COL759:
CRYPTOGRAPHY AND COMPUTER SECURITY

2022-23 (SEMESTER 1)

LECTURE 24: ENCRYPTION SECURITY AGAINST ACTIVE ATTACKS

REVIEW OF LAST LECTURE

(Enc, Dec) secure against active attacks

Confidentiality + Integrity

Semantic Security

After seeing many ct,
Adversary cannot
learn msg in challenge ct

Integrity of plaintext

After seeing many ct,
Adversary cannot
produce enc. of new msg

Integrity of ciphertext

After seeing many ct,
Adversary cannot
produce new valid ct

REVIEW OF LAST LECTURE

Semantic security guarantees confidentiality

$(\mathrm{Enc}_{\mathrm{ro}}, \mathrm{Dec}_{\mathrm{ro}})$

MAC guarantees integrity of message

(Sign , Verify)

How to combine both to get security against active attacks?

REVIEW OF LAST LECTURE

MAC and Encrypt

$\mathsf{Enc}(m, (k_{\text{ro}}, k_{\text{mac}}))$

$\mathsf{ct}_{\text{ro}} \leftarrow \mathsf{Enc}_{\text{ro}}(m, k_{\text{ro}})$

$\sigma \leftarrow \mathsf{Sign}(m, k_{\text{mac}})$

Output $(\sigma, \mathsf{ct}_{\text{ro}})$

Not semantically secure!

MAC then Encrypt

$\mathsf{Enc}(m, (k_{\text{ro}}, k_{\text{mac}}))$

$\sigma \leftarrow \mathsf{Sign}(m, k_{\text{mac}})$

$\mathsf{ct}_{\text{ro}} \leftarrow \mathsf{Enc}_{\text{ro}}(m \mid\mid \sigma, k_{\text{ro}})$

Output ct_{ro}

Semantically secure ✓

Plaintext integrity ✓ (to prove)

Ciphertext integrity ??

Encrypt then MAC

$\mathsf{Enc}(m, (k_{\text{ro}}, k_{\text{mac}}))$

$\mathsf{ct}_{\text{ro}} \leftarrow \mathsf{Enc}_{\text{ro}}(m, k_{\text{ro}})$

$\sigma \leftarrow \mathsf{Sign}(\mathsf{ct}_{\text{ro}}, k_{\text{mac}})$

Output $(\sigma, \mathsf{ct}_{\text{ro}})$

Semantically secure ✓

Plaintext integrity ✓

Ciphertext integrity ✓ (to prove)

IS SEMANTIC SECURITY + PLAINTEXT INTEGRITY GOOD ENOUGH?

SSL 3.0 uses MAC-then-Encrypt (CBC-mode encryption + ECBC-MAC)

Construction is semantically secure and satisfies plaintext int.

But there exist real world attacks!

Question from last class:

The attack on SSL3.0 is because of the implementation detail.
Is there any practical implication of an enc. scheme not satisfying ciphertext integrity but satisfying plaintext integrity?

