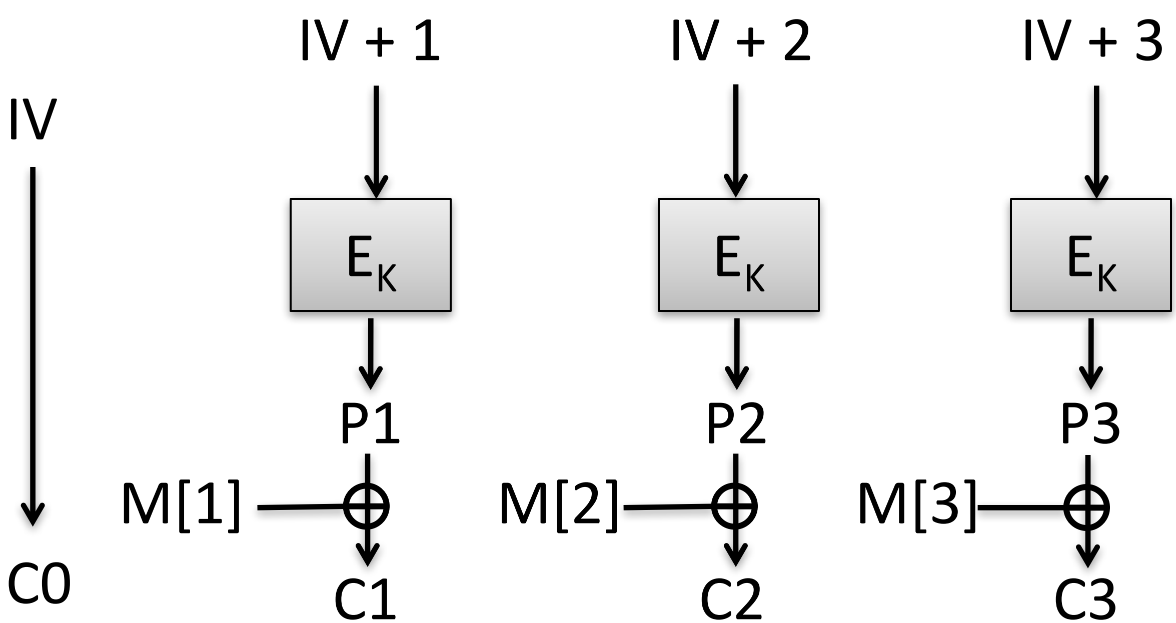
Today in Cryptography (5830)

CBC mode
Padding oracle attacks against CBC mode

Recap: CTR mode

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



CTR-mode security

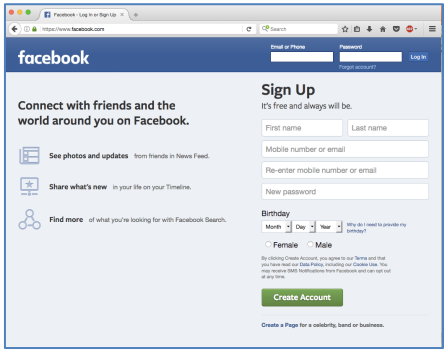
Thm. Let $\rho : \{0,1\}^n \text{->} \{0,1\}^n$ be a random function. Then CTR-mode using E is (t,q,L,ϵ)-secure for $\epsilon \leq (\sigma q)^2 / 2^n$ for $\sigma = \lceil L/n \rceil$.

