A Posteriori Openable Public Key Encryption*

Xavier Bultel[1,2] Pascal Lafourcade[1,2]

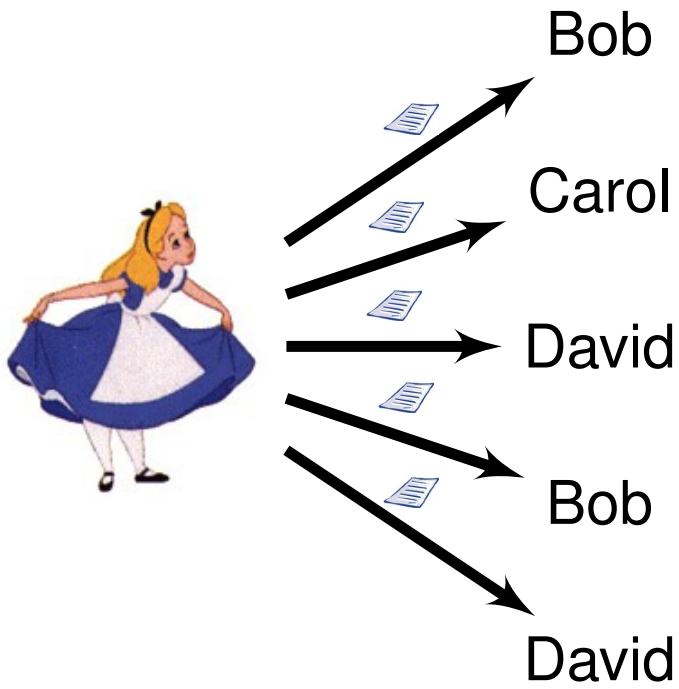
CNRS, UMR 6158, LIMOS, F-63173 Aubière, France

Université Clermont Auvergne, LIMOS, BP 10448, 63000 Clermont-Ferrand, France

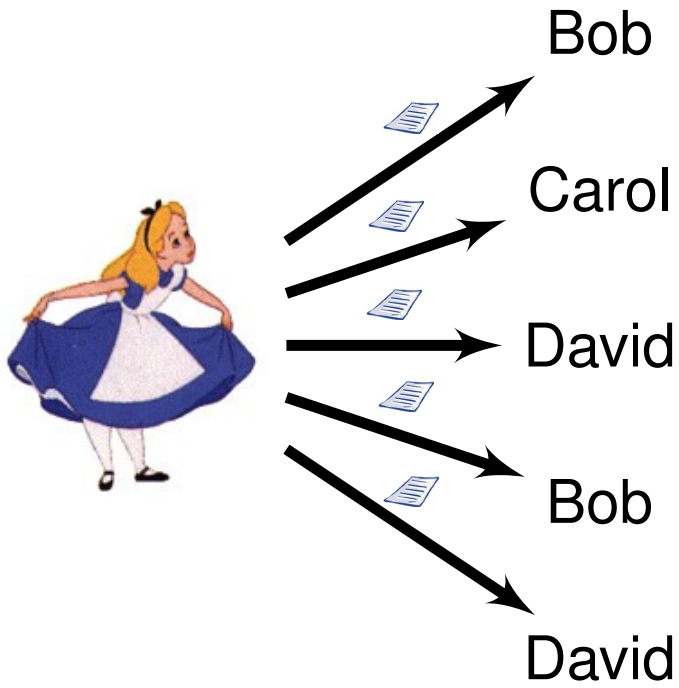
May 27, 2016

*This research was conducted with the support of the "Digital Trust" Chair from the University of Auvergne Foundation.

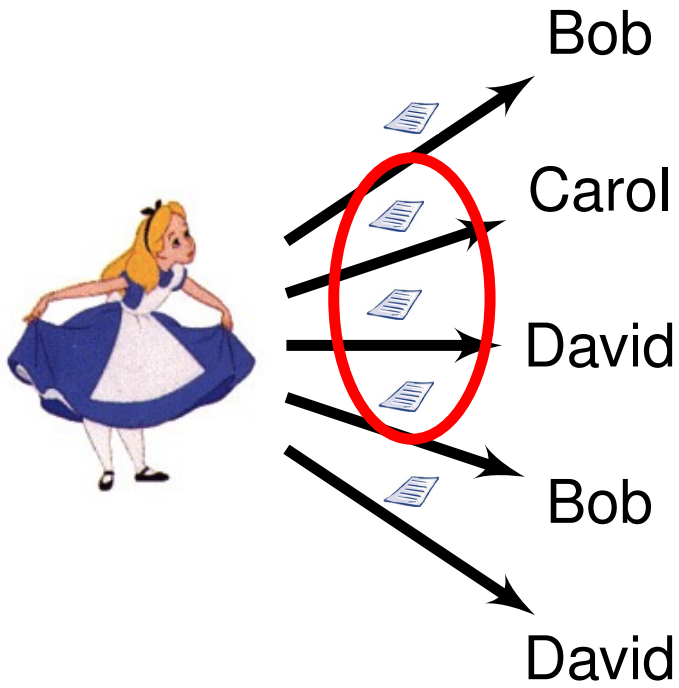
Motivation



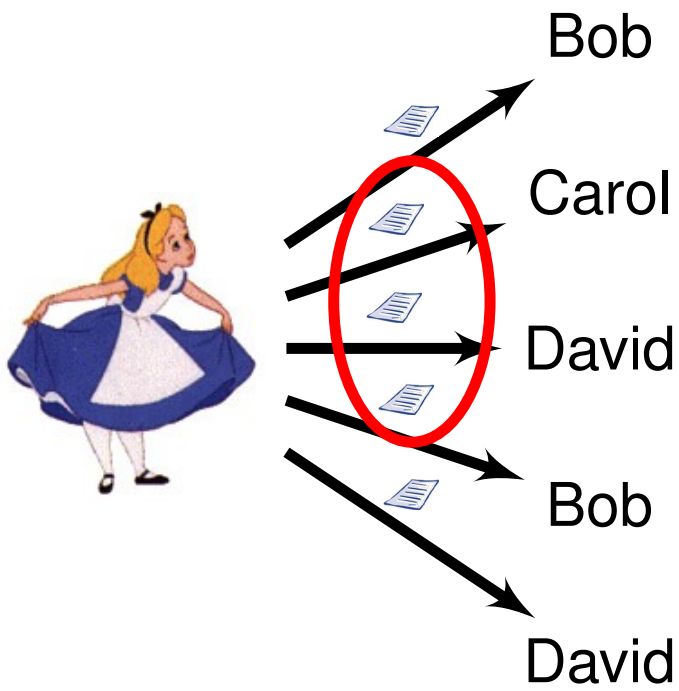
Motivation



Motivation



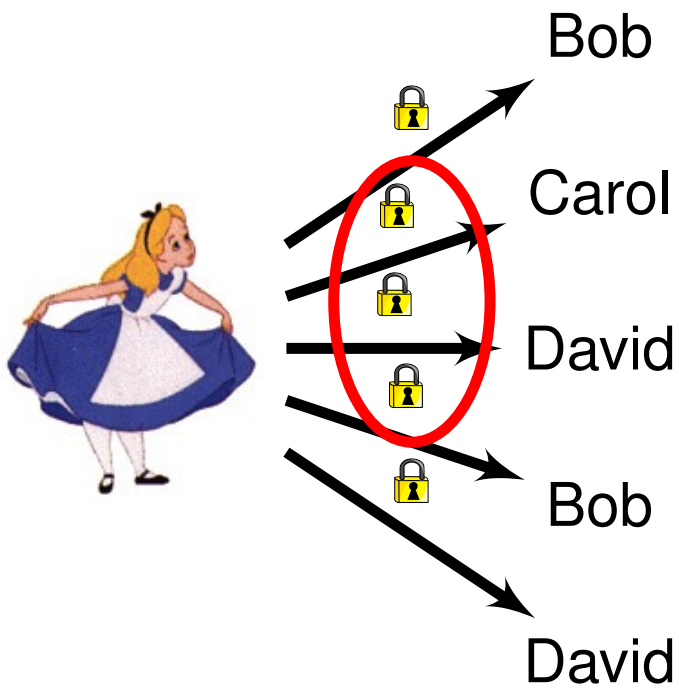
Motivation



Plain mails

No privacy!

