Computational Resources for Mobile E-Wallet System with Observers

Eligijus Sakalauskas[1], Jonas Muleravicius[1], Inga Timofejeva[1]
[1]*Kaunas University of Technology, Department of Applied Mathematics,*
Studentu St. 50, LT-51368, Kaunas
eligijus.sakalauskas@ktu.lt

***Abstract*—The next generation of individual payment systems that will replace traditional smart cards will be mobile devices supplied with e-wallet functions. The spread of such e-wallet systems will depend on their security, functionality, ease of use and the effectiveness of realization. This paper proposes an effective realization of e-wallet system, providing e-cash divisibility, off-line payment and deposit options as well as purchaser's anonymity against the vendor and satisfying general electronic payment security requirements of double spending prevention, unforgeability as well as unclonability of purchaser's identity. The latter is achieved through the implementation of bank's tamper resistant hardware agent, known as observer, provided with physically unclonable function (PUF), capable of yielding a unique unclonable code (UUC), in user's mobile device. The estimation of the necessary computational resources in the usage of the proposed mobile e-wallet system is presented in order to determine whether the proposal is worth the computation cost.**

***Index Terms*—Cryptography, E-wallet, E-wallet system, PUF.**

I. Introduction

The electronic wallet, i.e. the consumer device designed to store and manage electronic cash (e-cash), has received much attention lately. However, most current e-wallet systems lack an important e-cash property, namely, the off-line e-cash option which could provide consumers with an increased autonomy since they could make payments or manage their electronic funds without the necessity of on-line connection to the bank or any other trusted third party. One of the possible ways of ensuring the described e-cash property is the implementation of a tamper resistant bank's hardware agent, known as observer, which would provide bank with the full control of e-money circulation. The main idea is to embed such observers in customers' mobile devices in a similar manner to providing customers with SIM cards. Assuming the suitability of integrated cryptographic functions, such novelty could ensure the main security and functional requirements formulated for e-money systems, such as e-money divisibility, validity, untraceability, anonymity, double spending prevention and off-line payment option. One of the steps in attempting to ensure the suitability of cryptographic functions is implementing the e-wallet system in observers with (at least) a password protected access to those functions. However, such kind of protection is not secure enough, since cryptographic credentials in observers can be cloned by physical-electronical examination as illustrated by the two following facts: 1) In 2008 C. Tarnovsky hacked TPM 1.2 chip [1]; 2) Afterwards, at Black Hat conference, in 2010 he hacked Infineon SLE 66 CL PE chip [2] – widely used in computers, identity cards, and other electronic devices. A better way of ensuring the suitability of cryptographic functions could be the use of secure next generation chips supplied with so called physically unclonable functions (PUF). One of the first ideas mentioned in literature involved implementing PUFs for device authentication [3]. In [4], authors described the PUF's capability of generating volatile secret keys for cryptographic operations, while using low amount of memory. Tiny variations in the semiconductor manufacturing process make each transistor and each piece of silicon unique as a fingerprint. Since such variations are random and uncontrollable, it is impossible to make an exact clone of a chip. PUFs use random patterns to differentiate chips from one another. Physical unclonable functions also allow user to extract a unique unclonable code (UUC) for the chip and to create a unique cryptographic key. The new technology of chips with PUF is under the development [3]. It is expected that PUFs will essentially improve the level of security of future e-money systems and, furthermore, will reveal the new opportunities for secure and customer friendly e-wallet systems. The ones introduced above are the main reasons for using PUFs in the construction of the proposed e-wallet implemented in purchaser's observer.

As presented above, e-wallet carries the meaning of a software and hardware system with divisible and secure e-cash used to store funds downloaded from a bank account or transfer certain amounts to other e-wallets in order to make purchases. In the general scheme, e-wallet is integrated in the entire e-money circulation system. The main actors of this system are Bank (***B***), Purchaser (***P***) and Vendor (***V***). We are assuming that ***P*** and ***V*** have their accounts in ***B***. ***B*** is issuing e-money and is responsible for its secure transfer to ***P***'s e-wallet. ***B*** or any other trusted third party (TTP) is also responsible for double spending prevention and dispute arbitration between ***P*** and ***V***. The agent performing ***B***'s

Manuscript received 13 December, 2016; accepted 27 March, 2017.

functions in e-wallet is Observer (***O***). Furthermore, we assume that ***P*** gets some amount of e-cash from ***B*** and obtained e-money is placed in his e-wallet incorporating ***O***. The portion of this e-cash can be withdrawn from e-wallet and paid to ***V*** in order to make a purchase. In our analysis we consider these two actions separately as withdrawal and payment protocols. We estimate the computation time required for withdrawal protocol that involves only ***O***'s computational resources as well as the computation time required to verify the validity of e-cash in payment protocol, where both ***P***'s and ***V***'s resources are used.