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

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

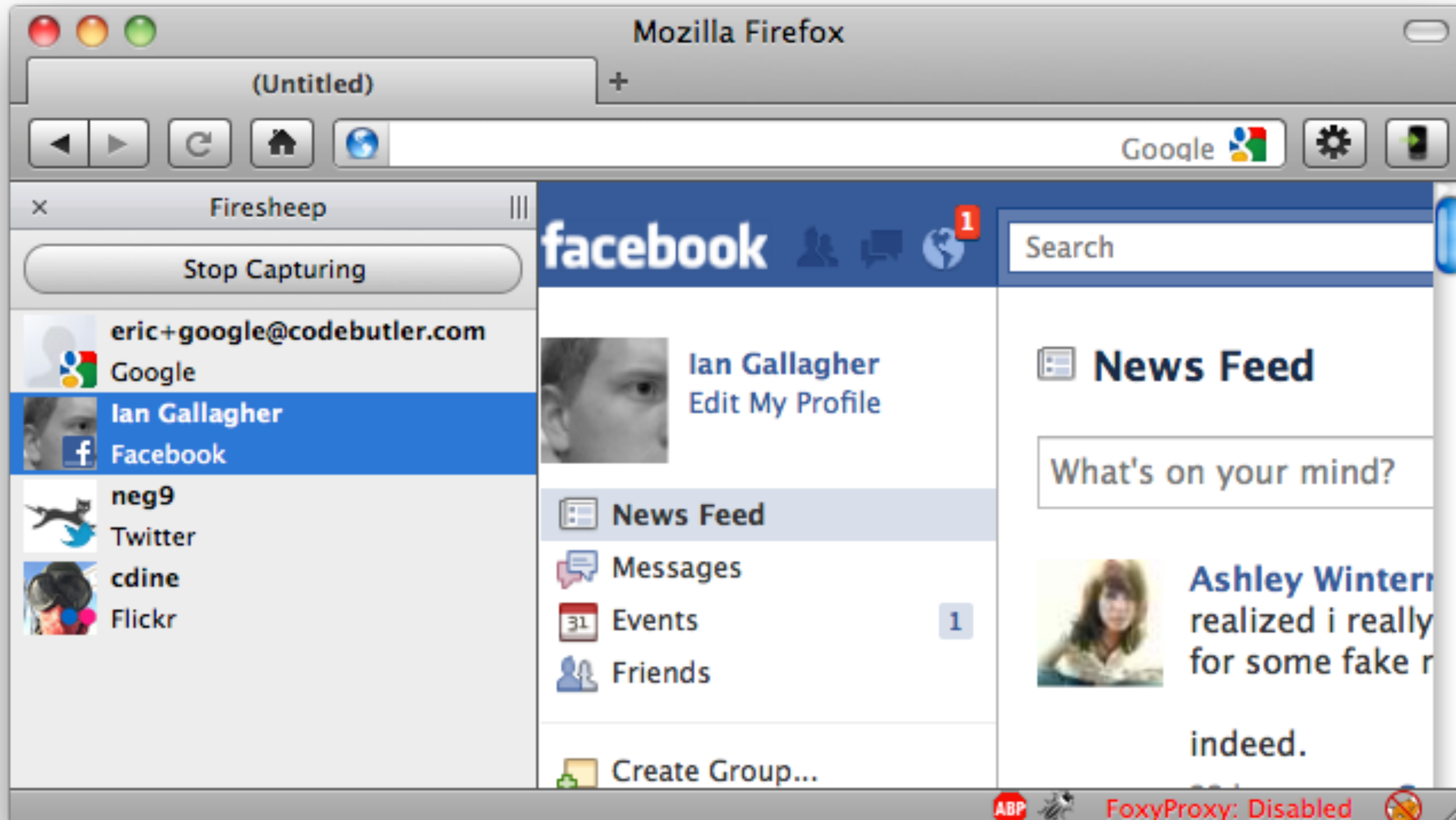
Set-Cookie: SessID=83431Adf

Nowadays increasingly all HTTPS

GET /account.html

Cookie: SessID=83431Adf

Session Hijacking



From http://codebutler.com/firesheep

Security problems here?