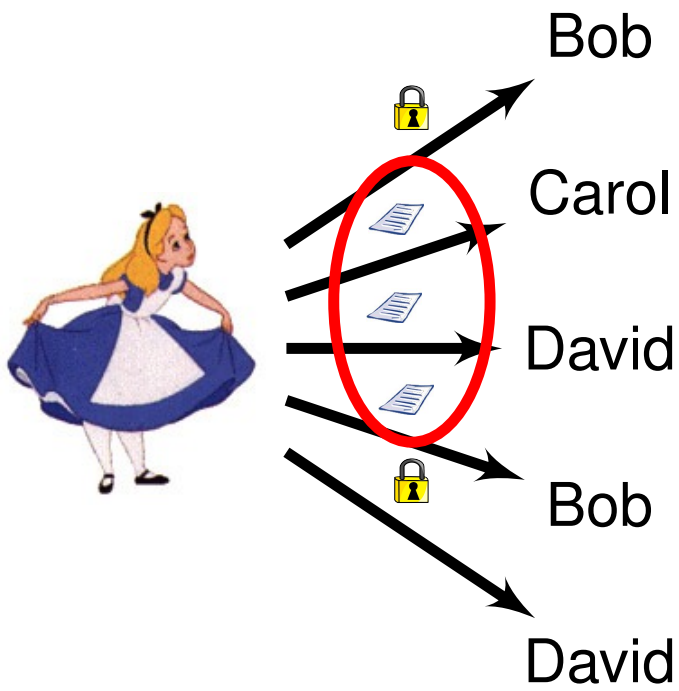
Motivation



~~Plain mails~~
~~No privacy!~~

Encrypted mails
No evidence!

Motivation



~~Plain mails~~
~~No privacy!~~

~~Encrypted mails~~
~~No evidence!~~

Using APO PKE
Privacy & evidence!

Naive solution

Using ElGamal encryption:

- The ith messages m_i for public key pk_i is encrypted as follows.
  - Picks $r_i \xleftarrow{\$} \mathbb{Z}_p^*$.
  - Compute $c_i = (g^{r_i}, \mathsf{pk}_i^{r_i} \cdot m_i)$.
- Alice stores the random coin r_i.
- For the interval $[a, b]$, Alice sends $K = \{r_i\}_{a \leq i \leq b}$ to the judge.
- The judge opens c_i as follows:
  - $m_i = \frac{\mathsf{pk}_i^{r_i} \cdot m_i}{\mathsf{pk}_i^{r_i}}$.

Problematic

Problems

- Storage grows linearly with the number of messages.
- K grows linearly with the number of messages in the interval.

Our contribution

- an efficient and secure solution where the storage size, the interval key size, and this key generation complexity are constants.

Related Works

Time-Release Encryption (T. May, 1993):

- Notion of time
- Use a time server (proxy)
- Decryption in the future
- Decryption rights chosen a priori

Time Specific Encryption
(Kenneth G. Paterson and Elizabeth A. Quaglia, 2010):

- Extension of TRE
- Interval of time
- Limited encryption

Formal Definition: APO-PKE

A Posteriori Openable Public Key Encryption

$\mathsf{APOgen}(1^k)$: return $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{APOini}(1^k)$: initialize st.

$\mathsf{APOenc}^{\mathsf{st}}_{\mathsf{pk}}(m)$: return C, update st.

$\mathsf{APOdec}_{\mathsf{sk}}(C)$: return m.

$\mathsf{APOext}^{\mathsf{st}}_{\mathsf{pko}}(C_i, C_j)$: return $K^{\mathsf{pko}}_{i \to j}$.

$\mathsf{APOpen}_{\mathsf{sko}}(K^{\mathsf{pko}}_{i \to j}, \{C_x\}_{i \leq x \leq j}, \{\mathsf{pk}_x\}_{i \leq x \leq j})$: return $\{m_x\}_{i \leq x \leq j}$.

Chosen Plaintext Attack (CPA)

IND-CPA game

Adversary $\mathcal{A}$		**challenger**
		$b \xleftarrow{\$} \{0,1\}$
	$\xleftarrow{\text{pk}}$	$(\text{pk}, \text{sk}) \leftarrow \text{Gen}(t)$
	$\xrightarrow{(m_0, m_1)}$	$c \leftarrow \text{Enc}_{\text{pk}}(m_b; r)$
	$\xleftarrow{c}$	
	$\xrightarrow{b'}$	If $b = b'$, $\mathcal{A}$ wins.
		If $b \neq b'$, $\mathcal{A}$ loses.

- Secure: $|\Pr[b = b'] - \frac{1}{2}|$ is negligible for any polynomial time $\mathcal{A}$.

Chosen Plaintext Attack (CPA)

IND-CPA game

Adversary $\mathcal{A}$		**challenger**
		$b \xleftarrow{\$} \{0,1\}$
	$\xleftarrow{\text{pk}}$	$(\text{pk}, \text{sk}) \leftarrow \text{Gen}(t)$
	$\xrightarrow{(m_0, m_1)}$	$c \leftarrow \text{Enc}_{\text{pk}}(m_b; r)$
	$\xleftarrow{c}$	
	$\xrightarrow{b'}$	If $b = b'$, $\mathcal{A}$ wins.
		If $b \neq b'$, $\mathcal{A}$ loses.

- Secure: $|\Pr[b = b'] - \frac{1}{2}|$ is negligible for any polynomial time $\mathcal{A}$.

In APO-PKE

Two ways to decrypt (decrypt and open)
$\Rightarrow$ two different IND-CPA security definitions.

Chosen Plaintext Attack (CPA)

OT-IND-CPA security for APO-PKE:

- $\mathcal{C}$ initializes a global state st_*.
- $\mathcal{C}$ encrypts m_0 or m_1 for one receiver.
- $\mathcal{A}$ is a collusion between:
  - The judge.
  - All other receivers.
- Two oracles:
  - **Encryption oracle**: encrypt a chosen message using st_*.
  - **Extraction oracle**: usable only one time. Output an interval key (interval do not contain the challenge).

