Published in IET Information Security
Received on 6th December 2011
Revised on 22nd June 2012
doi: 10.1049/iet-ifs.2011.0347

ISSN 1751-8709

Mobile encryption for laptop data protection (MELP)

Y.-W. Kao[1,2] *X. Zhang*[1] *A. Studer*[1] *A. Perrig*[1]

[1]*Cylab, Carnegie Mellon University, Pittsburgh, PA, USA*
[2]*Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan*

Abstract: Based on the advances in laptop technologies and the mobility characteristics, laptops have become a vital device used at various places. Usually, numerous sensitive files such as credit card numbers and Web cookies are stored on laptops for convenient usage. However, if a laptop is stolen, the data stored on it is easily leaked; which may cause serious consequences. Encrypting files by encryption keys is a general solution; however, if the decryption keys are also stored on laptops, the files can also be decrypted by adversaries easily. To solve this problem, this paper proposes the Mobile Encryption for Laptop data Protection (MELP) system. MELP includes the design of an online server and mobile phone, and encrypts each sensitive file by a file system encryption key, which is further sequentially encrypted twice by the phone's and server's encryption keys. The reason of adopting a mobile phone is that at least one simple confirmation of execution must be performed by a user, and the reason of adopting an online server is that if both user's laptop and mobile phone are stolen, users can still disable the online decryption process on the server.

1 Introduction

As a benefit from the high-speed computation, large storage and mobility characteristics of laptops, numerous people tend to store their sensitive data on personal laptops at various places. Examples of sensitive data include (i) browser cookies that remember the login credentials or even credit card numbers for financial, shopping and social network sites, (ii) personal email stored in e-mail client applications, (iii) chat histories stored in instant messaging applications and (iv) personal photos. Moreover, enterprise employees or managers usually store more sensitive business secrets on their laptops such as financial, operational records or trade secrets of their companies. Owing to the mobility characteristic, laptops are easily stolen or lost. Given the vast amount of secrets stored on personal laptops, data leakage because of lost or stolen laptops constitutes a major threat to both individuals and enterprises [1–3]. The destructive consequence of the exposure of confidential data because of lost/stolen laptops warrants the pursuit of a secure and easy-to-use solution to protect the privacy and secrecy of laptop data. The scheme must be resilient to dictionary attacks and password recovery. Right after the loss of a laptop, the secret files on the laptop must be automatically encrypted without user effort and instantly become inaccessible to a thief. Moreover, even if both the laptop and key-storing device are stolen, the user must still be able to disable the process of laptop data decryption.

Regrettably, existing solutions for protecting laptop data privacy and secrecy have several drawbacks. For example, the password-based file protection or disk encryption incurs poor security since users tend to choose low-entropy passwords (e.g. '123456') that can be easily reconstructed via dictionary attacks. Furthermore, the USB key-based data encryption mechanism causes the usability problem because the user must bring one more additional hardware equipment for each laptop. The recently proposed mobile user location-specific encryption (MULE) [4] system automatically decrypts the sensitive laptop data only at a pre-specified trusted location, and transparently re-encrypts the data outside the trusted location. Data decryption key is automatically transmitted by the device deployed at this location so that the data can be decrypted without passwords entered. However, MULE still hampers usability because it restricts the use of sensitive data only at a specific trusted location, whereas at other locations the users must enter their passwords. Moreover, trusted platform module (TPM) hardware is required to be supported on the laptop in MULE; however, TPM hardware has not been supported by most laptops. Another solution is the keypad [5] system, which encrypts laptop data and stores the decryption key on a Web server. Each time a file is being accessed, the laptop automatically triggers the Web server to transmit the corresponding decryption key back; therefore the Web server can exactly log which files have been accessed. However, once a laptop is stolen, the thief can still access the files via the automatic decryption process; in other words, keypad focused on auditing more than security.

This paper proposes the mobile encryption for laptop data protection (MELP) system, which utilises mobile phones for protecting the sensitive data on laptops. The critical data on a laptop is encrypted with a file system protection key, which is further encrypted twice with two encryption keys. These two decryption keys are stored on a mobile phone and an online server, respectively. Accessing the secret files on laptop

requires the decryption of file system protection key by both the two corresponding decryption keys stored on the smart phone and server; it is performed automatically with little user interference. After the file system protection key is decrypted, it is transmitted from mobile phone to laptop via a 'constrained communication channel' [e.g. via a two-dimensional (2D) bar code or Bluetooth]; this constrained channel can only be established within a short distance between mobile phone and laptop. In this manner, if the laptop is stolen but the mobile phone is not, the constrained communication channel is automatically broken.

This paper contains eight sections. In Section 2, we introduce several related works of MELP. The system assumptions of MELP are defined in Section 3. The overview and details of MELP are described in Sections 4 and 5, respectively. We discuss the security analysis in Section 6. System implementation issues and evaluations are presented in Section 7. Finally, we present our conclusion and discuss future work in Section 8.