We propose an e-wallet system incorporating ***O*** supplied with PUF capable of yielding a unique and unclonable code (UUC). ***B*** associates this UUC with the ***P***'s identification number that we denote by Id_P. Proposed system provides ***P***'s anonymity against ***V***. Moreover, anonymity of the dishonest ***P*** can be revealed by ***B*** if double spending takes place. We consider e-cash as a banknote represented by the amount of e-cash as well as its unique number, which is signed with ***B***'s e-signature. This e-signature is formed in ***O*** where ***B***'s private key is implemented in protected memory. Untraceability and anonymity are achieved by introducing asymmetric encryption function employed for the encryption of ***P***'s Id_P. This function is also implemented in ***O*** and encryption is performed using ***B***'s public key. If the same e-cash is spent by ***P*** twice, then ***B*** can decrypt Id_P of dishonest ***P*** and take measures to punish one.

For the sake of illustration, we use ElGamal cryptosystem for asymmetric encryption and e-signing [5]. We present ElGamal encryption and signature schemes in the following Sections II and III.

II. ElGamal Asymmetric Encryption Scheme

Generate a prime p of order 2^{2048} and find a generator g of the multiplicative group $Z_p^* = \{1,\ldots,p-1\}$.

System parameters: $SP = (p, g)$.

Key generation:

Select a random integer $1 < x < p-1$.

Compute

$$a = g^x \bmod p. \tag{1}$$

Private and public keys: $PrK = (x); PuK = (a)$.

Encryption:

Let $m \in \{2,\ldots,p-2\}$ – be a message to be encrypted.

Select a random integer $k, 1 < k < p-1$.

Compute:

$$\begin{cases} E = ma^k \bmod p, \\ D = g^k \bmod p. \end{cases} \tag{2}$$

The ciphertext

$$c = Enc_{PuK}(m) = (E, D). \tag{3}$$

Decryption:

Compute

$$m = Dec_{PrK}(c) = Dec_{PrK}(E, D) = E \cdot D^{-x} \bmod p. \tag{4}$$

III. ElGamal Signature Scheme

Since the proposed e-cash system is exploiting e-signature scheme with appendix, all parties must agree on the hash function $H: \{0,1\}^* \to Z_p$. Hence, the hash function is added to the set of system parameters defined above, i.e. $SP = (p, g, H())$. For the e-signature the same key pair (see Section II) $PrK = (x); PuK = (a)$ is used.

Signature generation:

Let m be a message to be signed.

Compute

$$h = H(m). \tag{5}$$

Select a random secret integer $k, 1 < k < p-1$, with $\gcd(k, p-1) = 1$.

Compute:

$$\begin{cases} r = g^k \bmod p, \\ s = k^{-1}(h - xr) \bmod (p-1). \end{cases} \tag{6}$$

The signature for m is the pair (r, s).

Signature verification:

Verify that $1 \le r \le p-1$; if not, reject the signature.

Compute:

$$\begin{cases} v_1 = a^r r^s \bmod p, \\ v_2 = g^h \bmod p. \end{cases} \tag{7}$$

Accept the signature if and only if $v_1 = v_2$.

IV. Payment Parameters

Payment parameters are embedded in ***P***'s observer ***O***. These are public parameters $(p, g, H())$. One of the possible candidates for the hash function $H()$, considered in this paper is SHA-256 algorithm. The other payment parameters are ***B***'s public key $PuK = (a)$ and private key $PrK = (x)$. Note that PrK is not accessible outside the ***O***. As it was pointed out above, PUF capable of yielding a UUC associated with ***P***'s identity is also embedded in ***O***'s chip, in order to increase the security of e-wallet. Furthermore, ***O*** also contains the maximal amount of e-money m_{max} ***P*** is allowed to spend. Finally, ***O*** is supplied with the random number generator, employed for the generation of e-cash serial number.

V. Description of Payment Protocol

Let ***P*** intends to pay e-cash of amount m_i to the ***V***, whose identity is associated with the identification number Id_V that is signed with ***B***'s e-signature S_{IdV}. Let this operation is performed at the time moment t_i.

