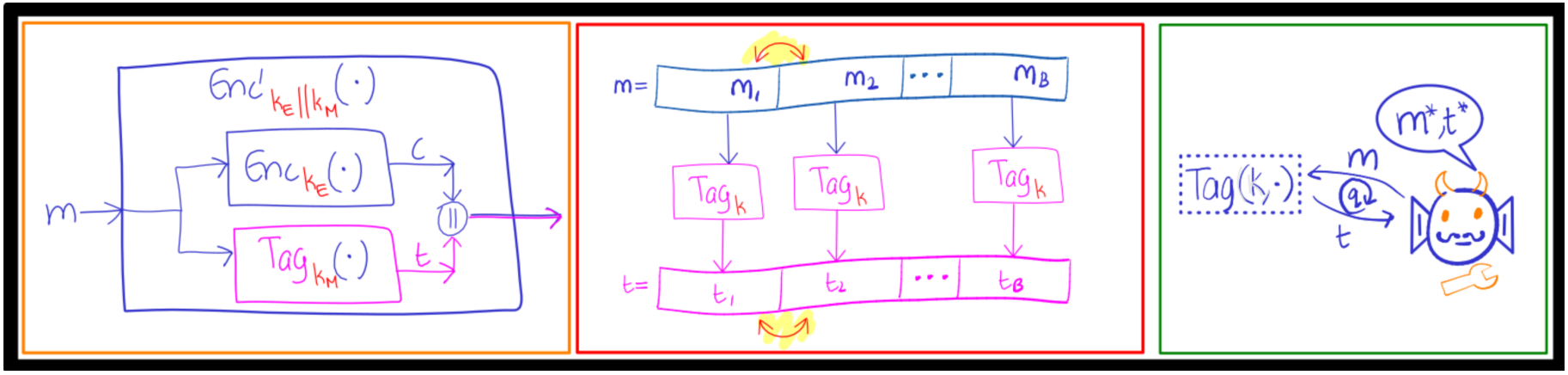


CS409m: Introduction to Cryptography

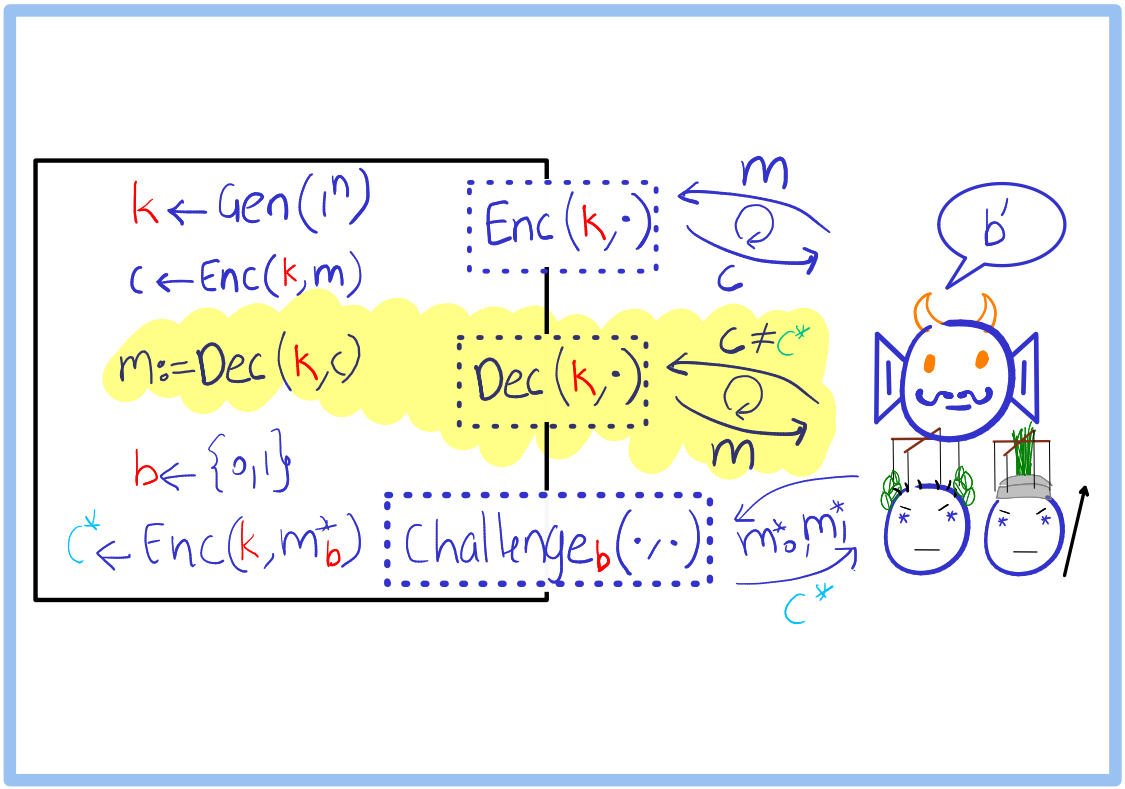
Lecture 11 (12/Sep/25)

Instructor: Chethan Kamath

Recall from Previous Lecture...

- Task: secure comm. of *multiple long messages* with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

IND-CCA

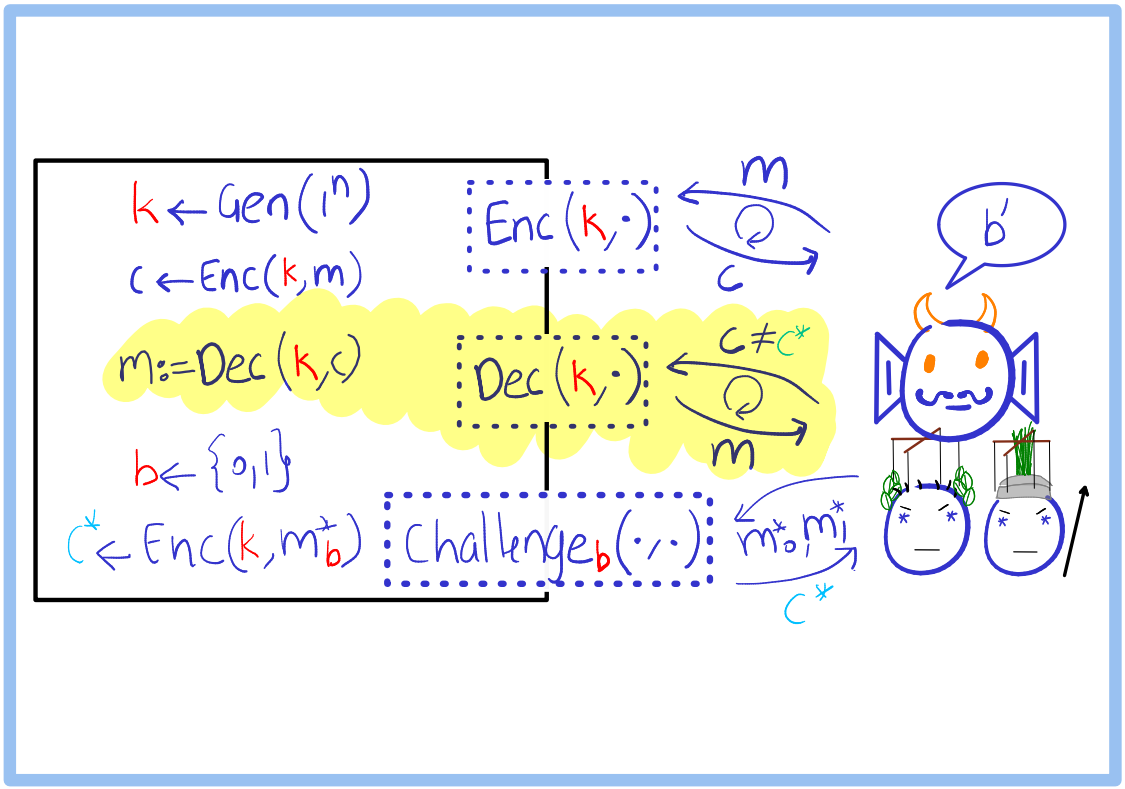


Takeaway: ciphertext malleability can lead to vulnerability to CCA

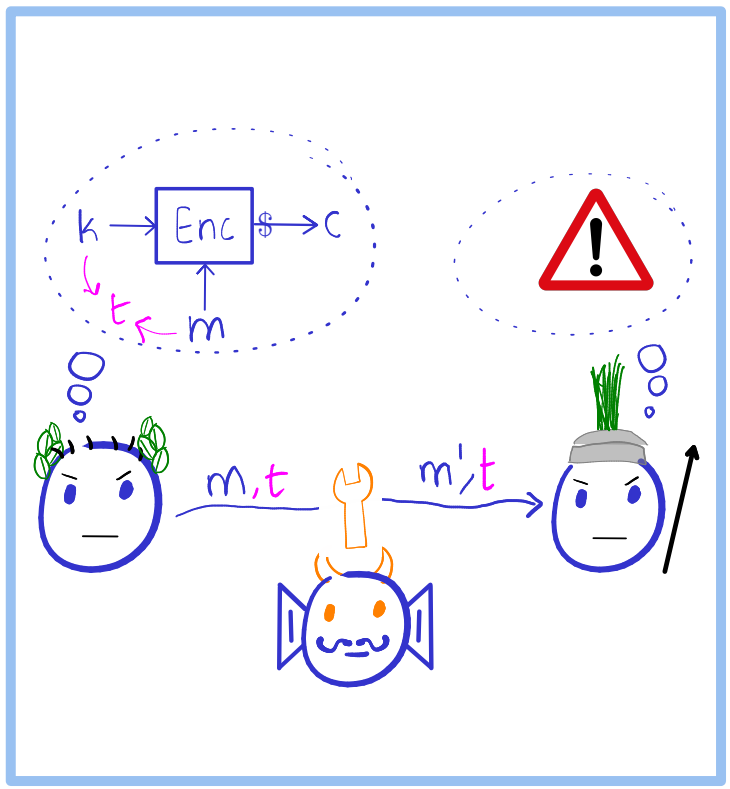
Recall from Previous Lecture...

- Task: secure comm. of *multiple long messages* with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

IND-CCA



MAC

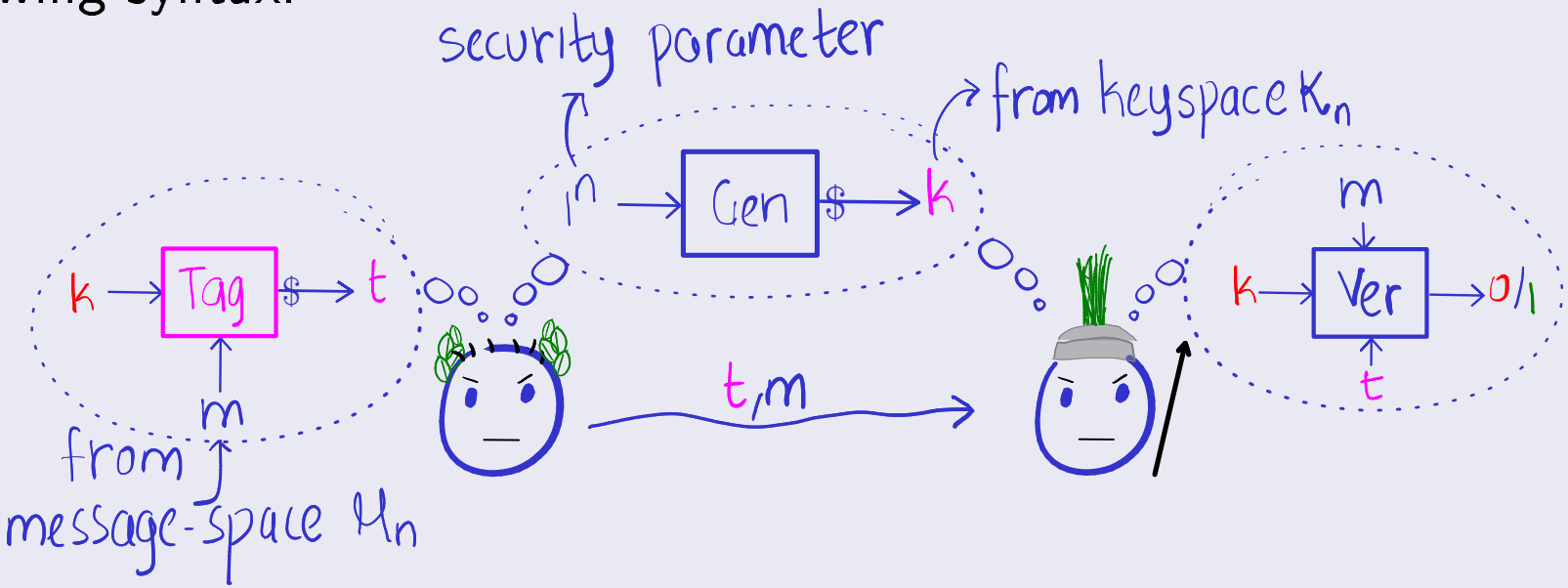


Takeaway: ciphertext malleability can lead to vulnerability to CCA

Recall Message-Authentication Code (MAC)...

Definition 1 (Lecture 10, Syntax of MAC)

A MAC M is a triple of efficient algorithms (Gen, Tag, Ver) with the following syntax:



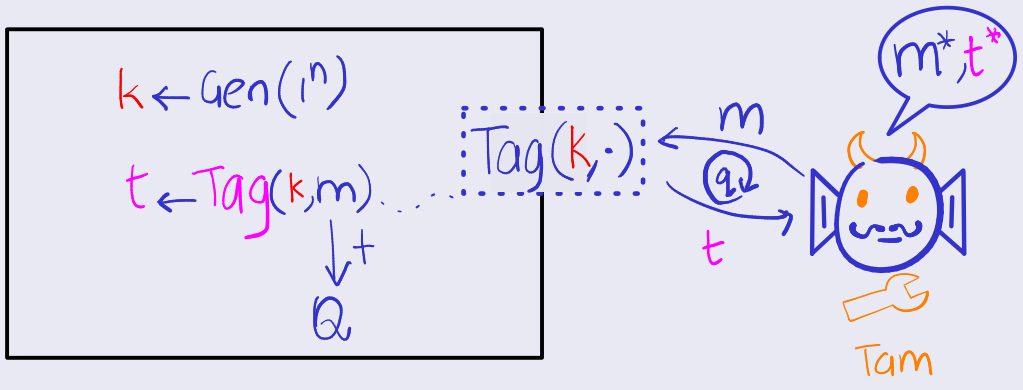
- Correctness of verification: for every $n \in \mathbb{N}$, message $m \in \mathcal{M}_n$,

$$\Pr_{k \leftarrow \mathrm{Gen}(1^n), t \leftarrow \mathrm{Tag}(k,m)} [\mathrm{Ver}(k, t, m) = 1] = 1$$

Recall Message-Authentication Code (MAC)...

Definition 2 (Lecture 10, EU-CMA)

A MAC $M = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is (ϵ, q)-EU-CMA secure if no PPT tampering adversary Tam that makes at most q queries can break M as below with probability more than ϵ

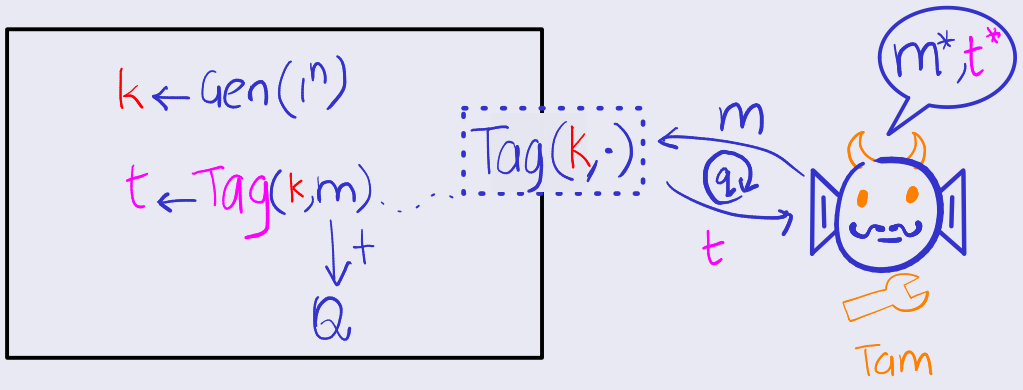


- Tam makes q queries to Tag(k,·) oracle
- In the end Tam outputs (m*,t*) and breaks if
  1) $m^* \notin Q$
  2) $\mathsf{Ver}(k, t^*, m^*) = 1$

Recall Message-Authentication Code (MAC)...

