Advanced Cryptographic Primitives
Lecture 6

Scribe: Henri Derycke

October 13, 2014

1 Hierarchical Identity-Based Encryption (HIBE)[5]

Although having a single Private Key Generator (PKG) would completely eliminate the need for online public key lookup, it is undesirable for a large network because generating private keys for all users becomes a bottleneck for the PKG. Hierarchical ID-based encryption (HIBE) allows a root PKG to distribute the workload by delegating private key generation and identity authentication to lower-level PKGs (users or authorities organized in hierarchy.[6]

Each node at level t has a local identifier ID_t and an address HID which is obtained as the concatenation $HID = \widetilde{HID}||ID_t$ and $\widetilde{HID}$ is the father node of HID.

Given a private key d_{HID} for $HID = (ID_1, ID_2, \ldots ID_t)$, one can compute a key $d_{HID'}$ for $HID' = (ID_1, ID_2, \ldots ID_{t+1})$ where $ID_{t+1} \in \{0,1\}^\star$

1.1 Definition of HIBE scheme

An HIBE scheme is a tuple (Setup, Keygen, Derive, Encrypt, Decrypt) with the following specifications:

Setup(λ, L)**:** Given a security parameter $\lambda \in \mathbb{N}$ and the maximal number of level $L \in poly(\lambda)$, output a pair (MPK, MSK)

Keygen$(MSK, HID = (ID_1, ID_2, \ldots ID_t))$**:** Generates a key d_{HID} if $t \leq L$ and outputs $\perp$ otherwise.

Derive(MPK, d_{HID}, HID')**:** Given a key d_{HID} for $HID = (ID_1, ID_2, \ldots ID_t)$ and another $HID = (ID'_1, ID'_2, \ldots ID'_{t+1})$, return $\perp$ if $\exists i \in \{1, \ldots t\}$ such that $ID_i \neq ID'_i$ otherwise output a derived key $d_{HID'}$ for HID'

Encrypt(MPK, M, HID)**:** output a ciphertext C for the user whose hierarchical identity is HID

Decrypt(MPK, d_{HID}, C)**:** output either a message M or $\perp$

Correctness For any $HID = (ID_1, ID_2, \ldots ID_t)$:

- For any $d_{HID'} \leftarrow$ Derive(MPK, d_{HID}, HID') where

$$HID' = (ID_1, ID_2, \ldots ID_t, ID_{t+1})$$

and for any $C \leftarrow$ Encrypt(MPK, M, HID'), $M =$ Decrypt(MPK, d_{HID}, C).

- The two distributions $D_0 = \{d_{HID'} \leftarrow \text{Derive}(MPK, d_{HID}, HID')\}$ and $D_1 = \{d_{HID'} \leftarrow \text{Keygen}(MSK, HID = (ID_1, ID_2, \ldots ID_{t+1})\}$ have to be statistically close.

1.2 Indistinguishability under chosen plaintext attack for HIBE scheme (IND-HID-CPA)

A HIBE scheme is *IND-HID-CPA* if no probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ with non-negligible advantage wins this game:

1. The challenger generates $(MPK, MSK) \leftarrow \text{Setup}(\lambda, L)$, gives MPK to the adversary $\mathcal{A}$ and initializes a set $Q = \emptyset$

2. $\mathcal{A}$ makes private key queries:

   - $\mathcal{A}$ chooses a hierarchical identity $HID = (ID_1, ID_2, \ldots ID_t)$ where $t \leq L$

   - The challenger returns a key $d_{HID} \leftarrow \text{Keygen}(MSK, HID)$ and updates $Q := Q \cup \{HID\}$

3. $\mathcal{A}$ chooses M_0, M_1, and $HID^\star$ such that no prefix of $HID^\star$ is in Q and obtains $C^\star \leftarrow \text{Encrypt}(MPK, M_\gamma, HID^\star)$ where $\gamma \leftarrow^R \{0, 1\}$

4. $\mathcal{A}$ makes more private key queries under the restriction that no prefix of $HID^\star$ can be in Q at any time.

