You Can Sign but Not Decrypt: Hierarchical Integrated Encryption and Signature

SHANDONG UNIVERSITY

Min Zhang, Binbin Tu, and Yu Chen

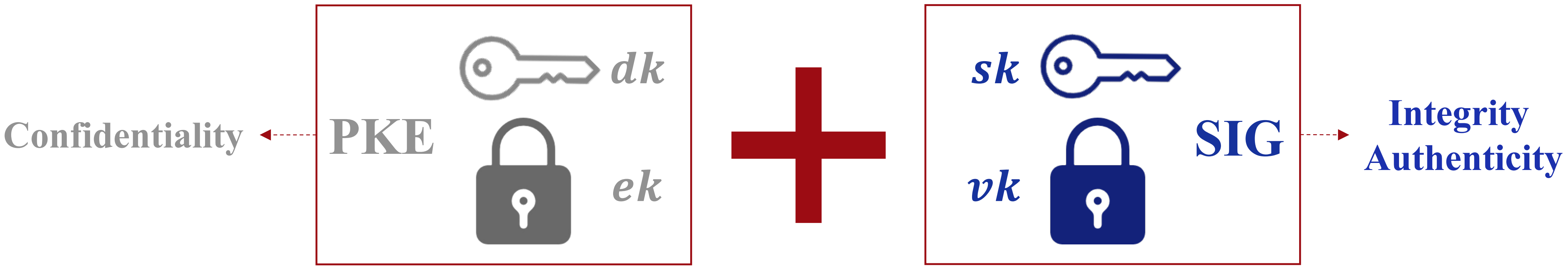
INSCRYPT 2022

11/12/2022

Outline

Outline

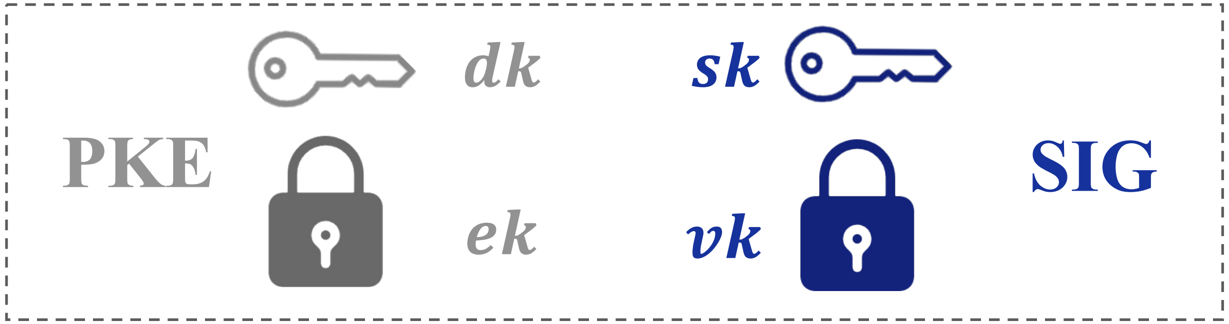
Background



- Classical examples:
  - Secure communication software: PGP, WhatsApp
  - Privacy-preserving cryptocurrency: Zether, PGC
- Security requirement (Joint security):
  - IND-CCA security for PKE: holds even in the presence of $\mathcal{O}_{\text{sign}}$
  - EUF-CMA security for SIG: holds even in the presence of $\mathcal{O}_{\text{dec}}$

Two Typical Schemes

CP-CPK
(Cartesian-Product Combined Public-Key Scheme)



- Key usage strategy: Key Separation
- Strength: rich functionalities (decryption & signature delegation[1])
- Weakness: expensive key management complexity and certificate costs[2]

ISE
(Integrated Signature and Encryption)

- Key usage strategy: Key Reuse
- Strength: low key management complexity and certificate costs
- Weakness: no rich functionalities (decryption & signature delegation is not supported)

