A Forward-Secure Public-Key Encryption Scheme

Ran Canetti[1], Shai Halevi[1], and Jonathan Katz[2]

[1] IBM T. J. Watson Research Center, Hawthorne, NY.
{canetti,shaih}@watson.ibm.com
[2] Dept. of Computer Science, University of Maryland, College Park, MD.
jkatz@cs.umd.edu

Abstract. Cryptographic computations are often carried out on insecure devices for which the threat of key exposure represents a serious and realistic concern. In an effort to mitigate the damage caused by exposure of secret data (e.g., keys) stored on such devices, the paradigm of *forward security* was introduced. In a forward-secure scheme, secret keys are updated at regular periods of time; furthermore, exposure of a secret key corresponding to a given time period does not enable an adversary to "break" the scheme (in the appropriate sense) for any *prior* time period. A number of constructions of forward-secure digital signature schemes, key-exchange protocols, and symmetric-key schemes are known.

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. It is practical, and all complexity parameters grow at most logarithmically with the total number of time periods. The scheme can also be extended to achieve security against chosen ciphertext attacks.

Key words: Bilinear Diffie-Hellman, Encryption, Forward security, Key exposure.

1 Introduction

Exposure of secret keys can be a devastating attack on a cryptosystem since such an attack typically implies that all security guarantees are lost. Indeed, standard notions of security offer no protection whatsoever once the secret key of the system has been compromised. 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.

A variety of methods (including secret sharing [33], threshold cryptography [13], and proactive cryptography [30, 20]) have been introduced in an attempt to deal with this threat. One promising approach — which we focus on here — is to construct cryptosystems which are *forward secure*. This notion was first proposed in the context of key-exchange protocols by Günther [18] and Diffie, et al. [14]: a forward-secure key-exchange protocol guarantees that exposure of long-term secret information does not compromise the security of previously-generated session keys. We remark that 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 [3] suggested forward security for the more challenging non-interactive setting. Here, as in the case of forward-secure signature schemes (formalized by Bellare and Miner [6] and constructed there and in [26, 1, 22, 27, 25]), the lifetime of the system is divided into N intervals (or time periods) labeled $0, \ldots, N-1$. The decryptor initially stores secret key SK_0 and this secret key "evolves" with time. At the beginning of time period i, the decryptor applies some function to the "previous" key SK_{i-1} to derive the "current" key SK_i; key SK_{i-1} is then *erased* and SK_i 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 SK_i (for some i), messages encrypted during all time periods *prior* to i remain secret. (Note that since the adversary obtains all secrets existing at time i, the model inherently cannot protect the secrecy of messages encrypted at time i and at all subsequent time periods.)

Forward security for non-interactive, symmetric-key encryption has also been studied [7]. However, no forward-secure (non-interactive) public-key encryption (PKE) schemes were previously known. Forward-secure PKE has the obvious practical advantage that a break-in to the system does not compromise the secrecy of previously-encrypted information; it is thus appropriate for use in devices with low security guarantees, such as mobile devices. In particular, it provides a practical encryption mechanism that is secure against *adaptive* adversaries.

1.1 Our Contributions

Forward secure encryption. We rigorously define a notion of 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 (cf. [24, 9]). 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; in particular, 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. (Recall that a proof in the random oracle

	Standard model	Random oracle model
Key generation time	$\mathcal{O}(\log N)$	$\mathcal{O}(1)$
Encryption/Decryption time	$\widetilde{\mathcal{O}}(\log N)$	$\mathcal{O}(\log N)$
Key update time	$\mathcal{O}(\log N)$	$\mathcal{O}(1)$
Ciphertext length	$\mathcal{O}(\log N)$	$\mathcal{O}(\log N)$
Public key size	$\mathcal{O}(\log N)$	$\mathcal{O}(1)$
Secret key size	$\mathcal{O}(\log N)$	$\mathcal{O}(\log N)$

Table 1. Summary of dependencies on the total number of time periods N.

model provides no guarantee as to the security of the protocol once the random oracle is instantiated with an efficiently computable function, such as a "cryptographic hash function", and thus can only be regarded as a heuristic argument.)

The key parameters of the two schemes are summarized in Table 1. We stress that both schemes are efficient not only in an asymptotic sense; indeed, they are roughly as efficient as $\log_2 N$ invocations of the Boneh-Franklin identity-based encryption scheme [9] and are therefore practical for reasonable values of N. Using the techniques of Malkin, et al. [27], our schemes can be adapted to cases where the number of time periods is not known in advance. This has the added advantage that the efficiency and security of the schemes depend only on the number of time periods elapsed thus far. We also sketch two ways to extend our schemes to achieve security against adaptive chosen ciphertext attacks [31, 15, 8]. One method is based on Sahai's construction [28, 32] and works in the standard model. The other is based on the Fujisaki-Okamoto transformation [16] and is analyzed in the random oracle model.

Other contributions and applications. Our constructions are based on the Gentry and Silverberg [17] construction of a hierarchical identity-based encryption (HIBE) scheme [21, 17]. As a first step towards our construction, we define a relaxed variant of HIBE which we call binary tree encryption (BTE). We show how to modify the Gentry-Silverberg construction to yield a BTE scheme without adding much complexity. Remarkably, the modified construction works in the standard model and can handle trees of polynomial depth. (In contrast, the main construction of Gentry and Silverberg is analyzed only in the random oracle model, and only for trees of constant depth.) We then construct a forward-secure encryption scheme from any BTE scheme. In addition, the BTE primitive may prove interesting in its own right.

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 $O(\log N)$ to $O(1)$ in all known tree-based forward-secure signature schemes [6, 1, 27].