Definition 2 (Lecture 10, EU-CMA)

A MAC $M = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is (ϵ, q)-EU-CMA secure if no PPT tampering adversary Tam that makes at most q queries can break M as below with probability more than ϵ



- Tam makes q queries to Tag(k,·) oracle
- In the end Tam outputs (m*,t*) and breaks if
  i) $m^* \notin Q$
  ii) $\mathsf{Ver}(k, t^*, m^*) = 1$

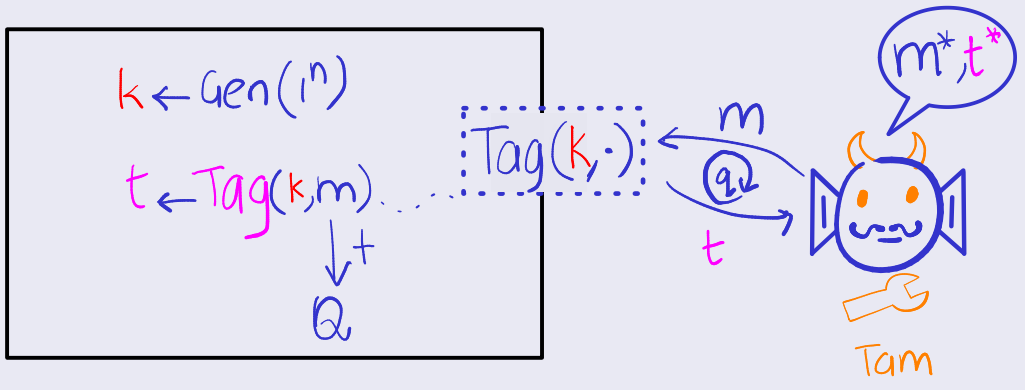
❓ If $(\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is EU-CMA, is $(\mathsf{Gen}, \mathsf{Tag}', \mathsf{Ver}')$ also EU-CMA?

1. Leaky MAC:
   - $\mathsf{Tag}'(k, m) := \mathsf{Tag}(k, m) \| m$
   - $\mathsf{Ver}'(k, t\|m', m)$, where $m' \in \mathcal{M}_n$: accept if $\mathsf{Ver}(k, t, m) = 1$

Recall Message-Authentication Code (MAC)...

Definition 2 (Lecture 10, EU-CMA)

A MAC $M = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is (ϵ, q)-EU-CMA secure if no PPT tampering adversary Tam that makes at most q queries can break M as below with probability more than ϵ



- Tam makes q queries to Tag(k,·) oracle
- In the end Tam outputs (m*,t*) and breaks if
  i) $m^* \notin Q$
  ii) $\mathsf{Ver}(k, t^*, m^*) = 1$

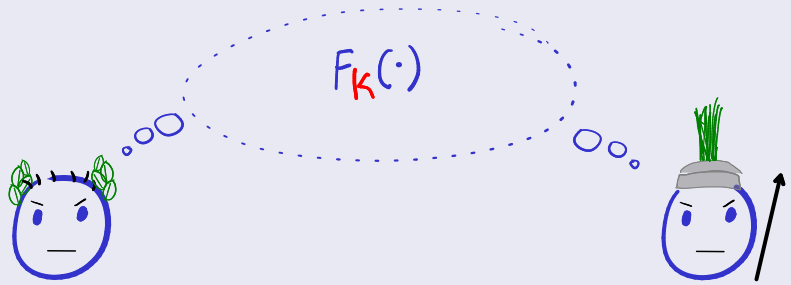
If $(\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is EU-CMA, is $(\mathsf{Gen}, \mathsf{Tag}', \mathsf{Ver}')$ also EU-CMA?

1. Leaky MAC:
   - $\mathsf{Tag}'(k, m) := \mathsf{Tag}(k, m) \| m$
   - $\mathsf{Ver}'(k, t \| m', m)$, where $m' \in \mathcal{M}_n$: accept if $\mathsf{Ver}(k, t, m) = 1$
2. Append-0 MAC
   - $\mathsf{Tag}'(k, m) := \mathsf{Tag}(k, m) \| 0$
   - $\mathsf{Ver}'(k, t \| b, m)$, where $b \in \{0, 1\}$: accept if $\mathsf{Ver}(k, t, m) = 1$

How to Construct a MAC?

Use a PRF to generate the tag!

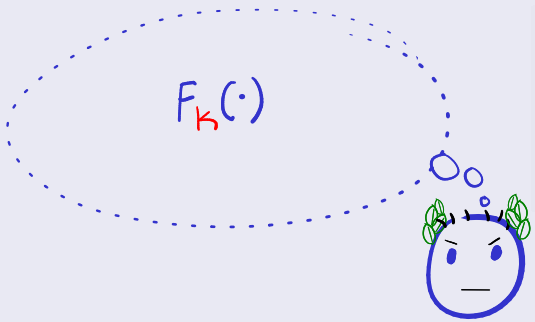
Construction 1 (for $\mathcal{M}_n = \{0,1\}^n$ using PRF $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$)

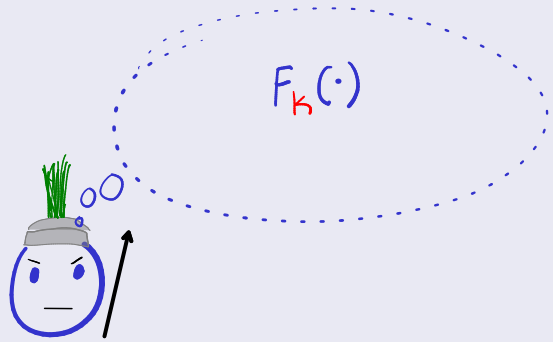


How to Construct a MAC?

Use a PRF to generate the tag!

Construction 1 (for $\mathcal{M}_n = \{0,1\}^n$ using PRF $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$)

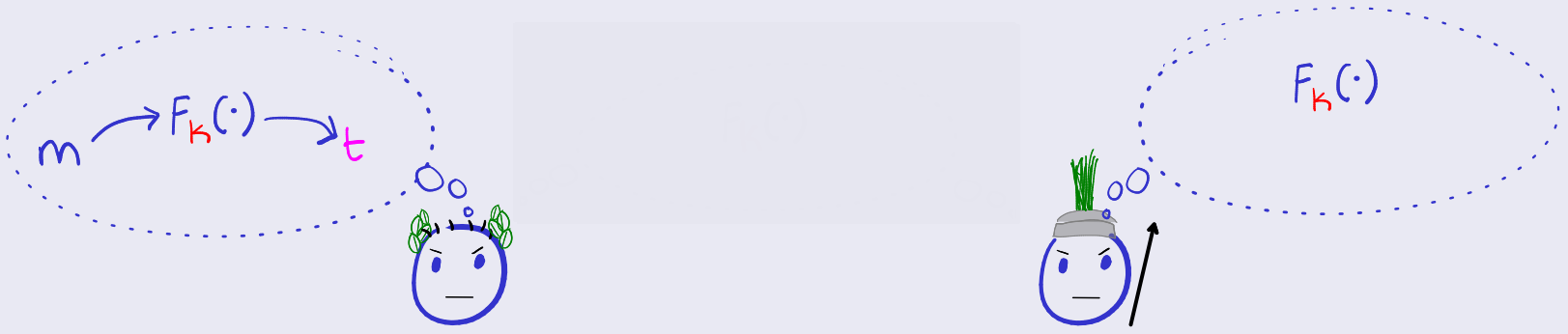




How to Construct a MAC?

Use a PRF to generate the tag!

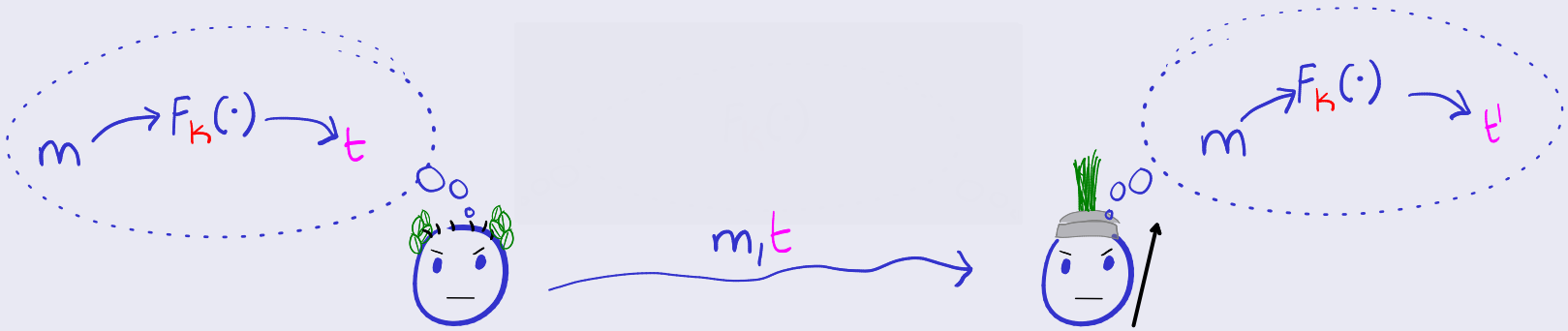
Construction 1 (for $\mathcal{M}_n = \{0,1\}^n$ using PRF $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$)



How to Construct a MAC?

Use a PRF to generate the tag!

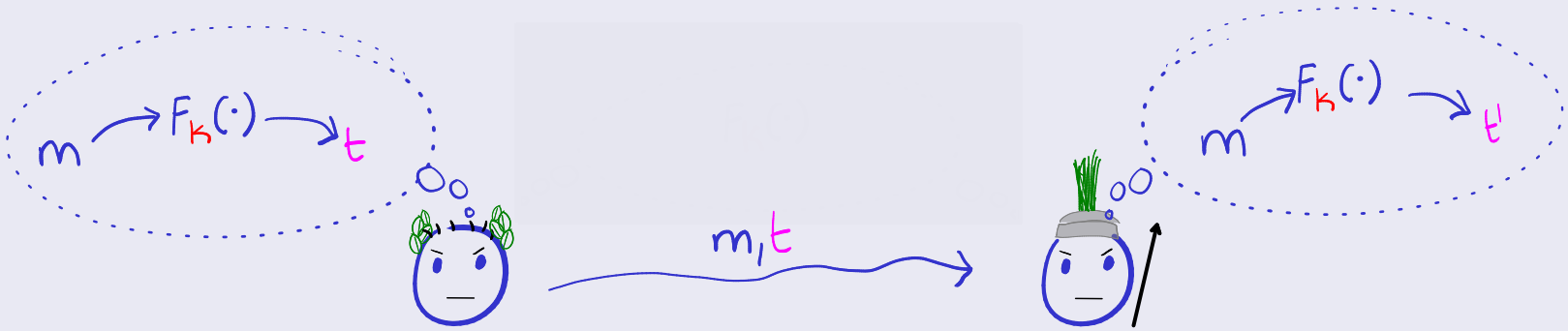
Construction 1 (for $\mathcal{M}_n = \{0,1\}^n$ using PRF $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$)



How to Construct a MAC?

Use a PRF to generate the tag!

Construction 1 (for $\mathcal{M}_n = \{0,1\}^n$ using PRF $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$)



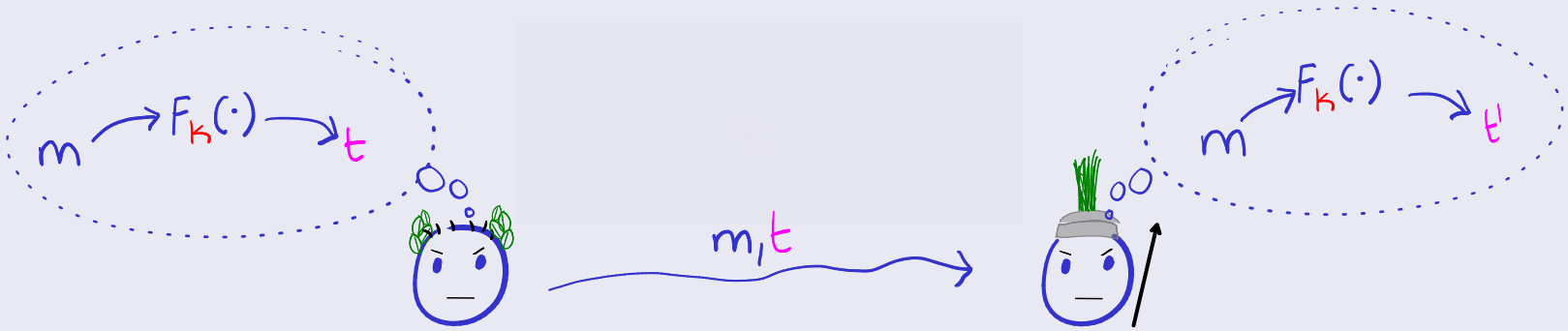
Theorem 3

If $\{F_k : \{0,1\}^n \to \{0,1\}^n\}_{k \in \{0,1\}^n}$ *is a PRF then Construction 1 is EU-CMA-secure*

How to Construct a MAC?

Use a PRF to generate the tag!

Construction 1 (for $\mathcal{M}_n = \{0,1\}^n$ using PRF $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$)



Theorem 3

If $\{F_k : \{0,1\}^n \to \{0,1\}^n\}_{k \in \{0,1\}^n}$ *is a PRF then Construction 1 is EU-CMA-secure*

Proof by reduction.

