https://ia.cr/2024/1454

Interval Key-Encapsulation Mechanism

Alexander Bienstock
J.P. Morgan AI Research &
J.P. Morgan AlgoCRYPT CoE

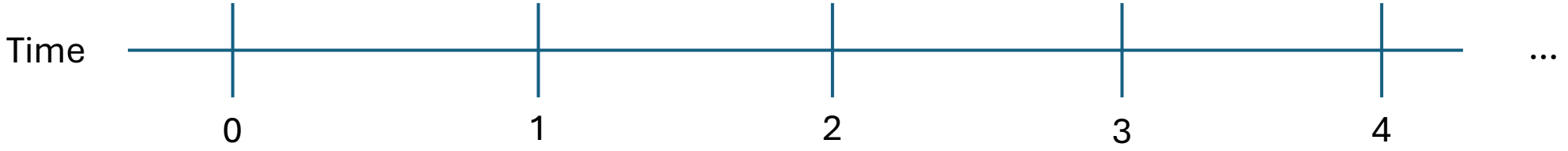
Yevgeniy Dodis
New York University

Paul Rösler
FAU Erlangen-Nürnberg

Daniel Wichs
Northeastern University & NTT Research

Starting Point: Forward Secure KEM [CHK03]

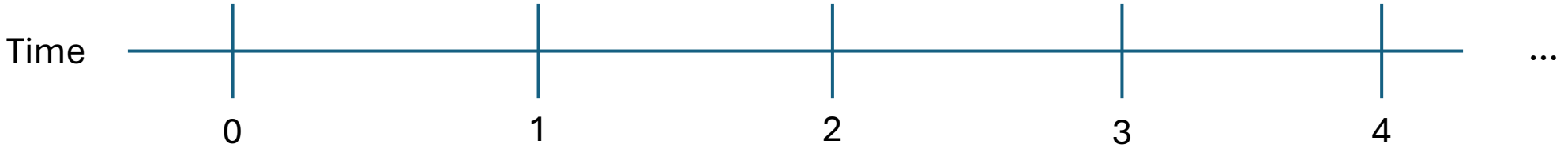
$(sk_0, pk) \leftarrow Gen()$



Starting Point: Forward Secure KEM [CHK03]

pk

sk_0

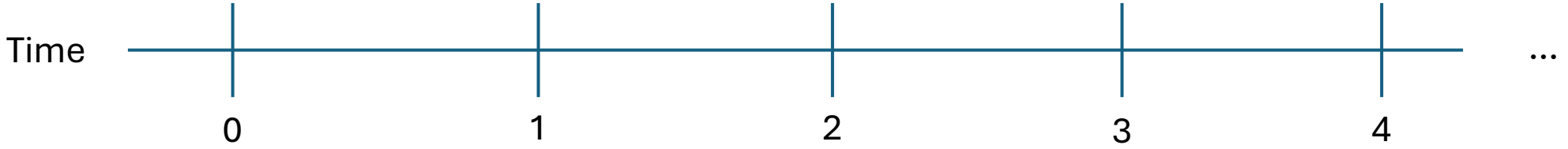


Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

sk_0

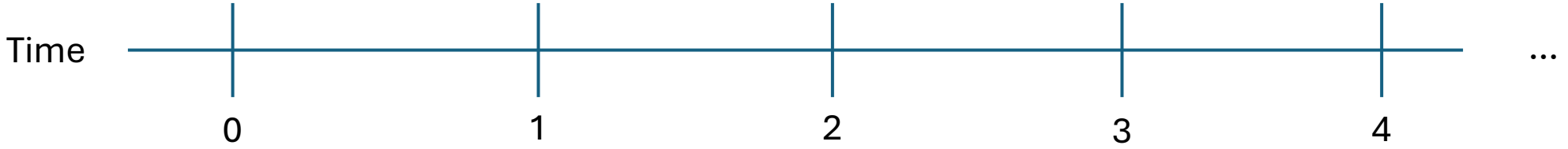


Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

sk_0

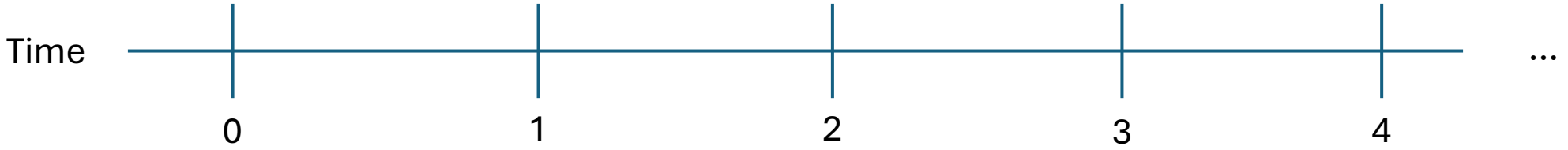


Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

sk_0 $k_0 \leftarrow Dec(sk_0, 0, ct_0)$



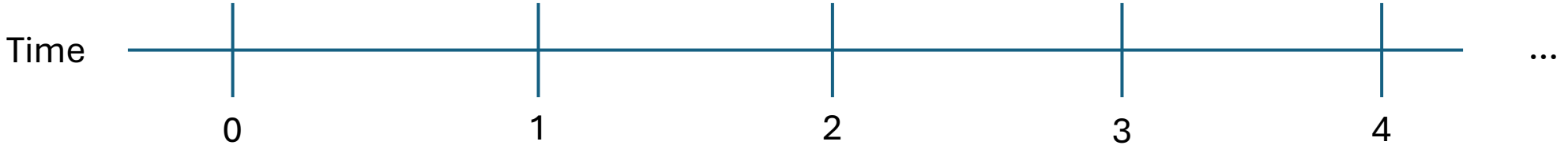
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_2, ct_2) \leftarrow Enc(pk, 2)$

$\boxed{(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)}$

sk_0



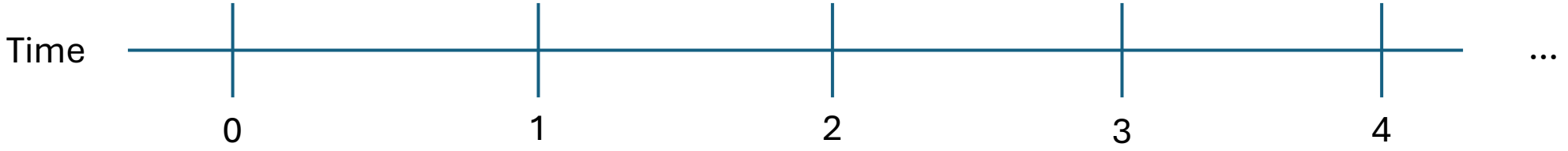
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

sk_0



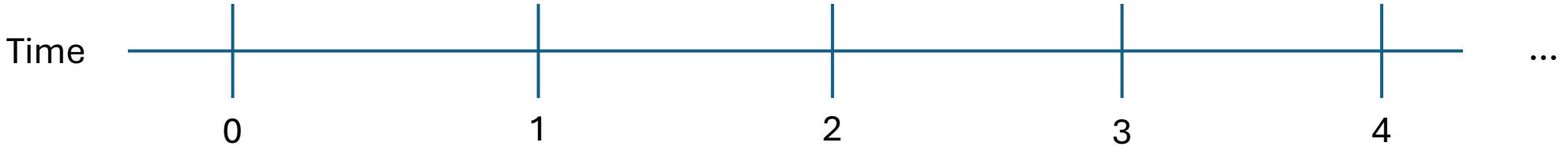
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

sk_0 $k_2 \leftarrow Dec(sk_0, 2\ ct_2)$



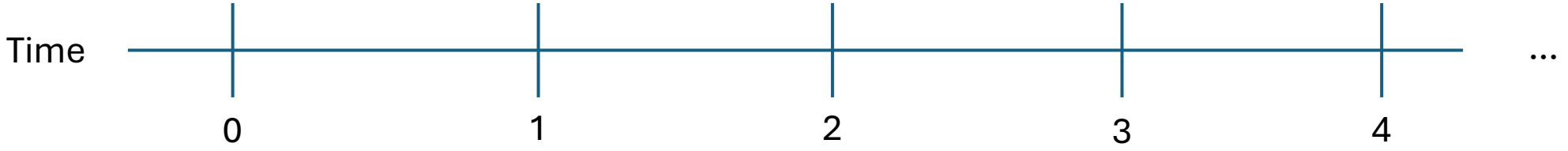
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

sk_0



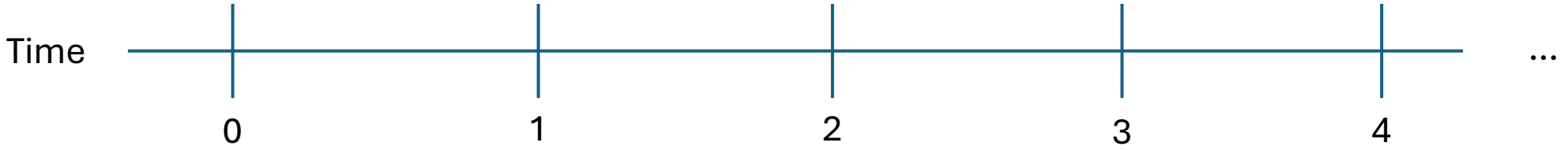
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

$sk_1 \leftarrow Up(sk_0)$



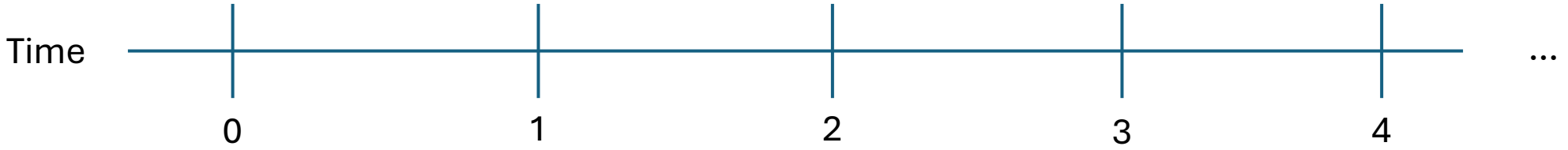
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

$sk_2 \leftarrow Up(sk_1)$



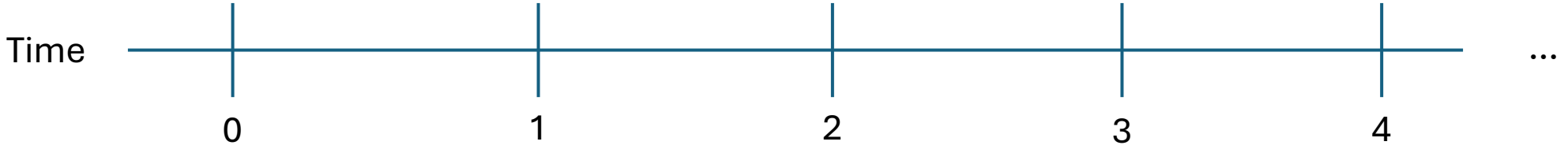
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

sk_2



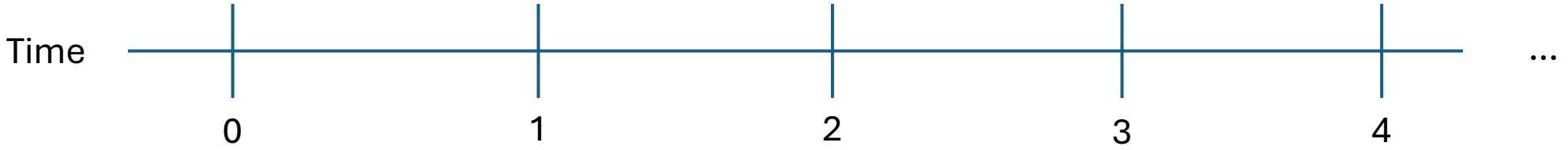
Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

sk_2



Starting Point: Forward Secure KEM [CHK03]

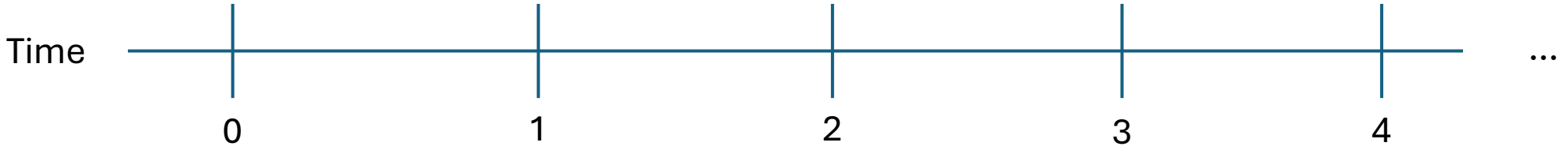
pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

Insecure

sk_2



Starting Point: Forward Secure KEM [CHK03]

pk

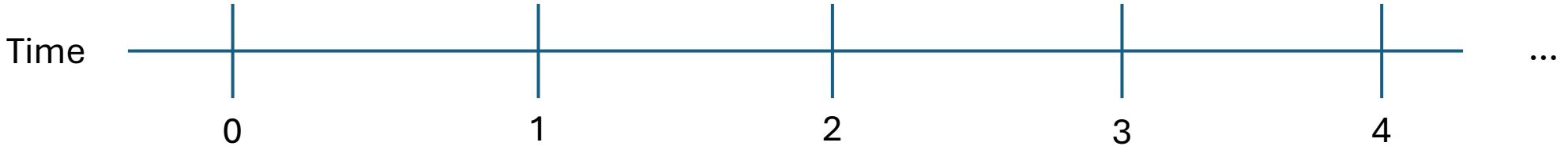
$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

Secure – FS!