Forward-secure PKE may be used to drastically improve the efficiency of non-interactive *adaptively secure* encryption in the context of secure multi-party protocols. In such protocols, an adaptive adversary may choose to corrupt a player at some point in the protocol, after that player had already received several encrypted messages. Learning the player's secret key will (in general) allow the adversary to read all past messages, thereby making it much harder to prove any simulation-based notion of security. In all known adaptively secure non-interactive encryption schemes (e.g., [4, 11, 5, 12]) the size of the decryption key must exceed the total length of messages to be decrypted throughout the life-time of the system. Furthermore, Nielsen has recently shown that this property is *essential* for encryption schemes that are not key-evolving [29]. (This holds even if the model allows data erasures.) Forward-secure encryption enables us to circumvent this lower bound, by having each player update its (short) key after every message. This way, an adversary who corrupts a player at some point in the protocol will be unable to read past messages that were sent to this player, thereby enabling a simulation-based proof of security.

Organization. In Section 2 we define and construct our underlying primitive, BTE, and prove its security under the decisional bilinear Diffie-Hellman assumption (also described in that section). Then in Section 3 we define forward secure public key encryption and show how it can be constructed from any BTE scheme. Putting these two results together, we get a construction with the advertised security and efficiency parameters.

2 A Binary Tree Encryption Scheme

This section defines binary tree encryption (BTE), and presents a construction based on the bilinear Diffie-Hellman assumption. As discussed in the introduction, a BTE scheme is a relaxed version of hierarchical identity-based encryption (HIBE) [21, 17]. In addition to being an essential step in our construction of forward-secure encryption, BTE may be an interesting primitive by itself. In particular, we also show how to implement a full-blown HIBE from BTE, and since we describe a BTE whose security can be proven in the standard model, this also implies a secure HIBE in the standard model (albeit, with a somewhat weaker notion of security than that considered in [17]).

As in HIBE, in BTE too we have a public key associated with a tree, and each node in this tree has a corresponding secret key. To encrypt a message destined for some node, one uses both the tree public key and the name of the target node. The resulting ciphertext can then be decrypted using the secret key of the target node. Moreover, just as in HIBE, the secret key of a node can also be used to derive the secret keys of the children of that node. The only difference between HIBE and BTE is that in the latter we insist that the tree be a binary tree, where the children of a node w are labeled $w0$ and $w1$. (Recall that in HIBE, the tree can have arbitrary degree, and a child of node w can be labeled $w.s$ for any arbitrary string s.) The formal functional definition follows.

Definition 1. *A* binary tree public-key encryption (BTE) scheme *is a 4-tuple of* PPT *algorithms* $(\mathsf{Gen}, \mathsf{Der}, \mathsf{Enc}, \mathsf{Dec})$ *such that:*

– *The* key generation algorithm Gen *takes as input a security parameter* 1^k*, and returns a public key* PK *and an initial (root) secret key* SK_ε*.*

– *The* key derivation algorithm Der *takes the name of a node* $w \in \{0,1\}^*$ *and its associated secret key* SK_w*, and returns the two secret keys* SK_{w0}, SK_{w1} *for the two children of* w.
– *The* encryption algorithm Enc *takes a public key* PK*, the name* w *of a node, and a message* m*. It returns a ciphertext* c.
– *The* decryption algorithm Dec *takes as input a public key* PK*, a node name* w*, its secret key* SK_w*, and a ciphertext* c*. It returns a message* m.

We make the standard correctness requirement; namely, for any (PK, SK_ε) *output by* Gen*, any node* $w \in \{0,1\}^*$*, and a secret key* SK_w *correctly generated for node* w*, and all* m*, we have* $m = \mathsf{Dec}(PK, w, SK_w, \mathsf{Enc}(PK, w, m))$.

The security notion that we present here for BTE is somewhat weaker than the standard notion for HIBE, in that we require that the an attacker commits to the node to be attacked *even before seeing the public key.* We call this attack scenario *selective-node* attack (cf. "selective forgery" of signatures [19]). For simplicity, we formally define here only security against chosen plaintext attacks. Security against adaptive chosen ciphertext attacks is defined analogously.

Definition 2 (SN-CPA security). *Let* $W \subset \{0,1\}^*$ *be a set that is closed under prefix. A BTE scheme is* secure against selective-node, chosen-plaintext attacks *(*SN-CPA*) with respect to* W *if any* PPT *adversary succeeds in the following game with probability at most negligibly over one half:*

1. *The adversary generates a name* $w^* \in W$ *of a node in the tree.*
2. *Algorithm* $\mathsf{Gen}(1^k, N)$ *is run, with output* (PK, SK_ε)*, and the adversary is given* PK*. In addition, the algorithm* $\mathsf{Der}(\cdots)$ *is run to generate the secret keys of all the nodes on the path from the root to* w^**, as well as the children of* w^**. The adversary gets the secret key* SK_w *of any node* $w \in W$ *such that*
   – *either* $w = w'\overline{b}$*, where* $w'b$ *is a prefix of* w^* *(a sibling);*
   – *or* $w = w^*0$ *or* $w = w^*1$ *(a child).*
   (Note that this allows the adversary to compute $SK_{w'}$ *for any node* $w' \in W$ *that is not a prefix of* w^**.)*
3. *The adversary generates a request* $\mathsf{challenge}(m_0, m_1)$*. In response, a random bit* b *is selected and the adversary is given* $c^* = \mathsf{Enc}(PK, w^*, m_b)$.
4. *The adversary outputs a guess* $b' \in \{0,1\}$*. It* succeeds *if* $b' = b$.

2.1 The Bilinear Diffie-Hellman Assumption

The security of our binary tree encryption scheme is based on the difficulty of the bilinear Diffie-Hellman (BDH) problem as formalized by Boneh and Franklin [9] (see also [24]). The computational version of this assumption has been used for a number of cryptographic constructions (e.g., [23, 10, 34, 21, 17]); furthermore, as noted in [9], the decisional version of the assumption (called the BDDH assumption there) is also believed to hold. We review the relevant definitions as they appear in [9, 17]. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of prime order q, where $\mathbb{G}_1$ is represented additively and $\mathbb{G}_2$ is represented multiplicatively. We assume a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ for which the following hold:

1. The map $\hat{e}$ is *bilinear*; that is, for all $P_0, P_1 \in \mathbb{G}_1$ and all $\alpha, \beta \in \mathbb{Z}_q$ we have $\hat{e}(\alpha P_0, \beta P_1) = \hat{e}(P_0, P_1)^{\alpha\beta}$.
2. There is an efficient algorithm to compute $\hat{e}(P_0, P_1)$ for any $P_0, P_1 \in \mathbb{G}_1$.

