Cryptography Basics – Block ciphers and modes

ECEN 4133
Jan 26, 2021

Alternative to stream cipher: Block Ciphers

Today's most common block cipher:

AES (**Advanced Encryption Standard**)

- Designed by NIST competition, long public comment/discussion period
- Widely believed to be secure, *but we don't know how to prove it*
- Variable **key size** and **block size**
- We'll use 128-bit key, 128-bit block (also exist 192-bit and 256-bit versions)
- Ten **rounds**: Split **k** into ten **subkeys**, performs set of operations ten times, each with diff. subkey

Each AES round

128-bits in, 128-bit sub-key, 128-bits out

Four steps:
picture as operations on a 4x4 grid of 8-bit values

1. Non-linear step
   Run each byte thru a non-linear function (lookup table)

2. Shift step
   Circular-shift each row: **i**th row shifted by **i** (0-3)

3. Linear-mix step
   Treat each column as a 4-vector; multiply by a constant invertible matrix

4. Key-addition step
   XOR each byte with corresponding byte of round subkey

To decrypt, just undo the steps, in reverse order

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

Remaining problem: How to encrypt longer messages?

Padding

Can only encrypt in units of cipher blocksize, but message might not be multiples of blocksize

Solution: Add padding to end of message

Must be able to recognize and remove padding afterward

Common approach:
Add **n** bytes that have value **n**

[Caution: What if message ends at a block boundary?]

Cipher modes

We know how to encrypt one block,
but what about multiblock messages?

Different methods, called "cipher modes"

Straightforward (but bad) approach:

ECB mode (**encrypted codebook**)

Just encrypt each block independently

$\mathbf{C}_i := \boldsymbol{E}_k(\mathbf{P}_i)$

[Disadvantages? Solutions?]

Cipher modes

We know how to encrypt one block,
but what about multiblock messages?

Different methods, called "cipher modes"

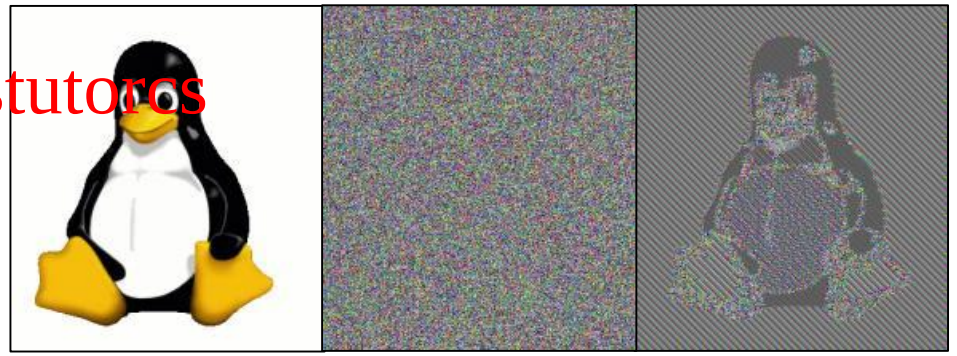
Straightforward (but bad) approach:

ECB mode (**encrypted codebook**)

Just encrypt each block independently

$C_i := E_k(P_i)$

[Disadvantages? Solutions?]



Plaintext Pseudorandom ECB mode

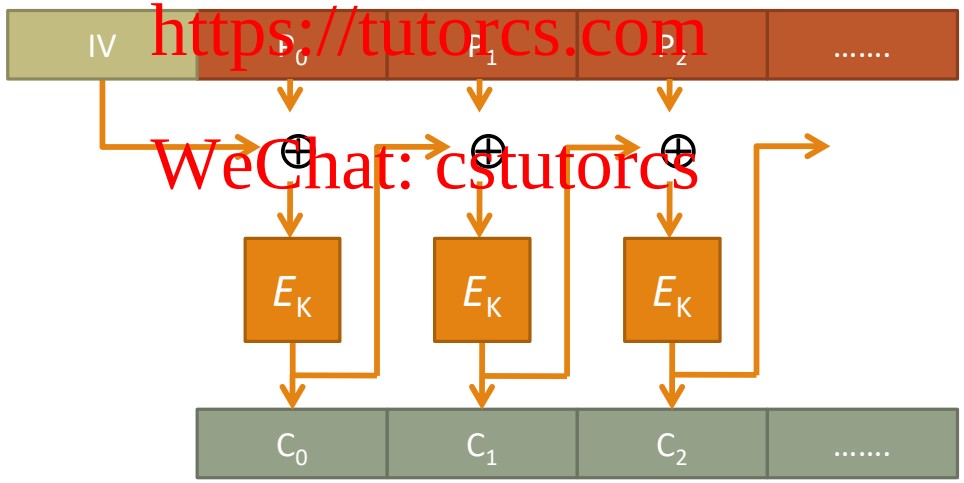
Better (and common): CBC mode (cipher-block chaining)