---

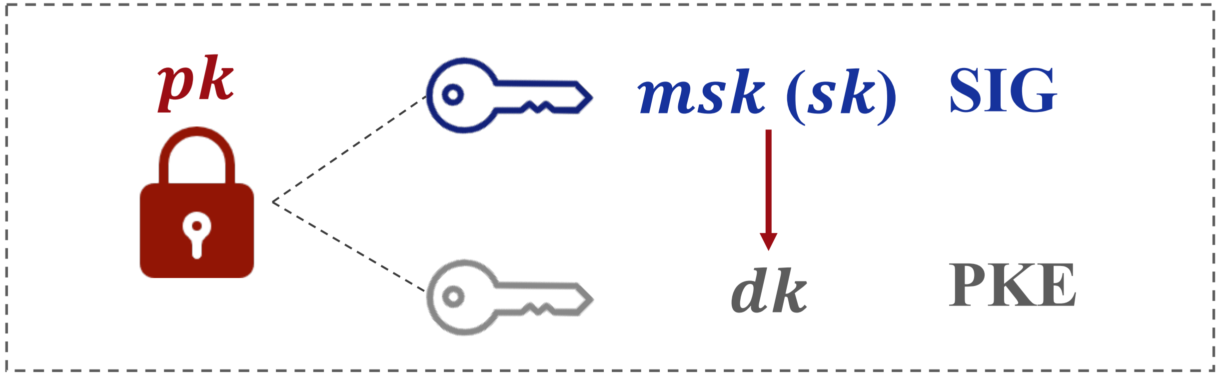
[1]The owner delegates his decryption (signing) capability to others while retaining his right of signing (decryption).

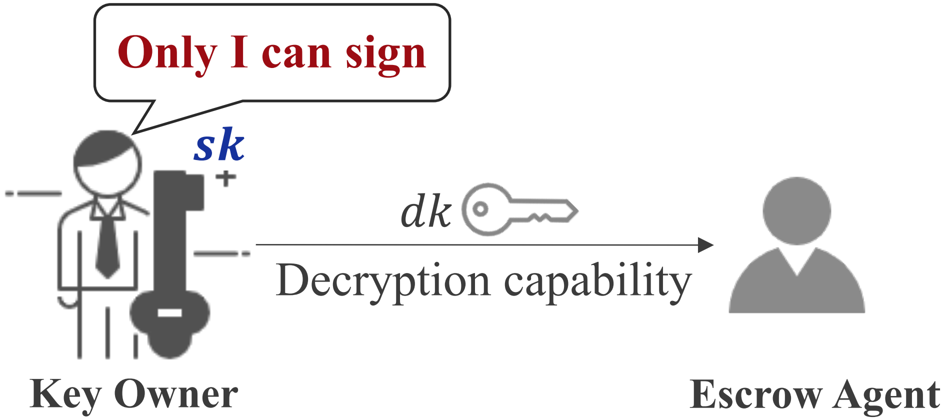
[2]The costs include but not limited to registration, issuing, storage, transmission, verification, and building/recurring fees.

Related New Scheme

HISE[CTW21]

(Hierarchical Integrated Signature and Encryption)





- Key usage strategy: a single public key, derive a decryption key from signing key
- Strength:
  - Low key management and certificate costs
  - Support to delegate decryption capability
- Weakness: Signature delegation is not supported

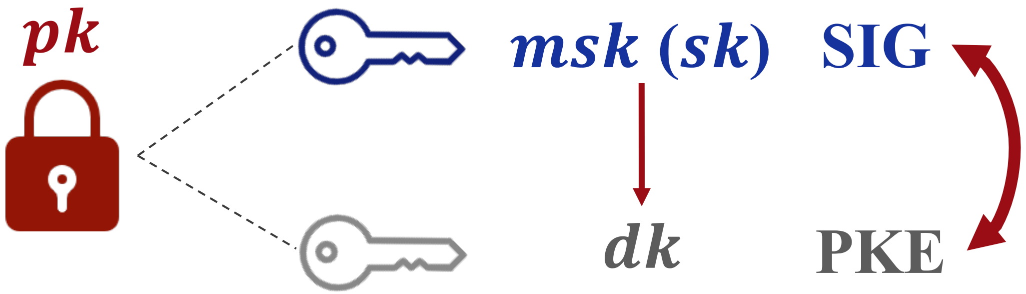
Strike a sweet balance between key separation & key reuse

Motivations

Is it possible to consider the dual notion of HISE?

—an open problem in [CTW21]

HISE[CTW21]



Reverse the hierarchy between signing key and decryption key

It is useful in scenarios where decryption capability is a first priority.

