Today in Cryptography (5830)

Length-extending encryption
Padding oracle attacks against CBC mode

Recap: Block ciphers, feistel & length preserving encryption

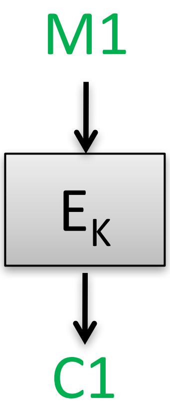
Block cipher is a map $E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$

Each key K defines permutation $E_K : \{0,1\}^n \rightarrow \{0,1\}^n$

Permutation: 1-1, onto

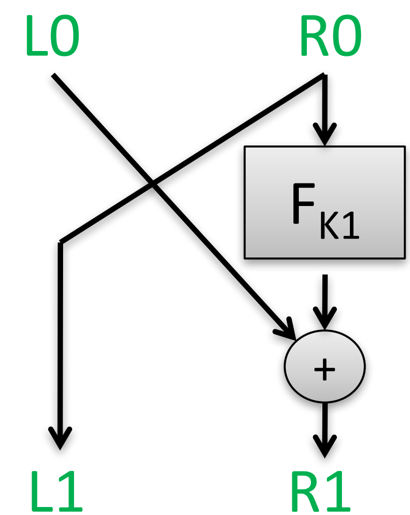
Block ciphers must be efficient

Should behave like random permutation

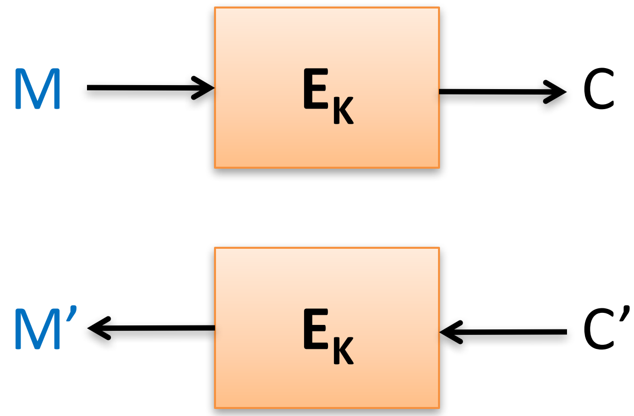


Feistel networks turn function into permutation.

- Used in DES
- Useful for building length-preserving encryption on arbitrary length messages



Security problems with length-preserving encryption?



Looks like a random string (subject to permutivity)

Looks like a random string (subject to permutivity)

Any C’ ≠ C

But determinism has problems:

	Plaintext	Ciphertext
Jane Doe	1343-1321-1231-2310	1049-9310-3210-4732
Thomas Ristenpart	9541-3156-1320-2139	7180-4315-4839-0142
John Jones	2321-4232-1340-1410	5731-8943-1483-9015
Eve Judas	1343-1321-1231-2310	1049-9310-3210-4732

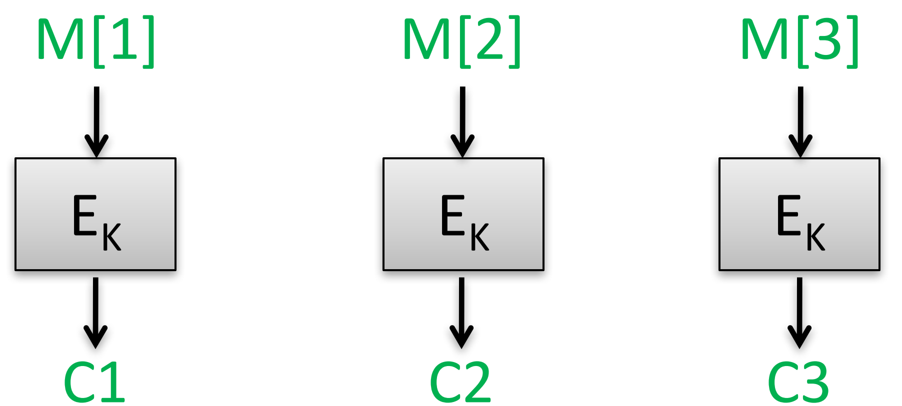
Length-extending encryption security

- Not a bit of information about plaintext leaked
  - Equality of plaintexts hidden
  - Even in case of active attacks
    - Padding oracles we will see later
- Eventually: authenticity of messages as well
  - Decryption should reject modified ciphertexts