A *BDH parameter generator* $\mathcal{IG}$ is a randomized algorithm that takes a security parameter 1^k, runs in polynomial time, and outputs the description of two groups $\mathbb{G}_1, \mathbb{G}_2$ and a map $\hat{e}$ satisfying the above conditions. We define the *computational BDH problem with respect to* $\mathcal{IG}$ as the following: given $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$ output by $\mathcal{IG}$ along with random $P, \alpha P, \beta P, \gamma P \in \mathbb{G}_1$, compute $\hat{e}(P, P)^{\alpha\beta\gamma}$. We say that $\mathcal{IG}$ *satisfies the computational BDH assumption* if the following is negligible (in k) for all PPT algorithms A:

$$\Pr\left[\begin{array}{l}(\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}(1^k); P \leftarrow \mathbb{G}_1; \alpha, \beta, \gamma \leftarrow \mathbb{Z}_q : \\ A(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, \alpha P, \beta P, \gamma P) = \hat{e}(P, P)^{\alpha\beta\gamma}\end{array}\right].$$

Informally speaking, the *decisional BDH problem* is to distinguish between tuples of the form $(P, \alpha P, \beta P, \gamma P, \alpha\beta\gamma P)$ and $(P, \alpha P, \beta P, \gamma P, \mu P)$ for random $P, \alpha, \beta, \gamma, \mu$. More formally, we say that $\mathcal{IG}$ *satisfies the decisional BDH assumption* (BDDH) if the following is negligible (in k) for all PPT algorithms A:

$$\left|\Pr\left[\begin{array}{l}(\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}(1^k); P \leftarrow \mathbb{G}_1; \alpha, \beta, \gamma, \leftarrow \mathbb{Z}_q : \\ A(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, \alpha P, \beta P, \gamma P, \alpha\beta\gamma P) = 1\end{array}\right]\right.$$
$$\left.- \Pr\left[\begin{array}{l}(\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}(1^k); P \leftarrow \mathbb{G}_1; \alpha, \beta, \gamma, \mu \leftarrow \mathbb{Z}_q : \\ A(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, \alpha P, \beta P, \gamma P, \mu P) = 1\end{array}\right]\right|.$$

The decisional BDH assumption immediately implies that it is computationally infeasible to distinguish between tuples of the form $(P, \alpha P, \beta P, \gamma P, \hat{e}(P, P)^{\alpha\beta\gamma})$ and $(P, \alpha P, \beta P, \gamma P, r)$ for random $P, \alpha, \beta, \gamma, r$.

BDH parameter generators believed to satisfy the above assumptions can be constructed from Weil and Tate pairings associated with super-singular elliptic curves or Abelian varieties. As our results do not depend on any specific instantiation, we refer the interested reader to [9] for details.

2.2 A construction

Theorem 1. *There exists a BTE scheme that is secure in the sense of SN-CPA under the BDDH assumption.*

To prove Theorem 1 we describe such a BTE scheme. The starting point for our construction is the hierarchical identity-based encryption scheme of Gentry and Silverberg [17]. Our construction, unlike the original scheme, can be proven secure in the standard model. (In fact, from our argument it follows that in the random oracle model, the scheme of [17] is itself a secure BTE.) The HIBE of Gentry and Silverberg (as well as the IBE scheme of Boneh and Franklin [9]) uses two random oracles: one is used to derive random elements from the identities, and the other is used to achieve CCA security of the encryption. The latter use of the random oracle can be removed when one considers only CPA security.

Below we show that in the context of selective node security, one can replace the first random oracle by a fully specified "t-wise independent" hash function.

Notations and conventions. In the description below we denote the i-bit prefix of a word $w_1 w_2 \ldots w_\ell$ by $w|_i$. Namely, $w|_i = w_1 \ldots w_i$. In defining our scheme, we use a $(2t+1)$-wise independent, efficiently sampleable family $\mathcal{H}$ of functions $H : \{0,1\}^{\leq t} \to \mathbb{G}_1$. By *efficiently sampleable* we mean that, given elements $x_1, \ldots, x_k \in \{0,1\}^{\leq t}$ and $g_1, \ldots, g_k \in \mathbb{G}_1$ (with $k \leq 2t+1$), it is possible to efficiently sample a random $H \in \mathcal{H}$ such that $H(x_i) = g_i$ for $i = 1, \ldots, k$.) One possible instantiation is to let $\mathcal{H} = \{H_{h_0,\ldots,h_{2t}}(x)\}_{h_0,\ldots,h_{2t} \in \mathbb{G}_1}$, where $H_{h_0,\ldots,h_{2t}}(x) \stackrel{\text{def}}{=} h_0 + \tilde{x} h_1 + \cdots + \tilde{x}^{2t} h_{2t}$ and $\tilde{x}$ represents some standard one-to-one encoding of $x \in \{0,1\}^{\leq t}$ as an element in $\mathbb{Z}_q$. Let $\mathcal{IG}$ be a BDH parameter generator for which the decisional BDH assumption holds. For our construction we need an upper bound on the height of the tree, and we denote this bound by t. The construction is described below.

$\mathsf{Gen}(1^k, t)$ does the following:

1. Run $\mathcal{IG}(1^k)$ to generate groups $\mathbb{G}_1, \mathbb{G}_2$ of prime order q and bilinear map $\hat{e}$.
2. Select a random generator $P \in \mathbb{G}_1$ and a random $\alpha \in \mathbb{Z}_q$. Set $Q = \alpha P$.
3. Choose a random function $H \in \mathcal{H}$. Note that this merely requires choosing $2t+1$ random elements $h_0, \ldots, h_{2t} \in \mathbb{G}_1$.
4. The public key is $PK = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, Q, t, H)$ (where H is represented by $h_0, \ldots, h_{2t}$). The root secret key is $S_\varepsilon = \alpha H(\varepsilon)$.

