Hierarchy Integrated Signature and Encryption

(Key Separation vs. Key Reuse: Enjoy the Best of Both Worlds)

Yu Chen  Qiang Tang  Yuyu Wang

ASIACRYPT 2021
Outline

1. Background

2. Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3. Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4. Comparison and Experimentation

5. Summary
Outline

1 Background

2 Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI coversion)

3 Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4 Comparison and Experimentation

5 Summary
PKE and SIG are “workhorse” primitives that are typically used simultaneously to secure communication

- **PKE** ⇒ protect confidentiality
- **SIG** ⇒ protect authenticity: data integrity & authenticated data source

**Classical examples**

- Secure communication software: PGP, WhatsApp
- Privacy-preserving cryptocurrency: Zcash, Zether, PGC

**A Subtle Point:** Joint security (somewhat akin to UC)

- EUF-CMA security for SIG: holds even in the presence of $O_{\text{dec}}$
- IND-CCA security for PKE: holds even in the presence of $O_{\text{sign}}$
Two Principal When Using PKE and SIG

Key Separation vs. Key Reuse
Key Separation: Cartesian-Product Combined Public-Key Scheme

SIG

\[ sk \]

\[ vk \]

PKE

\[ dk \]

\[ ek \]

Engineering folklore: using different keypairs for different cryptographic operations

Pros

- joint security is immediate & construction is off-the-shelf
- naturally admits individual key escrow \( \rightsquigarrow \) achieve a balance between user’s authenticity requirement and society’s auditing requirement

Cons

- double key management complexity and certificate cost\(^1\)
- complicate the design of high-level protocol: tricky address derivation

\(^1\)Certificate costs include but not limit to registration, issuing, storage, transmission, verification, and building/recurring fees.
Key Reuse: Integrated Signature and Encryption

Pros
- reduce key management complexity, certificate cost, and cryptographic footprint
- simplify the design of high-level protocol

Cons
- joint security is not immediate (consider textbook RSA) & require careful design
- does not admit individual key escrow
- does not admit classified protection of secret keys

Deployed in EMV standard, Ping Identity, Zether and PGC
Motivation

We are facing a dilemma between key reuse that brings performance benefit and key separation that supports individual key escrow.

Can we enable individual key escrow mechanism while retaining the merits of key reuse?
Outline

1 Background

2 Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3 Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4 Comparison and Experimentation

5 Summary
Hierarchy Integrated Signature and Encryption

- Setup($1^λ$) → $pp$
- KeyGen($1^λ$) → ($pk$, $sk$). $pk$ serves as encryption and verification key; $sk$ is the signing key, serving as master secret key.
- Derive($sk$) → $dk$ used only for decryption
- Enc($pk$, $m$) → $c$
- Dec($dk$, $c$) → $m$
- Sign($sk$, $\tilde{m}$) → $σ$
- Vefy($pk$, $\tilde{m}$, $σ$) → 0/1
Strong Joint Security

**IND-CCA security in the presence of a signing oracle (unrestricted access)**

\[
\Pr \left[ b = b' : \begin{array}{l}
    pp \leftarrow \text{Setup}(1^\lambda); \\
    (pk, sk) \leftarrow \text{KeyGen}(pp); \\
    (m_0, m_1) \leftarrow \mathcal{A}_{\text{dec}, \text{sign}}(pp, pk); \\
    b \leftarrow \{0, 1\}, c^* \leftarrow \text{Enc}(pk, m_b); \\
    b' \leftarrow \mathcal{A}_{\text{dec}, \text{sign}}(c^*);
\end{array} \right] - \frac{1}{2} \leq \text{negl}(\lambda).
\]

**EUF-CMA security in the presence of a decryption key**

\[
\Pr \left[ \text{Vrfy}(pk, m^*, \sigma^*) = 1 \land m^* \notin Q : \begin{array}{l}
    pp \leftarrow \text{Setup}(1^\lambda); \\
    (pk, sk) \leftarrow \text{KeyGen}(pp); \\
    dk \leftarrow \text{Derive}(sk); \\
    (m^*, \sigma^*) \leftarrow \mathcal{A}_{\text{sign}}(pp, pk, dk);
\end{array} \right] \leq \text{negl}(\lambda).
\]
Application of HISE