For each block $\mathbf{P}_i$:

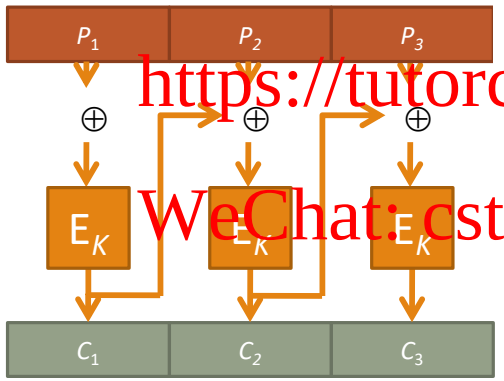
$\mathbf{C}_i := \boldsymbol{E}_k(\mathbf{P}_i \text{ xor } \mathbf{C}_{i-1})$

(Need to generate random IV ("Initialization Vector") at start)

[Pros and cons?]

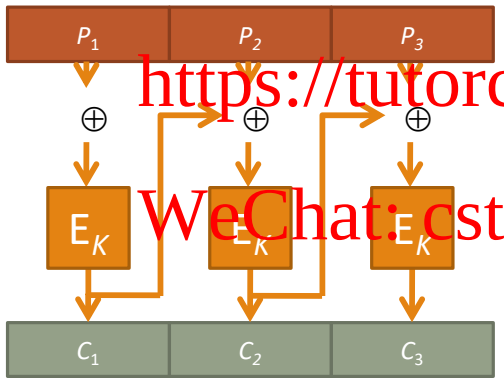


Padding oracle attack



P_1 = TT TT TT TT
P_2 = 68 65 6c 6c
P_3 = 6f ?? ?? ??

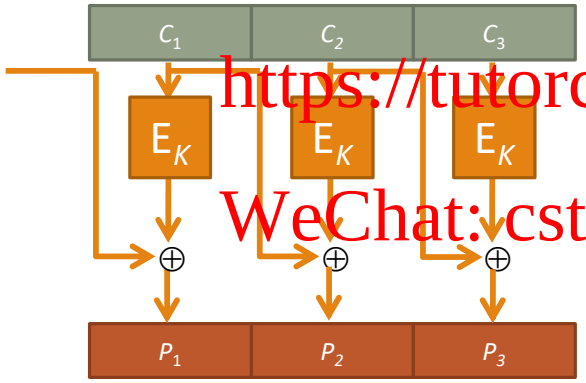
Padding oracle attack



P_1 = TT TT TT TT
P_2 = 68 65 6c 6c
P_3 = 6f 03 03 03

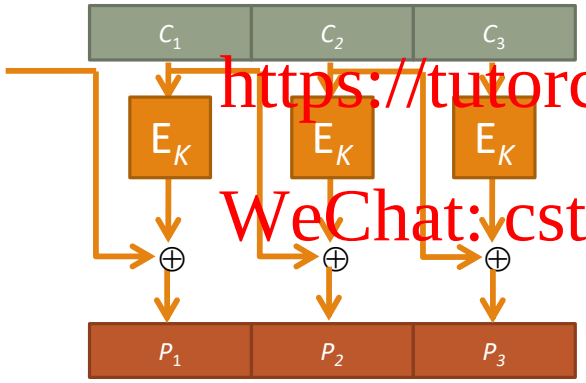
Padding oracle attack

Decryption:



$$P_1 = D(C1) \oplus IV$$
$$P_2 = D(C2) \oplus C1$$
$$P_3 = D(C3) \oplus C2$$

Padding oracle attack



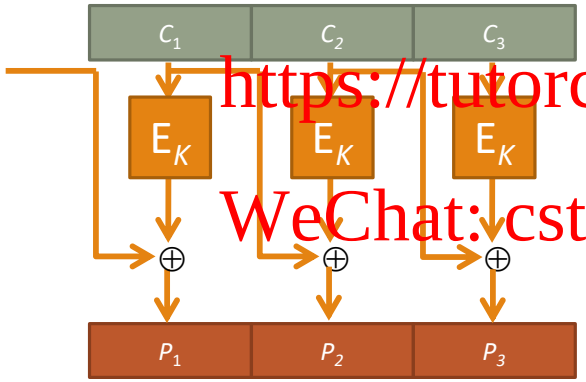
$D(C_3)$	5b	d8	99	ee
C_2	34	db	9a	ed
P_3	6f	03	03	03