Facebook.com

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

Cookie: AnonSessID=134fds1431

Set-Cookie: SessID=83431Adf

Secret key K only known to server

GET /account.html

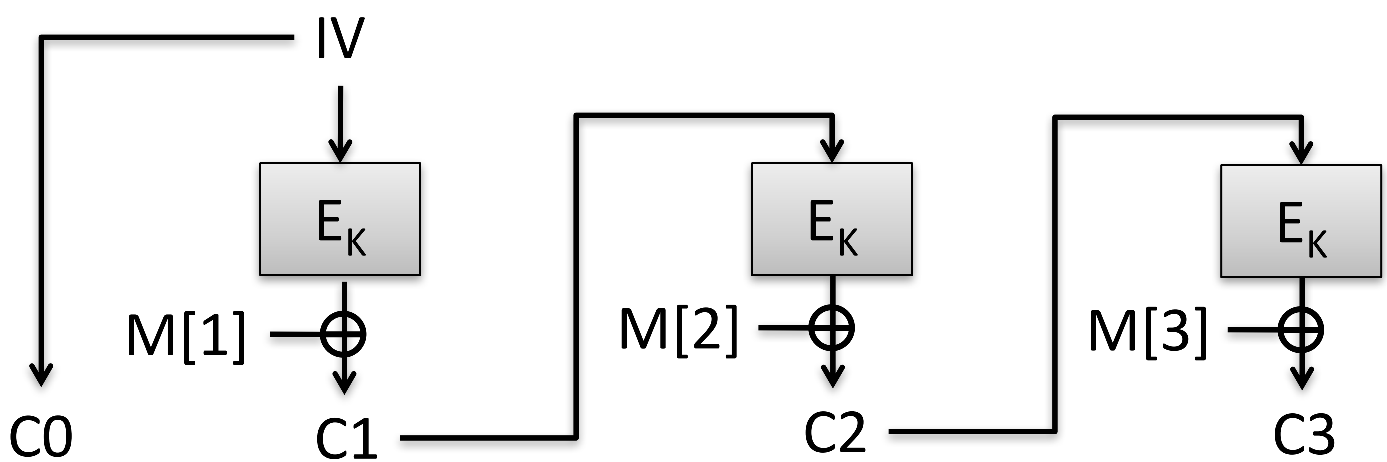
Cookie: SessID=83431Adf

83431Adf = CTR-Enc(K, “admin=0”)

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

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?

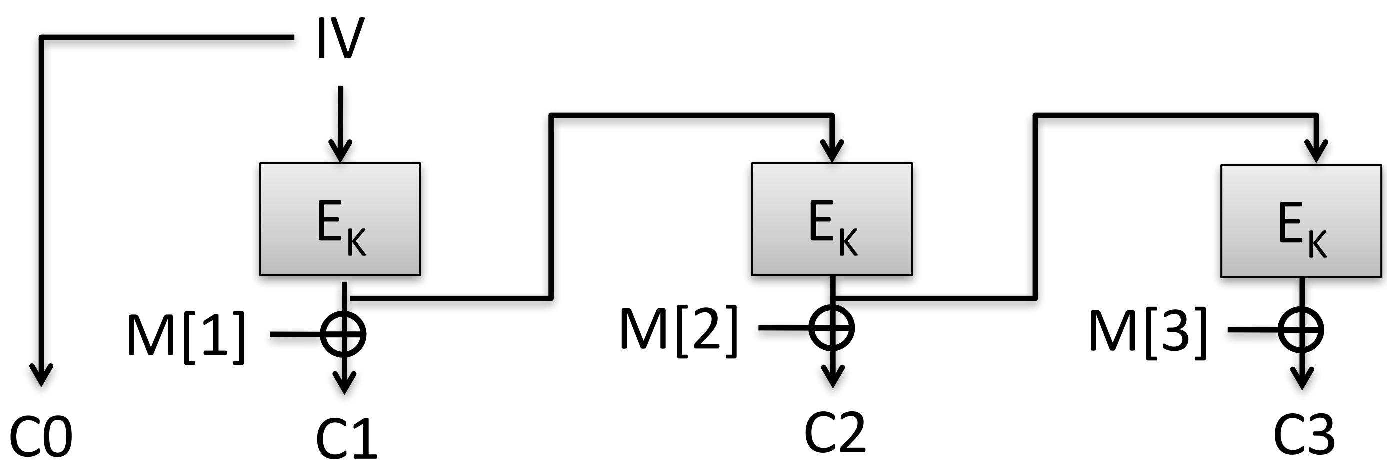
OFB mode

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?

