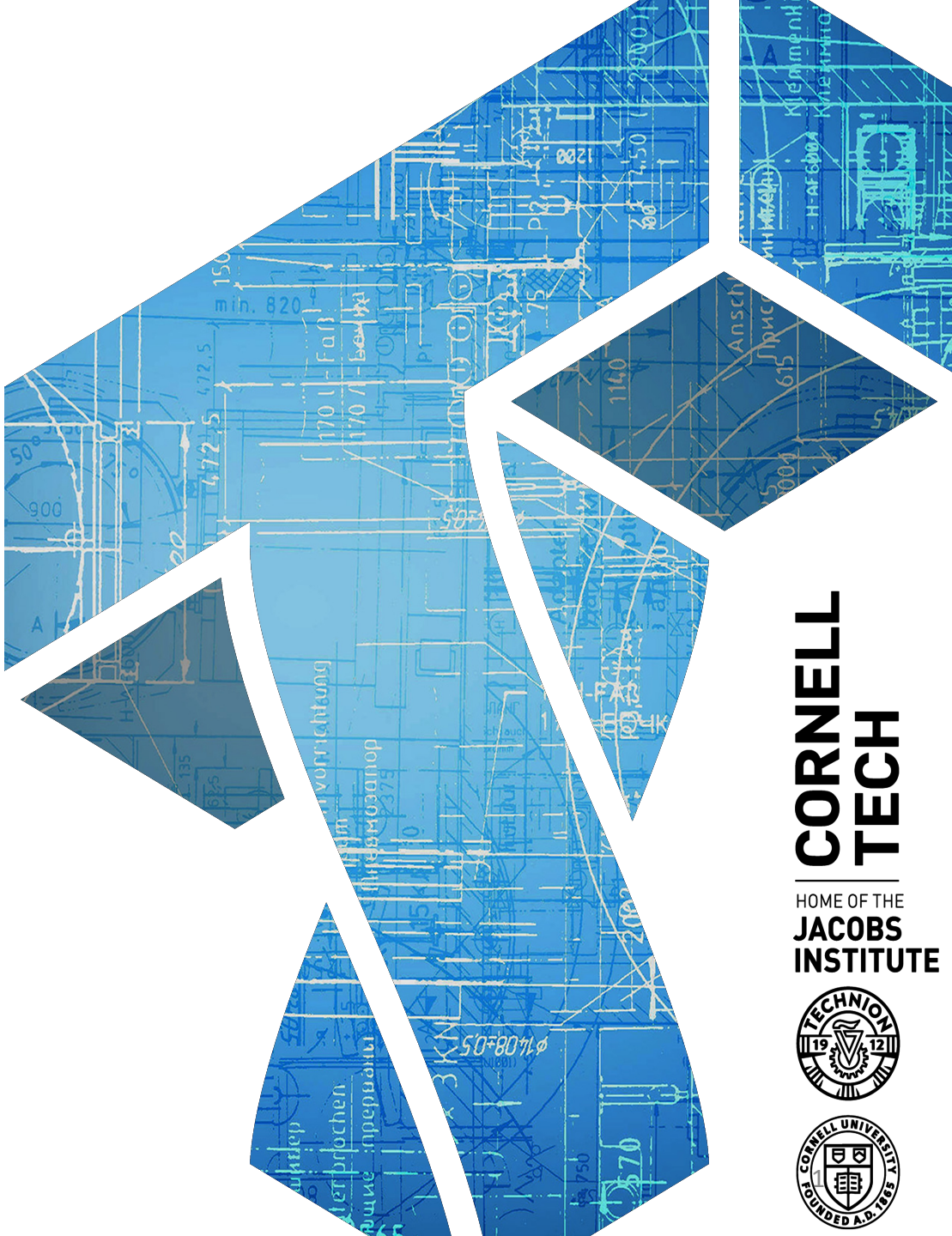
CS 5830
Cryptography



CORNELL TECH

HOME OF THE JACOBS INSTITUTE

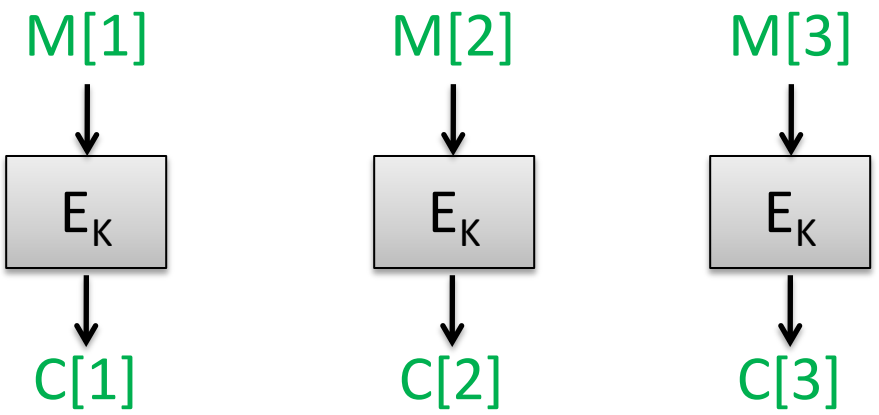
Recap and where we're at

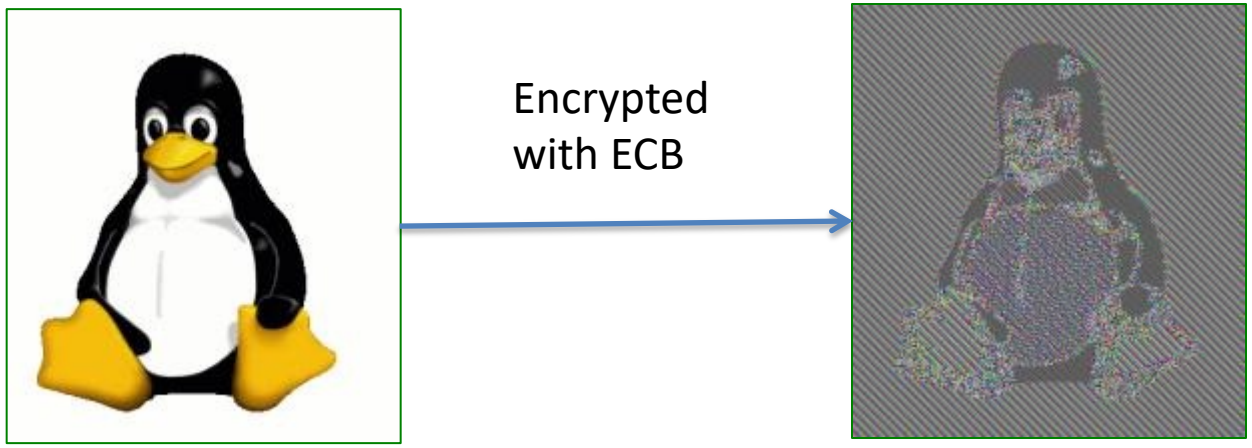
- Blockciphers and their security goals
  - Assume good blockciphers that achieve PRF security up to implications of best-known generic attacks
  - ~ 2^k time (exhaustive key search)
  - ~ $2^{n/2}$ time (birthday attacks)

