Authenticated Encryption
+
Message Authentication Codes

CS 284, Lecture 12, Autumn 2021

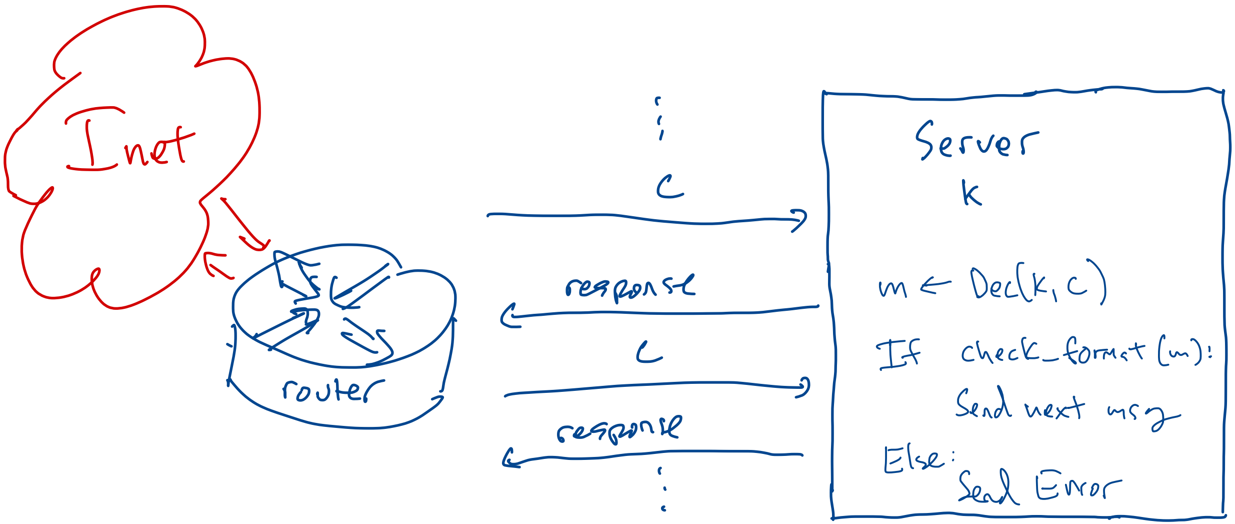
Outline

1. Chosen-Ciphertext Security for Encryption
2. Message Authentication Codes (MACs)
3. MAC Constructions from a Blockcipher
4. Combining Encryption with a MAC

