Secret Key:
stream ciphers & block ciphers

- Cybersecurity

Secret Key cryptography

Alice and Bob share

- A crypto protocol E
- A secret key K
- They communicate using E with key K
- Adversary knows E, knows some exchanged messages but ignores K

Two approaches:

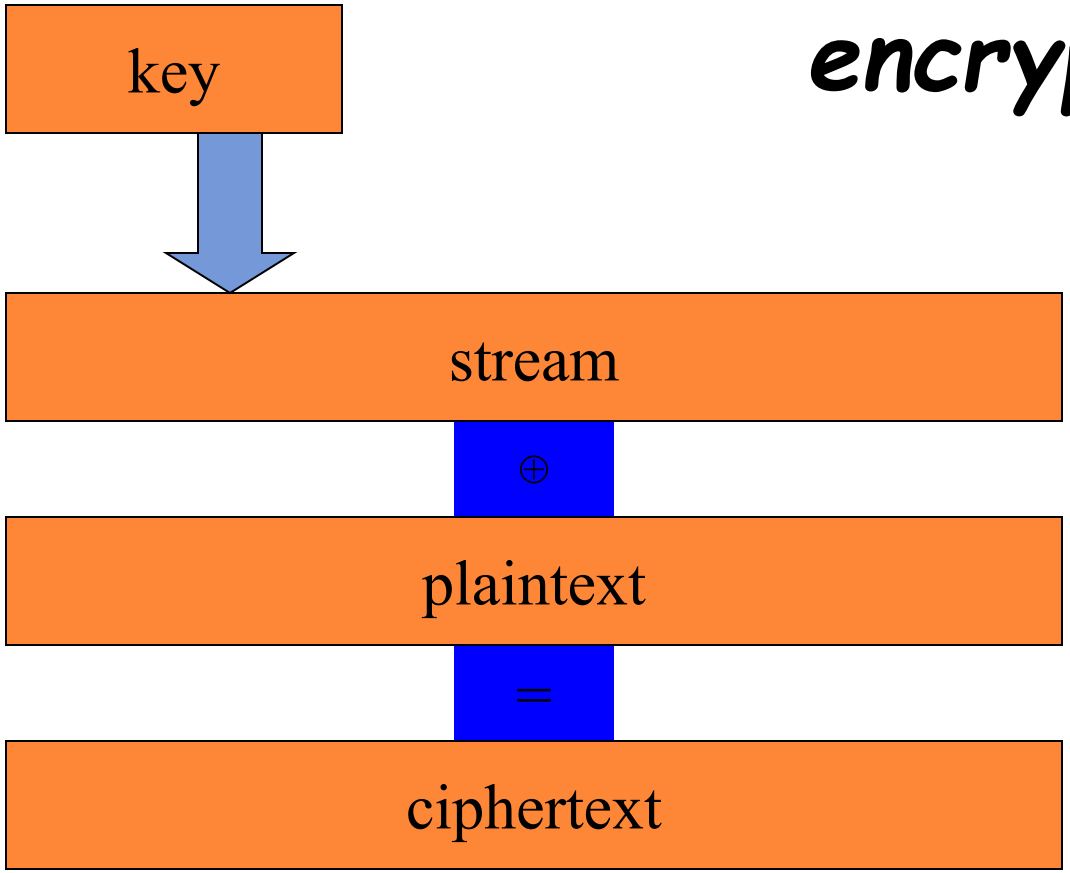
- Stream Cipher
- Block ciphers

Stream Ciphers

Idea: try to simulate *one-time pad*

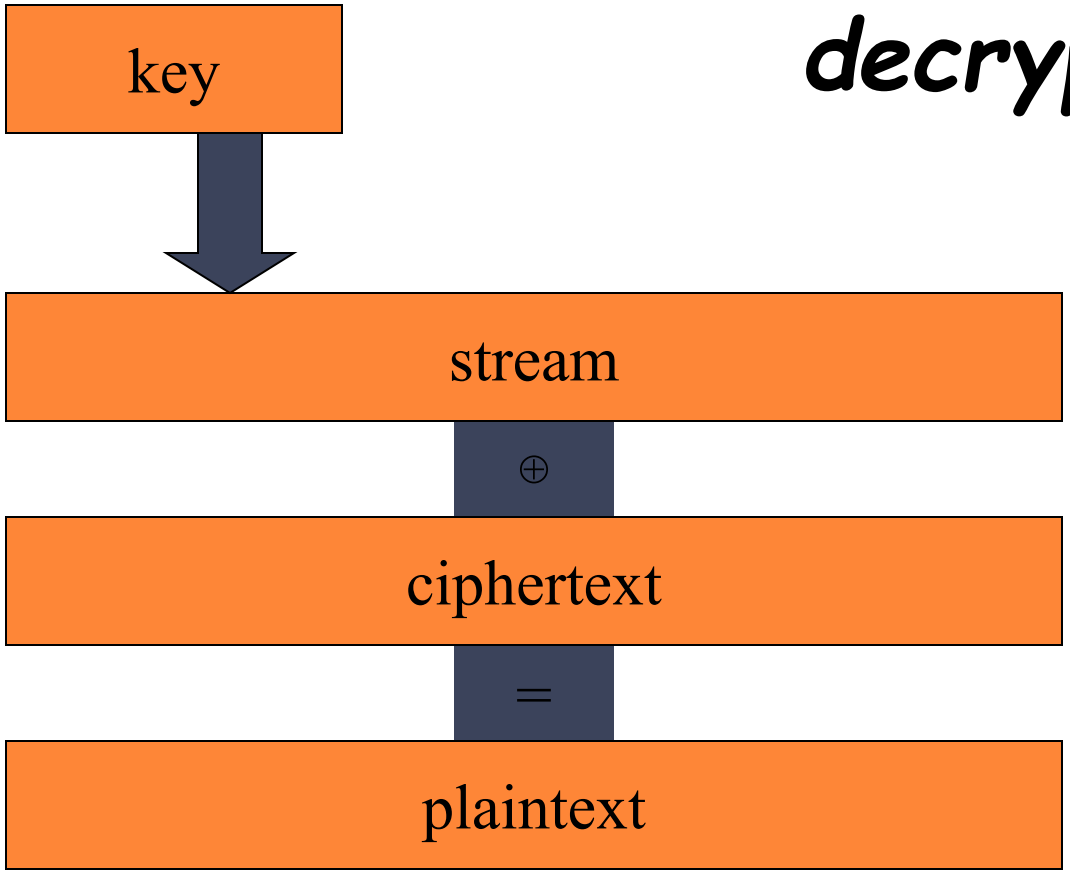
- define a secret key ("*seed*")
- using the seed generate a byte stream (*Keystream*): *i*-th byte is function of
  - only key (*synchronous* stream cipher), or
  - both key and first *i*-1 bytes of ciphertext (*asynchronous* stream cipher)
- obtain ciphertext by bitwise XORing plaintext and keystream