Merit of HISE
- compact public key size
- reduce key management complexity
- simplify the design and analysis of high-level protocols

suitable for scenarios that simultaneously require privacy, authenticity and key escrow

Zether/PGP

\((pk, sk)\)

outsource costly operations e.g., expensive decryption

auditing
Application of HISE

Merit of HISE

- compact public key size
- reduce key management complexity
- simplify the design and analysis of high-level protocols

suitable for scenarios that simultaneously require privacy, authenticity and key escrow

Zether/PGP

outsource costly operations e.g., expensive decryption

delegate decryption capability

\((pk, sk)\)

sk

compromise the security of SIG

auditing
Application of HISE

Merit of HISE
- compact public key size
- reduce key management complexity
- simplify the design and analysis of high-level protocols

suitable for scenarios that simultaneously require privacy, authenticity and key escrow

Zether/PGP

\[(pk, sk)\]

delegate decryption capability

\[dk\]

retain the security of SIG

outsource costly operations e.g., expensive decryption

auditing
Outline

1. Background

2. Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3. Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4. Comparison and Experimentation

5. Summary
Paterson et al. [PSST11] give an elegant ISE construction from IBE.

ISE from IBE does not lend itself to HISE

\[ I_0 = 0|v, v \in \{0, 1\}^n \]

\[ I_1 = 1|v, v \in \{0, 1\}^n \]

\[ (mpk, msk) \]

Naor transform

\[ \sigma \leftarrow \text{Extract}(msk, 0|\tilde{m}) \]

\[ m \leftarrow M \]

\[ c \leftarrow \text{Enc}(mpk, 0|\tilde{m}, m) \]

\[ m \overset{?}{=} \text{Dec}(\sigma, c) \]

\[ (vk, sk) \leftarrow \text{OTS.KeyGen}(1^\lambda) \]

\[ c \leftarrow \text{Enc}(mpk, 1|vk, m) \]

\[ \sigma \leftarrow \text{OTS.Sign}(sk, c) \]

\[ 1 \overset{?}{=} \text{OTS.Vefy}(vk, c, \sigma); \]

\[ m \leftarrow \text{Dec}(sk_{1|vk}, c) \]

\[ \text{bit prefix partition trick} \quad \Rightarrow \text{joint security} \]
HISE from Constrained IBE for Prefix Predicate

Main idea: $\text{msk}$ acts as $\text{sk}$, secret keys for identities in $I_1$ as decryption key

Technical hurdle: decryption key should be short $\Rightarrow$ we need a succinct representation for all secret keys for identities in $I_1 \Leftarrow$ constrained IBE for prefix predicates $\Leftarrow$ BTE

$$I_0 = 0|v, v \in \{0, 1\}^n$$

$$\sigma \leftarrow \text{Extract}(\text{msk}, 0|\tilde{m})$$

$$m \overset{R}{\leftarrow} M$$

$$c \leftarrow \text{Enc}(\text{mpk}, 0|\tilde{m}, m)$$

$$m = \text{Dec}(\sigma, c)$$

$$I_1 = 1|v, v \in \{0, 1\}^n$$

$$\text{bit prefix partition trick} \Rightarrow \text{joint security}$$

$$(\text{pk}, \text{sk}) \leftarrow \text{OTS.KeyGen}(1^\lambda)$$

$$c \leftarrow \text{Enc}(\text{mpk}, 1|vk, m)$$

$$\sigma \leftarrow \text{OTS.Sign}(sk, c)$$

$$1 = \text{OTS.Vefy}(vk, c, \sigma)$$

$$m \leftarrow \text{Dec}(sk_1|vk, c)$$
The above generic construction from constrained IBE enjoys joint security in the standard model.

- constrained IBE is still less efficient

In applications where IND-CPA security suffice, or one is willing to accept IND-CCA security in the random oracle model, we have a simpler and more efficient construction of HISE from any IBE.
Outline

1 Background

2 Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3 Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4 Comparison and Experimentation

5 Summary
**HISE from PKE and NIZKPoK (HI conversion)**

**Goal:** add signing functionality to PKE in a generic manner
- bootstrap PKE in-use to HISE \(\rightsquigarrow\) enables a seamless upgrade

**Idea:** create hierarchical key structure via OWF

1. picks \(sk \leftarrow \{0, 1\}^n\) as signing key
2. maps \(sk\) to randomness \(r\) via uniform OWF: \(F(sk) \rightarrow r\)
3. runs PKE.KeyGen\((r) \rightarrow (pk, dk)\)

![Diagram: The hierarchical key structure](image)

**Figure:** The hierarchical key structure
HISE from PKE and NIZKPoK (HI conversion)

The encryption component of HISE is simple: same as that of the underlying PKE.