- Today: modes of operation, IND-CPA security, & (time allowing) chosen-ciphertext attacks

Block cipher modes of operation

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

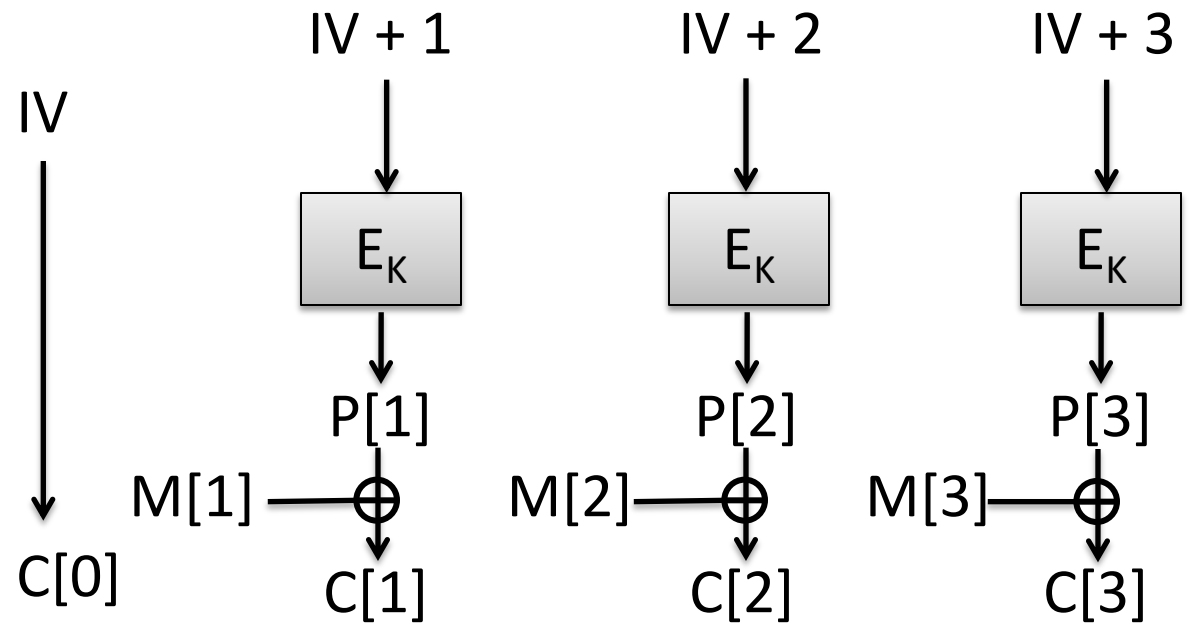




Images courtesy of
http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation

CTR mode

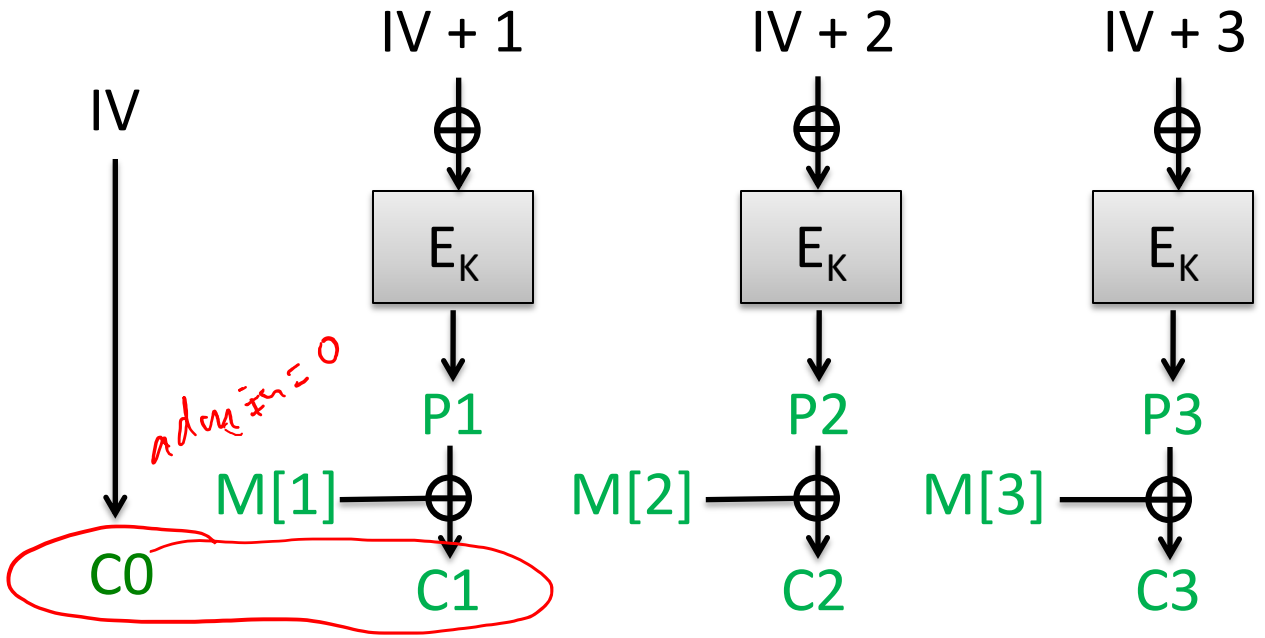
Block cipher $E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$ is a family of permutations
Should be secure as a pseudorandom function (PRF)