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Padding oracle attack



$D(C_3)$	5b	d8	99	ee
C_2	34	db	9a	00
P_3	6f	03	03	ee

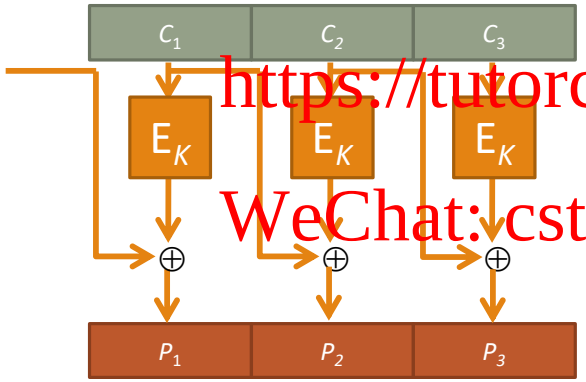
PADDING ERROR

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Padding oracle attack



$D(C_3)$	5b	d8	99	ee
C_2	34	db	9a	01
P_3	6f	03	03	ef

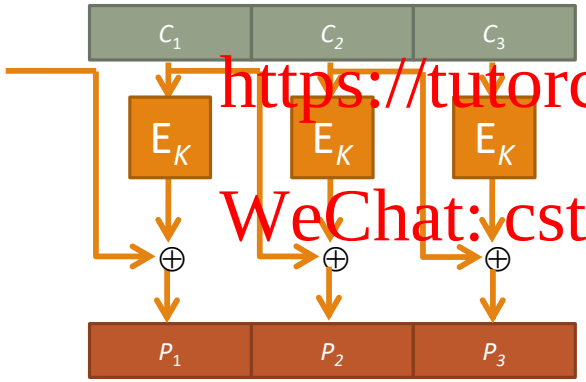
PADDING ERROR

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Padding oracle attack

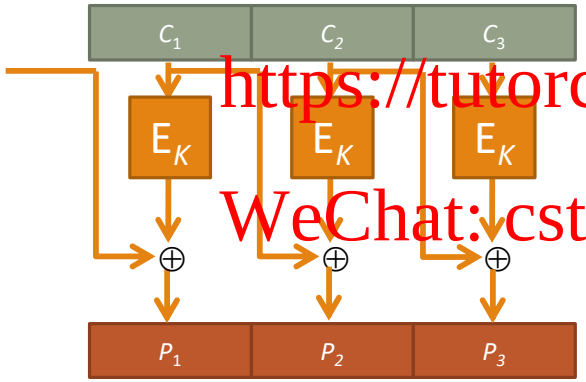


$D(C_3)$ 5b d8 99 ee
C_2 34 db 9a 02
P_3 6f 03 03 ec

PADDING ERROR

$P_1 = D(C1) \oplus IV$
$P_2 = D(C2) \oplus C1$
$P_3 = D(C3) \oplus C2$

Padding oracle attack

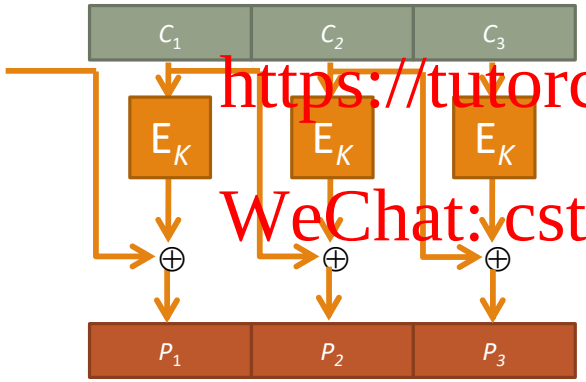


$D(C_3)$ 5b d8 99 ee
C_2 34 db 9a . .
P_3 6f 03 03 . .