2 Related work

2.1 Stolen laptop problem

As people tend to store sensitive data on laptops for convenience, secrets are possible to be leaked if laptops are stolen. Encrypting files or the file system of laptop is a common solution to solve the stolen laptop problem [1–3]. In general, the encryption keys are generated based on two kinds of information: user password [6–8] and large random value [9–11]. In the case of using passwords provided by users, since these passwords are usually selected to be remembered easily, they are easy to be reconstructed by dictionary attack. By contrast, in the case of using a large random value, since it is too complicated to be remembered, it must be maintained by a system or stored on a digital storage. Studer and Perrig [4] proposed the MULE system, which encrypted and decrypted laptop data based on the location factor. In MULE, the decryption key is transferred from a trusted location device deployed at a trusted location through a constrained channel, such as infrared (IR) LED. If the laptop is not located at the trusted location, it cannot receive the key since the constrained channel cannot be established; thus the decryption process fails. However, in this manner, users can only access their secret data at some specific locations. In order to support mobile usage, the MULE system also protects the decryption key with the TPM hardware equipped on laptop. However, since TPM hardware has not been supported by most laptops until now, this solution is not applicable for most users. Corner and Noble [12] proposed the zero-interaction authentication (ZIA) system, which encapsulated the decryption key into a token and stored this token onto a wearable device such as IBM Linux watch. However, this kind of wearable device also has not been popular until now.

2.2 Device paring and constrained channel

In systems such as ZIA [12], the laptop is usually required to be paired or bound to a mobile device for establishing a communication channel. Wireless technologies such as IEEE 802.11 or Bluetooth are usually adopted so that the communication channel can be constructed automatically without user's intervention. Both the MULE and ZIA systems emphasised that this channel must be a short-range or constrained channel for preventing the transmitted information from being eavesdropped. Constrained channels can be constructed by various technologies with distinct properties. IR LED is not supported by most mobile devices; thus, it is not applicable for the mobile scenario. Bluetooth and other wireless technologies provide the capability of zero user intervention, but in this manner, the constrained channel can be established without owner's awareness. Near-field communication (NFC) [13–15] is a new technology, which enables mobile devices to establish a short-range wireless and contactless communication channel. If NFC is deployed on mobile phones, users can also use it for conducting authentication based on the SIM cards. The wireless channel established based on NFC is a very secure constrained channel, because that it provides a very short range (<0.2 m) of communication; in the case of laptop protection, the attacker must be very close to the laptop to access this channel if the laptop has not been stolen, it is easily noted by the laptop's owner. However, most laptops have not supported NFC yet until now.

2.3 2D barcode and quick response code

Compared with 1D barcode, 2D barcode [16–18] is able to store more information within an image for information exchange. The channel between 2D barcode displayer and scanner is a constrained channel, since that if images are taken with longer distance, they become smaller, more ambiguous and more difficult to be decoded correctly. This kind of constrained channel requires users to perform a simple action: locate the scanner and displayer together. In this manner, the establishment of this channel can be confirmed by users with little user intervention. Quick response (QR) code [19–21] is a kind of 2D barcode proposed in 1994; it is designed for encoding and decoding data between text contents and images rapidly. QR code is an easily used technology to store information such as URL of Web site, phone number and e-mail address. In numeric format, the max size of QR code message can be up to 7089 characters; in binary format, the maximum amount is 2953 bytes. QR code also supports error correction for fault tolerance. Fault tolerance is a crucial feature since that perfect photographing cannot be guaranteed. With the fault tolerance feature, users do not have to try multiple times for decoding the captured images with few errors. In recent years, since the QR code technology has been supported by numerous mobile phones such as Android phones, it has been widely adopted in various mobile applications, and most mobile users are already familiar with using it.

2.4 TPM-based data protection

TPM [22–24] is a hardware device or specification, which offers multiple security facilities such as secures the generation of cryptographic keys, secures the usage of these keys and generates of hardware-based pseudo-random number. TPM can protect sensitive data on laptops simply by sealing the data with a non-migratable TPM storage key. However, this solution is not feasible so far because TPM chip has not been supported by most laptops. In addition, another problem of TPM hardware is that it only protects the cryptography keys, but not the secret data directly. If secret data are decrypted and stored into memory temporarily, it may be stolen by a malicious program. To solve this problem, our previous work, TrustVisor [25], was proposed. TrustVisor leveraged the hypervisor [26–28] and TPM technologies to provide code integrity, data integrity

doi: 10.1049/iet-ifs.2011.0347

and data secrecy for selected portions of an application. In this manner, the execution of using secret data can be isolated from malware.