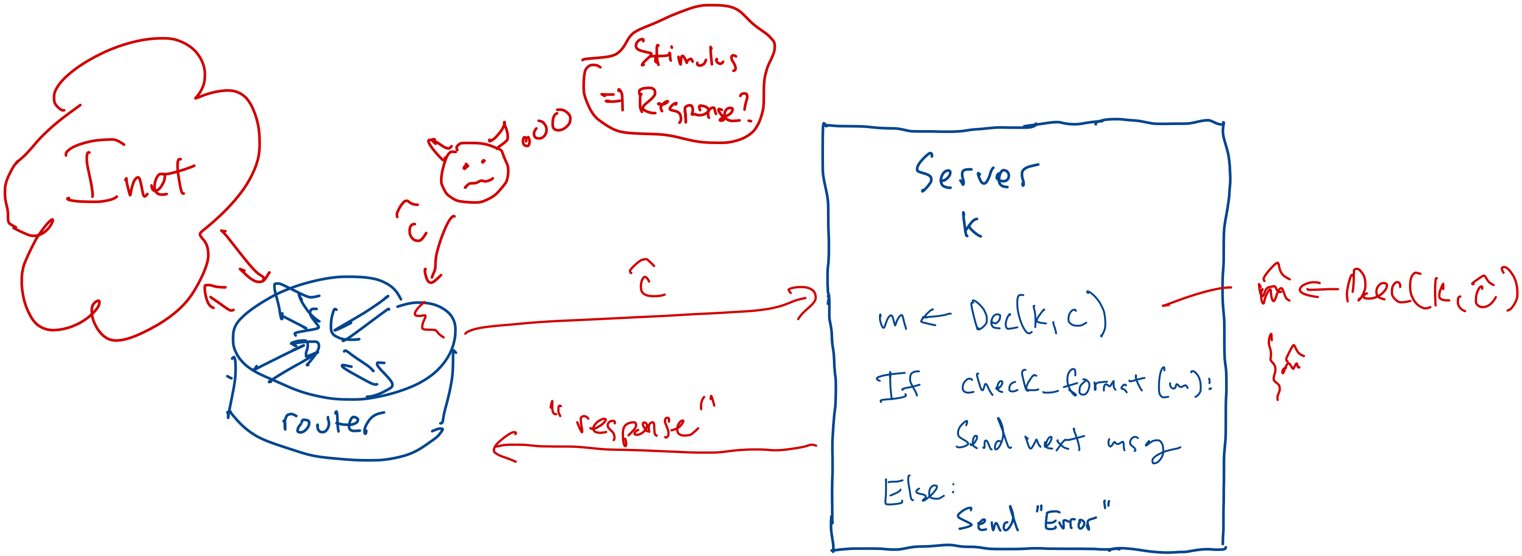
Outline

1. Chosen-Ciphertext Security for Encryption
2. Message Authentication Codes (MACs)
3. MAC Constructions from a Blockcipher
4. Combining Encryption with a MAC

A typical attack setting: Server connected to the Internet



A typical attack setting: Server connected to the Internet



→ "Chosen-Ciphertext Attacks" (CCA)

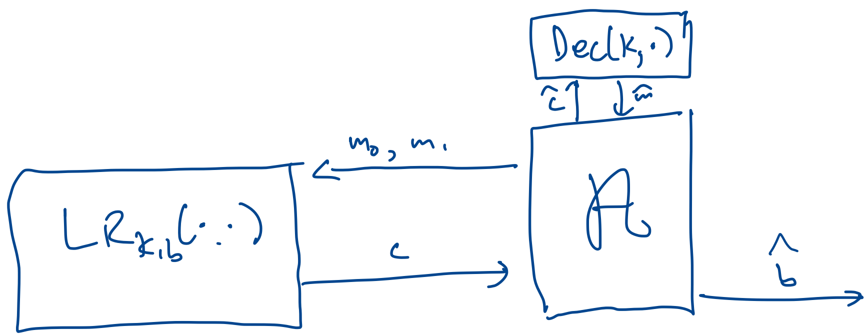
(plaintext) vs CPA

Chosen-Ciphertext Security: Plan for Definition

Goal: Definition for hiding all plaintext info against such attackers

→ Follow left/right idea from CPA.

→ Need to allow $\mathcal{A}$ ability to inject ciphertexts and observe responses.



* Like CPA, but give $\mathcal{A}$ a $Dec(k,\cdot)$ oracle!

CCA Definition

<u>Def</u> Let $\Pi = (\mathrm{Enc}, \mathrm{Dec})$ be a randomized encryption scheme with respect to $\mathcal{K}, \mathcal{M}, \mathcal{R}, \mathcal{C}$. Assume $\mathcal{M} \subseteq \{0,1\}^*$. Let $\mathcal{A}$ be an alogrithm. Define

$\mathrm{Expt}^{cca}_{\Pi}(\mathcal{A})$

Pick $k \in \mathcal{K}$, $b \in \{0,1\}$

Run $\mathcal{A}^{LR_{k,b}(\cdot,\cdot), \mathrm{Dec}(k,\cdot)}$. $\mathcal{A}$ halts with output $\hat{b}$.

If $\mathcal{A}$ ever queried $\mathrm{Dec}(k,\cdot)$ on some c previously output by LR: output 0.