A. E-Cash Withdrawal from *O* Protocol

P sends Id_V along with the signature S_{IdV}, sum m_i and time t_i to ***O***.

$$\boldsymbol{P} \longrightarrow Id_V, S_{IdV}, t_i, m_i \longrightarrow \boldsymbol{O}$$

Observer ***O*** performs the following actions:

O verifies that the received time moment is latter than the time moment of the last e-cash withdrawal, m_i is less than m_{max} and S_{IdV} is the valid signature on Id_V. Symbolically, it can be performed by the following verification functions:

$$Ver(t_i > t_{w0}), Ver(m_i \le m_0), Ver(Id_V; S_{IdV}).$$

O generates a random number n_i, computes values E_{Pi}, D_{Pi}, F_{Pi} and signs those values:

$$\begin{cases} n_i \leftarrow \text{Gen}, \\ E_{Pi} = Id_P \times a^{n_i}, \\ D_{Pi} = g^{n_i}, \\ F_{Pi} = E_{Pi} \times D_{Pi} \times Id_V \times a^{t_i + m_i}, \\ S_{Ei} = Sig_x\left(E_{Pi}\right), \\ S_{Fi} = Sig_x\left(F_{Pi}\right), \\ S_{Di} = Sig_x\left(D_{Pi}\right). \end{cases} \quad (8)$$

O saves the received time as the time of the last withdrawal and subtracts the withdrawn sum m_i from m_{max}:

$$t_{w0} \leftarrow t_i, m_{max} \leftarrow (m_{max} - m_i)$$

Finally, ***O*** sends all computed values and signatures to ***P***.

$$\boldsymbol{O} \longrightarrow E_{Pi}, D_{Pi}, F_{Pi}, S_{Ei}, S_{Di}, S_{Fi} \longrightarrow \boldsymbol{P}$$

After the completion of e-cash withdrawal protocol, payment protocol can be executed.

B. Payment Protocol

Firstly, ***P*** sends ***V*** the sum m_i and the time moment t_i of the withdrawal protocol along with all values received from ***O***, except E_{Pi}.

$$\boldsymbol{P} \longrightarrow t_i, m_i, F_{Pi}, D_{Pi}, S_{Ei}, S_{Fi}, S_{Di} \longrightarrow \boldsymbol{V}$$

Afterwards, ***V*** verifies the following: the difference between the received time moment t_i and the time on the vendor's computer t_V is less than Δt_i, which corresponds to the maximal agreed network delay, S_{Ei} is the valid signature on the ***V***'s computed value E_{Pi}', S_{Di} is the valid signature on the D_{Pi} and S_{Fi} is the valid signature on the F_{Pi}:

$$\begin{cases} Ver(|t_i - t_V| < \Delta t), \\ E_{Pi}' = F_{Pi} \times D_{Pi}^{-1} \times Id_V^{-1} \times (a^{m_i + t_i})^{-1}, \\ Ver\left(E_{Pi}'; S_{Ei}\right), \\ Ver\left(D_{Pi}; S_{Di}\right), \\ Ver\left(F_{Pi}; S_{Fi}\right). \end{cases} \quad (9)$$

If all verifications succeed, ***V*** accepts e-cash, which he is able to send to ***B*** to deposit this amount to his bank account. Furthermore, ***V*** is able to place any allowed sum from his bank account to his e-wallet using special protocol between ***B*** and ***V***. However, we do not consider such protocol in this study.

C. Double Spending Prevention

In the case of dishonest ***P*** attempting to double spend some e-cash, the identity of such purchaser can be determined by ***B***, since ***B*** is able to decrypt ***P***'s Id_P using his PrK and values E_{Pi}', D_{Pi}.

VI. The Estimation of the Computation Time for the Payment Protocol

Since the considerable amount of the operations carried out during the payment protocol are performed in ***O***, which has restricted computation resources, the effectiveness of the proposed e-wallet system depends on the estimation of the necessary computation time. The computation time is directly related with the processor's clock frequency. If, for example, the processor is running at 1 GHz clock frequency, then its clock cycle takes $10^{-9}\, s = 1\, ns$.

The list of operations required to perform a payment protocol includes multiplication, addition, shifting and modulo p operation. We refer to those operations as elementary operations. We assume that 32 bit microprocessor is used in ***O***, which is far less than the bit length of parameters (represented by 2048 bit integers) used in payment protocol. Without the loss of generality, we assume that all elementary operations take one clock cycle.

