A Comparative study of Forward Secure Publickey Method Using HIBE and BTE

G.Appa Rao[$] Srinivasan.Nagaraj[#] B.Prakash[$]

$Asst.professor Dept of CSE GITAM University
Visakhapatnam-530045 , AP, India
gapparao@gmail.com
#Asst.professor Dept of CSE GMR Inst. of technology
RAJAM-532127 , AP, India
sri.mtech04@gmail.com
Dr.V.Valli Kumari[©] Dr KVSVN Raju[©]
©AU College of Engineering, AU
Visakhapatnam.-AP.India.
Vallikumari@gmail.com , kvsvn.raju@gmail.com

Abstract
The threat of key exposure becoming more acute as cryptographic computations are performed more frequently on poorly protected devices (smart-cards, mobile phones, even PCs), new techniques are needed to deal with this concern. One promising approach which we focus on here is to construct *forward secure* cryptosystems. The existence of non-trivial, forward-secure public-key encryption (PKE) schemes,however, has been open .Forward-secure PKE has the obvious practical advantage that a compromise of the system does not compromise the secrecy of previously-encrypted information; it is thus appropriate for devices operating in insecure environments. Furthermore, using such a scheme enables some measure of security against *adaptive* adversaries who may choose which parties to corrupt based on information learned in the course of a given protocol. We presented in this paper variant of scheme with better complexity; in particular, the public-key size and the key-generation/key-update times are independent of *N* . We suggested a method to achieve chosen ciphertext security for HIBE schemes using the CHK transformation . The resulting schemes are selective-ID chosen-ciphertext secure without random oracles, based on the BTE .

Keywords :Encryption, forward security , linear complexity, BTE ,HIBE

I .INTRODUCTION

A.Forward-secure Public-Key Encryption :

fs-PKE (Canetti, Halevi, and Katz 2003)

Used to protect the private key of one userBased on Gentry-Silverberg HIBE A time period is a binary string

Private key contains decryption key and future secrets

Erase past secrets in algorithm Update .

Based on HIBE [Gentry Silverberg 02] and fs-PKE (Canetti Halevi Katz 03] schemes: Scalable, efficient, and provable secure ,Forward security and Dynamic joins ,Joining-time obliviousness and Collusion resistance ,Chosen-ciphertext secure against adaptive-chosen-(ID-tuple, time) adversary

B .Applications of fs-HIBE :

a)Forward-secure public-key broadcast encryption (fs-BE) .

BE schemes: [Fiat Naor 93] [Luby Staddon 98] [Garay Staddon Wool 00] [Naor Naor Lotspiech 01] [Halevy Shamir 02] [Kim Hwang Lee 03] [Goodrich Sun Tamassia 04] [Gentry Ramzan 04]

HIBE is used in public-key broadcast encryption [Dodis Fazio 02]

Forward security is especially important in BE

b)Multiple HIBE: Encryption scheme for users with multiple roles

From BTE to HIBE : We now show a simple transformation which converts an SN-secure BTE scheme to an SN-secureHIBE scheme. In this transformation we use collision-resistant hashing to map an ID-vector with a bounded number of entries to a bounded-length string. Namely, we apply the hash function separately to each entry in the vector, thus obtaining a string whose length depends only on the number of entries in the input ID-vector (and not the length of these entries). Collision-resistant hashing. Recall that a collision-resistant hash function [12] consists of two algorithms: the seed-generation algorithm sGen that (given the security parameter) picks a seed for the function, and a hashing algorithm Hash that given a seed and an arbitrary-length input string, produces the fixed-length output. The function has the property that for any seed generated by sGen(1k), the output length of the hashing algorithm is always equal to k. The collision-resistance property asserts that any poly-time adversary that is given a random seed s (generated by sGen(1k)), can only _nd strings r1 6= r2 such that Hash(s; r1) = Hash(s; r2) with probability negligible in k.Below it will be convenient to refer to the entry-wise application of the hash function to IDvectors. If s is a seed generated by sGen(1k) and v = (v1; : : : ; v`) is an ID-vector, then w = Hash(s; v) refers to the string H(s; v1)j _ _ _ jH(s; v`).Note that if v 2 (f0; 1g_)t then w 2 f0; 1gkt .

