Impact Factor (SJIF): 4.542

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 4, Issue 10, October-2017

A Key Encryption for Digital Signature by Forward Security

Dhanshree S. Madnaik[1], Suhas B. Bhagate[2]

[1]Department of Computer Science and Engineering, D.K.T.E. Society's Textile and Engineering Institute, Ichalkaranji.

[2]Department of Computer Science and Engineering, D.K.T.E. Society's Textile and Engineering Institute, Ichalkaranji.

Abstract - ***In a cryptography there are certain insecure devices in which the threat of key exposure represents a serious problems. In an effort the damage caused by exposure of secrete key stored on such devices. The digital signature scheme in which the public key is fixed but the secret signing key is updated at regular intervals so as to provide a forward security. The exposure of a secret key corresponding to a given time period does not enable an adversary to "break" the scheme for any prior time period. This can be useful to mitigate the damage caused by key exposure without requiring distribution of keys. We present the first constructions of a (non-interactive) forward-secure public-key encryption scheme. Our main construction achieves security against chosen plaintext attacks under the decisional bilinear Diffie Hellman assumption in the standard model.***

Keywords- Bilinear Diffie-Hellman, Encryption, Forward security, Key exposure, Digital Signature.

I. INTRODUCTION

The greatest threat against the security of a digital signature scheme is exposure of the secret signing key, due to compromise of the security of the underlying system or machine storing the key. The most widely considered solution to the problem of key exposure is distribution of the key across multiple servers via secret sharing [1]. There are numerous instantiations of this idea including threshold signatures [2] and proactive signatures [3]. Distribution however is quite costly. With the threat of key exposure becoming more acute as cryptographic computations are performed more frequently on small, unprotected, and easily-stolen devices such as smart-cards or mobile phones, new techniques are needed to deal with this concern. While a large corporation or a certification authority might be able to distribute their keys, the average user, with just one machine, does not have this option. Thus while we expect digital signatures to be very widely used, we do not expect most people to have the luxury of splitting their keys across several machines. Distribution may not provide as much security as one might imagine. For example, distribution is susceptible to common mode failures. A system hole that permits break in might be present on all the machines since they are probably running a common operating system, and once found, all the machines can be compromised.

II. FORWARD SECURE SIGNATURE

The goal of forward security is to protect some aspects of signature security against the risk of exposure of the secret signing key, but in a simple way, in particular without requiring distribution or protected storage devices, and without increasing key management costs. Once a signing key is exposed, the attacker can forge signatures. The idea of "forward security" is however that a distinction can be made between the security of documents pertaining to the past and those pertaining to the period after key exposure. A forward-secure key-exchange protocol by Gunther [4] and Diffie [5], guarantees that exposure of long-term secret information does not compromise the security of previously generated session keys.

A forward-secure key-exchange protocol naturally gives rise to a forward-secure interactive encryption scheme in which the sender and receiver first generate a shared key K, the sender then encrypts his message using K, and both parties promptly erase the shared key. Subsequently, Anderson [6] suggested forward security for the more challenging non-interactive setting. Here, as in the case of forward-secure signature schemes by Bellare and Miner [7] and constructed there and in the lifetime of the system is divided into N intervals or time periods labelled 0. . . N -1. The decryption initially stores secret key SK0 and this secret key "evolves" with time. At the beginning of time period i, the decryption applies some function to the "previous" key SKi-1 to derive the "current" key SKi; key SKi-1 is then erased and SKi is used for all secret cryptographic operations during period i. The public encryption key remains fixed throughout the lifetime of the system. This is crucial

for making such a scheme viable. A forward-secure encryption scheme guarantees that even if an adversary learns SKi for some i, messages encrypted during all time periods prior to i remain secret. The security benefit of this key evolution paradigm is that loss of the current secret key will not enable the adversary to forge signatures with a "date" prior to the one at which key exposure occurred. This "date" has a precise meaning, it is the value of the period during which the signature is produced, which as indicated above is always included in the signature itself. This can be viewed as a means of protecting the authenticity of past transactions even in the event of key exposure. It is quite a useful property and can mitigate the damage of key exposure in several ways.