On the whiteboard □

Plan for Today's Lecture

- Task: secure comm. of *multiple long* messages with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

Plan for Today's Lecture

- Task: secure comm. of *multiple long* messages with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

CPA+EU-CMA $\Longrightarrow$ CCA

© VAwebteam-Wikipedia ©svgsilh.com/23776

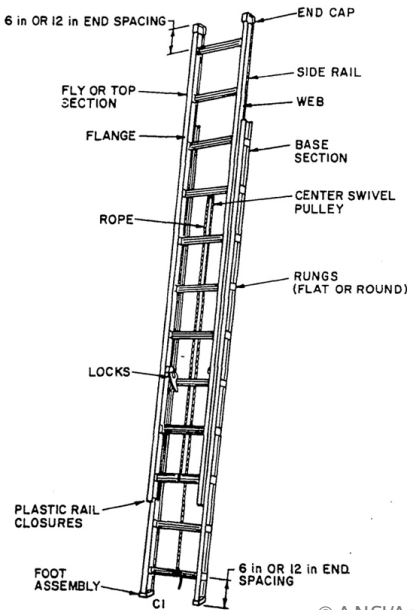
Plan for Today's Lecture

- Task: secure comm. of *multiple long* messages with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

CPA+EU-CMA $\Longrightarrow$ CCA

© VAwebteam-Wikipedia ©svgsilh.com/23776

Domain Extension for MAC



© ANSI/Archive.org

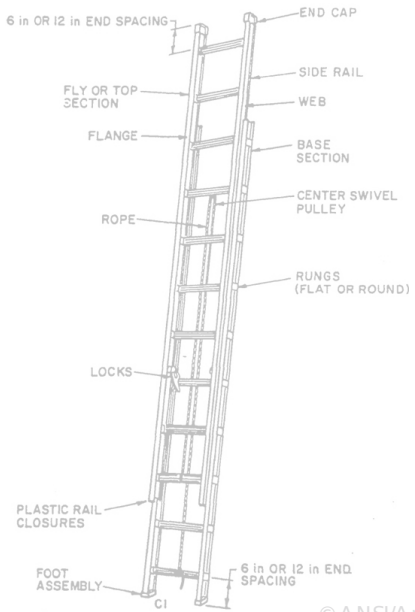
Plan for Today's Lecture

- Task: secure comm. of *multiple long* messages with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

CPA+EU-CMA $\Longrightarrow$ CCA

© VAwebteam-Wikipedia ©svgsilh.com/23776

Domain Extension for MAC



Attempt I: Authenticate-and-Encrypt

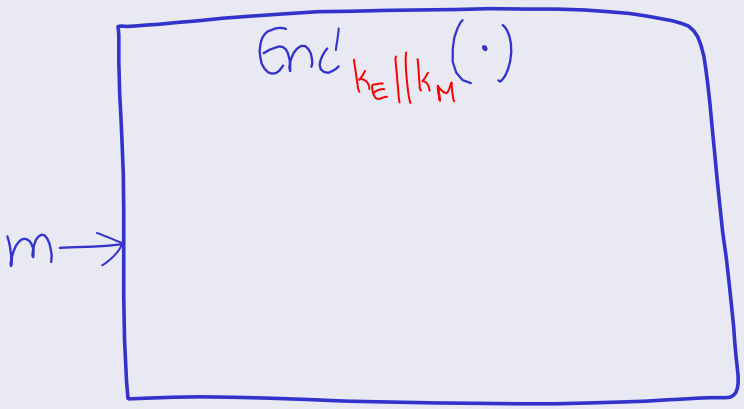
(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

Construction 2 (IND-CPA Π + EU-CMA M $\Rightarrow$ IND-CCA Π')

Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

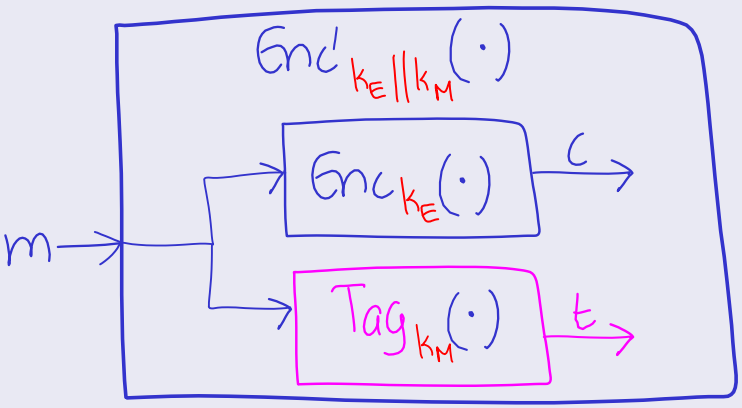
Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

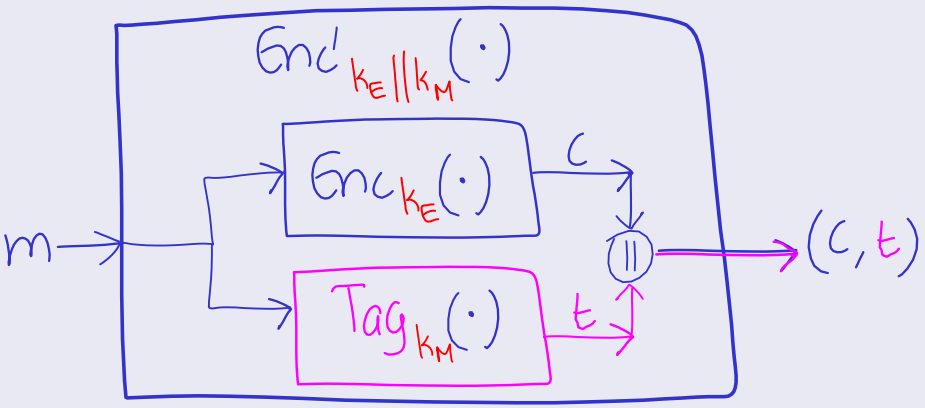
Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

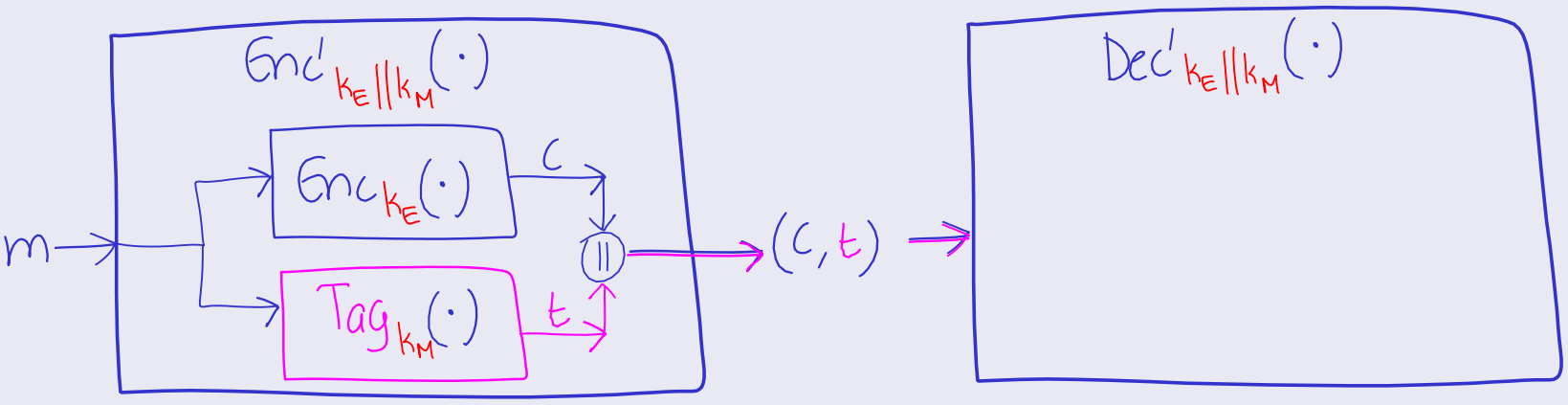
Construction 2 (IND-CPA Π + EU-CMA M $\Rightarrow$ IND-CCA Π')



Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

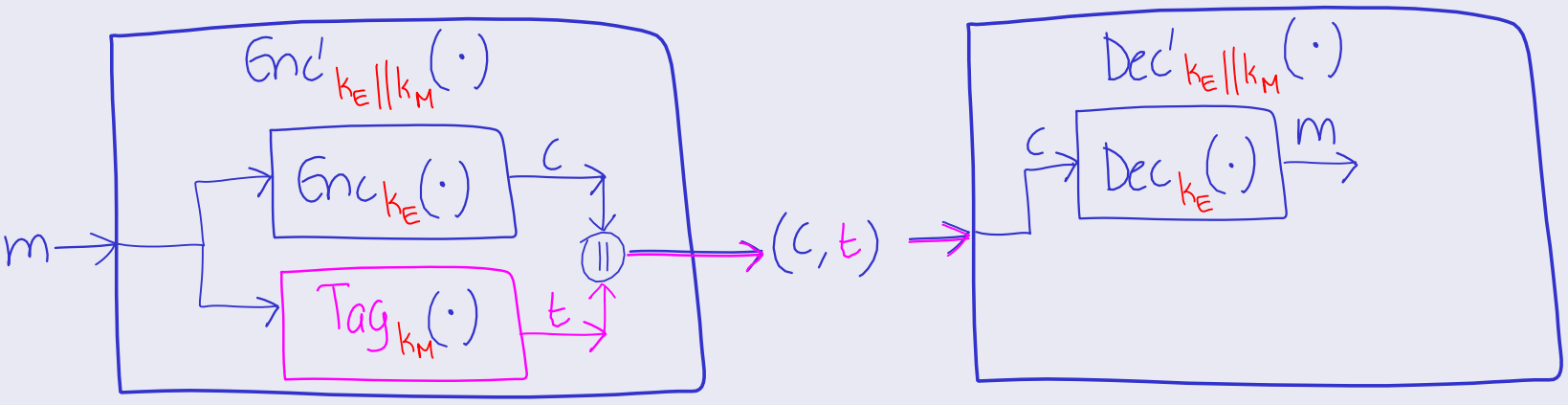
Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

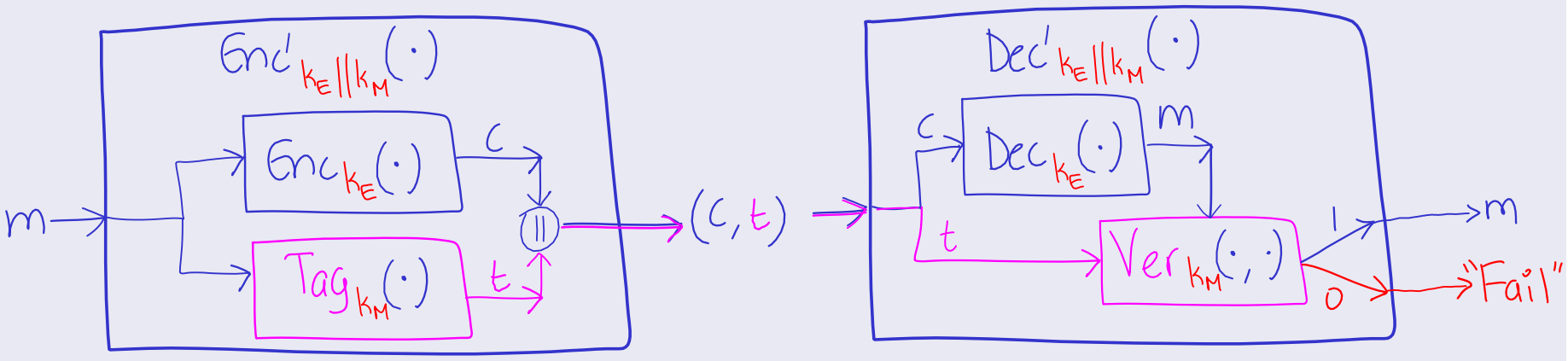
Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

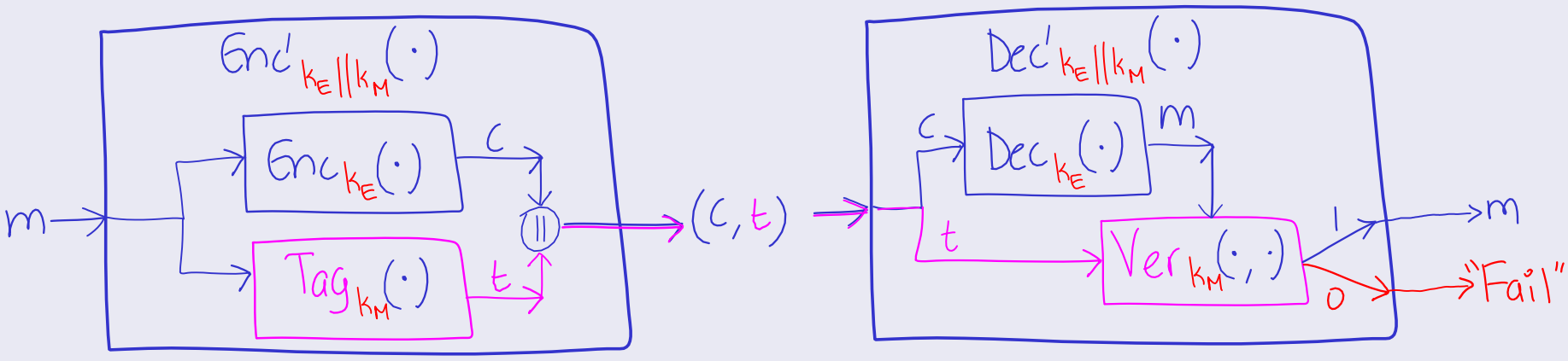
Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)

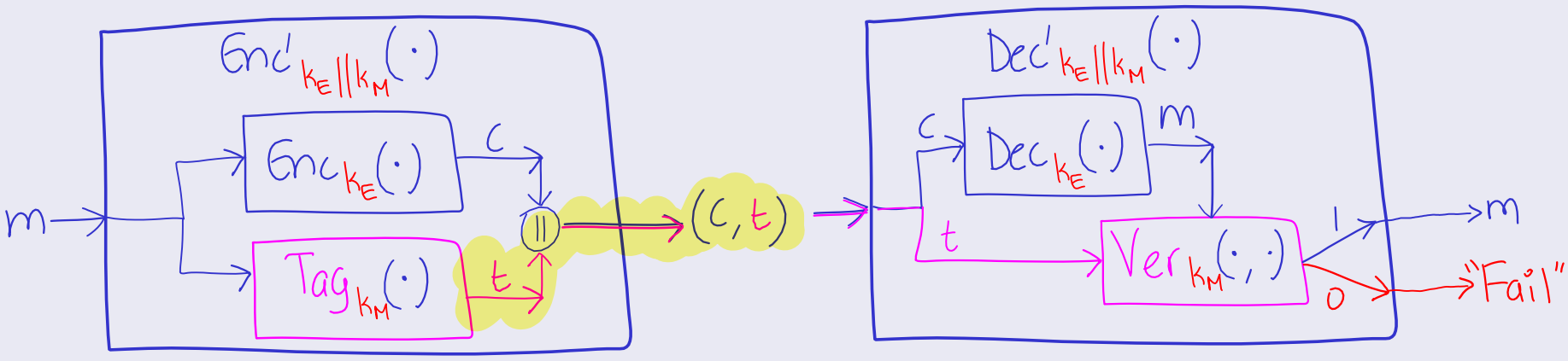


? Is Construction 2 IND-CCA-secure?

Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)

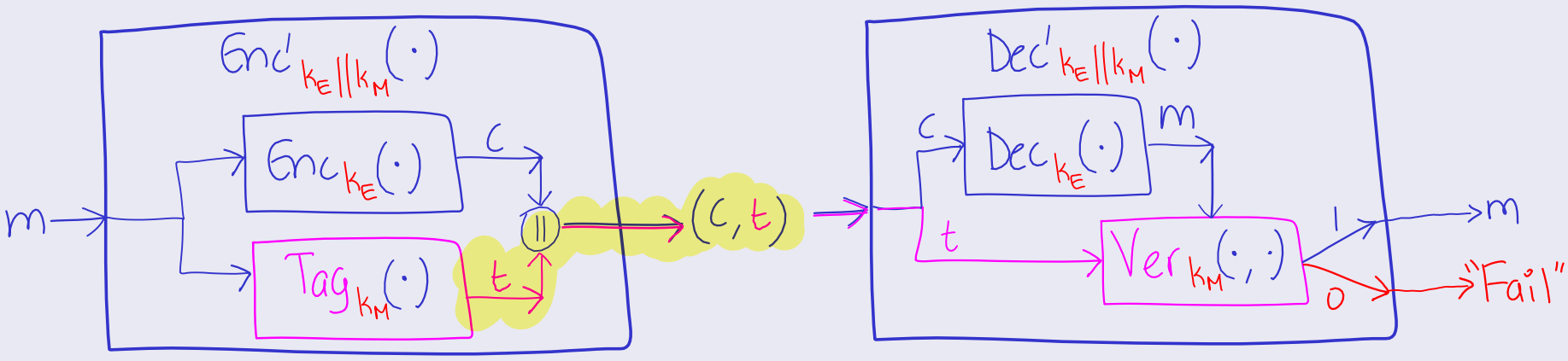


- Is Construction 2 IND-CCA-secure?
- No, *totally insecure* if M's tag leaks information about message!
  - E.g., consider M = Leaky MAC

Attempt I: Authenticate-and-Encrypt

(Gen, Enc, Dec) (Gen, Tag, Ver) (Gen', Enc', Dec')

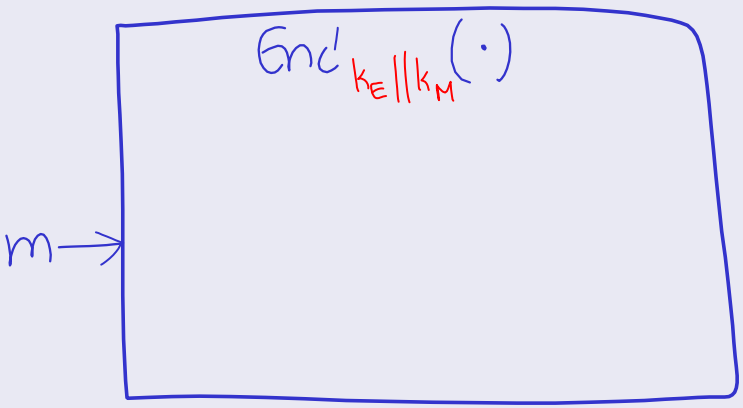
Construction 2 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



- Is Construction 2 IND-CCA-secure?
- ⚠ No, *totally insecure* if M's tag leaks information about message!
  - E.g., consider M = Leaky MAC
    - ⚠ Attack:
      1. Challenge on (arbitrary) m_0^*, m_1^* to obtain (c^*, t^*)
      2. Output 0 if t^* contains m_0^*; otherwise output 1

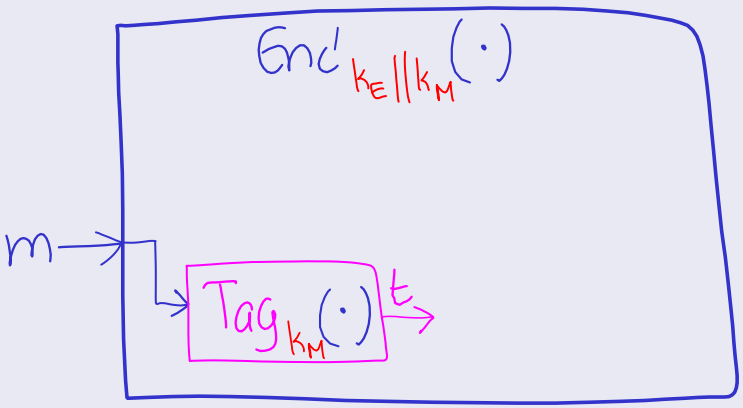
Attempt II: Authenticate-then-Encrypt

Construction 3 (IND-CPA Π + EU-CMA M $\Rightarrow$ IND-CCA Π')



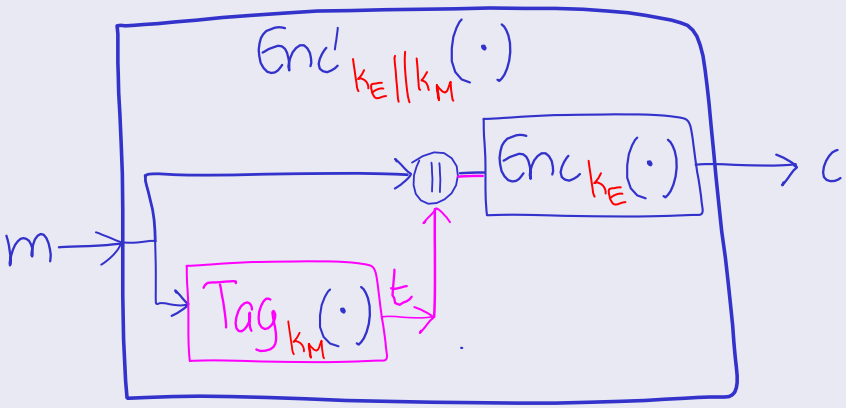
Attempt II: Authenticate-then-Encrypt

Construction 3 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



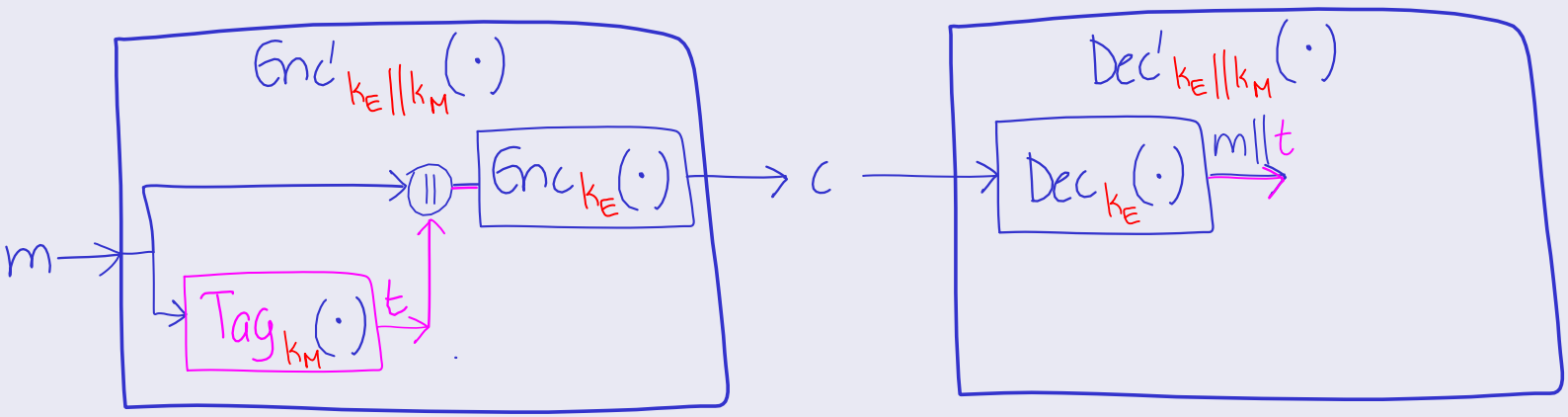
Attempt II: Authenticate-then-Encrypt

Construction 3 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



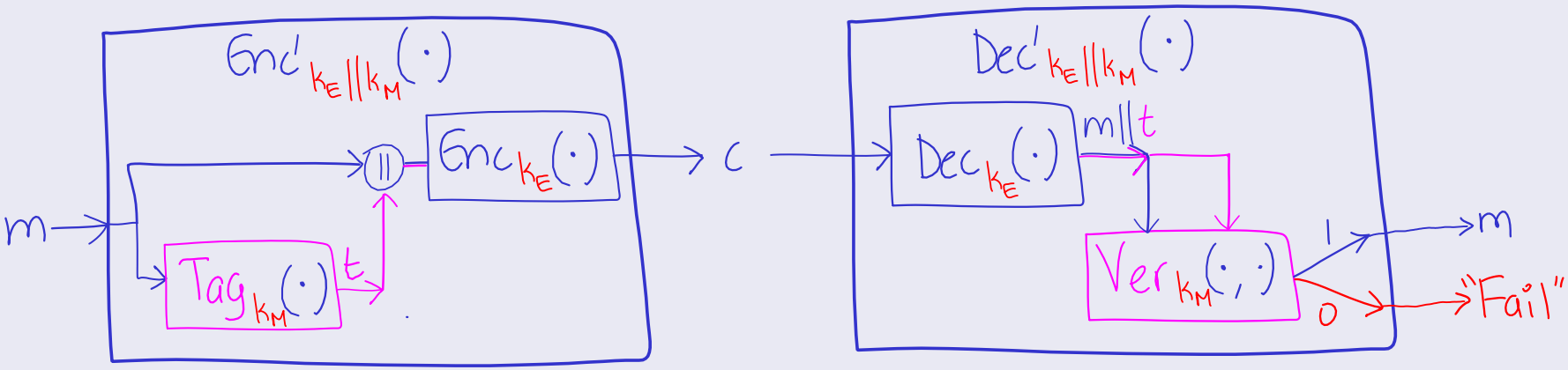
Attempt II: Authenticate-then-Encrypt

Construction 3 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



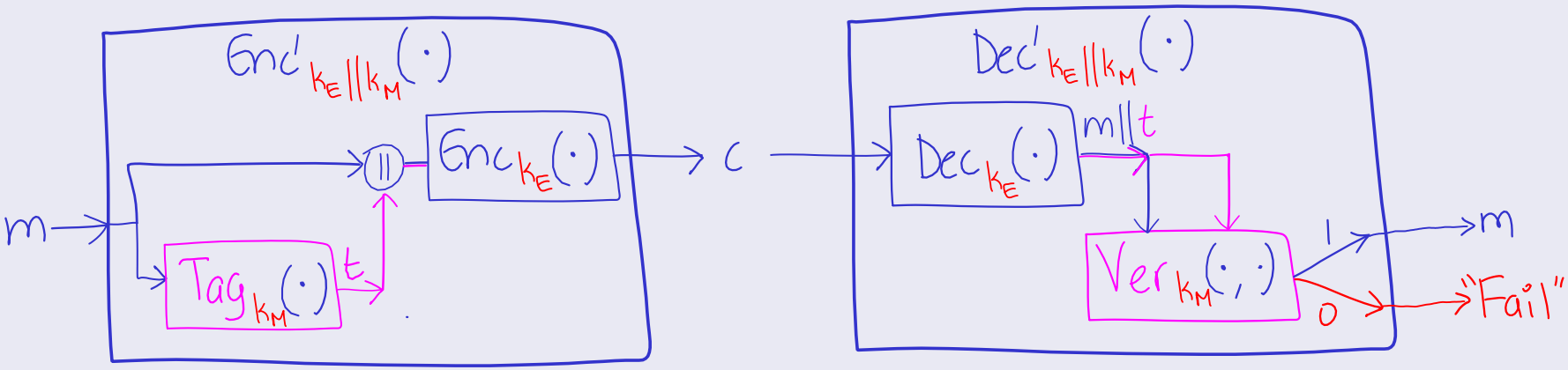
Attempt II: Authenticate-then-Encrypt

Construction 3 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt II: Authenticate-then-Encrypt

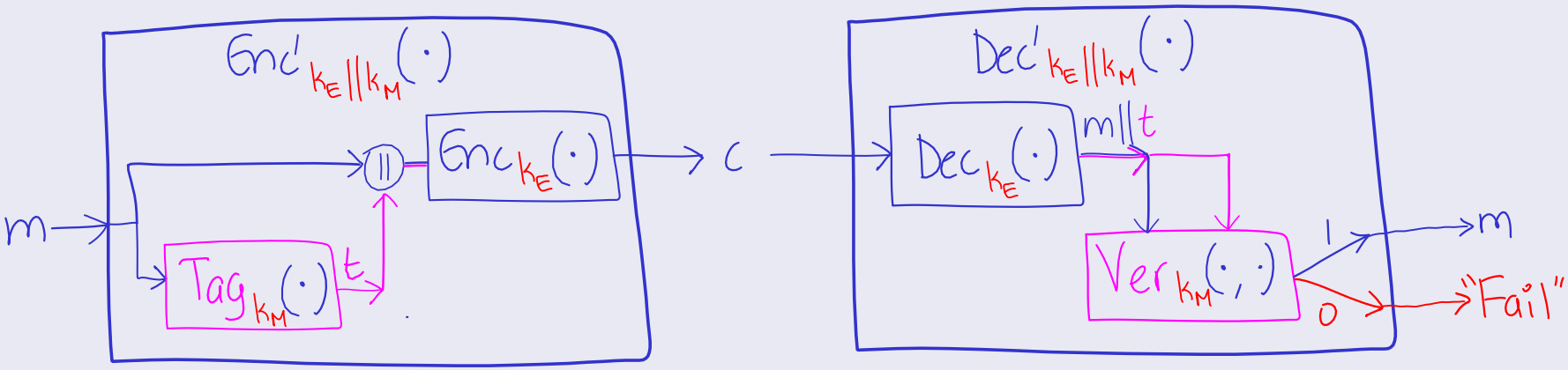
Construction 3 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



❓ Is Construction 3 IND-CCA-secure?

Attempt II: Authenticate-then-Encrypt

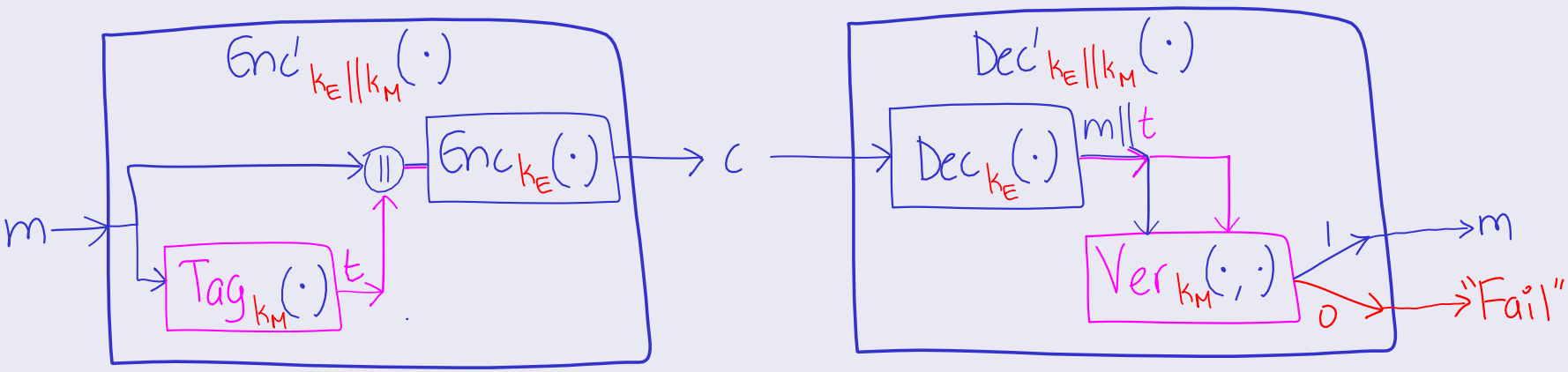
Construction 3 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



- Is Construction 3 IND-CCA-secure?
- No, Π and thus Π′ might be *malleable*!
  - E.g., consider Π = CBC mode (Lecture 09)

Attempt II: Authenticate-then-Encrypt

Construction 3 (IND-CPA Π + EU-CMA M $\Rightarrow$ IND-CCA Π')



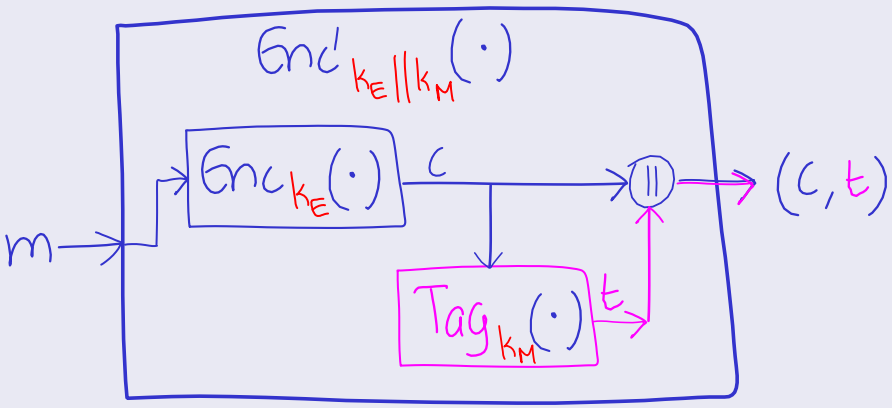
- ? Is Construction 3 IND-CCA-secure?
- ⚠ No, Π and thus Π' might be *malleable*!
  - E.g., consider Π = CBC mode (Lecture 09)