3 System assumptions

To define the scope of MELP, several assumptions are defined. Four main parties are included in MELP: mobile phone, laptop, TPM server and flash drive. One user account can be associated with only one mobile phone but multiple laptops. On each laptop, the secret files are encrypted by a symmetric encryption key; attackers cannot access the secret files without possessing the file system root key. To make MELP applicable on most of the laptop products, MELP does not rely on the TPM mechanism on laptops; therefore we do not assume that TPM hardware is supported on the laptop. In addition, the laptop is possible to be stolen, but no malware is installed on it before it is stolen; otherwise, the attacker can easily obtain the file system root key when it is decrypted by the user. On the TPM server side, we assume that the OS of TPM server is compromisable, but the TPM hardware, the hypervisor and the software module which is protected by both the TPM and hypervisor, is not. The security issues of TPM hardware, hypervisor and the software module protected by them, are out of scope of this paper. Moreover, the organisation which maintains the TPM server must already have a PKI, $\langle EK_\tau, DK_\tau \rangle$, to use, and the public key EK_τ is included directly into the laptop and mobile phone applications. Since numerous researches [29–31] have proposed various revocation mechanisms for PKI, such as creating certificates via CA, this paper does not focus on this issue. On the mobile phone side, we assume that the mobile phone is possible to be stolen, but it is more difficult to steal both laptop and phone than only the laptop. Also, no malware is installed on the mobile phone before it is stolen; otherwise, the attacker can obtain the phone decryption key without stealing the phone. In addition, users can choose an optional backup mechanism by using their flash drives and flash drive passwords in case that they forget to bring their phones or want to access secret files offline. In this case, the flash drive maintains copies of phone decryption key as well as server decryption key, and plays the role as a representative of both mobile phone and TPM server. We assume that the flash drive may be stolen, but the thief does not know the flash drive password. However, this design provides more usability than security, because the thief may perform dictionary attack to discover the flash drive password. Therefore users have to take the risk if they really want to create backups on flash drives. Finally, we assume that RSA is not compromisable, and use RSA in most encryption and decryption operations of MELP.

4 Overview

The overview of MELP is shown in Fig. 1. In this figure, the golden key is a symmetric key; the red, blue and black keys represent the private key parts of asymmetric key pairs, and the red, blue and black locks represent the public key parts of asymmetric key pairs. To protect secret files on laptop, the golden file system root key is generated by the laptop application to encrypt these files. The file system root key is further encrypted by the black phone lock and blue server lock sequentially, and the original copy of file system root key is deleted thereafter. Therefore in the online decryption scenario, only when all the parties of laptop, mobile phone and TPM server (or their representatives) participate in the decryption process, the secret files on laptop can be accessed. The black and blue key-lock pairs are initially generated and distributed by the mobile application during the phone registration process.

Whenever the user wants to access the secret files on laptop, the laptop application sends the twice-encrypted result of golden key blinded by a random blinding factor to TPM server for the first-round decryption. After receiving the first-round decryption result, the laptop application blinds it with another blinding factor, sends the result to TPM server again, and asks the user to execute mobile application for second-round decryption. Then, the phone application acquires the secondly blinded result five from TPM server and decrypts it by phone decryption key. Finally, the phone application sends back the decryption result to the laptop application via a constrained channel. In this manner, user can access the secret files based on the existence of the registered laptop and mobile phone, online service of TPM server and constrained channel, with little user intervention. In other words, if any one of these factors is absent, the secret files cannot be stolen.

In addition to the basic decryption scenario, users can choose an optional backup mechanism to access secret files offline. The backup on flash drive is created by acquiring the phone decryption key as well as server decryption key and then encrypting them by the yellow flash drive password as well as the red laptop decryption key. In other words, only the person who possesses the flash drive, the laptop and the flash drive password can access the secret files stored on laptop.

5 MELP details

5.1 Initialisation phase

During the deployment of MELP, several initialisation steps must be conducted. First of all, the MELP TPM server must select a K_σ, which is a symmetric key for protecting all the secrets on it. All the functionalities of the TPM server must be implemented within a software module, SM, which is protected by a hypervisor, HP, which is further protected by the TPM hardware. The SM maintains a public/private key pair $\langle EK_\tau, DK_\tau \rangle$ for protecting messages in communications established with mobile phones or laptops, and then encrypts the secret DK_τ by K_σ, into $Etpm = Ea(EK_\tau, K_\sigma)$; here the function Ea is an asymmetric encryption function. The public key EK_τ is included into the distributions of mobile applications and laptop applications, so that these applications can establish secure channels with the TPM server. Finally, the SM encrypts the K_σ with the assistance of HP by executing the binding function of TPM, and then deletes DK_τ as well as K_σ.

On the user side, users must download the mobile applications and laptop applications from the Android market and trusted Web sites, respectively. At the first time when the laptop application is executed, a symmetric key K_{file} is generated for protecting the data which users want to protect. Also, a pair of public/private keys $\langle EK_\lambda, DK_\lambda \rangle$ of laptop is created for protecting and signing messages. At the first time when the mobile application is executed, two pairs of public/private keys $\langle EK_\sigma, DK_\sigma \rangle$ and $\langle EK_\mu, DK_\mu \rangle$, which are designed to be maintained by the TPM server and the user's mobile phone, respectively, are generated for protecting the K_{file} on laptop. Each key in these two pairs are composed of the exponent part and the modulus part, so