Synchronous Stream Cipher



encryption

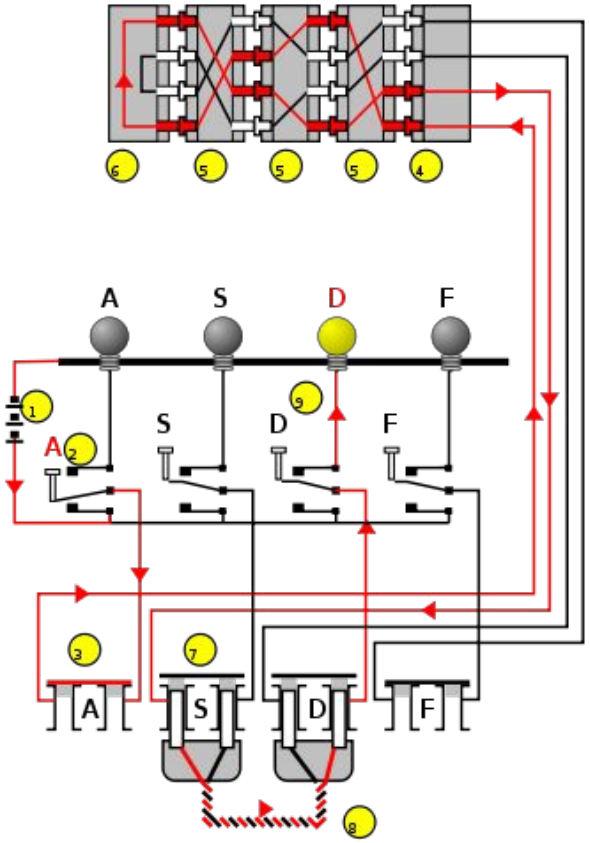
Synchronous Stream Cipher



decryption

Streams Ciphers in practice

- Many codes before 1940
- Enigma - II world war (Germany)
- A5 – GSM (encryption cell phone-base station)
- WEP - used in Ethernet 802.11 (wireless)
- RC-4 (Ron's Code)



Enigma wiring diagram showing current flow. The A key is encoded to the D lamp. D yields A, but A never yields A; this property was due to a patented feature unique to the Enigmas, and could be exploited by cryptanalysts in some .situations

A5/1

- Stream cipher (1987) used to provide over-the-air communication privacy in the GSM cellular telephone standard
- There was a terrific row between the NATO signal intelligence agencies in the mid 1980s over whether GSM encryption should be strong or not
- Used in Europe and in the United States. A5/2 was a deliberate weakening of the algorithm for certain export regions
- Initially kept secret, but the general design was leaked in 1994, and the algorithms were entirely reverse engineered in 1999 by Marc Briceno
  - A number of serious weaknesses in the cipher have been identified

RC-4

Ronald Linn Rivest

- RC: Ron's Code
  - (Ron = Ronald Rivest, MIT, born in 1947 in NY state)
- Considered safe: 1987 - 1994 kept secret, after '94 extensively studied
- Good for exporting (complying with US restrictions)
- Easy to program, fast
- Very popular: Lotus Notes, SSL, Wep etc.
- RC4's weak key schedule can give rise to a variety of serious problems

RC4: PROPERTIES

- variable key length (byte)
- synchronous
- starting from the key, it generates an apparently random permutation
- eventually the sequence will repeat
- however, long period > 10^{100} (in this way it simulates one-time-pad)
- very fast: 1 byte of output requires 8-16 instructions

RC-4 INITIALIZATION

Goal: generate a (pseudo)random permutation of the first 256 natural numbers

1. j=0
2. S_0=0, S_1=1, …, S_{255}=255
3. Assume a key of 256 bytes k_0,…,k_{255} (if the key is shorter, repeat)
4. for i=0 to 255 do
    1. j = (j + S_i + k_i) mod 256
    2. exchange S_i and S_j

In this way we obtain a permutation of 0, 1, …,255, the resulting permutation is a function of the key