CTR mode provides message confidentiality (nothing about message from ciphertext) assuming E is a PRF and number of message blocks encrypted $<< 2^{n/2}$

CTR is a blockcipher mode of operation

- How do we encrypt long messages with block cipher?
  - Modes of operation
- Long history: NIST standard
  - First published in 1980 specifically for DES
  - 2001 version: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf
- Analyses starting in 1990s for chosen-plaintext attacks
- Analyses starting in 2000s for chosen-ciphertext attacks



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 break PRF security of E

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

Formalizing security for CPA attacks

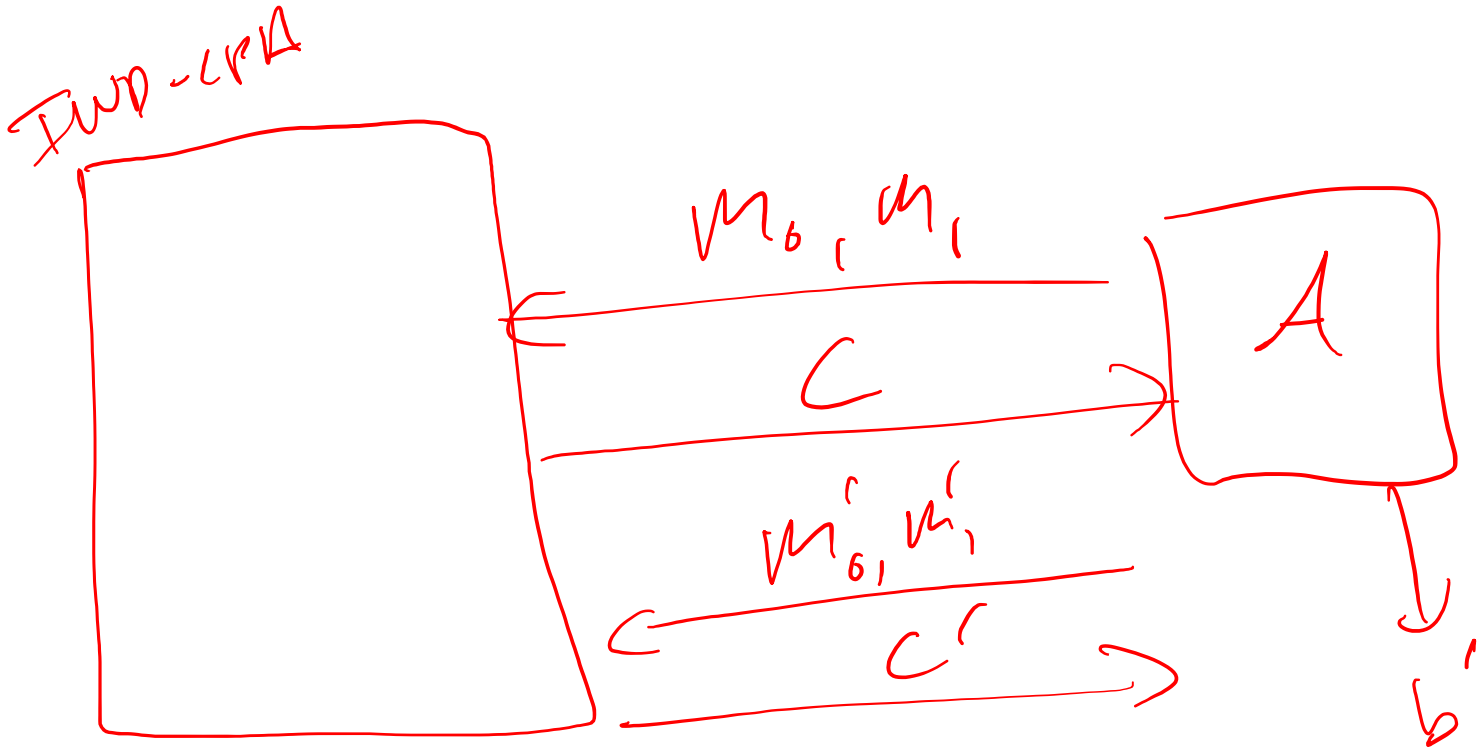
- Indistinguishability under chosen-plaintext attacks (IND-CPA)
  - Multi-message, adaptive security
  - Query encryption oracle q times adaptively with pair of equal-length messages M_0, M_1 ; get back $Enc(K, M_b)$ for secret key K
  - Can adversary infer b?

