Mild Asymmetric Message Franking: Illegal-Messages-Only and Retrospective Content Moderation

Zhengan Huang[1] Junzuo Lai[2] Gongxian Zeng[1]
Jian Weng[2]

Speaker: **Yijian Zhang** (University of Wollongong)

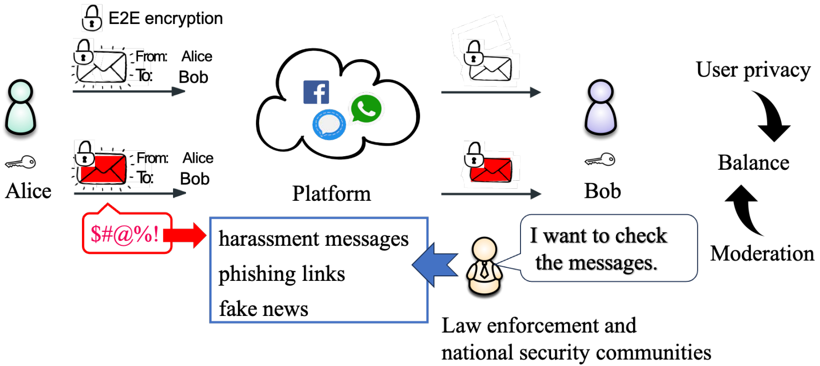
[1]Pengcheng Laboratory, Shenzhen, China

[2]College of Information Science and Technology, Jinan University, Guangzhou, China

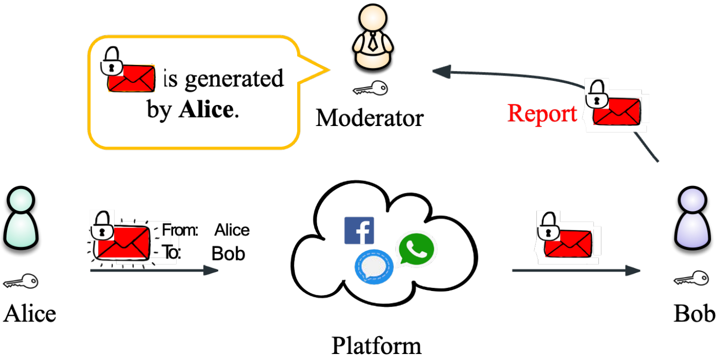
Agenda

1. Background & Motivation
2. Contributions
3. MAMF primitive
4. MAMF construction
5. References

Background: Balance between user privacy and moderation

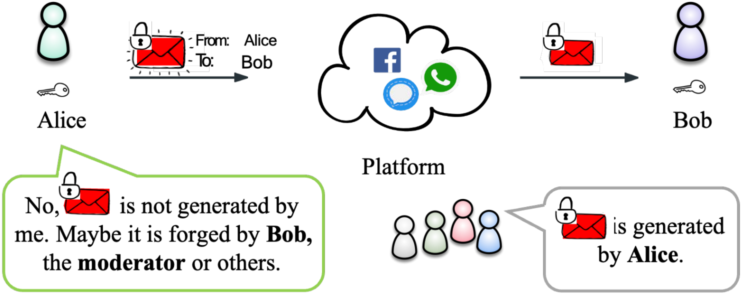


Background: Message franking (MF) [Fac16, GLR17, TGL$^+$19]



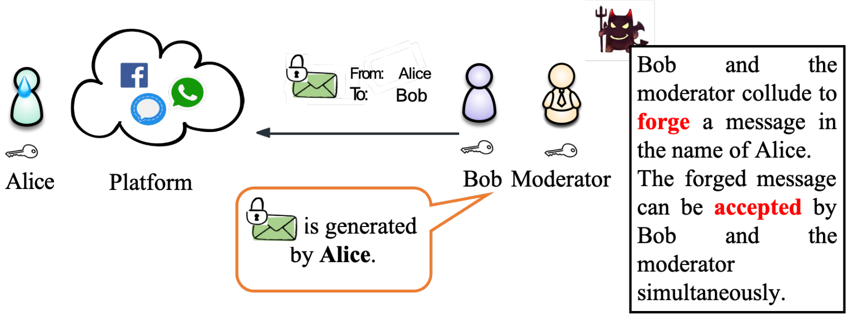
If the moderator is different to the platform, it is called asymmetric message franking (AMF) [TGL$^+$19].

Background: Deniability in AMF



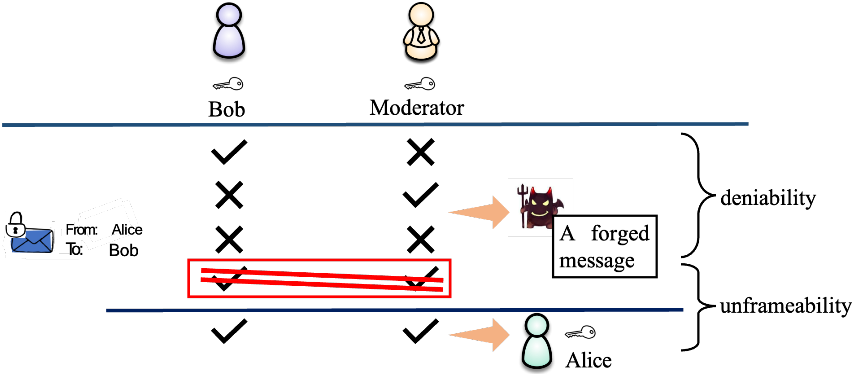
A message can be forged by the Bob, the moderator, or even the public, in the name of Alice. Usually, the public cannot distinguish between the forged messages and the normally generated messages.

Motivation: Problem I – Deniability and unframeability



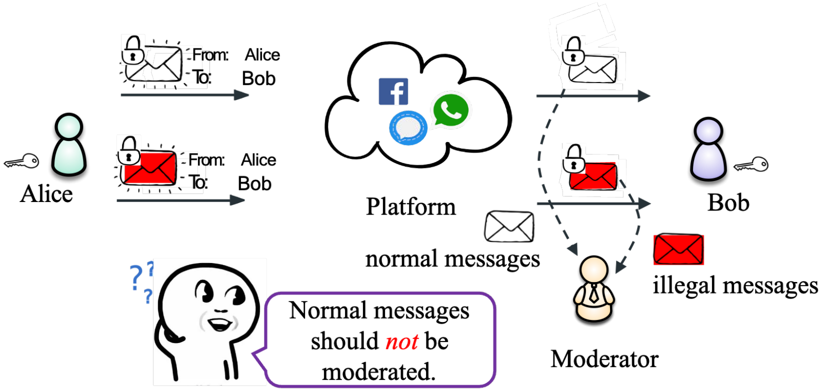
As mentioned in [BGJP23], deniability in [TGL+19] is conflict with unframeability.

Motivation: Goal I – Support deniability and unframeability simultaneously

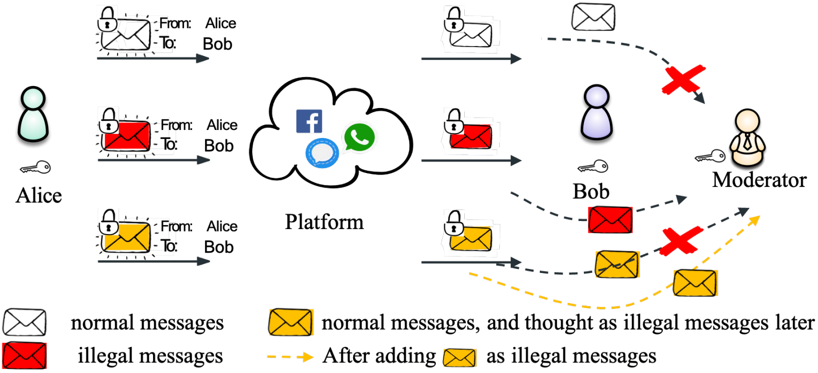


No forged message can be accepted by the Bob and the moderator simultaneously, therefore supporting unframeability.