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

Insecure

sk_2



Starting Point: Forward Secure KEM [CHK03]

pk

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

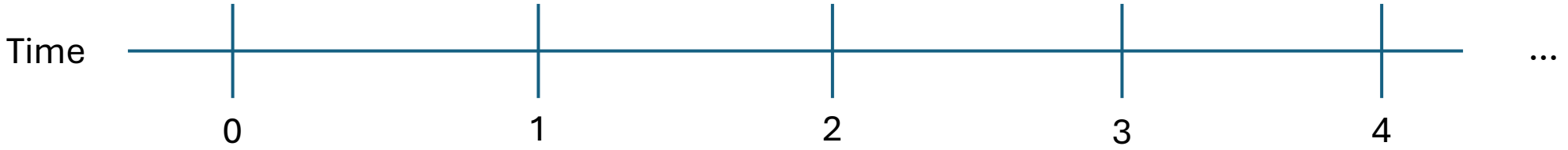
Secure – FS!

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

Insecure

sk_2

[CHK03] construct FS-KEM using HIBE-like techniques



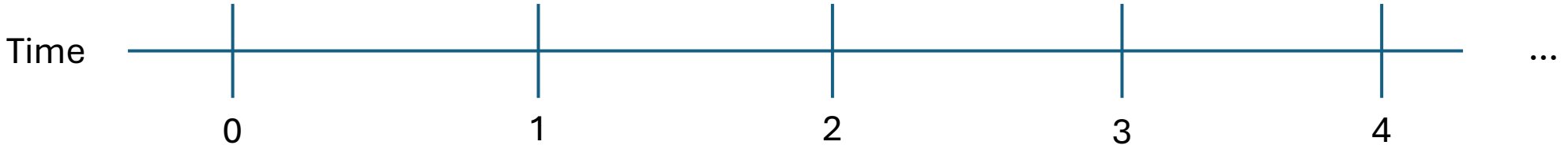
Disadvantages of FS-KEM

pk

sk_2

[CHK03] construct FS-KEM using HIBE-like techniques

INEFFICIENT!

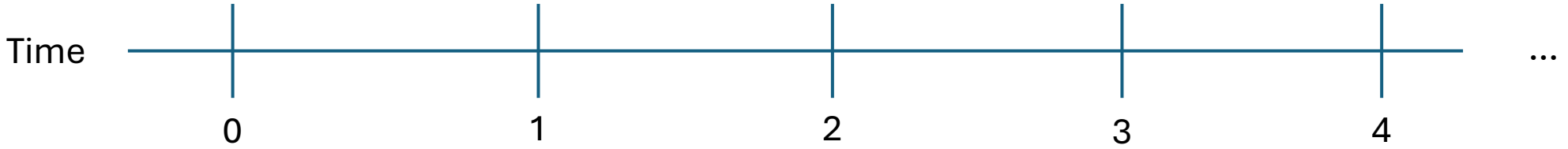


Disadvantages of FS-KEM

pk

sk_2

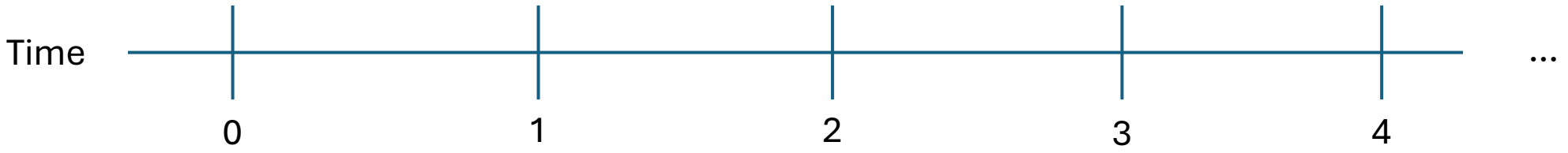
ct's to all future times insecure!



Interval KEM

pk

sk_2

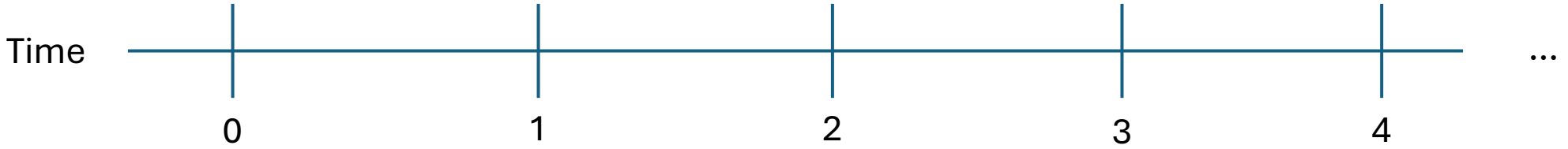


Interval KEM

pk

sk_2

$(sk_3, pk_3) \leftarrow Update(sk_2)$

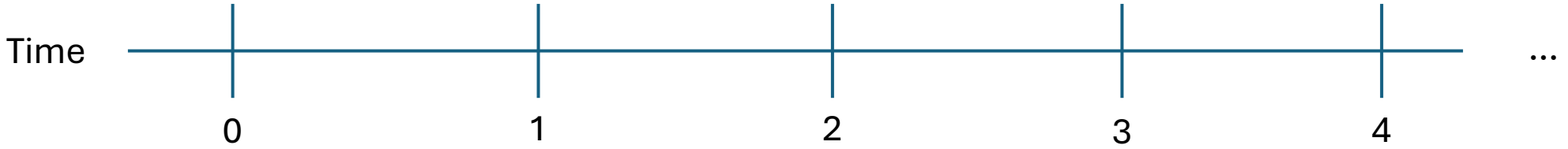


Interval KEM

pk_3

sk_2

sk_3



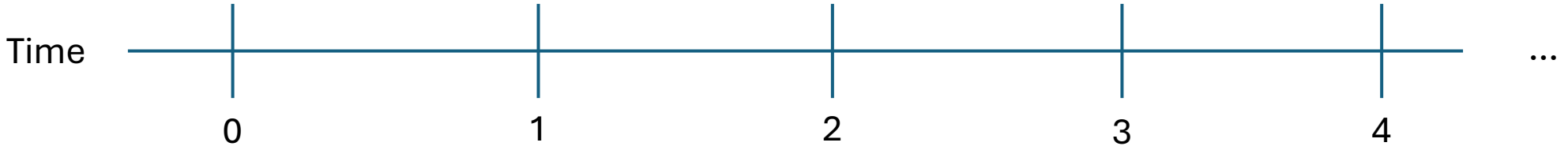
Interval KEM

pk_3

$(k_3, ct_3) \leftarrow_{\$} Enc(pk_3, 3)$

sk_2

sk_3



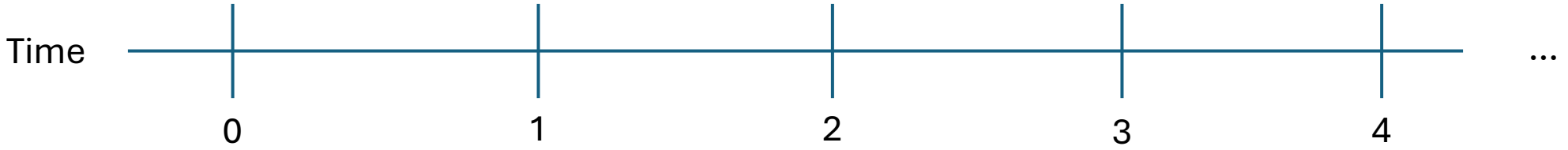
Interval KEM

pk_3

$(k_3, ct_3) \leftarrow_{\$} Enc(pk_3, 3)$

sk_2

sk_3



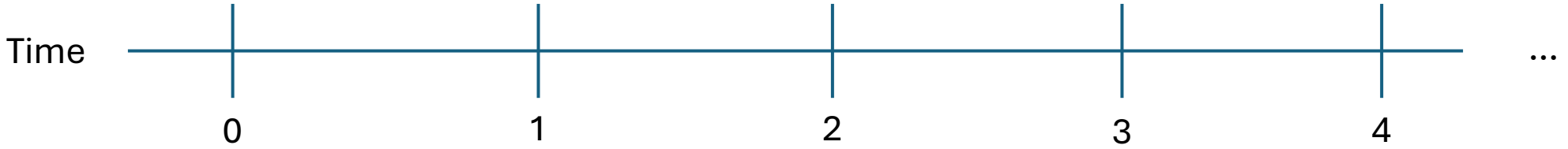
Interval KEM

pk_3

$(k_3, ct_3) \leftarrow_{\$} Enc(pk_3, 3)$

sk_2

sk_3 $k_3 \leftarrow Decrypt(sk_3, 3, ct_3)$



Interval KEM

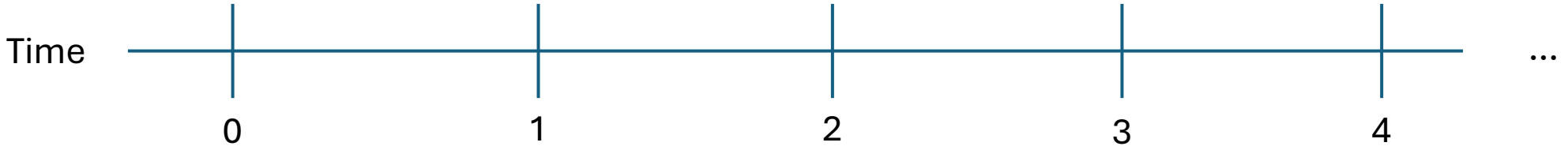
pk_3

$(k_3, ct_3) \leftarrow_{\$} Enc(pk_3, 3)$

Secure – PCS!

sk_2

sk_3 $k_3 \leftarrow Decrypt(sk_3, 3, ct_3)$



Interval KEM

pk_3

$(k_0, ct_0) \leftarrow_{\$} Enc(pk, 0)$

Secure – FS!

$(k_2, ct_2) \leftarrow_{\$} Enc(pk, 2)$

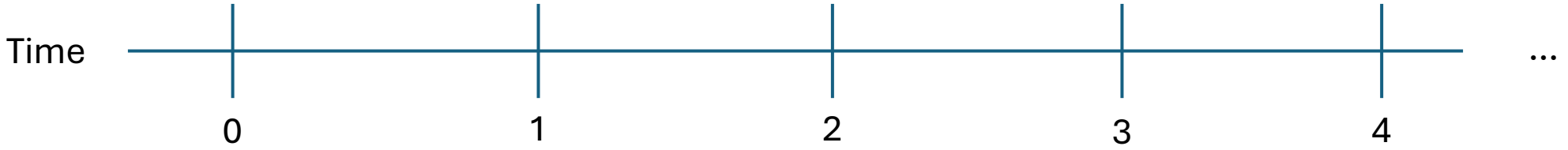
Insecure

$(k_3, ct_3) \leftarrow_{\$} Enc(pk_3, 3)$

Secure

sk_2

$sk_3 \quad k_3 \leftarrow Decrypt(sk_3, 3, ct_3)$



Comparison to Session-Based Messaging

- There are many session-based messaging protocols with FS & PCS
  - Double Ratchet Algorithm [PM16]
- Requires Key Management for each session!
  - Each update requires communication linear in # sessions
- With IKEM, just maintain one total secret key for every session
  - Updates are global!
  - Everyone encrypts to corresponding public key!

KEM-based IKEM

- Given $KM.gen(), KM.enc(), KM.dec()$

KEM-based IKEM

- Given $KM.gen(), KM.enc(), KM.dec()$

- $Gen(): KM.gen() \rightarrow (sk_0, pk_0)$
- $Up(sk_t): KM.gen \rightarrow (sk_{t+1}, pk_{t+1})$
- $Enc(pk_t): KM.enc(pk_t) \rightarrow_{\$} (k, ct)$
- $Dec(sk_t, ct): KM.dec(sk_t, ct) \rightarrow k$

KEM-based IKEM

- Given $KM.gen(), KM.enc(), KM.dec()$

- $Gen()$: $KM.gen() \rightarrow (sk_0, pk_0)$
- $Up(sk_t)$: $KM.gen \rightarrow (sk_{t+1}, pk_{t+1})$
- $Enc(pk_t)$: $KM.enc(pk_t) \rightarrow_{\$} (k, ct)$
- $Dec(sk_t, ct)$: $KM.dec(sk_t, ct) \rightarrow k$