IND-CPA(SE, $\mathcal{A}$):
K <-$ Kg ; b <-$ {0,1}
b' <-$ $\mathcal{A}^{LR}$
Return (b = b')

LR(M_0,M_1):
C <-$ Enc(K,M_b)
Return C

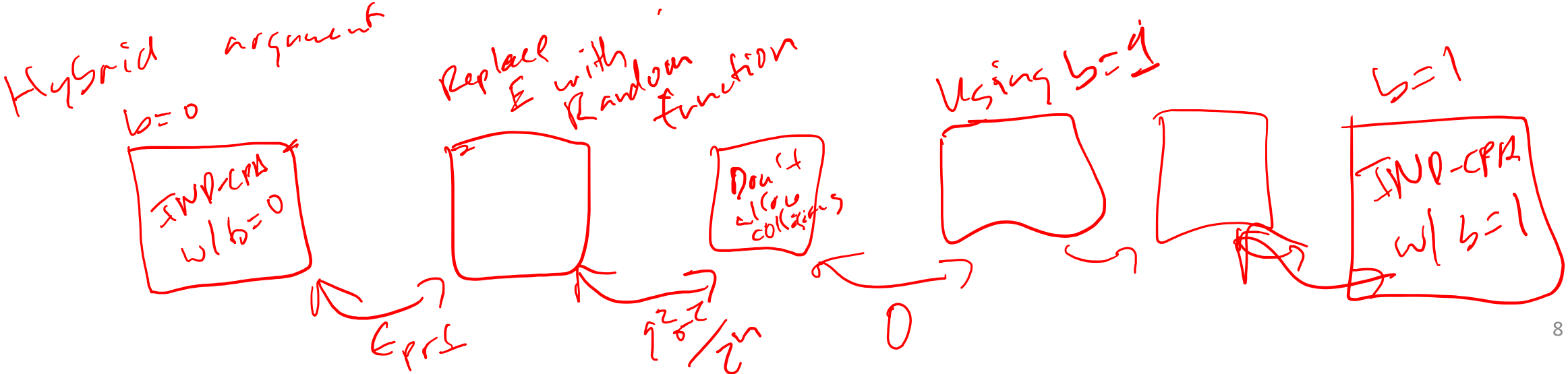
Def. A symmetric encryption scheme is (t, q, L, ϵ)-IND-CPA if for any adversary $\mathcal{A}$ running in time at most t and making at most q queries of length at most L bits, it holds that

$$\Pr[\text{IND-CPA}(SE, \mathcal{A}) = 1] \leq 1/2 + \epsilon$$



IND-CPA(SE, $\mathcal{A}$):
K <-$ Kg ; b <-$ {0,1}
b' <-$ $\mathcal{A}^{LR}$
Return (b = b')

LR(M_0,M_1):
C <-$ Enc(K,M_b)
Return C



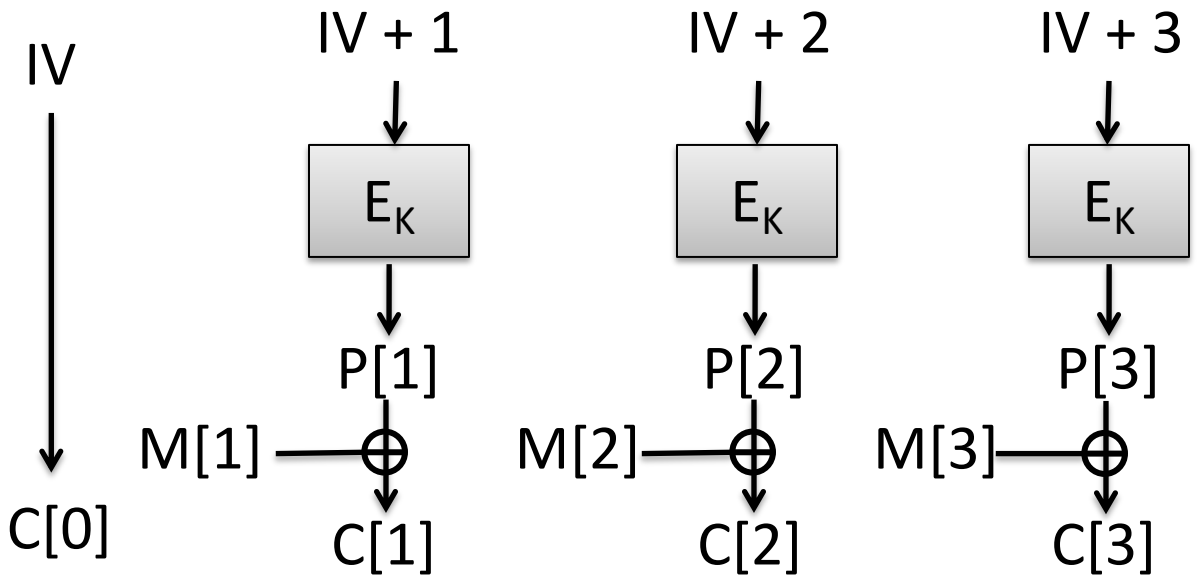
CTR mode IND-CPA security

Thm. Let E be a blockcipher with blocksize n. Then CTR-mode using E is (t,q,L,ϵ_{cpa})-IND-CPA with

$$\epsilon_{cpa} \leq \epsilon_{prf} + (\sigma q)^2 / 2^n$$

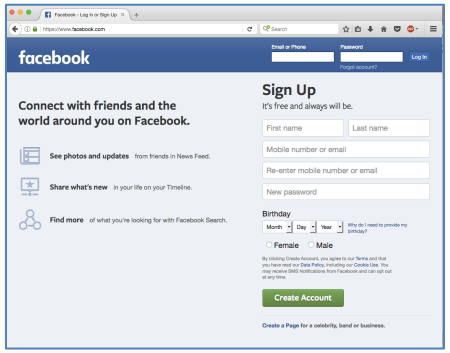
assuming E is $(t,q\sigma,\epsilon_{prf})$-PRF secure blockcipher for $\sigma = \lceil L/n \rceil$.