5. $\mathcal{A}$ outputs γ' and wins if $\gamma = \gamma'$

The advantage of $\mathcal{A}$ on this game is:

$$\text{Adv}(\mathcal{A}) := |\Pr[\gamma = \gamma'] - 1/2|$$

1.3 Hierarchical extension of the BF-IBE [5]

The scheme hereunder, due to Gentry and Silverberg [5], extends the Boneh-Franklin IBE scheme in a natural way.

Setup(λ, L):

1. Choose cyclic groups (G, G_T) of prime order $p > 2^\lambda$ with a bilinear map $e : G \times G \to G_T$ and a generator $g \xleftarrow{R} G$

2. Choose $\alpha \xleftarrow{R} \mathbb{Z}_p$ and compute $g_1 = g^\alpha$

3. Choose a hash function $H : \{0, 1\}^* \to G$

Define $MPK := ((G, G_T), g, g_1 = g^\alpha, H)$ and $MSK := \alpha$

Keygen($MSK, HID = (ID_1, ID_2, \ldots ID_t)$): Given $MSK = \alpha$

- Choose $r_2, r_3, \ldots r_t \xleftarrow{R} \mathbb{Z}_p$
- Compute $d_{HID} = (d_1, \ldots, d_t) \in G^t$ where

$$d_1 = H(ID_1)^\alpha \cdot \prod_{i=2}^{t} H(ID_1, ID_2, \ldots, ID_i)^{r_i}$$

$$d_i = g^{r_i} \qquad \forall 2 \le i \le t$$

- Return d_{HID}

Derive(MPK, d_{HID}, HID'): Given a private key $d_{HID} = (d_1, \ldots d_t)$ for the identity $HID = (ID_1, ID_2, \ldots ID_t)$ and $HID' = (ID_1, ID_2, \ldots ID_t, ID_{t+1})$

- Choose $r'_2, r'_3, \ldots r'_{t+1} \xleftarrow{R} \mathbb{Z}_p$
- Compute $d_{HID'} = (d'_1, \ldots, d'_{t+1}) \in G^{t+1}$ where

$$d'_1 = d_1 \cdot \prod_{i=2}^{t+1} H(ID_1, ID_2, \ldots, ID_i)^{r'_i}$$

$$d'_i = d_i \cdot g^{r_i} \qquad \forall 2 \le i \le t$$

$$d'_{t+1} = g^{r'_{t+1}}$$

- Return $d_{HID'}$

Encrypt(MPK, M, HID): To encrypt $M \in G_T$

- Choose $s \xleftarrow{R} \mathbb{Z}_p$
- Compute $c = (c_0, \ldots c_t) \in G_T \times G^t$ where

$$c_0 = M \cdot e(g_1, H(ID_1))^s$$

$$c_1 = g^s$$

$$c_i = H(ID_1, ID_2, \ldots, ID_i)^s \qquad \forall 2 \le i \le t$$

- Output c

Decrypt(MPK, d_{HID}, C): Given $d_{HID} = (d_1, \ldots d_t)$ and $c = (c_0, \ldots c_t)$,

- Compute

$$M = c_0 \cdot \frac{\prod_{i=2}^{t} e(c_i, d_i)}{e(c_1, d_1)}$$

- Output M

Correctness. for any well-formed private key $d_{HID} = (d_1, \ldots, d_t)$, we have

$$e(d_1, g) = e(g_1, H(ID_1)) \cdot \prod_{i=2}^{t} e(H(ID_1 \cdot ID_2 \cdot \ldots ID_i), d_i),$$

so that

$$e(d_1, g^s) = e(g_1, H(ID_1))^s \cdot \prod_{i=2}^{t} e(H(ID_1 \cdot ID_2 \cdot \ldots ID_i)^s, d_i),$$

and then

$$M \cdot e(d_1, \underbrace{g^s}_{c_1}) = \underbrace{M \cdot e(g_1, H(ID_1))^s}_{c_0} \cdot \prod_{i=2}^{t} e(\underbrace{H(ID_1 \cdot ID_2 \cdot \ldots ID_i)^s}_{c_i}, d_i),$$

which explains why the decryption algorithm correctly decrypts $c = (c_0, \ldots c_t)$.