The following example, based on an idea due to Diffie. Suppose Sara mortgage payment is due on the first of each month. When it is delivered to the bank, the bank issues a signed and dated receipt. First suppose we are using a standard not forward-secure signature scheme. Then for example, if Sara paid rupee 1,000 on January 1st 2014, the text "Sara paid rupee 1,000. On January 1st, 2014" is signed under the public key of the bank. On February 1st, Sara is broke and doesn't pay, and of course the bank issues no receipt. But on February 2nd, Sara cracks the bank computer. Under this normal signing scheme, she gets the signing key, and can forge a note saying "sara paid rupee 1,000 on February 1st, 2014". But not under a forward secure scheme. Under a forward secure scheme assuming daily key updates the signature that would have been produced on February 1st would have the form <32, ζ> where ζ is a tag for the text "sara paid rupee 1, 000," and we are assuming for simplicity that the scheme is initialized on January 1st, so that February 1st, being the 32nd day of the year, has corresponding date or period the number "32". Sara gets only the signing key for February 2nd, the 33rd period. She could forge any signature of the form <j, ζ> for j > 33, for any text of her choice, but not one of the form <32, ζ> So she cannot claim to have paid up on February 1st.

2.1 Keys and key generation

The signer's public key contains a modulus N and l points $U_1, \ldots, U_l$ in Z^*_N. The corresponding base secret key SK_0 contains points $S_1, \ldots, S_l$ in Z^*_N, where S_j is a 2^{T+1th} root of U_j for j = 1, . . . , T. The signer generates the keys by running the following key generation process, which takes as input the security parameter k determining the size of N, the number l of points in the keys, and the number T of time periods over which the scheme is to operate.

Algorithm KG(k; l; T)

Pick random, distinct k/2 bit primes p, q each congruent to 3 mod 4
N←pq
For i = 1, . . . , l do $S_i \leftarrow Z^*_N$; $U_i \leftarrow S_i^{2(T+1)} \bmod N$ EndFor
$SK_0 \leftarrow (N, T, 0, S_{1,0}, \ldots, S_{l,0})$; $PK \leftarrow (N, T, U_1, \ldots, U_l)$
Return (PK, SK_0)

As the code indicates, the keys contain some sundry information in addition to that mentioned above. Specifically the number T of time periods is thrown into the public key. This enables the verifier to know its value, which might vary with different signers. It is also thrown into the secret key for convenience, as is the modulus N. The third component of the base secret key, namely "0", is there simply to indicate that this is the base key, in light of the fact that the key will be evolving later. The modulus is a Blum-Williams integer, meaning the product of two distinct primes each congruent to 3 mod 4. We refer to U_i as the i^{th} component of the public key. The public key PK is treated like that of any ordinary signature scheme as far as registration, certification and key generation are concerenced. The base secret key SK_0 is stored privately. The factors p, q of N are deleted once the key generation process is complete, so that they are not available to an attacker that might later break into the system on which the secret key is stored.

III. FORWARD SECURE ENCRYPTION

Security for forward-secure public-key encryption and also give efficient constructions of schemes satisfying this notion. We prove semantic security of one scheme in the standard model based on the decisional version of the bilinear Diffie-Hellman assumption [8]. All salient parameters of this scheme are logarithmic in N, the number of time periods. We also sketch a variant of this scheme with better complexity. The public-key size and key-generation, key-update times are all independent of N. This variant is proven semantically-secure in the random oracle model under the computational bilinear Diffie-Hellman assumption. Either of our schemes may be extended so as to achieve security against adaptive chosen-ciphertext attacks. The technique that we use for achieving O(1) key generation and key update time appears to be new, and can be used to improve the efficiency of the key generation/key update steps from O(logN) to O(1) in all known tree-based forward-secure signature schemes [7].

3.1. Definitions

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

- 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 SK_0.
- The key update algorithm Upd takes a secret key SK_{i-1} as well as the index i of the current time period. It returns a secret key SK_i for period i.
- The encryption algorithm Enc takes a public key PK, the index i of the current time period, and a message M. It returns a ciphertext C for period i.
- The decryption algorithm Dec takes as input a secret key SK_i and a ciphertext (i, C). It returns a message M. We denote this by M: = Dec_{SKi} (i, C).
- For correctness we require that for any (PK, SK_ε) output by Gen, any secret key SK_i correctly generated for time i, and all M, we have M = Dec_{SKi} (ENC (PK, i, M)).