$\mathsf{Der}(PK, w, SK_w)$ takes as input the tree public key and the secret key associated with node w, and outputs the secret keys for nodes $w0$ and $w1$. In general, for $w = w_1 \ldots w_\ell$, the secret key of node w consists of $\ell + 1$ group elements, $SK_w = (R_{w|_1}, R_{w|_2}, \ldots, R_w, S_w)$. The algorithm $\mathsf{Der}(PS, w, SK_w)$ runs as follows:

1. Choose random $\rho_{w0}, \rho_{w1} \in \mathbb{Z}_q$. Set $R_{w0} = \rho_{w0} P$ and $R_{w1} = \rho_{w1} P$, and also $S_{w0} = S_w + \rho_{w0} H(w0)$ and $S_{w1} = S_w + \rho_{w1} H(w1)$.
2. Output $SK_{w0} = (R_{w|_1}, \ldots, R_w, R_{w0}, S_{w0})$ and $SK_{w1} = (R_{w|_1}, \ldots, R_w, R_{w1}, S_{w1})$.

$\mathsf{Enc}(PK, w, m)$ (where $m \in \mathbb{G}_2$) does the following:

1. Let $w = w_1 \ldots w_\ell$. Select random $\gamma \in \mathbb{Z}_q$.
2. Output $\bar{C} = (\gamma P, \gamma H(w|_1), \gamma H(w|_2), \ldots, \gamma H(w),\ m \cdot d)$, where $d = \hat{e}(Q, H(\varepsilon))^\gamma$.

$\mathsf{Dec}(PK, w, sk_w, \bar{C})$ does the following: Let $w = w_1 \cdots w_\ell$, $sk_w = (R_{w|_1}, \ldots, R_w, S_w)$, and $\bar{C} = (U_0, U_1, \ldots, U_\ell, v)$. Compute $m = v/d$, where

$$d = \frac{\hat{e}(U_0, S_w)}{\prod_{i=1}^{\ell} \ \hat{e}(R_{w|_i}, U_i)}$$

We now verify that decryption is performed correctly. When encrypting, we have $d = \hat{e}(Q, H(\varepsilon))^\gamma = \hat{e}(P, H(\varepsilon))^{\alpha\gamma}$. When decrypting, we have $U_0 = \gamma P$, and

$U_i = \gamma H(w|_i)$ for $i \geq 1$. Hence,

$$d = \frac{\hat{e}(U_0, S_w)}{\prod_{i=1}^{\ell} \hat{e}(R_{w|_i}, U_i)} = \frac{\hat{e}\left(\gamma P, \alpha H(\varepsilon) + \sum_{i=1}^{\ell} \rho_{w|_i} H(w|_i)\right)}{\prod_{i=1}^{\ell} \hat{e}\left(\rho_{w|_i} P, \gamma H(w|_i)\right)}$$
$$= \frac{\hat{e}(P, H(\varepsilon))^{\alpha\gamma} \cdot \prod_{i=1}^{\ell} \hat{e}\left(P, H(w|_i)\right)^{\gamma \rho_{w|_i}}}{\prod_{i=1}^{\ell} \hat{e}\left(P, H(w|_i)\right)^{\gamma \rho_{w|_i}}} = \hat{e}(P, H(\varepsilon))^{\gamma\alpha}$$

and thus decryption succeeds in recovering m. The security of this scheme is established below.

Proposition 1. *If $\mathcal{IG}$ satisfies the decisional BDH assumption, the above BTE scheme is secure against* $\mathsf{SN\text{-}CPA}$.

Proof. Assume a PPT adversary A attacking the above scheme in the SN-CPA attack scenario, and denote the probability that A succeeds by $\Pr_A[\mathsf{Succ}]$. We construct a new adversary B which attempts to solve the decisional BDH problem with respect to $\mathcal{IG}$. Relating the advantage of B to the advantage of A will give the desired result. In the description below we denote by $w|_{\bar{i}}$ the sibling of $w|_i$, namely the string consisting of the $(i-1)$-bit prefix of w, followed by the negations of the i'th bit. (In other words, $w|_i$ and $w|_{\bar{i}}$ agree on their first $i-1$ bits, and differ only the last bit.)

B is given the bound t, the output $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$ of $\mathcal{IG}$, and also $(P,\ Q = \alpha P,\ I_\varepsilon = \beta P,\ U_0 = \gamma P,\ V' = \mu P)$, where P, α, β, γ are chosen at random, and B's goal is to determine if μ was also chosen at random, or was it set to $\mu = \alpha\beta\gamma$. B attempts to simulate an instance of the encryption scheme for A as follows: B initiates a run of A, waiting for A to commit to the target node. Denote this target node by $w^* = w_1^* w_2^* \ldots w_\ell^*$ (with $\ell \leq t$). Next, for $i = 1 \ldots \min(t, \ell+1)$, B randomly chooses $\chi_i, \chi_i', \varphi_i, \varphi_i' \in \mathbb{Z}_q$. Now, B chooses the function $H : \{0,1\}^{\leq t} \to \mathbb{G}_1$ at random from the family $\mathcal{H}$, subject to the following constraints:

$H(\varepsilon) = I_\varepsilon$
$H(w^*|_i) = \chi_i P$, for $i = 1, \ldots, \ell$
$H(w^*|_{\bar{i}}) = \chi_i' P - \frac{1}{\varphi_i} I_\varepsilon$ for $i = 1, \ldots, \ell$, and
if $\ell < t$ then also $H(w0) = \chi_{\ell+1} P - \frac{1}{\varphi_{\ell+1}} I_\varepsilon$ and $H(w1) = \chi_{\ell+1}' P - \frac{1}{\varphi_{\ell+1}'} I_\varepsilon$

Since $\mathcal{H}$ was constructed to be efficiently sampleable, B can efficiently select a (random) $H \in \mathcal{H}$ satisfying the above set of at most $2t+1$ equations. Furthermore, from the point of view of A a random H chosen subject to the above constraints is distributed identically to H chosen uniformly from $\mathcal{H}$ (as in the real experiment). B sets $PK = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, Q, t, H)$ and gives PK to A.