Chosen Set of Plaintext Attack (CSPA)

IND-CSPA security for APO-PKE:

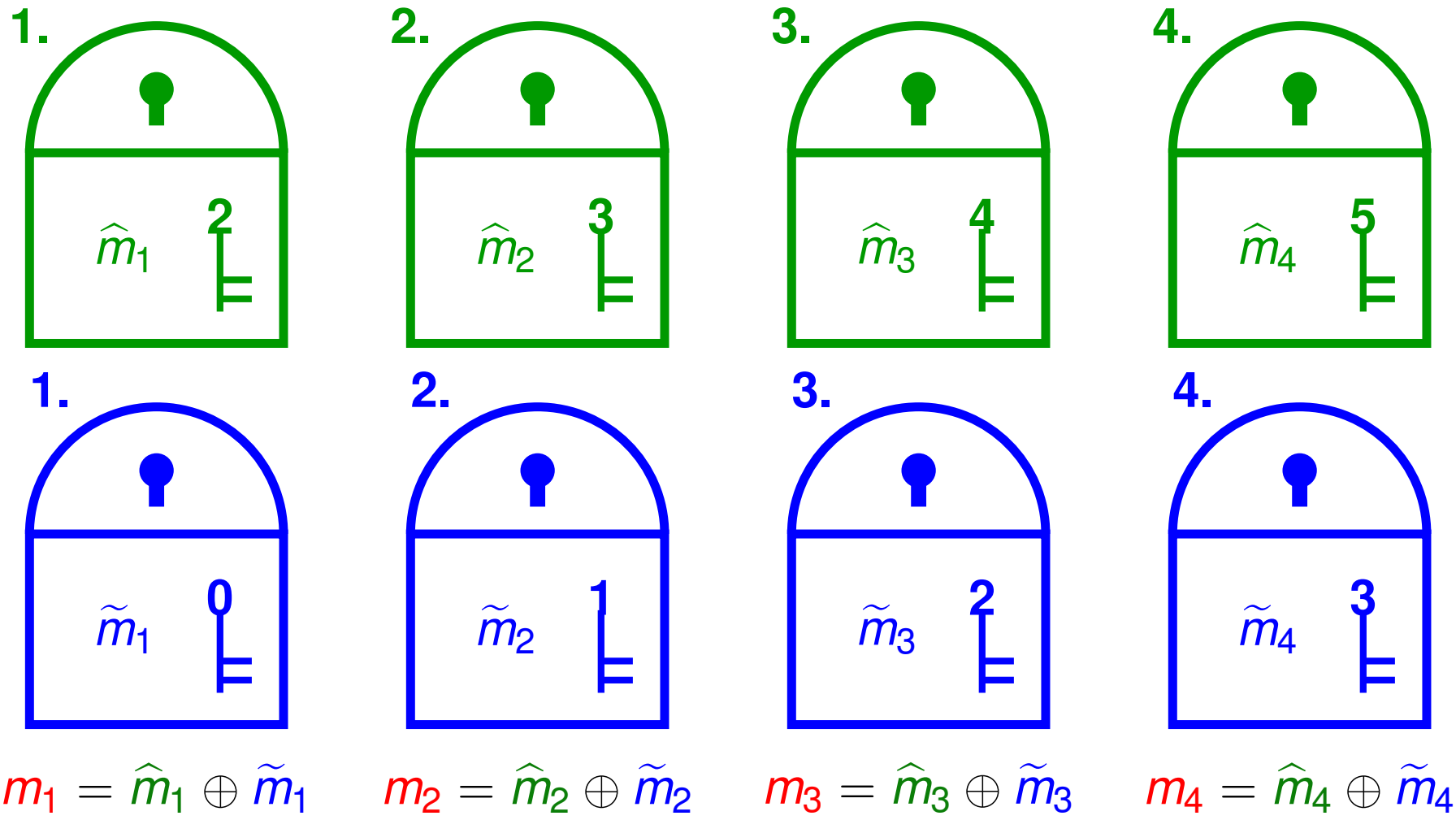
- $\mathcal{C}$ initializes a global stat st_*.
- $\mathcal{C}$ sends the judge public key pko_*.
- $\mathcal{A}$ sends the sets
  - $\{m_x^0\}_{n<x\leq n+q}$
  - $\{m_x^1\}_{n<x\leq n+q}$
  - $\{\mathrm{pk}_x\}_{n<x\leq n+q}$ (public key of honest users only)
- The challenge is
  - The encryptions of $\{m_x^b\}_{n<x\leq(n+q)}$ using $\{\mathrm{pk}_x\}_{n<x\leq n+q}$ and st_*.
  - The interval key $K^{\mathrm{pko}_*}_{n\to n+q}$.
- $\mathcal{A}$ is a collusion between dishonest receivers.
- Three oracles:
  - **Generation oracle**: generate a key for an honest user.
  - **Encryption oracle**: encrypt a chosen message using st_*.
  - **Extraction oracle**: output an interval key for the judge using pko_*.

Integrity

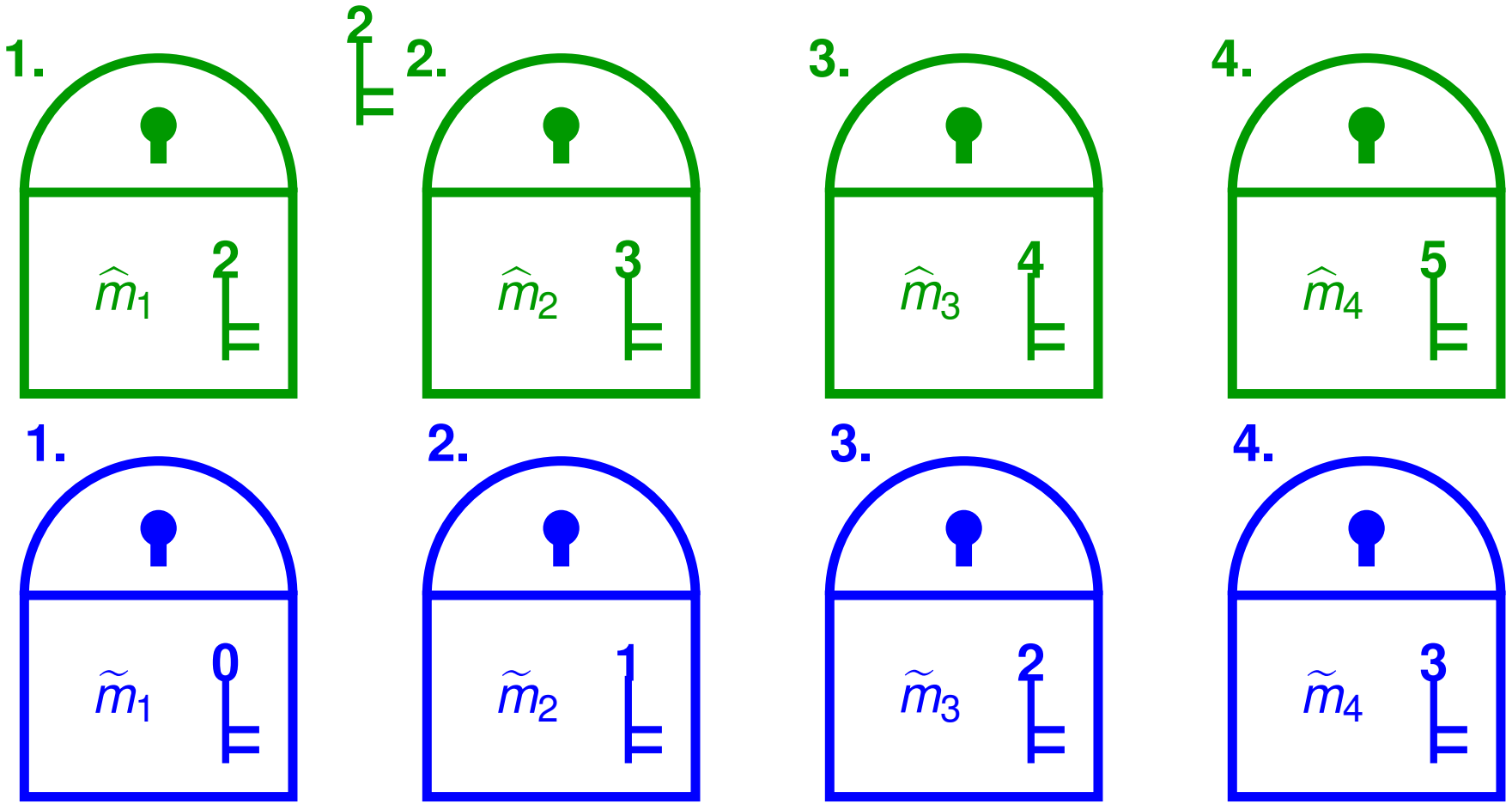
Integrity for APO-PKE:

- The adversary outputs
  - A set of ciphertexts for given receivers.
  - An interval key.
- He wins the experiment if decryption and open algorithm give two different plaintexts for the same ciphertext.