- Each (sk_t, pk_t) indep ⇒ FS and PCS

Communication in Mesh Networks

pk_2

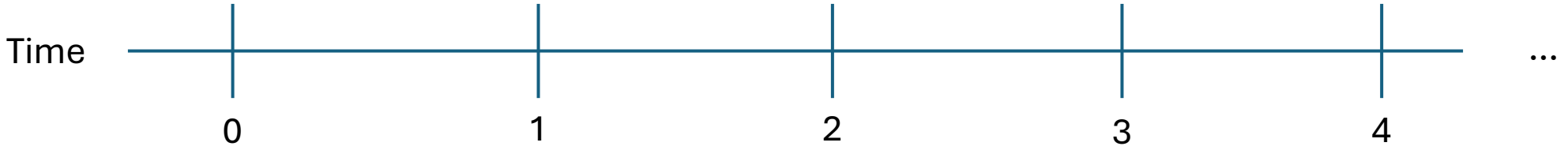
pk_2

pk_2

pk_2

pk_2

sk_2



Communication in Mesh Networks


pk_2

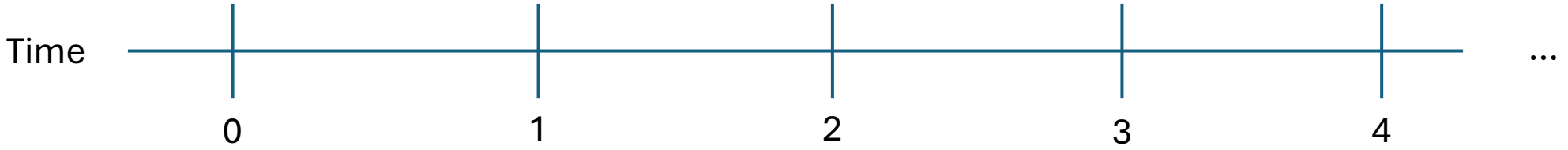

pk_2


pk_2

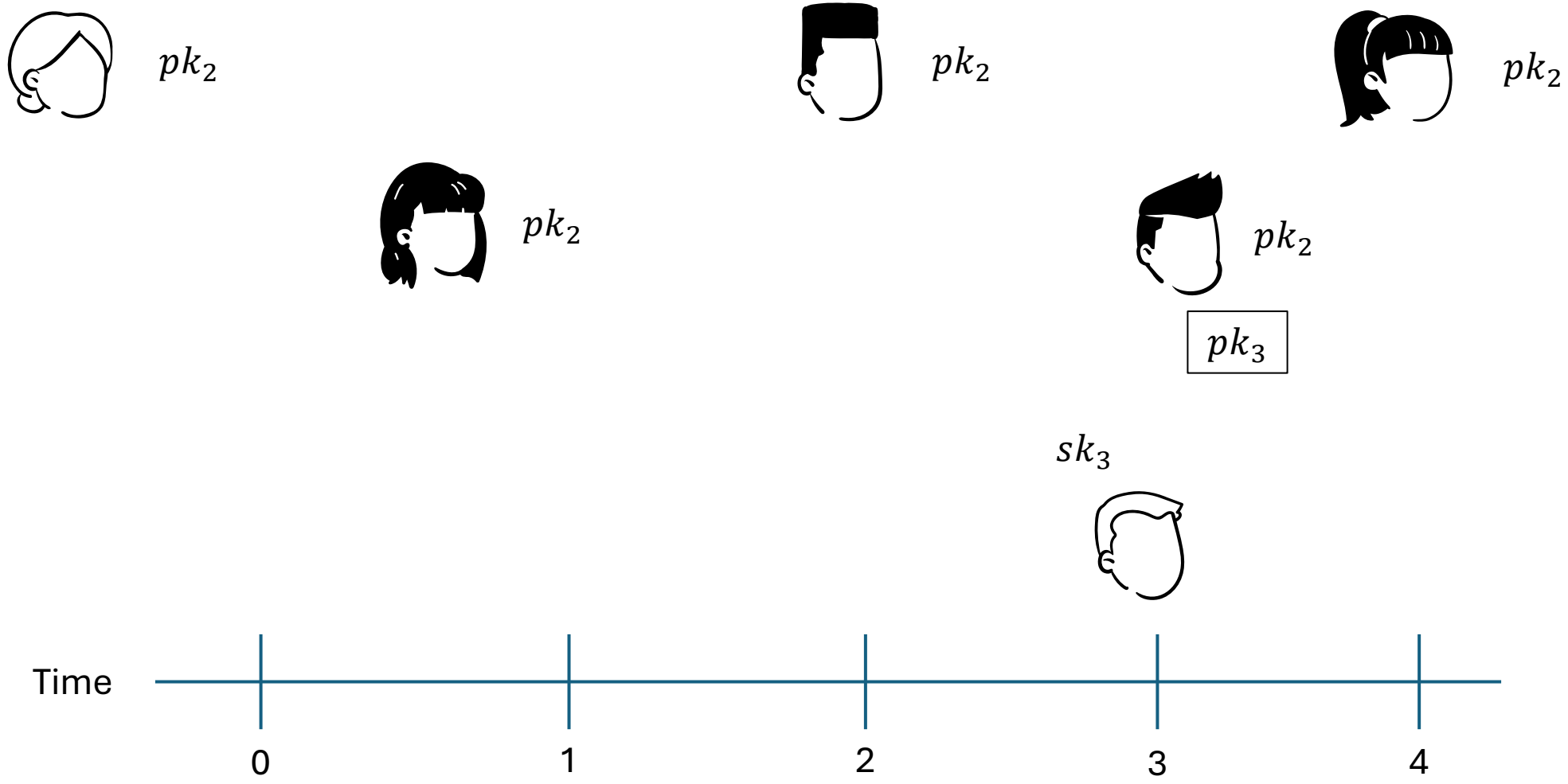

pk_2


pk_2

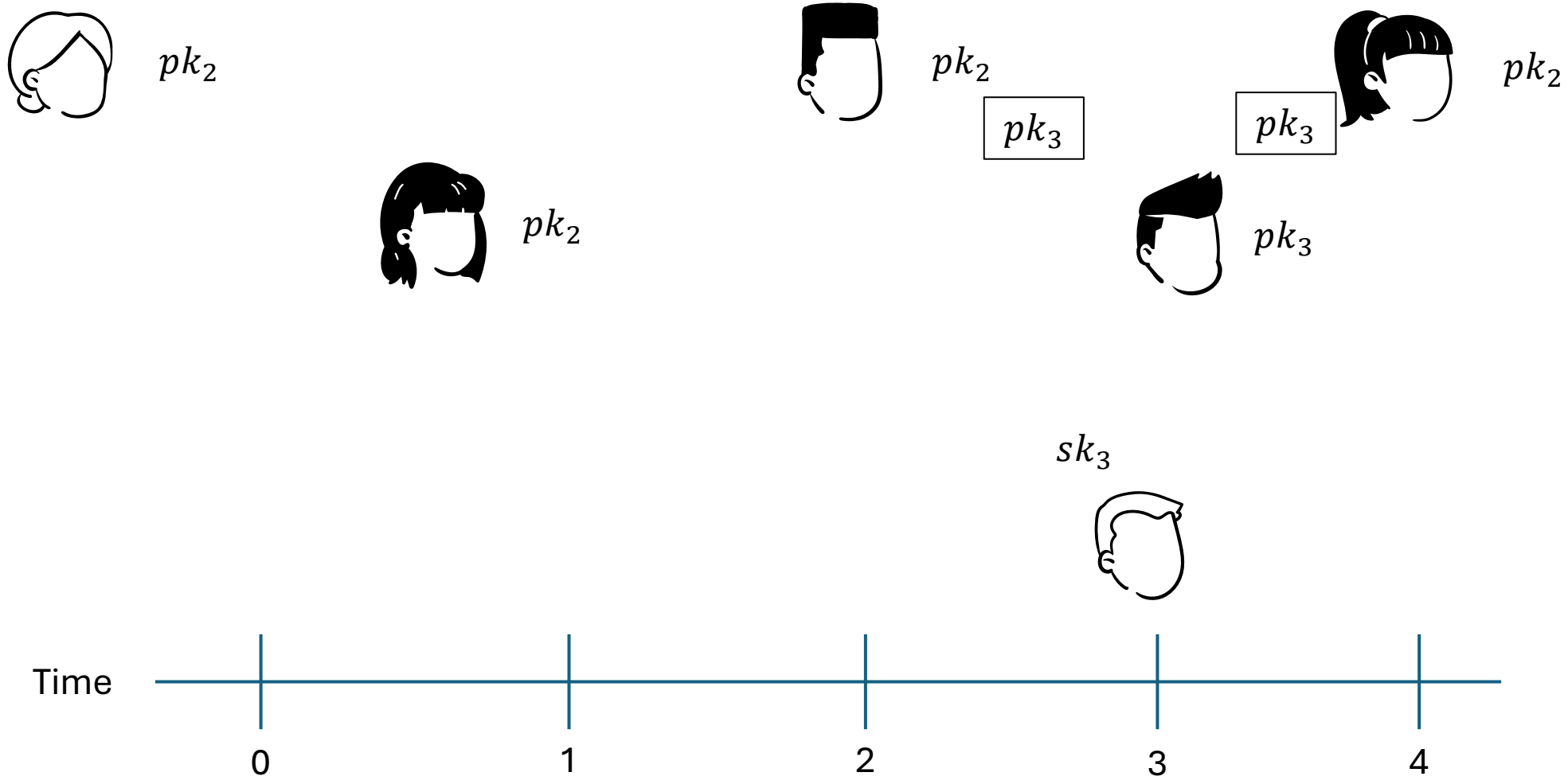
$(sk_3, pk_3) \leftarrow Up(sk_3)$



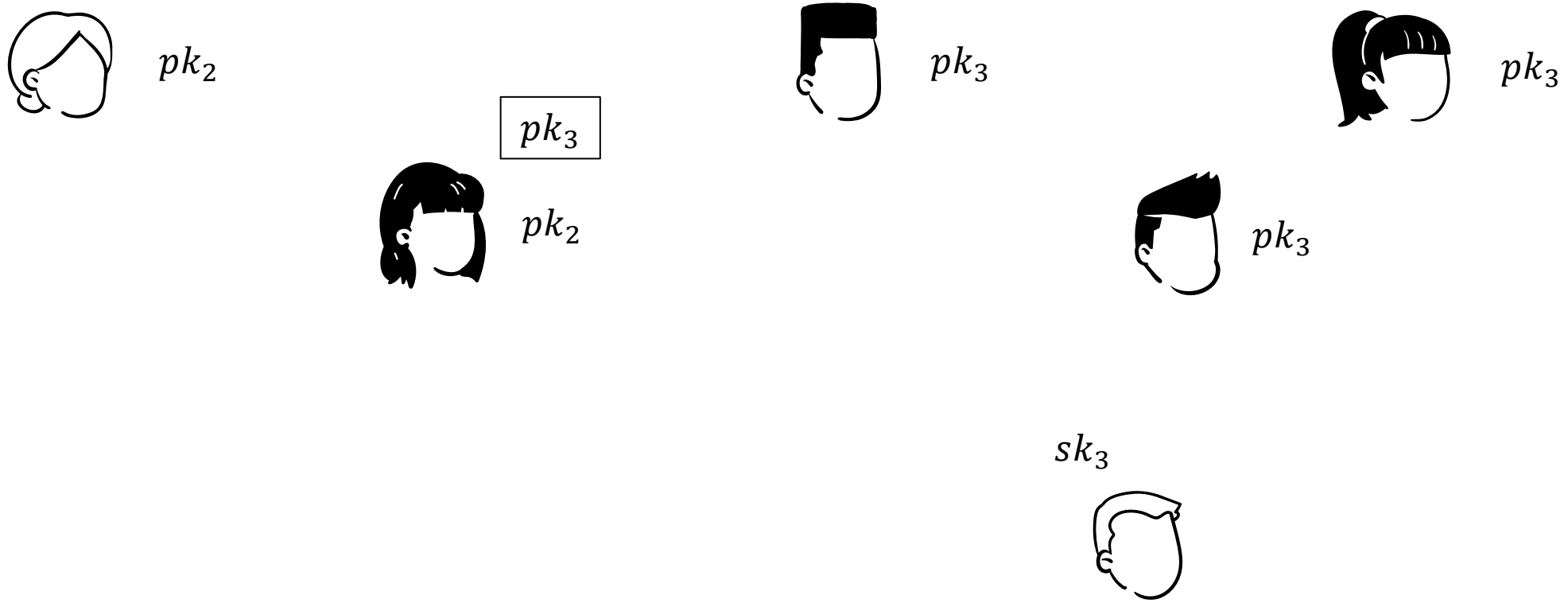
Communication in Mesh Networks



Communication in Mesh Networks



Communication in Mesh Networks



Communication in Mesh Networks

pk_2

$(k, ct) \leftarrow Enc(pk_2)$

pk_3

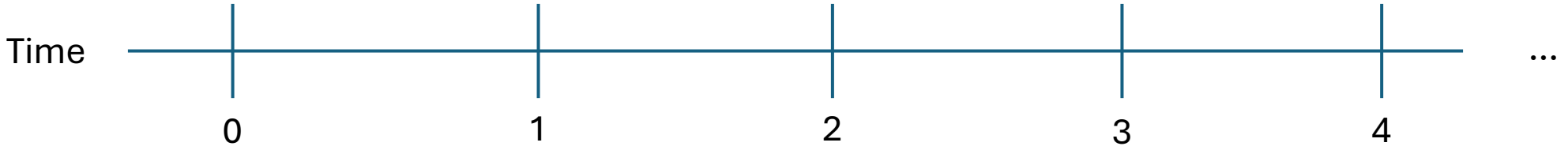
pk_3

pk_3

pk_2

pk_3

sk_3



Communication in Mesh Networks

pk_2

$(k, ct) \leftarrow Enc(pk_2)$

pk_3

pk_2

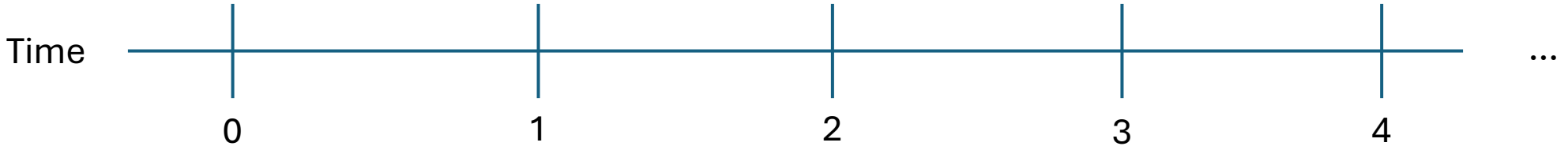
pk_3

pk_3

pk_3

sk_3

Can’t decrypt!!