CTR mode and different threat models



Attack type	Capability	Goal
Indistinguishability under chosen plaintext attack	Observe ciphertexts under secret key of chosen plaintexts with >=1 bit of uncertainty	Learn at least one bit of information about plaintext
Ciphertext integrity attack	Get example ciphertext(s); maul ciphertexts and submit to decryption oracle	Trick recipient into accepting forged plaintext
Indistinguishability under chosen-ciphertext attack	Get example ciphertext(s); maul ciphertexts and submit to partial decryption oracle	Learn information about plaintexts

Session handling and login



GET /index.html

Set-Cookie: AnonSessID=134fds1431

Protocol is HTTPS. Elsewhere else just HTTP.

POST /login.html?name=bob&pw=12345

Cookie: AnonSessID=134fds1431

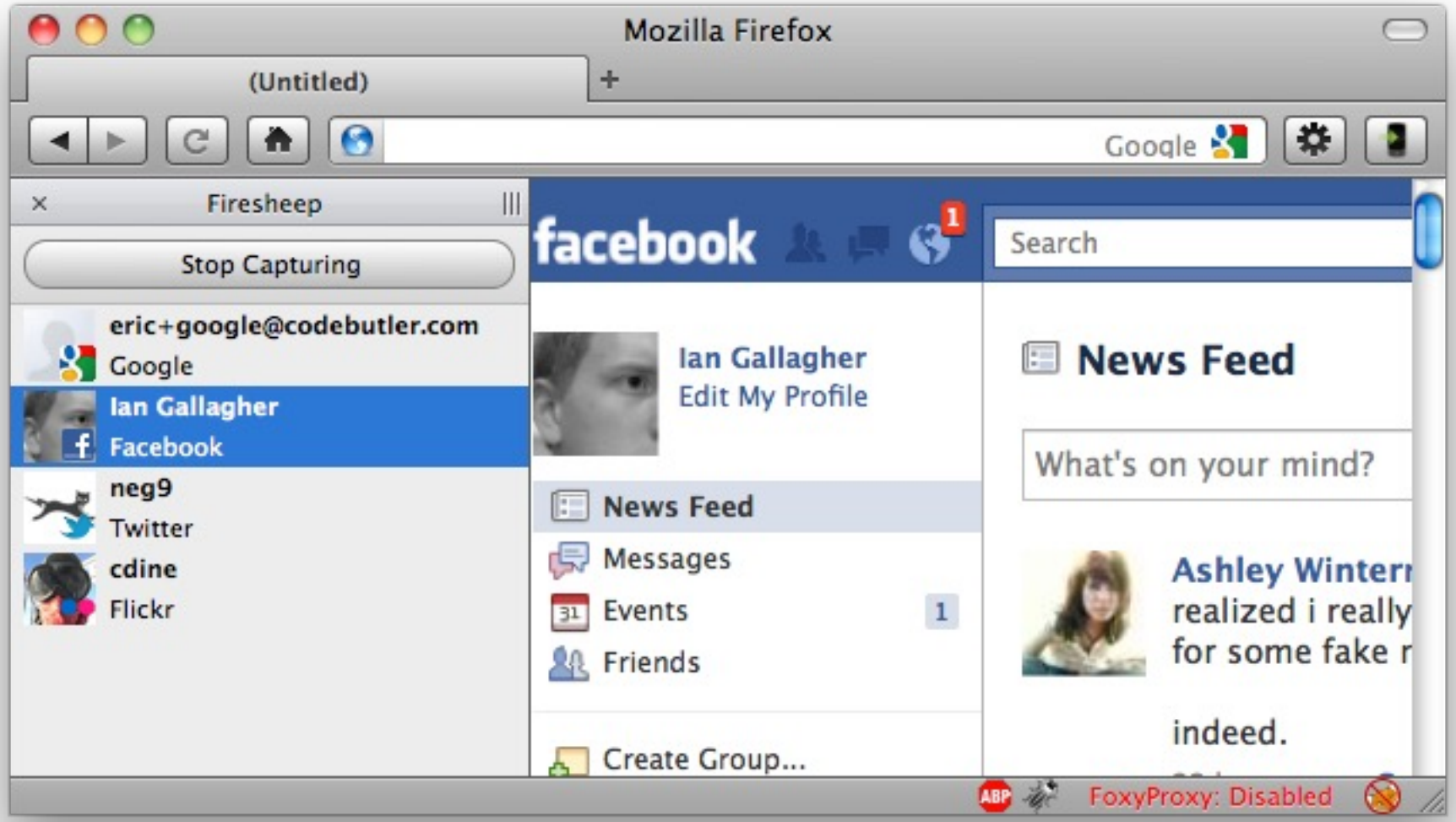
Set-Cookie: SessID=83431Adf

Nowadays increasingly all HTTPS

GET /account.html

Cookie: SessID=83431Adf

Without HTTPS: session hijacking often trivial



From http://codebutler.com/firesheep

Security problems here?

Facebook.com

POST /login.html?name=bob&pw=12345

Cookie: AnonSessID=134fds1431

Set-Cookie: SessID=83431Adf

GET /account.html

Cookie: SessID=83431Adf

Secret key K only known to server

83431Adf = CTR-Enc(K, “admin=0”)