But, we are facing the following technical hurdle when designing signature:
- \( sk \) is unstructured bit string, how to create the signing functionality?
- the signature should remain secure even in the presence of \( dk \) (partial leakage of \( sk \)) \( \Rightarrow \) strong joint security

Solution
- using general-purpose public-coin ZKPoK to prove knowledge of \( sk \) given \( pk \)
- require \( R_{\text{key}} \) to be leakage-resilient one-way w.r.t. leakage \( r \) and thus certainly \( dk \)
- minimum requirement on G: target-collision resistant

Strong joint security:
- SIG component: Sigma protocol for leakage-resilient one-way relation \( \sim \) leakage-resilient SIG
- PKE component: zero-knowledge property \( \sim \) \( O_{\text{sign}} \) is useless + uniformity of F admits security reduction to the underlying PKE
The above construction is still less practical for real world applications. The bottleneck lies at general-purpose ZKPoK.

- We left more efficient instantiation as an interesting open problem.
Outline

1 Background

2 Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3 Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4 Comparison and Experimentation

5 Summary
Motivation of Global Escrow

Motivating example: large-scale collaborative working Apps such as Slack is getting popular ⇒ encrypted communication may contain proprietary information

- employer may have the right to get access to all private communications for various reasons
  - naive solution: collect individual decryption key one by one ⇒ impractical and inefficient
- employees need to be assured that even a malicious employer cannot slander them by forging signatures for fabricated communications

We further expect global escrow property

- there is a “super” key that can decrypt any ciphertext under any public key
- signature remains secure even in the presence of the “super” key

To attain global escrow property for HISE in a generic manner, we first take a detour to revisit global escrow PKE.
Global escrow PKE: an escrow agent holds a global escrow decryption key that can decrypt ciphertexts encrypted under any public key

\[(pk_1, sk_1) \quad \ldots \quad (pk_i, sk_i) \quad \ldots \quad (pk_n, sk_n)\]

The state of the art of global escrow PKE is less satisfactory
- long overdue for formal definition and generic construction
- the only known practical scheme is the escrow ElGamal PKE proposed by Boneh and Franklin from bilinear maps
**Formal Definition**

$$(epk, edk) \leftarrow \text{Setup}(1^\lambda)$$

$c \leftarrow \text{Enc}(pk_r, m)$

$m \leftarrow \text{Dec}(edk, c)$ \quad || \quad m \leftarrow \text{Dec}(sk_r, c)$

**Correctness:** honestly generated CTs decrypting to the same result under $edk$ and $sk_r$

**Consistency:** no PPT adversary can generate an ill-formed CT decrypting different results under $edk$ and $sk_r$

**Failure attempts**

1. Identity-based encryption: does not directly lend itself to global escrow PKE (users must be able to generate keypairs themselves)
2. Broadcast encryption: sender could be malicious especially when he has incentive to evade oversight
Outline

1. Background

2. Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3. Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4. Comparison and Experimentation

5. Summary
Global Escrow PKE from PKE and NIZK (GE conversion)

escrow center

\[ pp_{\text{nizk}} \leftarrow \text{NIZK.Setup}(1^\lambda) \]
\[ pp_{\text{pke}} \leftarrow \text{PKE.KeyGen}(1^\lambda) \]
\[ (epk, edk) \leftarrow \text{PKE.KeyGen}(pp_{\text{pke}}) \]
\[ pp = (pp_{\text{nike}}, pp_{\text{pke}}, epk), edk \]

NIZK.\text{Verify}(c_1, c_2, \pi) \stackrel{?}{=} 1
\[ m \leftarrow \text{PKE.Dec}(edk, c_2) \]

sender

\[ c_1 \leftarrow \text{PKE.Enc}(pk, m; r_1) \]
\[ c_2 \leftarrow \text{PKE.Enc}(epk, m; r_2) \]

\[ \pi \leftarrow \text{NIZK.Prove}(c_1, c_2, (m, r_1, r_2)) \]

receiver

NIZK.\text{Verify}(c_1, c_2, \pi) \stackrel{?}{=} 1
\[ m \leftarrow \text{PKE.Dec}(sk, c_1) \]