Definition 2. A key-evolving public-key encryption scheme is forward-secure against chosen plaintext attacks (fs-CPA) if any adversary succeeds in the following game with probability at most negligibly over one half:

- Setup: Gen (1^k, N) is run, with output (PK, SK_0). The adversary is given PK.
- Attack: The adversary issues one breakin(i) query and one challenge(j,M_0,M_1) query, in either order, subject to $0 \le j < i < N$. These queries are answered as:
- On query breakin (i), key SK_i is computed via Upd (· · · Upd (SK_0, 1), · · · , i). This key is then given to the adversary.
- On query challenge (j, M_0, M_1) a random bit b is selected and the adversary is given $C^* = Enc_{PK}$ (j,M_b).

Definition 3. A key-evolving public-key encryption scheme is forward-secure against chosen ciphertext attacks (fs-CCA) if any adversary has only negligible advantage in the following game:

- Setup: Gen (1^k, N) is run, yielding (PK, SK_0). The adversary is given PK.
- Attack: The adversary issues one breakin(i) query, one challenge(j,M_0,M_1) query, and multiple decrypt(k, C) queries, in either order, subject to $0 \le j < i < N$ and $k \le N$. These queries are answered as follows:
- On query breakin (i), key SK_i is computed via Upd (· · · Upd (SK_0, 1), · · · , i). This key is then given to the adversary.
- On query challenge (j, M_0, M_1) a random bit b is selected and the adversary is given $C^* = Enc_{PK}$ (j, Mb).
- A query decrypt (k, C) is answered as follows. If a challenge query was already made (at time unit j), $C = C^*$, and j = k, then the answer is ⊥. Otherwise, the answer is Dec_{SKk}(k, C).

Definition 2 allows the adversary to make the breakin and the challenge queries in any order. However, without loss of generality we may assume that the adversary makes the breakin query first. (Specifically, given an adversary that makes the challenge query before the breakin query, it is easy to construct an adversary that makes the breakin query first and achieves exactly the same advantage.) Assuming that the adversary makes the challenge query first seems to result in a slightly weaker concrete security. Specifically, transforming an adversary that first makes the breakin query into an adversary that first makes the challenge query results in an N-fold loss in the advantage. When N is polynomial in the security parameter, this reduction in security is tolerable.

Definition 3 allows the adversary to make decryption queries for various time periods in arbitrary order Furthermore, the adversary is allowed to obtain the decryption of the challenge ciphertext C^*, as long as the decryption is for a different time period than the one in which the ciphertext was generated. This extra power

given to the adversary results in a definition that is probably stronger than what is needed in most settings, and can potentially be relaxed and still provide adequate security.

IV. Forward-Secure Key Encryption approaches with Linear Complexity

One trivial solution is to generate N independent public-/private- key pairs $\{(sk_i, pk_i)\}$ and to set $PK = (pk_0. . . pk_{N-1})$. In this scheme, the key SK_i for time period i will simply consist of $(sk_i. . . sk_{N-1})$. 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 [6] noted that a somewhat 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 sk_i is computed as the "personal secret key" of a user with identity i. This solution achieves O(1) public key size, but still has O(N) secret key size and key-generation time. In fact, one could improve this last solution even more: 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 stores the secret key sk_i encrypted for time period i − 1. At the beginning of period i, the user obtains record i, uses key sk_{i-1} to recover sk_i, and then erases sk_{i-1}. This solution achieves essentially the same efficiency as the "simple forward secure signatures" of Krawczyk [10] and in particular requires O(N) non-secret storage and key-generation time.