You Can Sign but Not Decrypt

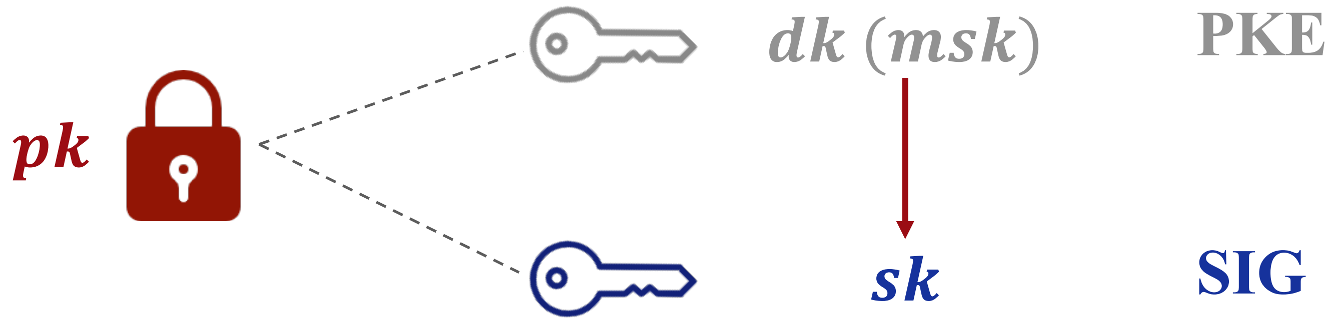
decrypt secret documents

Boss

Assistants

Outline

Hierarchical Integrated Encryption and Signature (HIES)



Key Usage Strategy

- $\text{Setup}(1^\lambda) \rightarrow pp$
- $\text{KeyGen}(pp) \rightarrow (pk, dk)$: pk serves as encryption and verification key; dk is the decryption key, serving as master secret key.
- $\text{Derive}(dk) \rightarrow sk$: sk is the signing key.

PKE

- $\text{Enc}(pk, m) \rightarrow c$
- $\text{Dec}(dk, c) \rightarrow m$

SIG

- $\text{Sign}(sk, \widetilde{m}) \rightarrow \sigma$
- $\text{Vrfy}(pk, \widetilde{m}, \sigma) \rightarrow 0/1$

Joint Security

- PKE is IND-CCA secure in the presence of a signing key

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

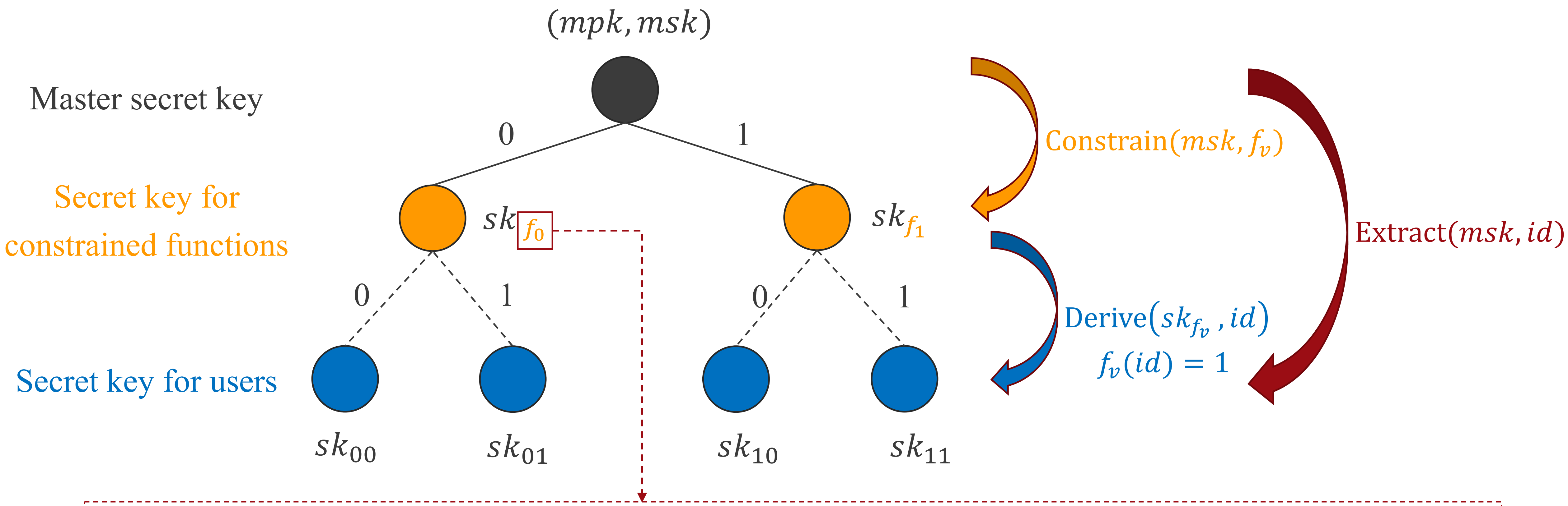
- SIG is EUF-CMA secure in the presence of a decryption oracle $\mathcal{O}_{dec}$