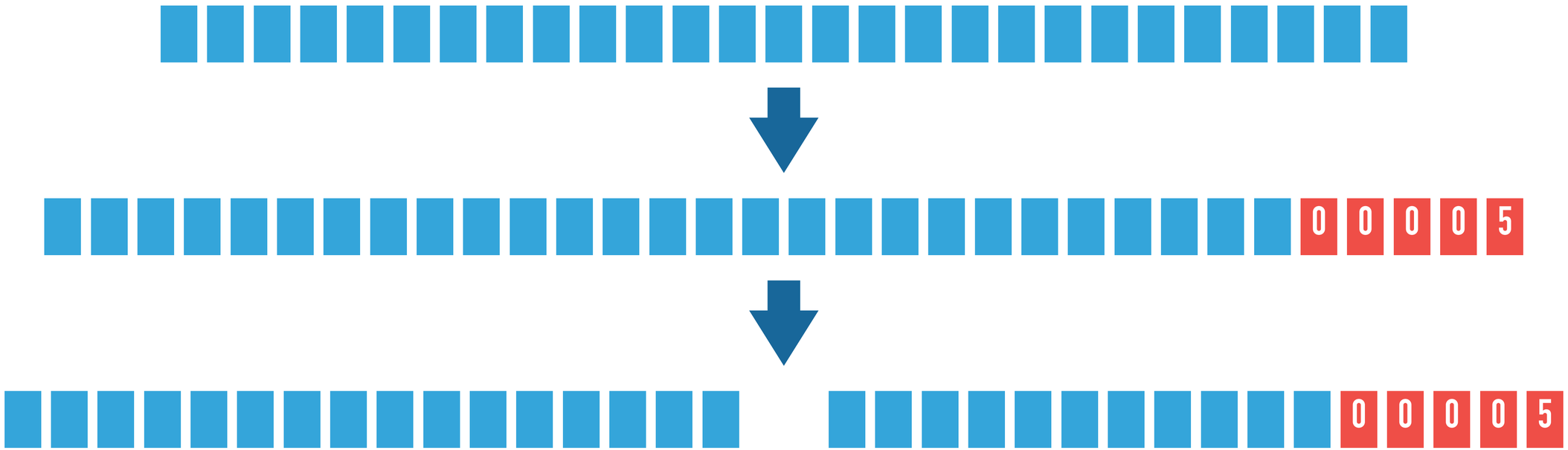
Suppose semantic security + ptxt. integrity was the 'gold standard' defn. for encryption security.

1. Proposed a construction
2. Proved sem. security, plaintext integrity
3. Implemented the construction (correctly)

Attack on implementation!

Problem with implementation, or with definition?

CBC-MODE ENCRYPTION



Question from last class:

How is a message converted to binary? When we say message m in binary, is 001 and 01 different?

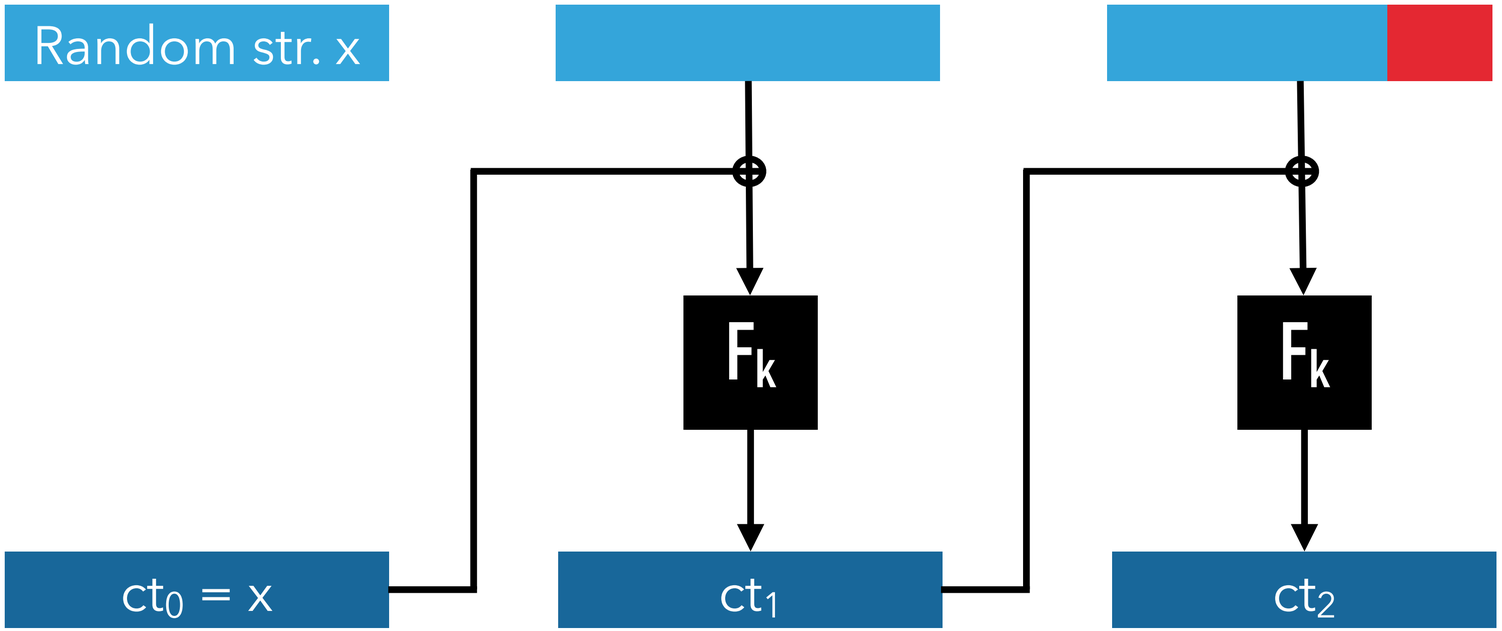
It was not intuitive for me to pad on the right. I feel that padding on left with zeros is better