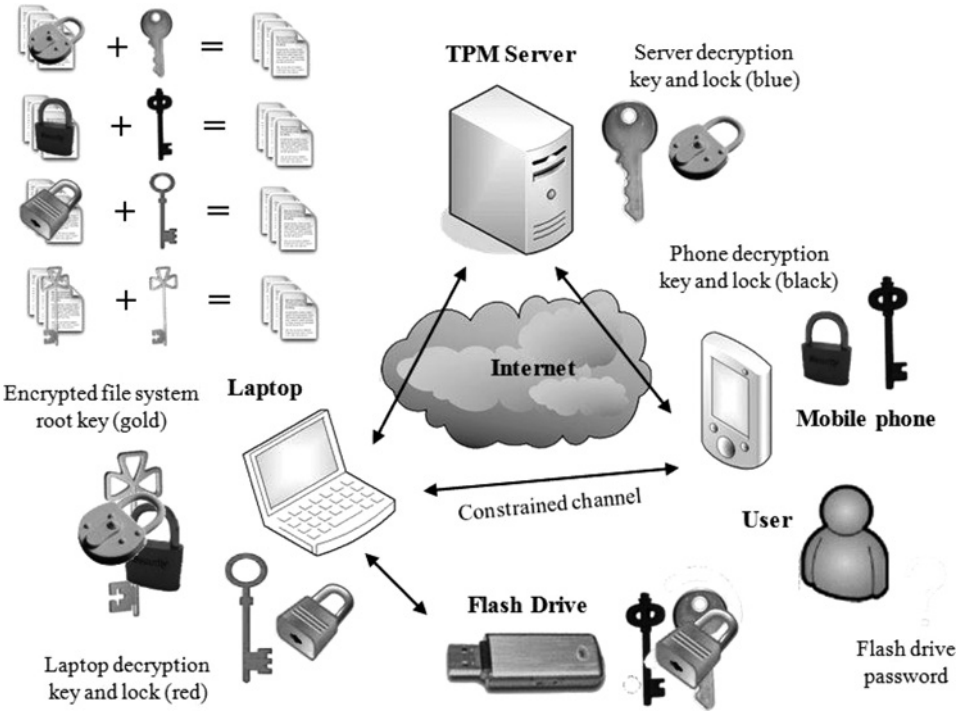

Fig. 1 *MELP overview*

that $EK_\sigma = \langle es, Ns \rangle$, $DK_\sigma = \langle ds, Ns \rangle$, $EK_\mu = \langle em, Nm \rangle$ and $DK_\mu = \langle dm, Nm \rangle$, where the Nm must be smaller than Ns.

5.2 Registration phase

Two kinds of devices are owned by users: mobile phone and laptop, so users must register them first before encrypting or decrypting data on laptop. The registration process is divided into two stages: mobile phone registration and laptop registration, which are shown in Figs. 2 and 3, respectively. Since we assume that one account can be associated to only one mobile phone but multiple laptops, users can execute the mobile phone registration once but the laptop registration for multiple times for each of their accounts.

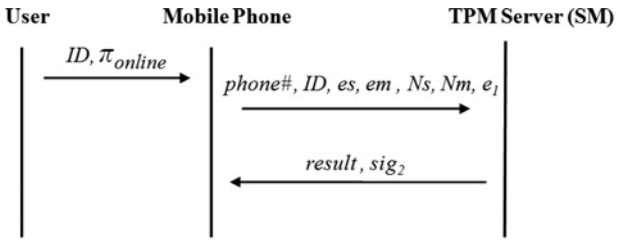

Fig. 2 *Mobile phone registration*

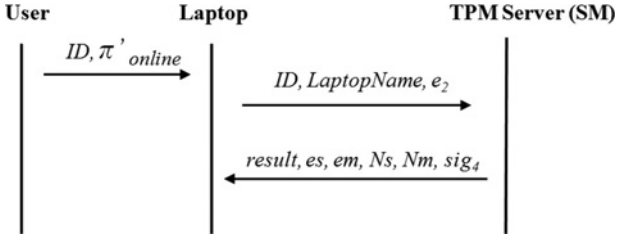

Fig. 3 *Laptop registration*

5.2.1 Mobile phone registration: In the beginning of mobile phone registration, the mobile application receives the input of ID (the user's account) and π_{online} (the user's online password) from user. Then, the mobile application generates a registration message m_1 and sends it to the SM on TPM server.

$$e_1 = Ea(R_1 \| h(\pi_{\text{online}}) \| ds \| (\text{Ext}(h(\pi_{\text{online}})) \oplus dm), EK_\tau)$$
$$m_1 = \text{phone\#} \| \text{ID} \| es \| em \| Ns \| Nm \| e_1$$

In m_1, R_1 is a random nonce, which makes the encrypted result diverse each time for preventing replay attack, h is a hash function, which prevents the π_{online} from being exposed and the function Ext is an extension function, which ensures that the length of extended hashed result equals to the length of dm. If users do not want to delegate their mobile phone decryption keys dm to TPM server, they can choose not to calculate the $\text{Ext}(h(\pi_{\text{online}})) \oplus dm$ and include it into e_1. After sending m_1, the mobile application deletes ds, π_{online}, $h(\pi_{\text{online}})$ and $\text{Ext}(h(\pi_{\text{online}})) \oplus \text{dm}$.