We show how B can generate the secret key sk_w for the sibling of any node w on the path from the root to w^*, as well as for the two children of w^* in the tree if $\ell < t$. (Recall that from these secret keys, A can derive the secret key sk_w for any other node w which is not a prefix of w^*.)

For $i = 1 \dots \ell$, B sets $R_{w^*|_i} = \varphi_i Q$, and $R_{w^*|_{\bar{i}}} = \varphi'_i Q$, and if $\ell < t$ then also $R_{w^*0} = \varphi_{\ell+1} Q$ and $R_{w^*1} = \varphi'_{\ell+1} Q$. Also, for $w = w^*|_{\bar{i}}$ or $w = w^*1$, B sets

$$S_w = \varphi'_i \chi'_i Q + \sum_{j=1}^{i-1} \varphi_j \chi_j Q$$

where $i = |w|$. (For $i = 1$, the upper limit of the summation is less than the lower limit; by convention, we let the sum in this case be 0.) For $w = w^*0$, B sets $S_{w^*0} = \sum_{j=1}^{\ell+1} \varphi_j \chi_j Q$. The secret key sk_w for node $w = w^*|_{\bar{i}}$ or $w = w^*0$ or $w = w^*1$ is then just $(R_{w|_1}, R_{w|_2}, \dots, R_w, S_w)$.

We now verify that these keys have the correct distribution. For $i = 1 \dots \ell$, define $\rho_{w^*|_i} \stackrel{def}{=} \alpha\varphi_i$, and $\rho_{w^*|_{\bar{i}}} \stackrel{def}{=} \alpha\varphi'_i$, and if $\ell < t$ then also $\rho_{w^*0} \stackrel{def}{=} \alpha\varphi_{\ell+1}$ and $\rho_{w^*1} \stackrel{def}{=} \alpha\varphi'_{\ell+1}$. The ρ_w's are all random and independent in $\mathbb{Z}_q$, and we indeed have $R_w = \rho_w P$ for any $w = w^*|_i$ or $w = w^*|_{\bar{i}}$ or $w = w^*0$ or $w = w^*1$. As for the S_w's, recall that in the scheme we have for any $w = w_1 \dots w_i$ $S_w = \alpha H(\varepsilon) + \sum_{j=1}^{i} \rho_{w|_j} H(w|_j)$. For $w = w^*|_{\bar{i}}$ or $w = w^*1$, substituting $\alpha\varphi$ and $\alpha\varphi'$ for the ρ's and the right expressions for the $H(\cdot)$'s (and letting $i = |w|$), this means

$$\begin{aligned} S_w &= \alpha H(\varepsilon) + \sum_{j=1}^{i} \rho_{w|_j} H(w|_j) \\ &= \alpha I_\varepsilon + \sum_{j=1}^{i-1} \alpha\varphi_j (\chi_j P) \; + \alpha\varphi'_i \left(\chi'_i P - \frac{1}{\varphi'_i} I_\varepsilon \right) = \; \sum_{j=1}^{i-1} \varphi_j \chi_j Q \; + \varphi'_i \chi'_i Q \end{aligned}$$

For $w = w^*0$ we have the same expression ,except that χ'_i, φ'_i are replaced by χ_i, φ_i, respectively.

challenge query. B responds to the query $\mathsf{challenge}(m_0, m_1)$ by choosing random bit b and returning

$$\begin{aligned} \bar{C} &= (U_0, \chi_1 U_0, \dots, \chi_\ell U_0, \hat{e}(P, V') \cdot m_b) \;=\; (\gamma P, \chi_1 \gamma P, \dots, \chi_\ell \gamma P, \hat{e}(P, \mu P) \cdot m_b) \\ &= (\gamma P,\ \gamma H(w^*|_1),\ \dots,\ \gamma H(w^*),\ \hat{e}(P, P)^\mu \cdot m_b) \end{aligned}$$

Recalling that $Q = \alpha P$ and $H(\varepsilon) = I_\varepsilon = \beta P$, we can re-write the last component of $\bar{C}$ as $(e(Q, H(\varepsilon))^\gamma)^{\mu/\alpha\beta\gamma} \cdot m_b$. If $\mu = \alpha\beta\gamma$ then $\bar{C}$ is indeed a random encryption of m_b. When μ is random, the last element is a random element in $\mathbb{G}2$, regardless of b, and therefore $\bar{C}$ is independent of b.

Analysis of B. When $\mu = \alpha\beta\gamma$, A's view in the simulated experiment is distributed identically to A's view in the real experiment. Hence $\Pr[B \text{ outputs } 1] = \Pr_A[\mathsf{Succ}]$. On the other hand, when μ is uniformly distributed in $\mathbb{Z}_q$, A has no information about the value of b and hence it outputs $b' = b$ with probability of at most $1/2$. The advantage of B is therefore at least $\Pr_A[\mathsf{Succ}] - 1/2$. If $\mathcal{IG}$ satisfies the decisional BDH assumption, then the advantage of B is negligible, and therefore so is the advantage of A. This concludes the proofs of Proposition 1 and Theorem 1. $\square$

Efficiency parameters. In the construction above, a secret key of node w at level ℓ in the tree consists of $\ell + 1$ elements of $\mathbb{G}2$. However, only two of these elements are "new" (i.e., R_w and S_w), all the others already appear in the secret key of the parent of w. Thus, the secret keys of all the nodes on the path from the root to w can be stored using only $2\ell + 1$ group elements.