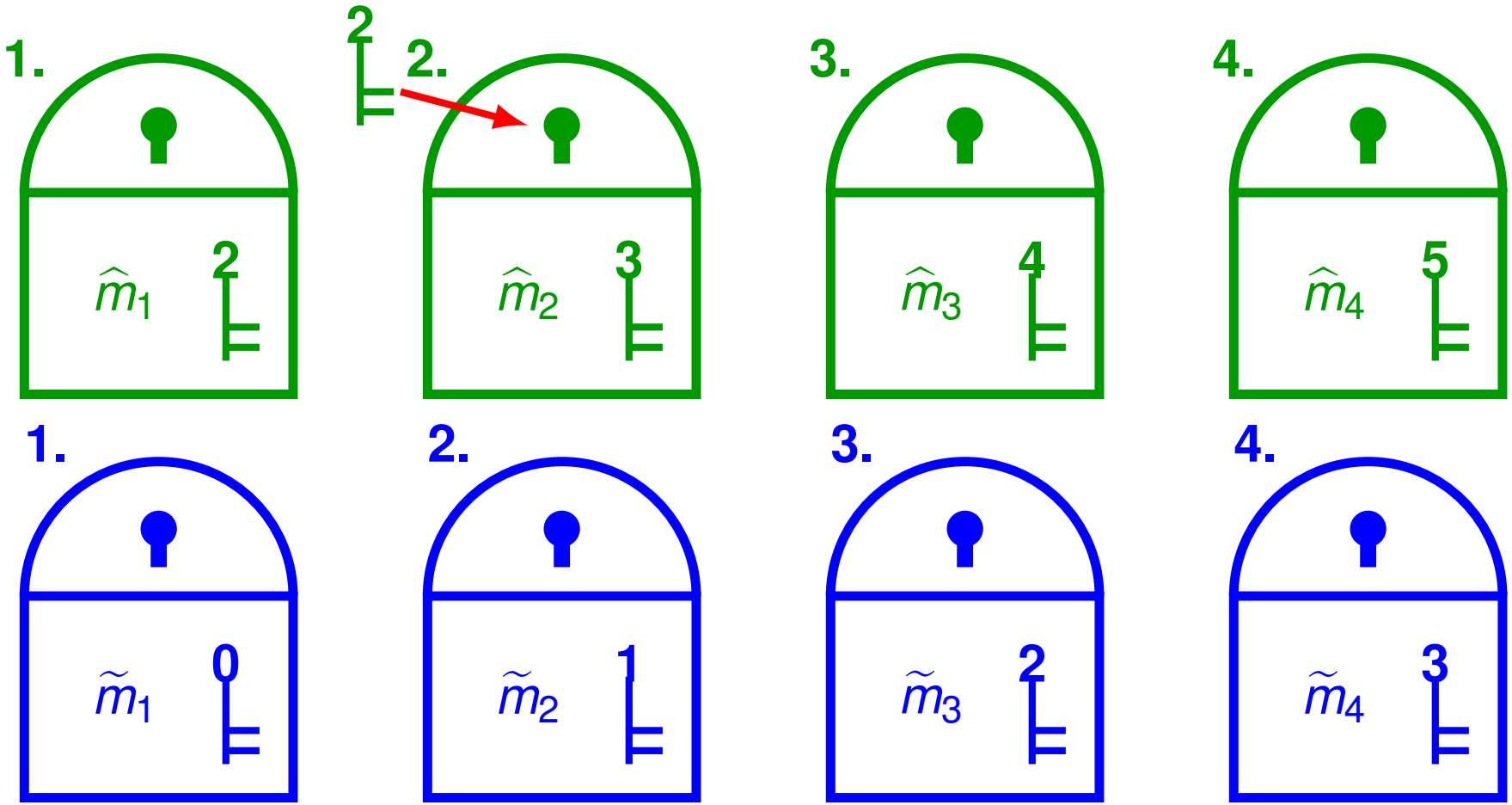
Idea


1.
$\widehat{m}_1$
2
2.
$\widehat{m}_2$
3
3.
$\widehat{m}_3$
4
4.
$\widehat{m}_4$
5
1.
$\widetilde{m}_1$
0
2.
$\widetilde{m}_2$
1
3.
$\widetilde{m}_3$
2
4.
$\widetilde{m}_4$
3
$m_1 = \widehat{m}_1 \oplus \widetilde{m}_1$
$m_2 = \widehat{m}_2 \oplus \widetilde{m}_2$
$m_3 = \widehat{m}_3 \oplus \widetilde{m}_3$
$m_4 = \widehat{m}_4 \oplus \widetilde{m}_4$