Once the SM receives the registration message m_1, it first checks that whether the ID has already been used. If the ID is unused, the SM decrypts e_1 by decrypting K_σ and $Etpm$. The SM encrypts several secrets of user into $E\pi$, Eds and Edm for this ID by the symmetric encryption function Es with K_σ. $E\pi$ is designed for user authentication, Eds is for server-side online decryption and Edm is for delegated phone-side decryption or the recovery operation of mobile phone decryption key

$$E\pi = Es(h(h(\pi_{\text{online}})), K_\sigma)$$
$$Eds = Es(ds||Ns, K_\sigma)$$
$$Edm = Es(\text{Ext}(h(\pi_{\text{online}})) \oplus dm, K_\sigma)$$

Finally, the SM sends back the m_2 message, which includes the registration result and the signature of result. The

doi: 10.1049/iet-ifs.2011.0347

signature is also used for server authentication; only the server which has the key (DK_τ) can sign the result correctly

$$\text{sig}_2 = Da(R_1 \| \text{result}, DK_\tau)$$
$$m_2 = \text{result} || \text{sig}_2$$

5.2.2 Laptop registration: After registering the mobile phones with accounts and online passwords, users can register their laptops by using the same accounts and passwords. First of all, the laptop application sends a laptop registration message m_3, to the SM according to the ID and π'_{online} input by user

$$e_2 = Ea(R_2 \| EK_\lambda \| h(\pi'_{\text{online}}), EK_\tau)$$
$$m_3 = \text{ID} \| \text{Laptop name} \| e_2$$

After receiving m_3, the SM extracts EK_λ and $h(\pi'_{\text{online}})$ by decrypting K_σ and $Etpm$. By comparing $h(h(\pi'_{\text{online}}))$ with the $h(h(\pi_{\text{online}}))$ decrypted from $E\pi$, this message can be authenticated; if authentication fails, the SM delays retries of this ID. By contrast, if authentication is successful, the SM sends an m_4 message, which includes the result and encryption keys, to the laptop application

$$d_4 = \text{result} || es || em || Ns || Nm$$
$$\text{sig}_4 = Da(h(d_4), DK_\tau)$$
$$m_4 = d_4 || \text{sig}_4$$

After the laptop registration is successful, the laptop application can use the encryption keys in m_4 to encrypt K_{file} into $e_3 = (K_{\text{file}}^{es-em} \text{mod Nm}) \text{ mod } Ns$, and then delete K_{file}. The K_{file} must be deleted completely and all the bits of it must be set to 0 before it is deleted no matter it is stored on disk, RAM or cache to ensure that the attacker cannot obtain the key in any manner if the laptop is stolen.

5.3 Normal online decryption phase

Fig. 4 shows the normal decryption process. Once the laptop application requires K_{file} for decrypting files or encrypting new files, it must decrypt e_3 into K_{file} first. To do so, e_3 must be blinded by a random number R_3 in message m_5, and sent to the SM.

$$e_4 = (R_3^{es} (K_{\text{file}}^{es-em}) \text{mod} Nm) \text{mod } Ns$$
$$d_5 = \text{ID} \| \text{Laptop name} \| e_4$$
$$\text{sig}_5 = Da(h(d_5), DK_\lambda)$$
$$m_5 = d_5 \| \text{sig}_5$$

After receiving the decryption message m_5, the SM verifies the signature of laptop application, decrypts the corresponding Eds into DK_σ, and creates e_5 by signing e_4 with the signing key DK_σ. Thereafter, the SM sends back a reply message m_6 to the laptop application

$$e_5 = (R_3^{es-ds} (K_{\text{file}}^{es-em}) \text{mod } Nm) \text{mod } Ns$$
$$\text{sig}_6 = Da(h(e_5), DK_\lambda)$$
$$m_6 = e_5 \| \text{sig}_6$$

After receiving the decryption result coming from SM, the laptop application unblinds e_5 into $K_{\text{file}}^{es-em-ds} \text{mod } Nm$, blinds the result with another random number R_4 to create e_6, and sends the message m_9 to SM

$$e_6 = R_4^{em} K_{\text{file}}^{es-em-ds} \text{mod } Nm$$
$$d_7 = \text{ID} \| \text{Laptop name} \| e_6$$
$$\text{sig}_7 = Da(h(d_7), DK_\lambda)$$
$$m_7 = d_7 || \text{sig}_7$$

At this time, the laptop application asks the users to execute their mobile applications and establishes a constrained channel with them. After the mobile applications are executed, they check that whether the e_6 exits on the TPM server by sending a message m_8

$$e_7 = Ea(R_5 \| h(\pi'_{\text{online}}), EK_\tau)$$
$$d_8 = \text{ID} || e_7$$
$$\text{sig}_8 = Da(h(d_8), DK_\tau)$$
$$m_8 = d_8 || \text{sig}_8$$

If e_6 exists, this means that an ongoing decryption process has not been finished, then the SM forwards e_6 by sending a message m_9 to the mobile application

$$d_9 = \text{Laptop name} || e_6$$
$$\text{sig}_9 = Da(h(d_9), DK_\tau)$$
$$m_9 = d_9 || \text{sig}_9$$

Finally, the mobile application signs e_6, and sends the result m_{10} to the laptop application via a constrained channel, so that the laptop application can unblind m_{10} and obtain the decrypted K_{file} for decrypting or encrypting local files later

$$m_{10} = R_4^{em-dm} K_{\text{file}}^{es-em-ds-dm} \text{mod } Nm = (R_4 \times K_{\text{file}}) \text{mod } Nm$$