If $\hat{b} = b$ output 1 Else output 0

Oracle $LR_{k,b}(m_0, m_1)$

If m_0, m_1 are not same length:
return $\perp$

Pick $r \in \mathcal{R}$

$c \leftarrow \mathrm{Enc}(k, m_b, r)$

Return c

The CCA advantage of $\mathcal{A}$ against Π is

$$\mathrm{Adv}^{cca}_{\Pi}(\mathcal{A}) = \left| \Pr[\mathrm{Expt}^{cca}_{\Pi}(\mathcal{A}) = 1] - \frac{1}{2} \right|$$

CCA Example

Let $\Pi = (\mathrm{Enc}, \mathrm{Dec})$ be defined by $\mathrm{Enc}(k, m, r) = (r, E(k,r) \oplus m)$,

$\mathrm{Dec}(k, (r, \gamma)) = E(k,r) \oplus \gamma \rightarrow 0^\ell$

$E(k,r)$

Claim There is an efficient $\mathcal{A}$ such that $\mathrm{Adv}^{cca}_{\Pi}(\mathcal{A}) = 1/2$.

LR $\quad$ Dec

$\mathcal{A}^{O_1(\cdot,\cdot),\, O_2(\cdot)}$

Fix $m_0, m_1 \in \mathcal{M}$, distinct, $m_0 \neq 0^\ell$

Query $(r, \gamma) \leftarrow O_1(m_0, m_1)$

Set $\gamma' = \gamma \oplus m_0$.

Query $m' \leftarrow O_2((r, \gamma'))$ // $\mathrm{Dec}(k, (r, \gamma'))$

If $m' = 0^\ell$ output 0, Else output 1.

$(r, \gamma) = (r, E(k,r) \oplus m_b)$

$\gamma' = \gamma \oplus m_0 = E(k,r) \oplus m_b \oplus m_0$

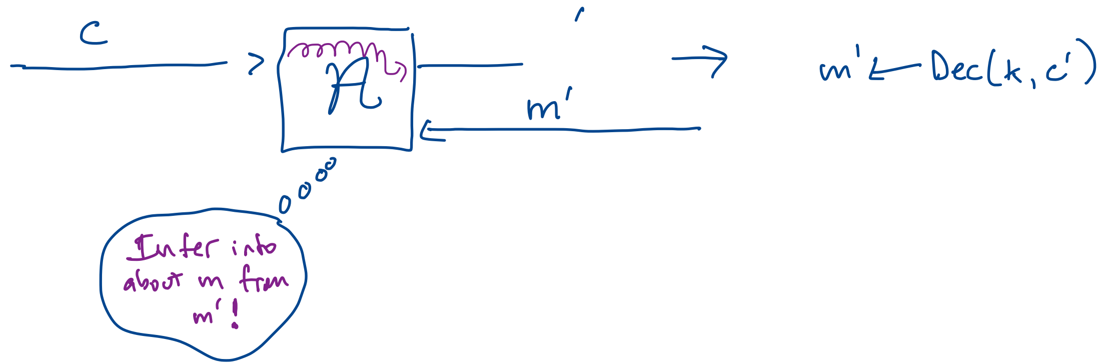
$$= \begin{cases} E(k,r) & b = 0 \\ E(k,r) \oplus m_0 \oplus m_1 & b = 1 \end{cases}$$

Claim: $b=0 \Rightarrow \mathrm{Dec}(k,(r,\gamma')) = 0^\ell$

Claim $b=1 \Rightarrow \mathrm{Dec}(k,(r,\gamma')) \neq 0^\ell$

CCA Security ≈ Security Against "Malleability"

$c \leftarrow Enc(k, m, r)$

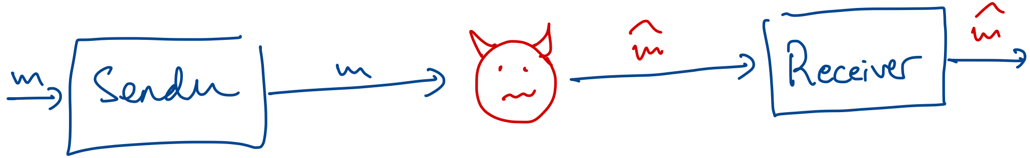


<u>Plan</u>: Make it "hard" for $\mathcal{A}$ to come up with <u>any</u> $c' \neq c$ without getting caught!