4.1 Logarithmic Complexity Construction

We construct a fs-CPA encryption scheme from any SN-CPA BTE scheme. For this purpose, we use a BTE scheme with full tree of depth log N, together with a tree-traversal technique to assign nodes to time periods. This is somewhat similar to prior forward-secure signature schemes [7], [11] except that we utilize all the nodes in the tree, rather than only the leaves. This results in complexity gain (from O(log n) to O(1)) in some of the parameters. The scheme is very simple: For a forward-secure scheme with $N = 2^{n+1}$ time periods, use a BTE with full binary tree of N nodes and depth n. That is, use the set $W = \{0, 1\}^{\leq n}$. At time i, the "active node", denoted w^i, is the ith node in a pre-order traversal of the BTE tree. (Pre-order traversal can be defined as follows: $w^1 = \varepsilon$. For $i > 1$, if w^i is an internal node ($|w^i| < n$), then $w^{i+1} = w^i0$. If w^i is a leaf ($|w^i| = n$), then $w^{i+1} = w1$, where w is the longest string such that w0 is a prefix of w^i.) Encryption in time period i uses the tree public key and the name of node w^i. Ciphertexts for time unit i are decrypted using the secret key of node w^i. In addition, in time unit i we also keep in memory the secret keys of the "right siblings" of the nodes on the path from the root to w^i. That is, whenever w'0 is a prefix of w^i, we keep in memory the secret key of node w'1. At the end of period i, we compute the secret key of w^{i+1} and erase the secret key of w^i. Note that w^{i+1} is either the left child of w^i, or one of the nodes whose secret keys were stored in memory. Hence, we need at most one application of the Der function to compute the new secret key.

Formally, given a BTE scheme (Gen, Der, Enc, Dec), construct a key-evolving scheme (Gen', Upd, Enc', Dec') as follows.

- Algorithm $Gen'(1^k,N)$ runs $Gen(1^k)$, and obtains PK,SK_ε. It then outputs $PK' = PK$, and $SK'_0 = SK_\varepsilon$.

- Algorithm $Upd(i+1, SK'_i)$: The secret key SK'_i is organized as a stack of node keys, with the secret key SK_w^i on top. We first pop the current secret key, SK_w^i, off the stack. If w^i is a leaf ($|w^i| = n$) then we are done. The next key on top of the stack is SK_w^{i+1}. Otherwise (w^i is an internal node, $|w^i| < n$), we set $(SK_w^i0, SK_w^i1) \leftarrow Der(PK,w^i, SK_w^i)$, and push SK_w^i1 and then SK^i_w0 onto the stack. The new top is SK_w^i0 (and indeed $w^{i+1} = w^i0$). In either case, Upd erases the key SK_w^i.

- Algorithm $Enc'(PK', i,M)$ runs $Enc(PK,w^i,M)$.
- Algorithm $Dec'(SK'_i, i,M)$ runs $Dec(SK_w^i, w^i,M)$.

4.1.1 Thereom

The scheme (Gen', Upd, Enc', Dec') is fs-CPA secure, assuming that the underlying scheme (Gen, Der, Enc, Dec) is a CPA-secure BTE scheme.

The proof proceeds via straightforward reduction. Assume we have an adversary A' that has advantage ε in a CPA attack against the forward-secure scheme (Gen', Upd, Enc', Dec'). We construct an adversary A that obtains
advantage ε/N in the corresponding attack against the underlying BTE scheme (Gen, Der, Enc, Dec). Adversary A proceeds as follows.

- A chooses at random a time period $i^* \in_R \{1...N\}$, and declares that the BTE node to be attacked is w^{i*} . Next, A obtains the public key PK and the appropriate secret keys for the BTE scheme.

- A runs A', giving it public key PK.

- When A' generates a challenge (i,M_0,M_1), if $i = i^*$ then A outputs a random bit and halts. If $i = i^*$ then A generates a challenge (M_0,M_1), obtains the ciphertext $C^* = Enc(PK,w^{i*},M_b)$ and hands it over to A'.

- When A' generates a breakin query for time unit i, if $i \leq i^*$ then A outputs a random bit and halts. If $i > i^*$ then A hands A' the secret key SK'_i can be computed from the secret keys known to A.

- When A' generates a decryption request for a ciphertext $C \neq C^*$ at time unit i' for which A has the corresponding decryption key $SK_w^{i'}$, then A decrypts C and hands the answer to A'. If A does not have the corresponding decryption key then it hands $(C, w^{i'})$ to its own decryption oracle, and forwards the answer to A'.