$$\Pr\left[\begin{array}{c} Vrfy(pk, m^*, \sigma^*) = 1 \\ \wedge\, m^* \notin Q \end{array} : \begin{array}{c} pp \leftarrow \text{Setup}(1^\lambda); \\ (pk, dk) \leftarrow \text{KeyGen}(pp); \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{dec}, \mathcal{O}_{sign}}(pp, pk); \end{array}\right] \leq \text{negl}(\lambda)$$

Outline

HIES from Constrained Identity-Based Encryption(IBE)

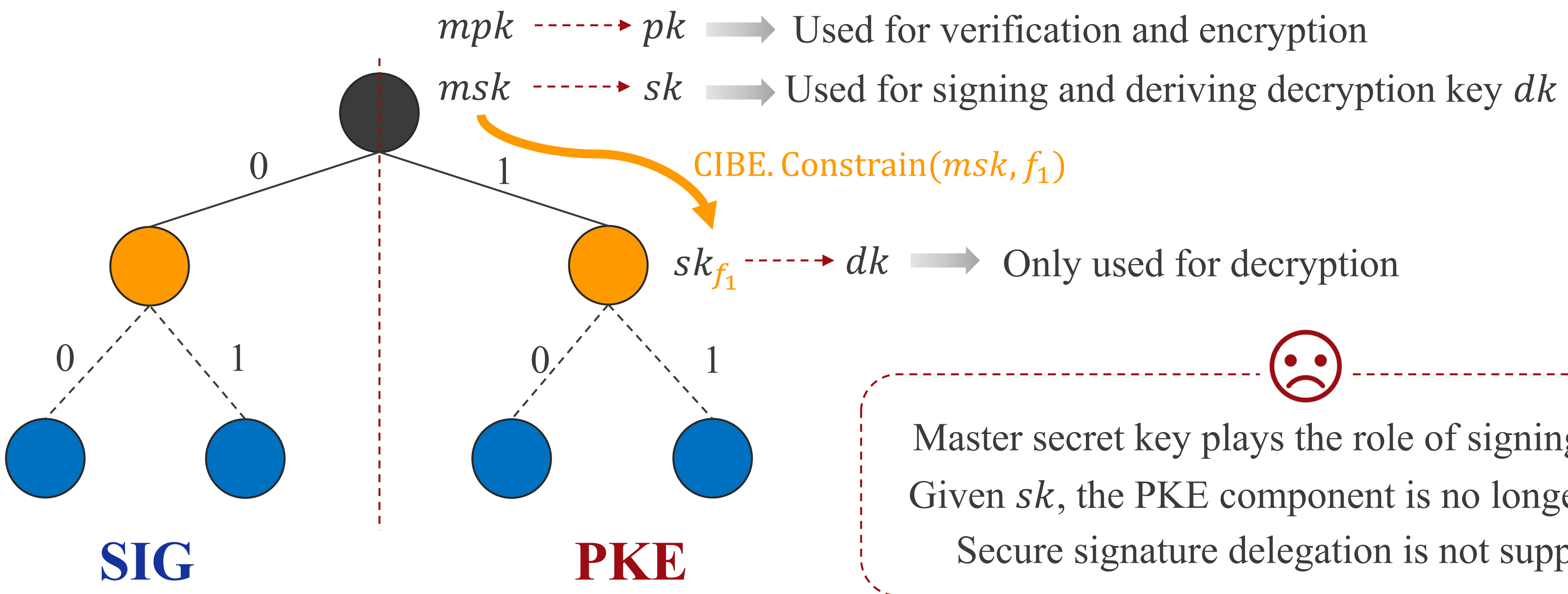
- Constrained IBE[CWT21]: an IBE in which master secret key allows efficient delegation with respect to a family of predicates over identity space.