Outline

1. Chosen-Ciphertext Security for Encryption
2. Message Authentication Codes (MACs)
3. MAC Constructions from a Blockcipher
4. Combining Encryption with a MAC

Message Authenticity

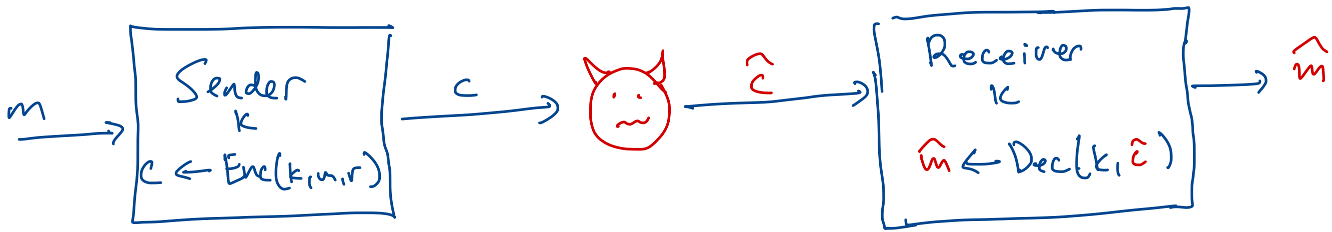


$\hat{m}$ really sent by Sender??

- Different from secrecy → Not trying to hide m from 

- Can ask for both (will later)

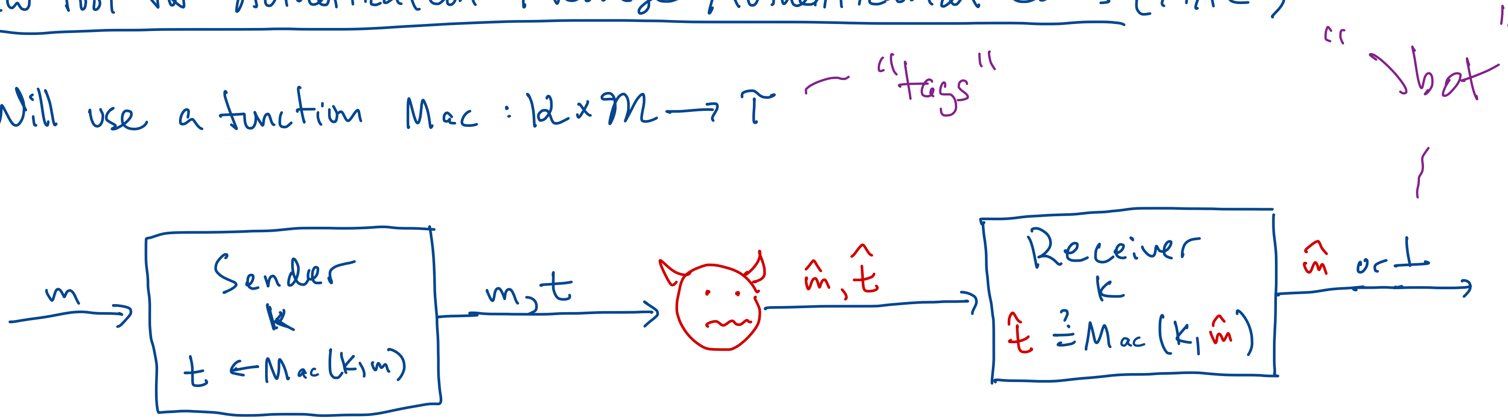
Encryption May not Provide Message Authenticity



-  may be able to get receiver to accept another $\hat{m}$ undetected
  - ↳ ex: Enc is OTP, then flipping bits of c will work

New tool for Authentication: Message Authentication Codes (MAC)

Will use a function $\mathrm{Mac} : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ "tags"