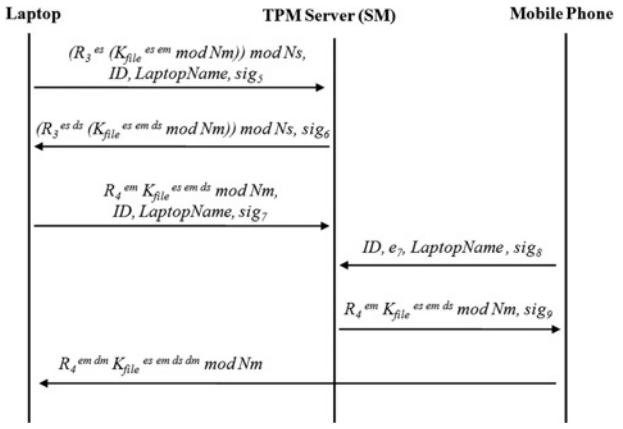

Fig. 4 *Normal online decryption*

Fig. 5 *QR code constrained channel construction*

6 Security analysis

Four parties are included in MELP: mobile phone (M), laptop (L), TPM server (T) and flash drive (F). If any one of {M, L, T, F} is compromised or stolen, the secret files are not disclosed: without possessing laptop, there is no encrypted data to be decrypted; if only laptop is stolen, secret files cannot be decrypted without decryption keys or the online password for accessing the keys. If the attacker tries to use dictionary attack to guess online password, the TPM server performs the delay operation, which is protected by the hypervisor and TPM hardware, and make it infeasible. Similarly, if only {M, T}, {M, F} or {T, F} are controlled, the encrypted secret files are not available. If only {L, F} are stolen, the file cannot be decrypted without knowing the flash drive password. However, if the thief conduct dictionary attack to decrypt the backup on flash drive, the secret files are possible to be disclosed. If only {L, T} are stolen or compromised, the files are still protected by the encryption of phone encryption key. If {M, L} are both stolen, the secret files can be disclosed, because the TPM server responses to requests automatically. However, since mobile phones and laptops are usually vital to people, users can find that they are stolen in a very short period of time, and disable the online decryption service as soon as possible. Also, it is more difficult to steal two devices than only one of them at the same time. As the analysis of three of these parties, the {M, L, T} case is the same as {M, L}, {M, L, F} is the same as {M, L} as well as {M, F}, {M, T, F} is the same as {M}, and {L, T, F} is the same as {L, T} as well as {L, F}. Finally, the {M, L, T, F} case is the same as {M, L} and {L, F}.

During the mobile phone registration phase, messages are basically encrypted by R_1 and EK_μ, which are further protected by EK_τ; therefore attackers cannot obtain the secrets such as ds without knowing DK_τ. During the laptop registration phase, except the hashed online password, which is protected by EK_τ, there are no secrets exchanged. In the normal decryption messages, the key K_{file} is always encrypted before the decryption operation on mobile phone. Although it is decrypted and transmitted to the laptop, it is blinded by a blinding factor R_5 and transmitted via a constrained channel, which is not easy to be accessed.

It concludes there are only two cases that the attacker can access the secret files. The first manner is to steal both the laptop and mobile phone, and decrypt the files before the user disable the online decryption service; it is challenging

Fig. 6 *Screenshots of laptop application*

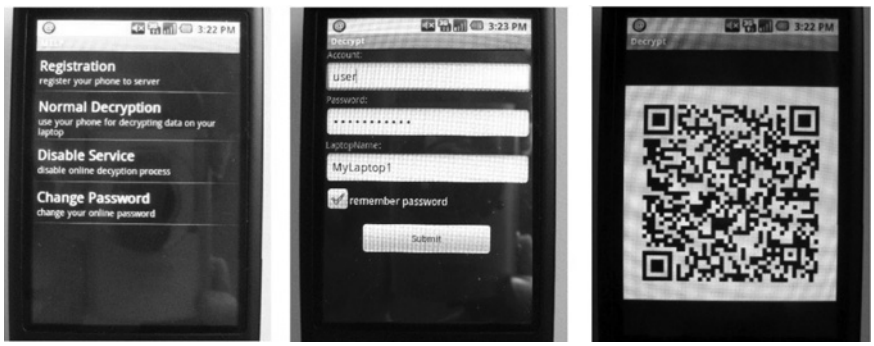

Fig. 7 *Screenshots of mobile application*

doi: 10.1049/iet-ifs.2011.0347

Table 1 Performance of mobile phone registration

	Average consumed time, ms	Standard deviation, σ
calculating m_1	122.97	1.17
transmitting m_1	1.84	0.05
calculating m_2	691.91	5.28
transmitting m_2	62.59	27.96
total	879.31	22.91

Table 2 Performance of laptop registration

	Average consumed time, ms	Standard deviation, σ
calculating m_3	115.29	1.23
transmitting m_3	1.10	0.07
calculating m_4	521.57	12.77
transmitting m_4	0.88	0.05
total	638.81	12.06

to attackers. The other manner is to steal both the laptop and the backup on flash drive, and then conduct dictionary attack to decrypt the files. Apparently, the former case is more secure than the latter one. Although the users can store the backups on flash keys, and protect it as normal physical keys, they cannot do anything if both the laptop and the backups are stolen. Therefore the backup on flash drive is only an optional choice for users if they care about usability more than security, and users have the responsibility of protecting it.