Exercise 1 ⚠

Extend the padding oracle attack to Construction 3. (Hint: assume different error messages for decryption failure and tag verification failure)

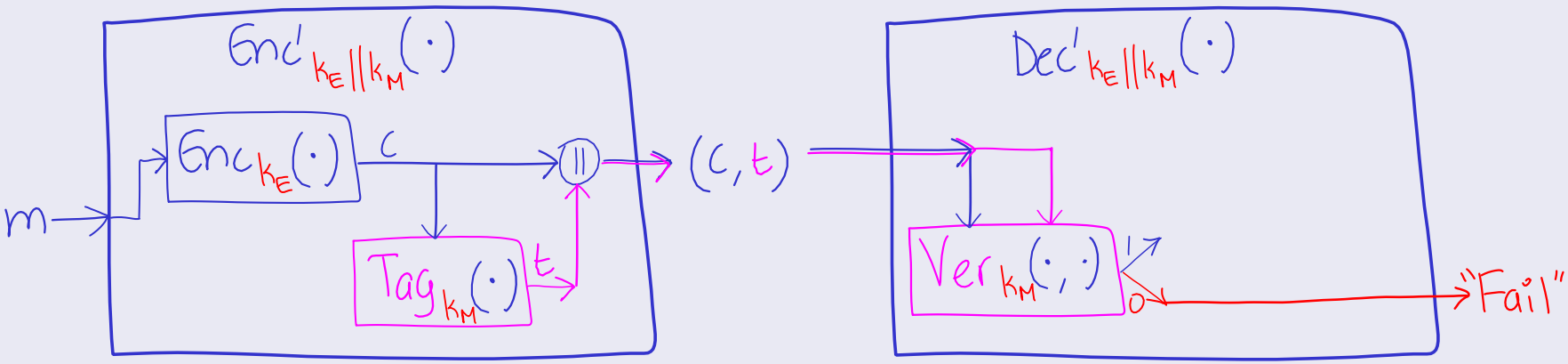
Attempt III: Encrypt-then-Authenticate

Construction 4 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



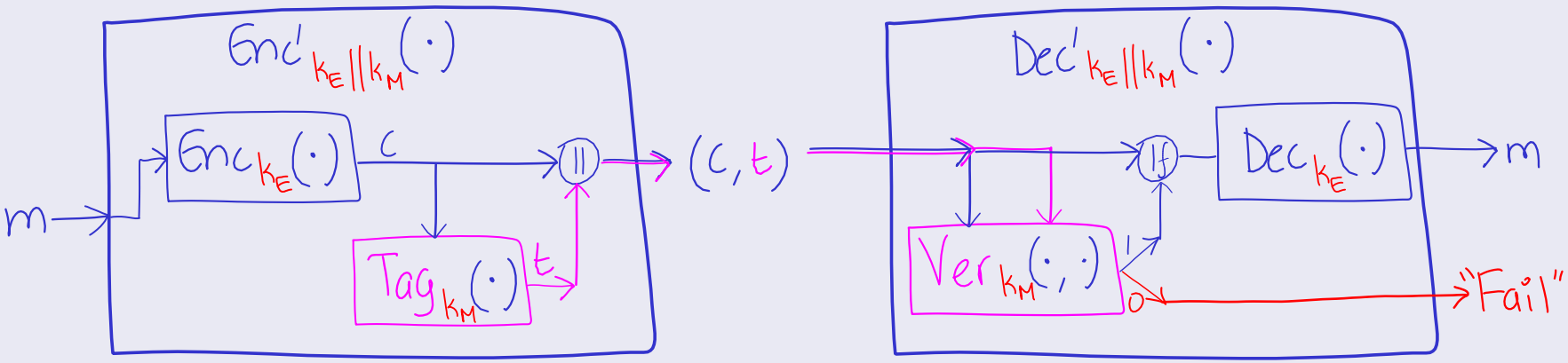
Attempt III: Encrypt-then-Authenticate

Construction 4 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



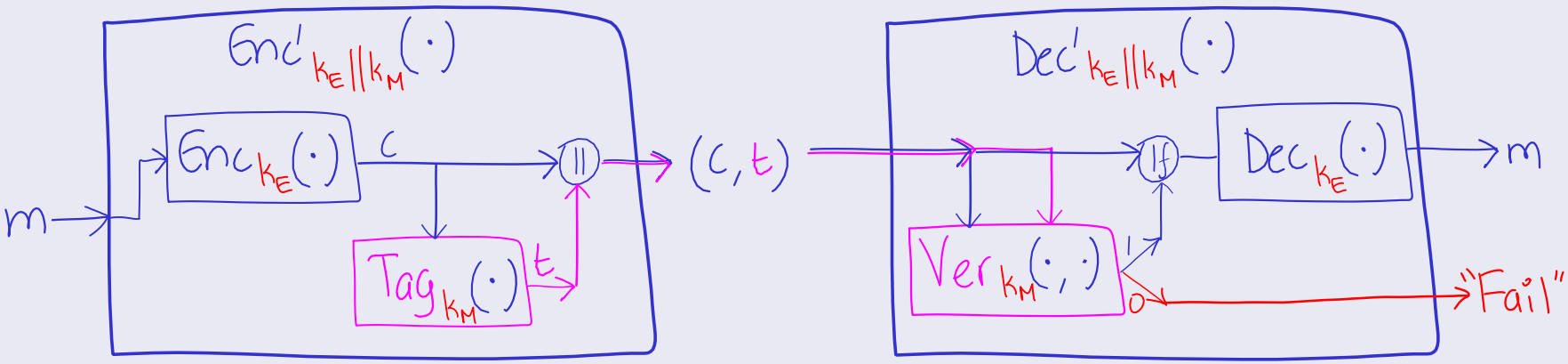
Attempt III: Encrypt-then-Authenticate

Construction 4 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



Attempt III: Encrypt-then-Authenticate

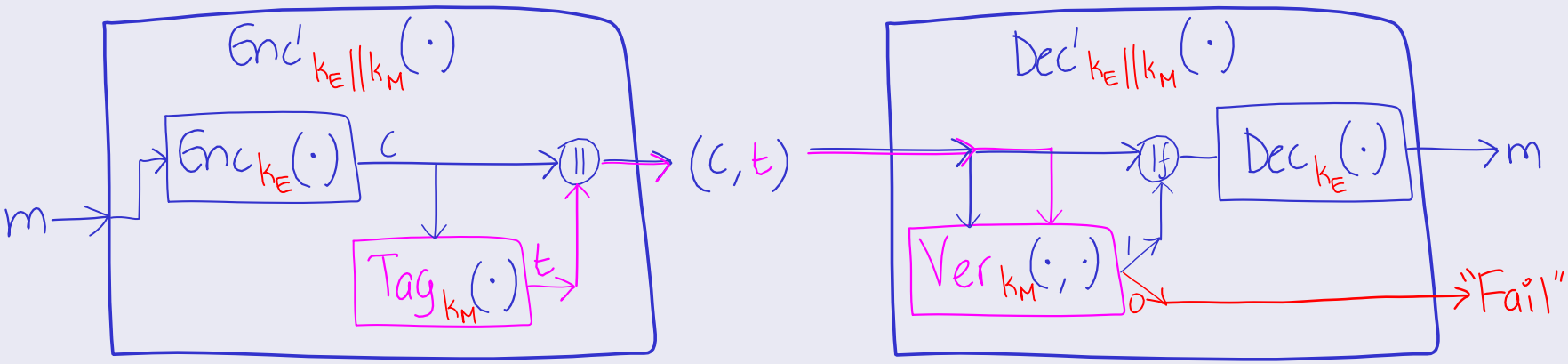
Construction 4 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



❓ Is Construction 4 IND-CCA-secure?

Attempt III: Encrypt-then-Authenticate

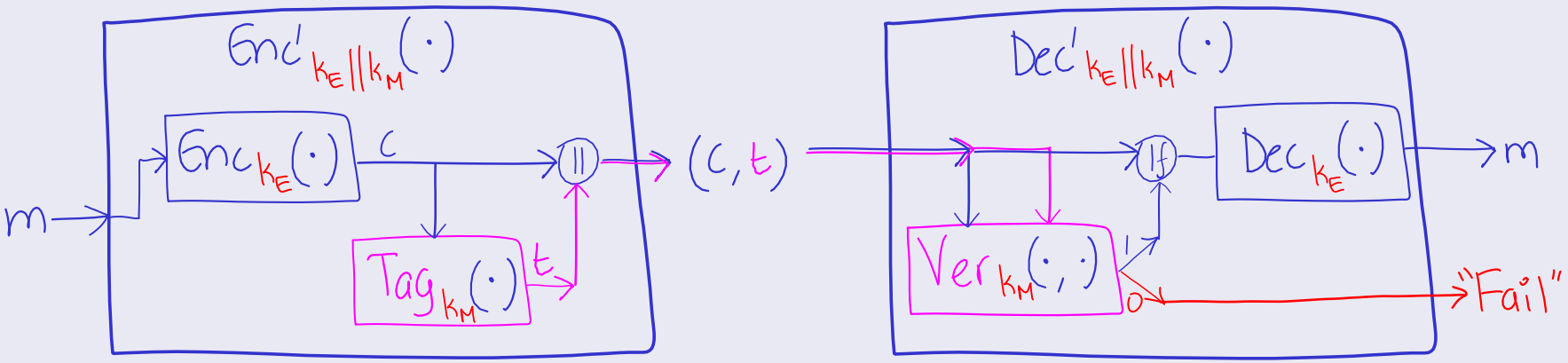
Construction 4 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



- Is Construction 4 IND-CCA-secure?
- ⚠ No, M might be "malleable"
  - E.g., consider M = Append-0 MAC

Attempt III: Encrypt-then-Authenticate

Construction 4 (IND-CPA Π + EU-CMA M ⇒ IND-CCA Π′)



- Is Construction 4 IND-CCA-secure?
- No, M might be "malleable"
  - E.g., consider M = Append-0 MAC
    - Attack:
      1. Challenge on (arbitrary) m_0^*, m_1^* to obtain $(c^*, t^* \| 0)$
      2. Query decryption oracle on $(c^*, t^* \| 1)$ to obtain m^*
      3. Output 0 if $m^* = m_0^*$; otherwise output 1

Encrypt-then-Authenticate with *Strong* MAC

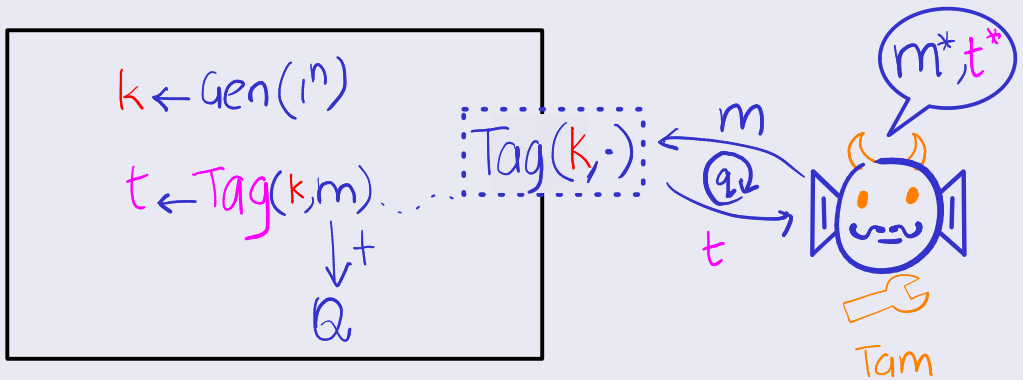
Solution: use MAC that is non-malleable = *strongly* unforgeable

Encrypt-then-Authenticate with *Strong* MAC

Solution: use MAC that is non-malleable = *strongly* unforgeable

Definition 4 (Lecture 10, EU-CMA)

A MAC $M = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is (ϵ, q)-EU-CMA secure if no PPT tampering adversary Tam that makes at most q queries can break M as below with probability more than ϵ



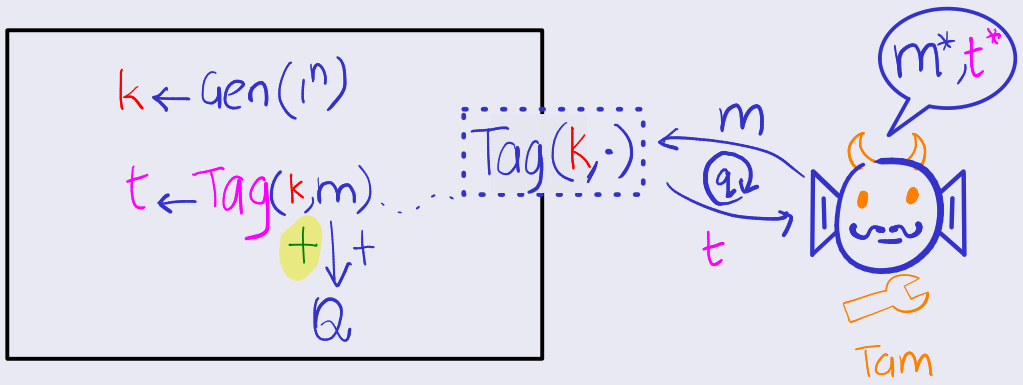
- Tam makes q queries to Tag(k,·) oracle
- In the end Tam outputs (m*,t*) and breaks if
  - i) $m^* \notin Q$
  - ii) $\mathsf{Ver}(k, t^*, m^*) = 1$

Encrypt-then-Authenticate with *Strong* MAC

Solution: use MAC that is non-malleable = *strongly* unforgeable

Definition 4 (Lecture 10, S EU-CMA)

A MAC M = (Gen, Tag, Ver) is (ϵ, q)-S EU-CMA secure if no PPT tampering adversary Tam that makes at most q queries can break M as below with probability more than ϵ



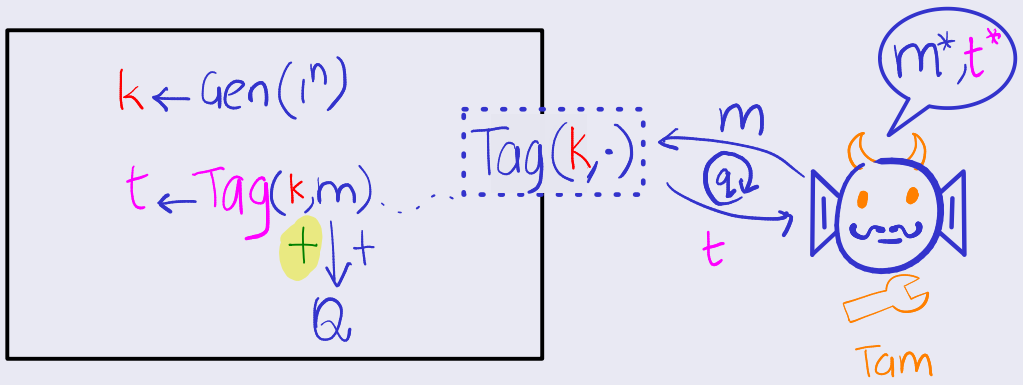
- Tam makes q queries to Tag(k,·) oracle
- In the end Tam outputs (m*,t*) and breaks if
  i) (m*,t*) ∉ Q
  ii) Ver(k, t*, m*) = 1

Encrypt-then-Authenticate with *Strong* MAC

Solution: use MAC that is non-malleable = *strongly* unforgeable

Definition 4 (Lecture 10, EU-CMA)

A MAC $M = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is (ϵ, q)-EU-CMA secure if no PPT tampering adversary Tam that makes at most q queries can break M as below with probability more than ϵ



- Tam makes q queries to Tag(k,·) oracle
- In the end Tam outputs (m*,t*) and breaks if
  i) (m*,t*) ∉ Q
  ii) Ver(k,t*,m*) = 1

Exercise 2

Show that if Π is IND-CPA secure and M is sEU-CMA secure, then Construction 4 is IND-CCA secure