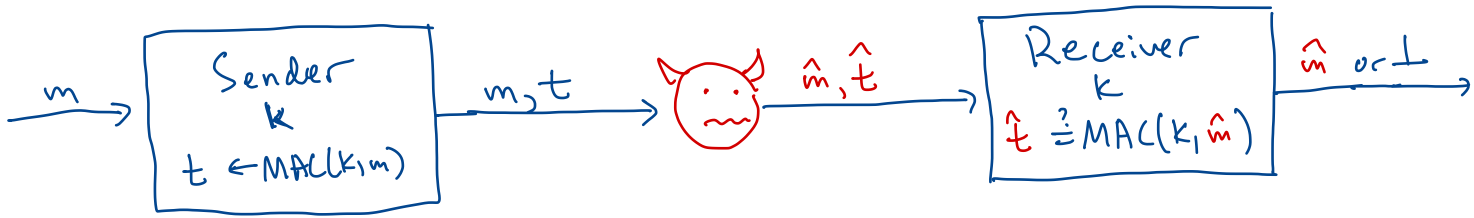


- **Goal:** Should be hard for to find correct $\hat{t}$ for some $\hat{m}$

- Give up on "replay attacks", where forwards some m, t that really did come from sender.

MAC Security: Ideas

Will use a function MAC: $\mathcal{K} \times \mathcal{M} \to \mathcal{T}$



1. tries to "forge" a $\hat{t}$ on some $\hat{m}$

2. gets to see many tags t on messages of its choice.

3. gets to see if receiver will accept many $\hat{m}, \hat{t}$ inputs, again of its choice.

MAC Security Definition

Def Let $\mathrm{Mac}: \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ and let $\mathcal{A}$ be an adversary. Define

$\mathrm{Expt}^{\mathrm{uf}}_{\mathrm{Mac}}(\mathcal{A})$

Pick $k \leftarrow \mathcal{K}$

Run $\mathcal{A}^{\mathrm{Mac}(k,\cdot),\, \mathrm{Vrfy}_k(\cdot,\cdot)}$ until it halts

If $\mathcal{A}$ ever queried 2nd oracle on $\hat{m}, \hat{t}$ such that

(1) $\hat{t} = \mathrm{Mac}(k, \hat{m})$, and

(2) $\hat{m}$ was never previously queried to $\mathrm{Mac}(k,\cdot)$ oracle

then output 1.

Else output 0.

Oracle $\mathrm{Vrfy}_k(\hat{m}, \hat{t})$

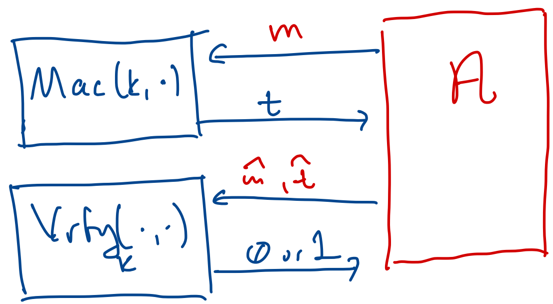
If $\mathrm{Mac}_k(\hat{m}) = \hat{t}$: return 1

Else: return 0

"uf" = "unforgeability"

Define $\mathrm{Adv}^{\mathrm{uf}}_{\mathrm{Mac}}(\mathcal{A}) = \Pr[\mathrm{Expt}^{\mathrm{uf}}_{\mathrm{Mac}}(\mathcal{A}) = 1]$.

MAC Security Definition Picture



To win: Send a correct message/tag pair to Vrfy with asking for a tag.

MAC Example

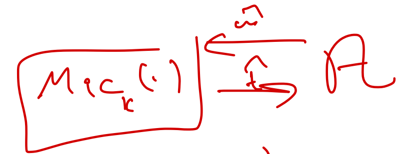
Define $Mac(k,m) = k \oplus m$.

$A^{Mac(k,\cdot),\, Vrfy_k(\cdot,\cdot)}$

$\hat{t} \leftarrow Mac(k, 0^\ell)$

$t' \leftarrow \hat{t} \oplus 1^\ell$ $= m'$

Query $Vrfy_k(1^\ell, t')$



m', t' forgery

$\hat{m} = 0^\ell \rightarrow \hat{t} = k$ (!)