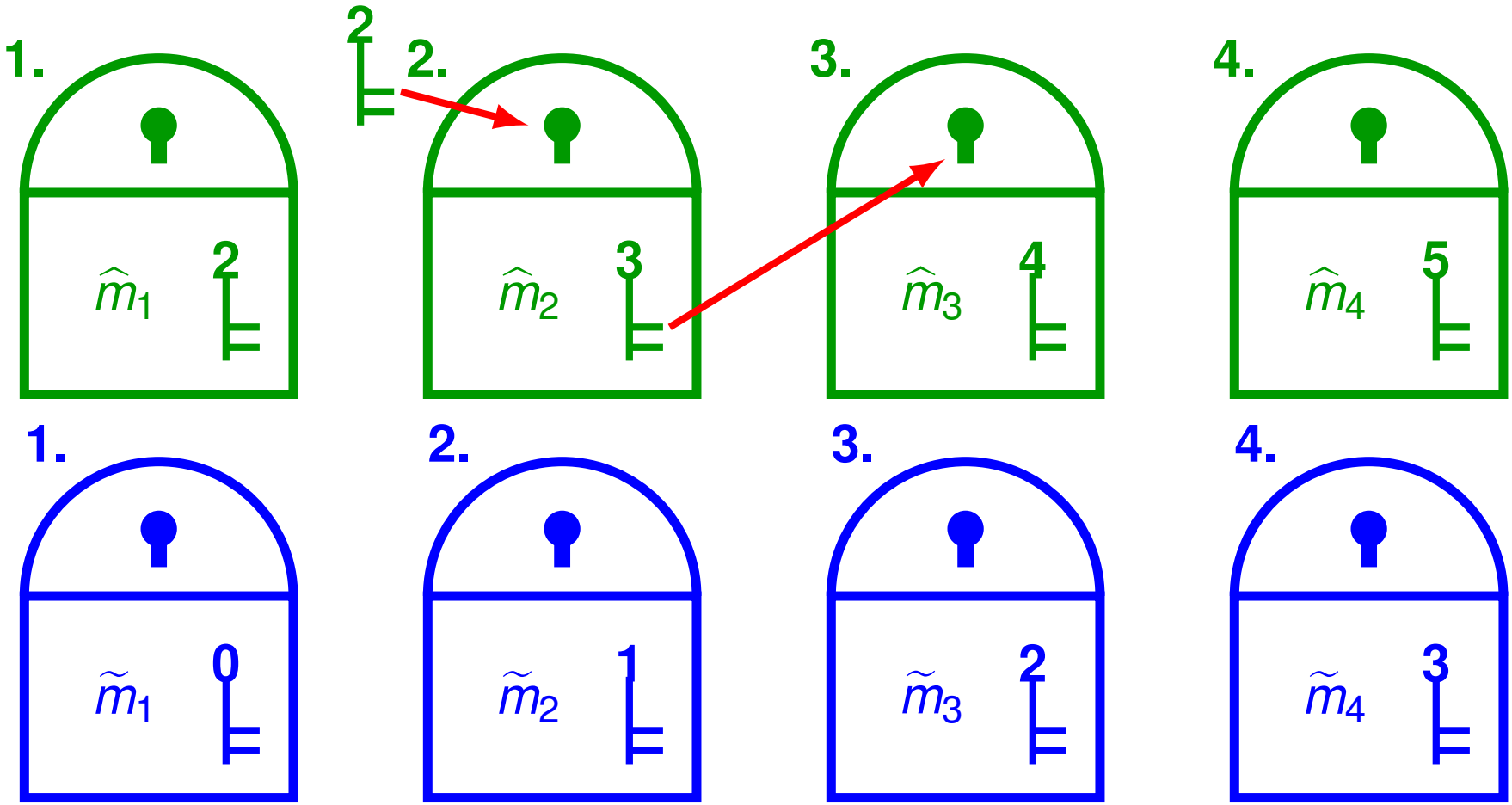
Idea


1.
2
2.
3.
4.
$\widehat{m}_1$
2
$\widehat{m}_2$
3
$\widehat{m}_3$
4
$\widehat{m}_4$
5
1.
2.
3.
4.
$\widetilde{m}_1$
0
$\widetilde{m}_2$
1
$\widetilde{m}_3$
2
$\widetilde{m}_4$
3

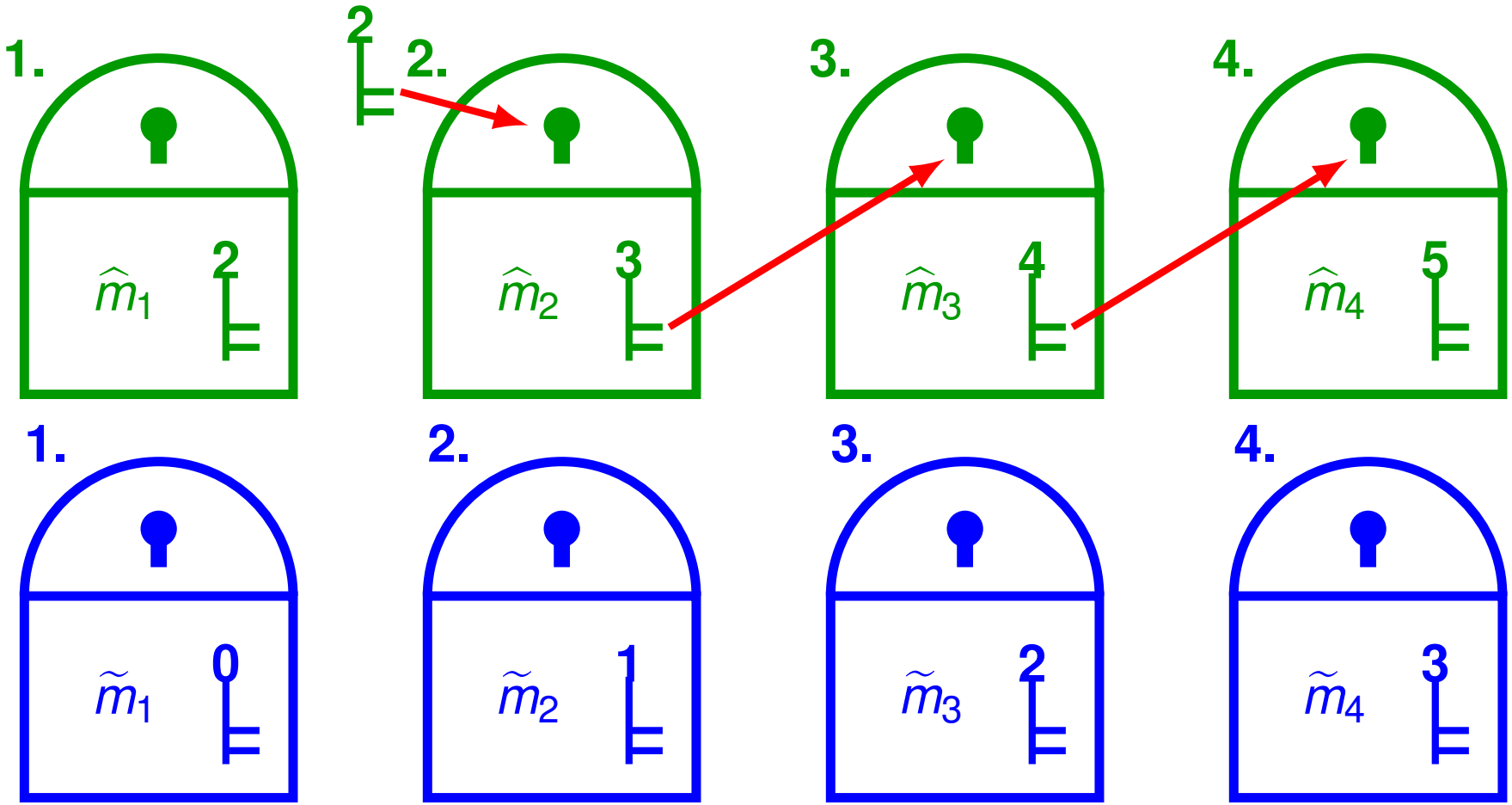
Idea


1.
2
2.
3.
4.
$\widehat{m}_1$
2
$\widehat{m}_2$
3
$\widehat{m}_3$
4
$\widehat{m}_4$
5
1.
2.
3.
4.
$\widetilde{m}_1$
0
$\widetilde{m}_2$
1
$\widetilde{m}_3$
2
$\widetilde{m}_4$
3