Theorem ([5])**.** *In the Random Oracle Model (ROM), any PPT adversary $\mathcal{A}$ with advantage ϵ against the IND-HID-CPA security of the scheme implies a PPT DBDH distinguisher $\mathcal{B}$ with advantage*

$$\epsilon \geq \frac{1}{e^L \cdot (q+1)^2}$$

where L is the maximal number of levels and q is the number of private key queries.

1.4 Hierarchical extension of Boneh-Boyen IBE

In standard model, we can use the Boneh-Boyen IBE to construct a HIBE scheme with selective security.

Setup(λ, L)**:**

1. Choose cyclic groups (G, G_T) of prime order $p > 2^\lambda$ with a bilinear map $e : G \times G \to G_T$ and a generator $g \stackrel{R}{\leftarrow} G$
2. Choose $\alpha \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and compute $g_1 = g^\alpha$
3. Choose $g_2, h_1, \ldots, h_L \stackrel{R}{\leftarrow} G$

Define $MPK := \big((G, G_T), g, g_1 = g^\alpha, g_2, \{h_i\}_{i=1}^{L}\big)$ and $MSK := g_2^\alpha$

Keygen$(MSK, HID = (ID_1, ID_2, \ldots ID_t))$**:** Given $MSK = g_2^\alpha$ and the hierarchical identity $HID = (ID_1, ID_2, \ldots ID_t)$,

- Choose $r_1, r_2, \ldots r_t \stackrel{R}{\leftarrow} \mathbb{Z}_p$
- Compute $d_{HID} = (d_0, \ldots, d_t) \in G^{t+1}$ where

$$d_0 = g_2^\alpha \cdot \prod_{i=1}^{t} H_i(ID_i)^{r_i}$$

$$d_i = g^{r_i} \qquad \forall 1 \leq i \leq t$$

  and with

$$H_i(ID_i) = g_1^{ID_i} h_i \qquad \forall 1 \leq i \leq t$$

- Return d_{HID}

Derive(MPK, d_{HID}, HID'): Given a private key $d_{HID} = (d_1, \ldots d_t)$ for the identity $HID = (ID_1, ID_2, \ldots ID_t)$ and $HID' = (ID_1, ID_2, \ldots ID_t, ID_{t+1})$

- Choose $r'_2, r'_3, \ldots r'_t, r'_{t+1} \stackrel{R}{\leftarrow} \mathbb{Z}_p$
- Compute $d_{HID'} = (d'_0, d'_1, \ldots, d'_{t+1}) \in G^{t+2}$ where

$$d'_0 = d_0 \cdot H_{t+1}(ID_{t+1})^{r'_{t+1}} \cdot \prod_{i=1}^{t} H_i(ID_i)^{r'_i}$$
$$d'_i = d_i \cdot g^{r_i} \qquad \forall 2 \leq i \leq t$$
$$d'_{t+1} = g^{r'_{t+1}}$$

- Return $d_{HID'}$

Encrypt(MPK, M, HID): To encrypt $M \in G_T$

- Choose $s \leftarrow^R \mathbb{Z}_p$
- Compute $c = (c_0, \ldots c_t, c_{t+1}) \in G^{t+1} \times G_T$ where

$$c_0 = g^s$$
$$c_i = H_i(ID_i)^s \qquad \forall 1 \leq i \leq t$$
$$c_{t+1} = M \cdot e(g_1, g_2)^s$$

- Output c

Decrypt(MPK, d_{HID}, C): Given $d_{HID} = (d_1, \ldots d_t)$ and $c = (c_0, \ldots c_t)$,

- Compute

$$M = c_{t+1} \cdot \frac{\prod_{i=1}^{t} e(c_i, d_i)}{e(c_0, d_0)}$$

- Output M

Correctness. For any valid private key $d_{HID} = (d_0, d_1, \ldots, d_t)$, we have the equality

$$e(d_0, g) = e(g_1, g_2) \cdot \prod_{i=1}^{t} e(H_i(ID_i), d_i),$$

which implies

$$e(d_0, g^s) = e(g_1, g_2)^s \cdot \prod_{i=1}^{t} e(H_i(ID_i)^s, d_i)$$

for any $s \in \mathbb{Z}_p$. Hence, we find

$$M \cdot e(d_0, \underbrace{g^s}_{c_0}) = \underbrace{M \cdot e(g_1, g_2)^s}_{c_{t+1}} \cdot \prod_{i=1}^{t} e(\underbrace{H_i(ID_i)^s}_{c_i}, d_i),$$

which explains whyt $c = (c_0, \ldots c_t, c_{t+1})$ is correctly decrypted by the decryption algorithm.