$m' = 1^\ell$

$t' = \hat{t} \oplus 1^\ell$

forgery

- never queried m' to $Mac(k,\cdot)$
- $Mac(k, 1^\ell) = k \oplus 1^\ell = \hat{t} \oplus 1^\ell = t'$

MAC Example

Define $\mathrm{Mac}(k,m) = k \oplus m$.

$\mathcal{A}^{\mathrm{Mac}(k,\cdot),\, \mathrm{Vrfy}_k(\cdot,\cdot)}$

Query 0^ℓ to $\mathrm{Mac}(k,\cdot)$. Call result t.

Set $\hat{m} = 1^\ell$, $\hat{t} = t \oplus 1^\ell$.

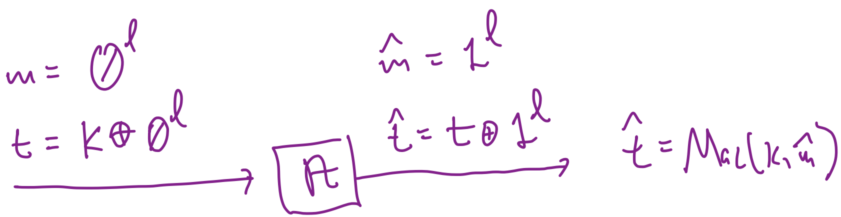
Query $(\hat{m}, \hat{t})$ to $\mathrm{Vrfy}_k(\cdot,\cdot)$

$\mathrm{Adv}^{uf}_{\mathrm{Mac}}(\mathcal{A}) = 1$ since:

$\mathrm{Mac}(k, \hat{m}) = k \oplus 1^\ell = t \oplus 1^\ell$

$(t = k \oplus 0^\ell = k)$.

and $\hat{m} = 1^\ell$ was never queried to Mac.



Outline

1. Chosen-Ciphertext Security for Encryption
2. Message Authentication Codes (MACs)
3. MAC Constructions from a Blockcipher
4. Combining Encryption with a MAC

Constructing MACs

$Mac_k(m) = AES(k, m)$ ✓

① In some sense, "A PRP is also a good MAC, as long as its output size is not too small"

② AES is a good PRP, therefore also a good MAC (128-bit output is enough). But it only takes 128-bit inputs.

⇒ So we build MACs for larger messages from AES.

MACs for Longer Messages

Naive attempt for two blocks:

$$F(k, m_1 \| m_2) = \underbrace{AES(k, m_1)}_{``t_1"} \,\|\, \underbrace{AES(k, m_2)}_{``t_2"}$$

$$t_1 \;\|\; t_2$$

$$t_2 \;\|\; t_1$$

is tag for $m_2 \| m_1$

$t_1 \| t_1$

$m_1 \| m_1$

MACs for Longer Messages

Naive attempt for two blocks:

$$F(k, m_1 \| m_2) = AES(k, m_1) \| AES(k, m_2)$$

Attack $\mathcal{A}$ with $\mathrm{Adv}^{uf}_{F}(\mathcal{A}) = 1$:

Query $0^{128} \| 1^{128}$ to $\mathrm{Mac}(k, \cdot)$. Get t.

Parse t as $t_0 \| t_1$.

Query $\hat{m} = 0^{128} \| 0^{128}$, $\hat{t} = t_0 \| t_0$ to $\mathrm{Vrfy}_k(\cdot, \cdot)$

Another Example

$$F(k_1, m_1 \| m_2) = AES(k_1, m_1) \oplus AES(k_1, m_2)$$

$m_1 = m_2$, $t = 0^{128}$ works!