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

Example of an ***integrity / authenticity violation***

Soon we will build authentication mechanisms to prevent this

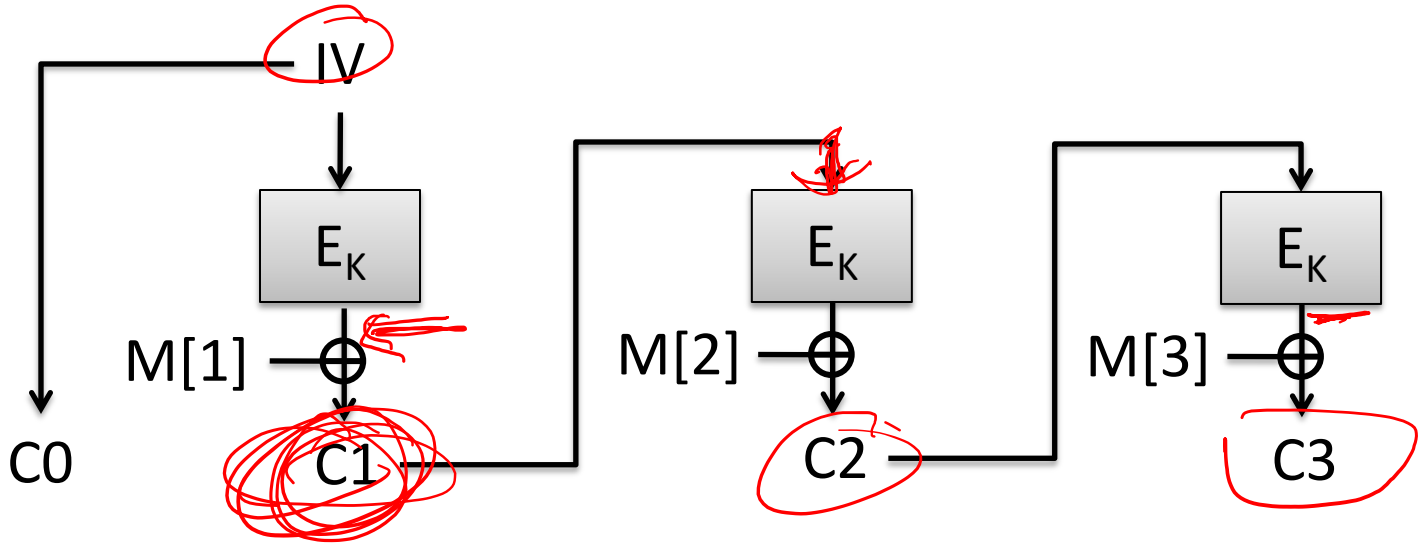
NIST Modes

- ~~Electronic codebook mode (ECB)~~
- Counter mode (CTR)
- Ciphertext feedback mode (CFB)
- Offset feedback mode (OFB)
- Ciphertext block chaining mode (CBC)

CTR, CBC found widespread use

CFB mode

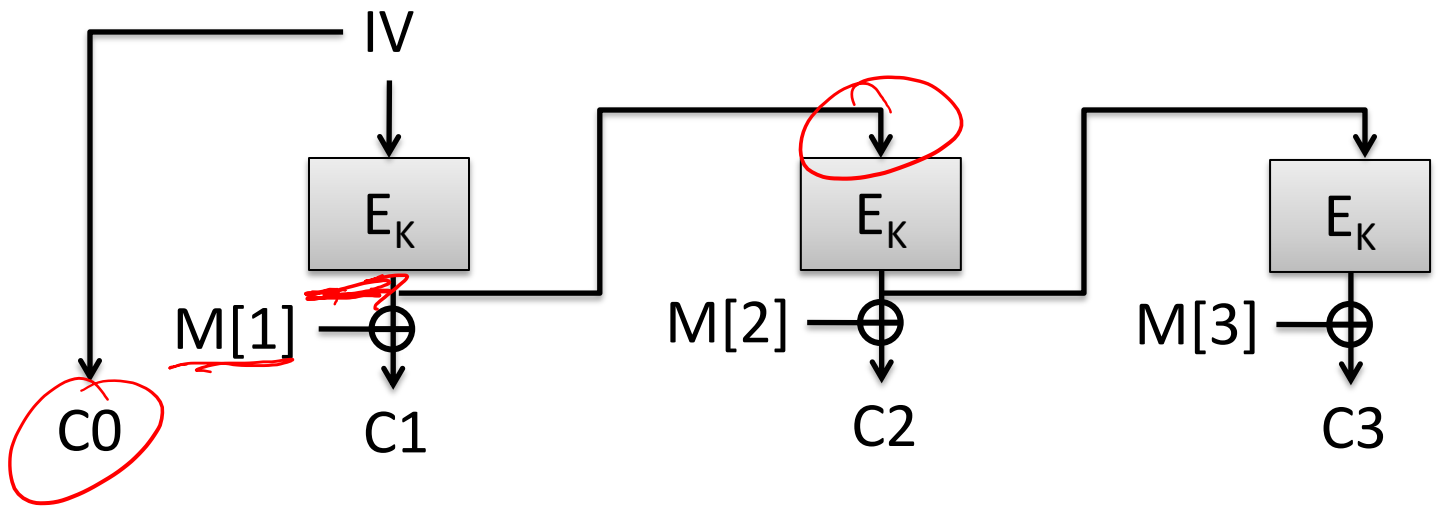
Ciphertext feedback mode (CFB)
Pad message M to M[1],M[2],M[3],... where each block M[i] is n bits
Choose random n-bit string IV
Then:



How do we decrypt?