Theorem ([2])**.** *This scheme is IND-sID-CPA secure under the DBDH assumption.*

1.5 HIBE with short ciphertexts [3]

Boneh, Boyen and Goh showed how to construct a HIBE scheme with short ciphertexts [3].

Setup(λ, L)**:**

1. Choose cyclic groups (G, G_T) of prime order $p > 2^\lambda$ with a bilinear map $e : G \times G \to G_T$ and a generator $g \xleftarrow{R} G$
2. Choose $\alpha \xleftarrow{R} \mathbb{Z}_p$ and compute $g_1 = g^\alpha$
3. Choose $g_2, h_0, h_1, \ldots, h_L \xleftarrow{R} G$

Define $MPK := \big((G, G_T), g, g_1 = g^\alpha, g_2, \{h_i\}_{i=0}^{L}\big)$ and $MSK := g_2^\alpha$

Keygen($MSK, HID = (ID_1, ID_2, \ldots ID_t)$)**:** Given $MSK = g_2^\alpha$ and the hierarchical identity $HID = (ID_1, ID_2, \ldots ID_t)$,

- Choose $r \xleftarrow{R} \mathbb{Z}_p$
- Compute

$$d_{HID} = (D_0, D_1, K_{t+1}, \ldots, K_L) = \left(g_2^\alpha \cdot \left(h_0 \prod_{i=1}^{t} h_i^{ID_i}\right)^r, g^r, h_{t+1}^r, \ldots, h_L^r\right) \in G^{L-t+2}$$

- Return d_{HID}

Derive(MPK, d_{HID}, HID')**:** Given a private key $d_{HID} = (D_0, D_1, K_{t+1} \ldots K_L)$ for the identity $HID = (ID_1, ID_2, \ldots ID_t)$ and $HID' = (ID_1, ID_2, \ldots ID_t, ID_{t+1})$

- Choose $r' \xleftarrow{R} \mathbb{Z}_p$
- Compute $d_{HID'} = (D'_0, D'_1, K'_{t+2}, \ldots, K'_L)$ where

$$D'_0 = D_0 \cdot K_{t+1}^{ID_{t+1}} \cdot \left(h_0 \prod_{i=1}^{t+1} h_i^{ID_i}\right)^{r'}$$

$$D'_1 = D_1 \cdot g^{r'}$$

$$K'_i = K_i \cdot h_i^{r'} \qquad \forall t+2 \leq i \leq L$$

- Return $d_{HID'}$

Encrypt(MPK, M, HID)**:** To encrypt $M \in G_T$

- Choose $s \xleftarrow{R} \mathbb{Z}_p$
- Compute

$$c = (c_0, c_1, c_2) = \left(g^s, \left(h_0 \prod_{i=1}^{t} h_i^{ID_i}\right)^s, M \cdot e(g_1, g_2)^s\right)$$

- Output c

Decrypt(MPK, d_{HID}, C): Given $d_{HID} = (D_0, D_1, K_{t+1}, \ldots, K_L)$ and the ciphertext $c = (c_0, c_1, c_2)$,

- Compute

$$M = c_2 \cdot \frac{e(c_1, D_1)}{e(c_0, D_0)}$$

- Output M

Correctness. For any well-formed private key $d_{HID} = (D_0, D_1, K_{t+1}, \ldots, K_L)$, we have

$$e(d_0, g) = e(g_1, g_2) \cdot e\left(\left(h_0 \prod_{i=1}^{t} h_i^{ID_i}\right), d_1\right),$$

so that

$$e(d_0, g^s) = e(g_1, g_2)^s \cdot e\left(\left(h_0 \prod_{i=1}^{t} h_i^{ID_i}\right)^s, d_1\right)$$

for any $s \in \mathbb{Z}_p$. It follows that

$$M \cdot e(d_0, \underbrace{g^s}_{c_0}) = \underbrace{M \cdot e(g_1, g_2)^s}_{c_2} \cdot e\left(\underbrace{\left(h_0 \prod_{i=1}^{t} h_i^{ID_i}\right)^s}_{c_1}, d_1\right),$$

which explains the decryption algorithm.