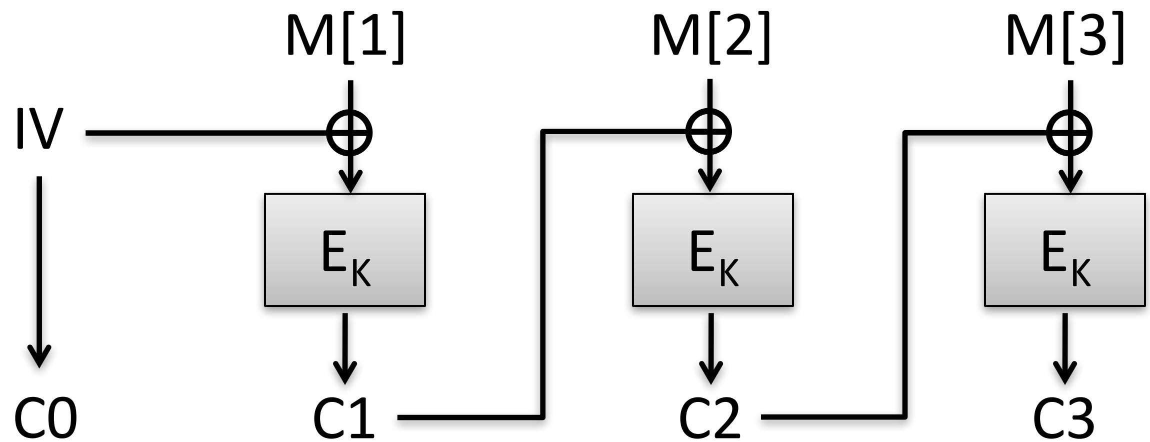
CBC mode

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?

CBC-mode SE scheme

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

Pick a random key

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

PadCBC unambiguously pads M to a string of mn bits

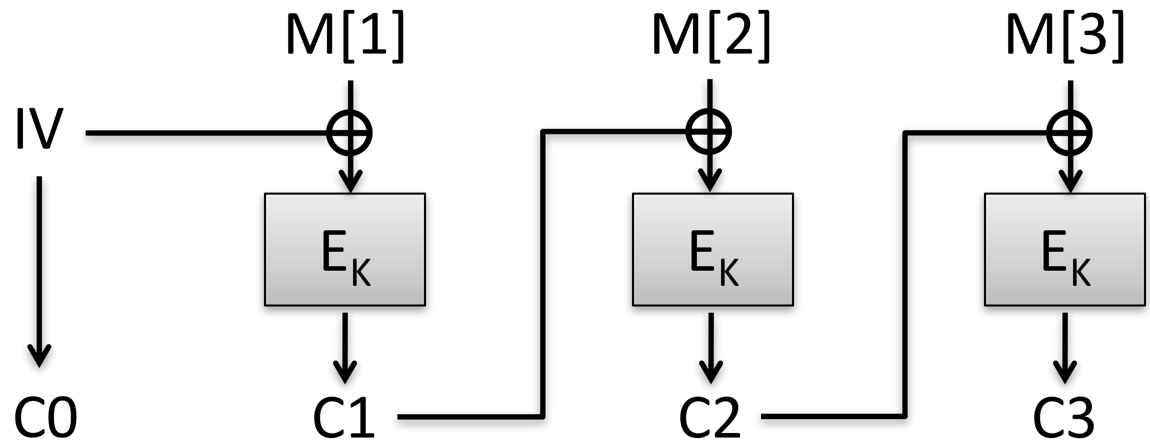
CBC-Dec($K,(C_0, C_1, \ldots, C_m)$):
For i = 1 to m do
$\quad M_i \leftarrow C_{i-1} \oplus D_K(C_i)$
$M \leftarrow \mathrm{UnpadCBC}(M_1,\ldots,M_m,n)$
Return M

UnpadCBC removes padding, returns appropriately long string

CBC-mode security

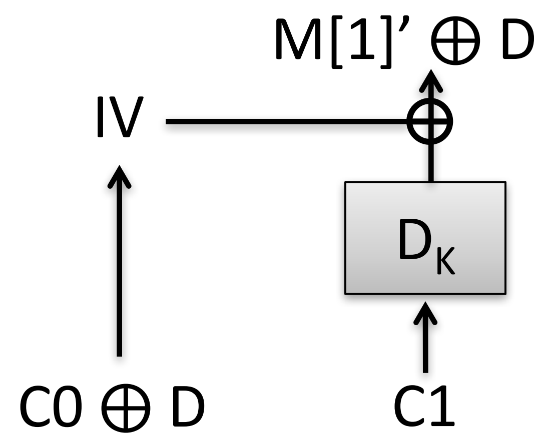
Analysis similar to CTR mode gives similar birthday-style security bound for chosen-plaintext security

CBC mode has “malleability” issues, too



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

For any D:



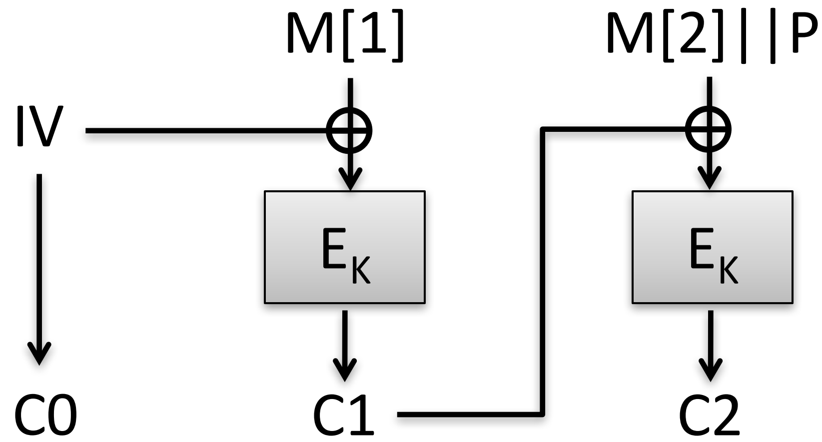
Padding for CBC mode

- 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$\oplus$1 , C2

error

<u>Dec(K, C’)</u>

M[1]’||M[2]’||P’ = CBC-Dec(K,C’)

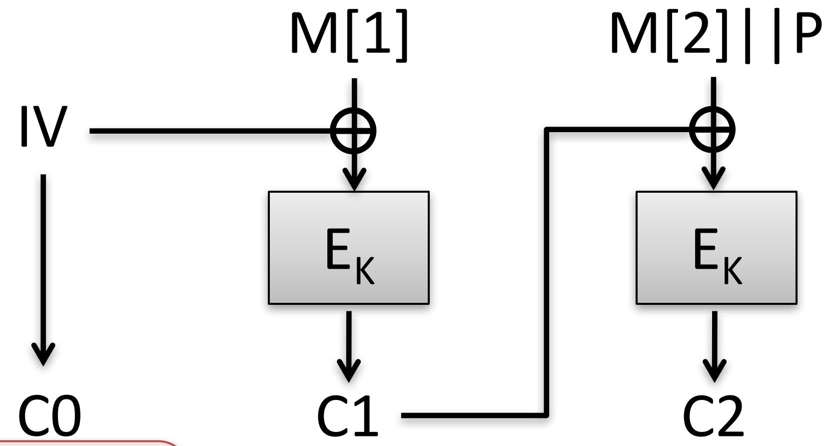
If P’ $\neq$ 0x00 then

    Return error

Else

    Return ok

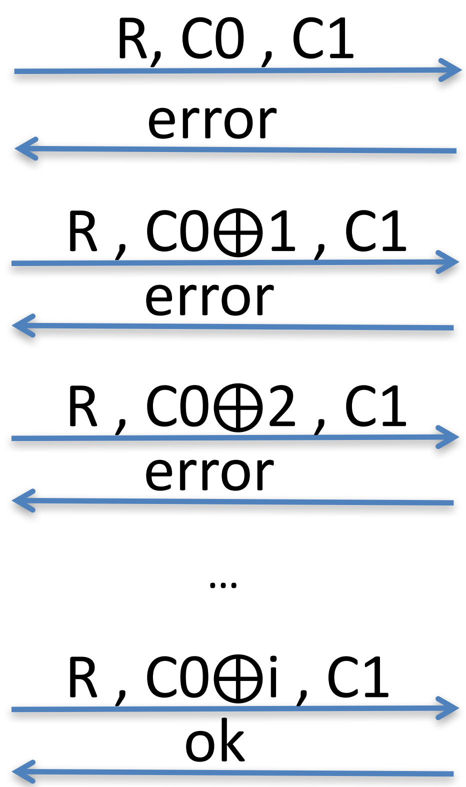
Simple situation: pad by 1 byte



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

P is one byte of padding that must equal 0x00

Low byte of M1 equals i



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

$$\text{PKCS\#7-Pad}(M) \quad = \quad M \,||\, \underbrace{P \,||\, \ldots \,||\, P}_{}$$

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

“ok” / “error” stand-ins for some other behavior:

- Passing data to application layer (web server)
- Returning other error code (if padding fails)

PKCS #7 padding oracles

Low byte of M[1] 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[m] = CBC-Dec(K,C)

P = RemoveLastByte(M[m])

while i < int(P):

    P’ = RemoveLastByte(M[m])

    If P’ != P then

        Return error

Return ok

<u>**Why?**</u> Let X[1] = D(K,C1)