Encryption scheme defines the message space.

Here, the message space was a sequence of bytes

Given sequence of bytes, enc. scheme defines how to pad it. Padding attack works for both left and right padding.

CBC-MODE ENCRYPTION



Attack 1: Given enc of $m = (m_1, m_2, \ldots, m_\ell)$, can produce encryption of (m_i)

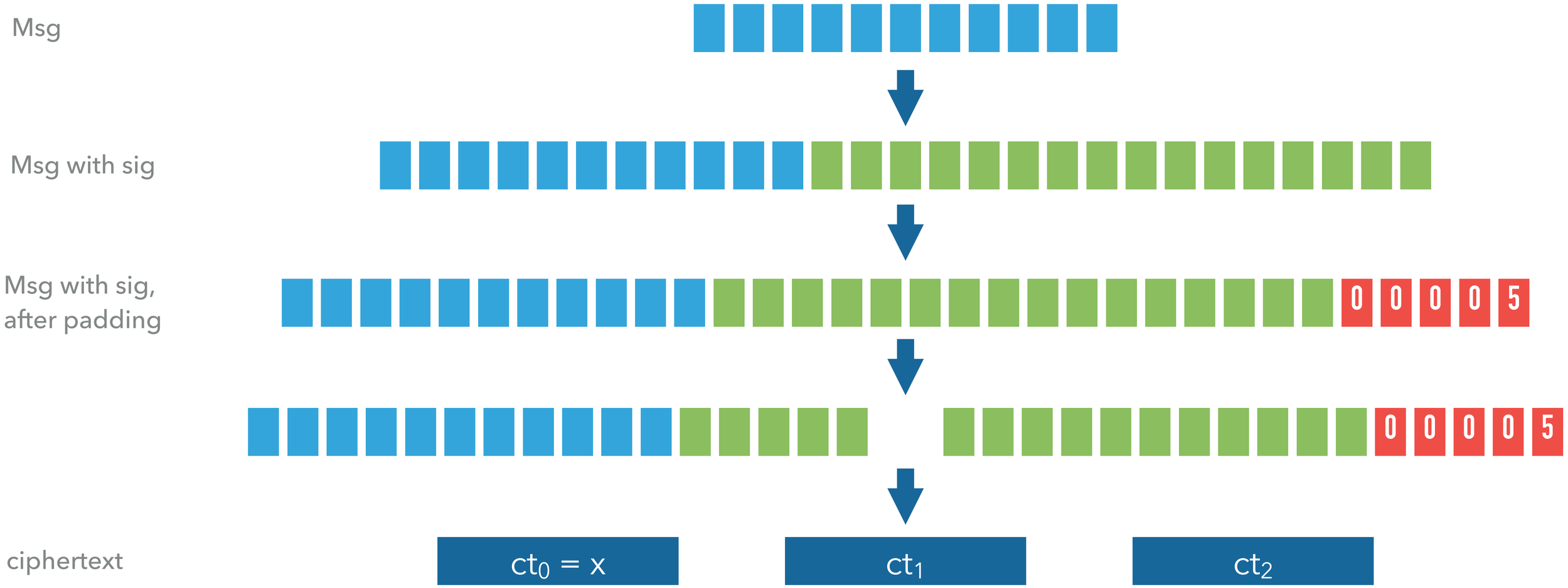
Hint: Use ct_{i-1} and ct_i

Attack 2: Given enc of $m = (m_1, m_2, \ldots, m_\ell)$, can produce encryption of $(m_\ell \oplus \alpha)$