II.PKE SCHEMES WTH LINEAR COMPLEXITY

For completeness, we discuss some simple approaches to forward-secure PKE yielding schemes with linear complexity in at least some parameters. One trivial solution is to generate N independent public-/private- key pairs {(ski; pki)} and to set PK = (pk0; : : : ; pkN-1). In this scheme, the key SKi for time period i will simply consist of (ski; : : : ; sk_N -1). Algorithms for encryption, decryption, and key update are immediate. The drawback of this trivial solution is an N-fold increase in the sizes of the public and secret keys, as well as in the key-generation time. Anderson [3] noted that an improved solution can be built from an identity-based encryption scheme. Here, the public key is the “master public key" of the identity-based scheme, and SKi is computed as the “personal secret key" of a user with identity i (the scheme is otherwise identical to the above). This solution achieves O(1) public key size, but still has O(N) secret-key size and key-generation time. In fact, one can improve this last solution somewhat: instead of a large secret key, it is enough if the user keeps a large non-secret file containing one record per period. The record for period i contains the secret key SKi encrypted under the public key for time period i-1. At the beginning of period i, the user obtains record i, uses key Ski-1 to recover SKi, and then erases Ski-1.

This solution achieves essentially the same efficiency as the “simple forward-secure signatures" of Krawczyk [29] (and in particular requires O(N) non-secret storage and key-generation time).

III.CONSTRUCTION WITH LOGARITHMIC COMPLEXITY

We now construct an encryption scheme secure in the sense of fs-CPA (resp. fs-CCA) from any BTE scheme secure in the sense of SN-CPA (resp. SN-CCA). Our construction is straightforward and is easily seen to be secure given the machinery we have developed for BTE schemes in the following section. At a high level, the construction proceeds as follows: To obtain a forward-secure scheme with N = 2^l+1 + 1 time periods (labeled 0 through N- 1), simply use a BTE of depth l and associate the time periods with all nodes of the tree according to a pre-order traversal. (Let w^i denote the node associated with period i. In a pre-order traversal, w0 = € and if w^i is an internal node then w^i +1 = w^i 0. If w^i is a leaf node and i < N -1 then w^i +1 = w^i 1 where w^i is the longest string such that w^1 0 is a prefix of wi.) The public key is simply the root public key for the BTE scheme; the secret key for period i consists of the secret key for node w^i as well as those for all right siblings of the nodes on the path from the root to w^i. To encrypt a message at time period i, the message is simply encrypted for node w^i using the BTE scheme; decryption is done in the obvious way using the secret key for node w^i (which is stored as part of the secret key for period i). Finally, the period secret key is updated at the end of period i in the following manner: if w^i is an internal node, then the secret keys for wi+1 and its sibling (i.e., the two children of w^i) are derived; otherwise, the secret key for node w^i +1 is already stored as part of the secret key. In either case, the key for node w^i is then deleted. Note that this maintains the property that SKi+1 contains the secret key for w^i +1 as well as those for all right siblings of the nodes on the path from the root to w^i +1. Also, at miost l keys are kept at any point in time. Our method of associating time periods with nodes of a binary tree is reminiscent of previous tree-based forward-secure signature schemes [6, 1, 31]. More formally, given a BTE scheme (Gen, Der, Enc, Dec), we may construct a ke-PKE scheme (Gen0, Upd, Enc0, Dec0) as follows.

a) Algorithm $Gen^1(1^K;N)$ runs $Gen(1^K;$ `), where $N \leq 2^{l+1}$-1, and obtains PK; SK€. It then outputs PK^1 0 = (PK;N), and SK^1 0 = SK€.

b)Algorithm Upd (PK; i; SK^1 i) has SK0 i organized as a stack of node keys, with the secret key SKwi on top. We first pop this key off the stack. If wi is a leaf node, the next key on top of the stack is SKwi+1. If wi is an internal node, compute (SKwi0; SKwi1) ← Der(PK;wi; SKwi) and push SKwi1 and then SKwi0 onto the stack. The new key on top of the stack is SKwi0 (and indeed wi+1 = wi0). In either case, node key SKwi is then erased.

c)Algorithm Enc^1 (PK^1 , i ,M) runs Enc(PK;wi;M). Note that wi is publicly computable given i

and N.

d)Algorithm Dec^1 (PK^1 , i , SK^1 i , M) runs Dec(PK;wi; SKwi ;M). Note that SKwi is always stored as part of $SK^{1\,i}$.

Theorem 1: If BTE scheme (Gen; Der; Enc; Dec) is secure in the sense of SN-CPA (resp. SN-CCA) then ke-PKE scheme (Gen0; Upd; Enc0; Dec0) is secure in the sense of fs-CPA (resp. fs-CCA)

Proof The proof proceeds via straightforward reduction. Assume we have an adversary A^1 with advantage €(k) in an fs-CPA (resp. fs-CCA) attack against (Gen^1 ; Upd; Enc^1; Dec^1). We construct an adversary A that obtains advantage €(k)/N in the corresponding attack against the underlying BTE scheme (Gen; Der; Enc; Dec). Since N is polynomial in the security parameter k, the theorem follows. We now define adversary A:

1. A chooses uniformly at random a time period i^* € [0;N) and outputs wi^* .. Next, A obtains
the public key PK and the appropriate secret keys for the BTE scheme.