The Gen algorithm takes time polynomial in the security parameter k (to run $\mathcal{IG}$) and linear in the height bound t (to choose a $(2t + 1)$-independent hash function). The key derivation algorithm only takes a constant number of multiplications in $\mathbb{G}_1$ and two applications of the hash function $H(\cdot)$. Encryption for a node at level ℓ in the tree takes ℓ applications of the function $H(\cdot)$, ℓ multiplications in $\mathbb{G}_1$, one application of $\hat{e}(\cdot, \cdot)$ and one multiplication and one exponentiation in $\mathbb{G}_2$. (Note that it is possible to evaluate $H(\cdot)$ on ℓ points in time $O(\ell \log^2 \ell)$, [2, Section 8.5].) Decryption at the same level takes $\ell + 1$ applications of $\hat{e}(\cdot, \cdot)$, ℓ multiplications and one division in $\mathbb{G}_2$.

Applications to HIBE. We briefly note that one can construct a full-blown HIBE from any BTE scheme, simply by encoding in binary all the identities in the system (possibly using a collision-intractable hashing at every level, to improve efficiency). The full version of this work will contain a definition of SN security for HIBE, and a proof of the following theorem in the *standard model*:

Theorem 2. *If there exists an SN-CPA secure BTE scheme, then there also exists an SN-CPA secure HIBE scheme.*

Corollary 1. *There exists a HIBE scheme that is secure in the sense of SN-CPA under the BDDH assumption.*

Construction in the random oracle model. It can be seen that the above scheme remains secure when the function H is modeled as a random oracle (a proof of security is immediate since a random oracle, in particular, acts as a $(2t+1)$-wise independent function). When the random oracle is instantiated with a "cryptographic hash function" whose complexity is (essentially) independent of N (as opposed to the $O(t) = O(\log N)$ complexity of a $(2t+1)$-wise independent hash function), the complexity of several parameters of the scheme improves from $O(\log N)$ to $O(1)$.

Once we are working in the random oracle model, the scheme may be further modified so that its security is based on the computational BDH assumption rather than the decisional version: simply replace the component $M \cdot \hat{e}(Q, H(\varepsilon))^r$ of the ciphertext by $M \oplus H'(\hat{e}(Q, H(\varepsilon))^r)$, where $H' : \mathbb{G}_2 \to \{0,1\}^n$ is also modeled as an independent random oracle.

Achieving chosen-ciphertext security. We sketch how our schemes may be modified so as to achieve security against adaptive chosen-ciphertext attacks. In the standard model, we may apply the technique of Sahai ([32], based on [28]) to construct a new scheme as follows: The public key consists of two independently-generated public keys PK_1, PK_2 for a BTE scheme secure in the sense of SN-CPA, along with a random string r. To encrypt a message M for node w, the sender computes $C_1 \leftarrow \mathsf{Enc}_{PK_1}(w, M)$ and $C_2 \leftarrow \mathsf{Enc}_{PK_2}(w, M)$, and sends

$\langle w, C_1, C_2, \pi \rangle$; here, π is a simulation-sound NIZK proof of consistency (i.e., a proof that C_1 and C_2 are both encryptions of the same message M for node w) with respect to string r (generic NIZK is typically implemented using trapdoor permutations; however, it is not hard to see that the BDDH assumption is sufficient for NIZK as well). A proof that this scheme is secure against chosen ciphertext attacks follows along the lines of [32].

In the random oracle model, we can achieve a more efficient scheme which is secure in the sense of SN-CCA by applying, e.g., the Fujisaki-Okamoto transformation [16]. We note that small modifications of the Fujisaki-Okamoto transformation are necessary to achieve our goal (in particular, the node name w must be included in the hash). Further details will appear in the final version.

3 Forward-Secure Public-Key Encryption

We define and construct forward-secure encryption schemes. After defining security against chosen plaintext attacks and chosen ciphertext attacks for such schemes, we start by presenting two "trivial" schemes with linear complexity. We then describe our main construction of a CPA-secure forward secure encryption scheme with logarithmic complexity, given any CPA-secure BTE scheme. If the underlying BTE scheme is CCA-secure, then the same construction yields a forward CCA secure encryption scheme.

3.1 Definitions

We define key-evolving public-key encryption schemes, and what it means for such a scheme to be forward-secure. The former definition is a straightforward adaptation of [6] and is reminiscent of the definition of binary tree encryption; the latter, however, is new and requires some care.

Definition 3. *A* key-evolving public-key encryption scheme ke-PKE *is a 4-tuple of* PPT *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* $\langle i, C \rangle$*. It returns a message* M*. We denote this by* $M := \mathsf{Dec}_{SK_i}(\langle i, C \rangle)$*.*

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* $M = \mathsf{Dec}_{SK_i}(ENC(PK, i, M))$*.*

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

Setup*:* $\mathsf{Gen}(1^k, N)$ *is run, with output* (PK, SK_0)*. The adversary is given* PK*.*

Attack*: The adversary issues one* $\mathsf{breakin}(i)$ *query and one* $\mathsf{challenge}(j, M_0, M_1)$ *query, in either order, subject to* $0 \leq j < i < N$*. These queries are answered as:*

- *On query* $\mathsf{breakin}(i)$*, key* SK_i *is computed via* $\mathsf{Upd}(\cdots \mathsf{Upd}(SK_0, 1), \cdots, i)$*. This key is then given to the adversary.*
- *On query* $\mathsf{challenge}(j, M_0, M_1)$ *a random bit* b *is selected and the adversary is given* $C^* = \mathsf{Enc}_{PK}(j, M_b)$*.*

Guess*: The adversary outputs a guess* $b' \in \{0, 1\}$*. It* succeeds *if* $b' = b$*.*

Definition 5. *A key-evolving public-key encryption scheme is* forward-secure against chosen ciphertext attacks *(*fs-CCA*) if any* PPT *adversary has only negligible advantage in the following game:*

Setup*:* $\mathsf{Gen}(1^k, N)$ *is run, yielding* (PK, SK_0)*. The adversary is given* PK*.*

Attack*: The adversary issues one* $\mathsf{breakin}(i)$ *query, one* $\mathsf{challenge}(j, M_0, M_1)$ *query, and multiple* $\mathsf{decrypt}(k, C)$ *queries, in either order, subject to* $0 \leq j < i < N$ *and* $k \leq N$*. These queries are answered as follows:*