Hint: Tamper ct_{i-1}, output (ct'_{i-1}, ct_{i-1})

CBC-MODE ENC. + MAC USING 'MAC-THEN-ENCRYPT'

MAC scheme outputs 16 byte signatures



CBC-MODE ENC. + MAC USING 'MAC-THEN-ENCRYPT'

Decryption:

Decrypt ciphertext using enc/dec key

Check that padding is valid. If not, output "bad padding"

Extract message and signature

Check validity of signature. If verification fails, output "bad signature"

Output message if both checks pass.

THE PADDING ORACLE ATTACK

Adversary intercepts the following ct:

ct_0 ct_1 ct_2

Generates a few invalid ciphertexts from ct, sends to server.

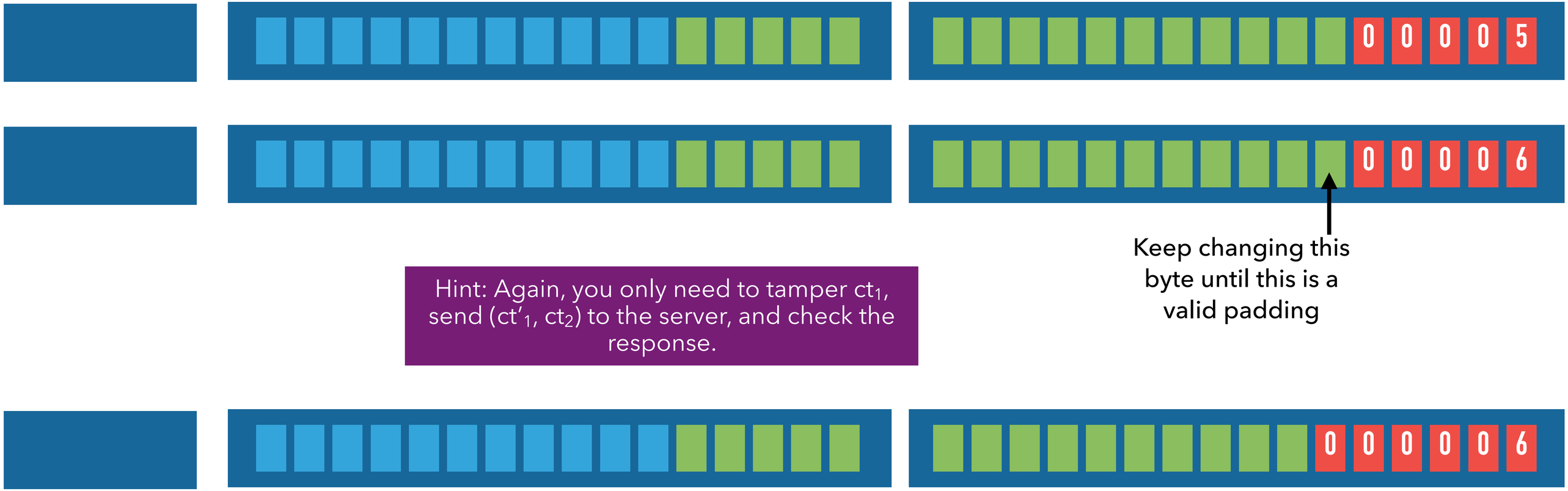
Server either sends “bad padding”, “bad signature” or sends nothing.

In at most 16 queries, adversary learns padding length.

Hint: Use ct_1 and ct_2 in the example above. To alter the padding, attacker should edit the message underlying ct_2, and therefore it needs to tamper ct_1, and send (ct'_1, ct_2) to the server, and check the response.

THE PADDING ORACLE ATTACK

In at most 256 queries, it learns last non-padding byte.