- When A' outputs a prediction bit b', A outputs b' and halts. Analysing A, it is straightforward to see that, conditioned on the event that $i = i^*$, the copy of A' running within A has exactly the same view as in a real CPA interaction with a BTE scheme. Consequently, A predicts the bit b with advantage ε/N.

Analysis of complexity parameters is each of the four operations (key generation, update, encryption, and decryption) requires at most one operation of the underlying BTE scheme. Thus, the complexity of our scheme is essentially the same as that of our BTE construction

REFERENCES

[1] A. Shamir, "How to share a secret," Communications of the ACM, 22(1979), 612-613.

[2] Y. Desmedt and Y. Frankel, "Threshold cryptosystems." *Advances in Cryptology - Crypto* 89 *Proceedings*, Lec. Notes in Comp. Sci. Vol. 435, G. Brassard ed., Springer-Verlag, 1989.

[3] A Herzberg, M. Jakobsson, S .Jarecki, H Krawczyk and M. Yung, "Proactive public key and signature schemes." Proceedings of the Fourth Annual Conference on Computer and Communications Security, ACM, 1997.

[4] C. G. Gunther. "An identity-based key-exchange protocol." Advances in Cryptology Eurocrypt '89, LNCS vol. 434, pp. 29–37, Springer-Verlag, 1989.

[5] W. Diffie, P. C. Van-Oorschot, and M. J.Weiner. "Authentication and authenticated key exchanges." *Designs, Codes, and Cryptography* 2:107–125, 1992.

[6] R. Anderson. "Two remarks on public key cryptology". Invited Lecture, ACM-CCS 97. http://www.cl.cam.ac.uk/ftp/users/rja14/forwardsecure.pdf.

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

[8] A. Joux and K. Nguyen. Separating decision diffie-hellman from diffie-hellman incryptographic groups. Manuscript, January 2001. Available at http://eprint.iacr.org/2001/003/.

[9] H. Krawczyk. "Simple forward-secure signatures from any signature scheme". *Proc. 7th ACM-CCS*, pp. 108–115, ACM, 2000.

[10] M. Abdalla and L. Reyzin. "A new forward-secure digital signature scheme". Asiacrypt '00, LNCS vol. 1976, pp. 116–129, Springer-Verlag, 2000.

[11] T. Malkin, D. Micciancio, and S. K. Miner. "Efficient generic forward-secure signatures with an unbounded number of time periods". Advances in Cryptology Eurocrypt 2002, LNCS vol. 2332, pp. 400–417, Springer-Verlag, 2002.

[12] M. Naor and M. Yung, "Public key cryptosystems provably secure against chosen ciphertext attacks", 22nd STOC, 427-437, 1990.

[13] J. B. Nielsen. "Separating random oracle proofs from complexity theoretic proofs": The non-committing encryption case. Crypto 02, LNCS vol. 2442, pp. 111–126,Springer-Verlag, 2002.

[14] R. Ostrovsky and M. Yung. "How to withstand mobile virus attacks". 10th Annual Symposium on Principles of Distributed Computing, pages 51–59, ACM, 1991.

[15] C. Rackoff and D. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack", Crypto 91, LNCS vol. 576, pp. 433–444, Springer-Verlag, 1991.

[16] D. Pointcheval and J. Stern, "Security proofs for signature schemes," Ad- vances in Cryptology Eurocrypt 96 Proceedings, Lec. Notes in Comp. Sci. Vol. 1070, U. Maurer ed., Springer-Verlag, 1996.

[17] V. Shoup, "On the security of a practical identication scheme," Advances in Cryptology Eurocrypt 96 Proceedings, Lec. Notes in Comp. Sci. Vol. 1070, U. Maurer ed., Springer-Verlag, 1996.

[18] H. Williams, "A Modication of the RSA Public-key Encryption Procedure," IEEE Transactions on Information Theory, Vol. IT-26, No. 6, 1980, pp. 726-729.

[19] J. Horwitz and B. Lynn. "Toward hierarchical identity-based encryption". Eurocrypt'02, LNCS vol. 2332, pp. 466–481, Springer-Verlag, 2002.

[20] G. Itkis and L. Reyzin. "Forward-secure signatures with optimal signing and verifying". Crypto '01, LNCS vol. 2139, pp. 499–514, Springer-Verlag, 2001.