OFB mode

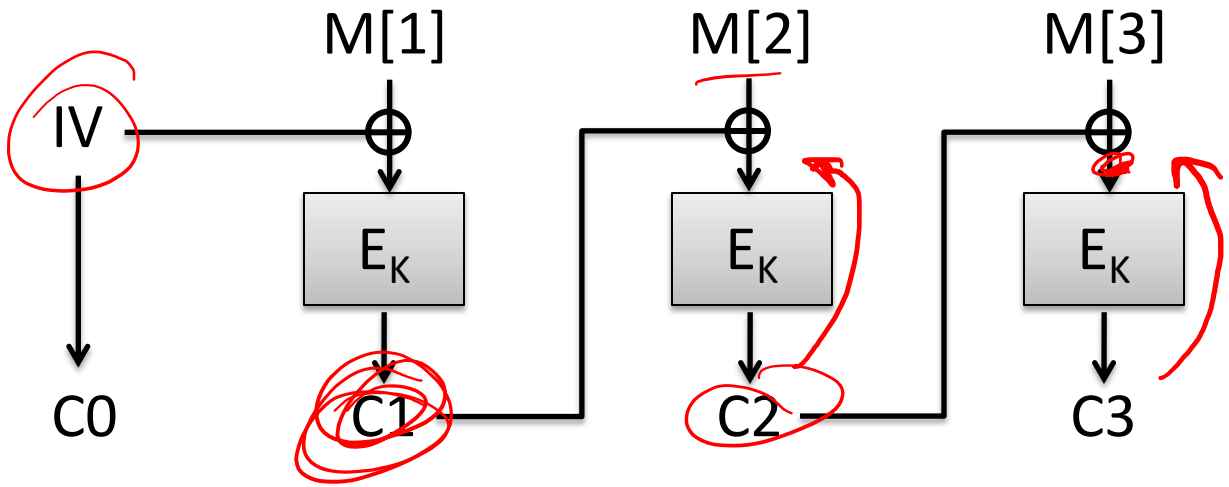
Offset feedback mode (OFB)
Pad message M to M[1],M[2],M[3],... where each block M[i] is n bits
Choose random n-bit string IV
Then:



How do we decrypt?

CBC mode

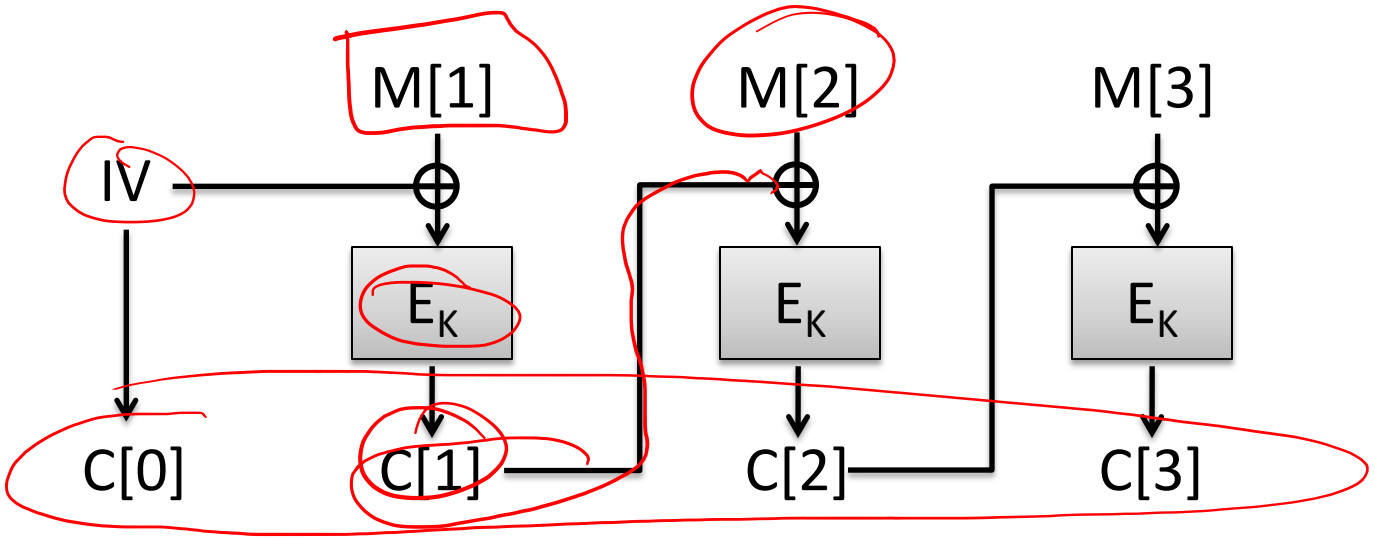
Ciphertext block chaining (CBC)
Pad message M to M[1],M[2],M[3],... where each block M[i] is n bits
Choose random n-bit string IV
Then:



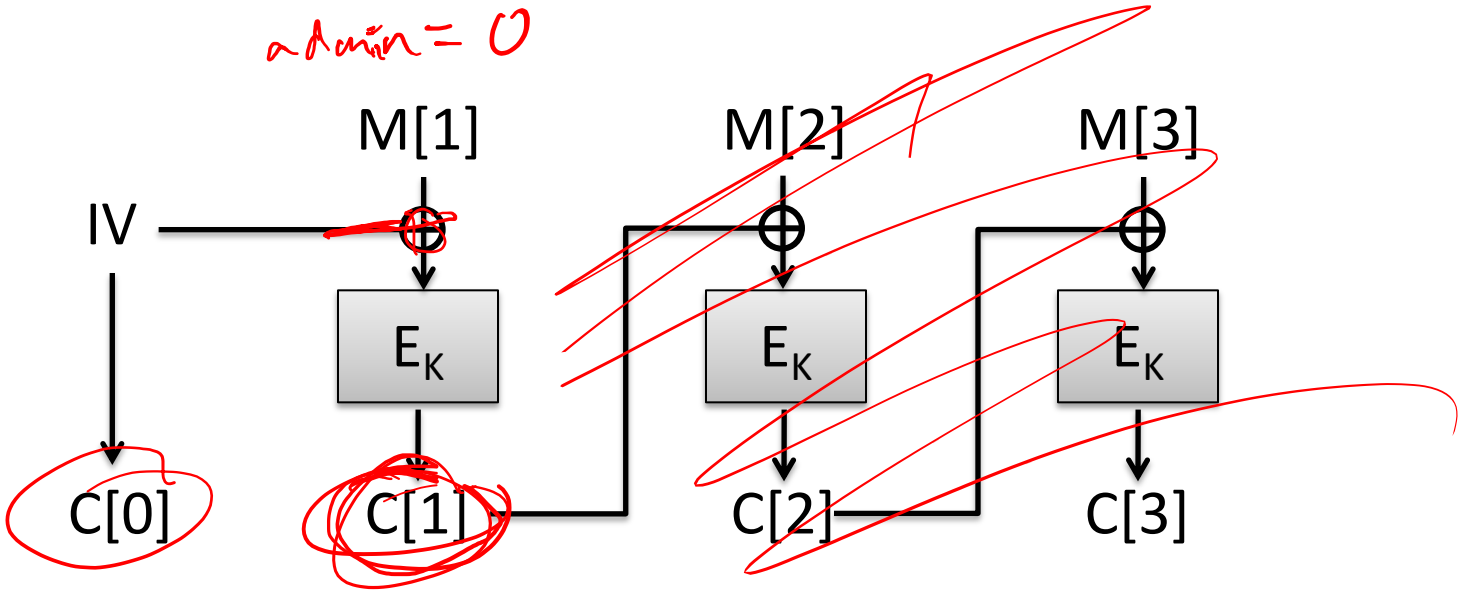
How do we decrypt?