Constrained function f_v: predicates over identity space, $f_v(id) = 1$ iff id prefixed with v.

HIES from Constrained IBE

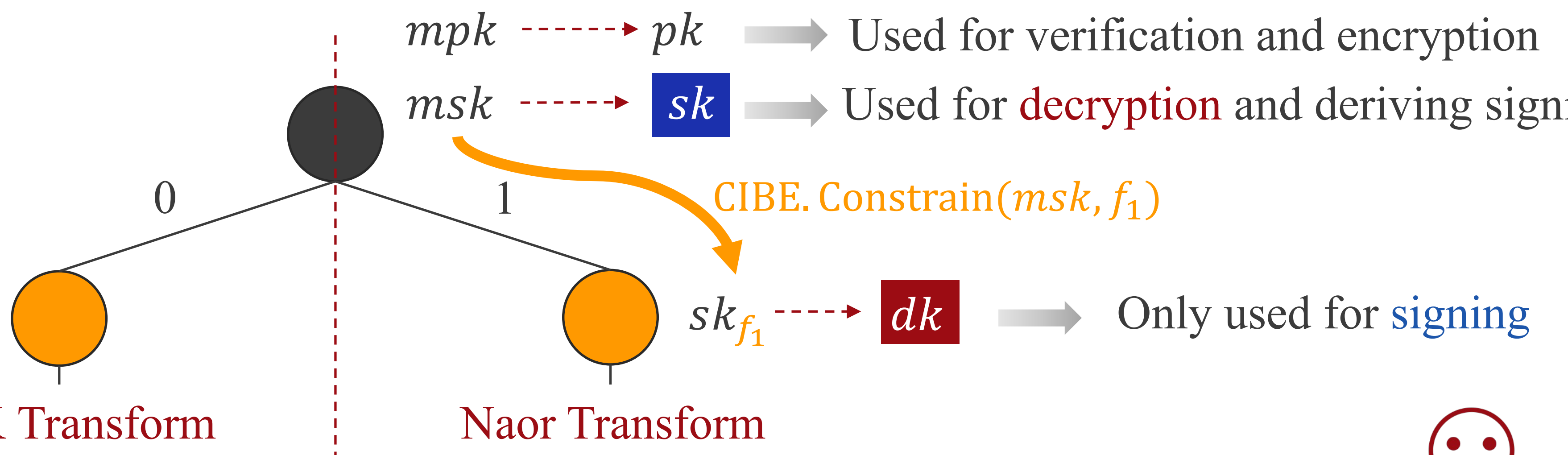
- Starting Point: Chen et al. [CTW21] give a generic construction of HISE from Constrained IBE for prefix predicates.



Master secret key plays the role of signing key sk.
Given sk, the PKE component is no longer secure!
Secure signature delegation is not supported!

HIES from Constrained IBE

- Our construction of HIES from constrained IBE: switch the roles the msk and sk_{f_1} play.



PKE $I_0 = \{id \text{ prefixed with } 0\}$

Enc: $(vk, sk) \leftarrow \mathrm{OTS.KeyGen}(1^\lambda)$
$c \leftarrow \mathrm{Enc}(mpk, 0|vk, m)$
$\sigma \leftarrow \mathrm{OTS.Sign}(sk, c)$
Dec: $1 =?\ \mathrm{OTS.Vrfy}(vk, c, \sigma)$
$m \leftarrow \mathrm{Dec}(sk_{0|vk}, c)$

$I_1 = \{id \text{ prefixed with } 1\}$ **SIG**

Sign: $\sigma \leftarrow \mathrm{Derive}(sk_1, 1|\tilde{m})$
Vrfy: $m \leftarrow_R M$,
$c \leftarrow \mathrm{Enc}(mpk, 1|\tilde{m}, m)$
$m =?\ \mathrm{Dec}(\sigma, c)$