C0[16] ⊕ X[1][16] = M[1][16]

C0[16] ⊕ i ⊕ X[1][16] = 01

M[1][16] ⊕ i = 01

Actually, it could be that:

M[1][16] ⊕ i = 02

Implies that M[1][15] = 02

We can rule out with an additional query

PKCS #7 padding oracles

Second lowest byte of M[1] equals $i \oplus 02$

Adversary obtains ciphertext $C = C0,C1,C2$

Let R be arbitrary n bits

R, C0 , C1 →

← error

R , $C0 \oplus 1 || j$, C1 →

← error

R , $C0 \oplus 2 || j$, C1 →

← error

…

R , $C0 \oplus i || j$, C1 →

← ok

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

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

Set $j = M[1][16] \oplus 01 \oplus 02$

Can we change decryption implementation?

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

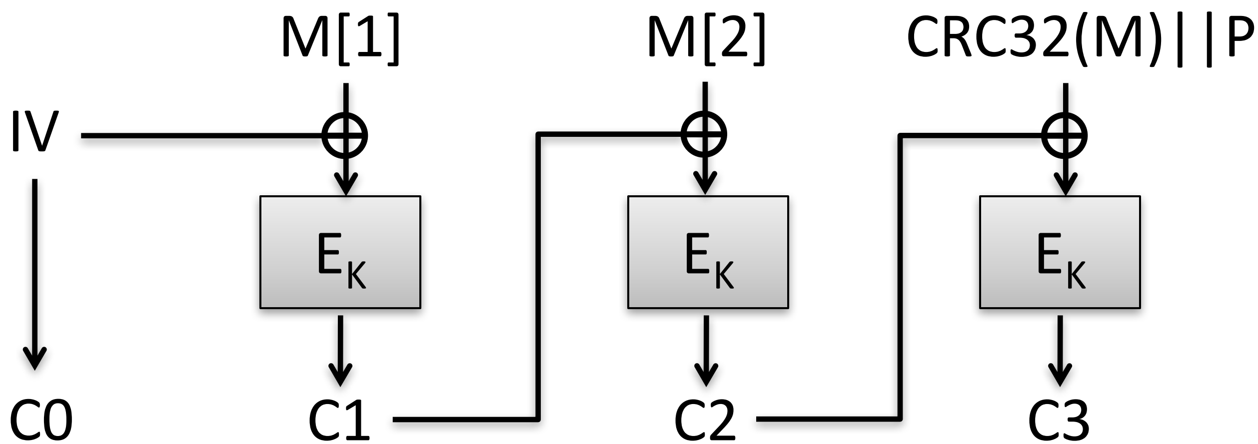
“ok” / “error” stand-ins for some other behavior:

- Passing data to application layer (web server)
- Returning other error code (if padding fails)

Chosen ciphertext attacks against CBC

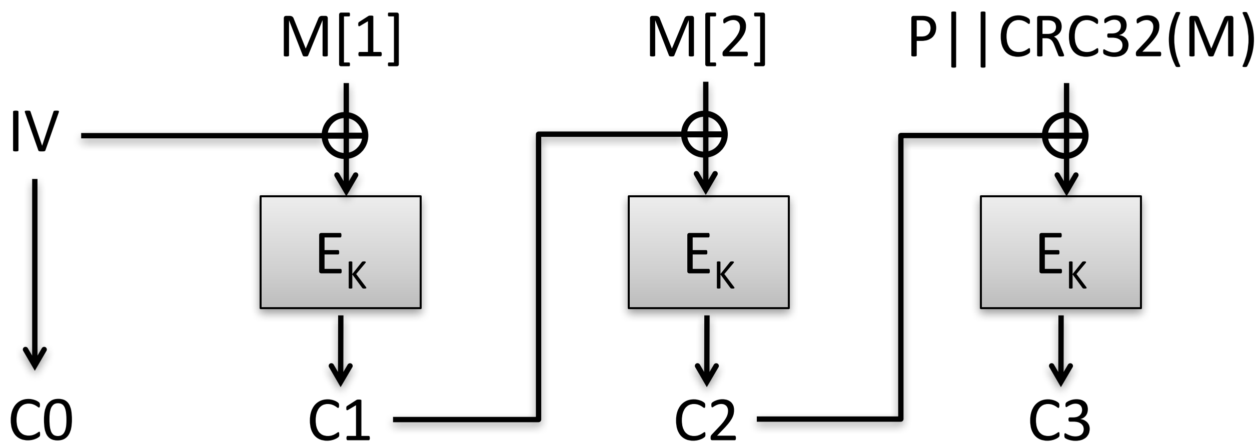
Attack	Description	Year
Vaudenay	10's of chosen ciphertexts, recovers message bits from a ciphertext. Called "padding oracle attack"	2001
Canvel et al.	Shows how to use Vaudenay's ideas against TLS	2003
Degabriele, Paterson	Breaks IPsec encryption-only mode	2006
Albrecht et al.	Plaintext recovery against SSH	2009
Duong, Rizzo	Breaking ASP.net encryption	2011
Jager, Somorovsky	XML encryption standard	2011
Duong, Rizzo	"Beast" attacks against TLS	2011
AlFardan, Paterson	Attack against DTLS	2012
AlFardan, Paterson	Lucky 13 attack against DTLS and TLS	2013
Albrecht, Paterson	Lucky microseconds against Amazon's s2n library	2016