RC-4 Key-stream generation

Input: permutation S of 0, 1, …, 255

1. i = 0, j = 0
2. while (true) // we'll not cycle forever, isn't it?
3. i = (i + 1) mod 256
4. j = (j + S_i) mod 256
5. exchange S_i and S_j
6. t = (S_i + S_j) mod 256
7. k = S_t // compute XOR

at every iteration compute the XOR between k and next byte of plaintext (or ciphertext)

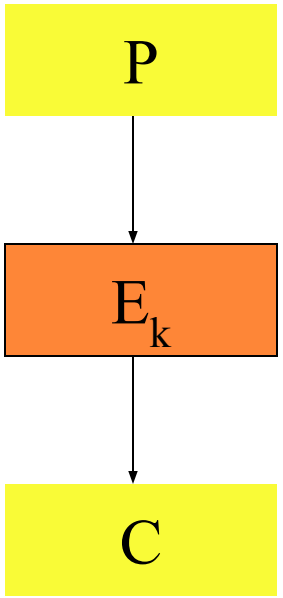
Block Ciphers

Given

- a block P of text of h bits (h fixed)
- a key k of fixed # of bits

a cryptographic protocol E_k produces a block C of h bits, function of P and k

Note: lengths of both block and key (# of bits) are fixed (not necessarily equal)



Real World Block Ciphers

- DES, 3-DES - (1976; 64 bit block, 56 bit key)
- RC-2 (1987)
  - designed for exporting cryptography within IBM Lotus Notes
  - 64 bit block, variable key size, vulnerable to an attack using 2^{34} chosen plaintexts
- IDEA (1991)
  - 64 bit block, 128 bit key
  - Strong, only weakened variants have been broken
- Blowfish (1993)
  - 64-bit block size and a variable key length from 32 up to 448 bits
  - Still strong

Real World Block Ciphers

- RC5 (1994)
  - variable block size - 32, 64 or 128 bits - key size (0 to 2040 bits) and number of rounds (0 to 255). The original suggested choice of parameters were a block size of 64 bits, a 128-bit key and 12 rounds.
  - Distributed.net has brute-forced RC5 messages encrypted with 56-bit and 64-bit keys and is working on cracking a 72-bit key; as of February 2014, 3.112% of the keyspace has been searched (it was 1.488% at March 2011). At the current rate, it will take approximately 287 years to test every possible remaining key
    - distributed.net (or Distributed Computing Technologies, Inc. or DCTI) is a worldwide distributed computing effort that is attempting to solve large scale problems using otherwise idle CPU or GPU time. It is a non-profit organization

- AES (Rijndael, 2001)
  - 128-bit block, 128-256 bit key
  - Very strong

Symmetric Block Ciphers

Standard

out in

DES AES

Historic Note

DES (data encryption standard) is a symmetric block cipher using 64 bit blocks and a 56 bit key.

Developed at IBM, approved by the US government (1976) as a standard. Size of key (56 bits) was apparently small enough to allow the NSA (US national security agency) to break it exhaustively even back in 70's.

In the 90's it became clear that DES is too weak for contemporary hardware & algorithmics (Matsui "linear attack", requires only 2^{43} known plaintext/ciphertext pairs; in 1999 Deep Crack and distributed.net break a DES key in 22 hours and 15 minutes)

LINEAR CRYPTANALYSIS

Wikipedia: *based on finding affine approximations to the action of a cipher.*

"There are two parts to linear cryptanalysis.

- The first is to construct linear equations relating plaintext, ciphertext and key bits that have a high bias; that is, whose probabilities of holding (over the space of all possible values of their variables) are as close as possible to 0 or 1.
- The second is to use these linear equations in conjunction with known plaintext-ciphertext pairs to derive key bits."

Historic Note (cont.)

The US government NIST (National Inst. of standards and technology) announced a call for an advanced encryption standard in 1997.

This was an international open competition. Overall, 15 proposals were made and evaluated, and 6 were finalists. Out of those, a proposal named Rijndael, by Daemen and Rijmen (two Belgians), was chosen in February 2001.

AES finalist	positive	negative
Rijndael	86	10
Serpent	59	7
Twofish	31	21
RC6	23	37
MARS	13	84

AES - Advanced Encryption Standard

- Symmetric block cipher (block size: 128 bits)
- Key lengths: 128, 192, or 256 bits
- Approved US standard (2001)
- Finite fields algebra