Give a generic approach to compile any PKE into global escrow PKE

- enrich the application scope of the Naor-Yung transform beyond CCA security
- achieve CCA security with no overhead
Efficient Instantiation - Global Escrow PKE Scheme 1

Choices of primitives
- **PKE**: ElGamal PKE in EC groups
- **NIZK**: Groth-Sahai proof in standard model or Sigma proof in random oracle model

**Improvement**
- When PKE satisfies the “randomness fusion” property [BMV16], we can safely reuse the randomness and then apply twisted Naor-Yung transform ⇒ better efficiency

plenty of PKE schemes from the DDH, quadratic residuosity, and subset sum assumptions satisfy randomness fusion property.
Outline

1. Background

2. Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3. Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4. Comparison and Experimentation

5. Summary
**Multiparty NIKE**

\[ \text{Setup}(1^\lambda, n) \rightarrow pp_{\text{nike}} \]

\[ \text{ShareKey}(sk_i, \{pk_1, \ldots, pk_n\}) \rightarrow k \]

- \( n = 2 \): Diffie-Hellman key exchange \([\text{DH76}]\)
- \( n = 3 \): Joux’s key exchange \([\text{Jou04}]\) from bilinear maps
- \( n \) is any positive integer
  - Boneh and Silverberg \([\text{BS02}]\) using multilinear maps
  - Alamati et al. \([\text{AMPR19}]\) using composable input homomorphic weak PRF

**mild property**

\( PK \) is efficiently recognizable
Global Escrow PKE from 3-party NIKE

\[ pp_{\text{nike}} \leftarrow \text{NIKE.Setup}(1^\lambda, 3) \]
\[ (pk_\gamma, sk_\gamma) \leftarrow \text{NIKE.KeyGen}(pp_{\text{nike}}) \]
\[ pp = (pp_{\text{nike}}, pk_\gamma), edk = sk_\gamma \]

\[ k \leftarrow \text{NIKE.ShareKey}(sk_\gamma, S') \]
\[ m \leftarrow \text{SKE.Dec}(k, c) \]

- security of NIZK (pseudorandomness of shared key \( k \)) \( \Rightarrow \) IND-CPA/CCA security
- \( PK \) is efficiently recognizable \( \Rightarrow \) consistency
Efficent Instantiation (First Attempt)

Joux’s 3-party NIZK from symmetric pairing

\( (a, g^a) \)

\( (b, g^b) \)

\( (c, g^c) \)

\[
pp = (\mathbb{G}, \mathbb{G}_T, e, g)
\]

\[
k \leftarrow e(g, g)^{abc}
\]

supersingular curve ss-1536

| \( |\mathbb{G}| = 1536 \) |
| \( |\mathbb{G}_T| = 1536 \) |
| \( |\mathbb{Z}_p| = 256 \) |

symmetric pairing is too slow

Recover the only prior known scheme Boneh-Franklin escrow ElGamal PKE

- **Setup** \((1^\lambda)\): \( edk \leftarrow \mathbb{Z}_p, \ epk \leftarrow g^{edk} \).

- **KeyGen** \((pp)\): \( sk \leftarrow \mathbb{Z}_p, \ pk \leftarrow g^{sk} \).

- **Enc** \((pk, m)\): \( sk_t \leftarrow \mathbb{Z}_p, \ pk_t \leftarrow g^{sk_t}; \ k \leftarrow \text{ShareKey}(sk_t, S = \{pk_t, pk, epk\}), \ c = (pk_t, m \oplus k) \)

- **Dec** \((sk, c)\): \( k \leftarrow \text{ShareKey}(sk, S = \{pk_t, pk, epk\}), \ m \leftarrow c_2 \oplus k \).