PADDING ERROR

$P_1 = D(C1) \oplus IV$
$P_2 = D(C2) \oplus C1$
$P_3 = D(C3) \oplus C2$

Padding oracle attack



$D(C_3)$ 5b d8 99 ee
C_2 34 db 9a ef
P_3 6f 03 03 01

MAC ERROR

$P_1 = D(C1) \oplus IV$
$P_2 = D(C2) \oplus C1$
$P_3 = D(C3) \oplus C2$

Padding oracle attack

Original C_2 : **34 db 9a ed**

Modified C_2' : **34 db 9a ef**

$D(C_3)$	5b	d8	99	ee
C_2	34	db	9a	ef
P_3	6f	03	03	01

MAC ERROR

Padding oracle attack

Original C_2 : **34 db 9a ed**

Modified C_2' : **34 db 9a ef**

To get a MAC Error, It must be: $D(C_3) \oplus C_2' = xx\ xx\ xx\ 01$ (valid padding)

So what is $D(C_3)$?

Padding oracle attack

Original C_2 : **34 db 9a ed**

Modified C_2' : **34 db 9a ef**

To get a MAC Error, It must be $D(C_3) \oplus C_2' = xx\ xx\ xx\ 01$ (valid padding)

So what is $D(C_3)$?

$D(C_3) = xx\ xx\ xx\ 01 \oplus C_2'$

$= xx\ xx\ xx\ 01 \oplus 34\ db\ 9a\ ef$

$= xx\ xx\ xx\ ee$

Also tells us the padding byte:

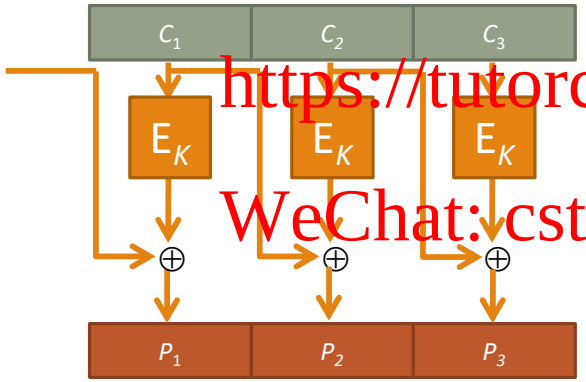
$P_3 = D(C_3) \oplus C_2$

$= xx\ xx\ xx\ ee \oplus 34\ db\ 9a\ ed$

$= xx\ xx\ xx\ 03$

$= xx\ 03\ 03\ 03$ (valid padding)

Padding oracle attack



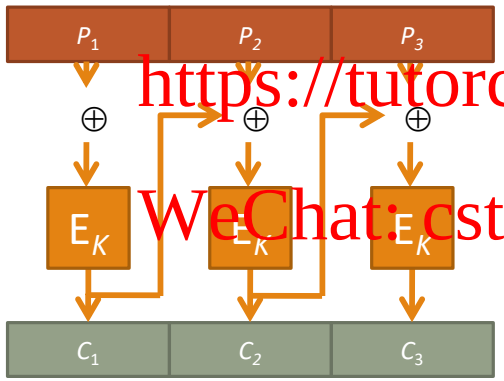
$D(C_3)$	5b	d8	99	ee
C_2	34	db	9a	ed
P_3	6f	03	03	03