Motivation: Problem II – A powerful moderator



Motivation: Goal II – illegal messages only and retrospective content moderation



Contributions

Our main contributions can be summarized as follows:

1. A new primitive, mild asymmetric message franking (MAMF).
2. Two new building blocks, universal set pre-constrained encryption (USPCE) and dual hash proof system-based key encapsulation mechanism supporting Sigma protocols (dual HPS-KEM$^{\Sigma}$).
3. A framework of constructing MAMF from USPCE and dual HPS-KEM$^{\Sigma}$.

Roles

$\mathsf{MAMF} = ($

$\mathsf{Setup}, \mathsf{KG}_{\mathsf{Ag}}, \mathsf{KG}_{\mathsf{J}}, \mathsf{KG}_{\mathsf{u}},$ //setup and key generation

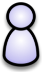
$\mathsf{Frank}, \mathsf{Verify}, \mathsf{TKGen}, \mathsf{Judge},$ //main body

$\mathsf{Forge}, \mathsf{RForge}, \mathsf{JForge}$ // for deniability

$)$

Roles:


Sender

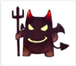

Receiver


Judge/moderator

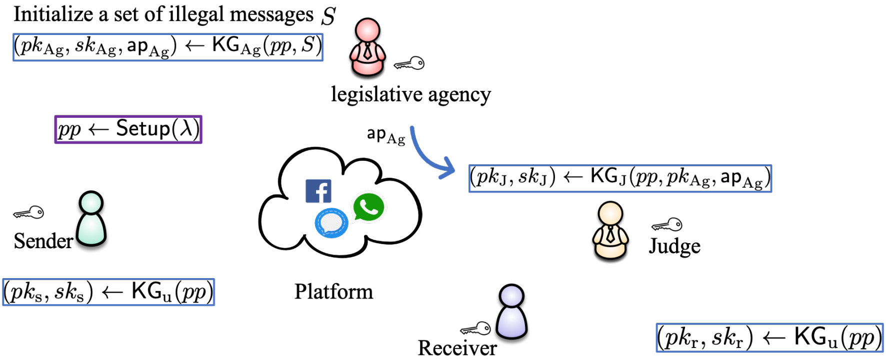

legislative agency

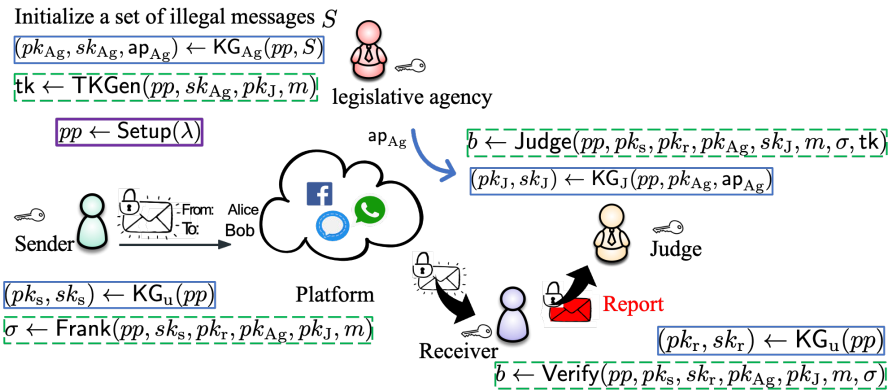
Other roles that would be used later:


The public


Compromised roles

Algorithms I




Initialize a set of illegal messages S
$(pk_{\mathrm{Ag}}, sk_{\mathrm{Ag}}, \mathsf{ap}_{\mathrm{Ag}}) \leftarrow \mathsf{KG}_{\mathrm{Ag}}(pp, S)$
$\mathsf{tk} \leftarrow \mathsf{TKGen}(pp, sk_{\mathrm{Ag}}, pk_{\mathrm{J}}, m)$
legislative agency
$pp \leftarrow \mathsf{Setup}(\lambda)$
$\mathsf{ap}_{\mathrm{Ag}}$
$b \leftarrow \mathsf{Judge}(pp, pk_{\mathrm{s}}, pk_{\mathrm{r}}, pk_{\mathrm{Ag}}, sk_{\mathrm{J}}, m, \sigma, \mathsf{tk})$
$(pk_{\mathrm{J}}, sk_{\mathrm{J}}) \leftarrow \mathsf{KG}_{\mathrm{J}}(pp, pk_{\mathrm{Ag}}, \mathsf{ap}_{\mathrm{Ag}})$
From: Alice
To: Bob
Sender
Judge
Platform
Report
$(pk_{\mathrm{s}}, sk_{\mathrm{s}}) \leftarrow \mathsf{KG}_{\mathrm{u}}(pp)$
$\sigma \leftarrow \mathsf{Frank}(pp, sk_{\mathrm{s}}, pk_{\mathrm{r}}, pk_{\mathrm{Ag}}, pk_{\mathrm{J}}, m)$
Receiver
$(pk_{\mathrm{r}}, sk_{\mathrm{r}}) \leftarrow \mathsf{KG}_{\mathrm{u}}(pp)$
$b \leftarrow \mathsf{Verify}(pp, pk_{\mathrm{s}}, sk_{\mathrm{r}}, pk_{\mathrm{Ag}}, pk_{\mathrm{J}}, m, \sigma)$

Algorithms III

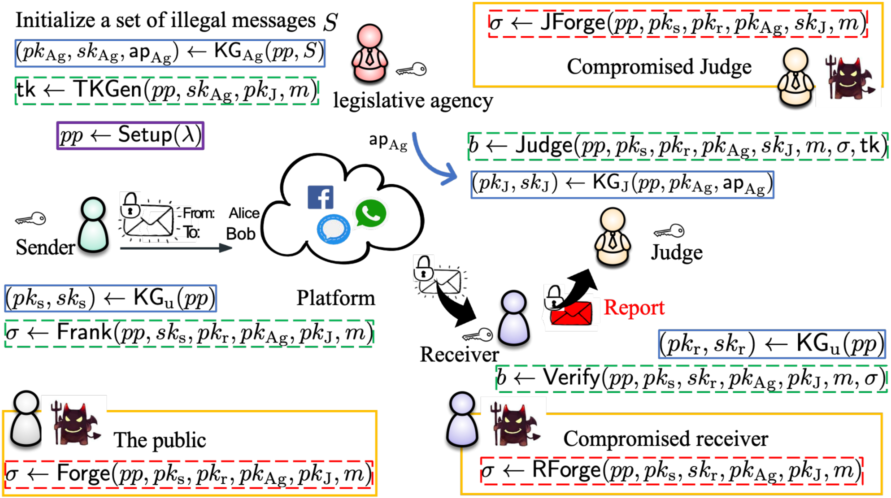


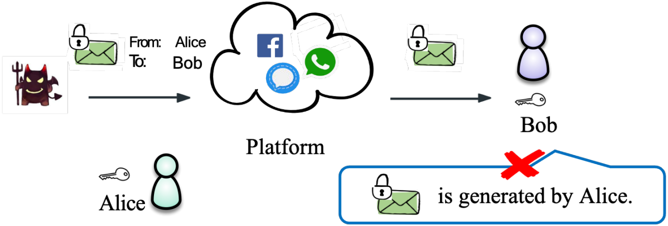
Table 1: Security properties in AMF [TGL+19] and MAMF

Security properties		AMF[TGL+19]	Our MAMF
Unforgeability		implied by s-bind and r-bind	✓
Accountability	s-bind	✓	✓
	r-bind	✓	✓
Deniability		✓	✓
Unframeability		–	✓
Untraceability		–	✓
Confidentiality of sets		–	✓

Note that unforgeability in MAMF cannot be implied by sender binding (s-bind) and receiver binding (r-bind).

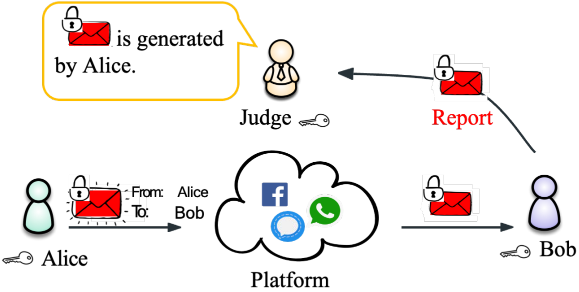
Security properties: unforgeability

Unforgeability in MAMF ensures prevention of successful impersonation, i.e., the receiver cannot be deceived into accepting a message not genuinely sent by the sender.