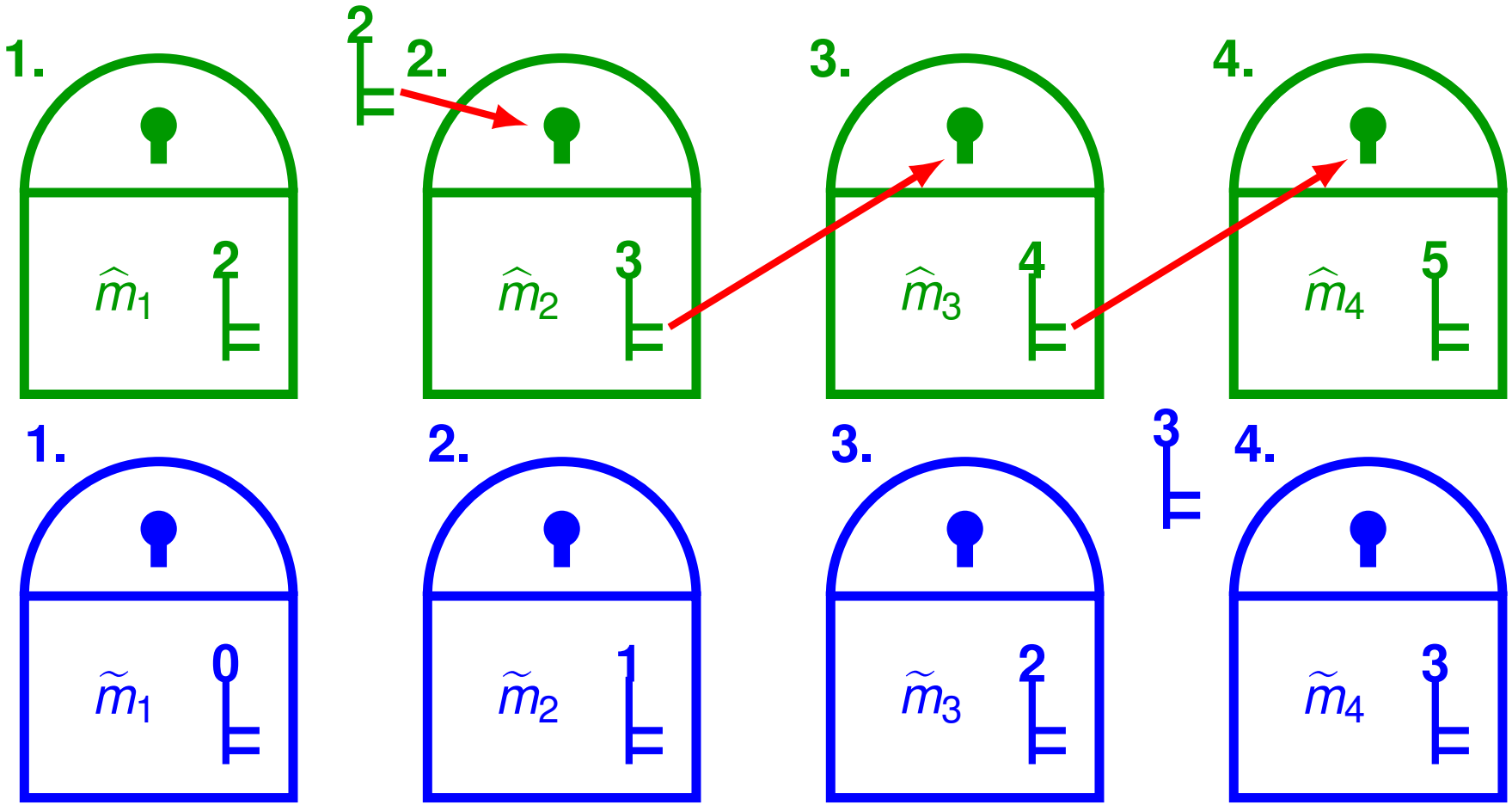
Idea


1.
2
2.
3.
4.
$\widehat{m}_1$
2
$\widehat{m}_2$
3
$\widehat{m}_3$
4
$\widehat{m}_4$
5
1.
2.
3.
4.
$\widetilde{m}_1$
0
$\widetilde{m}_2$
1
$\widetilde{m}_3$
2
$\widetilde{m}_4$
3

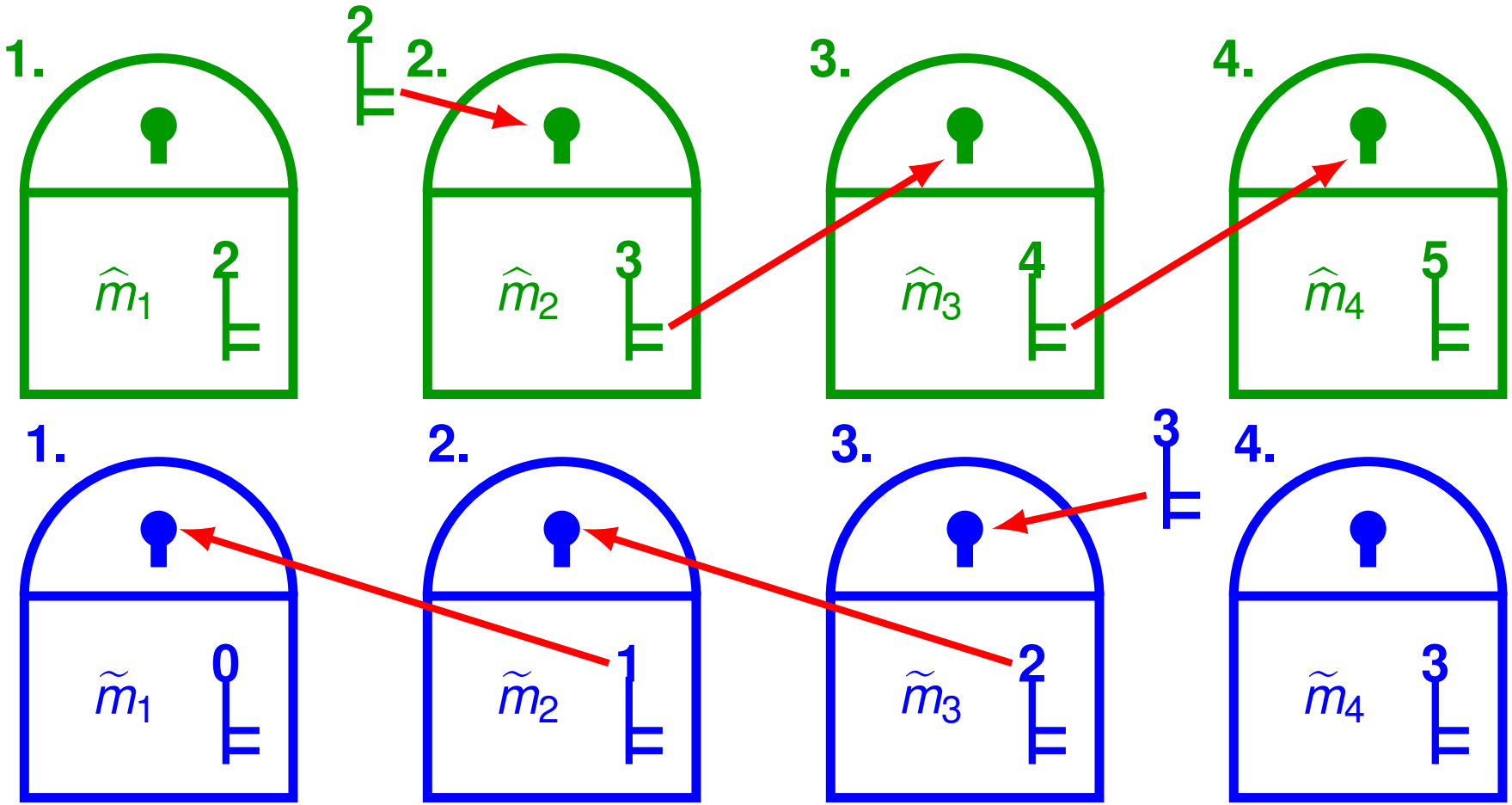
Idea


1.
2
2.
2.
$\hat{m}_1$
2
$\hat{m}_2$
3
3.
$\hat{m}_3$
4
4.
$\hat{m}_4$
5
1.
$\tilde{m}_1$
0
2.
$\tilde{m}_2$
1
3.
$\tilde{m}_3$
2
4.
$\tilde{m}_4$
3