7 System implementation and evaluation

7.1 System implementation

Based on the proposed system design, a system prototype of MELP is implemented. The implementation details are described as follows. First, the TPM server is implemented in C++ and C, and deployed on the Ubuntu 10.10 OS with Intel 2.4 GHz CPU and 4 GB RAM. Second, the laptop application is implemented in Java, C++ and C, and deployed on the Ubuntu 10.10 OS with Intel 2.5 GHz CPU and 4 GB RAM. Moreover, the laptop secret file encryption mechanism is implemented by the EncFS [32] which encrypts the file system with a file system encryption key K_{file}. Third, the mobile application is implemented in Java and deployed on the HTC G1 with Android 1.6 Donut OS. Finally, the constrained channel for sending data from mobile application to laptop application is implemented by displaying an L-mode QR code image and capturing this image from the camera of laptop. Fig. 5 shows the construction of QR code constrained channel. Furthermore, Figs. 6 and 7 show the screenshots of laptop application and mobile application, respectively.

Table 3 Performance of normal decryption

	Average consumed time, ms	Standard deviation, σ
calculating m_5	116.48	2.84
transmitting m_5	1.07	0.09
calculating m_6	467.85	9.44
transmitting m_6	0.90	0.13
calculating m_7	29.60	0.16
transmitting m_7	71.55	6.27
calculating m_8	355.56	5.25
transmitting m_8	876.45	39.51
calculating m_9	358.44	4.41
transmitting m_9	981.45	287.59
displaying m_{10}	1710.44	148.63
handling m_{10}	36.54	0.99
total	5006.30	193.24

Table 4 System comparison

	Restricted location	Adopting constrained communication channel	Using popular client device
MULE [4]	yes	yes (IR LED)	no
ZIA [12]	no	not specified	no
NFC [13–15]	no	yes (RFID)	no
MELP	no	yes (QR code image)	yes

7.2 Performance evaluation

Tables 1–3 evaluate and list the performances of mobile phone registration phase, laptop registration phase and normal decryption phase of MELP, respectively. In the normal decryption phase, the operation of placing the mobile phone in front of laptop's camera is not included, since it highly depends on user's behaviour. In Table 3, the operation that takes the most of time is displaying an encoded QR code on mobile phone, which is implemented by using the ZXing project [33].

Tables 1 and 2 show that the mobile phone and laptop registration process are very efficient; each of them requires only less than one second for registration. In the normal decryption phase, although the entire process costs about 5 s in average, the ZXing [33] execution spends one-third of the total time. Also, spending 5 s is still acceptable for sensitive data protection.

7.3 System comparison

The comparison of MELP and many related technologies mentioned in Section 2 are shown in Table 4. The MULE [4] system focused on data decryption at trusted locations, therefore users cannot use it anywhere. Moreover, the ZIA system focused on the data encryption and decryption scheme rather than the establishment of secure channel. Finally, since all the other solutions require the specific client device or hardware module, only MELP is applicable to be executed on popular laptops and mobile phones until now.

8 Conclusion and future work

In conclusion, laptop theft is a serious problem. Even if the sensitive data stored on a laptop is protected by a password, most passwords are relatively easy to be discovered by dictionary attacks. Although data can be encrypted by users, the leakage problem still exists if the decryption key is also stored on laptop. To solve this problem, MELP includes the user's mobile phone to ensure that users are aware of the construction of constrained channel during decryption

processes. Also, a TPM server, which is protected by a hypervisor and TPM hardware, is adopted so that if both the laptop and mobile phone are stolen, users can still disable the online decryption process on the server. Moreover, a constrained channel implemented by displaying and capturing a QR code is designed so that users can decrypt their files conveniently and securely. Finally, the security of MELP is analyzed and the performance of it is evaluated.

As NFC technology has not been supported by laptops, we currently adopt QR code images and cameras to implement the constrained channel. In the future, we will replace the QR code with NFC technology once NFC is supported on most laptops or even PCs. In this manner, the constrained channel can be constructed more easily and the usability would be improved.