- *On query* $\mathsf{breakin}(i)$*, key* SK_i *is computed via* $\mathsf{Upd}(\cdots \mathsf{Upd}(SK_0, 1), \cdots, i)$*. This key is then given to the adversary.*
- *On query* $\mathsf{challenge}(j, M_0, M_1)$ *a random bit* b *is selected and the adversary is given* $C^* = \mathsf{Enc}_{PK}(j, M_b)$*.*
- *A query* $\mathsf{decrypt}(k, C)$ *is answered as follows. If a challenge query was already made (at time unit* j*),* $C = C^*$*, and* $j = k$*, then the answer is* $\perp$*. Otherwise, the answer is* $\mathsf{Dec}_{SK_k}(k, C)$*.*

Guess*: The adversary outputs a guess* $b' \in \{0, 1\}$*. It* succeeds *if* $b' = b$*.*

The advantage of the adversary is defined as the absolute value of the difference between its success probability and 1/2.

Remark 1: On the order of the breakin and the challenge queries. Definition 4 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.)

Interestingly, 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. Still, it is better to avoid it.)

Remark 2: Relaxing chosen-ciphertext security. Note that Definition 5 allows the adversary to make decryption queries for various time periods in arbitrary order (not necessarily chronological). 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. Still, we present this notion since it is the strongest natural interpretation of the CCA paradigm and our scheme in the random oracle model achieves it.

Remark 3: The random oracle model. In order to adapt our definitions to the random oracle model, we additionally allow the adversary to make a polynomially-bounded number of queries to the random oracle. These queries may be interleaved in any order with the other queries. The answers of these oracles are computed based on the same instance of the random oracle.

3.2 Schemes with linear complexity

For completeness, we discuss some simple approaches to forward-secure PKE. These approaches yield schemes with linear complexity in at least some parameters.

One trivial solution is to generate N independent public-/private- key pairs $\{(sk_i, pk_i)\}$ and to set $PK = (pk_0, \ldots, pk_{N-1})$. In this scheme, the key SK_i for time period i will simply consist of $(sk_i, \ldots, sk_{N-1})$. 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 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 (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 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 (under the public key) 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 [26] (and in particular requires $O(N)$ non-secret storage and key-generation time).

3.3 A construction with logarithmic complexity

We construct a $\mathsf{fs\text{-}CPA}$ (resp., $\mathsf{fs\text{-}CCA}$) encryption scheme from any $\mathsf{SN\text{-}CPA}$ (resp., $\mathsf{SN\text{-}CCA}$) 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 [6, 1, 27], 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. We remark that our tree traversal method can be applied also to [6, 1, 27], with similar complexity gain.

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 $w^{i+1} = w^i 0$. 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 $(\mathsf{Gen}, \mathsf{Der}, \mathsf{Enc}, \mathsf{Dec})$, construct a key-evolving scheme $(\mathsf{Gen}', \mathsf{Upd}, \mathsf{Enc}', \mathsf{Dec}')$ as follows.

- Algorithm $\mathsf{Gen}'(1^k, N)$ runs $\mathsf{Gen}(1^k)$, and obtains PK, SK_ε. It then outputs $PK' = PK$, and $SK'_0 = SK_\varepsilon$.
- Algorithm $\mathsf{Upd}(i{+}1, SK'_i)$: 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 $(SK_{w^i 0}, SK_{w^i 1}) \leftarrow \mathsf{Der}(PK, w^i, SK_{w^i})$, and push $SK_{w^i 1}$ and then $SK^i_w 0$ onto the stack. The new top is $SK_{w^i 0}$ (and indeed $w^{i+1} = w^i 0$). In either case, Upd *erases* the key SK_{w^i}.
- Algorithm $\mathsf{Enc}'(PK', i, M)$ runs $\mathsf{Enc}(PK, w^i, M)$.
- Algorithm $\mathsf{Dec}'(SK'_i, i, M)$ runs $\mathsf{Dec}(SK_{w^i}, w^i, M)$.

Theorem 3. *The scheme* $(\mathsf{Gen}', \mathsf{Upd}, \mathsf{Enc}', \mathsf{Dec}')$ *is* $\mathsf{fs\text{-}CPA}$ *secure (resp.,* $\mathsf{fs\text{-}CCA}$ *secure), assuming that the underlying scheme* $(\mathsf{Gen}, \mathsf{Der}, \mathsf{Enc}, \mathsf{Dec})$ *is a* $\mathsf{SN\text{-}CPA}$ *secure (resp.,* $\mathsf{SN\text{-}CCA}$ *secure) BTE scheme.*

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

1. A chooses at random a time period $i^* \in_{\scriptscriptstyle R} \{1..N\}$, 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.
2. A runs A', giving it public key PK.
3. When A' generates a challenge (i, M_0, M_1), if $i \neq i^*$ then A outputs a random bit and halts. If $i = i^*$ then A generates a challenge (M_0, M_1), obtains the ciphertext $C^* = \mathsf{Enc}(PK, w^{i^*}, M_b)$ and hands it over to A'.
4. 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. (Observe that SK'_i can be computed from the secret keys known to A.)
5. (This activity is only relevant to the case of CCA security.) 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'.
6. When A' outputs a prediction bit b', A outputs b' and halts.

Analyzing 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 (resp., CCA) interaction with a BTE scheme. Consequently, A predicts the bit b with advantage ε/N.

Extension to an unbounded number of time periods. In our description of the various schemes thus far, the number of time periods N was assumed to be known at the time of key generation. As in [27], we can modify our schemes to support an "unbounded" number of time periods (i.e., the number of time periods need not be known in advance). This has the added advantage that the efficiency and security of the scheme depend only on the number of periods elapsed thus far.