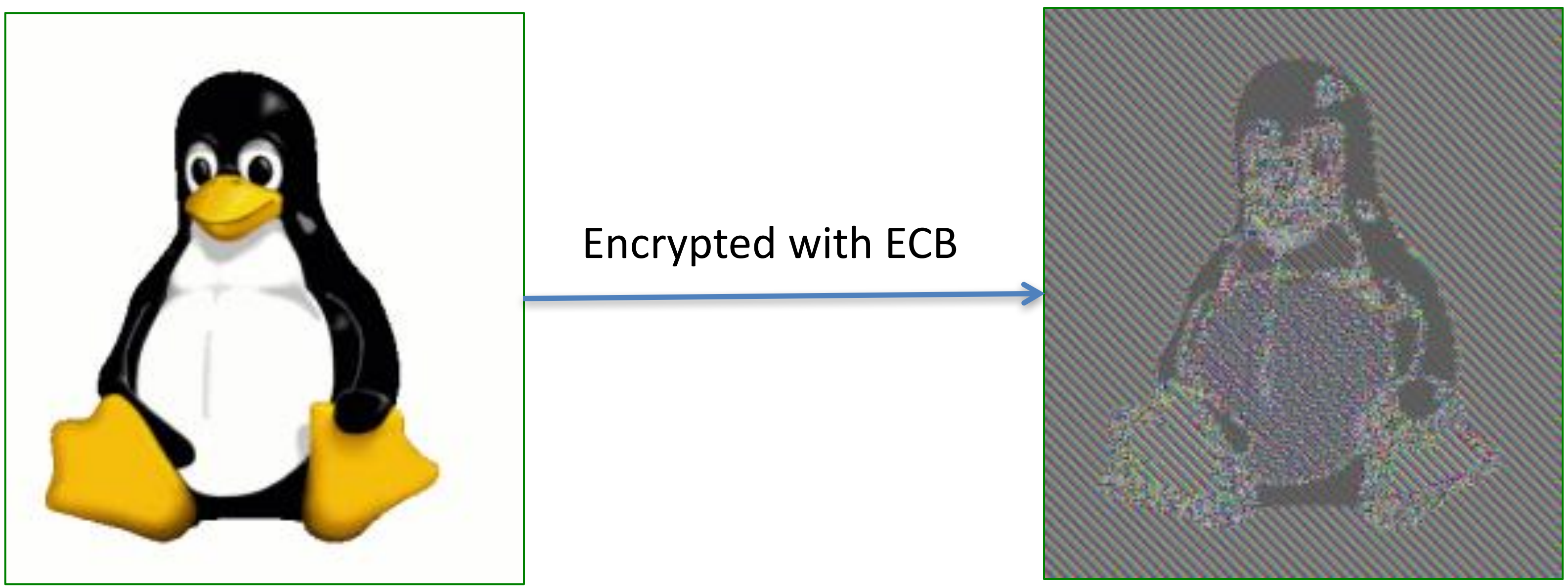
Block cipher modes of operation

How can we build an encryption scheme for arbitrary message spaces out of a block cipher?