Valid padding, bad signature. Decryption outputs different error messages

THE PADDING ORACLE ATTACK

After learning the last block, attacker drops the last block.
In at most 256 queries, it learns last byte of second last block.

Keep changing this byte until this is a valid padding

Hint: Suppose you have figured out the last $\ell - i$ blocks. Use $(\mathrm{ct}_{i-1}, \mathrm{ct}_i)$, repeatedly tamper ct_{i-1}, send $(\mathrm{ct}'_{i-1}, \mathrm{ct}_i)$ to server and see error message.

Valid padding, bad signature. Decryption outputs different error messages

THE PADDING ORACLE ATTACK: SUMMARY

- Adversary can learn l byte message using ~ 256.l queries to server
- Adversary can modify the contents within the ciphertext
- After modification, ct is invalid, but different types of decryption errors
- The type of decryption error tells whether padding is valid or not.
- Knowing whether padding is valid or not, it can learn full message.

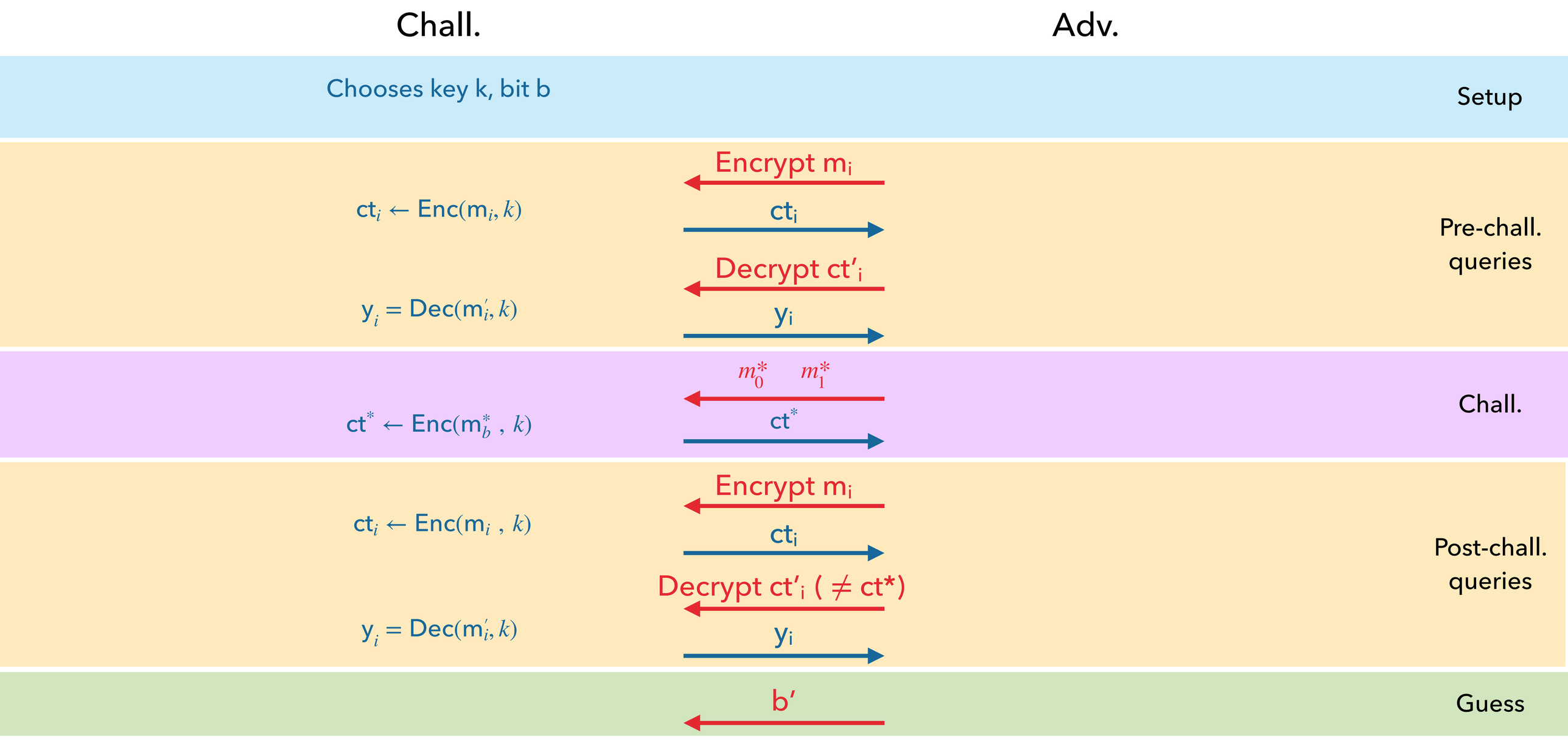
IS SEMANTIC SECURITY + PLAINTEXT INTEGRITY GOOD ENOUGH?

