9ème Colloque Franco-Roumain de Mathématiques Appliquées,
Braşov, Romanie, 2008.

MATHEMATICAL PROBLEMS AND ALGORITHMS FOR TIMED-RELEASE ENCRYPTION

KONSTANTINOS CHALKIAS, FOTEINI BALDIMTSI,
DIMITRIOS HRISTU-VARSAKELIS, GEORGE STEPHANIDES[1]

Abstract

There are nowdays various e-business applications, such as sealed-bid auctions and electronic voting, that require time-delayed decryption of encrypted data. The literature offers at least three main categories of protocols that provide such timed-release encryption (TRE). They rely either on forcing the recipient of a message to solve a time-consuming, non-paralellizable problem before being able to decrypt, or on the use of a trusted entity responsible for providing a piece of information which is necessary for decryption. This article reviews the mathematical background required for implementing TRE methods, including factorization, quadratic residues and the bilinear Diffie-Hellman problems, along with a sample protocol for each of the approaches studied here.

Part 1. Introduction

The essence of timed-release encryption (proposed by May in [15]) is to encrypt a message so that no one, including the designated recipient(s), will be able to decrypt it before a specified time instant. Various TRE solutions have been proposed in the literature. As a first cut,[15] descibed a basic mechanism in which a third party has the role of an escrow agent, storing the encrypted messages and transmit them to the recipient on the specified by the sender time instant. Since then, a number of new innovative mechanisms appeared, each with its own advanages and disadvantages. In 1996, [18] suggested the method of Time Lock Puzzles

[1]Computational Systems and Software Engineering Laboratory, Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece.
{chalkias, foteini}@java.uom.gr {dcv, steph}@uom.gr

(TLPs), which was based on a non-parallelizable problem that could be easily constructed, but required a minimum amount of time to solve. The same paper, also described how any assymetric encryption scheme could be easily modified to support TRE: A third entity, called a Key Generation Center(KGC), produces a key pair for each required time instant and publishes the public part of these keys right away, so that encryption is possible. Then, the KGC broadcasts each private (decryption) key at the correpsonding time instant. The above techniques were improved later in [14, 5]. Although additional aprroaches were also proposed [8], the breakthrough in the field of TRE came after the introduction of identity based encryption [3]. Beggining in 2003, [16], various protocols were proposed, based on the quadratic residue assumption, and on the properties of bilinear pairings on elliptic curve groups. The goal of this paper is to review the basic techniques used for TRE together with a representative protocol for each approach, and entry points into the relevant literature.

Part 2. Time-Lock Puzzles

All existing CPU-based TLP approaches are based on the same problem: Given a large composite number, n, and integers $t < n$ and a with $gcd(a,n) = 1$, compute the secret value (akin to a decryption key)

$$S = a^{2^t} (mod\ n). \tag{1}$$

It is known that, without factoring n, S can be computed in t squarings modulo n [14]. We remark that S can be easily computed by the sender, as she constructs n, and thus knows $\phi(n)$ (Euler's phi function of n), while it has been proven that this problem cannot be parallelized.

The following is a sample TLP-based protocol from [18]. We assume that we have a sender who wants to encrypt a message M via a time-lock puzzle, to be decrypted in at least T seconds. The steps that he is going to execute are:

1. generate a large composite number, $n = pq$, where p and q are randomly chosen secret primes.
2. compute $\phi(n) = (p-1)(q-1)$.
3. compute $t = TS$, where S is the number of squarings modulo n per second that can be performed by the receiver.
4. generate a random key K for a conventional cryptosystem, such as AES.
5. encrypt M with key K and encryption algorithm AES to obtain the ciphertext $C_M = AES(K,M)$.

6. choose a random a modulo n (with $1 < a < n$) and encrypt K as $C_K = K + a^{2^t} \pmod{n}$ - in order to increase the efficiency, one can initially compute $e = 2^t \pmod{\phi(n)}$ and $b = a^e \pmod{n}$.
7. produce as output the time-lock puzzle (n, a, t, C_K, C_M), and erase any other variables (such as p, q) created during this computation.

The only way for the receiver to decrypt the message is to start with a and perform t squarings sequentially (each time squaring the previous result).

As we saw above, the TLP approach puts immense computational overhead on the receiver, who must perform non-stop non-parallelizable computation in order to retrieve a time-encrypted message. This could be impractical (e.g., it would tie up the received's CPU) if the message is to be read sufficiently far into the future. Moreover, the total time needed to solve a puzzle depends on the receiver's CPU speed and on the time at which the decryption process is started, making it difficult to accurately predict exactly when the message will be "released". Existing TLP approaches include [18], [14], and the timed-release scheme for standard digital signatures in [10].