Electronic codebook (ECB) mode
Pad message M to M[1],M[2],M[3],... where each block M[i] is n bits
Then:



ECB mode is a more complicated looking substitution cipher



Images courtesy of
http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation

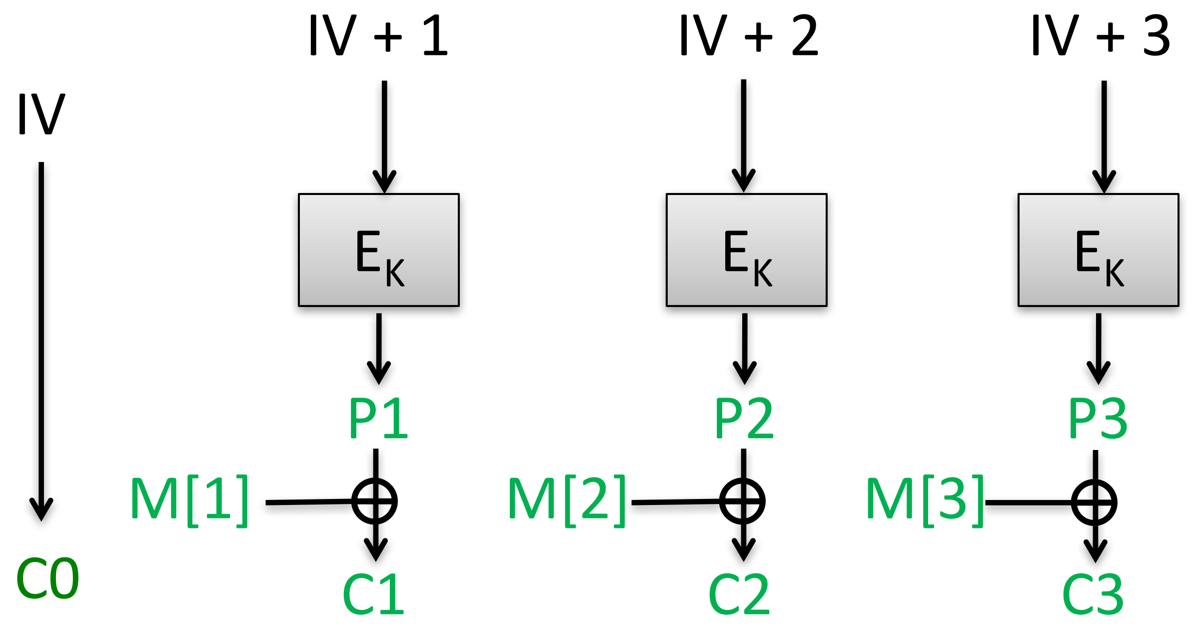
CTR mode encryption using block cipher

Counter mode (CTR)

Pad message M to M[1],M[2],M[3],... where each is n bits except last

Choose random n-bit string IV

Then:



Maybe use less than full n bits of P3

How do we decrypt?

CTR-mode SE scheme

Counter-mode using block cipher E is the following scheme:

<u>Kg():</u>
$K \xleftarrow{\$} \{0,1\}^k$

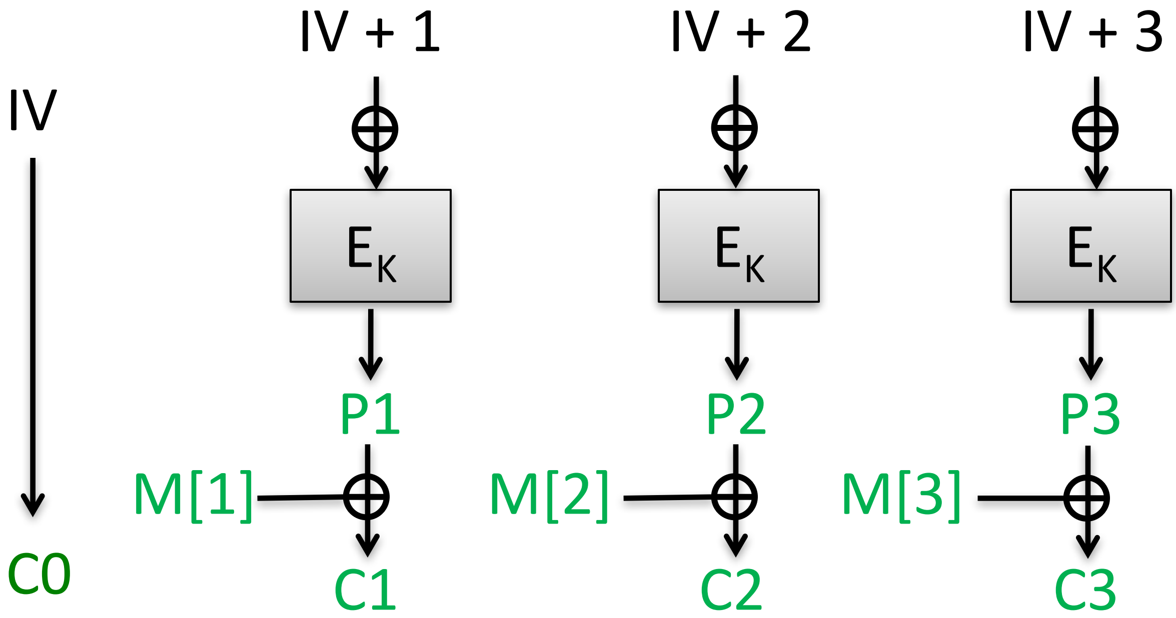
Pick a random key

<u>Enc(K,M):</u>
$L \leftarrow |M|$; $m \Leftarrow \mathrm{ceil}(L/n)$
$IV \xleftarrow{\$} \{0,1\}^n$
$P \leftarrow \mathrm{trunc}_L(E_K(IV \oplus 1) \parallel \cdots \parallel E_K(IV \oplus m))$
Return $(IV, P \oplus M)$

$\mathrm{trunc}_L(\ )$ outputs first L bits of input

<u>Dec(K,(IV,C)):</u>
$L \leftarrow |C|$; $m \Leftarrow \mathrm{ceil}(L/n)$
$P \leftarrow E_K(IV \oplus 1) \parallel \cdots \parallel \mathrm{trunc}(E_K(IV \oplus m))$
Return $(IV, P \oplus C)$

Assume ciphertext can be parsed into IV and remaining ciphertext bits



Can attacker learn K from just C0,C1,C2,C3?

Implies attacker can break E, i.e. recover block cipher key

Can attacker learn M = M[1],M[2],M[3] from C0,C1,C2,C3?

Implies attacker can invert the block cipher without knowing K

Can attacker learn one bit of M from C0,C1,C2,C3?

Implies attacker can break PRF security of E

Passive adversaries cannot learn anything about messages

Multi-message secure encryption

Security goal: Enc(K,M) doesn't even leak single bit about M

Def. (Asymptotic version)
There exists some negligible function ϵ, such that for all n, for all polynomials $q = q(n)$, for any messages $M_1, \ldots, M_q$ and $M_1', \ldots, M_q'$ with $|M_i| = |M_i'|$ for all i, and for any p.p.t. distinguisher D it holds that:

$$\Big| \Pr[D(\mathrm{Enc}(K, M_1),\ldots,\mathrm{Enc}(K,M_q')) = 1] - \Pr[D(\mathrm{Enc}(K, M_1'),\ldots,\mathrm{Enc}(K,M_q')) = 1] \Big| \leq \epsilon$$

where probabilities are over K and randomness used by Enc.