- **Dec’** \((edk, c)\): \( k \leftarrow \text{ShareKey}(edk, S = \{pk_t, pk, epk\}), \ m \leftarrow c_2 \oplus k \).
Efficient Instantiation (Second Attempt)

The original Joux’s protocol inherently relies on symmetric pairing
Second attempt to improve efficiency: adapt Joux’s protocol with asymmetric pairing

Joux’s 3-party NIZK from asymmetric pairing

shortcomings
(i) key and ciphertext size get doubled
(ii) decryption is expensive

we need pairing to check if ciphertext is valid

\[ e(g_1^b, g_2^b) = e(g_1^b, g_2) \]
Efficient Instantiation (Final Attempt) - Global Escrow PKE Scheme 2

relaxed 3-party NIKE from asymmetric pairing

\[ \text{type-A: } (a, g_1^a, g_2^a) \]
\[ \text{type-B: } (b, g_1^b) \]
\[ \text{type-C: } (c, g_2^c) \]

keypairs could be of different types
\[ \text{type-A+type-B+type-C } \Rightarrow k \]

\[ \text{curve bls12-381} \]
\[ |G_1| = 381 \]
\[ |G_2| = 762 \]
\[ |G_T| = 1524 \]
\[ |\mathbb{Z}_p| = 256 \]

much faster and compact

New Global Escrow PKE

- **Setup(1^\lambda):** \( edk \leftarrow \mathbb{Z}_p, \; epk = (g_1^{edk}, g_2^{edk}) \) (type-A)
- **KeyGen(pp):** \( sk \leftarrow \mathbb{Z}_p, \; pk \leftarrow g_2^{sk} \) (type-B)
- **Enc(pk, m):** \( sk_t \leftarrow \mathbb{Z}_p, \; pk_t \leftarrow g_1^{sk_t} \) (type-C);
  \[ k \leftarrow \text{ShareKey}(sk_t, S = \{pk_t, pk, epk\}), \; c = (pk_t, m \oplus k) \]
- **Dec(sk, c):** \( k \leftarrow \text{ShareKey}(sk, S = \{pk_t, pk, epk\}), \; m \leftarrow c_2 \oplus k \)
- **Dec'(edk, c):** \( k \leftarrow \text{ShareKey}(edk, S = \{pk_t, pk, epk\}), \; m \leftarrow c_2 \oplus k \)
Figure: Technology roadmap of global escrow HISE. The rectangles denote our newly introduced cryptographic schemes.
Efficient Instantiations of Global Escrow HISE

Boneh-Frankin IBE

HISE scheme 1

twisted Naor-Yung transform

add global escrow

global escrow

HISE scheme 1

3-party NIKE

global escrow

PKE scheme 2

Poseidon hash + Spartan

add hierarchy

global escrow

HISE scheme 2
Applications of Global Escrow HISE

The employer can perform efficient large-scale supervision over private communications with “super” key.

The employees are assured that even a malicious boss of the “super” key cannot slander them by forging signatures for fabricated communications.
Outline

1. Background

2. Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3. Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4. Comparison and Experimentation

5. Summary
Comparison with Cartesian-Product CPK and ISE

Table: Comparison between CP-CPK, ISE, and our (global escrow) HISE

<table>
<thead>
<tr>
<th>Scheme</th>
<th>strong joint security</th>
<th>individual escrow</th>
<th>global escrow</th>
<th>key reuse</th>
<th>certificate cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP-CPK [PSST11]</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>×2</td>
</tr>
<tr>
<td>ISE [PSST11]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>×1</td>
</tr>
<tr>
<td>HISE</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>×1</td>
</tr>
<tr>
<td>global escrow HISE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×1</td>
</tr>
</tbody>
</table>

For certificate cost, ×1 (resp. ×2) means the cost associated with one (resp. two) certificate(s). As aforementioned, certificate costs include but not limit to registration, issuing, storage, transmission, verification, and building/recurring fees. Take SSL certificate as an example, one certificate is roughly 1KB, takes roughly 200∼300ms to transmit in WAN setting with 50Mbps network bandwidth and 8ms to verify. The monetary cost for an SSL certificate varies depending on features and business needs. While the cost of an SSL certificate for common usage is $10∼$2000/year, the banks and large financial institutions could spend up to $500,000/year on an SSL certificate with high-level security guarantee.
### Experimental Results