2. A runs A^1 with public key (PK,N).

3. When A^1 queries breakin(j) (recall from Remark 1 that without loss of generality A^1 makes
its breakin query before its challenge query), if $j \leq i^*$ then A outputs a random bit and halts.
Otherwise, A computes the appropriate secret key SK^1j and gives this to A^1. (Observe that A
can efficiently compute SK^1 j for $j > i^*$ from the secret keys it has been given.)

4. When A^1 queries challenge(i;M0;M1), if $i \neq i$ * then A outputs a random bit and halts.
Otherwise, A obtains C ← challenge(M0;M1) and gives ciphertext C to A *.

5. If decryption queries are allowed, note that A can respond to queries Dec* 1_(k;C) of A^1 by
simply querying Dec^* (wk;C) and returning the result to A^1.

6. When A^1outputs b^1, A outputs b^1 and halts.

It is straightforward to see that when i_ = i the copy of A^1 running within A has exactly the same
view as in a real fs-CPA (resp. fs-CCA) interaction. Since A guesses i *= i with probability 1/N,
we have that A correctly predicts the bit b with advantage €(k) / N.

IV.FORWARD SECURE PUBLIC-KEY ENCRYPTION

We provide a definition of security for secure public-key encryption forward method and mention two \trivial" forward-secure schemes whose complexity is linear in the total number of time periods. As our main result -which is an immediate application of the BTE primitive discussed in the previous section -we then describe a construction of a forward-secure scheme all of whose parameters grow at most logarithmically with the total number of time periods.

A.Key concepts

We first provide a syntactic definition of key-evolving public-key encryption schemes, and then define what it means for such a scheme to achieve forward security. The former is a straightforward adaptation of the notion of key-evolving signature schemes [6]; the latter, however, is new and requires some care.

Definition 1: A (public-key) key-evolving encryption (ke-PKE) scheme is a 4-tuple of ppt algorithms (Gen; Upd; Enc; Dec) such that:

a)The key generation algorithm Gen takes as input a security parameter 1^K and the total number of time periods N. It returns a public key PK and an initial secret key SK0.

b)The key update algorithm Upd takes as input PK, an index i < N of the current time period, and the associated secret key SKi. It returns the secret key SKi+1 for the following time period.

c)The encryption algorithm Enc takes as input PK, an index $i \leq N$ of a time period, and a message M. It returns a ciphertext C.

The decryption algorithm Dec takes as input PK, an index $i \leq N$ of the current time period, the associated secret key SKi, and a ciphertext C. It returns a message M.

We make the standard correctness requirement: namely, for any (PK; SK0) output by Gen(1^K ;N),
any index i € [0;N) and secret key SKi correctly generated for this time period, and any message M, we have M = Dec(PK; i; SKi ; Enc(PK;w;M)).

Our definitions of forward-secure public-key encryption generalize the standard notions of security for PKE, similar to the way in which the definitions of [6] generalize the standard notion of security for signature schemes.

Definition 2: A ke-PKE scheme is forward-secure against chosen plaintext attacks (fs-CPA) if for all polynomially-bounded functions N(.), the advantage of any ppt adversary in the following game is negligible in the security parameter:

Setup: Gen(1^K;N(k)) outputs (PK; SK0). The adversary is given PK.

Attack: The adversary issues one breakin(i) query and one challenge(j;M0;M1) query, in either order, subject to $0 \leq j < i < N$. These queries are answered as follows:

On query breakin(i), key SKi is computed via Upd(PK; i -1; . . . Upd(PK; 0; SK0) . . .). This key is then given to the adversary.

On query challenge(j;M0;M1), a random bit b is selected and the adversary is given C^* = Enc(PK; j;Mb).

Guess: The adversary outputs a guess b^1 € {0,1}; it succeeds if b^1 = b. The adversary's advantage is the absolute value of the difference between its success probability and 1/2.

We give an analogous definition incorporating chosen-ciphertext attacks by the adversary.

Definition 3: A ke-PKE scheme is forward-secure against chosen-ciphertext attacks (fs-CCA) if for all polynomially-bounded functions N(.), the advantage of any ppt adversary in the following game is negligible in the security parameter:

Setup: Gen(1^K;N) outputs (PK; SK0). The adversary is given PK.