Sometimes called indistinguishability under chosen plaintext attack (IND-CPA) (slight technical differences)

Adaptive variant allows distinguisher to choose m_j as a function of $\mathrm{Enc}(K,m_i)$ for $i < j$

Multi-message secure encryption

Security goal: Enc(K,M) doesn't even leak single bit about M

Def. (Concrete version)
Enc is (t,q,L,ϵ)-secure if for all distinguishers D running in time at most t and for any messages $M_1, \ldots, M_q$ and $M_1', \ldots, M_q'$ with $|M_i| = |M_i'| \leq L$ for all i, it holds that

$$\left| \Pr[D(\mathrm{Enc}(K, M_1),\ldots,\mathrm{Enc}(K,M_q')) = 1] - \Pr[D(\mathrm{Enc}(K, M_1'),\ldots,\mathrm{Enc}(K,M_q')) = 1] \right| \leq \epsilon$$

where probabilities are over K and randomness used by Enc.

We will want ϵ tiny (ex: 2^{-50}), t and q pretty large (ex: 2^{80})

Example: $\epsilon < 2^{-50}$ $\quad q \leq 2^{50}$ $\quad t \leq 2^{80}$

CTR-mode security

Thm. Let $\rho : \{0,1\}^n \to \{0,1\}^n$ be a random function. Then CTR-mode using E is (t,q,L,ϵ)-secure for $\epsilon \leq (\sigma q)^2 / 2^n$ for $\sigma = \lceil L/n \rceil$.

Combine above theorem with PRF security of a block cipher E to show security of CTR using block cipher E.

(Time t arises in this step)

Birthday bound upper and lower bounds:
https://cseweb.ucsd.edu/~mihir/cse207/w-birthday.pdf

Malleability example: Encrypted cookies

http://startup.com

GET /login.html   HTTP/1.1

HTTP/1.1 200 OK
Content-type: text/html
Set-cookie: sessionToken=ab35h013490...

GET /admin.html   HTTP/1.1
Cookie: sessionToken=ab35h013490…

Decrypt and check whether admin or not

abc35h013490…   =   CTR-Mode(K, “admin=0”)

Malicious client can simply flip a few bits to change admin=1

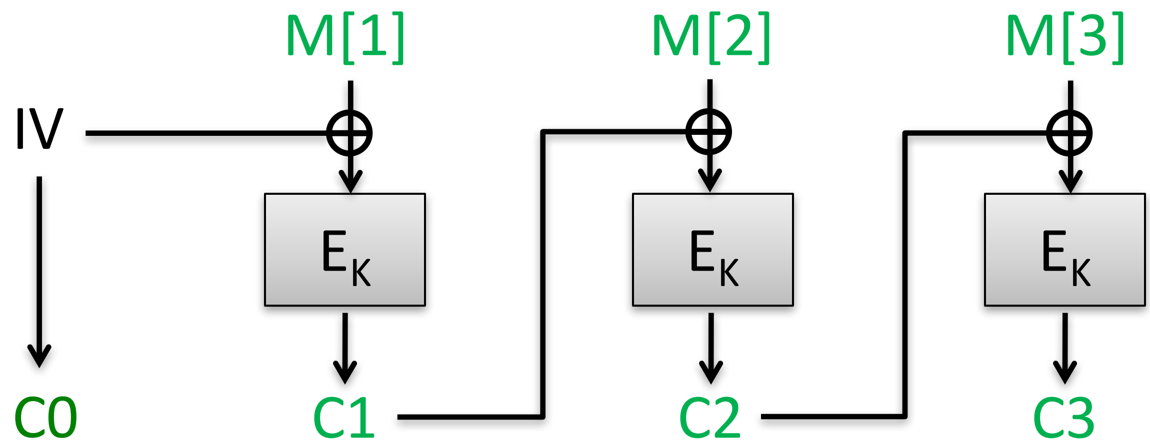
CBC mode

Ciphertext block chaining (CBC)

Pad message M to M[1],M[2],M[3],... where each block M[i] is n bits

Choose random n-bit string IV

Then:



How do we decrypt?