Part 3. Passive-Server TRE based on Quadratic Residues

When [3] introduced the idea of identity-based encryption(IBE), they refered to TRE as one of its possible applications. [16] implemented at idea, but did so using a different mechanism than that of [3]. In fact, at that time, there were two possible solutions for constructing IBE schemes, one based on bilinear pairings and another one based on quadratic residues [7]. In [16], the authors chose the second approach creating the first server-passive TRE scheme. In their protocol the sender does not communicate with the KGC (or time-server) at all. Thus, the KGCs sole responsibility is to periodically publish a piece of time-embedded information, also called a 'trapdoor', that is required for the decryption of messages. Each trapdoor corresponds to a unique time instant and is to be used by any and all users that want to decrypt a message at that time. The details are described next.

The QR-TRE approach

There are three entities involved in the scheme of [16], a sender (S), a receiver (R) and the KGC.

QR-TRE Initialization (run by the KGC)

1. Choose two different prime numbers p and q that are both congruent to 3 *mod* 4, so $p \equiv 3\%4$ and $q \equiv 3\%4$.
2. Compute the public modulus as $N = pq$.
3. p,q are kept secret and N is published.

QR-TRE Public IBE Key Construction (run by anyone)
This algorithm is used to create the IBE public key that corresponds to the time information and works as follows: A hash function that maps a string on an integer mod N value applied to the string representing the decryption time. The only restriction is that for the hash value, say h, the Jacobi symbol $(\frac{h}{N})$ is $+1$. For instance, if the disclosure time is to be on January, 1^{st} 2009, at 12:00 noon (GMT), the hash output is $h = hash(GMT200901011200)$.

QR-TRE Trapdoor Generator (run by the KGC)

1. Compute $h = hash(GMT200901011200)$ using the QR-TRE Public IBE Key Construction algorithm.
2. Compute the trapdoor $t \equiv sqrt(h)\ mod\ N$. Only the KGC can compute this value, by calculating $t = h^{\frac{N+5-(p+q)}{8}}\ mod\ N$. Such a t will indeed satisfy either $t^2 \equiv h\ mod\ N$ or $t^2 \equiv -h\ mod\ N$, depending upon which of t or $-t$ is a square modulo N.
3. Publish t at decryption time.

QR-TRE Encryption (run by sender)
Suppose that the sender has knowledge of the public value N, and selects a time-instant, say GMT200901011200, to send a single bit, m, to the receiver.

1. Let $r = 2m - 1$, thus $r = -1$ if $m = 0$ and $r = 1$ if $m = 1$.
2. Choose a random, $k \in 0...N-1$, such that the Jacobi symbol $\left(\frac{k}{N}\right) = r$.
3. Compute $h = hash(GMT200901011200)$ using the QR-TRE Public IBE Key Construction algorithm.
4. Compute $s \equiv (k + h/k)\ mod\ N$ and send it to the receiver.

QR-TRE Decryption (run by receiver)
The receiver knows the public value N and he has the encrypted message s.

1. At the appointed time he obtains the trapdoor t from the KGC.

2. Computes m =Jacobi symbol $\left(\frac{s+2t}{N}\right)$.
3. $msg = (m+1)/2$, i.e, $msg = 0$ if $m = -1$, otherwhise $msg = 1$.

Part 4. Modern TRE schemes based on Bilinear Pairings

Since the early work on TA-based TRE schemes, there have been many efforts in order to minimize server-user interaction, as well as to ensure scalability and user-anonymity. After the introduction of IBE several new and innovative TRE techniques appeared in the literature [1, 4, 9, 6, 17],making use of elliptic curve cryptography (ECC) and the efficient implementation of bilinear pairings on ECs.

All modern pairing-based TRE schemes require an abelian, additive finite group $\mathbb{G}_1$, of prime order q, and an abelian multiplicative cyclic group of the same order, $\mathbb{G}_2$. We will let P denote the generator of $\mathbb{G}_1$; H_n will be a secure hash function. Finally, $\widehat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ will be a bilinear pairing.