The first step towards the evaluation of the computation time for the payment protocol is the estimation of the number of elementary operations required for the calculation of the modular exponent function $r = g^k \bmod p$. According to [6]–[9], the modular exponent function can be computed using *addition chain method*. The following formula can be used to estimate the number of clock cycles required for the operation of modular exponentiation

$$MOD_E\left(k, p\right) = 1{,}5 \times l(k)[M\left(l\left(p\right)\right) + 2Mod\left(l\left(p\right)\right) + 1], \quad (10)$$

where:

$$M\left(w\right) = 3M\left(w/2\right) + 5A\left(w\right) + 2S, \quad (11)$$

$$A\left(w\right) = w/32, \quad (12)$$

$$Mod\left(w\right) = Mod\left(w/2\right) + 4M\left(w/2\right) + 1{,}5A\left(w\right) + 3S, \quad (13)$$

here $MOD_E(y,z)$ – denotes the operation of modular exponentiation $r = g^k \bmod p$; $M(w), A(w), Mod(w)$ – denote multiplication, addition and modulo operations, respectively, with w being the bit length of the operand; $l(w)$ – denotes the bit length of w; S – denotes the shifting operator.

The bit lengths of the variables in our scheme are the following:

$|m_i| \sim 18 bits, |t_i| \sim 14 bits, |n_i| \sim 256 bits, |h| \sim 256 bits$

The bit length of other parameters such as p, a, g, r, x, Id_V, E_{Pi} etc. is 2048 bits. By default, we use 82 clock cycles for SHA-256 computation [6].

After the number N of clock cycles is computed, the operation time can be estimated using the following formula: $Time = N \times T$, where $T = 1/F$ and F is a clock frequency. In this paper we consider clock frequencies of values 1 GHz, 1.6 GHz and 2 GHz. We used 1.6 GHz in further sections for the demonstration of calculations.

A. Computational Resources for the Withdrawal Protocol

The necessary computational resources for the verification $Ver(Id_V; S_{IdV})$ are presented in Table I.

TABLE I. THE NECESSARY COMPUTATIONAL RESOURCES FOR THE VERIFICATION.

Formula	Required clock cycles	Time, s
$a^r r^s \bmod p$	$MOD_E(r,p) = = MOD_E(2048,2048) = = 117903360$ $MOD_E(s,p) = = MOD_E(2048,2048) = = 117903360$	0.1473792
$h = H(Id_V)$	$82 \times 2048 = 167936$	0.00010496
$g^h \bmod p$	$MOD_E(h,p) = = MOD_E(256,2048) = = 14737920$	0.0092112
Total	250712576	0.15669536

It can be seen from the Table I that the number of clock cycles required for signature verification is 250712576 which corresponds to the operation time 0.157 s.

TABLE II. ESTIMATION OF THE COMPUTATIONAL RESOURCES NEEDED TO PERFORM OTHER ACTIONS IN WITHDRAWAL PROTOCOL.

Formula	Required clock cycles	Time, s
$Id_P \times a^{n_i} \bmod p$	$MOD_E(1,p) = = MOD_E(1,2048) = = 57570$ $MOD_E(n_i,p) = = MOD_E(256,2048) = = 14737920$	0.009247181
$g^{n_i} \bmod p$	$MOD_E(n_i,p) = = MOD_E(256,2048) = = 14737920$	0.0092112
$E_{Pi} \times D_{Pi} \times Id_V \times \times a^{t_i+m_i} \bmod p$	$3 \times MOD_E(1,p) = = 3 \times MOD_E(1,2048) = = 172710$ $3 \times MOD_E(1,p) = = 3 \times MOD_E(1,2048) = = 172710$	0.001259344
Total	31548360	0.019717725

It can be seen that the number of clock cycles and the amount of time needed for the generation of signatures $Sig_x(F_{Pi}) = S_{Fi}$ and $Sig_x(D_{Pi}) = S_{Di}$ are the same – 118149346 clock cycles or, approximately, 0.074 seconds. Therefore, the overall number of clock cycles for the withdrawal protocol is 636647534 and the time needed to perform the protocol is 0.398 seconds.

TABLE. III. COMPUTATIONAL RESOURCES FOR THE SIGNATURE GENERATION $Sig_x(E_{Pi}) = S_{Ei}$.