Congruence

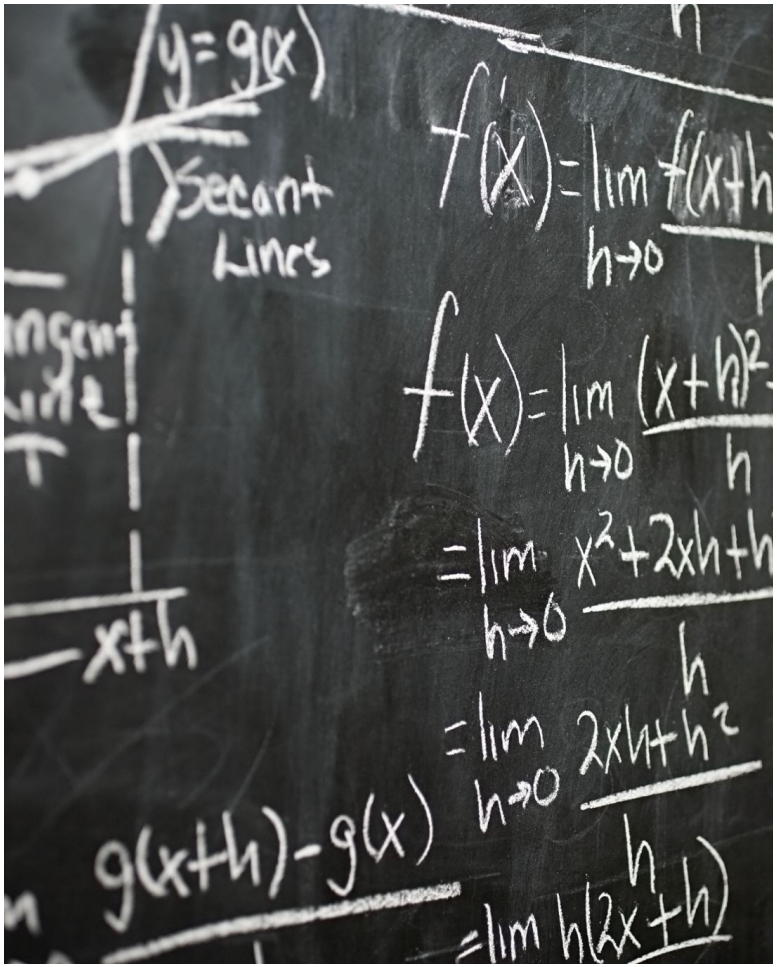
- two naturals *a* and *b* are said to be congruent modulo *n* (*n* is a positive integer)

$$a \equiv b \pmod{n}$$

  if $|a - b|$ is multiple of *n*, or, equivalently, the integer divisions of *a* and *n* and of *b* and *n* yield the same remainder
- the congruence relation is reflexive, symmetric and transitive, hence it is an equivalence relation
- the quotient set Z_n is the set of *n* classes of equivalence, congruent to 0, 1, …, *n*-1
  - -1 ≡ n - 1 (mod n), -2 ≡ n – 2 (mod n), etc.

Notation

- $Z_n = \{[0], [1], [2], \ldots, [n-1]\}$
  - here $[i]$ is the equivalence class of integers congruent to i (mod n)
  - for brevity people often write $Z_n = \{0, 1, 2, \ldots, n-1\}$
  - quotient set
- Z_m^* = natural numbers mod m that are relatively prime (co-prime) to m (multiplicative group of Z_m)



NOTEWORTHY RESULTS

Euler theorem

If a ed n are coprime positive integers (i.e. GCD(a, n) = 1) and $\varphi(n)$ is the is Euler's *totient function* (how many positive integers not greater than n are coprime with n)

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

Bézout's identity

Let a and b be nonzero integers with greatest common divisor d. Then there exist signed integers x and y such that $ax + by = d$.

x and y can be computed by the extended Euclidean algorithm.

(Used for finding the *multiplicative inverse*, for $d = 1$)

CHALLENGE (JUNE EXAM 2017)

compute

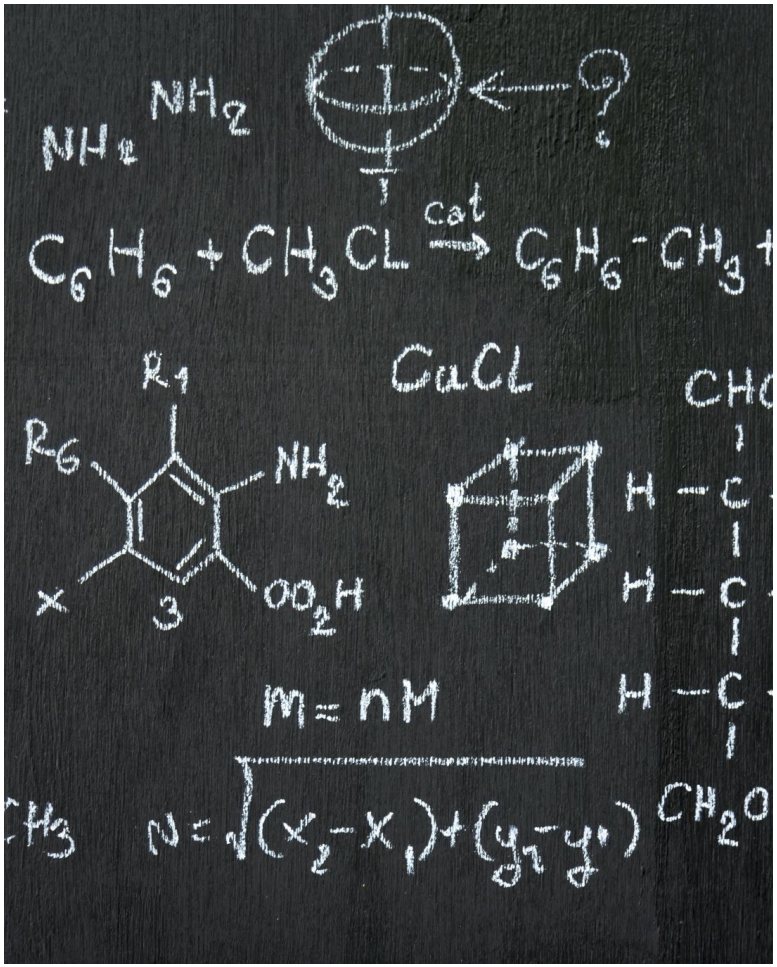
$$2^{200} \bmod 127$$

using only pen and paper….

