystream.
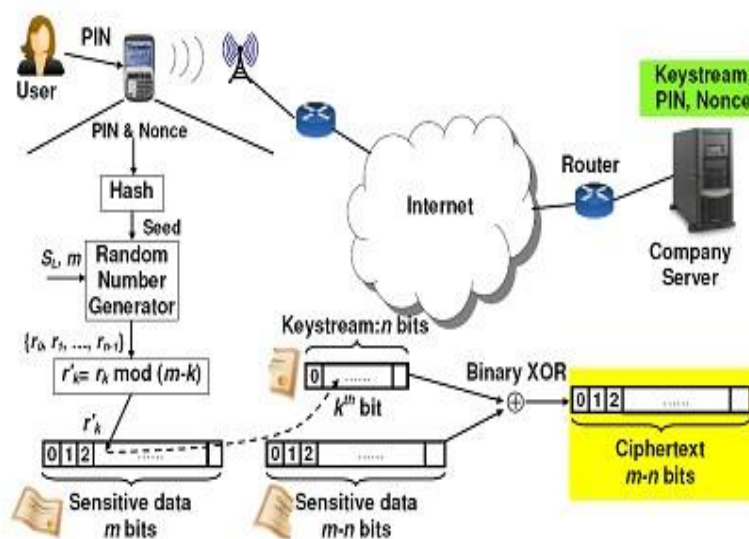

Figure 2. SE Scheme Working Flow Illustration.

Then the ciphertext is calculated as normal stream cipher does. The ciphertext is stored in the mobile device, the keystream, user's PIN, and the nonce are stored the secure server in the company. We will investigate the tradeoff between the performance and security regarding the information to be transferred back to the server. For instance, maybe it is more secure not to transfer the user's PIN and nonce, instead, backing up the sequence {r'0,r'1, r'n-1} is better.

Comparing to existing stream cipher schemes, computationally the proposed SE scheme is much more robust. The length of the keystream is not fixed except when the default value (256) is adopted, if the user selected security level SL = 0. This raises the bar of brute force attackers, the complexity is increased to O(2 m).

Furthermore, to recover the original data stream, the adversary needs to insert every bit of the keystream back correctly. The permutation in this operation is:

$$P_n^m = \frac{m!}{(m-n)!} = m \times (m-1) \times (m-2) \times \cdots \times (m-n+1) \quad (6)$$

The complexity of this part is O(mn).Then the total complexity is O(2mmn), which is much robust than the reported modern stream cipher schemes.

IV. MOTIVATION AND GOALS

➢ Keystream Generation: A New key stream is generated for every encryption of data. More flexible key generationmethods are used.
➢ Key Management: Key management is critical part in this context. Special care should be taken while storing, archivingand deleting key materials. More secure key management is used.

➢ Encryption/Decryption: The data is encrypted, in order to prevent unauthorized access and decrypted at receiver end. An every robust and effective algorithm is used.
➢ Self Destruction: In case of theft, if the thief can open the lock of mobile, we can secure our data by self destructing it.
➢ Backup of data: After self destructing, we will lose the data that is present in the device. So in order to recover that data, we will create a backup of that data.
➢ Securing of key and cipher text: The key is secured as it is stored at the server and the cipher text is stored at the mobile device.

V. CONCLUSION AND DISCUSSIONS

Lack of effective protection of sensitive data in mobile devices is a major concern that prevents the mobile devices from being used widely as part of enterprise networks or personal area networks. The proposed SE system will remove the barrier and enable employees to enjoy the high efficiency and convenience brought by mobile devices. It will lead to another wave of prosperity of wireless networks and pervasive computing. Physical attacks have been proved effective in breaking some well designed ciphers in practice [13]. Unfortunately, it is challenging to designers to theoretically investigate the robustness of a cipher scheme against various physical attacks. To address this problem, a prototype is going to be implemented on top of reconfigurable hardware devices (i.e. FPGAs). Particularly we will study the behavior of our SE prototype under local non-invasive attacks including timing analysis and differential power analysis (DPA) [20].

$Etotal= Ech +Enon-CH$

VI. RESULTS AND DISCUSSION

To validate the proposed clustering algorithm based on node degree and distance from the base station for cluster head selection, the performance of proposed method was compared with the existing clustering algorithm like HEED. Performance metrics are- total energy dissipation in node and Number of clusters was considered to evaluate the performance. Proposed algorithm and HEED simulation was carried out in MATLAB 7.0. First of all, 100 WSN nodes were randomly distributed in a spatial region of 100mx1000m network area. In HEED algorithm, 5% of the nodes have been taken as cluster-heads. Each node transmits 256 bits of information during transmission. For the better simulation the first order radio energy model parameters considered are Eelec=50 nJ/bit, Eamp=50 nJ/bit, EDA=5 NJ/bit and ERX=100nJ/bit.