Communication in Mesh Networks

pk_2

$(k, ct) \leftarrow Enc(pk_2)$

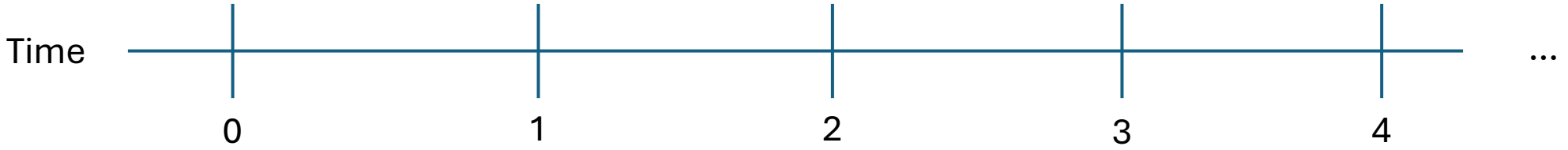
pk_3

pk_3

pk_2

pk_3

(sk_2, sk_3)



Communication in Mesh Networks

pk_2

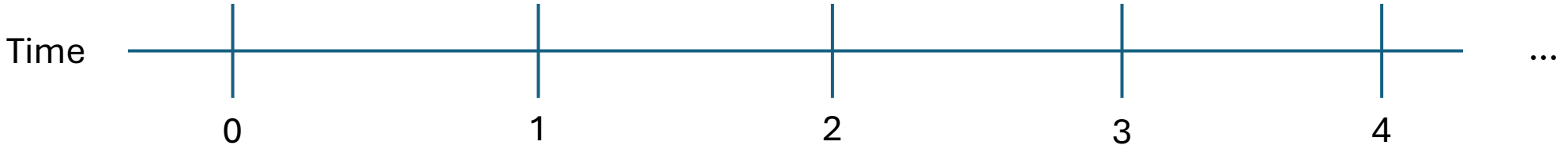
pk_3

pk_3

pk_2

pk_3

(sk_2, sk_3, sk_4)



Communication in Mesh Networks

pk_4

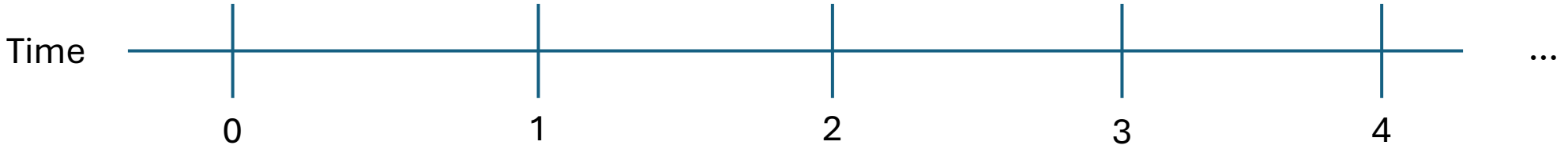
pk_4

pk_4

pk_4

pk_4

(sk_2, sk_3, sk_4)



Communication in Mesh Networks


pk_4

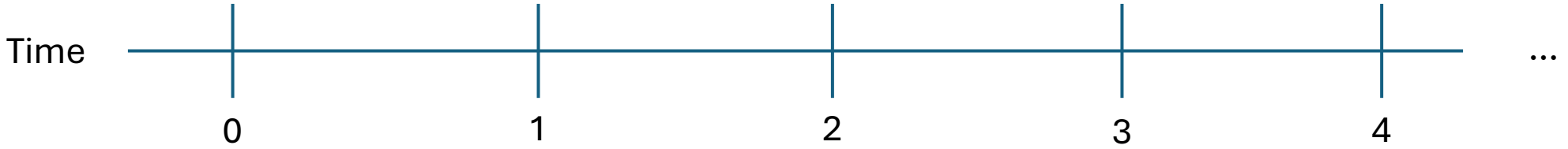

pk_4


pk_4


pk_4


pk_4

$Del((sk_2, sk_3, sk_4), (2,3))$



Communication in Mesh Networks

pk_4

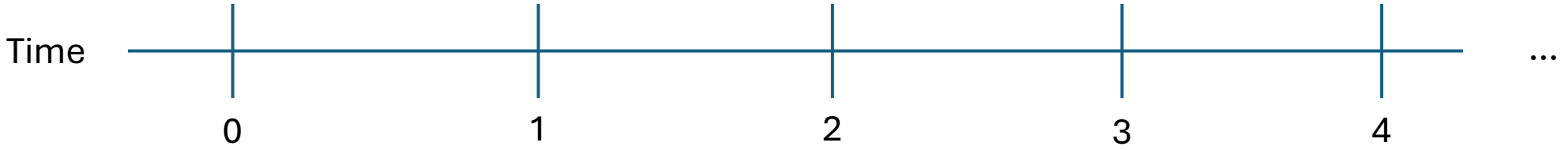
pk_4

pk_4

pk_4

pk_4

(sk_4)



KEM-based IKEM

- Given $KM.gen(), KM.enc(), KM.dec()$
- $Gen()$: $KM.gen() \rightarrow (sk_0, pk_0)$
- $Up\big((sk_{t_0}, \dots, sk_{t_1})\big)$:
  - $KM.gen \rightarrow (sk_{t_1+1}, pk_{t_1+1})$
  - Output $\big((sk_{t_0}, \dots, sk_t, sk_{t_1+1}), pk_{t_1+1}\big)$
- $Enc(pk_t)$: $KM.enc(pk_t) \rightarrow_{\$} (k, (t, ct))$
- $Dec\big((sk_{t_0}, \dots, sk_{t_1}), (t, ct)\big)$: $KM.dec(sk_t, ct) \rightarrow k$
  - Only works if $t \in [t_0, t_1]$

KEM-based IKEM

- Given $KM.gen(), KM.enc(), KM.dec()$
- $Gen()$: $KM.gen() \rightarrow (sk_0, pk_0)$
- $Up\big((sk_{t_0}, \dots, sk_{t_1})\big)$:
  - $KM.gen \rightarrow (sk_{t_1+1}, pk_{t_1+1})$
  - Output $\big((sk_{t_0}, \dots, sk_t, sk_{t_1+1}), pk_{t_1+1}\big)$
- $Enc(pk_t)$: $KM.enc(pk_t) \rightarrow_{\$} (k, (t, ct))$
- $Dec\big((sk_{t_0}, \dots, sk_{t_1}), (t, ct)\big)$: $KM.dec(sk_t, ct) \rightarrow k$
  - Only works if $t \in [t_0, t_1]$
- $Del\left(\left(sk_{t_0}, \dots, sk_{t_1}\right), \ell\right)$: Output $\left(sk_{t_0+\ell}, \dots, sk_{t_1}\right)$

Secret Key Size Lower Bound

- We want to decapsulate ct created at any time $t \in [t_0, t_1]$
- Secret state in $O(t_1 - t_0)$ inevitable?

  → Yes

- Intuition:
  - Don't know when compromises happen
    ⇒ Each $Up()$ needs independent randomness
  - Cannot compress independent randomness

Secret Key Size Lower Bound

- We want to decapsulate ct created at any time $t \in [t_0, t_1]$
- Secret state in $O(t_1 - t_0)$ inevitable?

  → Yes

- Intuition:
  - Don't know when compromises happen
    ⇒ Each $Up()$ needs independent randomness
  - Cannot compress independent randomness

- Circumvent lower bound via external (public) storage

Stronger Security for Mesh Networks [BRT23]

pk_2

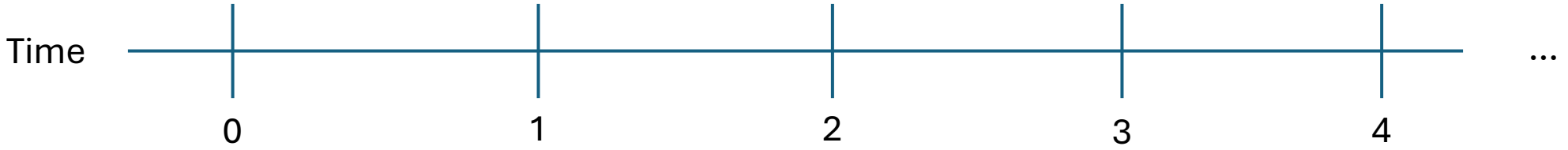
pk_2

pk_2

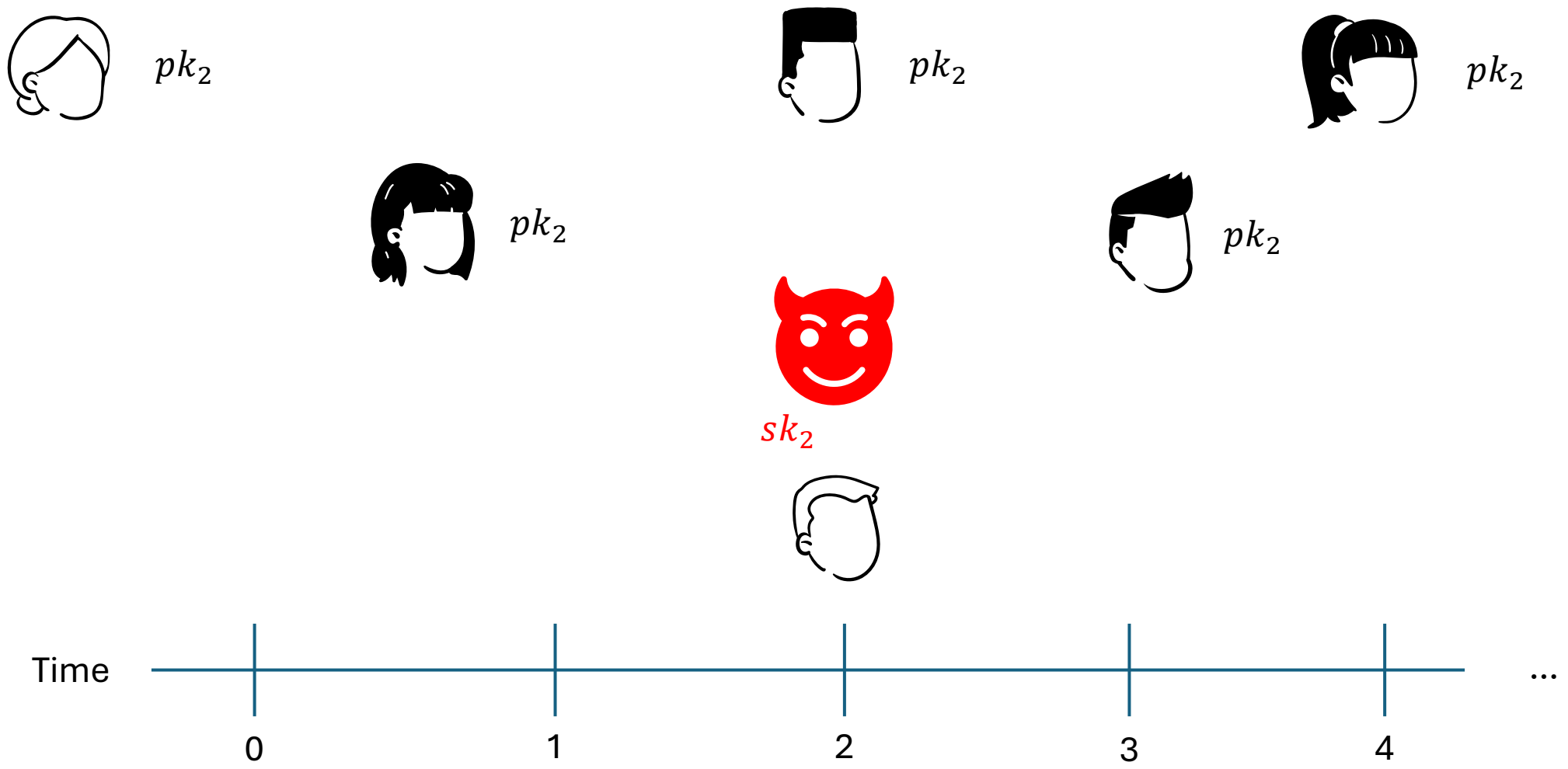
pk_2

pk_2

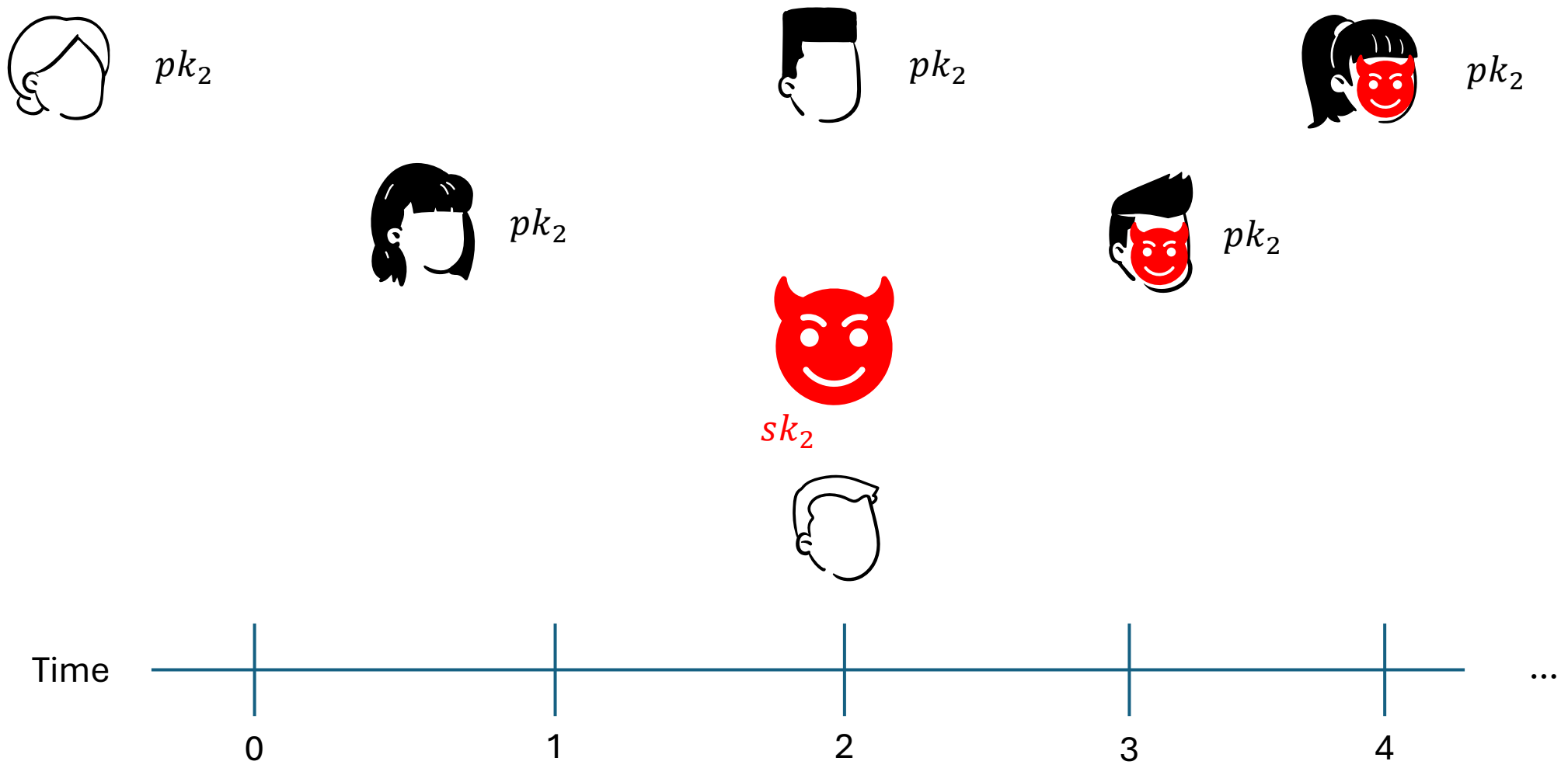
sk_2



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]