A proof of security for the "unbounded" scheme in the random oracle model is immediate, but in the standard model we must have an *a priori* upper-bound N^* on the total number of time periods so that (setting $t = \log N^*$) a $2t$-wise independent family $\mathcal{H}$ is used. However, this restriction is not very serious since we may set $t = \omega(\log k)$ (where k is the security parameter) and thus obtain a super-polynomial bound on the number of time periods while all parameters of the scheme remain polynomial in k.

Analysis of complexity parameters. Each of the four operations (key generation, update, encryption, 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, as discussed in Section 2. This justifies the claims given in Table 1.

Acknowledgments

The third author is very grateful to Craig Gentry for helpful discussions regarding [17] and for providing him with a preliminary version of that work.

References

1. M. Abdalla and L. Reyzin. A new forward-secure digital signature scheme. *Asiacrypt '00*, LNCS vol. 1976, pp. 116–129, Springer-Verlag, 2000.
2. A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1975.
3. R. Anderson. Two remarks on public key cryptology. Invited Lecture, ACM-CCS '97. `http://www.cl.cam.ac.uk/ftp/users/rja14/forwardsecure.pdf`.
4. D. Beaver and S. Haber. Cryptographic protocols provably secure against dynamic adversaries. In *Eurocrypt '92*, LNCS vol. 658, pp. 307–323, Springer-Verlag, 1992.
5. D. Beaver, Plug and play encryption, *Advances in Cryptology — Crypto '97*, LNCS vol. 1294, pp. 75–89, Springer-Verlag, 1997.
6. 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.
7. M. Bellare and B. Yee. Forward security in private-key cryptography. *Topics in Cryptology — CT-RSA 2003*, to appear. Preliminary version at `http://eprint.iacr.org/2001/035/`.
8. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, Relations Among Notions of Security for Public-Key Encryption Schemes, *Advances in Cryptology — Crypto'98*, Lecture Notes in Computer Science Vol. 1462, pp. 26–45, Springer-Verlag, 1998.
9. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. *Advances in Cryptology — Crypto 2001*, LNCS vol. 2139, pp. 213–229, Springer-Verlag, 2001. Full version to appear in *SIAM J. Computing* and available at `http://eprint.iacr.org/2001/090`.
10. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. *Asiacrypt '01*, LNCS vol. 2248, pp. 514–532, Springer-Verlag, 2001.
11. R. Canetti, U. Feige, O. Goldreich and M. Naor, Adaptively Secure Computation, *STOC '96*, pp. 639–648, ACM, 1996. Also MIT-LCS-TR #682, 1996.
12. I. Damgaard and J. B. Nielsen, Improved non-committing encryption schemes based on general complexity assumption, *Advances in Cryptology — Crypto '00*, LNCS vol. 1880, pp. 432–450, Springer-Verlag, 2000.
13. Y. Desmedt and Y. Frankel. Threshold cryptosystems. *Advances in Cryptology — Crypto '89*, LNCS vol. 435, pp. 307–315, Springer-Verlag, 1989.
14. W. Diffie, P. C. Van-Oorschot, and M. J. Weiner. Authentication and authenticated key exchanges. *Designs, Codes, and Cryptography* 2:107–125, 1992.
15. D. Dolev, C. Dwork and M. Naor, Non-malleable cryptography, *SIAM. J. Computing,* Vol. 30, No. 2, 2000, pp. 391-437. Preliminary version in *23rd Symposium on Theory of Computing (STOC),* ACM, 1991.
16. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Crypto '99*, LNCS 1666, pp. 537–554, Springer-Verlag, 1999.
17. C. Gentry and A. Silverberg. Hierarchical identity-based cryptography. *Asiacrypt 2002*, LNCS vol. 2501, pp. 548–566, Springer-Verlag, 2002.
18. C. G. Günther. An identity-based key-exchange protocol. *Advances in Cryptology — Eurocrypt '89*, LNCS vol. 434, pp. 29–37, Springer-Verlag, 1989.

19. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17(2):281–308, April 1988.
20. A. Herzberg, M. Jakobson, S. Jarecki, H. Krawczyk, and M. Yung. Proactive public key and signature systems. *Proceedings of 4th Conference on Computer and Communications Security*, pp. 100–110, ACM, 1997.
21. J. Horwitz and B. Lynn. Toward hierarchical identity-based encryption. *Eurocrypt '02*, LNCS vol. 2332, pp. 466–481, Springer-Verlag, 2002.
22. G. Itkis and L. Reyzin. Forward-secure signatures with optimal signing and verifying. *Crypto '01*, LNCS vol. 2139, pp. 499–514, Springer-Verlag, 2001.
23. A. Joux. A one round protocol for tripartite Diffie-Hellman. *4th International Symposium on Algorithmic Number Theory*, LNCS vol. 1838, pp. 385–394, Springer-Verlag, 2000.
24. A. Joux and K. Nguyen. Separating decision diffie-hellman from diffie-hellman in cryptographic groups. Manuscript, January 2001. Available at `http://eprint.iacr.org/2001/003/`.
25. A. Kozlov and L. Reyzin. Forward-secure signatures with fast key update. *Proc. 3rd Conference on Security in Communication Networks*, LNCS vol. 2576, pp. 247–262, Springer-Verlag, 2002.
26. H. Krawczyk. Simple forward-secure signatures from any signature scheme. *Proc. 7th ACM-CCS*, pp. 108–115, ACM, 2000.
27. 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.
28. M. Naor and M. Yung, Public key cryptosystems provably secure against chosen ciphertext attacks, *22nd STOC,* 427-437, 1990.
29. 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.
30. R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. *10th Annual Symposium on Principles of Distributed Computing*, pages 51–59, ACM, 1991.
31. 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.
32. A. Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. *Proc. of the 40th Annual Symposium on Foundations of Computer Science*, pages 543–553, IEEE, 1999.
33. A. Shamir. How to share a secret. *Comm. of the ACM* 22(11):612–613, 1979.
34. E. R. Verheul. Self-blindable credential certificates from the Weil pairing. *Asiacrypt 2001*, LNCS vol. 2248, pp. 533–551, Springer-Verlag, 2001.