REFERENCES

[1] J. Al-Muhtadi, D. Mickunas, and R. Campbell, "A Lightweight Reconfigurable Security Mechanism for 3G/4G Mobile Devices," IEEE Wireless Communications, April 2002.
[2] D. J. Bernstein, "Which eSTREAM ciphers have been , submitted 2008-broken?" http://www.ecrypt.eu.org/ stream/02-21.
[3] A. Biryukov, "Block Ciphers and Stream Ciphers: The State of the Art," Lecture Notes in Computer Science, in Proceedings of the COSIC Summer course, 2003.
[4] A. Biryukov and A. Shamir, "Cryptanalytic time/memory/data tradeoffs for stream ciphers," in Proceedings of Asiacrypt'00, no. 1976 in Lecture Notes in Computer Science, pp. 1–13, Springer-Verlag, 2000.
[5] W. Daniel, T. Pintaric, F. Ledermann, S. Dieter, "Towards Massively Multi-User Augmented Reality on Handheld Devices", International Conference on Pervasive Computing, Munich, Germany, 2005.
[6] D. E. Denning and D. K. Branstad, "A Taxonomy for Key Escrow Systems," Communications of the ACM, Vol. 39, Issue 3, 1996.
[7] eSTREAM, ECRYPT Stream Cipher Project, http://www.ecrypt.eu.org/stream
[8] N. Fournel, M. Minier, and S. Ubeda, "Survey and Benchmark of Stream Ciphers for Wireless Sensor Networks," the Workshop in Information Security Theory and Practices (WISTP'07), Crete, Greece, May 8-11, 2007.
[9] A. O. Freier, P. Karlton, and P. C. Kocher, "The SSL Protocol, Version 3.0," Internet draft, Networking Group, March 1996.
[10] C. Galdi, A. Del Sorbo, and G. Persiano, "Distributed Certified Information Access for Mobile Devices," Workshop in Information Security Theory and Practices (WISTP'07), Crete, Greece, May 8-11, 2007.
[11] O. Goldreich and L. A. Levin, "A hard core predicate for any one way function," in Proceedings of Symposium on Theory of Computing – STOC'89 , pp. 25–32, ACM Press, 1989.
[12] J. D. Golic, "Cryptanalysis of alleged A5 stream cipher," in Advances in Cryptology – EUROCRYPT'97, vol. 1233 of Lecture Notes in Computer Science, pp. 239–255, edited by
W. Fumy, Springer-Verlag, 1997.
[13] T. Good and M. Benaissa, "Hardware performance of eStream phase-III stream cipher candidates," the State of the Art of Stream Ciphers Workshop (SASC'08), Lausanne, Switzerland, Feb. 13-14, 2008.
[14] K. Greene, "Securing Cell Phones," Technology Review, MIT, Wednesday, Aug. 01, 2007.
[15] J. Hastad and M. Naslund, "Improved analysis of the BMGL keystream generator," in Proceedings of the Second NESSIE Workshop, 2001.
[16] D. Hwang, M. Chaney, S. Karanam, N. Ton, and K. Gaj, "Comparison of FPGA-Targeted Hardware Implementations of eSTREAM Stream Cipher Candidates," the State of the Art of Stream Ciphers Workshop (SASC'08), Lausanne, Switzerland, Feb. 13-14, 2008.
[17] Y. Jiang, C. Lin, M. Shi, and X. Shen, "Multiple Key Sharing and Distribution Scheme with (n, t) Threshold for NEMO Group Communications," IEEE Journal on Selected Areas in Communications, Vol. 24, No. 9, Sep. 2006.
[18] A. Kent and R. Atkinson, "Security Architecture for the Internet Protocol," RFC 2401, Networking Group, Nov. 1998.

[19] V. Kher and Y. Kim, "Securing Distributed Storage: Challenges, Techniques, and Systems," StorageSS'05, Fairfax,Virginia, USA, Nov. 11, 2005.

[20] P. Kocher, J. Jaffe and B. Jun, "Differential power analysis", Advances in Cryptology (Crypto '99), Lecture Notes inComputer Science, 1666 (1999), Springer-Verlag, 388-397.

[21] Mobile Defender, Sirius Information Technologies, http://sit.bulhost.com/index.html, as of Mar. 2008.

[22] NetFPGA official homepage, http://yuba.stanford.edu/ NetFPGA/, as of Feb. 1, 2008.

[23] A. J. Nicholson, M. D. Corner, and B. D. Noble, "Mobile Device Security Using Transient Authentication," IEEETransactions on Mobile Computing, vol. 5, no. 11, pp. 1489-
1502, Nov., 2006.

[24] S. Rafaeli and D. Hutchison, "A Survey of Key Management for Secure Group Communication," ACM ComputingSurveys, Vol. 35, Issue 3, Sept. 2003.

[25] D. Saha, A. Mukherjee, "Pervasive Computing: A Paradigm for the 21st Century," IEEE Computer, IEEE ComputerSociety Press, pp. 25-31, March 2003.

[26] P. E. Sevinc, M. Strasser, and D. Basin, "Securing the Distribution and Storage of Secrets with Trusted Platform Modules," Workshop in Information Security Theory and Practices (WISTP'07), Crete, Greece, May 8-11, 2007.

[27] A. Shamir, "Stream Ciphers: Dead or Alive?" invited talk, ASIACRYPT 2004, Jeju Island, Korea, Dec. 5-9, 2004.

[28] N. Wicaksono, "Connecting Windows Mobile with Vista in , 2007. New Ways", http://narn.my-sites.net

[29] E. Zenner, "Why IV Setup for Stream Ciphers is Difficult," in Proceedings of Dagstuhl Seminar on SymmetricCryptography, Jan. 2007.