Security properties: accountability I

Accountability ensures that the functionality of reporting illegal messages. In line with the definition in [TGL+19, LZH+23], accountability is formalized with two special properties: sender binding and receiver binding.



Security properties: accountability II

Sender binding (s-bind) ensures that the sender cannot trick the receiver into accepting unreportable messages.
Receiver binding (r-bind) ensures that the receiver cannot deceive the judge to frame an innocent sender.

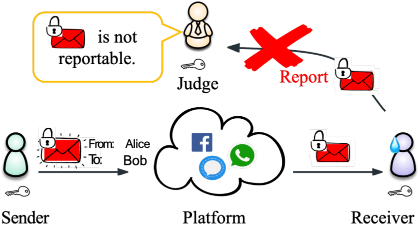


Figure 1: Attack on s-bind

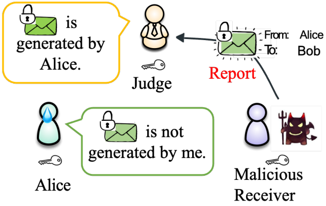
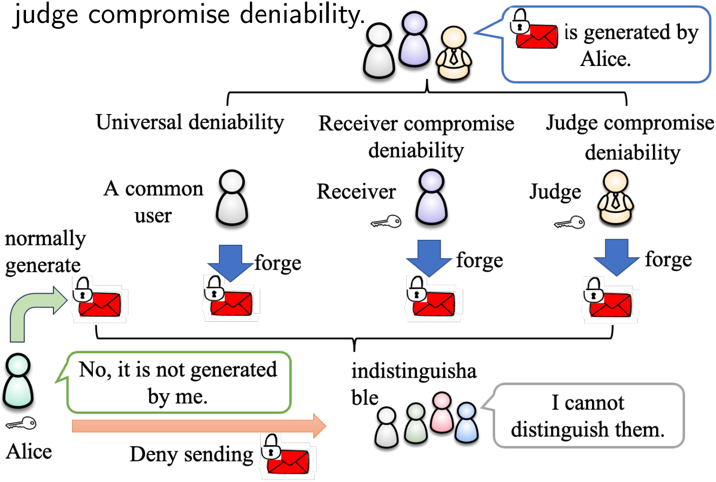


Figure 2: Attack on r-bind

Security properties: deniability

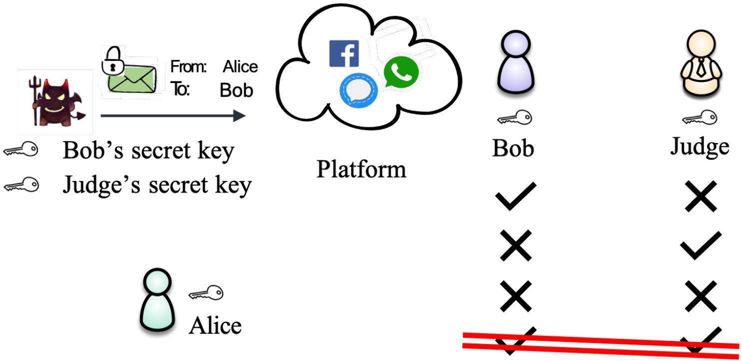
Deniability

- universal deniability;
- receiver compromise deniability;
- judge compromise deniability.



Security properties: unframeability

Unframeability of MAMF requires that no party, even given a receiver's secret key and the judge's secret key, is able to produce a signature acceptable to both the receiver and the judge.



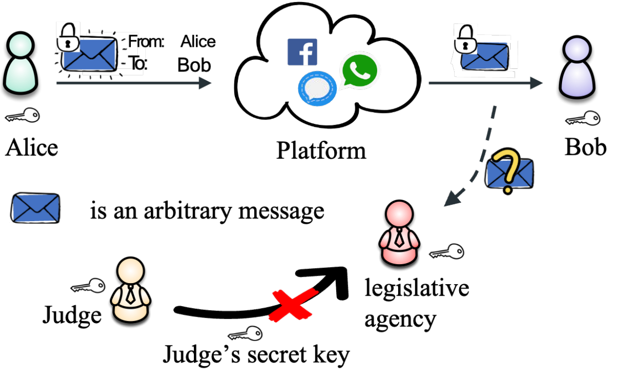
Security properties: untraceability I

Untraceability restricts the capabilities of both the legislative agency and the judge, thereby enhancing sender privacy. This concept formalizes into two distinct notions:

1. untraceability against legislative agency;
2. untraceability against judge.

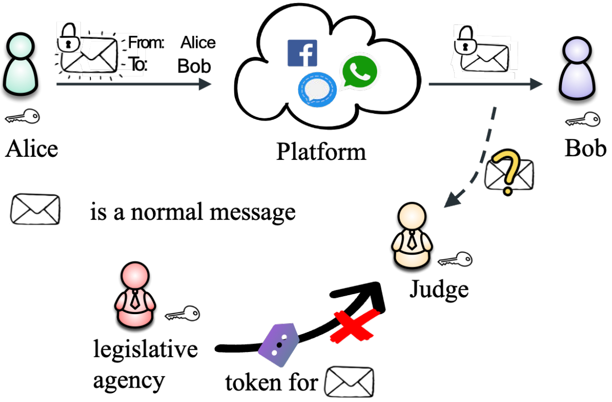
Security properties: untraceability II

Untraceability against legislative agency guarantees that the agency cannot determine if someone has actually sent a message, no matter whether it is in the set of illegal message or not.



Security properties: untraceability III

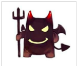
Untraceability against judge ensures that, without the assistance of the legislative agency, the moderator cannot ascertain the sender's identity when the message is not in the set of illegal messages.



Security properties: confidentiality of sets

Confidentiality of sets requires that the legislative agency's public key and the judge's public key will not disclose any information about the set of illegal messages.

$pp \leftarrow \mathsf{Setup}(\lambda) \quad b \leftarrow \{0,1\}$



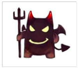
Choose two sets of illegal messages S_0 , S_1 .

legislative agency

$(pk_{\mathrm{Ag}}, sk_{\mathrm{Ag}}, \mathsf{ap}_{\mathrm{Ag}}) \leftarrow \mathsf{KG}_{\mathrm{Ag}}(pp, S_b)$

judge

$(pk_{\mathrm{J}}, sk_{\mathrm{J}}) \leftarrow \mathsf{KG}_{\mathrm{J}}(pp, pk_{\mathrm{Ag}}, \mathsf{ap}_{\mathrm{Ag}})$



Given $pk_{\mathrm{Ag}}, pk_{\mathrm{J}}$, guess b'. Wins, if $b = b'$

Two building blocks:

1. universal set pre-constrained encryption (USPCE);
2. dual hash proof system based key encapsulation mechanism supporting Sigma protocols (dual HPS-KEM$^{\Sigma}$).

Building block: USPCE

- Set pre-constrained encryption (SPCE) in [BGJP23]:
  - $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda, \mathsf{S})$, $\mathsf{S} \subseteq \mathcal{U}$;
  - $\mathsf{ct} \leftarrow \mathsf{Enc}(pk, x, m)$, $x \in \mathcal{U}$ is an item;
  - $m = \mathsf{Dec}(sk, \mathsf{ct})$ iff $x \in \mathsf{S}$.
- Insufficiency of SPCE to construct MAMF: decryption is infeasible when $x \notin \mathsf{S}$, so retrospective content moderation cannot be carried in MAMF, if adopting SPCE.