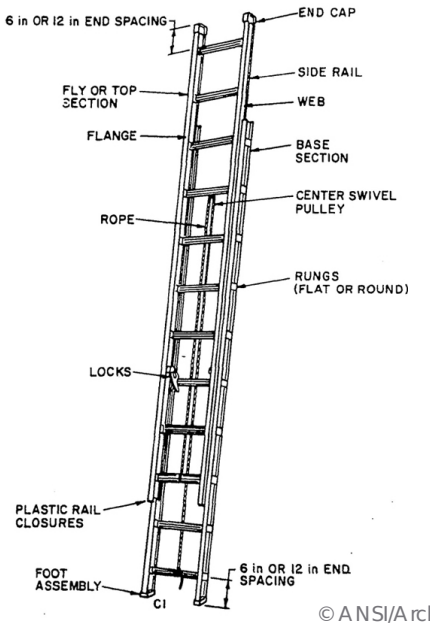
Plan for Today's Lecture

- Task: secure comm. of *multiple long* messages with shared keys
- Threat model: ind. against chosen-ciphertext attack (IND-CCA)

CPA+EU-CMA $\Longrightarrow$ CCA

© VAwebteam-Wikipedia ©svgsilh.com/23776

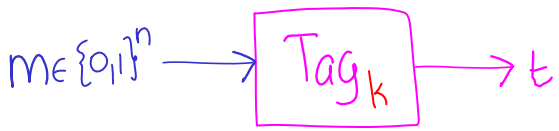
Domain Extension for MAC



© ANSI/Archive.org

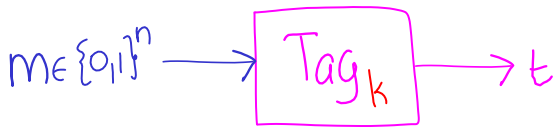
Domain Extension: Goal

- Given: MAC $\mathsf{M} = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ for $\mathcal{M}_n := \{0,1\}^n$

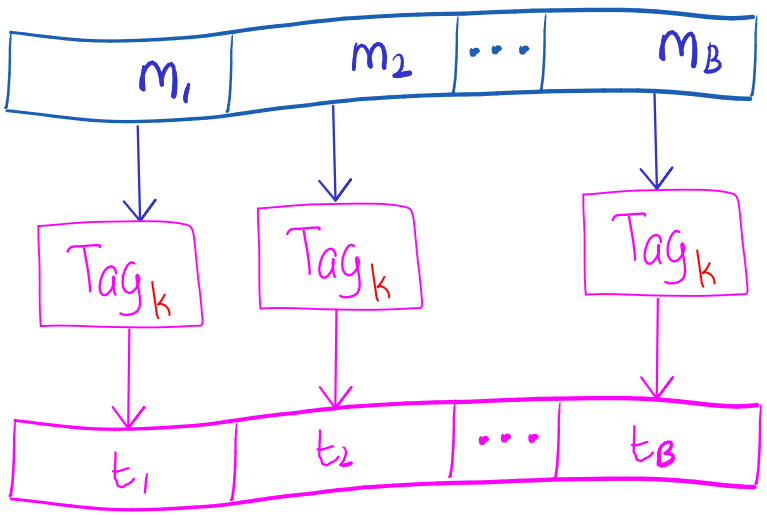


Domain Extension: Goal

- Given: MAC $\mathsf{M} = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ for $\mathcal{M}_n := \{0,1\}^n$

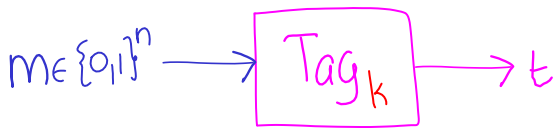


- Goal: design MAC M' for $m := m_1 \| \cdots \| m_B$, where $m_i \in \{0,1\}^n$

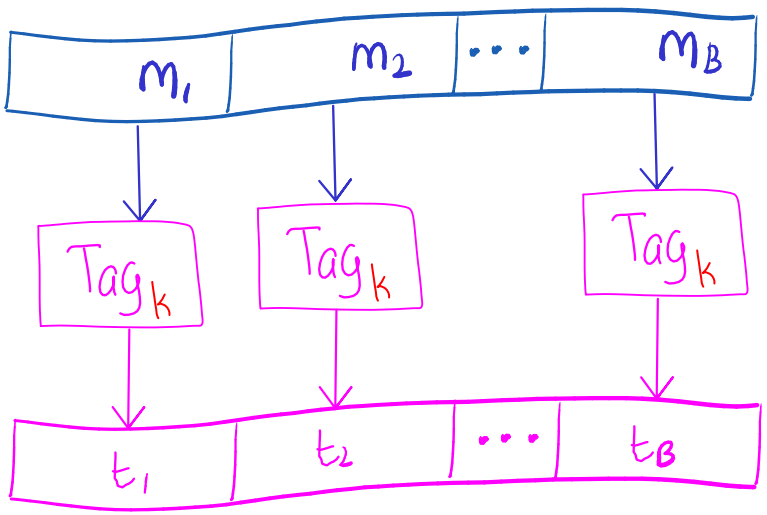


Domain Extension: Goal

- Given: MAC $\mathsf{M} = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ for $\mathcal{M}_n := \{0,1\}^n$



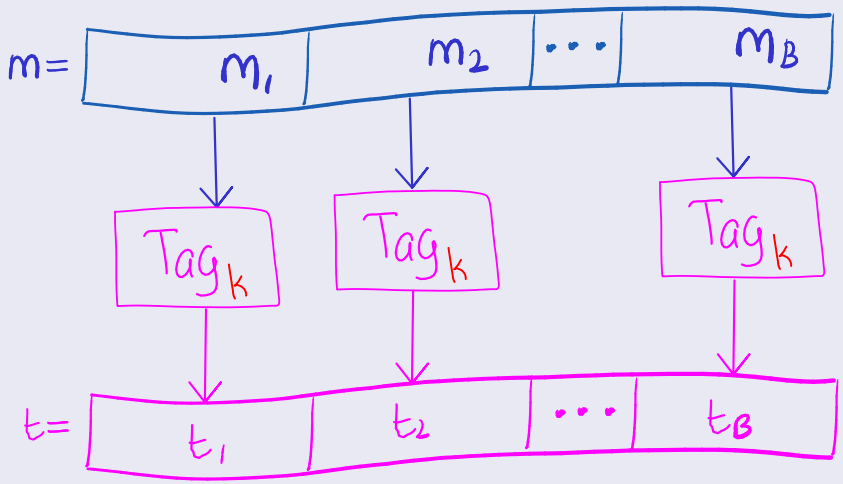
- Goal: design MAC M' for $m := m_1 \| \cdots \| m_B$, where $m_i \in \{0,1\}^n$



- Analogous to *modes of operation* for block ciphers (Lecture 08)

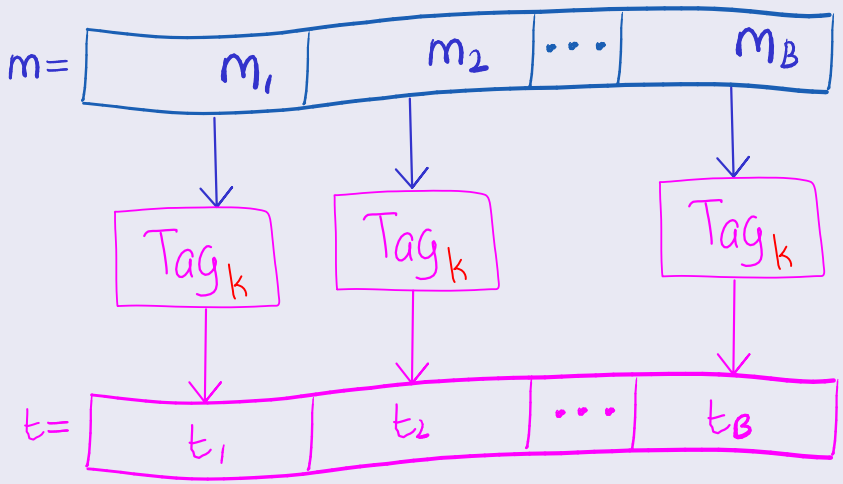
Attempt I: MAC Block-wise

Construction 5 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



Attempt I: MAC Block-wise

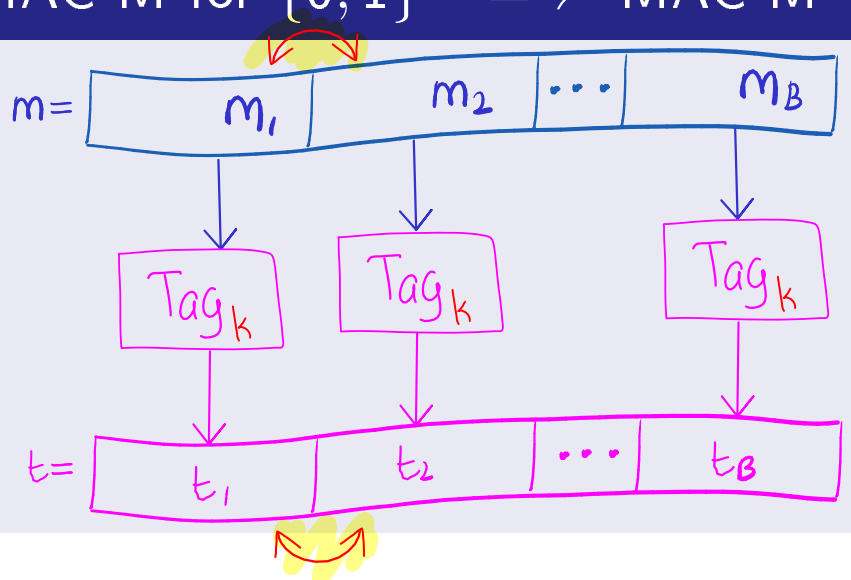
Construction 5 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



Is Construction 5 EU-CMA-secure?

Attempt I: MAC Block-wise

Construction 5 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



❓ Is Construction 5 EU-CMA-secure?

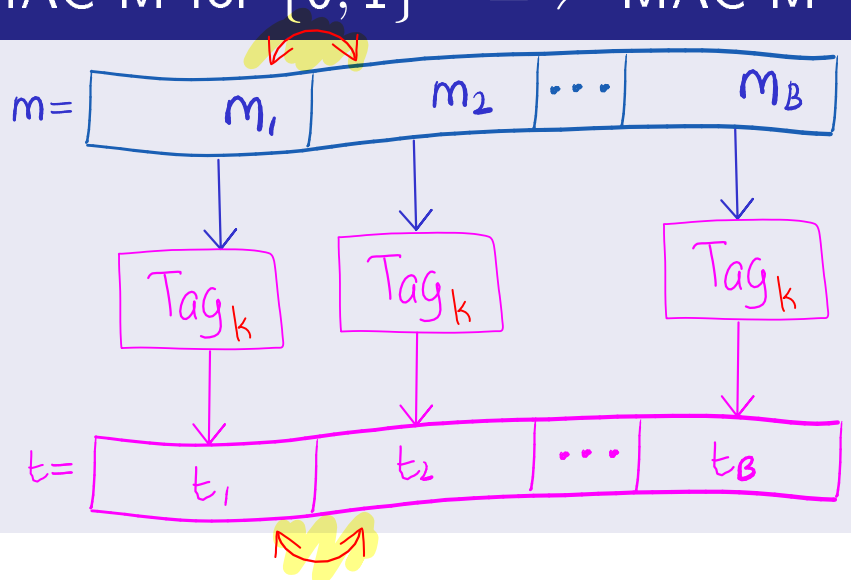
⚠ No, can *reorder* tag/message blocks!

⚠ Attack:

1. Query tag oracle on (m_1, m_2) to obtain (t_1, t_2)
2. Output (t_2, t_1) as tag on (m_2, m_1)

Attempt I: MAC Block-wise

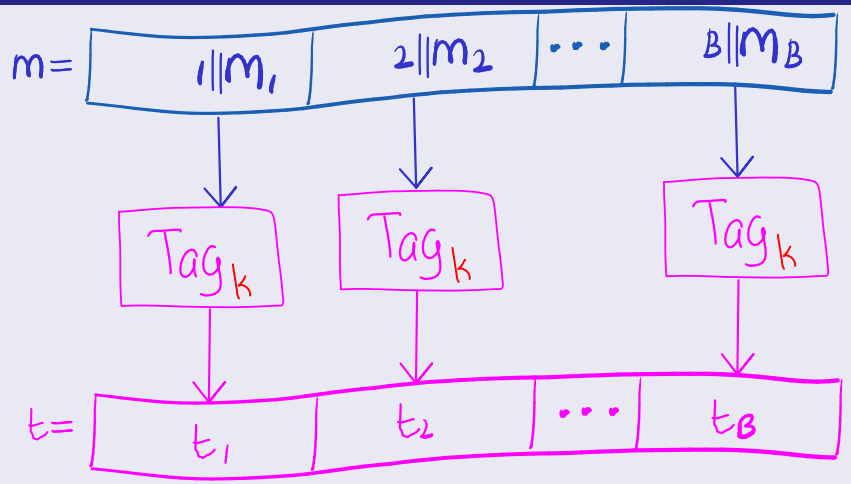
Construction 5 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



- Is Construction 5 EU-CMA-secure?
- No, can *reorder* tag/message blocks!
- Attack:
  1. Query tag oracle on (m_1, m_2) to obtain (t_1, t_2)
  2. Output (t_2, t_1) as tag on (m_2, m_1)
- Fix: prepend block number

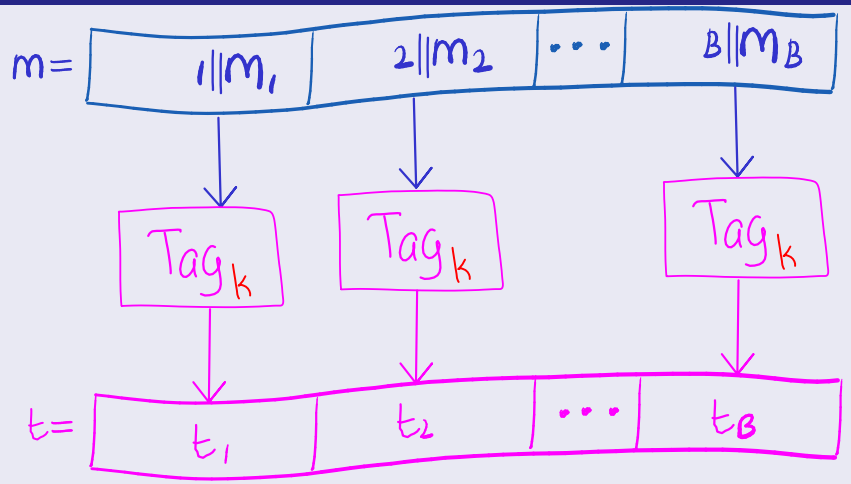
Attempt II: MAC Block-wise, with Block No.

Construction 6 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



Attempt II: MAC Block-wise, with Block No.

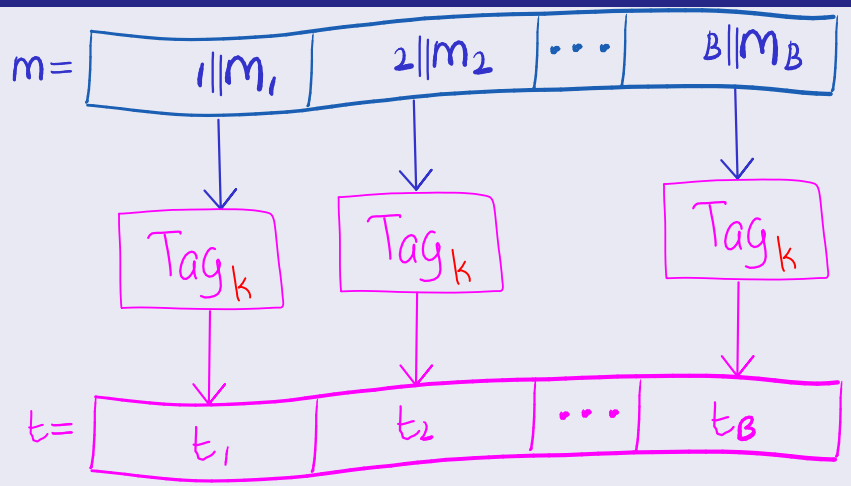
Construction 6 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



❓ Is Construction 6 EU-CMA-secure?