No! Neither semantic security nor plaintext integrity
address information leakage due to decryption queries.

Q1: How to formally capture
information leakage due to decryption queries?

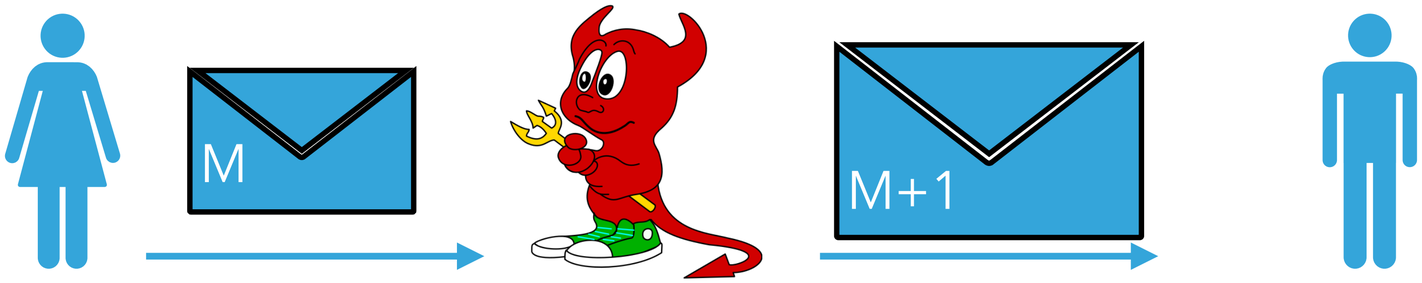
Q2: Does semantic security + ciphertext integrity
guarantee no information leakage due to decryption queries?

SECURITY AGAINST CHOSEN CIPHERTEXT ATTACKS



SECURITY AGAINST CHOSEN CIPHERTEXT ATTACKS

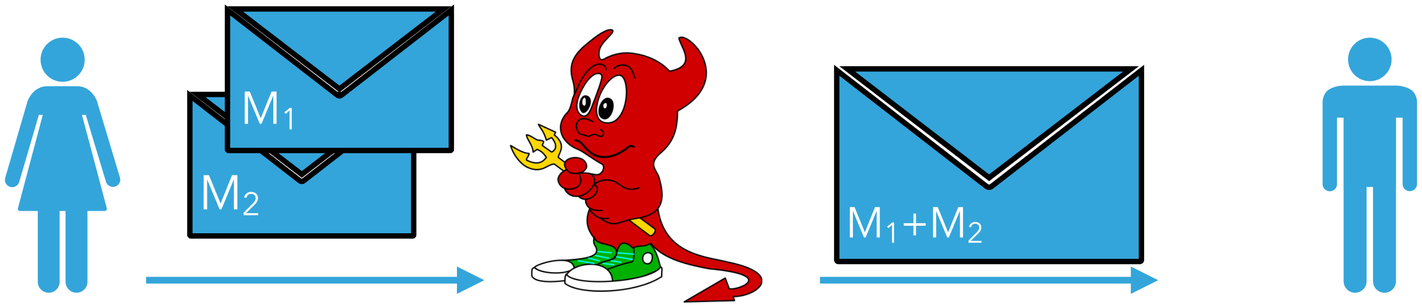
Does security against chosen ciphertext attacks prevent malleability attacks?