pk_2

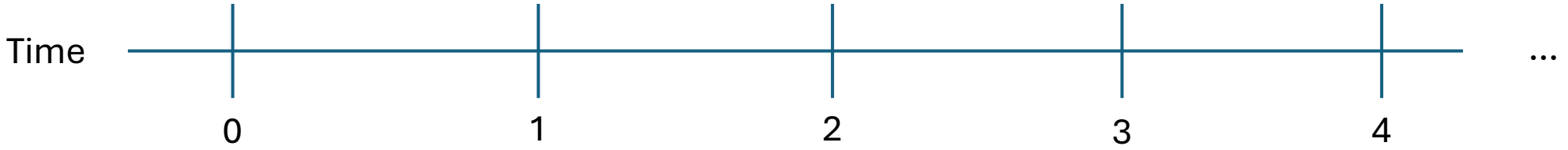
pk_2

pk_2

pk_2

pk_2

$((sk_2, sk_3), pk_3) \leftarrow Up(sk_2)$



Stronger Security for Mesh Networks [BRT23]

pk_2

pk_2

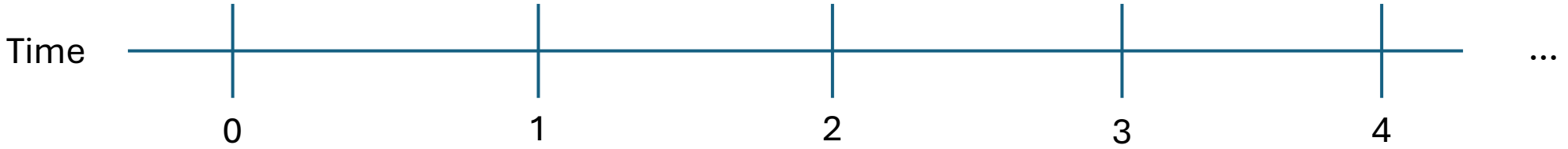
pk_2

pk_2

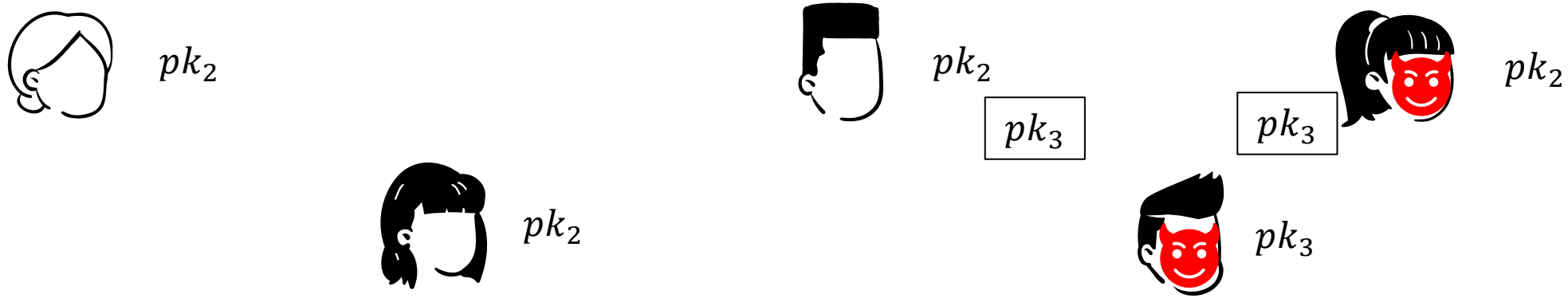
pk_2

pk_3

(sk_2, sk_3)



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]

pk_2

$(k, ct) \leftarrow Enc(pk_2)$

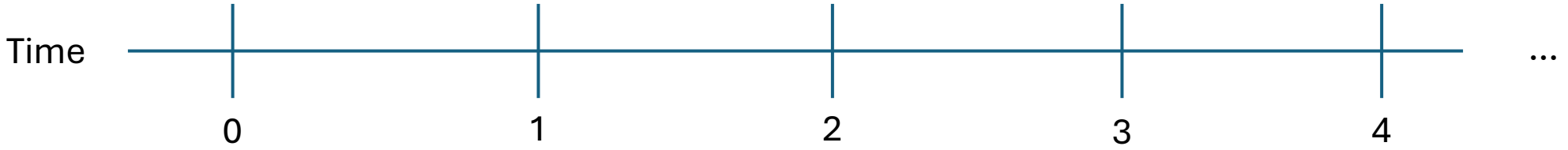
pk_3

pk_3

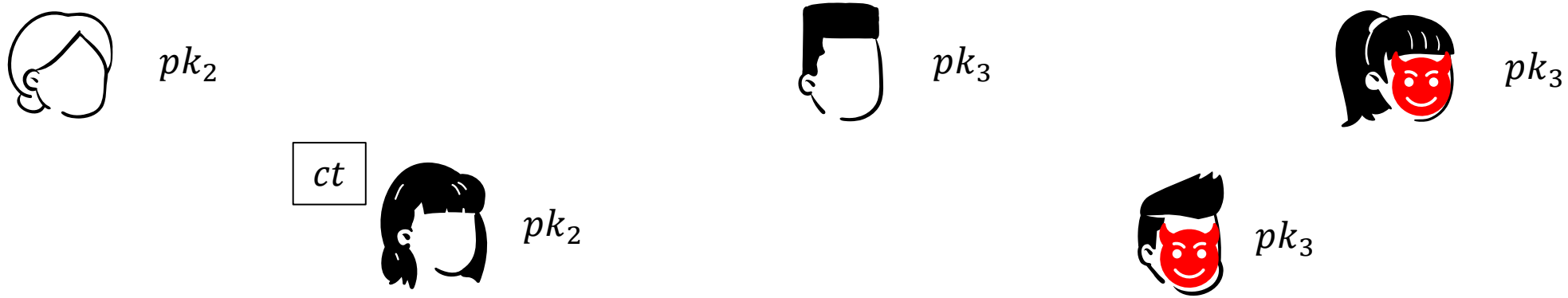
pk_2

pk_3

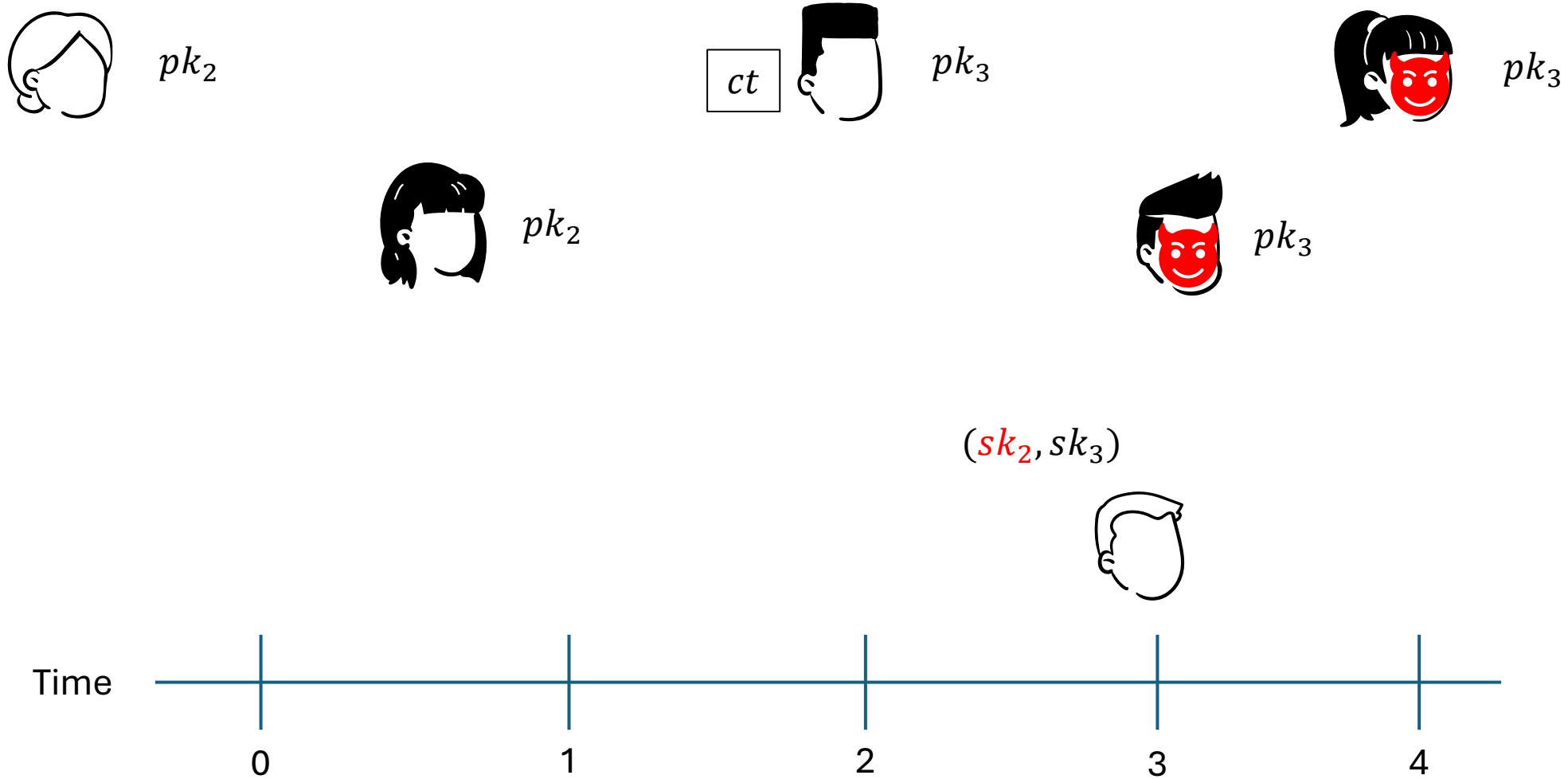
(sk_2, sk_3)



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]

pk_2

pk_3

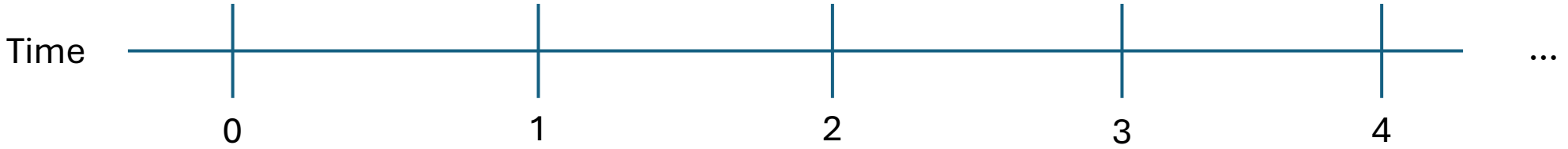
pk_3

$ct' \leftarrow Reenc(pk_3, ct)$

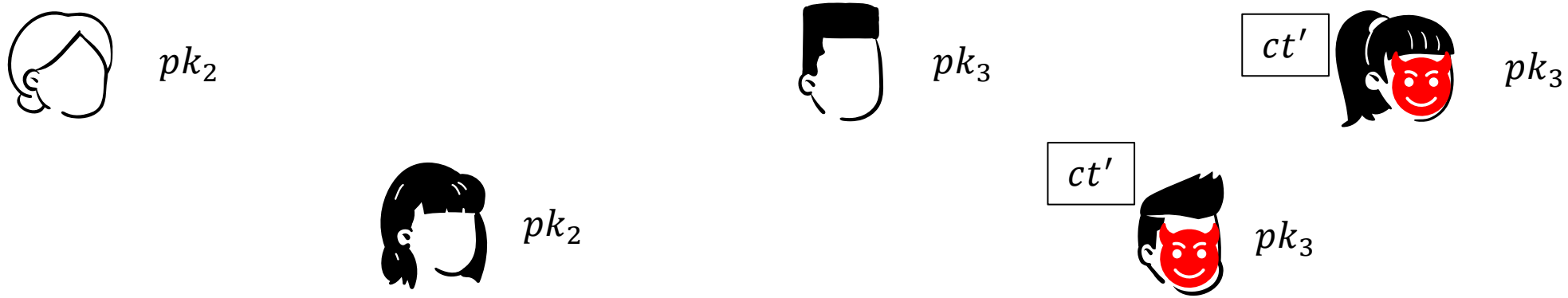
pk_2

pk_3

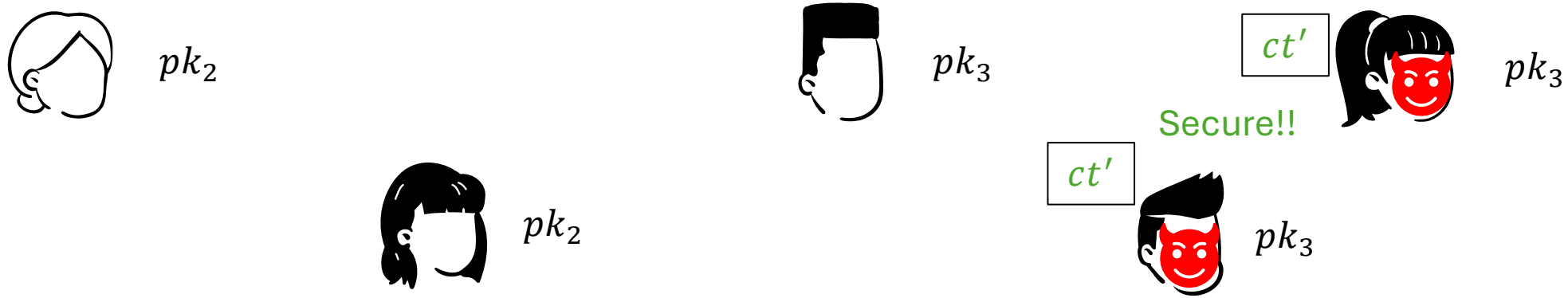
(sk_2, sk_3)