On the totient function

from previous definitions:

$\varphi(m)$ = Euler's totient function = $|Z_m^*|$ = size of the multiplicative group of Z_m



Galois Fields GF(p^k)

Theorem: For every prime power p^k ($k = 1, 2, \ldots$) there is a unique finite field containing p^k elements. These fields are denoted by GF(p^k).

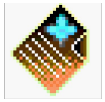
There are no finite fields with other cardinalities.

Évariste Galois (1811-1832)

Polynomials over Finite Fields

Polynomial equations and factorizations in finite .fields can be different than over the rationals

:Examples from an XMAPLE session



Maple Worksheet File

```
factor(x^6-1);  # over the rationals
```

$$(x-1)(x+1)(x^2+x+1)(x^2-x+1)$$

```
Factor(x^6-1) mod 7; # over Z7
```

$$(x+1)(x+3)(x+2)(4+x)(x+5)(x+6)$$

```
factor(x^4+x^2+x+1); # over the rationals
```

$$x^4+x^2+x+1$$

```
Factor(x^4+x^2+x+1) mod 2; # over Z2
```

$$(x+1)(x^3+x^2+1)$$

Irreducible Polynomials

A polynomial is irreducible in GF(p) if it does not factor over .GF(p). Otherwise it is reducible

:Examples

Maple Worksheet File

```
Factor(x^5+x^4+x^3+x+1) mod 5;
```

$$(x+2)(x^3+3x+2)(x+4)$$

```
Factor(x^5+x^4+x^3+x+1) mod 2;
```

$$x^5+x^4+x^3+x+1$$

.The same polynomial is reducible in Z_5 but irreducible in Z_2

Implementing GF(p^k) arithmetic

Theorem: Let f(x) be an irreducible polynomial .of degree k over Z_p

The finite field $GF(p^k)$ can be realized as the set of degree k-1 polynomials over Z_p, with addition .and multiplication done modulo f(x)

Example: Implementing $GF(2^5)$

By the theorem, the finite field $GF(2^5)$ can be realized as the set of degree 4 polynomials over Z_2, with addition and multiplication done modulo the irreducible polynomial $f(x)=x^5+x^4+x^3+x+1$.

The coefficients of polynomials over Z_2 are 0 or 1.
So, a degree k polynomial can be written down by k+1 bits:
For example, with k=4

$x^3+x+1 \longleftrightarrow (0,1,0,1,1)$

$x^4+x^3+x+1 \longleftrightarrow (1,1,0,1,1)$

Implementing $GF(2^5)$

Addition: bit-wise XOR (since 1+1=0)

x^3+x+1 (0,1,0,1,1)

+

x^4+x^3+x (1,1,0,1,0)

---

x^4+1 (1,0,0,0,1)

Implementing GF(2^5)

Multiplication: Polynomial multiplication, and then :remainder modulo the defining polynomial f(x)

```
> g(x):=(x^4+x^3+x+1)*(x^3+x+1);
```

$$g(x) := (x^4 + x^3 + x + 1)(x^3 + x + 1)$$

```
> f(x):=x^5+x^4+x^3+x+1;
```

$$f(x) := x^5 + x^4 + x^3 + x + 1$$

```
> rem(g(x),f(x),x);
```

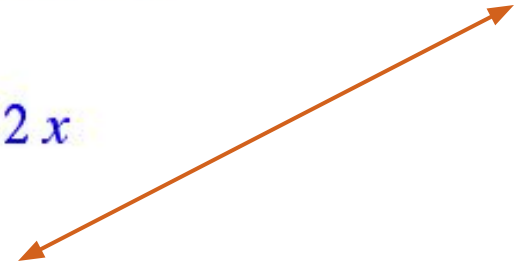
$$1 + 3x^4 + x^3 + 2x$$

```
> % mod 2;
```

$$1 + x^4 + x^3$$

$(0,1,0,1,1)^*$ $(1,1,0,1,1)$

$(1,1,0,0,1)$ =



For small size finite field, a lookup table is the most efficient .method for implementing multiplication

AES - Advanced Encryption Standard

- Symmetric block cipher
- Key lengths: 128, 192, or 256 bits

Rationale

- Resistance to all known attacks
- Speed and code compactness
  - good for devices with limited computing power, e.g. smart cards
- Simplicity

AES Specifications

- Input & output block length: 128 bits.
- State: 128 bits, arranged in a 4-by-4 matrix of bytes.

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

Each byte is viewed as an element in $GF(2^8)$

Input/Output: $A_{0,0}$, $A_{1,0}$, $A_{2,0}$, $A_{3,0}$, …, $A_{0,1}$

AES Specifications

- Key length: 128, 196, 256 bits.

Cipher Key Layout: n = 128, 196, 256 bits, arranged in a 4-by-n/32 matrix of bytes.