Definition 1 (Bilinear Pairings): Suppose $\mathbb{G}_1$ is an additive cyclic group generated by P, whose order is a prime q, and $\mathbb{G}_2$ is a multiplicative cyclic group of the same order. A map $\widehat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is called a bilinear mapping if it satisfies the following properties:

1. *Bilinear:* $\widehat{e}(aP,bQ) = \widehat{e}(abP,Q) = \widehat{e}(P,abQ) = \widehat{e}(P,Q)^{ab}$ *for all* $P,Q \in \mathbb{G}_1$ *and* $a,b \in \mathbb{Z}_q^*$
2. *Non-degenerate: there exists* $P,Q \in \mathbb{G}_1$ *such that* $\hat{e}(P,Q) \neq \mathbb{G}_1$
3. *Efficient: there exists an efficient algorithm to compute the bilinear map*

For our purposes, $\mathbb{G}_1$ will be the group of points on an elliptic curve, and $\mathbb{G}_2$ will be a multiplicative subgroup over a finite field. Currently, the Weil, Tate, Ate and η_T pairings can be used to construct an admissible bilinear pairing. Their implementation can be found in [13].

Definition 2 (Discrete Logarithm Problem (DLP)): Given $Q,R \in \mathbb{G}_1$ find an integer $a \in \mathbb{Z}_q^*$ such that $R = aQ$.

Definition 3 (Decisional Diffie-Hellman Problem (DDHP)): Given $Q \in \mathbb{G}_1$, aQ, bQ and cQ for some unknowns $a,b,c \in Z_q^*$ tell whether $c \equiv ab(\bmod q)$.

Definition 4 (Computational Diffie-Hellman Problem (CDHP)): Given $Q \in \mathbb{G}_1$, aQ, bQ for some unknowns $a,b \in Z_q^*$, compute abQ.

Definition 5 (Bilinear Diffie-Hellman Problem (BDHP)): Given $Q \in \mathbb{G}_1$, aQ, bQ and cQ for some unknowns $a, b, c \in Z_q^*$, compute $\hat{e}(Q,Q)^{abc}$.

A Modern Pairing-Based TRE Scheme

To illustrate how a Pairing-Based TRE scheme works, we review the protocol proposed by [11] choosen mainly because of its simplicity. Most, if not all anonymous TRE schemes with pre-open capability are defined by a set of polynomial-time algorithms similar to that described below.

We will denote time by $t \in 0,1^\tau$, $\tau \in \mathbb{N}$ where t indicates the τ-bit string representation of a specific time instant. To send a message, m, that will be decrypted at time t, the following protocol is to be executed:

TRE.Setup(run by the time-server) Given a security parameter k,
1. Output a k-bit prime number q, two groups $\mathbb{G}_1$, $\mathbb{G}_2$ of order q, an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2$ and an arbitrary generator $P \in \mathbb{G}_1$.
2. Choose the following cryptographic hash functions: $H_1 : \{0,1\}^\tau \mapsto \mathbb{G}_1^*$, $H_2 : \mathbb{G}_2^* \mapsto \{0,1\}^n$. T
3. Generate the time-server's private key $s \xleftarrow{R} \mathbb{Z}_q^*$ and the corresponding public key $S = sP \in \mathbb{G}_1^*$.
4. Choose the message space to be $m = \{0,1\}^n$ and the ciphertext space to be $C = \mathbb{G}_1 \times \{0,1\}^{n+\tau}$.
The public parameters are $params := \{k, q, \mathbb{G}_1, \mathbb{G}_2, P, S, \hat{e}, H_1, H_2, n, \tau, m, C\}$.

TRE.ReleaseT (run by the time-server) Given a time instant $t \in \{0,1\}^\tau$ and the server's private key $s \in \mathbb{Z}_q^*$, it returns the time-specific trapdoor $sk_T = sT \in \mathbb{G}_1^*$, where $T = H_1(t) \in \mathbb{G}_1^*$. We note that the trapdoor is in fact a time-server's short signature (as this proposed in [2]) on t, and is inherently self-authenticating. Thus, there is no need for an additional server signature: a user can simply check whether $\hat{e}(S,T) \stackrel{?}{=} \hat{e}(P, sk_T)$.

TRE.KeyGen (run by the receivers) Given *params*, choose a private key $b \in \mathbb{Z}_q^*$ and produce receiver's public key $B = bP \in \mathbb{G}_1^*$.

TRE.Enc (run by the senders) To encrypt $m \in \{0,1\}^n$ using the time information $t \in \{0,1\}^\tau$, the receiver's public key B and the server's public key S,
1. Choose $r \xleftarrow{R} \mathbb{Z}_q^*$.
2. Compute $T = H_1(t) \in \mathbb{G}_1^*$, and $Q = rT \in \mathbb{G}_1^*$.