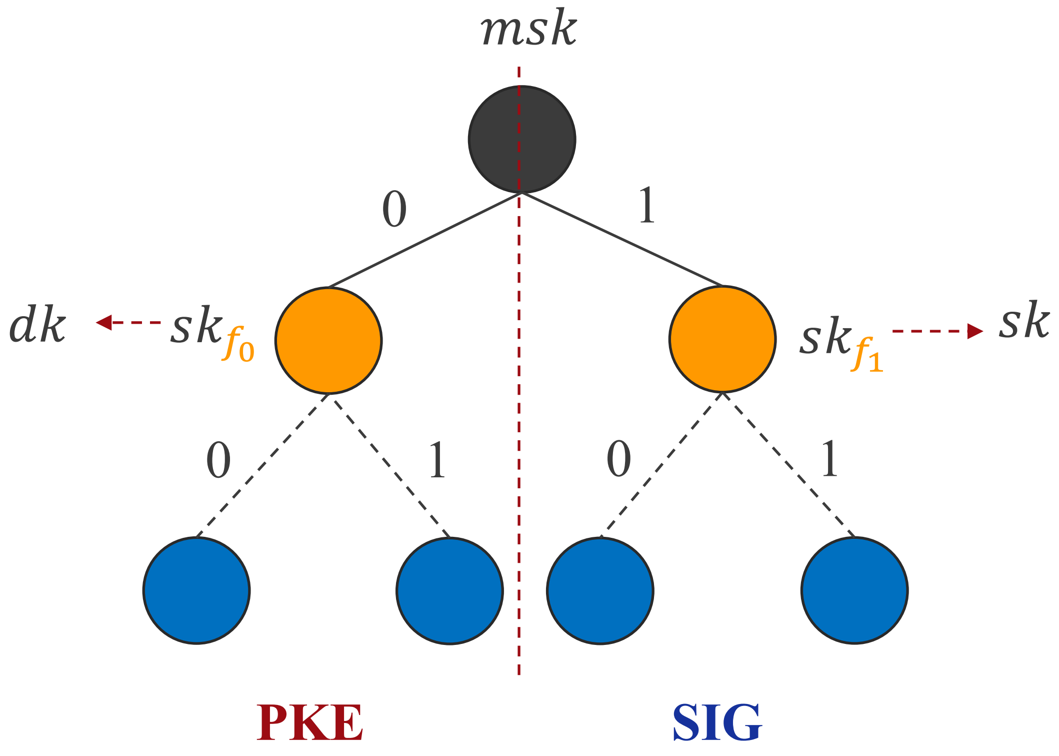
☺

Given sk,
PKE is still IND-CCA secure!
Support to delegate signing capability!

Outline

Extensions

- key observation: the prefix of an id can be assigned different and specific meanings.