Formula	Required clock cycles	Time, s
$r = g^k \bmod p$	$MOD_E(k,p) = = MOD_E(2048,2048) = = 117903360$	0.0736896
$h = H(E_{Pi})$	$82 \times 2048 = 167936$	0.00010496
$s = k^{-1}(h - xr) \bmod (p-1)$	$MOD_E(1,p) = = MOD_E(1,2048) = = 57570$	0.00003598
Total	118186436	0.073866523

B. Computational Resources for the Payment Protocol

TABLE. IV. *V* AFTER RECEIVING E-CASH FROM ***P.***

Formula	Clock cycle	Time, s
$E_{Pi}' = F_{Pi} \times \times D_{Pi}^{-1} \times Id_V^{-1} \times \times (a^{m_i+t_i})^{-1} \bmod p$	$3 \times MOD_E(1,p) = = 3 \times MOD_E(1,2048) = = 172710$ $MOD_E(t_i+m_i,p) = = MOD_E(32,2048) = = 1842240$	0.001259344

All such actions as $Ver(E_{Pi}'; S_{Ei})$, $Ver(D_{Pi}; S_{Di})$, $Ver(F_{Pi}; S_{Fi})$ take the same number 250712576 of clock cycles or 0.157 s each. Therefore, the overall number of clock cycles for the payment protocol is 754152678 and the time needed to perform the protocol is 0.471 seconds.

The necessary computation time for withdrawal and payment protocols for three different values of clock frequencies: 1 GHz, 1.6 GHz and 2 GHz, are presented in Table V.

TABLE V. THE COMPARISON OF COMPUTATION TIME FOR WITHDRAWAL AND PAYMENT PROTOCOLS FOR THE DIFFERENT VALUES OF CLOCK FREQUENCIES.

Protocols	Clock Cycles	Time, s (1 GHz)	Time, s (1.6 GHz)	Time, s (2 GHz)
Withdrawal	$6,37 \cdot 10^8$	0,637	0,398	0,318
Payment	$7,54 \cdot 10^8$	0,754	0,471	0,377
Total	$1,39 \cdot 10^8$	1,391	0,869	0,695

VII. CONCLUSIONS

The application of primitive cryptographic functions for the proposed realization of e-wallet satisfying general security requirements of the unclonability of purchaser's identity, anonymity against the vendor and double spending prevention is presented. The unclonability as well as the off-line payment options were achieved by incorporating observer provided with PUF into the e-wallet's architecture.

The analysis of the necessary computation resources for the purchaser's observer as well as the vendor's computation device showed that the proposed e-wallet system requires an acceptable computation time when microprocessors are running at 1 GHz, 1.6 GHz or 2 GHz clock frequency.

REFERENCES

[1] E. Messmer, *Black Hat: Researcher claims hack of chip used to secure computers*, 2010. Online. [Available]: http://www.computerworld.com/article/2520674/security0/black-hat--researcher-claims-hack-of-chip-used-to-secure-computers--smarcards.html

[2] T. Wilson, *Researcher Cracks Security Of Widely Used Computer Chip*, 2010. Online. [Available]: http://www.darkreading.com/risk/researcher-cracks-security-of-widely-used-computer-chip/d/d-id/1132874?

[3] G. E. Suh, S. Devadas, "Physical unclonable functions for device authentication and secret key generation", in *Proc. 44th annual Design Automation Conf.*, San Diego, California, 2007.

[4] *SRAM PUF, The Secure Silicon Fingerprint, White Paper*. Intrinsic ID, 2016. Online. [Available]: https://www.intrinsic-id.com/physical-unclonable-functions/free-white-paper-sram-puf-secure-silicon-fingerprint/

[5] A. J. Menezes, P. C. Van Oorschot, S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 1996.

[6] D. E. Knuth, *The Art of Computer Programming, Vol. 2 (Semi numerical Algorithms)*. Addison-Wesley: Massachusetts, 1980.

[7] R. J. Hwang, F. F. Su, Y. S. Yeh, C. Y. ChenAn, "Efficient decryption method for RSA cryptosystem", in *Proc. 19th Int. Conf. Advanced Information Networking and Applications (AINA 2005)*, 2005. [Online]. Available: http://dx.doi.org/10.1109/AINA.2005.97

[8] A. Bouti, J. Keller, *Towards Practical Homomorphic Encryption in J. Guilford, K. Yap, V. Gopal*, *Fast SHA-256 Implementations on Intel® Architecture Processors*, *White Paper*, 2012. [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/sha-256-implementations-paper.pdf

[9] Cloud Computing. 2015 IEEE Fourth Symposium on Network Cloud Computing and Applications (NCCA), 2015.