Remarks. Since there are fewer ciphertext components to compute, the encryption algorithm is faster and so is the decryption algorithm since only two pairing evaluations are sufficient. Another property of the scheme is that, unlike previous HIBE schemes, the size of the private key decreases at each key delegation.

Theorem ([3])**.** *The above HIBE scheme is IND-sHID-CPA secure if the weak L-Decision Bilinear Diffie-Hellman Inversion assumption holds.*

Definition 1 ([3])**.** *The weak L-Decision Bilinear Diffie-Hellman Inversion (L-wDBDHI) assumption says that, given*

$$(g, h, g^a, g^{(a^2)}, \ldots, g^{(a^L)}, T) \in G^{L+2} \times G_T,$$

where $g, h \stackrel{R}{\leftarrow} G$ *and* $a \stackrel{R}{\leftarrow} \mathbb{Z}_p$*, deciding whether* $T = e(g,h)^{1/a}$ *or* $T \in_R G_T$ *is hard.*

2 Application of HIBE : forward-secure encryption

2.1 Forward Security [1]

- The lifetime of a public key is divided into time periods $0, 1, ...T-1$

- Each period uses a different SK_i: at the beginning of period i, SK_{i-1}, is erased and replaced by an updated key SK_i
- In case of key exposure at period i, the current private key SK_i is compromised but $SK_0, \dots SK_{i-1}$ should remain infeasible to compute for the adversary.

Definition. *A Forward-Secure Public Key Encryption (FS-PKE) scheme is a tuple of algorithms:*

Keygen(λ, T)**:** *output a public key PK and an initial SK_0*

Update(SK_i, PK) : *if $i = T-1$ return $\perp$, otherwise return SK_{i+1} and erase SK_{i+1}*

Encrypt(PK, M, i)**:** *output a ciphertext c for M*

Decrypt(SK_i, i, c)**:** *output a message M for c or $\perp$*

Definition. *A FS-PKE is IND-CPA secure if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game:*

1. *The challenger generates $(PK, SK_0) \leftarrow Keygen(\lambda, T)$ and gives PK to $\mathcal{A}$*
2. *$\mathcal{A}$ makes exactly one query to each one of these two oracles:*
    - *Break-in (i): for the period $i \in \{1 \dots T-1\}$, $\mathcal{A}$ obtains SK_i*
    - *Challenge (j, M_0, M_1): for a time period $j \in \{0 \dots T-1\}$ and equal-length messages M_0, M_1, the adversary $\mathcal{A}$ obtains a challenge ciphertext $C_j = Encrypt(PK, M_\gamma, j)$ where $\gamma \leftarrow^R \{0,1\}$*

    under the constraint that $0 \leq j < i < T$
3. *$\mathcal{A}$ outputs $\gamma' \in \{0,1\}$ and wins if $\gamma' = \gamma$*

The advantage of $\mathcal{A}$ in this game is:

$$Adv_{\mathcal{A}}^{FS\text{-}PKE}(\lambda) = |\Pr[\gamma' = \gamma] - 1/2|$$

2.2 FS-PKE from IBE

It is known [4] that one can obtain a limited construction of FS-PKE scheme using any IBE scheme.

Keygen(λ, T)**:**

- Generate $(MPK, MSK) \leftarrow \text{Setup}^{IBE}(\lambda)$
- Set $PK^{FS} := MPK$
- For each $i \in \{0 \dots T-1\}$, compute $SK_i \leftarrow \text{Keygen}^{IBE}(MSK, i)$
- Set $SK_0^{FS} := \{SK_0, \dots, SK_{T-1}\}$

Update(SK_i^{FS}, PK^{FS}):

- Parse SK_i^{FS} as $\{SK_i, \ldots, SK_{T-1}\}$
- Output $SK_{i+1}^{FS} := \{SK_{i+1}, \ldots, SK_{T-1}\}$ and erase SK_i^{FS}

Encrypt(PK^{FS}, M, i):

- Compute $c = \text{Encrypt}^{IBE}(MPK, M, i)$

Decrypt(SK_i^{FS}, i, c)):

- Parse SK_i^{FS} as $\{SK_i, \ldots, SK_{T-1}\}$
- Compute $M = \text{Decrypt}^{IBE}(MPK, SK_i, c)$

The limitation of the latter construction is that private keys have size $O(T)$. The key generation phase also takes time $O(T)$. It is desirable to have a construction where the complexity is at most poly-logarithmic in T in all performance metrics.