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Padding oracle attack



$D(C_3)$	5b	d8	99	ee
C_2	00	dc	9d	ea
P_3	5b	04	04	04

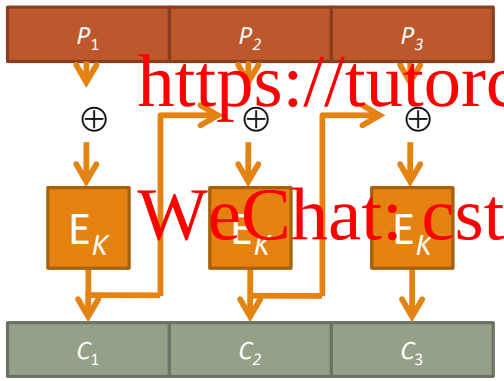
PADDING ERROR

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Padding oracle attack



$D(C_3)$ 5b d8 99 ee

C_2 01 dc 9d ea

P_3 5a 04 04 04

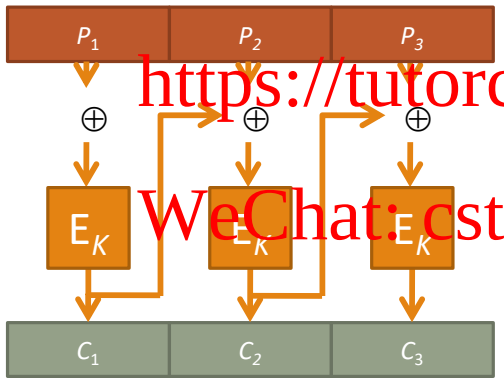
PADDING ERROR

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Padding oracle attack



$D(C_3)$ 5b d8 99 ee

C_2 5f dc 9d ea

P_3 04 04 04 04

MAC ERROR

$P_1 = D(C1) \oplus IV$

$P_2 = D(C2) \oplus C1$

$P_3 = D(C3) \oplus C2$

Other modes

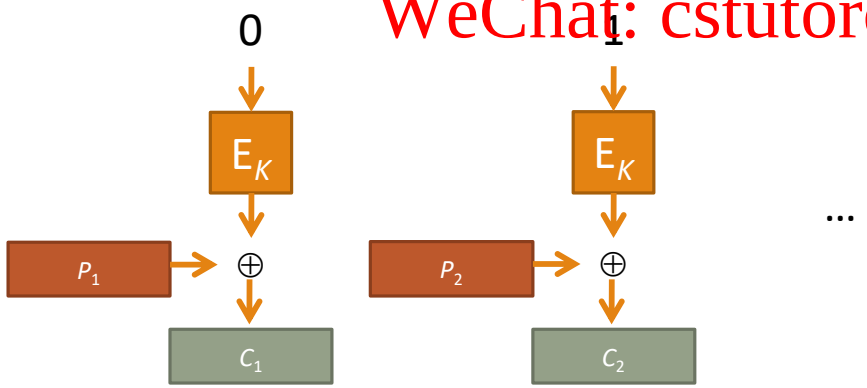
OFB, CFB, etc. – used less often

Counter mode (CTR)

Essentially uses block cipher as a pseudorandom generator

XOR **i**th block of message with $\boldsymbol{E}_k$(message_id || **i**)

Turns a block cipher into a stream cipher!



Building a secure channel

What if you want **confidentiality** and **integrity** at the same time?

Building a secure channel

What if you want **confidentiality** and **integrity** at the same time?

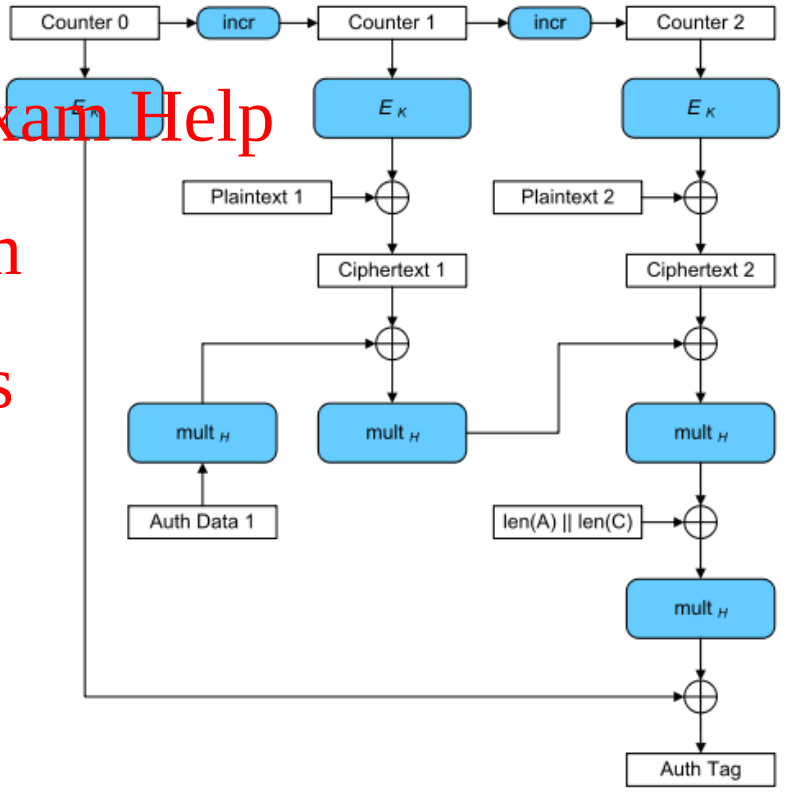
- *Encrypt, then add integrity*, not the other way around (reasons are subtle)
- Use separate keys for confidentiality and integrity
- Need two shared keys, but only have one? That's what PRGs are for!
- If there's a reverse (Bob to Alice) channel, use separate keys for that

Modern encryption mode: Authenticated Encryption

AES-GCM – Galois/Counter Mode

AES in CTR Mode for encryption

Galois Hashing for authentication



Assumption we've been making so far:
Alice and Bob shared a secret key in advance

Amazing fact:
Alice and Bob can have a public conversation to derive a shared key!