To address this challenge, we propose a new primitive, universal set pre-constrained encryption (USPCE)

Definition of USPCE I

$\mathsf{USPCE} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{TKGen}, \mathsf{Dec})$

- $(pp, \mathsf{ap}, msk) \leftarrow \mathsf{Setup}(\lambda, \mathsf{S})$: Inputting (λ, S), the setup algorithm outputs a public parameter pp, a auxiliary parameter ap and a master secret key msk.
- $(pk, sk) \leftarrow \mathsf{KG}(pp, \mathsf{ap})$: Inputting (pp, ap), the key generation algorithm run by the users, outputs (pk, sk).
- $\mathsf{ct} \leftarrow \mathsf{Enc}(pp, pk, x, m)$: Inputting (pp, pk, x, m), it outputs a ciphertext ct.
- $\mathsf{tk} \leftarrow \mathsf{TKGen}(pp, msk, x)$: Inputting (pp, msk, x), it outputs a token tk for x.
- $m/S_{\mathsf{m}} \leftarrow \mathsf{Dec}(pp, sk, \mathsf{ct}, \mathsf{tk})$: Inputting $(pp, sk, \mathsf{ct}, \mathsf{tk})$, it outputs either a message m or a polynomial-size set $S_{\mathsf{m}} \subset \mathcal{M}$.

Note that tk could be $\perp$ in Dec.

Definition of USPCE II

An USPCE scheme USPCE is *correct*, if for any $\lambda \in \mathbb{N}$, any set $\mathsf{S} \subset \mathcal{U}$, and any $m \in \mathcal{M}$, it holds that

- when $x \in \mathsf{S}$:

$$\Pr\left[\begin{array}{l} (pp, \mathsf{ap}, msk) \leftarrow \mathsf{Setup}(\lambda, \mathsf{S}) \\ (pk, sk) \leftarrow \mathsf{KG}(pp, \mathsf{ap}) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(pp, pk, x, m) \end{array} : \ m \in S_{\mathsf{m}} = \mathsf{Dec}(pp, sk, \mathsf{ct}, \perp)\right] = 1 - \mathsf{negl}(\lambda);$$

- when $x \notin \mathsf{S}$:

$$\Pr\left[\begin{array}{l} (pp, \mathsf{ap}, msk) \leftarrow \mathsf{Setup}(\lambda, \mathsf{S}) \\ (pk, sk) \leftarrow \mathsf{KG}(pp, \mathsf{ap}) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(pp, pk, x, m) \\ \mathsf{tk} \leftarrow \mathsf{TKGen}(pp, msk, x) \end{array} : \ m = \mathsf{Dec}(pp, sk, \mathsf{ct}, \mathsf{tk})\right] = 1 - \mathsf{negl}(\lambda).$$

Given a set $\mathsf{S} \subseteq \mathcal{U}$, for any pp and msk generated by $\mathsf{Setup}(\lambda, \mathsf{S})$, we define a relation as follows:

$$\mathcal{R}_{\mathsf{ct}} = \{((pk, x, \mathsf{ct}), (m, r)) : \mathsf{ct} = \mathsf{Enc}(pp, pk, x, m; r)\}.$$

Security properties of USPCE I

USPCE should satisfies:

- *Confidentiality against authority*,
- *Confidentiality against users*,
- *Confidentiality of sets*.

Definition 1 (Confidentiality against authority)

An USPCE scheme USPCE has *confidentiality against authority*, if for any set $\mathsf{S} \subseteq \mathcal{U}$ and any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathbf{Adv}^{\text{conf-au}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda) := |\Pr[\mathbf{G}^{\text{conf-au}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda) = 1] - \frac{1}{2}|$$

is negligible, where $\mathbf{G}^{\text{conf-au}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda)$ is shown in Fig. 3.

$\mathbf{G}^{\text{conf-au}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda)$:

$b \leftarrow \{0,1\}$, $(pp, \mathsf{ap}, msk) \leftarrow \mathsf{Setup}(\lambda, \mathsf{S})$, $(pk, sk) \leftarrow \mathsf{KG}(pp, \mathsf{ap})$

$(m_0, m_1, x^*, st_{\mathcal{A}}) \leftarrow \mathcal{A}_1(pp, msk, pk)$, $\mathsf{ct} \leftarrow \mathsf{Enc}(pp, pk, x^*, m_b)$, $b' \leftarrow \mathcal{A}_2(\mathsf{ct}, st_{\mathcal{A}})$

Return $(b' = b)$

Figure 3: Games $\mathbf{G}^{\text{conf-au}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda)$ for USPCE

Definition 2 (Confidentiality against users)

An USPCE scheme USPCE has *confidentiality against users*, if for any set $\mathsf{S} \subseteq \mathcal{U}$ and any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathbf{Adv}^{\text{conf-u}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda) := |\Pr[\mathbf{G}^{\text{conf-u}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda) = 1] - \frac{1}{2}|$$

is negligible, where $\mathbf{G}^{\text{conf-u}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda)$ is shown in Fig. 4.

$\mathbf{G}^{\text{conf-u}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda)$:	$\mathcal{O}^{\mathsf{TKGen}}(x')$:
$b \leftarrow \{0,1\}$, $(pp, \mathsf{ap}, msk) \leftarrow \mathsf{Setup}(\lambda, \mathsf{S})$, $Q_\mathsf{x} := \emptyset$, $U_\mathsf{x} := \emptyset$	If $x' \in U_\mathsf{x}$: Return $\perp$
$(pk, sk) \leftarrow \mathsf{KG}(pp, \mathsf{ap})$, $(m_0, m_1, x^*, st_\mathcal{A}) \leftarrow \mathcal{A}_1^{\mathcal{O}}(pp, pk, sk)$	$Q_\mathsf{x} \leftarrow Q_\mathsf{x} \cup \{x'\}$
If $(x^* \notin \mathcal{U}) \vee (x^* \in \mathsf{S}) \vee (x^* \in Q_\mathsf{x})$: Return $\perp$	Return
$U_\mathsf{x} \leftarrow U_\mathsf{x} \cup \{x^*\}$, $\mathsf{ct} \leftarrow \mathsf{Enc}(pp, pk, x^*, m_b)$, $b' \leftarrow \mathcal{A}_2^{\mathcal{O}}(\mathsf{ct}, st_\mathcal{A})$	$\mathsf{TKGen}(pp, msk, x')$
Return $(b' = b)$	

Figure 4: Games $\mathbf{G}^{\text{conf-u}}_{\text{USPCE},\mathcal{A},\mathsf{S}}(\lambda)$ for USPCE

Security properties of USPCE IV

Definition 3 (Confidentiality of sets)

A USPCE scheme USPCE supports *confidentiality of sets*, if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathbf{Adv}^{\textsf{conf-set}}_{\textsf{USPCE},\mathcal{A}}(\lambda) := |\Pr[\mathbf{G}^{\textsf{conf-set}}_{\textsf{USPCE},\mathcal{A}}(\lambda) = 1] - \frac{1}{2}|$$

is negligible, where $\mathbf{G}^{\textsf{conf-set}}_{\textsf{USPCE},\mathcal{A}}(\lambda)$ is shown in Fig. 5.

$\underline{\mathbf{G}^{\textsf{conf-set}}_{\textsf{USPCE},\mathcal{A}}(\lambda)\text{:}}$
$b \leftarrow \{0,1\}$, $(\mathsf{S}_0, \mathsf{S}_1, st_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\lambda)$ s.t $(\mathsf{S}_0 \subset \mathcal{U}) \wedge (\mathsf{S}_1 \subset \mathcal{U}) \wedge (|\mathsf{S}_0| = |\mathsf{S}_1|)$
$(pp, \mathsf{ap}, msk) \leftarrow \mathsf{Setup}(\lambda, \mathsf{S}_b)$, $(pk, sk) \leftarrow \mathsf{KG}(pp, \mathsf{ap})$, $b' \leftarrow \mathcal{A}_2(pp, pk, st_{\mathcal{A}})$
Return $(b' = b)$

Figure 5: Games $\mathbf{G}^{\textsf{conf-set}}_{\textsf{USPCE},\mathcal{A}}(\lambda)$ for USPCE

Building block: Dual HPS-KEM$^{\Sigma}$

HPS-KEM$^{\Sigma}$ was proposed in [LZH^{+}23] to construct asymmetric group message franking (AGMF), which is a variant of key encapsulation mechanism (KEM) satisfying that

(i) it can be interpreted from the perspective of hash proof system (HPS) [CS02],

(ii) for some special relations (about the public/secret keys, the encapsulated keys and ciphertexts), there exist corresponding Sigma protocols.