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]



Stronger Security for Mesh Networks [BRT23]

pk_2

pk_3

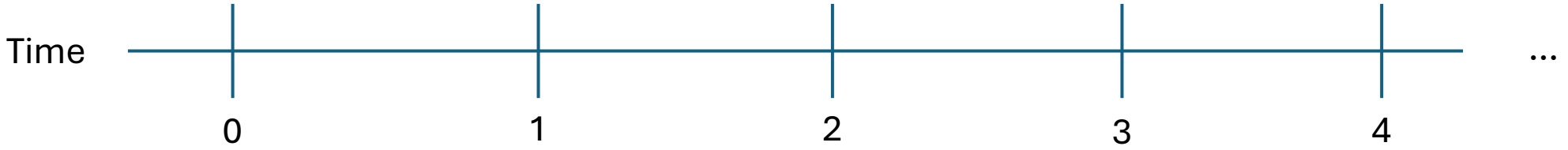
pk_3

Secure!!

pk_2

pk_3

(sk_2, sk_3) ct'



Stronger Security for Mesh Networks [BRT23]

pk_2

pk_3

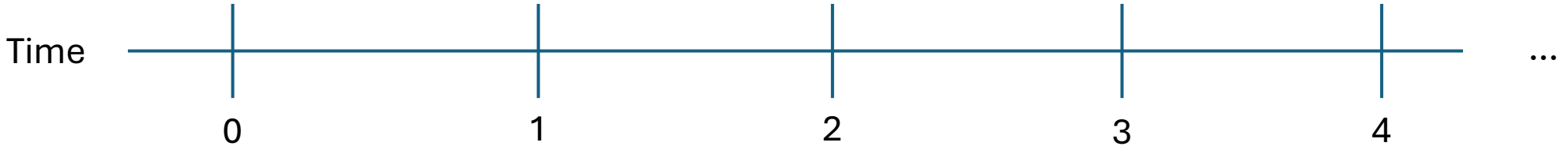
pk_3

Secure!!

pk_2

pk_3

(sk_2, sk_3) $k \leftarrow Dec((sk_2, sk_3), ct)$



Stronger Security for Mesh Networks [BRT23]

pk_2

pk_3

pk_3

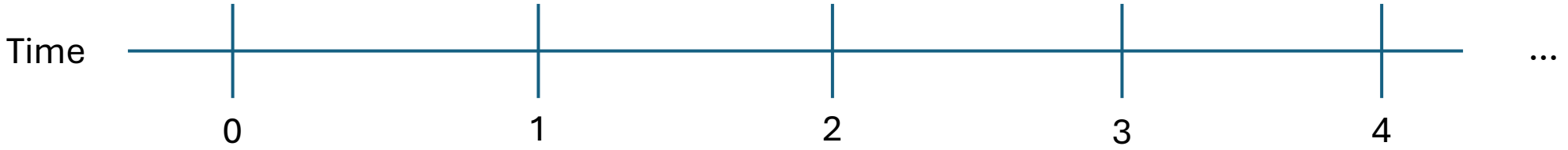
Secure!!

pk_2

pk_3

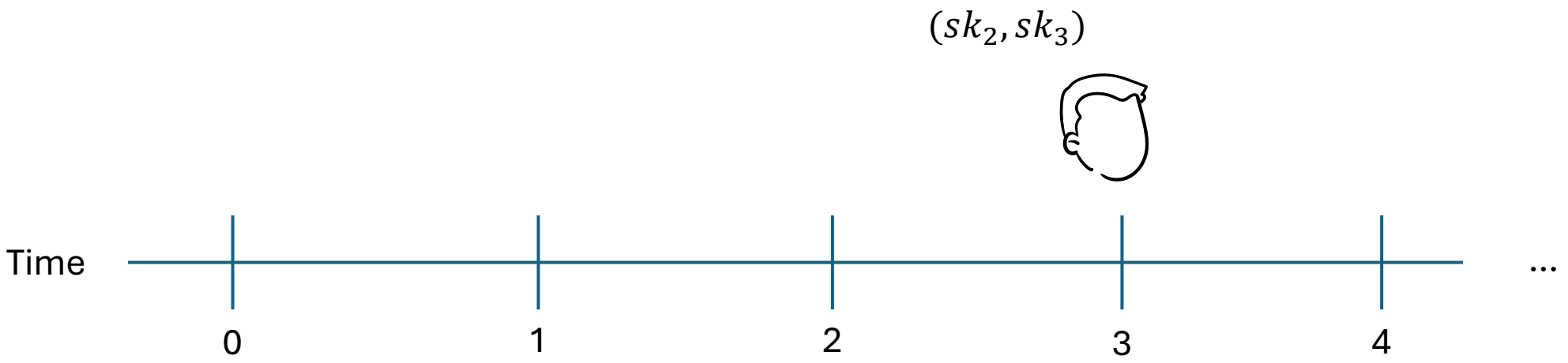
We call this extended notion IKEM*R*

(sk_2, sk_3) $k \leftarrow Dec((sk_2, sk_3), ct)$



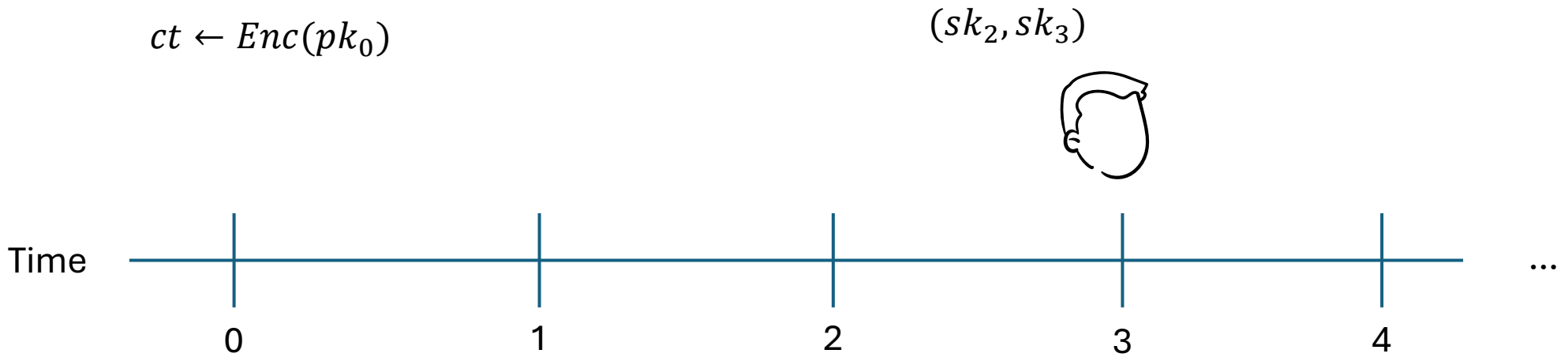
Security & Correctness for IKEMR

- Security & correctness wrt. time of **first** encapsulation



Security & Correctness for IKEMR

- Security & correctness wrt. time of **first** encapsulation



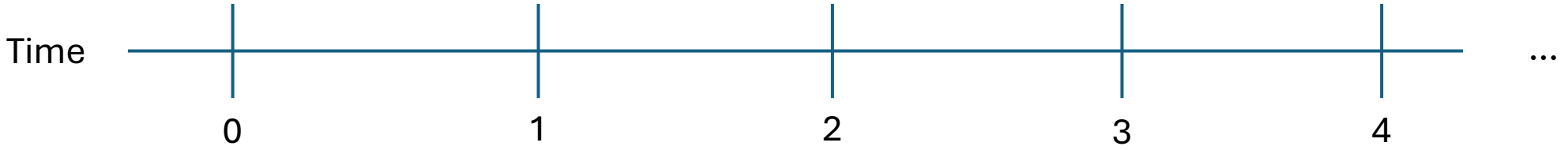
Security & Correctness for IKEMR

- Security & correctness wrt. time of **first** encapsulation

Cannot be re-encapsulated for Bob!

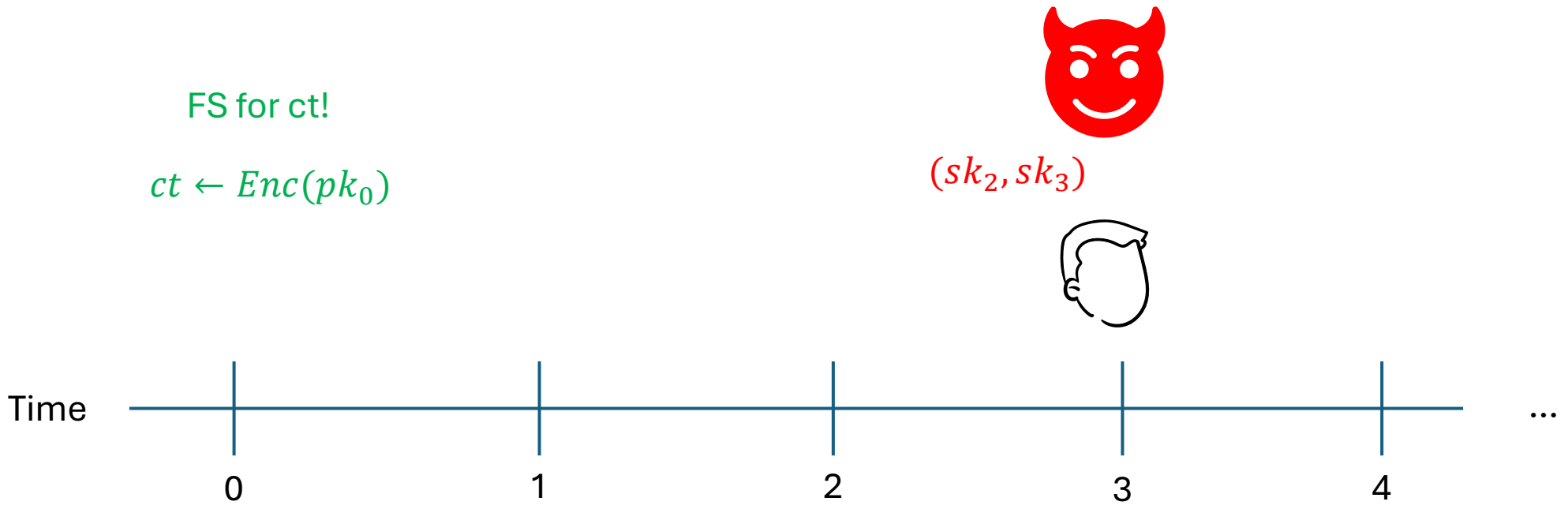
$ct \leftarrow Enc(pk_0)$

(sk_2, sk_3)



Security & Correctness for IKEMR

- Security & correctness wrt. time of **first** encapsulation



Comparison to Proxy Re-Encryption (PRE)

- PRE ciphertexts encrypted under pk_A can be re-encrypted to pk_B
- Key Derivation
  - PRE: *Re-encryption* key $rk_{A\to B}$ from sk_A, pk_B
  - IKEMR: (sk_B, pk_B) from sk_A
- Path
  - PRE: Cycles of re-encryption
  - IKEMR: Only to future
- Assumptions
  - Unidirectional PRE with $O(1)$ size ct's from $\geq$ FHE
  - IKEMR with $O(1)$ size ct's from TDPs! (see next)
- PRE seemingly useless for IKEMR
  - Continuous re-enc to future $\Rightarrow$ no FS!

Simple Extension of Basic IKEM Fails!