Hint: Given an adversary that can launch the above attack, it can be used to break the chosen ciphertext security. The reduction algorithm sends 0,1 as the challenge messages, and receives ct*. Next, it uses the attacker to produce either an encryption of 1 or encryption of 2. Send this as a decryption query to the CCA challenger, that reveals which message was encrypted by the challenger.

SECURITY AGAINST CHOSEN CIPHERTEXT ATTACKS

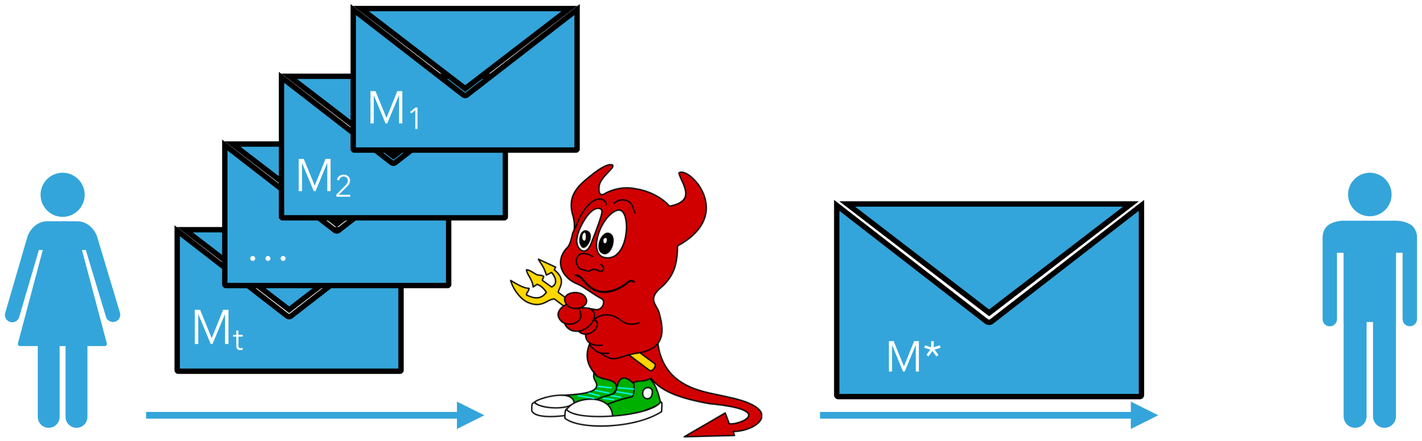
Does security against chosen ciphertext attacks prevent malleability attacks?



Hint: Given an adversary that can launch the above attack, it can be used to break the chosen ciphertext security. The reduction algorithm sends 0,1 as the challenge messages, and receives ct*. Next, it queries for encryption of '1'. It can then combine the challenge ciphertext, and encryption of 1, to get either an encryption of 1 or encryption of 2. Send this as a decryption query to the CCA challenger, that reveals which message was encrypted by the challenger.

SECURITY AGAINST CHOSEN CIPHERTEXT ATTACKS

Does security against chosen ciphertext attacks imply plaintext/ciphertext integrity?



Hint: using a strong PRP, we can construct a CCA secure scheme where every string is the encryption of some message. As a result, if attacker just sends a random message, that'll be the encryption of some message. The adversary doesn't even need any queries, but it breaks plaintext integrity.

Q1: How to formally capture
information leakage due to decryption queries?

Ans: Security against Chosen Ciphertext Attacks

Q2: Does semantic security + ciphertext integrity
guarantee no information leakage due to decryption queries?

Ans: Yes! Semantic Security + ciphertext integrity implies
Security against Chosen Ciphertext Attacks

NEXT LECTURE:

Semantic security and ciphertext integrity implies security against chosen ciphertext attacks

Encrypt-then-MAC achieves semantic security and ciphertext integrity

Summary of authenticated encryption and symmetric key encryption.

THANK YOU!