IND-CPA of CBC mode

- CBC mode can also be shown to be IND-CPA secure, with similar security level as that of CTR mode
  - What if IV collides?
  - Must switch keys before number of encryptions q gets close to $2^{n/2}$

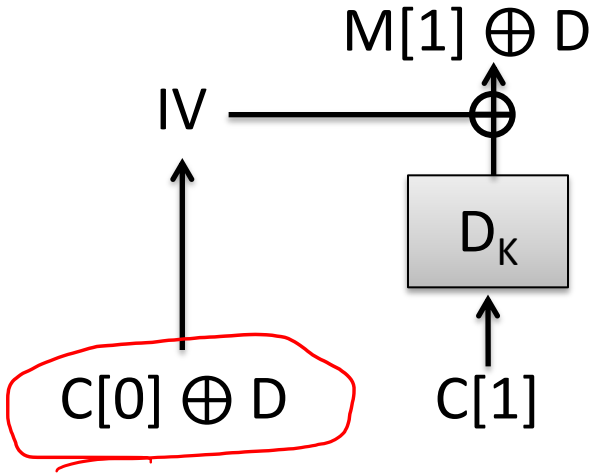


CBC mode has malleability issues, too



How do we change bits of M 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 message lengths
  - PadCBC, UnpadCBC map to, from strings of length multiple of n
  - Will specify example padding schemes later

Pseudocode for CBC mode with padding

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

Pick a random key

CBC-Enc(K,M):
L <- |M| ; m <- ceil(L/n)
C[0] <- IV <-$ $\{0,1\}^n$
M[1],…,M[m] <- PadCBC(M,n)
For i = 1 to m do
    C[i] <- $E_K(C[i-1] \oplus M[i])$
Return C[0] || C[1] || … || C[m]

PadCBC unambiguously pads M to a sequence of n bit message blocks

CBC-Dec(K,C):
For i = 1 to m do
    M[i] <- $C[i-1] \oplus D_K(C[i])$
M <- UnpadCBC(M[1],…,M[m],n)
Return M

UnpadCBC removes padding, returns appropriately long string
May output error if padding is wrong
In crypto, errors often denoted by $\perp$

Padding for CBC mode

- CBC mode handles messages with length a multiple of n bits
- We use padding to make it work for arbitrary message lengths
  - PadCBC, UnpadCBC map to, from strings of length multiple of n
  - Will specify example padding schemes later
- Padding checks often give rise to chosen-ciphertext attack called ***padding oracle attacks***
  - Given CBC mode encryption C = Enc(K,M) for unknown M
  - Access to oracle that reveals just whether decryption succeeds
  - Recover M

Partial decryption oracles arise frequently in practice

C_1 = Enc(K, “secret data”)

C_2 = Enc(K, “more secrets”)

K

K

C_1

C_3 = Enc(K, “more secrets”)

C’ = modified C_1

C_4 = Enc(K, “error”)

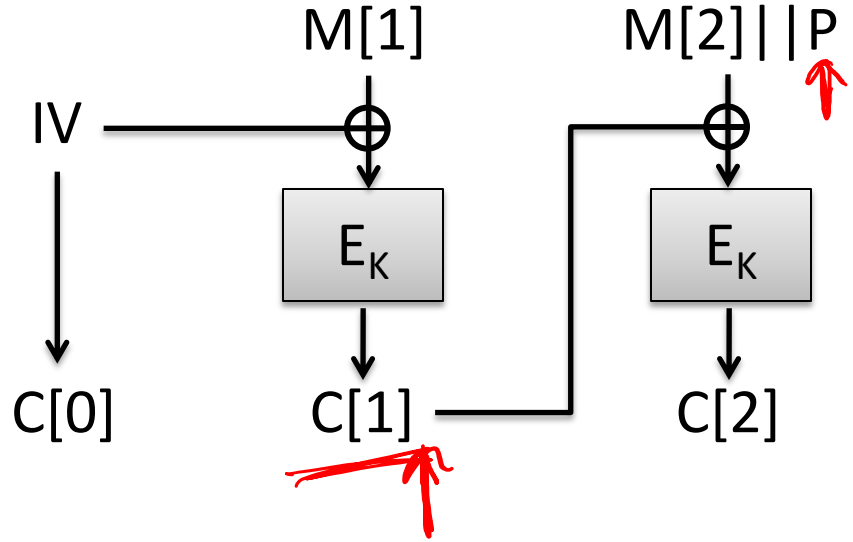
In practice usually easy to distinguish C_3 from C_4 even without K

$|C_4| \neq |C_3|$

Timing differs for successful vs. unsuccessful decryption

TLS/HTTPS canonical examples where decryption oracles arise

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

C[0] , C[1] , C[2]

ok

C[0], C[1]⊕1 , C[2]

error

Adversary obtains ciphertext C[0],C[1],C[2]

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