Non-cryptographic checksums?



CRC32(M) is cyclic redundancy code checksum.
Probabilistically catches random errors
Decryption rejects if checksum is invalid

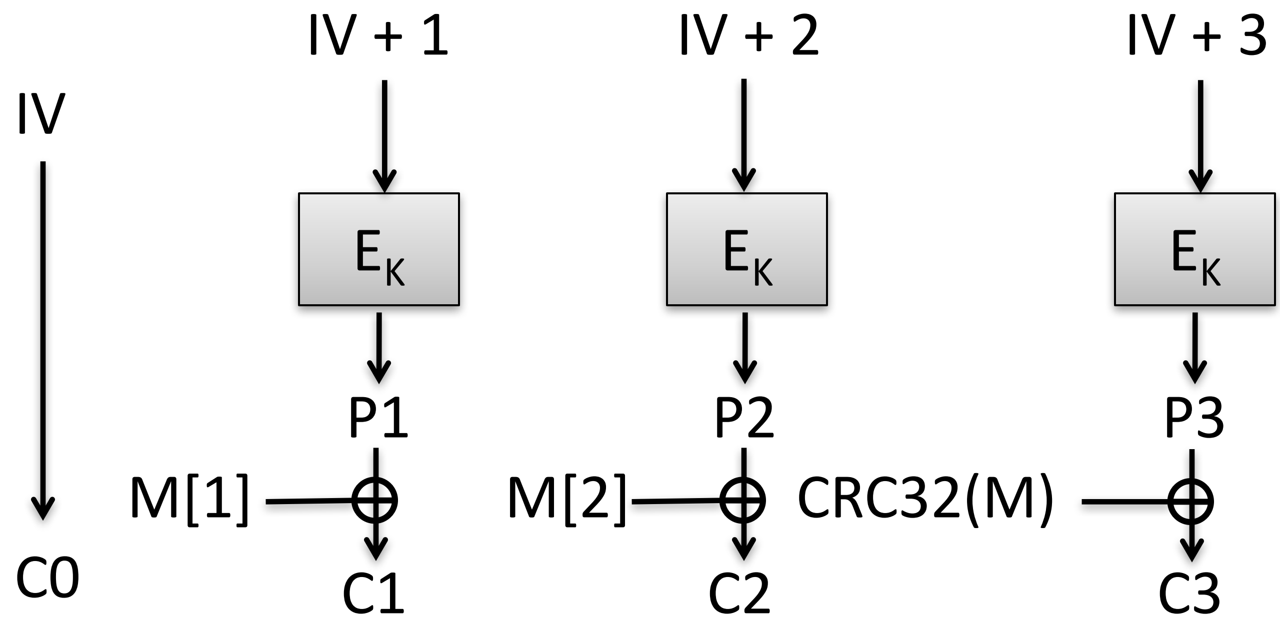
Non-cryptographic checksums?



CRC32(M) is cyclic redundancy code checksum.
Probabilistically catches random errors
Decryption rejects if checksum is invalid

Wagner sketched partial chosen plaintext, chosen ciphertext attack
(see Vaudenay 2002 paper)

Non-cryptographic checksums?



Can simply maul message and CRC32 checksum to ensure correctness

None of these modes are secure for general-purpose encryption

- 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