$K_{0,5}$	$K_{0,4}$	$K_{0,3}$	$K_{0,2}$	$K_{0,1}$	$K_{0,0}$
$K_{1,5}$	$K_{1,4}$	$K_{1,3}$	$K_{1,2}$	$K_{1,1}$	$K_{1,0}$
$K_{2,5}$	$K_{2,4}$	$K_{2,3}$	$K_{2,2}$	$K_{2,1}$	$K_{2,0}$
$K_{3,5}$	$K_{3,4}$	$K_{3,3}$	$K_{3,2}$	$K_{3,1}$	$K_{3,0}$

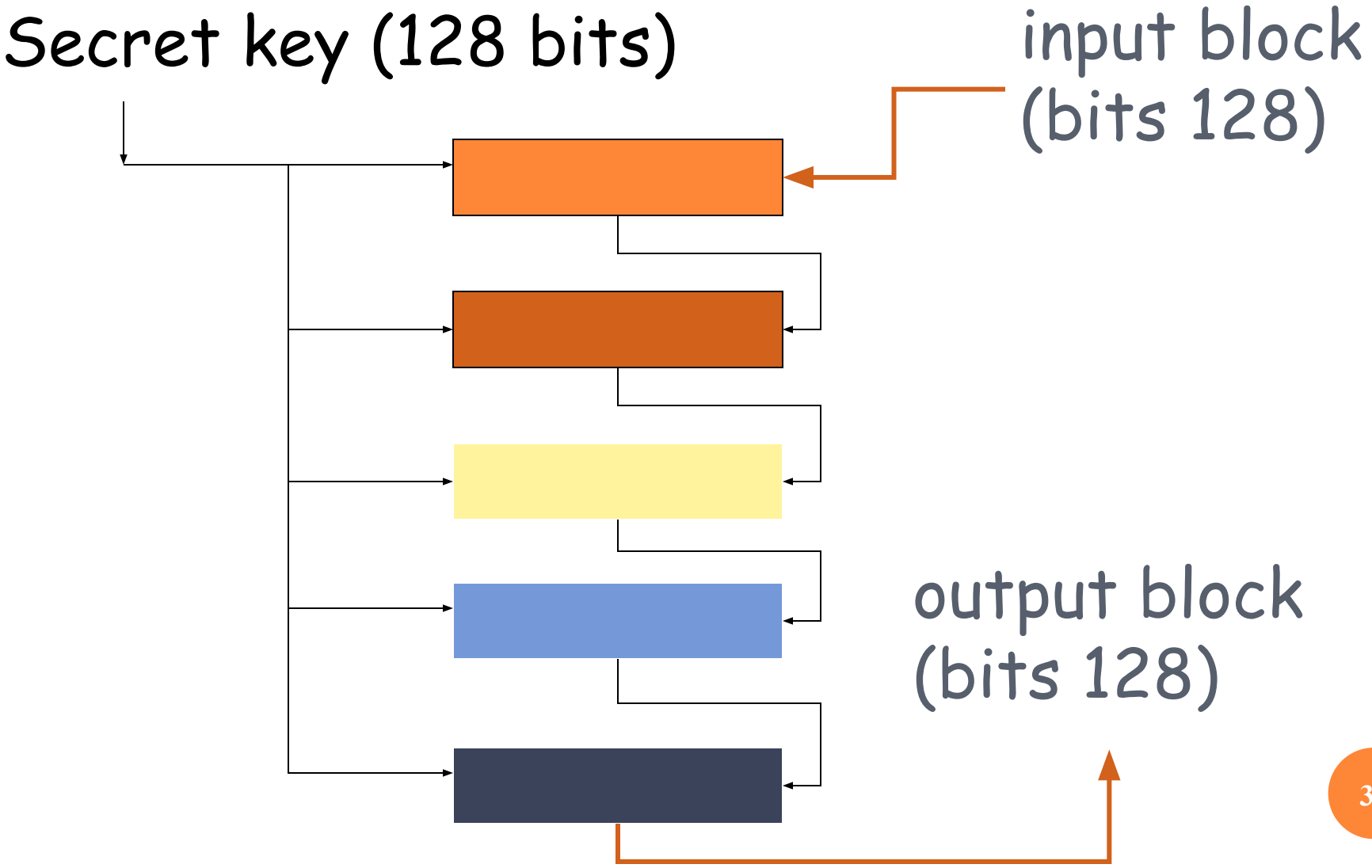
Initial layout: $K_{0,0}$, $K_{1,0}$, $K_{2,0}$, $K_{3,0}$, …, $K_{0,1}$

AES Specifications

High level code

- AES(State, Key)
  - KeyExpansion(Key, ExpandKey)
  - AddRoundKey(State, ExpandKey[0])
  - for (i = 1; i < R; i++) do
    Round(State, ExpandKey[i]);
  - FinalRound(State, ExpandKey[R]);

Encryption: Carried out in rounds



Rounds in AES

bits AES uses 10 rounds, no shortcuts 128 known for 6 rounds

- The secret key is expanded from 128 bits to 10 round keys, 128 bits each.
- Each round changes the state, then XORs the round key. (for longer keys, add one round for every extra 32 bits)

.Each rounds complicates things a little
Overall it seems infeasible to invert without .the secret key (but easy given the key)

AES Specifications: One Round

:Transform the state by applying

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

1. Substitution
2. Shift rows
3. Mix columns
4. XOR round key

Substitution (S-Box)

Substitution operates on every Byte separately: $A_{i,j} \leftarrow A_{i,j}^{-1}$ multiplicative inverse in GF(2^8)) (which is highly non linear

If $A_{i,j} = 0$, don't change $A_{i,j}$

.Clearly, the substitution is invertible

Cyclic Shift of Rows

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

no shift
shift 1 position
shift 2 positions
shift 3 positions

.Clearly, the shift is invertible

Mixing Columns

Every state column is considered as a Polynomial over $GF(2^8)$

Multiply with an invertible polynomial
$x^3 + 01x^2 + 01x + 02 \pmod{x^4 + 1}$ 03
Inv = 0B x^3 + 0D x^2 +09 x + 0E