Attempt II: MAC Block-wise, with Block No.

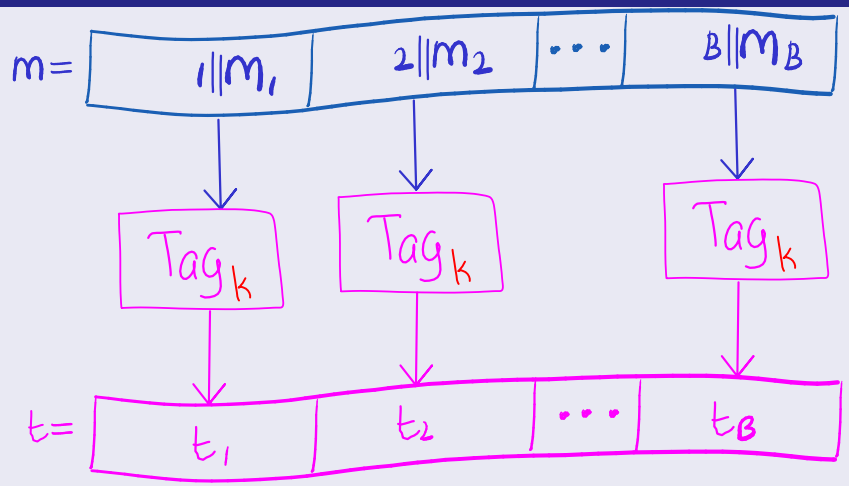
Construction 6 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



- Is Construction 6 EU-CMA-secure?
- No, can *mix and match* blocks!
- Attack:
  1. For $m_1 \neq m_2 \in \{0,1\}^n$, query tag oracle on (m_1, m_2) to get (t_1, t_2)
  2. Query tag oracle on (m_2, m_1) to obtain (t_1', t_2')
  3. Output (t_1, t_2') as tag on (m_1, m_1)

Attempt II: MAC Block-wise, with Block No.

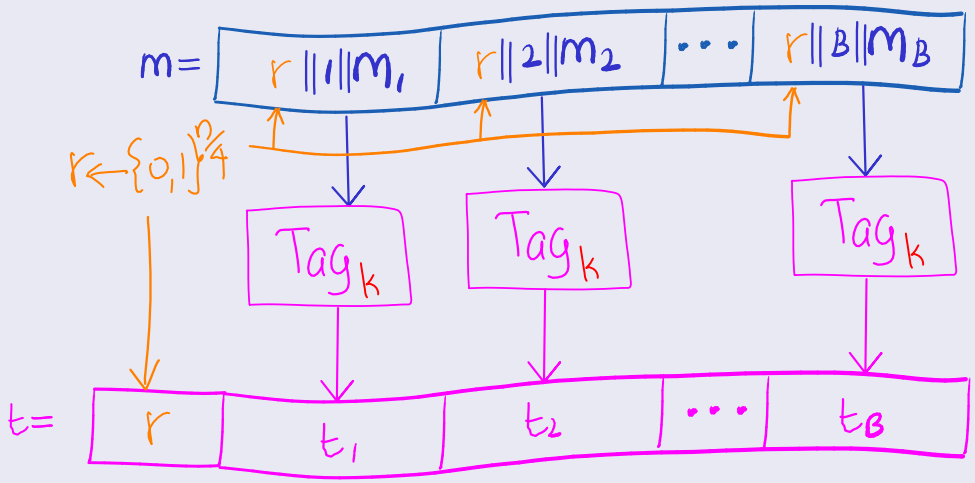
Construction 6 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



- Is Construction 6 EU-CMA-secure?
- No, can *mix and match* blocks!
- Attack:
  1. For $m_1 \neq m_2 \in \{0,1\}^n$, query tag oracle on (m_1, m_2) to get (t_1, t_2)
  2. Query tag oracle on (m_2, m_1) to obtain (t_1', t_2')
  3. Output (t_1, t_2') as tag on (m_1, m_1)
- Fix: also prepend a random "nonce"

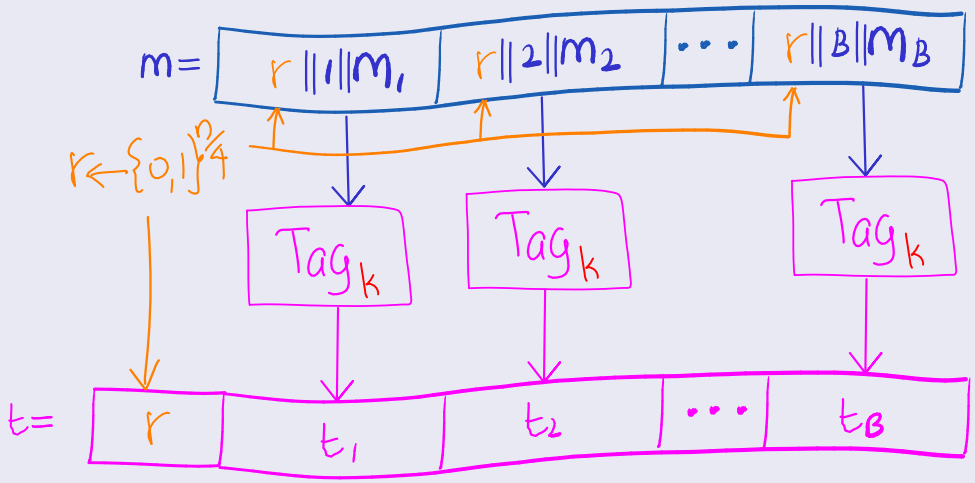
Domain Extension for MAC

Construction 7 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



Domain Extension for MAC

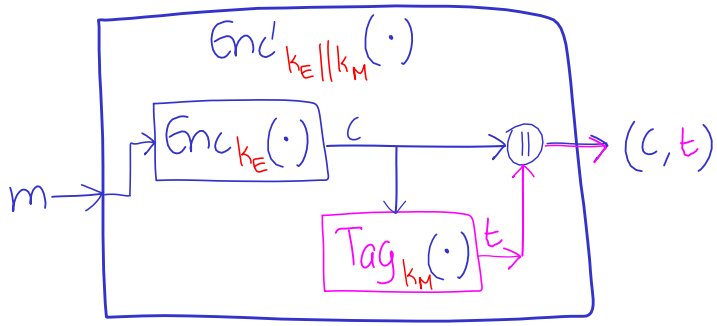
Construction 7 (MAC M for $\{0,1\}^n \implies$ MAC M′ for $\{0,1\}^{nB}$)



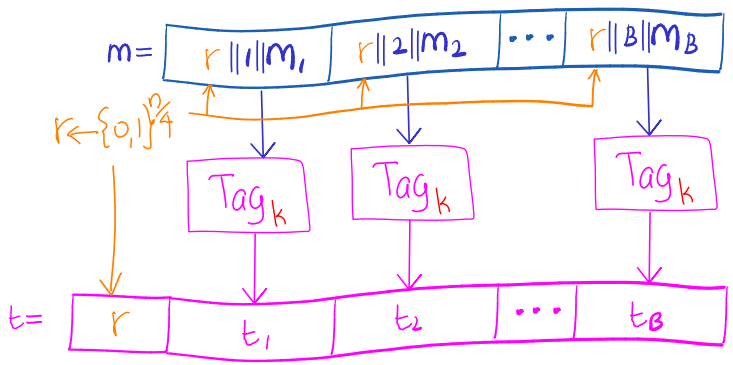
Theorem 5

If M is EU-CMA-secure MAC for $\{0,1\}^n$ then Construction 7 is EU-CMA-secure MAC for $\{0,1\}^{nB}$

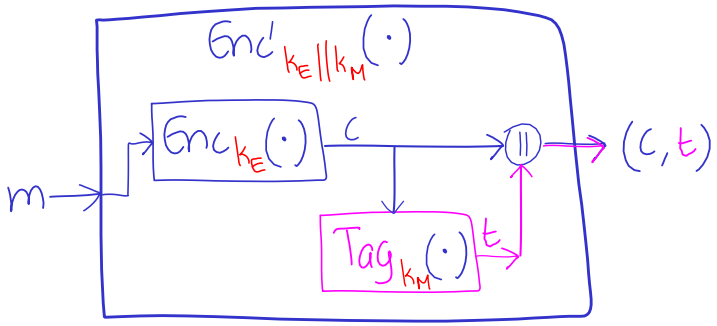
Recap/Next Lecture



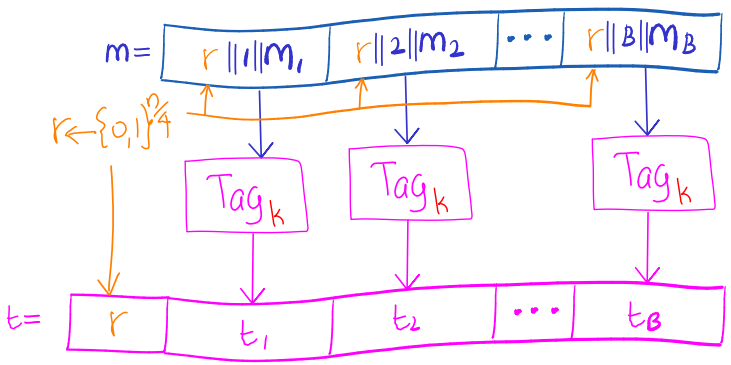
- Learnt how to construct CCA-secure SKE
  - Encrypt-then-Authenticate
  - ⚠ Encrypt-and-Authenticate and Authenticate-then-Encrypt insecure!
    - ⚠ Latter used in some configurations of TLS!
- Stronger notion than CCA: Authenticated encryption
- Domain extension for MAC



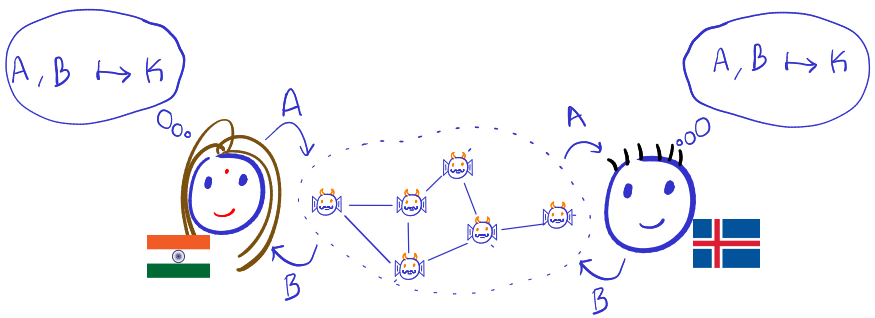
Recap/Next Lecture



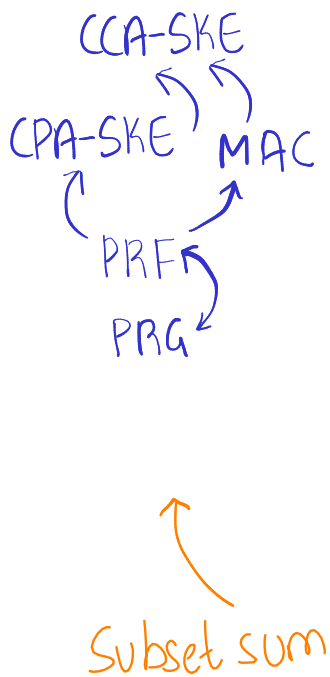
- Learnt how to construct CCA-secure SKE
  - Encrypt-then-Authenticate
  - ⚠ Encrypt-and-Authenticate and Authenticate-then-Encrypt insecure!
    - ⚠ Latter used in some configurations of TLS!
- Stronger notion than CCA: Authenticated encryption
- Domain extension for MAC



- Next lecture
  - We start public-key encryption module!
  - Basic group theory and number theory