CBC-mode SE scheme

<u>Kg():</u>
K <-$ $\{0,1\}^k$

Pick a random key

<u>Enc(K,M):</u>
L <- |M| ; m <= ceil(L/n)
C_0 <- IV <-$ $\{0,1\}^n$
$M_1,\dots,M_m$ <- PadCBC(M,n)
For i = 1 to m do
    $C_i \leftarrow E_K(C_{i-1} \oplus M_i)$
Return $(C_0, C_1, \dots, C_m)$

PadCBC unambiguously pads M to a string of mn bits

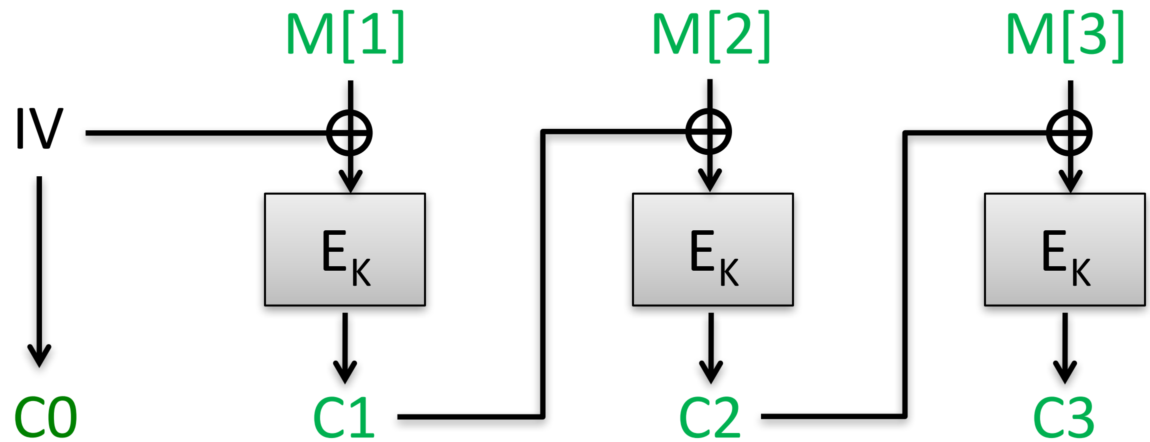
<u>Dec(K,(C_0 , C_1 , …, C_m)):</u>
For i = 1 to m do
    $M_i \leftarrow C_{i-1} \oplus D_K(C_i)$
M <- UnpadCBC($M_1,\dots,M_m$,n)
Return M

UnpadCBC removes padding, returns appropriately long string

CBC-mode security

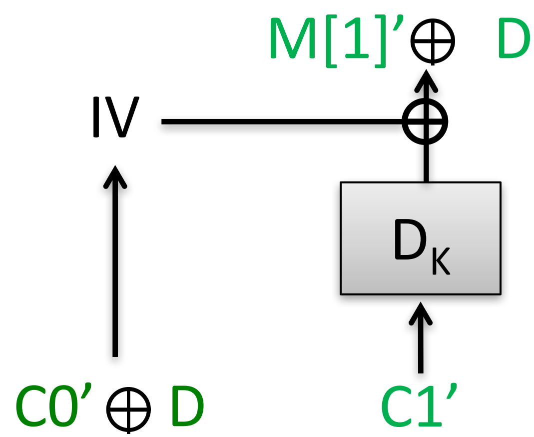
Analysis similar to CTR mode gives similar birthday-style security bound for chosen-plaintext security

CBC mode has “malleability” issues, too



How do we change bits of M1 received by server??