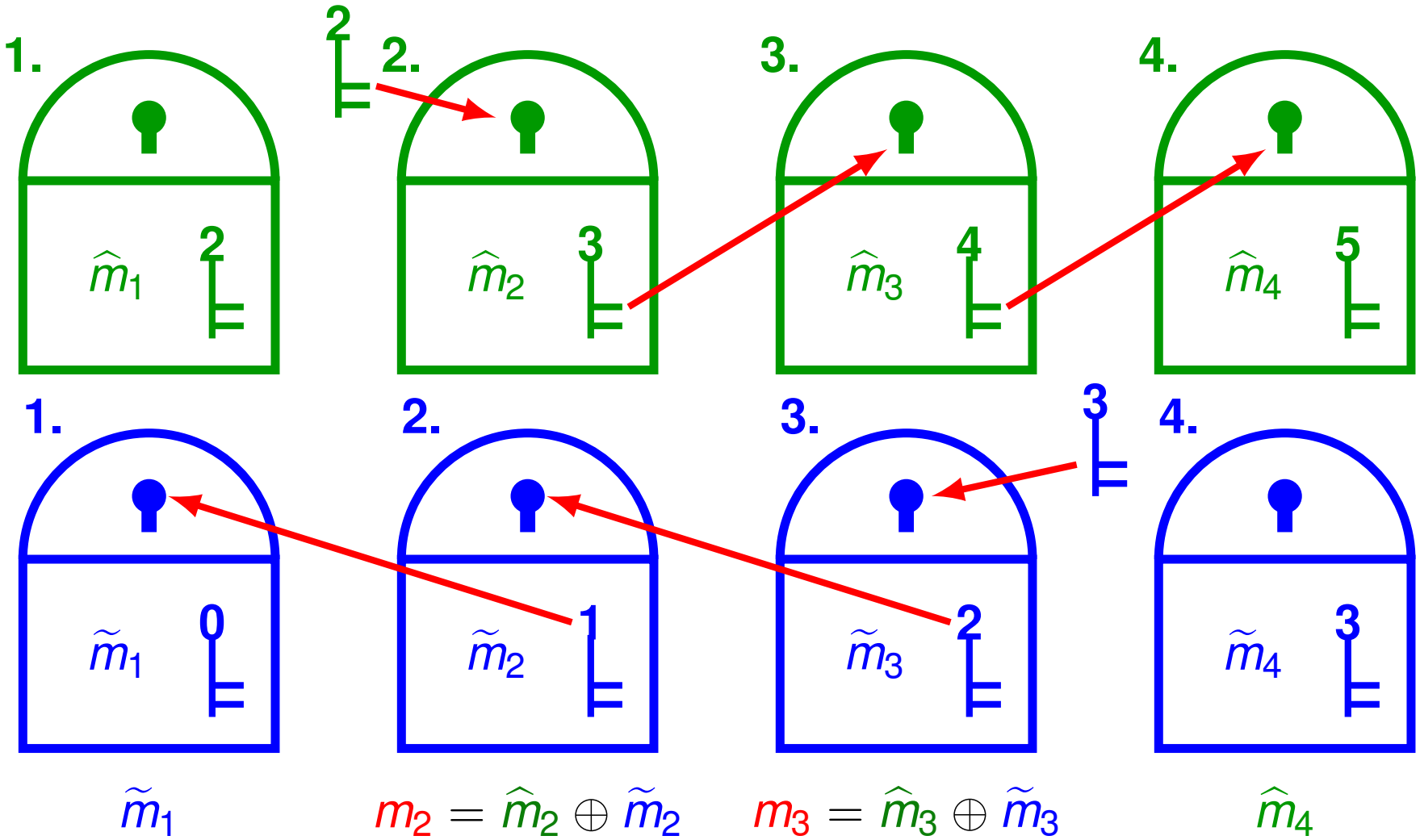
Idea


1.
2
2.
3.
4.
$\widehat{m}_1$
2
$\widehat{m}_2$
3
$\widehat{m}_3$
4
$\widehat{m}_4$
5
1.
2.
3.
3
4.
$\widetilde{m}_1$
0
$\widetilde{m}_2$
1
$\widetilde{m}_3$
2
$\widetilde{m}_4$
3

Idea


1.
2
2.
3.
4.
$\widehat{m}_1$
2
$\widehat{m}_2$
3
$\widehat{m}_3$
4
$\widehat{m}_4$
5
1.
2.
3.
3
4.
$\widetilde{m}_1$
0
$\widetilde{m}_2$
1
$\widetilde{m}_3$
2
$\widetilde{m}_4$
3

Idea


1.
2
2.
3.
4.
$\widehat{m}_1$
2
$\widehat{m}_2$
3
$\widehat{m}_3$
4
$\widehat{m}_4$
5
1.
2.
3.
3
4.
$\widetilde{m}_1$
0
$\widetilde{m}_2$
1
$\widetilde{m}_3$
2
$\widetilde{m}_4$
3
$\widetilde{m}_1$
$m_2 = \widehat{m}_2 \oplus \widetilde{m}_2$
$m_3 = \widehat{m}_3 \oplus \widetilde{m}_3$
$\widehat{m}_4$

Tools

Definition (Random Coin Decryptable PKE (RCD-PKE))

A PKE is RCD-PKE if there exists a PTT algo CDec such that:
$\mathsf{CDec}_\sigma(\mathsf{Enc}_{\mathsf{pk}}(m;\sigma),\mathsf{pk}) = m$.

Definition (Verifiable Key PKE (VK-PKE))

$(\mathsf{pk},\mathsf{sk})$ is valid for $\mathsf{PKE} = (\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ when
$\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m;\sigma)) = m$.
A PKE is verifiable-key when it exists an algorithm Ver such that
$\mathsf{Ver}(\mathsf{pk},\mathsf{sk}) = 1$ iff $(\mathsf{pk},\mathsf{sk})$ is valid.

For example ElGamal is RCD and verifiable.

Our Construction

Generic APO-PKE (G-APO):

Generic construction:

- $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ a RCD verifiable PKE
- 3 hash functions F,G,H.

Our Construction: Keys

APOgen(1^k): Generate $(\mathsf{pk}, \mathsf{sk})$ from Gen.

APOini(1^k):

- $\widehat{\sigma}_0 \xleftarrow{\$} \{0,1\}^k$
- $\widetilde{\sigma}_0 \xleftarrow{\$} \{0,1\}^k$
- $K \xleftarrow{\$} \{0,1\}^k$
- Return $\mathsf{st} = (K||\widehat{\sigma}_0||\widetilde{\sigma}_0)$.

Our Construction: Encryption

$\mathsf{APOenc}^{\mathsf{st}}_{\mathsf{pk}}(m)$: Using $\mathsf{st} = (K||\widehat{\sigma}_{(i-1)}||\widetilde{\sigma}_{(i-1)})$:

- Pick $\widehat{m}$ and $\widetilde{m}$:

$$m = \widehat{m} \oplus \widetilde{m}$$

- Pick $\widehat{\sigma}_i$ and $\widetilde{\sigma}_i$, update the state:

$$\mathsf{st} = (K||\widehat{\sigma}_i||\widetilde{\sigma}_i)$$

- Compute:

$$\widehat{C} = \mathsf{Enc}_{\mathsf{pk}}(\widehat{m}||(\widehat{\sigma}_i \oplus \mathsf{F}(\widehat{\sigma}_{(i-1)})); \mathsf{G}(\widehat{\sigma}_{(i-1)}))$$
$$\widetilde{C} = \mathsf{Enc}_{\mathsf{pk}}(\widetilde{m}||(\widetilde{\sigma}_{(i-1)} \oplus \mathsf{F}(\widetilde{\sigma}_i)); \mathsf{G}(\widetilde{\sigma}_i))$$
$$D = (\widehat{\sigma}_{(i-1)}||\widetilde{\sigma}_i) \oplus \mathsf{H}(K||\widehat{C}||\widetilde{C})$$

- Return $C = (\widehat{C}||\widetilde{C}||D)$.