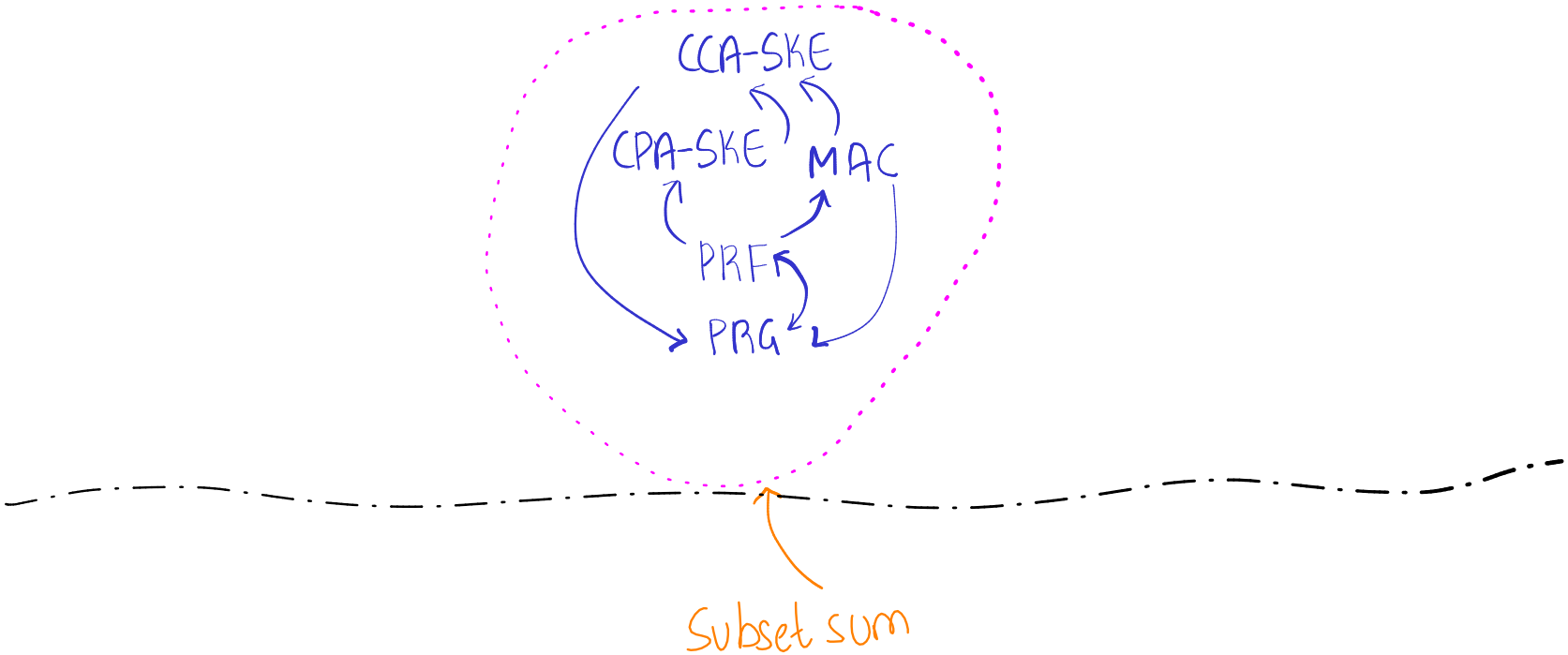


Next Module



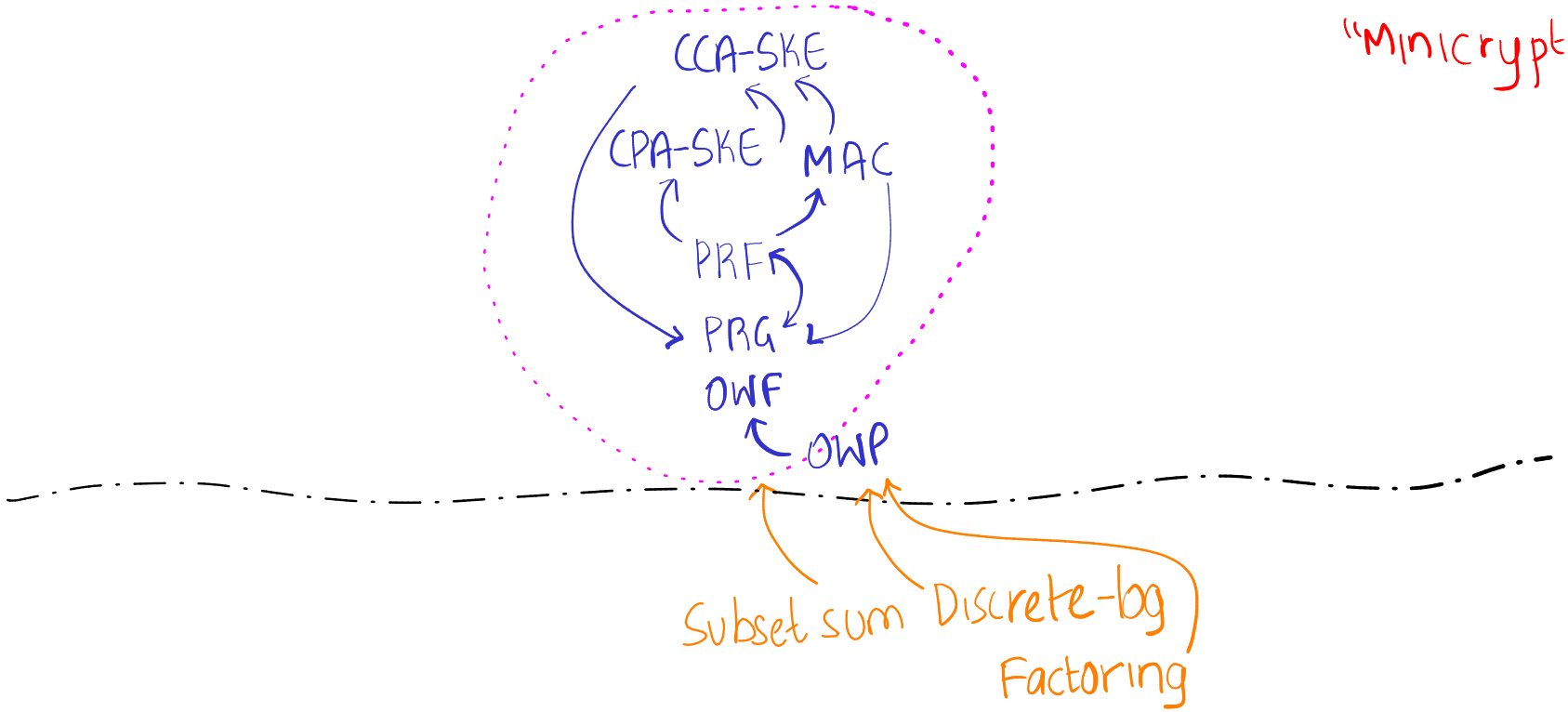
Good Luck for Mid-Sem!

Next Module



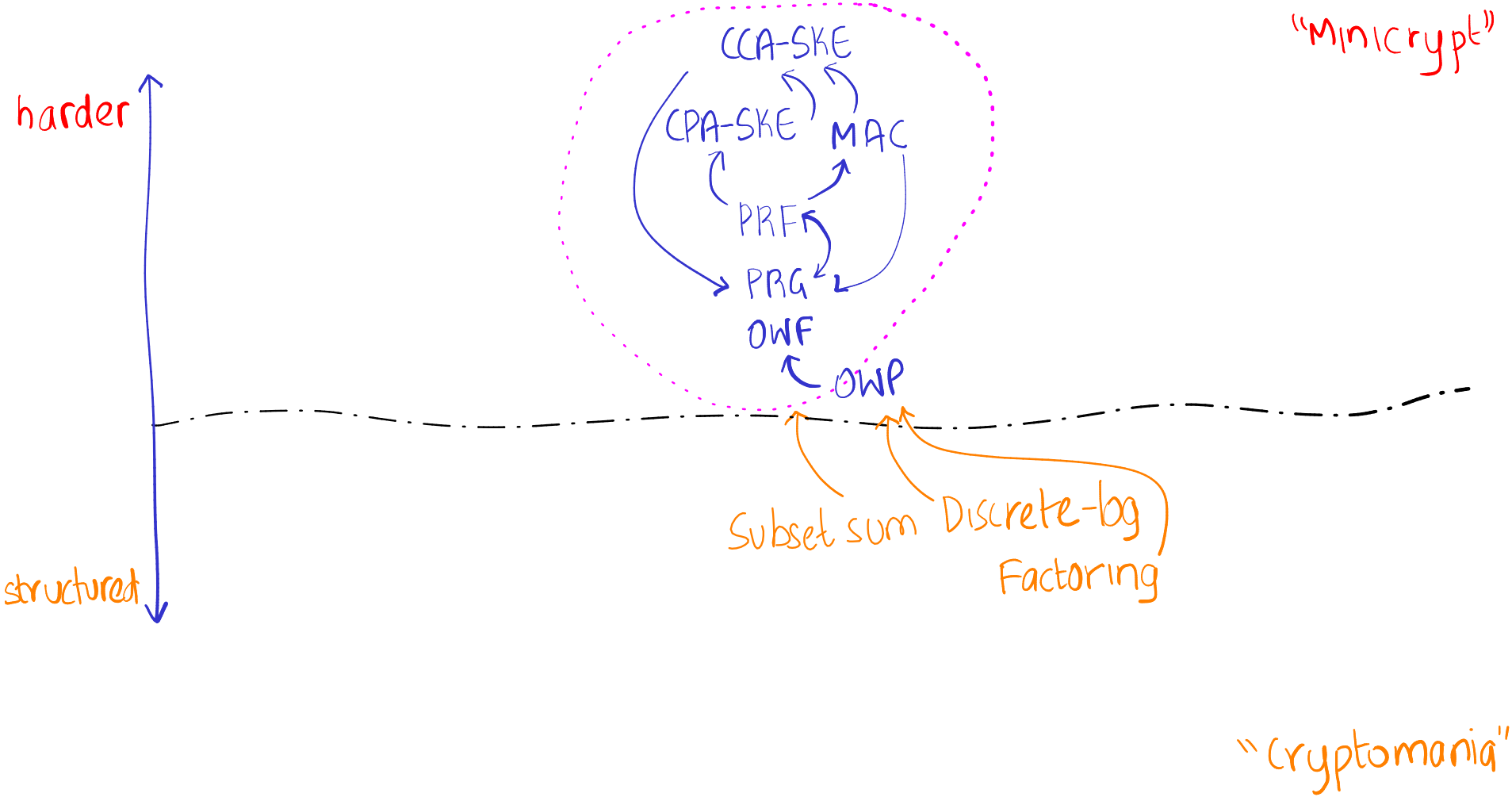
Good Luck for Mid-Sem!

Next Module



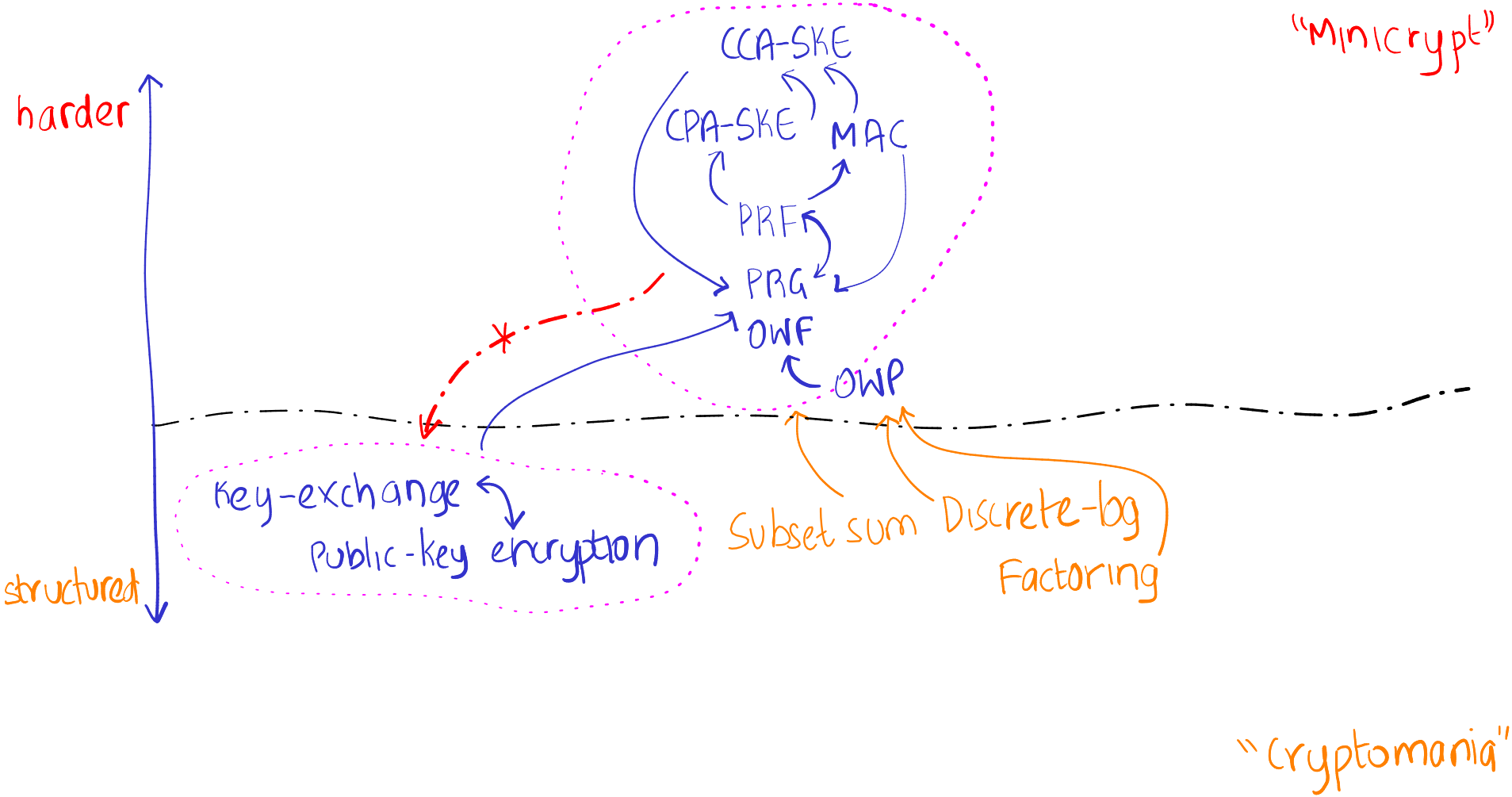
Good Luck for Mid-Sem!

Next Module



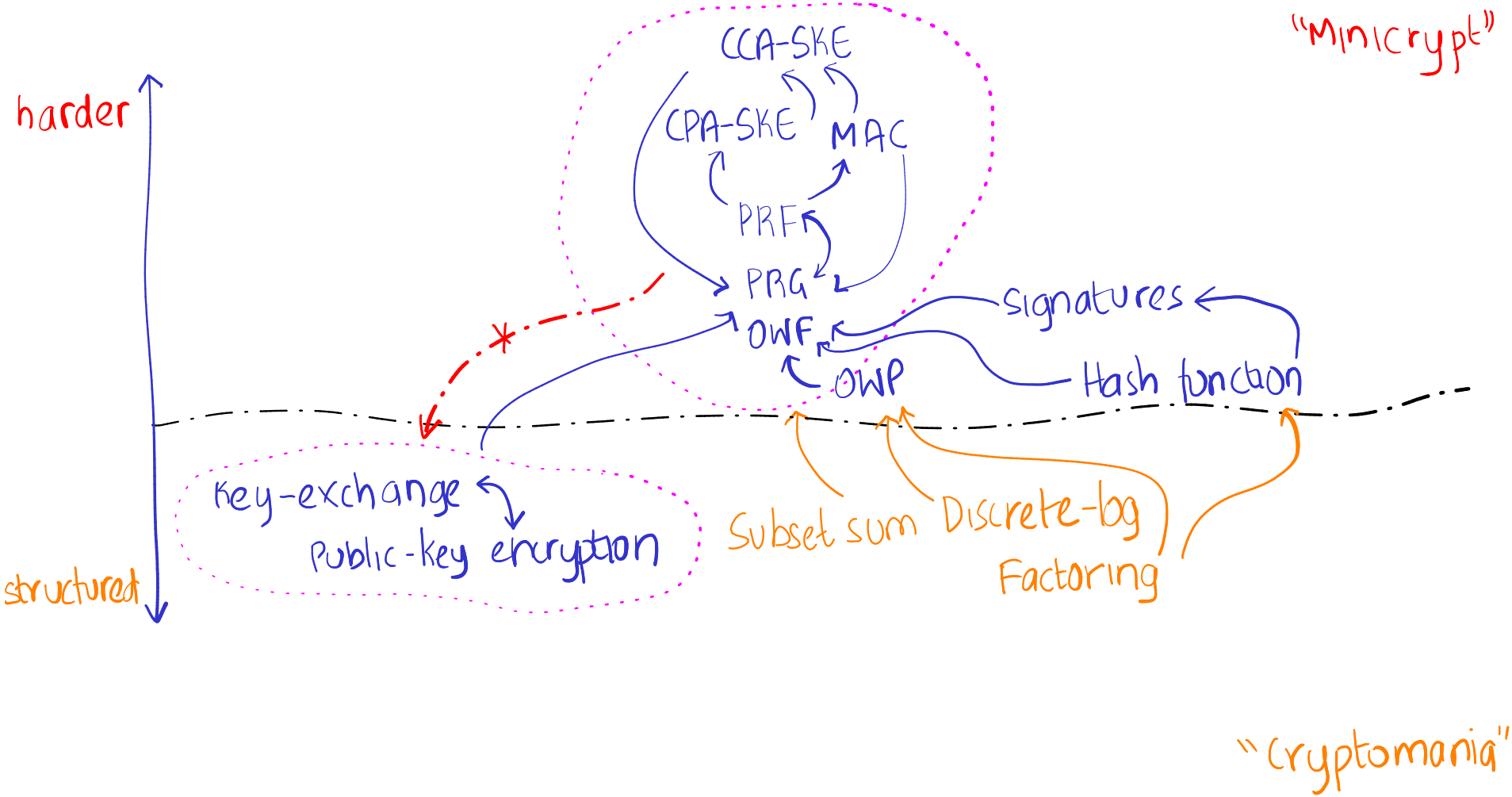
Good Luck for Mid-Sem!

Next Module



Good Luck for Mid-Sem!

Next Module



Good Luck for Mid-Sem!

Further Reading

1. You can find details of proof of Construction 1 in Theorem 4.6 in [KL14, §4.3.1].
2. [KL14, §5.3.1] contains discussion on our three attempts at construct CCA-secure PKE. You can also read more about authenticated encryption in [KL14, §5.2 and §5.3].
3. [KL14, §4.3.2] contains details on domain extension for MACs. In particular, proof Theorem 5 here corresponds to Theorem 4.8 in [KL14]


Jonathan Katz and Yehuda Lindell.
Introduction to Modern Cryptography (3rd ed.).
Chapman and Hall/CRC, 2014.