A Practical MAC: CBC-MAC

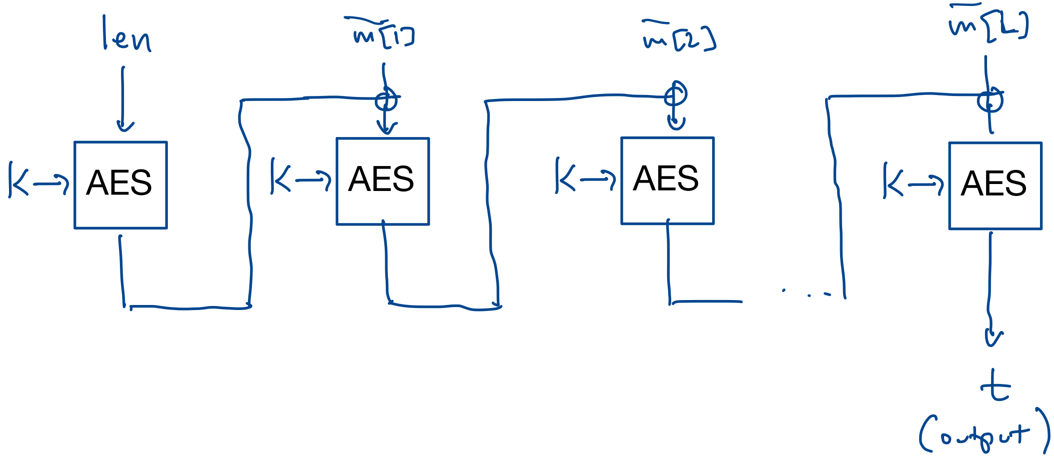
$pad_{CBC}(m) = \boxed{m}\boxed{1}\boxed{0 \cdots 0}$

Key $k \in \{0,1\}^{128}$

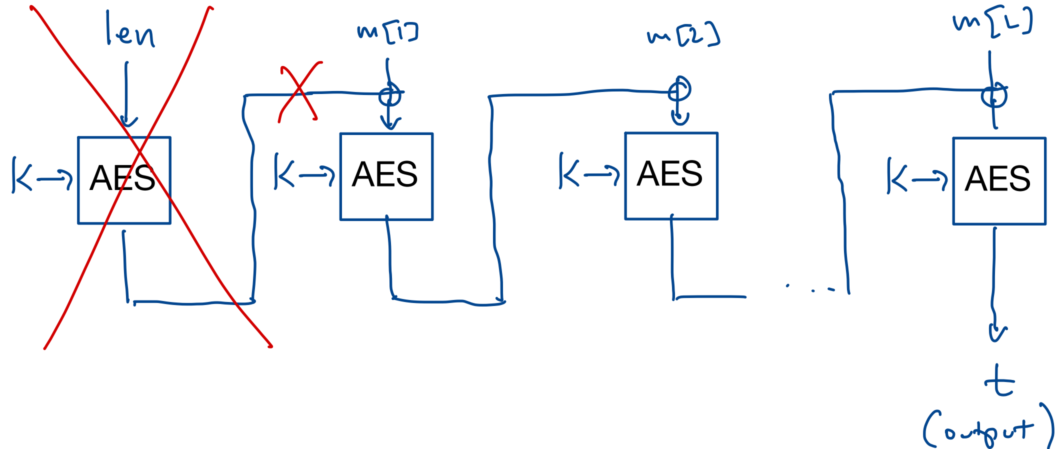
① Let $len \in \{0,1\}^{128}$ be the length of m, encoded as a 128-bit block.

② $\bar{m} \leftarrow pad_{CBC}(m)$ // pad to multiple of 128

③ parse $\bar{m}$ into blocks $m[1] \ldots m[L]$



CBC-MAC without length?