**Table:** Efficiency comparison of CPK and our proposed (global escrow) HISE schemes

<table>
<thead>
<tr>
<th>Scheme</th>
<th>efficiency (ms) [ # exp, #pairing]</th>
<th>sizes (bits) [ # G, # Zp]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KGen</td>
<td>Sign</td>
</tr>
<tr>
<td>CP-CPK</td>
<td>0.015</td>
<td>0.064</td>
</tr>
<tr>
<td></td>
<td>[2, 0]</td>
<td>[1, 0]</td>
</tr>
<tr>
<td>HISE scheme 1</td>
<td>0.057</td>
<td>0.148</td>
</tr>
<tr>
<td></td>
<td>[1, 0]</td>
<td>[1, 0]</td>
</tr>
<tr>
<td>HISE scheme 2</td>
<td>0.058</td>
<td>3.5s</td>
</tr>
<tr>
<td></td>
<td>[1, 0]</td>
<td>N/A</td>
</tr>
<tr>
<td>global escrow</td>
<td>0.057</td>
<td>0.148</td>
</tr>
<tr>
<td>HISE scheme 1</td>
<td>[1, 0]</td>
<td>[1, 0]</td>
</tr>
<tr>
<td>global escrow</td>
<td>0.057</td>
<td>3.5s</td>
</tr>
<tr>
<td>HISE scheme 2</td>
<td>[1, 0]</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Performance of Cartesian product CPK and (global escrow) HISE schemes with 128-bit security level. $(G_1, G_2, G_T)$ refers to asymmetric pairing groups. $G$ refers to ordinary elliptic group. The symbol ⊘ indicates that there is no corresponding algorithm. The symbol N/A indicates that the efficiency (or bandwidth) is hard to measure by algebra operations (or elements).
## A Byproduct: Global Escrow PKE

Table: Comparison of escrow ElGamal PKE [BF03] and our global escrow PKE

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Setup (ms)</th>
<th>KGen (# exp)</th>
<th>Enc (# pairing)</th>
<th>Dec (# pairing)</th>
<th>Dec’ (# pairing)</th>
<th>sizes (bits)</th>
<th>pp</th>
<th>edk</th>
<th>pk</th>
<th>sk</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boneh-Franklin escrow ElGamal PKE</td>
<td>2.879</td>
<td>2.014</td>
<td>8.723</td>
<td>6.654</td>
<td>6.745</td>
<td>3072</td>
<td>256</td>
<td>1536</td>
<td>256</td>
<td>3072</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2G</td>
<td>Zp</td>
<td>G</td>
<td>Zp</td>
<td>[G, G_T]</td>
<td></td>
</tr>
<tr>
<td>our proposed global escrow PKE</td>
<td>0.243</td>
<td>0.058</td>
<td>0.680</td>
<td>0.579</td>
<td>0.586</td>
<td>2286</td>
<td>256</td>
<td>381</td>
<td>256</td>
<td>2286</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2G_1, 2G_2</td>
<td>Zp</td>
<td>G_1</td>
<td>Zp</td>
<td>[G_2, G_T]</td>
<td></td>
</tr>
</tbody>
</table>

Performance of global escrow PKE schemes with 128-bit security level. (G_1, G_2, G_T) refers to asymmetric pairing groups. (G, G_T) refers to symmetric pairing groups. We report times for setup, key generation, encryption, and (escrow) decryption, as well as the sizes of public parameters pp, global escrow decryption key edk, public key pk, secret key sk, and ciphertext c.

12 ∼ 30× speed up

Our implementation is released on Github: [https://github.com/yuchen1024/HISE](https://github.com/yuchen1024/HISE)
Outline

1 Background

2 Hierarchy Integrated Signature and Encryption
   - HISE from (Constrained) IBE
   - HISE from PKE and NIZKPoK (HI conversion)

3 Global Escrow Property
   - Global Escrow PKE from PKE and NIZK (GE conversion)
   - Global Escrow PKE from 3-party NIKE

4 Comparison and Experimentation

5 Summary
Summary

HISE (formal definition + generic constructions)
- reconcile the apparent conflict between key separation and key reuse
- resolve the problem left open in Verheul [Ver01] at Eurocrypt 2001
- can be used as a drop-in replacement of PKE+SIG in scenarios that requires authenticity, confidentiality and auditibility simultaneously
- both users and authority have incentives to deploy

Global escrow PKE revisit (formal definition + generic constructions)
- indicate a new application of Naor-Yung paradigm
- establish a novel connection from 3-party NIKE
Thanks for Your Attention!

Any Questions?
Reference I