9 References

1 Foster, A.L.: 'Increase in stolen laptops endangers data security', *Chronicle High. Educ.*, 2008, **54**, (43)
2 Wyld, D.C.: 'Help! someone stole my laptop!: how RFID technology can be used to counter the growing threat of lost laptops', *J. Appl. Secur. Res.*, 2009, **4**, (3), pp. 363–373
3 Wyld, D.C.: 'Preventing the "worst case scenario": combating the lost laptop epidemic with rfid technology', Novel algorithms and techniques in telecommunications and networking, pp. 29–33
4 Studer, A., Perrig, A.: 'Mobile user location-specific encryption (MULE): using your office as your password'. Proc. ACM Conf. Wireless Network Security (WiSec), March 2010, pp. 151–162
5 Geambasu, R., John, J.P., Gribble, S.D., Kohno, T., Levy, H.M.: 'Keypad: an auditing file system for theft-prone devices'. Proc. European Conf. Computer Systems (EuroSys), 2011, pp. 1–16
6 Morris, R., Thompson, K.: 'Password security: a case history', *Commun. ACM*, 1979, **22**, (11), pp. 594–597
7 Seeley, D.: 'Password cracking: a game of wits', *Commun. ACM*, 1989, **32**, (6), pp. 700–703
8 Gait, J.: 'Easy entry: the password encryption problem', *ACM SIGOPS Oper. Syst. Rev.*, 1978, **12**, (3), pp. 54–60
9 Rivest, R.L., Shamir, A., Adleman, L.: 'A method for obtaining digital signatures and public-key cryptosystems', *Commun. ACM*, 1978, **21**, (2), pp. 120–126
10 Lai, X., Massey, J.L.: 'A proposal for a new block encryption standard' (*LNCS*, **437**), 2006, pp. 389–404
11 Tatebayashi, M., Matsuzaki, N., Newman, D.B.: 'Key distribution protocol for digital mobile communication systems', in Brassard, G. (Ed.): CRYPTO 1989, (*LNCS*, **435**), 1990, pp. 324–334
12 Corner, M.D., Noble, B.D.: 'Zero-interaction authentication'. Proc. Eighth Annual Int. Conf. Mobile Computing and Networking (MobiCom '02), 2002, pp. 1–11
13 Finkenzeller, K.: 'RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication' (Wiley & Sons, 2010, 3rd edn.), ISBN-10: 0470695064, ISBN-13: 978-0470695067
14 Haselsteiner, E., Breitfuss, K.: 'Security in near field communication (NFC) strengths and weaknesses'. Workshop on RFID Security, 2006, pp. 1–11
15 Michahelles, F., Thiesse, F., Schmidt, A., Williams, J.R.: 'Pervasive RFID and near field communication technology', *IEEE Pervasive Comput.*, 2007, **6**, (3), pp. 94–96
16 Ottaviani, E., Pava, A., Bottazi, M., Bruncllli, E., Casclli, F., Guerreo, M.: 'A common image processing framework for 2D barcode reading'. Seventh Int. Conf. Image Processing and its Applications, 1999, pp. 652–655
17 Gao, J.Z., Prakash, L., Jagatesan, R.: 'Understanding 2D-BarCode technology and applications in M-commerce – design and implementation of a 2D barcode processing solution'. Proc. 31st Int. Conf. Annual Int. Computer Software and Applications, 2007, pp. 49–56
18 Kato, H., Tan, K.T.: '2D barcodes for mobile phones'. Proc. Second Int. Conf. Mobile Technology, Applications and Systems, 2005, p. 8
19 Seino, K., Kuwabara, S., Mikami, S., *et al.*: 'Development of the traceability system which secures the safety of fishery products using the QR code and a digital signature'. Proc. MTS/IEEE TECHNO-OCEAN, Kobe, 2004, vol. 1, pp. 476–481
20 Ohbuchi, E., Hanaizumi, H., Hock, L.A.: 'Barcode readers using the camera device in mobile phones'. IEEE Int. Conf. Cyberworlds (CW04), 2004, pp. 260–265
21 Chaisatien, P., Akahori, K.: 'Introducing QR code in classroom management and communication via mobile phone application system'. Proc. World Conf. Educational Multimedia, Hypermedia and Telecommunications, 2006, pp. 2181–2187
22 Bajikar, S.: 'Trusted platform module (TPM) based security on notebook PCs-white paper' (Mobile Platforms Group Intel Corporation, 2002), pp. 1–20
23 Strasser, M., Stamer, H.: 'A software-based trusted platform module emulator', 2008, (*LNCS*, **4968**), pp. 33–47
24 Aaraj, N., Raghunathan, A., Ravi, S., Jha, N.K.: 'Energy and execution time analysis of a software-based trusted platform module', in Proc. of 2007 Design, Automation & Test in Europe Conference & Exhibition (DATE '07), 2007, pp. 1128–1133
25 McCune, J.M., Li, Y., *et al.*: 'TrustVisor: efficient TCB reduction and attestation'. Proc. IEEE Symp. Security and Privacy (Oakland 2010), 2010, pp. 143–158
26 Bressoud, T.C., Schneider, F.B.: 'Hypervisor-based fault-tolerance'. Proc. Symp. on Operating Systems Principles, 1995, pp. 1–11
27 Sailer, R., Jaeger, T., Valdez, E., *et al.*: 'Building a MAC-based security architecture for the xen open-source hypervisor'. Proc. 2005 Annual Computer Security Applications Conf., 2005, pp. 276–285
28 Seshadri, A., Luk, M., Qu, N., Perrig, A.: 'SecVisor: a tiny hypervisor to provide lifetime Kernel code integrity for commodity OSes'. ACM Symp. Operating Systems Principles (SOSP), 2007, pp. 335–350
29 Arnes, A., Knapskog, S.J.: 'Selecting revocation solutions for PKI'. Proc. NORDSEC 2000, Fifth Nordic Workshop on Secure IT Systems, 2000, pp. 1–16
30 Khurana, H., Gligor, V.D.: 'Review and revocation of access privileges distributed with PKI certificates'. Proc. Security Protocols Workshop, 2002, pp. 100–112
31 Critchlow, D., Zhang, N.: 'Revocation invocation for accountable anonymous PKI certificate trees'. The nineth IEEE Symp. on Computers and Communications (ISCC'2004), 2004, pp. 386–392
32 Gough, V.: The EncFS, available at http://www.arg0.net/encfs, 2011
33 Xing, Z.: available at http:// www.code.google.com/p/zxing/, 2011

doi: 10.1049/iet-ifs.2011.0347