In this work, we consider its dual version. Thus, we firstly recall HPS-KEM$^{\Sigma}$, then show the dual version.

Definition of HPS-KEM$^{\Sigma}$ I

HPS-KEM$^{\Sigma}$ = (KEMSetup, KG, Encap_c, Encap_c^*, Encap_k, Decap, CheckKey, CheckCwel)

- $pp \leftarrow \text{KEMSetup}(1^\lambda)$: it outputs a public parameter pp.
- $(pk, sk) \leftarrow \text{KG}(pp)$: it outputs a key pairs (pk, sk).
- $c \leftarrow \text{Encap}_c(pp; r)$: it outputs a well-formed ciphertext c.
- $c \leftarrow \text{Encap}_c^*(pp; r^*)$: it outputs a ciphertext c.
- $k \leftarrow \text{Encap}_k(pp, pk; r)$: it outputs an encapsulated key $k \in \mathcal{K}$. it outputs a ciphertext c.
- $k' \leftarrow \text{Decap}(pp, sk, c)$: it outputs an encapsulated key $k' \in \mathcal{K}$.
- $b \leftarrow \text{CheckKey}(pp, sk, pk)$: it checks whether the keys are well-formed.
- $b \leftarrow \text{CheckCwel}(pp, c, r^*)$: it checks whether the ciphertext is well-formed.

Definition of HPS-KEM$^\Sigma$ II

Correctness:

(1) For any pp generated by KEMSetup(1^λ), and any (pk, sk) output by KG(pp), CheckKey$(pp, sk, pk) = 1$.

(2) For any pp generated by KEMSetup(1^λ), any (pk, sk) satisfying CheckKey$(pp,\ sk, pk) = 1$, and any $c \leftarrow \mathsf{Encap_c}(pp; r)$, $k \leftarrow \mathsf{Encap_k}(pp, pk; r)$, it holds that Decap$(pp, sk, c) = k$.

(3) For any pp generated by KEMSetup(1^λ), and any c generated with $\mathsf{Encap}^*_\mathsf{c}(pp; r^*)$, CheckCwel$(pp, c, r^*) = 1$ if and only if c is well-formed.

Definition of HPS-KEM$^{\Sigma}$ III

For any pp generated by $\mathsf{KEMSetup}(1^\lambda)$, we define some relations as follows and we require there exists a Sigma protocol for each relation:

$$\mathcal{R}_{\mathsf{s}} = \{(sk, pk) : \mathsf{CheckKey}(pp, sk, pk) = 1\}$$

$$\mathcal{R}_{\mathsf{c,k}} = \{(r, (c, k, pk)) : c = \mathsf{Encap}_{\mathsf{c}}(pp; r) \wedge k = \mathsf{Encap}_{\mathsf{k}}(pp, pk; r)\}$$

$$\mathcal{R}^*_{\mathsf{c}} = \{(r^*, c) : c = \mathsf{Encap}^*_{\mathsf{c}}(pp; r^*)\}$$

where

- $\mathcal{R}_{\mathsf{s}}$ is a relation proving that the keys are valid,
- $\mathcal{R}_{\mathsf{c,k}}$ is a relation proving that (c, k) are generated via $\mathsf{Encap}_{\mathsf{c}}$ and $\mathsf{Encap}_{\mathsf{k}}$,
- $\mathcal{R}^*_{\mathsf{c}}$ is a relation proving that c is a ciphertext output by $\mathsf{Encap}^*_{\mathsf{c}}$.

Dual version I

Compared with HPS-KEM$^{\Sigma}$, dHPS-KEM$^{\Sigma}$ has extra four algorithms:

- $k_{\mathsf{d}} \leftarrow \mathsf{dEncap}_{\mathsf{k}}(pp, pk, t; r)$: On input the public parameter pp, a public key pk, and a tag $t \in \mathcal{T}$ with inner randomness $r \in \mathcal{RS}$, it outputs an encapsulated key $k \in \mathcal{K}$.
- $k'_{\mathsf{d}} \leftarrow \mathsf{dDecap}(pp, sk, t, c)$: On input the public parameter pp, a secret key sk, a tag $t \in \mathcal{T}$ and a ciphertext c, it outputs an encapsulated key $k'_{\mathsf{d}} \in \mathcal{K}$.
- $k \leftarrow \mathsf{SamEncK}(pp; r^*_{\mathsf{k}})$: On input the public parameter pp with inner randomness $r^*_{\mathsf{k}} \in \mathcal{RS}^*$, it outputs an encapsulated key $k \in \mathcal{K}$.
- $k_{\mathsf{d}} \leftarrow \mathsf{dSamEncK}(pp, t; r^*_{\mathsf{k}})$: On input the public parameter pp and a tag $t \in \mathcal{T}$ with inner randomness $r^*_{\mathsf{k}} \in \mathcal{RS}^*$, it outputs an encapsulated key $k_{\mathsf{d}} \in \mathcal{K}$.

Dual version II

Correctness:

(1) For any pp generated by $\mathsf{KEMSetup}(1^\lambda)$, any (pk, sk) satisfying $\mathsf{CheckKey}(pp,\ sk, pk) = 1$, and any $c \leftarrow \mathsf{Encap_c}(pp; r)$, $k_\mathsf{d} \leftarrow \mathsf{dEncap_k}(pp, pk, t; r)$, it holds that $\mathsf{dDecap}(pp, sk, t, c) = k_\mathsf{d}$.

Dual version III

For any pp generated by $\mathsf{KEMSetup}(1^\lambda)$, we define some relations as follows and we require there exists a Sigma protocol for each relation:

$$\mathcal{R}^{\mathsf{d}}_{\mathsf{c,k}} = \{((c, k_{\mathsf{d}}, pk), (t, r)) : (c = \mathsf{Encap_c}(pp; r)) \wedge (k_{\mathsf{d}} = \mathsf{dEncap_k}(pp, pk, t; r))\}$$

$$\mathcal{R}^{*}_{\mathsf{k}} = \{(k, r^*_{\mathsf{k}}) : k = \mathsf{SamEncK}(pp; r^*_{\mathsf{k}})\},$$

$$\mathcal{R}^{\mathsf{d}*}_{\mathsf{k}} = \{(k_{\mathsf{d}}, (t, r^*_{\mathsf{k}})) : k_{\mathsf{d}} = \mathsf{dSamEncK}(pp, t; r^*_{\mathsf{k}})\}$$

where

- $\mathcal{R}^{\mathsf{d}}_{\mathsf{c,k}}$ is a relation proving that (c, k) are generated via $\mathsf{Encap_c}$ and $\mathsf{dEncap_k}$,
- $\mathcal{R}^{*}_{\mathsf{k}}$ is a relation proving that k is sampled from $\mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{K}$ (using the randomness r^*_{k}),
- $\mathcal{R}^{\mathsf{d}*}_{\mathsf{k}}$ is a relation proving that k_{d} is sampled from $\mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{K}$ with a tag $t \in \mathcal{T}$ (using the randomness r^*_{k}).

Security properties of dual HPS-KEM$^{\Sigma}$ I

The security properties of dual HPS-KEM$^{\Sigma}$ includes:

- the properties in HPS-KEM$^{\Sigma}$:
  - *universality*,
  - *ciphertext unexplainability*,
  - *indistinguishability*,
  - *SK-second-preimage resistance(SK-2PR)*,
  - *smoothness*.
- and some new properties:
  - *extended universality*,
  - *key unexplainability*,
  - *extended key unexplainability*,
  - *extended smoothness*,
  - *special extended smoothness*,
  - *Uniformity of sampled keys*.

Security properties of dual HPS-KEM$^{\Sigma}$ II

Definition 4 (Universality)

dHPS-KEM$^{\Sigma}$ is *universal*, if for any computationally unbounded adversary $\mathcal{A}$,
$\mathbf{Adv}^{\mathsf{univ}}_{\mathsf{dHPS\text{-}KEM}^{\Sigma},\mathcal{A}}(\lambda) := \Pr[\mathbf{G}^{\mathsf{univ}}_{\mathsf{dHPS\text{-}KEM}^{\Sigma},\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$,
where $\mathbf{G}^{\mathsf{univ}}_{\mathsf{dHPS\text{-}KEM}^{\Sigma},\mathcal{A}}(\lambda)$ is defined in Fig. 6.