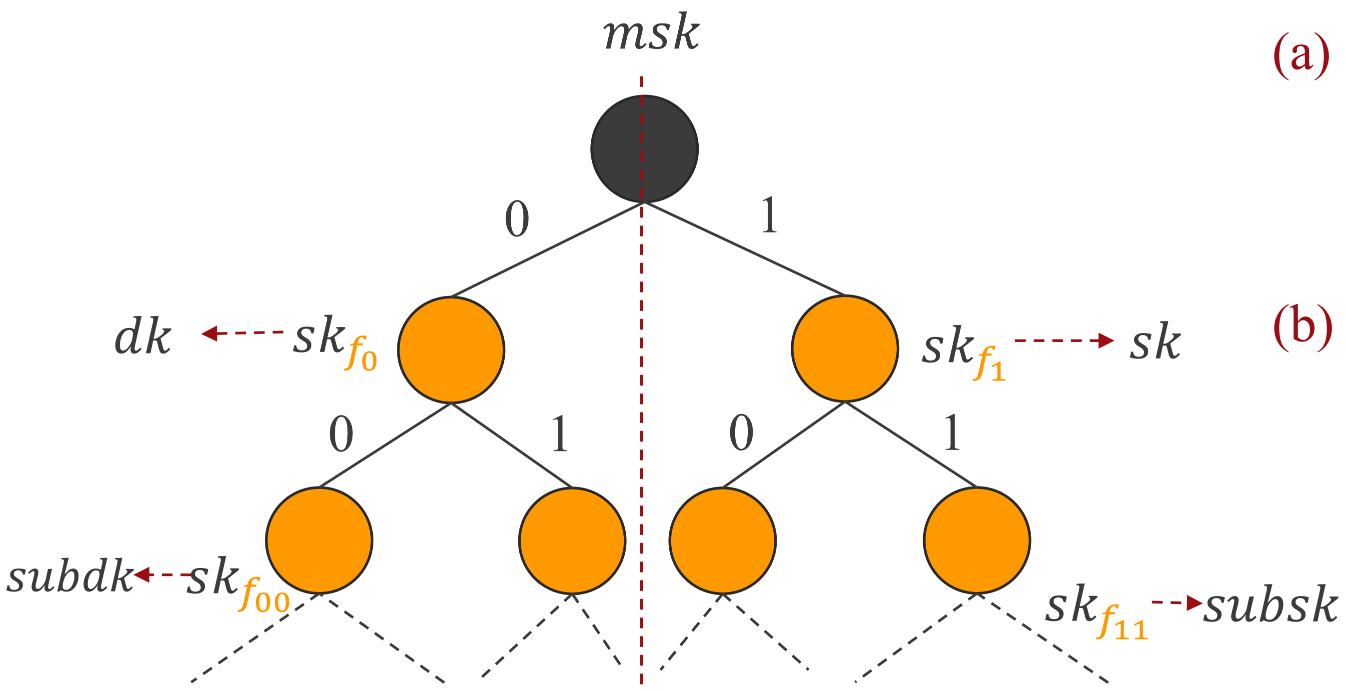


(a) Flexible Delegation

(a) Flexible Delegation: the secret key owner have the flexibility of choosing to delegate which permission (to sign or decrypt) .

Extensions

- key observation: the prefix of an id can be assigned different and specific meanings.