Round: SubBytes(State)
ShiftRows(State)
MixColumns(State)
AddRoundKey(State,ExpandedKey[i])

Key Expansion

- Generate a "different key" per round
- Need a 4 x 4 matrix of values (over $GF(2^8)$) per round
- Based upon a non-linear transformation of the original key.
- Details available: *The Design of Rijndael*, Joan Daemen and Vincent Rijmen, Springer

- animation

Breaking AES

.Breaking 1 or 2 rounds is easy

.It is not known how to break 5 rounds

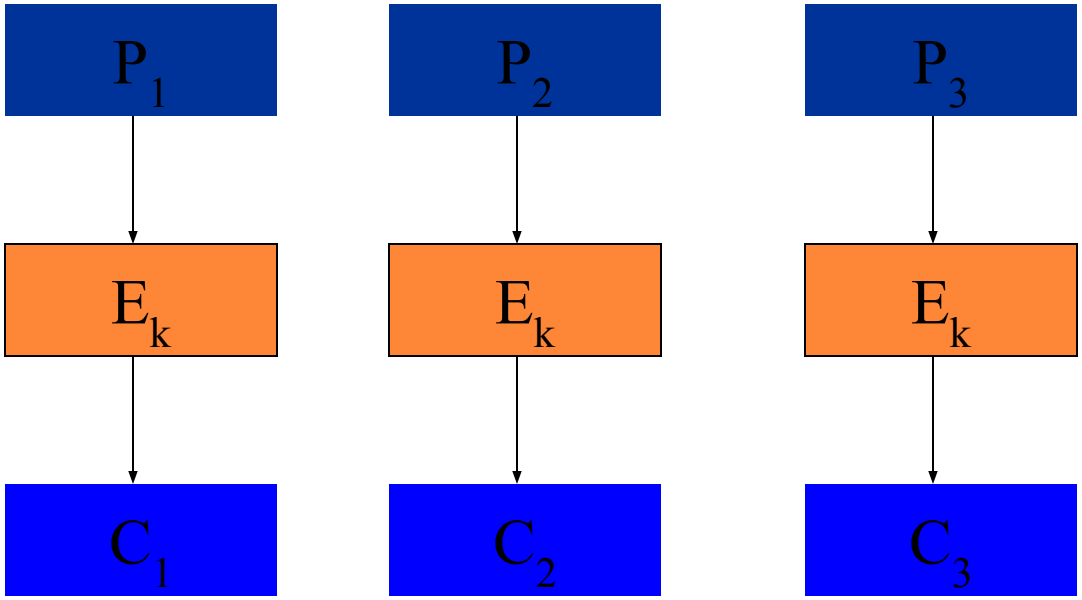
Breaking the full 10 rounds AES efficiently say 1 year on existing hardware, or in) less than 2^{128} operations) is considered impossible ! (a good, tough challenge…)

Block cipher modes of operation

- block ciphers operate on blocks of fixed length, often 64 or 128 bits
- because messages may be of any length, and because encrypting the same plaintext under the same key always produces the same output,

several modes of operation have been invented which allow block ciphers to provide confidentiality for messages of arbitrary length

ECB Mode Encryption (Electronic Code Book)



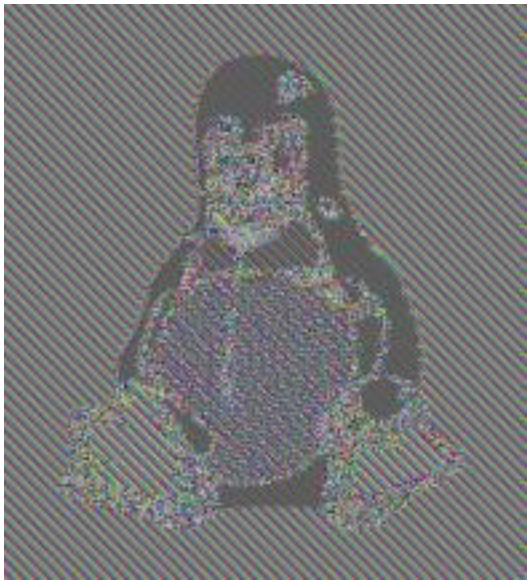
encrypt each plaintext block separately

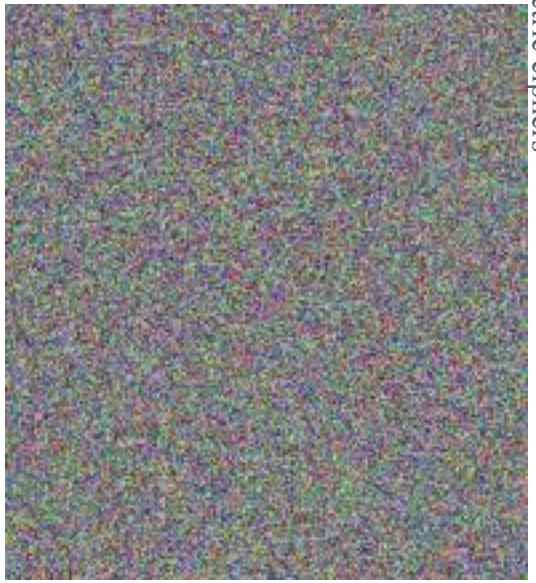
Properties of ECB

- Simple and efficient
- Parallel implementation possible
- Does not conceal plaintext patterns
- Active attacks are possible (plaintext can be easily manipulated by removing, repeating, or interchanging blocks).