$\underline{\mathbf{G}^{\mathsf{univ}}_{\mathsf{dHPS\text{-}KEM}^{\Sigma},\mathcal{A}}(\lambda)}$:
$pp \leftarrow \mathsf{KEMSetup}(\lambda)$, $(pk, sk) \leftarrow \mathsf{KG}(pp)$
$(c, k, r_{\mathsf{c}}^*) \leftarrow \mathcal{A}(pp, pk)$
s.t. $((c, r_{\mathsf{c}}^*) \in \mathcal{R}_{\mathsf{c}}^*) \wedge (\mathsf{CheckCwel}(pp, c, r_{\mathsf{c}}^*) = 0)$
If $k = \mathsf{Decap}(pp, sk, c)$: Return 1
Else Return 0

Figure 6: Game for defining universality of dHPS-KEM$^{\Sigma}$

Definition 5 (Extended universality)

dHPS-KEM$^{\Sigma}$ is *extended universal*, if for any computationally unbounded adversary $\mathcal{A}$,
$\textbf{Adv}^{\text{ex-univ}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) := \Pr[\textbf{G}^{\text{ex-univ}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) = 1] \leq \text{negl}(\lambda)$,
where $\textbf{G}^{\text{ex-univ}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$ is defined in Fig. 7.

$\underline{\textbf{G}^{\text{ex-univ}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)}$:
$pp \leftarrow \text{KEMSetup}(\lambda)$, $(pk, sk) \leftarrow \text{KG}(pp)$
$(c, k, r^*_{\mathsf{c}}, t) \leftarrow \mathcal{A}(pp, pk)$
s.t. $((c, r^*_{\mathsf{c}}) \in \mathcal{R}^*_{\mathsf{c}}) \wedge (\text{CheckCwel}(pp, c, r^*_{\mathsf{c}}) = 0)$
If $k = \text{dDecap}(pp, sk, t, c)$: Return 1
Else Return 0

Figure 7: Game for defining extended universality of dHPS-KEM$^{\Sigma}$

Security properties of dual HPS-KEM$^{\Sigma}$ IV

Definition 6 (Ciphertext unexplainability)

dHPS-KEM$^{\Sigma}$ is *ciphertext-unexplainable*, if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\text{C-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) := \Pr[\mathbf{G}^{\text{C-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) = 1] \leq \text{negl}(\lambda)$, where $\mathbf{G}^{\text{C-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$ is defined in Fig. 8.

$\mathbf{G}^{\text{C-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$:

$pp \leftarrow \text{KEMSetup}(\lambda)$,
$(c, r_{\mathsf{c}}^*) \leftarrow \mathcal{A}(pp)$ s.t. $(c, r_{\mathsf{c}}^*) \in \mathcal{R}_{\mathsf{c}}^*$
If $\text{CheckCwel}(pp, c, r_{\mathsf{c}}^*) = 1$: Return 1
Else Return 0

Figure 8: Game for defining ciphertext unexplainability of dHPS-KEM$^{\Sigma}$

Security properties of dual HPS-KEM$^{\Sigma}$ V

Definition 7 (Key unexplainability)

dHPS-KEM$^{\Sigma}$ is *key-unexplainable*, if for any PPT adversary $\mathcal{A}$,
$\mathbf{Adv}^{\text{K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) := \Pr[\mathbf{G}^{\text{K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) = 1] \leq \text{negl}(\lambda)$,
where $\mathbf{G}^{\text{K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$ is defined in Fig. 9.

$\mathbf{G}^{\text{K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$:

$pp \leftarrow \text{KEMSetup}(\lambda)$, $(pk, sk) \leftarrow \text{KG}(pp)$
$(c, r^*_{\text{c}}, k, r^*_{\text{k}}) \leftarrow \mathcal{A}(pp, pk, sk)$
s.t. $((c, r^*_{\text{c}}) \in \mathcal{R}^*_{\text{c}}) \wedge ((k, r^*_{\text{k}}) \in \mathcal{R}^*_{\text{k}})$
If $\text{Decap}(pp, sk, c) = k$: Return 1
Else Return 0

Figure 9: Game for defining key unexplainability of dHPS-KEM$^{\Sigma}$

Definition 8 (Extended key unexplainability)

dHPS-KEM$^{\Sigma}$ is *extended key-unexplainable*, if for any PPT adversary $\mathcal{A}$,
$\mathbf{Adv}^{\text{ex-K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) := \Pr[\mathbf{G}^{\text{ex-K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) = 1] \leq \text{negl}(\lambda)$,
where $\mathbf{G}^{\text{ex-K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$ is defined in Fig. 10.

$\mathbf{G}^{\text{ex-K-unexpl}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$:
$pp \leftarrow \text{KEMSetup}(\lambda)$, $(pk, sk) \leftarrow \text{KG}(pp)$
$(c, r_{\text{c}}^*, k_{\text{d}}, t, r_{\text{k}}^*) \leftarrow \mathcal{A}(pp, pk, sk)$
s.t. $((c, r_{\text{c}}^*) \in \mathcal{R}_{\text{c}}^*) \wedge ((k_{\text{d}}, (t, r_{\text{k}}^*)) \in \mathcal{R}_{\text{k}}^{\text{d}*})$
If $\text{dDecap}(pp, sk, t, c) = k_{\text{d}}$: Return 1
Else Return 0

Figure 10: Game for defining extended key unexplainability of dHPS-KEM$^{\Sigma}$

Definition 9 (Indistinguishability)

dHPS-KEM$^\Sigma$ is *indistinguishable*, if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{ind}}_{\mathsf{dHPS\text{-}KEM}^\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathbf{G}^{\mathsf{ind}}_{\mathsf{dHPS\text{-}KEM}^\Sigma,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}| \leq \mathsf{negl}(\lambda)$, where $\mathbf{G}^{\mathsf{ind}}_{\mathsf{dHPS\text{-}KEM}^\Sigma,\mathcal{A}}(\lambda)$ is defined in Fig. 11.

$\mathbf{G}^{\mathsf{ind}}_{\mathsf{dHPS\text{-}KEM}^\Sigma,\mathcal{A}}(\lambda)$:

$b \leftarrow \{0,1\}$, $pp \leftarrow \mathsf{KEMSetup}(1^\lambda)$

$c_0 \leftarrow \mathsf{encap_c}(pp)$, $c_1 \leftarrow \mathsf{encap^*_c}(pp)$

$b' \leftarrow \mathcal{A}(pp, c_b)$

Return $(b' \stackrel{?}{=} b)$

Figure 11: Games for defining indistinguishability of dHPS-KEM$^\Sigma$

Definition 10 (SK-2PR)

dHPS-KEM$^{\Sigma}$ is *SK-second-preimage resistant*, if for any PPT adversary $\mathcal{A}$,
$\mathbf{Adv}^{\text{sk-2pr}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) := \Pr[\mathbf{G}^{\text{sk-2pr}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda) = 1] \leq \text{negl}(\lambda)$,
where $\mathbf{G}^{\text{sk-2pr}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)$ is defined in Fig. 11.

$\underline{\mathbf{G}^{\text{sk-2pr}}_{\text{dHPS-KEM}^{\Sigma},\mathcal{A}}(\lambda)}$:
$pp \leftarrow \text{KEMSetup}(1^{\lambda})$, $(pk, sk) \leftarrow \text{KG}(pp)$
$sk' \leftarrow \mathcal{A}(pp, pk, sk)$
If $(sk' \neq sk) \wedge (\text{CheckKey}(pp, sk', pk) = 1)$: Return 1
Return 0

Figure 12: Games for defining SK-second-preimage resistance of dHPS-KEM$^{\Sigma}$

Security properties of dual HPS-KEM$^{\Sigma}$ IX

Definition 11 (Smoothness)

dHPS-KEM$^{\Sigma}$ is *smooth*, if for any fixed pp generated by KEMSetup and any fixed pk generated by KG,

$$\Delta((c,k),(c,k')) \leq \mathsf{negl}(\lambda),$$

where $c \leftarrow \mathsf{Encap}^*_\mathsf{c}(pp)$, $k \leftarrow \mathcal{K}$, $sk \leftarrow \mathcal{SK}_{pp,pk}$ and $k' = \mathsf{Decap}(pp, sk, c)$.