For any D:

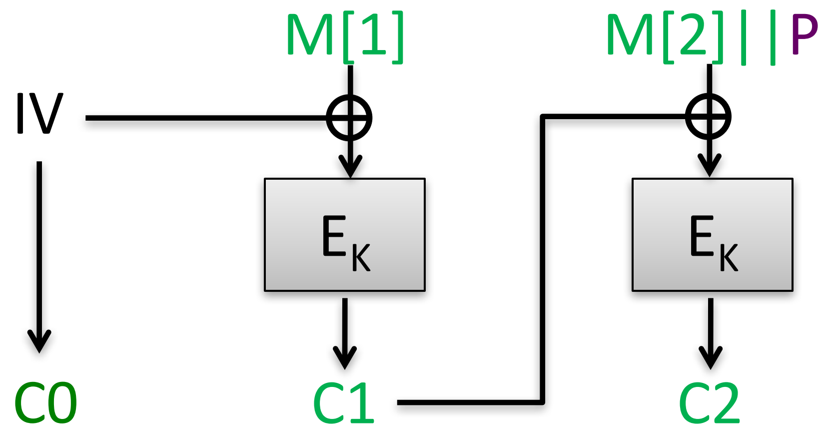


Padding for CBC mode

- CBC mode handles messages with length a multiple of n bits
- We use padding to make it work for arbitrary encryption schemes

- Padding checks often give rise to padding oracle attacks

Simple situation: pad by 1 byte



Assume that M[1]||M[2] has length 2n-8 bits

P is one byte of padding that must equal 0x00

C0 , C1 , C2

ok

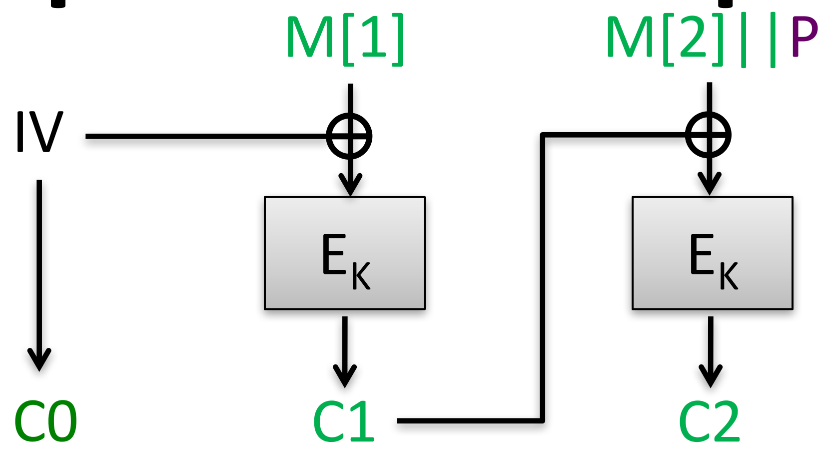
Adversary obtains Ciphertext C0,C1,C2

C0, C1⊕1 , C2

error

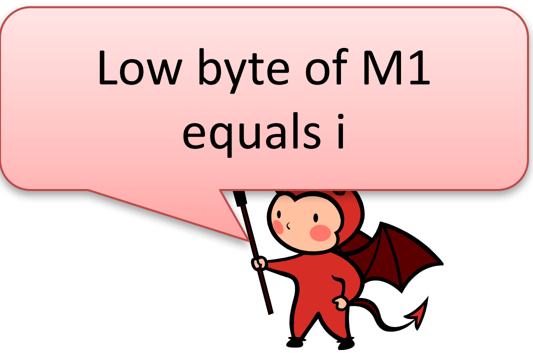
<u>Dec(K, C’)</u>
M[1]’||M[2]’||P’ = CBC-Dec(K,C’)
If P’ ≠ 0x00 then
    Return error
Else
    Return ok