2.3 FS-PKE with poly-logarithmic complexity in T from any Selectively Secure HIBE [4]

Consider a binary tree with $L = \log T$ levels. In the tree, each node at depth ℓ has an ℓ-bit label. The root of the tree, at depth 0, has the empty string ϵ.

We 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, $w^0 = \epsilon$ and if w^i is an internal node then $w^{i+1} = w_i0$. If w^i is a leaf node and $i < N - 1$ then $w^{i+1} = w'1$ where w' is the longest string such that $w'0$ is a prefix of w^i.) 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.

Keygen(λ, T):

- Generate $(MPK, MSK) \leftarrow \text{Setup}^{HIBE}(\lambda, L)$, where $L = \log(T)$
- Define $SK_\epsilon = MSK$
- Set $PK^{FS} := MPK$ and $SK_0^{FS} := \{SK_\epsilon\}$

Update(SK_i^{FS}, PK^{FS}):

- Parse SK_i^{FS} as a stack of SK_i with SK_{w^i} on the top.
- Pop SK_{w^i} from the stack,
  - if w^i is a internal node, compute
  $$SK_{w^i0} \leftarrow \text{Derive}^{HIBE}(MPK, SK_{w^i}, w^i0)$$
  and $SK_{w^i1} \leftarrow \text{Derive}^{HIBE}(MPK, SK_{w^i}, w^i1)$, push SK_{w^i1} then SK_{w^i0} on the stack
  - if w^i is a leaf, the next key on top of the stack is $SK_{w^{i+1}}$
- Set SK_{i+1}^{FS} := the new stack

Encrypt(PK^{FS}, M, i)**:**

- Compute $c = \text{Encrypt}^{HIBE}(MPK, M, w^i)$

Decrypt(SK_i^{FS}, i, c)**:**

- Compute $M = \text{Decrypt}^{HIBE}(MPK, SK_{w^i}, c)$ (Note that SK_{w^i} is stored as a part of SK_i^{FS})

Theorem ([4])**.** *The above FS-PKE is IND-CPA secure if the underlying HIBE is IND-sHID-CPA secure.*

Remarks:

- The number T of time periods is assumed to be polynomial in λ to guarantee a polynomial reduction in the above theorem.
- Private keys SK_i^{FS} consist of $O(\log T)$ HIBE private keys.
- Ciphertext size is the same as in the HIBE.
    - The Boneh-Boyen HIBE implies a FS-PKE with ciphertexts of size $O(\log T)$ and private keys of size $O(\log^2 T)$.
    - The Boneh-Boyen-Goh HIBE implies a FS-PKE with ciphertexts of size $O(1)$ and private keys of size $O(\log^2 T)$.
- The Canetti-Halevi-Katz construction [4] assigns time periods to all nodes of the tree in order to have faster key update and key generation algorithms (their complexity reduces from $O(\log T)$ to $O(1)$). It is also possible to only assign time periods to the leaves of the tree. This was the approach taken in [7].

References

[1] R. Anderson. Two remarks on public key cryptology. *ACM CCCS '97*, 1997.

[2] D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT*, pages 223–238, 2004.

[3] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology–EUROCRYPT 2005*, pages 440–456. Springer Berlin Heidelberg, 2005.

[4] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In E. Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer Berlin Heidelberg, 2003.

[5] C. Gentry and A. Silverberg. Hierarchical id-based cryptography. In Y. Zheng, editor, *Advances in Cryptology — ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer Berlin Heidelberg, 2002.

[6] J. Horwitz and B. Lynn. Toward hierarchical identity-based encryption. In L. Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer Berlin Heidelberg, 2002.

[7] J. Katz. A forward-secure public-key encryption scheme. Cryptology ePrint Archive: Report 2002/060, 2002.