Attack: The adversary issues one breakin(i) query, one challenge(j;M0;M1) query, and multiple Dec(k, C) queries, in any order, subject to $0 \leq J < i < N$ and $0 \leq k < N$. These queries are answered as follows:

The breakin and challenge queries are answered as in Definition 3.1.

On query Dec_(k;C), the appropriate key SKk is first derived using SK0 and PK. Then, the adversary is given the output Dec(PK; k; SKk;C). If the adversary has already received response C^* from query challenge(j;M0;M1), then query $Dec^*(j;C^*)$ is disallowed (but queries $Dec(k;C^*)$, with $k \neq j$, are allowed).

Guess: The adversary outputs a guess b^1 € {0,1}; it succeeds if $b^1 = b$.. The adversary's advantage is the absolute value of the difference between its success probability and 1/2.

Remark 1: On the order of the breaking/challenge queries. The definitions above allow the adversary to make the breaking and the challenge queries in either order. Without loss of generality, however, we may assume the adversary makes the breakin query first. (Specifically, given an adversary A that queries challenge(j;M0;M1) before its breakin query, it is easy to construct an adversary B that queries breakin(j +1) followed by this same challenge query and can then answer any subsequent breakin queries of A; this B will achieve the same advantage as A.)

V. ANALYSIS OF COMPLEXITY PARAMETERS

Each of the four operations (key generation, key update, encryption, and decryption) requires at most one operation of the underlying BTE scheme. We have also noted in the previous section how the secret keys corresponding to any time period can be stored using only O(logN) group elements (rather than the naive O(log2 N)). This justifies the claims given in Table 1 (for schemes achieving security in the sense of fs-CPA), and yields the following corollary.

Corollary 1 Under the decisional BDH assumption, there exists a ke-PKE scheme that is secure in the sense of fs-CPA. Furthermore, all parameters of this scheme are polylogarithmic in the total number of time periods.

Supporting an unbounded number of time periods. In our description above, we have assumed that the number of time periods N is known at the time of key generation. However, it is easy to modify our scheme to support an “unbounded" (i.e., arbitrary polynomial) numberof time periods by using a BTE scheme with depth l= w(log k). Following [31], we can further improve this scheme so that its efficiency depends only logarithmically on the number of time periods elapsed thus far (a simple pre-order traversal using a tree of depth w(log k) results in a scheme with superlogarithmic dependence on N for any N = poly(k)).

CONCLUSION

Our approach could be applied to schemes with BTE and HIBE -like structures There is no HIBE scheme which is fully secure (against adaptive adversaries) with a tight security reduction and without random oracles . A number of constructions of forward-secure signature/identification schemes are known [6, 1,] and forward security for non-interactive, symmetric-key encryption has also been studied [7]. The existence of non-trivial, forward-secure public-key encryption (PKE) schemes, however, has been open since the question was first posed by Anderson [3]. Forward-secure PKE has the obvious practical advantage that a compromise of the system does not compromise the secrecy of previously-encrypted information; it is thus appropriate for devices operating in insecure environments. Furthermore, using such a scheme enables some measure of security against adaptive adversaries who may choose which parties to corrupt based on information learned in the course of a given protocol .

REFERENCES

[1] O. Goldreich. Foundations of Cryptography, vol. 1: Basic Tools. Cambridge University Press, 2001.

[2] O. Goldreich. Foundation of Cryptography, vol. 2. Available on-line from http://www.wisdom.weizmann.ac.il/~oded/foc-vol2.html.

[3] M. Abdalla and L. Reyzin. A new forward-secure digital signature scheme. Asiacrypt 2000,

LNCS vol. 1976, pp. 116{129, Springer-Verlag, 2000.

[3] A. Aho, J. Hopcroft, and J. Ullman. The Design and Analysis of Computer Algorithms.Addison-Wesley, 1975.

[4] R. Anderson. Two remarks on public key cryptology. Invited Lecture, ACM-CCS '97.

http://www.cl.cam.ac.uk/ftp/users/rja14/forwardsecure.pdf.

[5] D. Beaver and S. Haber. Cryptographic protocols provably secure against dynamic adversaries. In Eurocrypt '92, LNCS vol. 658, pp. 307{323, Springer-Verlag, 1992.

[6] D. Beaver. Plug-and-play encryption. Crypto '97, LNCS vol. 1294, pp. 75{89, Springer-Verlag, 1997.

[7] M. Bellare and S. K. Miner. A forward-secure digital signature scheme. Crypto '99, LNCS vol. 1666, pp. 431{448, Springer-Verlag, 1999.