$(k, ct) \leftarrow KM.Enc(pk_2)$

pk_2

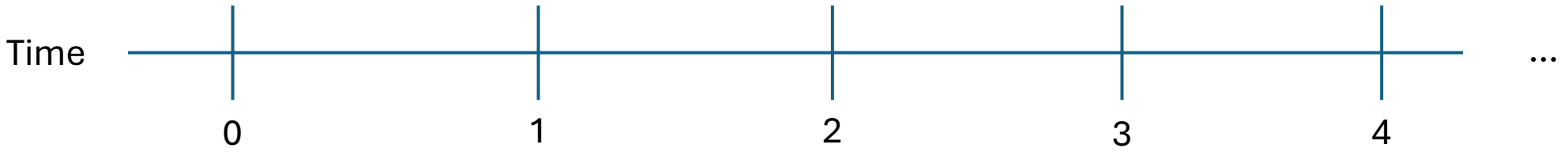
pk_2

pk_2

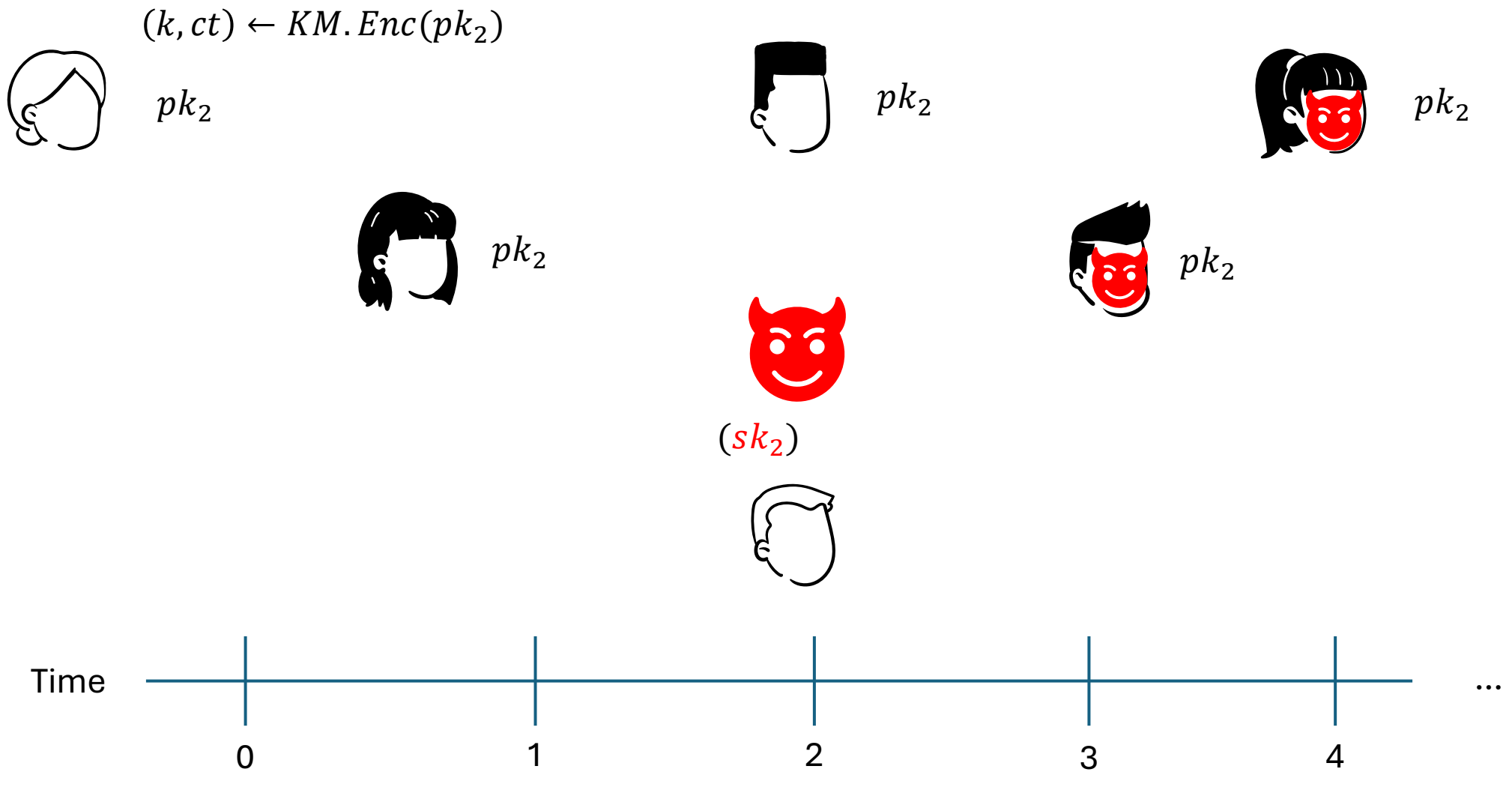
pk_2

pk_2

(sk_2)



Simple Extension of Basic IKEM Fails!



Simple Extension of Basic IKEM Fails!

pk_2

Still secure!
(Not yet seen by $\mathcal{A}$)

ct

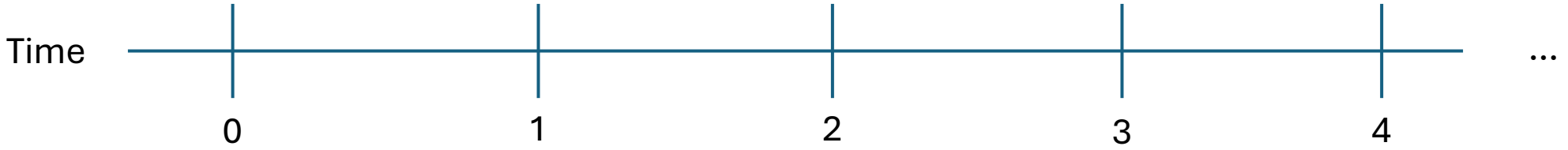
pk_3

pk_3

pk_3

pk_3

(sk_2, sk_3)



Simple Extension of Basic IKEM Fails!

pk_2

pk_3

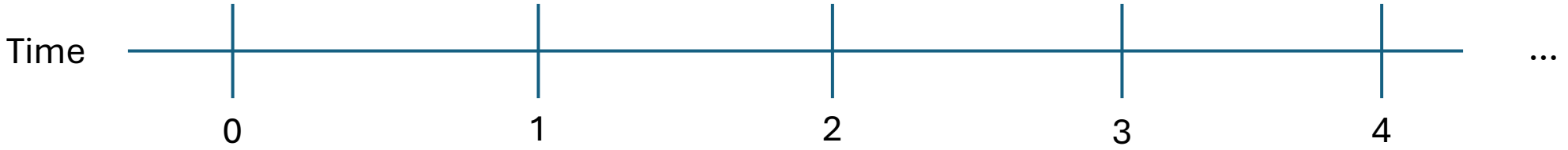
pk_3

pk_3

Compute $(k', ct_1') \leftarrow KM.Enc(pk_3)$
and $ct_2' \leftarrow k' \oplus ct$

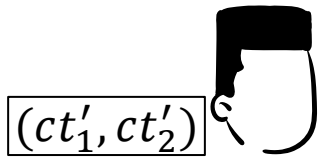
pk_3

(sk_2, sk_3)



Simple Extension of Basic IKEM Fails!

pk_2



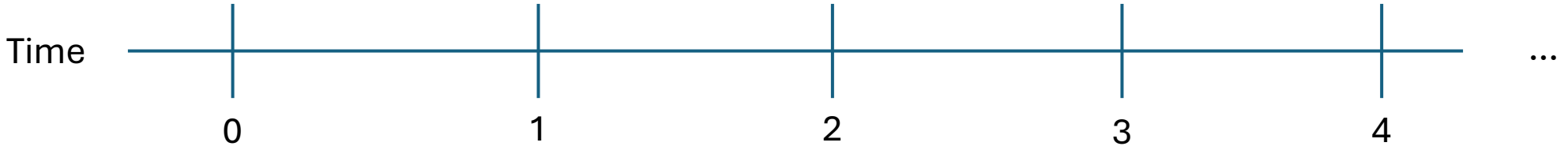
pk_3

pk_3

pk_3

pk_3

(sk_2, sk_3)



Simple Extension of Basic IKEM Fails!

pk_2

pk_3

pk_3

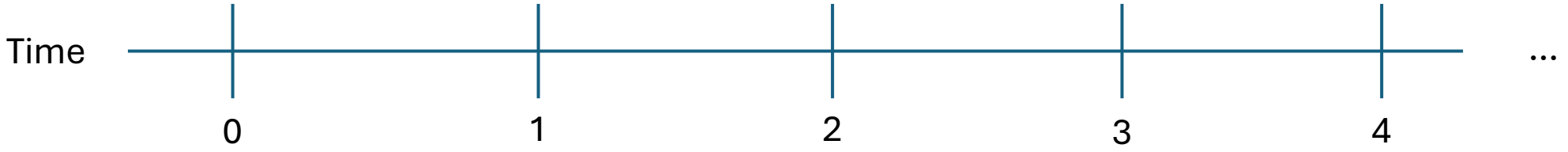
pk_3

Still secure!
(re-enc using pk_3)

$\boxed{(ct_1', ct_2')}$

pk_3

(sk_2, sk_3)



Simple Extension of Basic IKEM Fails!

pk_2

pk_3

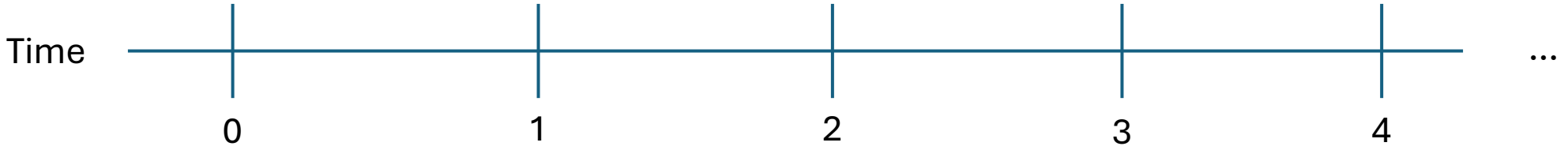
pk_3

pk_3

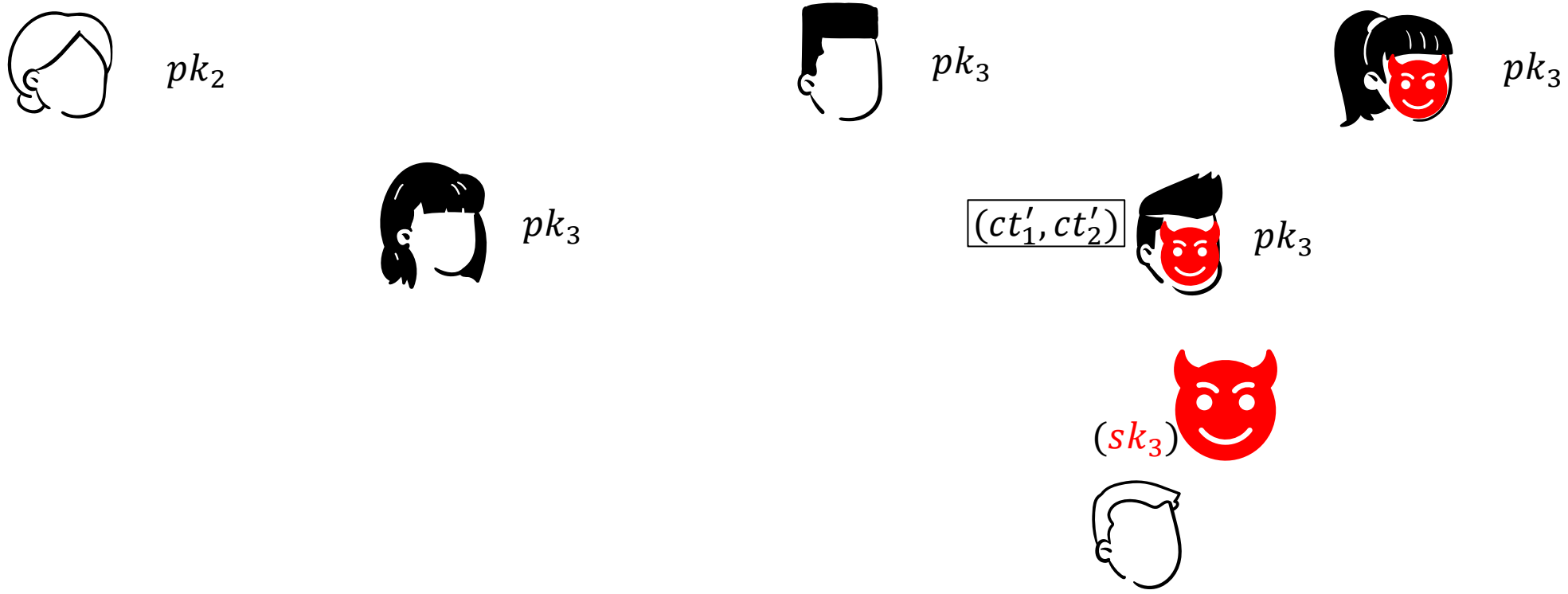
$\boxed{(ct_1', ct_2')}$

pk_3

(sk_3)



Simple Extension of Basic IKEM Fails!