Definition 12 (Extended smoothness)

dHPS-KEM$^{\Sigma}$ is *extended smooth*, if for any fixed pp generated by KEMSetup and any fixed pk generated by KG,

$$\Delta((c,k,t),(c,k',t)) \leq \mathsf{negl}(\lambda),$$

where $c \leftarrow \mathsf{Encap}^*_\mathsf{c}(pp)$, $k \leftarrow \mathcal{K}$, $t \leftarrow \mathcal{T}$, $sk \leftarrow \mathcal{SK}_{pp,pk}$ and $k' = \mathsf{dDecap}(pp, sk, t, c)$.

Security properties of dual HPS-KEM$^{\Sigma}$ X

Definition 13 (Special extended smoothness)

dHPS-KEM$^{\Sigma}$ is *special extended smooth*, if for any fixed pp generated by KEMSetup and any fixed (pk, sk) generated by KG,

$$\Delta((c, k), (c, k')) \leq \mathsf{negl}(\lambda),$$

where $c \leftarrow \mathsf{Encap}^*_{\mathsf{c}}(pp)$, $k \leftarrow \mathcal{K}$, $t \leftarrow \mathcal{T}$ and $k' = \mathsf{dDecap}(pp, sk, t, c)$.

Definition 14 (Uniformity of sampled keys)

dHPS-KEM$^{\Sigma}$ has *uniformity of sampled keys*, if for any pp generated by KEMSetup and any $t \in \mathcal{T}$, it holds that

$$\Delta(k, k') = 0 \quad \text{and} \quad \Delta(k, k'') = 0$$

where $k \leftarrow \mathcal{K}$, $k' \leftarrow \mathsf{SamEncK}(pp)$ and $k'' \leftarrow \mathsf{dSamEncK}(pp, t)$.

Construction of MAMF I

$\underline{\mathsf{Setup}(\lambda):}$

Return $pp := pp_{\mathsf{KEM}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{KEMSetup}(\lambda)$

$\underline{\mathsf{KG}_{\mathsf{Ag}}(pp, \mathsf{S}):}$

Return $(pk_{\mathsf{Ag}}, sk_{\mathsf{Ag}}) := (pp_{\mathsf{USPCE}}, msk_{\mathsf{USPCE}}) \leftarrow \mathsf{USPCE.Setup}(\lambda, \mathsf{S})$

$\underline{\mathsf{KG}_{\mathsf{J}}(pp, pk_{\mathsf{Ag}}):}$

$(pk'_{\mathsf{J}}, sk'_{\mathsf{J}}) \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{KG}(pp_{\mathsf{KEM}})$
$(pk_{\mathsf{USPCE}}, sk_{\mathsf{USPCE}}) \leftarrow \mathsf{USPCE.KG}(pp_{\mathsf{USPCE}})$
Return $(pk_{\mathsf{J}} = (pk_{\mathsf{USPCE}}, pk'_{\mathsf{J}}), sk_{\mathsf{J}} = (sk_{\mathsf{USPCE}}, sk'_{\mathsf{J}}))$

$\underline{\mathsf{KG}_{\mathsf{u}}(pp):}$

Return $(pk, sk) \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{KG}(pp_{\mathsf{KEM}})$

Figure 13: Algorithm descriptions of Setup, $\mathsf{KG}_{\mathsf{Ag}}$, KG_{J} and KG_{u}

Construction of MAMF II

$\underline{\mathsf{Frank}(pp, sk_{\mathsf{s}}, pk_{\mathsf{r}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, m):}$

$(pk_{\mathsf{USPCE}}, pk'_{\mathsf{J}}) \leftarrow pk_{\mathsf{J}},\ r \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{RS}$

$c \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{Encap_c}(pp_{\mathsf{KEM}}; r),\ k_{\mathsf{r}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{Encap_k}(pp_{\mathsf{KEM}}, pk_{\mathsf{r}}; r)$

$t \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{T},\ k_{\mathsf{J}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{dEncap_k}(pp_{\mathsf{KEM}}, pk'_{\mathsf{J}}, t; r)$

$r_{\mathsf{USPCE}} \leftarrow \mathsf{USPCE}.\mathcal{RS},\ c_{\mathsf{t}} \leftarrow \mathsf{USPCE}.\mathsf{Enc}(pk_{\mathsf{USPCE}}, m, t; r_{\mathsf{USPCE}})$

$x \leftarrow (sk_{\mathsf{s}}, t, r, \perp, \perp, r_{\mathsf{USPCE}}),\ y \leftarrow (pp, pk_{\mathsf{s}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}, m)$

$\pi \leftarrow \mathsf{NIZK}^{\mathcal{R}}.\mathsf{Prove}(pk_{\mathsf{r}}, y, x)$ // $\mathsf{NIZK}^{\mathcal{R}}$ will be explained later

Return $\sigma \leftarrow (\pi, c, k_{\mathsf{r}_i}, k_{\mathsf{J}}, c_{\mathsf{t}})$

$\underline{\mathsf{Verify}(pp, pk_{\mathsf{s}}, sk_{\mathsf{r}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, m, \sigma):}$

$(\pi, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}) \leftarrow \sigma,\ y \leftarrow (pp, pk_{\mathsf{s}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}, m)$

If $\mathsf{NIZK}^{\mathcal{R}}.\mathsf{Verify}(pk_{\mathsf{r}}, \pi, y) = 0$: Return 0

If $\mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{Decap}(pp_{\mathsf{KEM}}, sk_{\mathsf{r}}, c) = k_{\mathsf{r}}$: Return 1

Return 0

Figure 14: Algorithms Frank and Verify of MAMF

$\underline{\mathsf{TKGen}(pp, sk_{\mathsf{Ag}}, pk_{\mathsf{J}}, m)}$:
$\mathsf{tk} \leftarrow \mathsf{USPCE.TKGen}(pp_{\mathsf{USPCE}}, msk_{\mathsf{USPCE}}, m)$
Return tk

$\underline{\mathsf{Judge}(pp, pk_{\mathsf{s}}, pk_{\mathsf{r}}, pk_{\mathsf{Ag}}, sk_{\mathsf{J}}, m, \sigma, \mathsf{tk})}$:
$(sk_{\mathsf{USPCE}}, sk'_{\mathsf{J}}) \leftarrow sk_{\mathsf{J}}$, $(\pi, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}) \leftarrow \sigma$, $y \leftarrow (pp, pk_{\mathsf{s}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}, m)$
If $\mathsf{NIZK}^{\mathcal{R}}.\mathsf{Verify}(pk_{\mathsf{r}}, \pi, y) = 0$: Return 0
If $\mathsf{tk} \neq \perp$:
  $t' \leftarrow \mathsf{USPCE.Dec}(pp_{\mathsf{USPCE}}, sk_{\mathsf{USPCE}}, c_{\mathsf{t}}, \mathsf{tk})$
  If $\mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{dDecap}(pp_{\mathsf{KEM}}, sk'_{\mathsf{J}}, t', c) = k_{\mathsf{J}}$: Return 1
  Return 0
If $\mathsf{tk} = \perp$:
  $S_{\mathsf{t}} \leftarrow \mathsf{USPCE.Dec}(pp_{\mathsf{USPCE}}, sk_{\mathsf{USPCE}}, c_{\mathsf{t}}, \perp)$
  For $t' \in S_{\mathsf{t}}$:
    If $\mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{dDecap}(pp_{\mathsf{KEM}}, sk'_{\mathsf{J}}, t', c) = k_{\mathsf{J}}$: Return 1
  Return 0

Figure 15: Algorithms Judge and TKGen of MAMF

Construction of MAMF IV

$\underline{\mathsf{Forge}(pp, pk_{\mathsf{s}}, pk_{\mathsf{r}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, m):}$