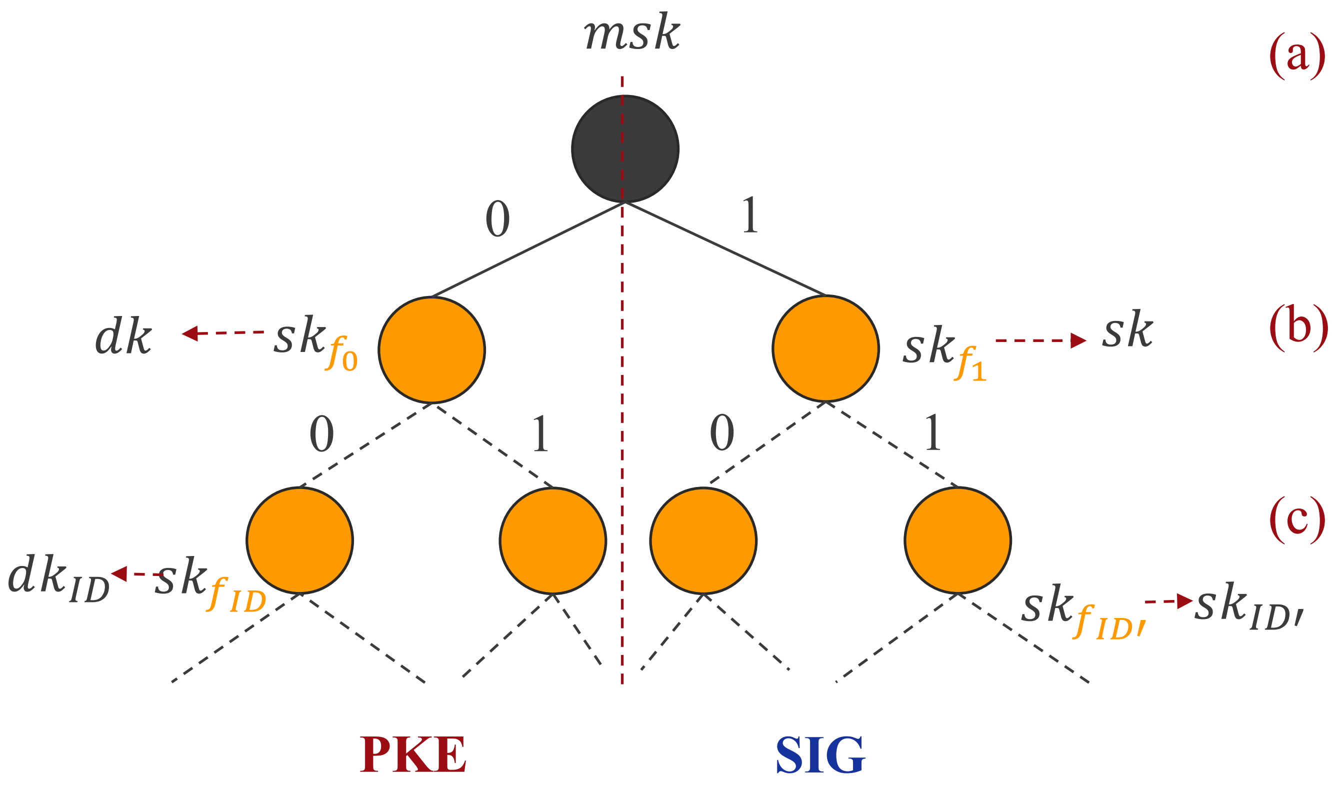
(b) Limited Delegation

(a) Flexible Delegation: the secret key owner have the flexibility of choosing to delegate which permission (to sign or decrypt) .

(b) Limited Delegation: the secret key owner can give partial signature or decryption permission to others.

Extensions

- key observation: the prefix of an id can be assigned different and specific meanings.



(c) Fine-grained Delegation

(a) Flexible Delegation: the secret key owner have the flexibility of choosing to delegate which permission (to sign or decrypt) .

(b) Limited Delegation: the secret key owner can give partial signature or decryption permission to others.

(c) Fine-grained Delegation: the secret key owner can derive delegation keys for designated persons w.r.t. their ID (identifier information such as email address) or departments w.r.t. their number.

Outline

Instantiation and Implementation Details

- Instantiation: hierarchical IBE (BB_1-IBE)
- Baseline: CP-CPK (ElGamal PKE and Schnorr signature).
- Implementation:
  - CP-CPK: **secp256k1** with 128-bit security, in which $|\mathbb{G}|$=256 bits and $|\mathbb{Z}_p|$=256 bits.
  - Our HIES scheme: **bls12-381** with 128-bit security level, in which $|\mathbb{G}_1|$=381 bits, $|\mathbb{G}_2|$=762 bits, $|\mathbb{Z}_p|$=256 bits, and $|\mathbb{G}_1|$=1524 bits.
- Open source C++ implementation: https://github.com/yuchen1024/HISE/tree/master/hies.

Comparison with CP-CPK

Functionality	strong joint security	individual escrow	key reuse	certificate costs
CP-CPK	√	√	χ	×2
HIES	√	√	√	×1

Sizes (bits)	$\lvert pk \rvert$	$\lvert sk \rvert$	$\lvert c \rvert$	$\lvert \sigma \rvert$
CP-CPK	512	512	512	512
HIES	381	762	2667	1524

Efficiency (ms)	**KeyGen**	**Derive**	**Enc**	**Dec**	**Sign**	**Vrfy**
CP-CPK	0.015	----	0.118	0.056	0.064	0.120
HIES	0.111	0.116	0.500	0.621	0.117	1.022

Though the performance of our HIES is not exciting, it has **shorter** public key size and **lower** key management complexity and key certificate costs compared to the most efficient CP-CPK.

Outline

Summary

- Formalize the dual notion of HISE[CTW21]: Hierarchical Integrated Encryption and Signature, **HIES**.
  - **Formal definition and formal joint security**
- Give a **generic construction** of HIES from constrained IBE.
- Propose three **extensions** of HIES to meet the requirements of different applications.
- Have concrete **instantiation** and open sourced **implementation**.

(a) CP-CPK (Key Seperation) **(b) ISE (Key Reuse)** **(c) HISE** **(d) HIES (this work)**

Though our construction is limited and the performance is not exciting, we emphasize the theoretical significance of HIES for solving the open problem in [CTW21] and completing the last piece of the key usage strategy puzzle.

Thank you!
Questions or comments?