Simple Extension of Basic IKEM Fails!

pk_2

pk_3

pk_3

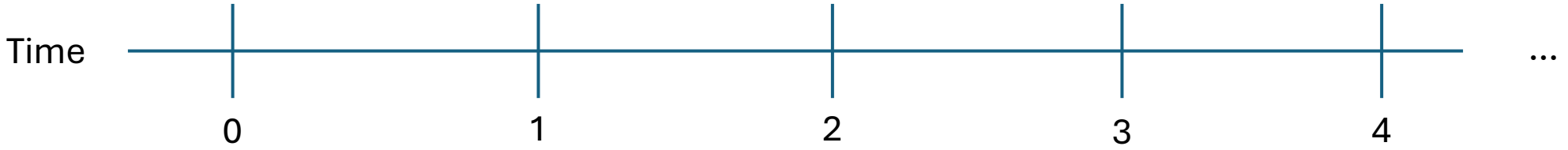
pk_3

$\boxed{(ct_1', ct_2')}$

pk_3

Want this to still be secure!
(Interval shortened to exclude $t = 2$)

(sk_3)



Simple Extension of Basic IKEM Fails!

pk_2

pk_3

pk_3

pk_3

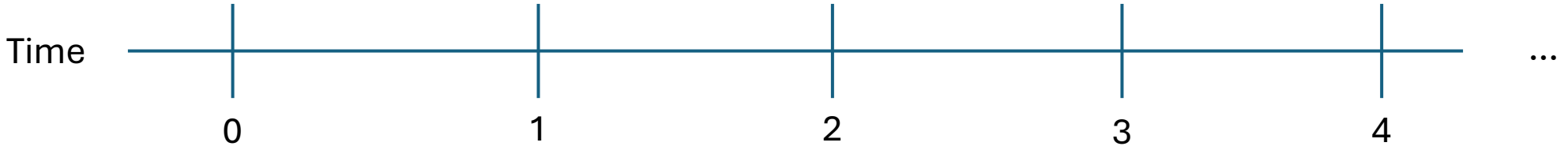
$\boxed{(ct_1', ct_2')}$

pk_3

Want this to still be secure!
(Interval shortened to exclude $t = 2$)

(sk_3)

But both sk_2 and sk_3 leaked! => insecure!



Simple Extension of Basic IKEM Fails!

pk_2

pk_3

pk_3

pk_3

$\boxed{(ct_1', ct_2')}$

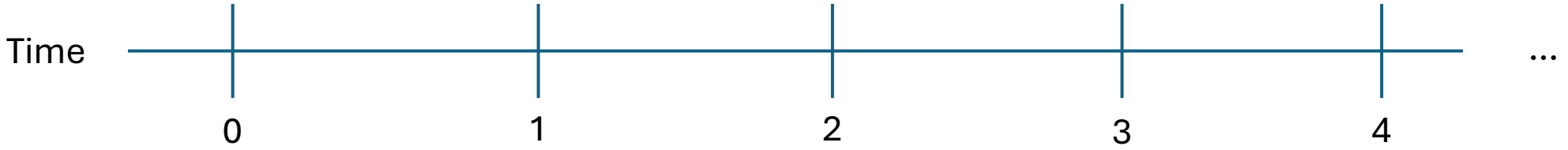
pk_3

Want this to still be secure!
(Interval shortened to exclude $t = 2$)

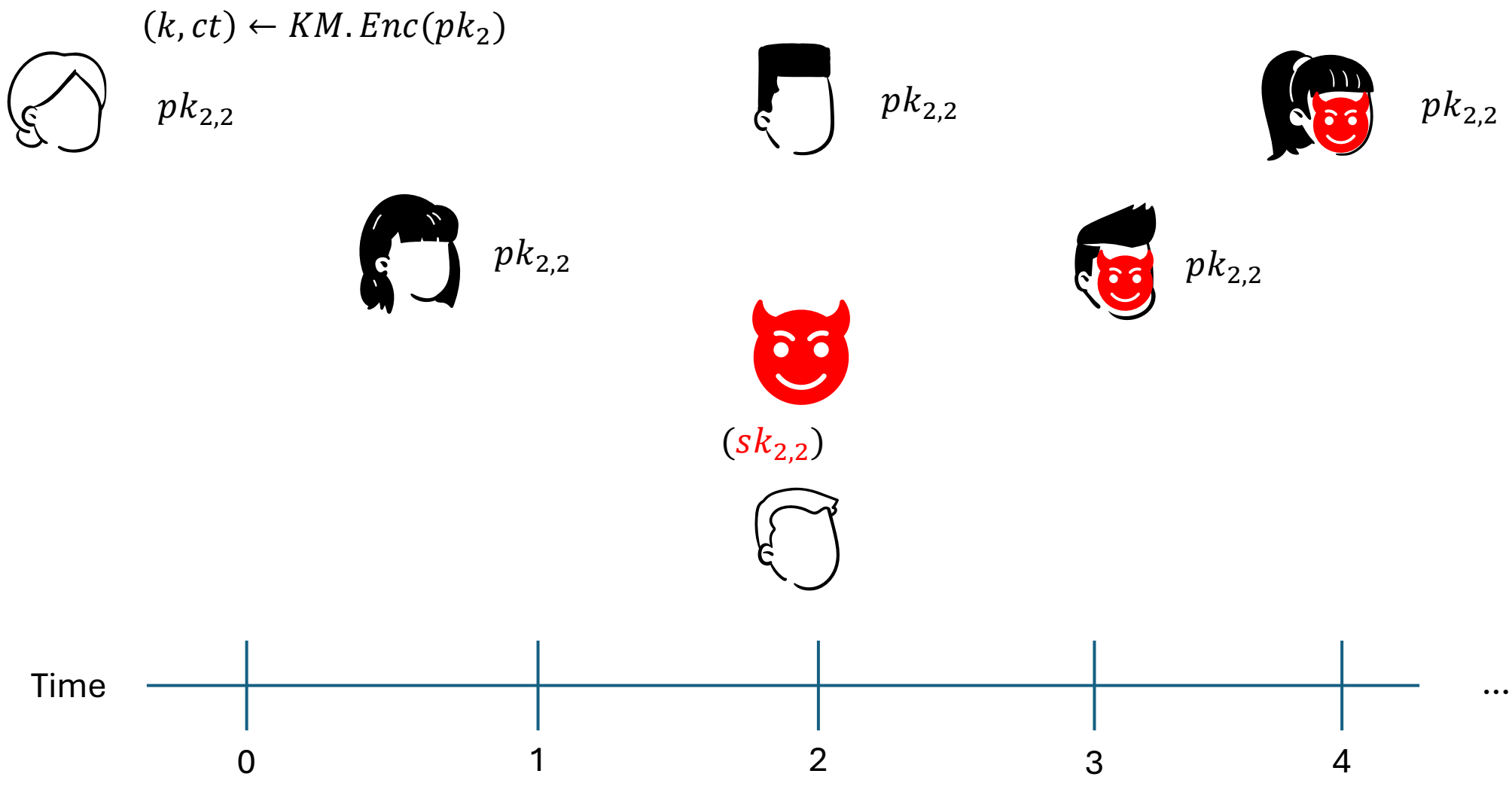
(sk_3)

But both sk_2 and sk_3 leaked! => insecure!

Note: can't just delete sk_3!



The Fix


$(k, ct) \leftarrow KM.Enc(pk_2)$
$pk_{2,2}$
$pk_{2,2}$
$pk_{2,2}$
$pk_{2,2}$
$pk_{2,2}$
$(sk_{2,2})$
Time
0
1
2
3
4
...

The Fix

$pk_{2,2}$

ct

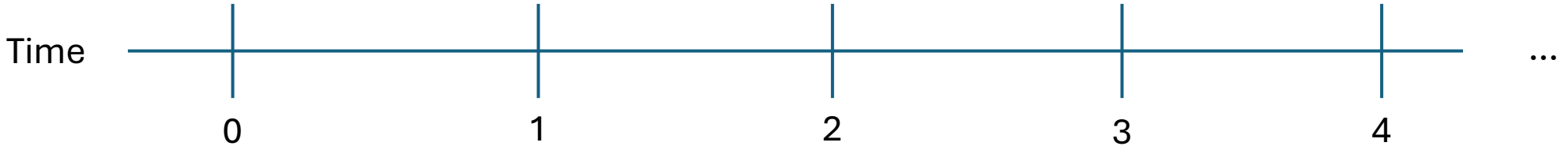
$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

$(sk_{2,2}, sk_{3,2})$
$(sk_{3,3})$



The Fix

$pk_{2,2}$

$(pk_{3,2}, pk_{3,3})$

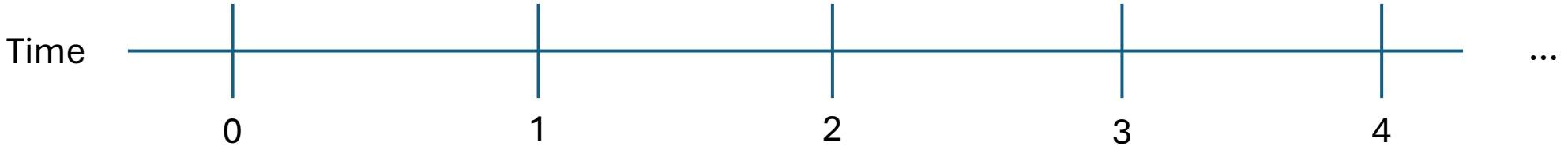
$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

Compute $(k', ct_1') \leftarrow KM.Enc(pk_{3,2})$
and $ct_2' \leftarrow k' \oplus ct$

$(pk_{3,2}, pk_{3,3})$

$(sk_{2,2}, sk_{3,2})$
$(sk_{3,3})$



The Fix

$pk_{2,2}$

$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

Still secure!
(re-enc using $pk_{3,2}$)

$\boxed{(ct'_1, ct'_2)}$

$(pk_{3,2}, pk_{3,3})$

$(sk_{2,2}, sk_{3,2})$
$(sk_{3,3})$

Time

0 1 2 3 4 ...

The Fix

$pk_{2,2}$

$(pk_{3,2}, pk_{3,3})$

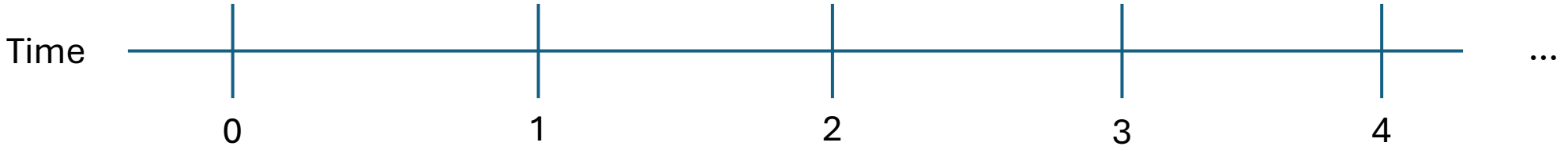
$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

$\boxed{(ct'_1, ct'_2)}$

$(pk_{3,2}, pk_{3,3})$

$(sk_{3,3})$



The Fix

$pk_{2,2}$

$(pk_{3,2}, pk_{3,3})$

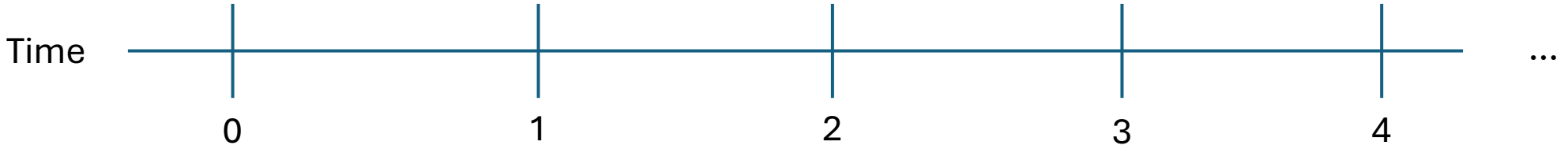
$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

$\boxed{(ct'_1, ct'_2)}$

$(pk_{3,2}, pk_{3,3})$

$(sk_{3,3})$



The Fix

$pk_{2,2}$

$(pk_{3,2}, pk_{3,3})$

$(pk_{3,2}, pk_{3,3})$

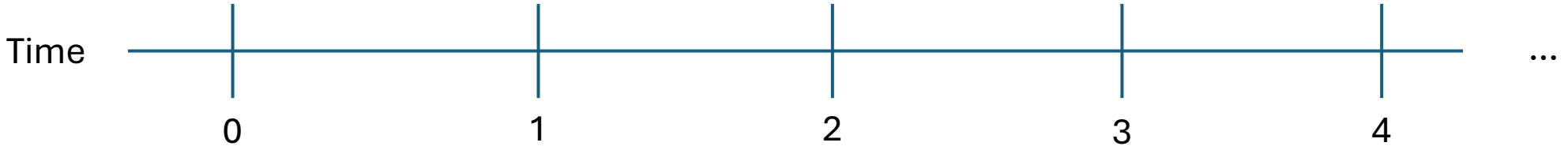
$(pk_{3,2}, pk_{3,3})$

$\boxed{(ct'_1, ct'_2)}$

$(pk_{3,2}, pk_{3,3})$

Still secure! $sk_{3,2}$ gone!

$(sk_{3,3})$



The Fix – Efficiency

- Secret state in $O(\ell)$ = "decryption interval"
  - PRG chain: start at seed $s_{t,t-\ell}$ and derive $(sk_{t,t-\ell}, \ldots, sk_{t,t})$ iteratively
  - To remove time $t - \ell$: eval PRG$(s_{t,t-\ell}) \rightarrow (s_{t,t-\ell+1}, sk_{t,t-\ell+t})$

- Ciphertext grows with # of re-encs
- Public key at each time t in $O(\ell)$

IKEMR with Small Ciphertexts

- Trapdoor Permutation (TDP) instead of KEM
- (Cryptographic) ct stays the same size after each re-enc

- Decryptor: which keys to use to decrypt?
  - ct only grows by + $\log \lambda$ (time label) with each re-enc
  - (Instead of + λ factor)

→ ct always $O(1)$; pk_t grows to $O(\ell^2)$

Additional Properties

- Alternatively make each pk_t $O(1)$ size
  - Uses FS-PKE
  - Cts grow additive λ factor with each re-enc
  - Sk size now $\ell \log \ell$

- Replayable CCA Security! (Like PRE; hard to define CCA-Security)
  - Some subtleties for definition; constructions use standard techniques

Thanks!

https://ia.cr/2024/1454