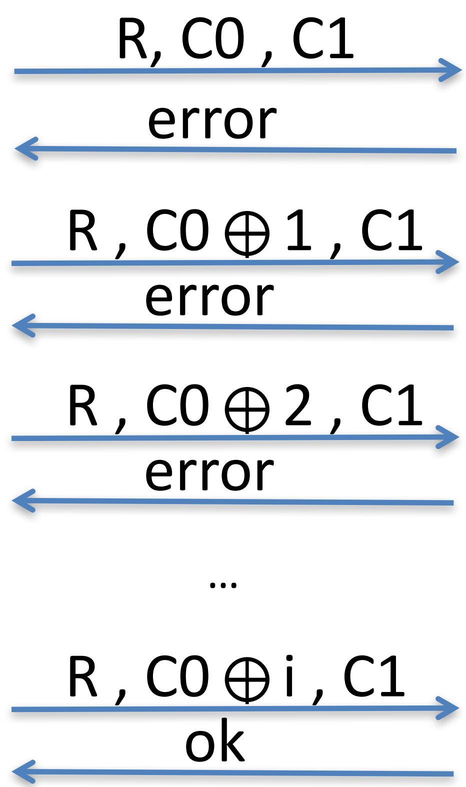
Simple situation: pad by 1 byte



Assume that M[1]||M[2] has length 2n-8 bits

P is one byte of padding that must equal 0x00





Adversary obtains ciphertext C = C0,C1,C2

Let R be arbitrary n bits

<u>Dec(K, C’)</u>

M[1]’||M[2]’||P’ = CBC-Dec(K,C’)

If P’ ≠ 0x00 then

  Return error

Else

  Return ok

PKCS #7 Padding

$$\text{PKCS\#7-Pad}(M) \quad = \quad M \,||\, \underbrace{P \,||\, \ldots \,||\, P}_{\text{P repetitions of byte encoding number of bytes padded}}$$

Possible paddings:

01
02 02
03 03 03
04 04 04 04
…
FF FF FF FF … FF

For block length of 16 bytes, never need more than 16 bytes of padding (10 10 … 10)

Decryption

(assuming at most one block of padding)

```
Dec( K, C )
M[1] || … || M[n] = CBC-Dec(K,C)
P = RemoveLastByte(M[n])
while i < int(P):
      P’ = RemoveLastByte(M[n])
      If P’ != P then
            Return error
Return ok
```

PKCS #7 padding oracles

Low byte of M1 most likely equals i ⊕ 01

Adversary obtains ciphertext C = C0,C1,C2

Let R be arbitrary n bits

R, C0 , C1

error

R , C0 ⊕ 1 , C1

error

R , C0 ⊕ 2 , C1

error

…

R , C0 ⊕ i , C1

ok

<u>Dec(K, C)</u>

```
M[1] || … || M[n] = CBC-Dec(K,C)
P = RemoveLastByte(M[n])
while i < int(P):
      P' = RemoveLastByte(M[n])
      If P' != P then
            Return error
Return ok
```

PKCS #7 padding oracles

Second lowest byte of M1 equals i xor 02

R, C0 , C1

error

R , C0 ⊕ 1||j , C1

error

R , C0 ⊕ 2||j , C1

error

…

R , C0 ⊕ i||j , C1

ok

Dec(K, C)
M1 || … || Mn = CBC-Dec(K,C)
P = RemoveLastByte(Mn)
while i < int(P):
    P' = RemoveLastByte(Mn)
    If P' != P then
        Return error
Return ok

Adversary obtains ciphertext C = C0,C1,C2

Let R be arbitrary n bits

Set j = i

Chosen ciphertext attacks against CBC

Attack	Description	Year
Vaudenay	10’s of chosen ciphertexts, recovers message bits from a ciphertext. Called “padding oracle attack”	2001
Canvel et al.	Shows how to use Vaudenay’s ideas against TLS	2003
Degabriele, Paterson	Breaks IPsec encryption-only mode	2006
Albrecht et al.	Plaintext recovery against SSH	2009
Duong, Rizzo	Breaking ASP.net encryption	2011
Jager, Somorovsky	XML encryption standard	2011
Duong, Rizzo	“Beast” attacks against TLS	2011

None of these modes are secure for general-purpose encryption

- ECB is obviously insecure

- CTR mode and CBC mode fail in presence of active attacks
  - Cookie example
  - Padding oracle attacks

- ***Next lecture***: adding authentication mechanisms to prevent chosen-ciphertext attacks