$(pk_{\mathsf{USPCE}}, pk'_{\mathsf{J}}) \leftarrow pk_{\mathsf{J}}$, $r^*_{\mathsf{c}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{RS}^*$

$c \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{Encap}^*_{\mathsf{c}}(pp_{\mathsf{KEM}}; r^*_{\mathsf{c}})$

$r^*_{\mathsf{k}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{RS}^*$, $k_{\mathsf{r}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{SamEncK}(pp_{\mathsf{KEM}}; r^*_{\mathsf{k}})$

$t \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{T}$, $k_{\mathsf{J}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{K}$

changes in $\underline{\mathsf{RForge}(pp, pk_{\mathsf{s}}, sk_{\mathsf{r}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, m):}$

$k_{\mathsf{r}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{Decap}(pp_{\mathsf{KEM}}, sk_{\mathsf{r}}, c)$, $t \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{T}$

$r^*_{\mathsf{k}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{RS}^*$, $k_{\mathsf{J}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{dSamEncK}(pp_{\mathsf{KEM}}, t; r^*_{\mathsf{k}})$

changes in $\underline{\mathsf{JForge}(pp, pk_{\mathsf{s}}, pk_{\mathsf{r}}, pk_{\mathsf{Ag}}, sk_{\mathsf{J}}, m):}$

$(sk_{\mathsf{USPCE}}, sk'_{\mathsf{J}}) \leftarrow sk_{\mathsf{J}}$

$r^*_{\mathsf{k}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{RS}^*$, $k_{\mathsf{r}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{SamEncK}(pp_{\mathsf{KEM}}; r^*_{\mathsf{k}})$

$t \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathcal{T}$, $k_{\mathsf{J}} \leftarrow \mathsf{dHPS\text{-}KEM}^{\Sigma}.\mathsf{dDecap}(pp_{\mathsf{KEM}}, sk'_{\mathsf{J}}, t, c)$

$r_{\mathsf{USPCE}} \leftarrow \mathsf{USPCE}.\mathcal{RS}$, $c_{\mathsf{t}} \leftarrow \mathsf{USPCE}.\mathsf{Enc}(pk_{\mathsf{USPCE}}, m, t; r_{\mathsf{USPCE}})$

$x \leftarrow (\perp, t, \perp, r^*_{\mathsf{c}}, r^*_{\mathsf{k}}, r_{\mathsf{USPCE}})$, $y \leftarrow (pp, pk_{\mathsf{s}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}, m)$

$\pi \leftarrow \mathsf{NIZK}^{\mathcal{R}}.\mathsf{Prove}(pk_{\mathsf{r}}, y, x)$, Return $\sigma \leftarrow (\pi, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}})$

Figure 16: Algorithm: Forge, RForge and JForge

The NIZK relation in MAMF and security analysis I

Applying Fait-Shamir transform to the Sigma protocols for the following relation, we can get an efficient NIZK scheme $\mathsf{NIZK}^{\mathcal{R}}$.

$$\mathcal{R} = \{((pp, pk_{\mathsf{s}}, pk_{\mathsf{Ag}}, pk_{\mathsf{J}}, c, k_{\mathsf{r}}, k_{\mathsf{J}}, c_{\mathsf{t}}, m), (sk_{\mathsf{s}}, t, r, r_{\mathsf{c}}^*, r_{\mathsf{k}}^*, r_{\mathsf{USPCE}})) : (A_1 \wedge A_2) \vee (A_3 \wedge A_4) \vee (A_3 \wedge A_5)\}$$

$$A_1 : (pk_{\mathsf{s}}, sk_{\mathsf{s}}) \in \mathcal{R}_{\mathsf{s}}, \qquad A_3 : (c, r_{\mathsf{c}}^*) \in \mathcal{R}_{\mathsf{c}}^*$$

$$A_2 : ((c, k_{\mathsf{J}}, pk_{\mathsf{J}}), (t, r)) \in \mathcal{R}_{\mathsf{c,k}}^{\mathsf{d}} \ \wedge_{\mathsf{eq}} \ ((pk_{\mathsf{USPCE}}, m, c_{\mathsf{t}}), (t, r_{\mathsf{USPCE}})) \in \mathcal{R}_{\mathsf{ct}}$$

$$A_4 : (k_{\mathsf{r}}, r_{\mathsf{k}}^*) \in \mathcal{R}_{\mathsf{k}}^* \ \wedge \ ((pk_{\mathsf{USPCE}}, m, c_{\mathsf{t}}), (t, r_{\mathsf{USPCE}})) \in \mathcal{R}_{\mathsf{ct}}$$

$$A_5 : (k_{\mathsf{J}}, (t, r_{\mathsf{k}}^*)) \in \mathcal{R}_{\mathsf{k}}^{\mathsf{d}*} \ \wedge_{\mathsf{eq}} \ ((pk_{\mathsf{USPCE}}, m, c_{\mathsf{t}}), (t, r_{\mathsf{USPCE}})) \in \mathcal{R}_{\mathsf{ct}}$$

- $A_1 \wedge A_2$: the sender's secret key is known, and $(c, k_{\mathsf{J}}, c_{\mathsf{t}})$ are generated by Frank. ($\Rightarrow$ accountability)
- $A_3 \wedge A_4$: c is ill-formed, and k_{r} is obtained by $\mathsf{SamEncK}(pp_{\mathsf{KEM}}; r_{\mathsf{k}}^*)$. ($\Rightarrow$ universal and judge compromise deniability)

The NIZK relation in MAMF and security analysis II

- $A_3 \wedge A_5$: c is ill-formed, and k_{J} is obtained by $\mathsf{dSamEncK}(pp_{\mathsf{KEM}}, t; r^*_{\mathsf{k}})$. ($\Rightarrow$ receiver compromise deniability)
- $(A_3 \wedge A_4) \vee (A_3 \wedge A_5)$: In fact, when A_3 is true, only A_4 or A_5 can be true. ($\Rightarrow$ unframeability)
- $(A_1 \wedge A_2) \vee (A_3 \wedge A_4) \vee (A_3 \wedge A_5)$: It guarantees that the signature would be accepted by receiver and the judge, only when it is generated by Frank. ($\Rightarrow$ unforgeability)

Untraceability and confidentiality of sets are guaranteed by the underlying USPCE.

MAMF concrete construction

In our paper [HLZW24], We present the concrete constructions of USPCE and dual HPS-KEM, thus implying a concrete construction of MAMF.

For more details, please refer to our paper [HLZW24].

References

[BGJP23] James Bartusek, Sanjam Garg, Abhishek Jain, and Guru-Vamsi Policharla.
End-to-end secure messaging with traceability only for illegal content.
In *EUROCRYPT 2023*, pages 35–66. Springer, 2023.

[CS02] Ronald Cramer and Victor Shoup.
Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption.
In *EUROCRYPT*, pages 45–64. Springer, 2002.

[Fac16] Facebook.
Messenger secret conversations technical whitepaper.
2016.
`https://fbnewsroomus.files.wordpress.com/2016/07/secret_conversations_whitepaper-1.pdf`.

[GLR17] Paul Grubbs, Jiahui Lu, and Thomas Ristenpart.
Message franking via committing authenticated encryption.
In *CRYPTO 2017*, pages 66–97. Springer, 2017.

[HLZW24] Zhengan Huang, Junzuo Lai, Gongxian Zeng, and Jian Weng.
Mild asymmetric message franking: Illegal-messages-only and retrospective content moderation.
IACR Cryptol. ePrint Arch., page 1608, 2024.

[LZH+23] Junzuo Lai, Gongxian Zeng, Zhengan Huang, Siu Ming Yiu, Xin Mu, and Jian Weng.
Asymmetric group message franking: Definitions and constructions.
In *EUROCRYPT 2023*, pages 67–97. Springer, 2023.

[TGL+19] Nirvan Tyagi, Paul Grubbs, Julia Len, Ian Miers, and Thomas Ristenpart.
Asymmetric message franking: Content moderation for metadata-private end-to-end encryption.
In *CRYPTO 2019*, pages 222–250. Springer, 2019.

Thank you