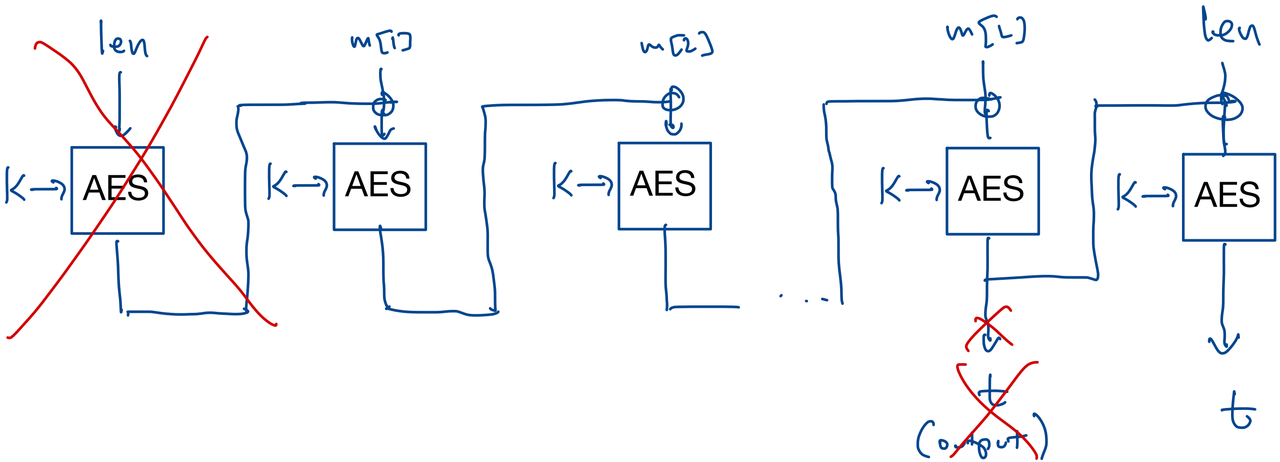


Insecure! Fun exercise.

(Hint: Query Mac(k, 0^{127}) set t

Now find a two-block message that will verify with same t.

CBC-MAC with length at end?



Insecure! A bit harder, but still simple.

Outline

1. Chosen-Ciphertext Security for Encryption
2. Message Authentication Codes (MACs)
3. MAC Constructions from a Blockcipher
4. Combining Encryption with a MAC

Plan For Achieving CCA Security

1. Build a CPA-secure encryption scheme (Enc, Dec)

2. Build a secure MAC Mac

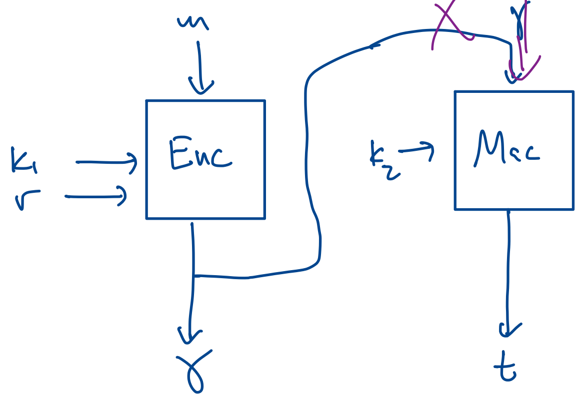
3. Build encryption scheme (Enc', Dec') that runs both (Enc, Dec) + Mac.

Combining CPA-Secure Encryption and a MAC: "Encrypt-then-Mac"

"Encrypt and Mac"

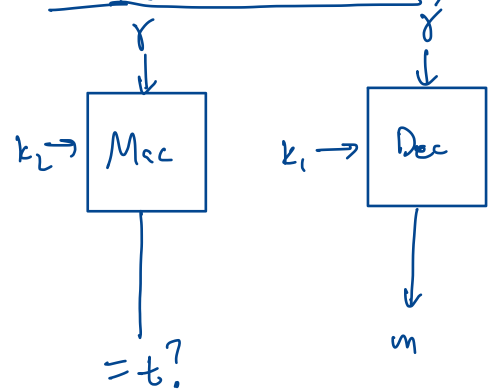
Given $\Pi = (\mathrm{Enc}, \mathrm{Dec})$ and Mac, build new encryption scheme $\Pi' = (\mathrm{Enc}', \mathrm{Dec}')$:

$\underline{\mathrm{Enc}'(k_1 \| k_2, m, r)}$



output $c = (\gamma, t)$

$\underline{\mathrm{Dec}'(k_1 \| k_2, (\gamma, t))}$



If t correct, output m

__Thm__ (CS 388): If Π is "CPA secure" and Mac is "secure" then Π' is "CCA secure".