To decrypt, use decryption algorithm of $\mathcal{E}$ and compute $m = \widehat{m} \oplus \widetilde{m}$.

Our Construction: Extraction

$\mathsf{APOext}^{\mathsf{st}}_{\mathsf{pko}}(C_i, C_j)$: Using values:

- $\mathsf{st} = (K||\widehat{\sigma}||\widetilde{\sigma})$
- $C_i = (\widehat{C}_i||\widetilde{C}_i||D_i)$
- $C_j = (\widehat{C}_j||\widetilde{C}_j||D_j)$

Compute:

- $(\widehat{\sigma}_{i-1}||\widetilde{\sigma}_i) = D_i \oplus \mathsf{H}(K||\widehat{C}_i||\widetilde{C}_i)$
- $(\widehat{\sigma}_{j-1}||\widetilde{\sigma}_j) = D_j \oplus \mathsf{H}(K||\widehat{C}_j||\widetilde{C}_j)$.

Return $K^{\mathsf{pko}}_{i\to j} = \mathsf{Enc}_{\mathsf{pko}}((\widehat{\sigma}_{i-1}||\widetilde{\sigma}_j); r)$.

Our Construction: Open

$\mathsf{APOpen}_{\mathsf{sko}}(K^{\mathsf{pko}}_{i\to j}, \{(\widehat{C}_x||\widetilde{C}_x||D_x)\}_{i\le x\le j}, \{\mathsf{pk}_x\}_{i\le x\le j})$:

- Decrypt $K^{\mathsf{pko}}_{i\to j}$ and deduce $(\widehat{\sigma}_{i-1}||\widetilde{\sigma}_j)$.

Our Construction: Open

$\mathsf{APOpen}_{\mathsf{sko}}(K^{\mathsf{pko}}_{i\to j}, \{(\widehat{C}_x||\widetilde{C}_x||D_x)\}_{i\le x\le j}, \{\mathsf{pk}_x\}_{i\le x\le j})$:

- Decrypt $K^{\mathsf{pko}}_{i\to j}$ and deduce $(\widehat{\sigma}_{i-1}||\widetilde{\sigma}_j)$.
- From $\widehat{\sigma}_{i-1}$:
  - Using $\mathsf{G}(\widehat{\sigma}_{(i-1)})$, decrypt $\widehat{C}_i = \mathsf{Enc}_{\mathsf{pk}_i}(\widehat{m}_i||(\widehat{\sigma}_i \oplus \mathsf{F}(\widehat{\sigma}_{i-1})); \mathsf{G}(\widehat{\sigma}_{i-1}))$.
  - Using $\mathsf{F}(\widehat{\sigma}_{i-1})$, compute $\widehat{\sigma}_i$.
  - Use $\widehat{\sigma}_i$ to decrypt $\widehat{C}_{i+1}$. etc...

Our Construction: Open

$\mathsf{APOpen}_{\mathsf{sko}}(K^{\mathsf{pko}}_{i\to j}, \{(\widehat{C}_x||\widetilde{C}_x||D_x)\}_{i\le x\le j}, \{\mathsf{pk}_x\}_{i\le x\le j})$:

- Decrypt $K^{\mathsf{pko}}_{i\to j}$ and deduce $(\widehat{\sigma}_{i-1}||\widetilde{\sigma}_j)$.
- From $\widehat{\sigma}_{i-1}$:
  - Using $\mathsf{G}(\widehat{\sigma}_{(i-1)})$, decrypt $\widehat{C}_i = \mathsf{Enc}_{\mathsf{pk}_i}(\widehat{m}_i||(\widehat{\sigma}_i \oplus \mathsf{F}(\widehat{\sigma}_{i-1})); \mathsf{G}(\widehat{\sigma}_{i-1}))$.
  - Using $\mathsf{F}(\widehat{\sigma}_{i-1})$, compute $\widehat{\sigma}_i$.
  - Use $\widehat{\sigma}_i$ to decrypt $\widehat{C}_{i+1}$. etc...
- From $\widetilde{\sigma}_j$:
  - Using $\mathsf{G}(\widetilde{\sigma}_j)$, decrypt $\widetilde{C}_j = \mathsf{Enc}_{\mathsf{pk}_j}(\widetilde{m}||(\widetilde{\sigma}_{j-1} \oplus \mathsf{F}(\widetilde{\sigma}_j)); \mathsf{G}(\widetilde{\sigma}_j))$.
  - Using $\mathsf{F}(\widetilde{\sigma}_j)$, compute $\widetilde{\sigma}_{j-1}$.
  - Use $\widehat{\sigma}_{j-1}$ to decrypt $\widetilde{C}_{j-1}$. etc...

Our Construction: Open

$\mathsf{APOpen}_{\mathsf{sko}}(K^{\mathsf{pko}}_{i\to j}, \{(\widehat{C}_x||\widetilde{C}_x||D_x)\}_{i\le x\le j}, \{\mathsf{pk}_x\}_{i\le x\le j})$:

- Decrypt $K^{\mathsf{pko}}_{i\to j}$ and deduce $(\widehat{\sigma}_{i-1}||\widetilde{\sigma}_j)$.
- From $\widehat{\sigma}_{i-1}$:
  - Using $\mathsf{G}(\widehat{\sigma}_{(i-1)})$, decrypt $\widehat{C}_i = \mathsf{Enc}_{\mathsf{pk}_i}(\widehat{m}_i||(\widehat{\sigma}_i \oplus \mathsf{F}(\widehat{\sigma}_{i-1})); \mathsf{G}(\widehat{\sigma}_{i-1}))$.
  - Using $\mathsf{F}(\widehat{\sigma}_{i-1})$, compute $\widehat{\sigma}_i$.
  - Use $\widehat{\sigma}_i$ to decrypt $\widehat{C}_{i+1}$. etc...
- From $\widetilde{\sigma}_j$:
  - Using $\mathsf{G}(\widetilde{\sigma}_j)$, decrypt $\widetilde{C}_j = \mathsf{Enc}_{\mathsf{pk}_j}(\widetilde{m}||(\widetilde{\sigma}_{j-1} \oplus \mathsf{F}(\widetilde{\sigma}_j)); \mathsf{G}(\widetilde{\sigma}_j))$.
  - Using $\mathsf{F}(\widetilde{\sigma}_j)$, compute $\widetilde{\sigma}_{j-1}$.
  - Use $\widehat{\sigma}_{j-1}$ to decrypt $\widetilde{C}_{j-1}$. etc...
- $m_x = \widetilde{m}_x \oplus \widehat{m}_x$ for $i \le x \le j$.

Our Construction: Security

Theorem

Let E be an IND-CPA secure verifiable RCD-PKE, then G-APO based on E is OT-IND-CPA and IND-CSPA secure in the random oracle model, and it satisfies the integrity property.

Conclusion

Our construction is:

- Generic.
- Efficient.
  - Constant size for interval key, storage and extraction complexity.
  - Small size of ciphertexts.
- CPA secure.

Conclusion

Our construction is:

- Generic.
- Efficient.
  - Constant size for interval key, storage and extraction complexity.
  - Small size of ciphertexts.
- CPA secure.

Future works:

- Several interval keys for one opener.
- Chosen ciphertext security.
- Without random oracle.

Thanks for your attention.

Questions?