3. Compute $K = \hat{e}(S,Q) = \hat{e}(sP,rT) = \hat{e}(P,T)^{rs} \in \mathbb{G}_2^*$.
4. Compute $c_1 = rB = rbP \in \mathbb{G}_1^*$ and $c_2 = m \oplus H_2(K) \in \{0,1\}^n$, where $\oplus$ denotes the XOR function. The ciphertext is $C := \langle c_1, c_2, t \rangle$.

TRE.Dec (run by the receivers) Given $C := \langle c_1, c_2, t \rangle$, the trapdoor sk_T and his private key b,
1. Compute $R = b^{-1}c_1 = b^{-1}brP = rP$. R can also be pre-computed (before the release time),
2. The session key is $K = \hat{e}(R, sk_T) = \hat{e}(rP, sT) = \hat{e}(P,T)^{rs} \in \mathbb{G}_2^*$.
3. The message is $m = H_2(K) \oplus c_2$.

This protocol does not allow for pre-opening. If pre-opening is needed (a protocol which supports this function is described in [12]) then the sender must be equipped with an additional algorithm, which can produce a "release key". The latter acts as a secondary trapdoor and permits the receiver to decrypt without waiting (see [12] for additional discussion).

TRE.Genr_k (run by the sender of a message m): Using a randomly-chosen secret value v, generate and output the release key, r_k.

Part 5. Conclusions

We have reviewed the three best-known approaches to TRE. Current challenges in that area include the efficient implementation of the TRE schemes and their practical use as a primitive in a number of real world applications that require such a functionality, such as e-voting, e-lotteries and blind auctions.

References

[1] I. F. Blake and A. C.-F. Chan. Scalable, server-passive, user-anonymous timed release cryptography. In *25th IEEE International Conference on Distributed Computing Systems, pp. 504-513*. IEEE Computer Society, 2005.

[2] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *http://eprint.iacr.org/2005/015*, 2005.

[3] D. Boneh and M.Franklin. Identity based encryption from the weil pairing. In *LNCS 2139, pp. 213-229*. Springer, 2000.

[4] J. Cathalo, B. Libert, and J.-J. Quisquater. Efficient and non-interactive timed-release encryption. In *LNCS 3783, pp. 291-303*. Springer, 2005.

[5] K. Chalkias and G. Stephanides. Timed release cryptography from bilinear pairings using hash chains. In *CMS '06, 10th IFIP International Conference on Communications and Multimedia Security,pp. 130-140*. Springer, 2006.

[6] K. Chalkias, D. Hristu Varsakelis, and G. Stephanides. Improved anonymous timed-release encryption. In *Volume 4734, pp. 311-326*. Springer, 2007.

[7] C. Cocks. An identity based encryption scheme based on quadratic residues. In *http://www.cesg.gov.uk/technology/id-pkc/media/ciren.pdf*, 2001.

[8] G. Di. Crescenzo, R. Ostrovsky, and S. Rajagopalan. Conditional oblivious transfer and timed-release encryption. In *LNCS 1592, pp. 74-89*. Springer, 1999.

[9] A. W. Dent and Q. Tang. Revisiting the security model for timed-release public-key encryption with pre-open capability. In *http://eprint.iacr.org/2006/306.pdf*, 2006.

[10] J. Garay and M. Jakobsson. Timed release of standard digital signatures. In *Financial Cryptography '02, LNCS 2357, pp. 168-182*. Springer, 2002.

[11] D. Hristu-Varsakelis, K. Chalkias, and G. Stephanides. Low-cost anonymous timed-release encryption. In *3rd International Symposium on Information Assurance and Security (IAS '07), IEEE CS*, 2007.

[12] D. Hristu-Varsakelis, K. Chalkias, and G. Stephanides. A versatile secure protocol for anonymous timed-release encryption. In *Journal of Information Assurance and Security, JIAS (to appear)*, 2008.

[13] S. S. Ltd. Miracl - multiprecision integer and rational arithmetic c/c++ library,(see: http://indigo.ie/mscott/).

[14] W. Mao. Timed release cryptography. In *Selected Areas in Cryptography 2001, LNCS 2259, pp. 342-357*. Springer, 2001.

[15] T. May. Timed-release crypto. In *manuscript*, 1993.

[16] M. C. Mont, K. Harrison, and M. Sadler. The HP time vault service: Innovating the way confidential information is disclosed at the right time. In *International World Wide Web Conference, pp. 160-169*. ACM Press, 2003.

[17] Osipkov, I. Kim, Y., and J.-H Cheon. Timed-release public key based authenticated encryption. In *http://eprint.iacr.org/2004/231*, 2004.

[18] R.L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. In *MIT Laboratory for Computer Science Technical Report 684*. Massachusetts Institute of Technology, 1996.