ECB: PLAINTEXT REPETITIONS

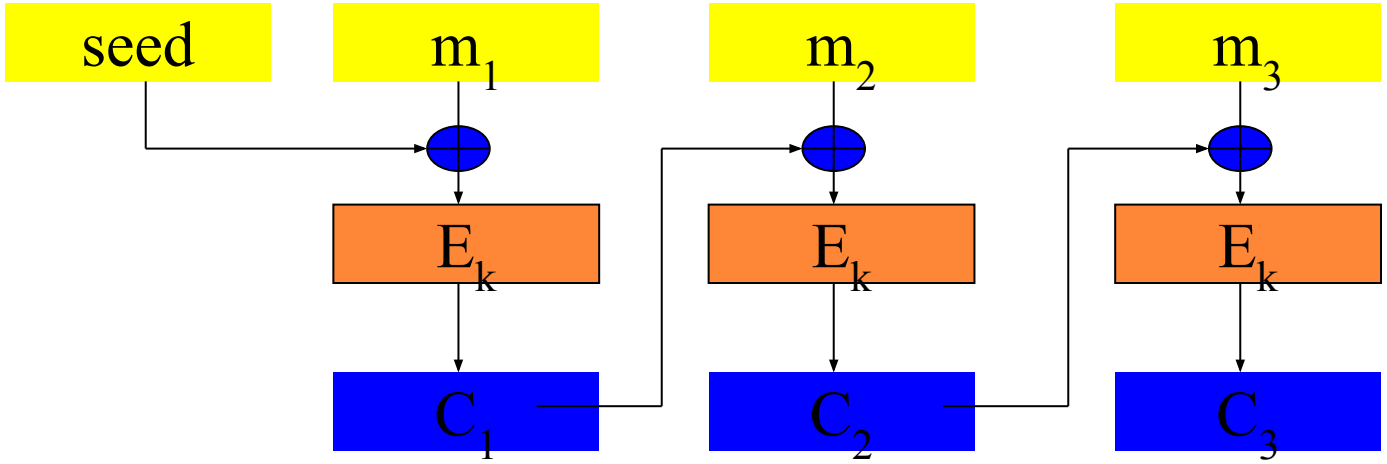
plaintext ciphertext ECB good ciphertext





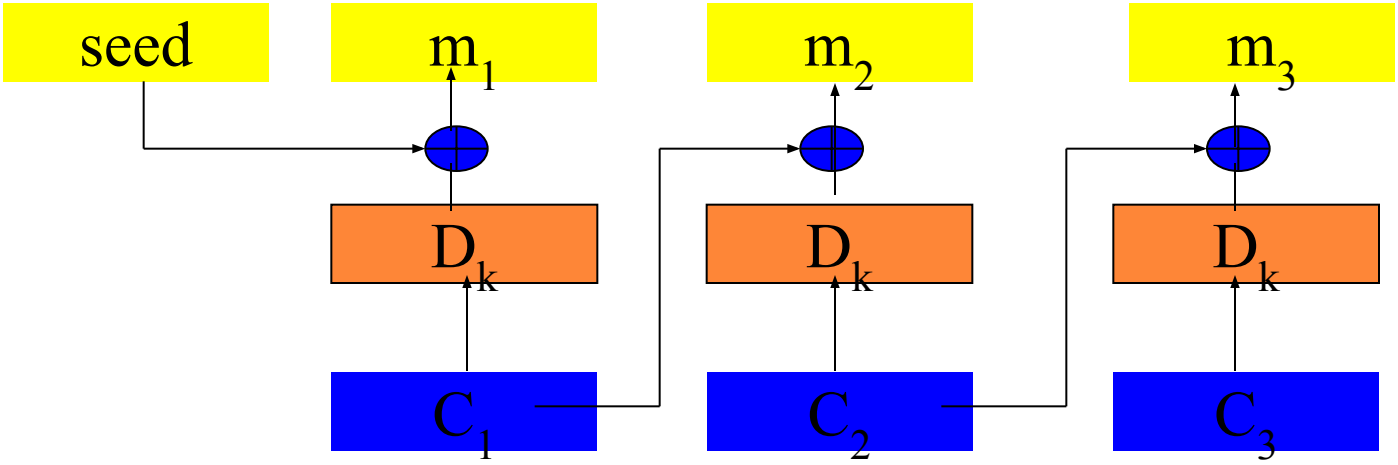
CBC (Cipher Block Chaining) mode

IBM, 1976



- Previous ciphertext is XORed with current plaintext before encrypting current block
- Seed is used to start the process; it can be sent without encryption
- Seed = 0 safe in most but NOT all cases (e.g. assume the file with salaries is sent once a month, with the same seed we can detect changes in the salaries) therefore a random seed is better

CBC (Cipher Block Chaining): decryption



Problem

IF a transmission error changes one bit of $C_{(i-1)}$

THEN block m_i changes in a predictable way (this can be exploited by adversary)

;BUT there are unpredictable changes in $m_{(i-1)}$

Solution: always use error detecting codes (for example CRC) to check quality of transmission

Properties of CBC

- Asynchronous stream cipher
- Errors in one ciphertext block propagate
  - a one-bit change to the ciphertext causes complete corruption of the corresponding block of plaintext, and inverts the corresponding bit in the following block of plaintext
- Conceals plaintext patterns
- No parallel implementation known
  - no parallel encryption
  - what about decryption?
- Plaintext cannot be easily manipulated
- Standard in most systems: SSL, IPSec etc.

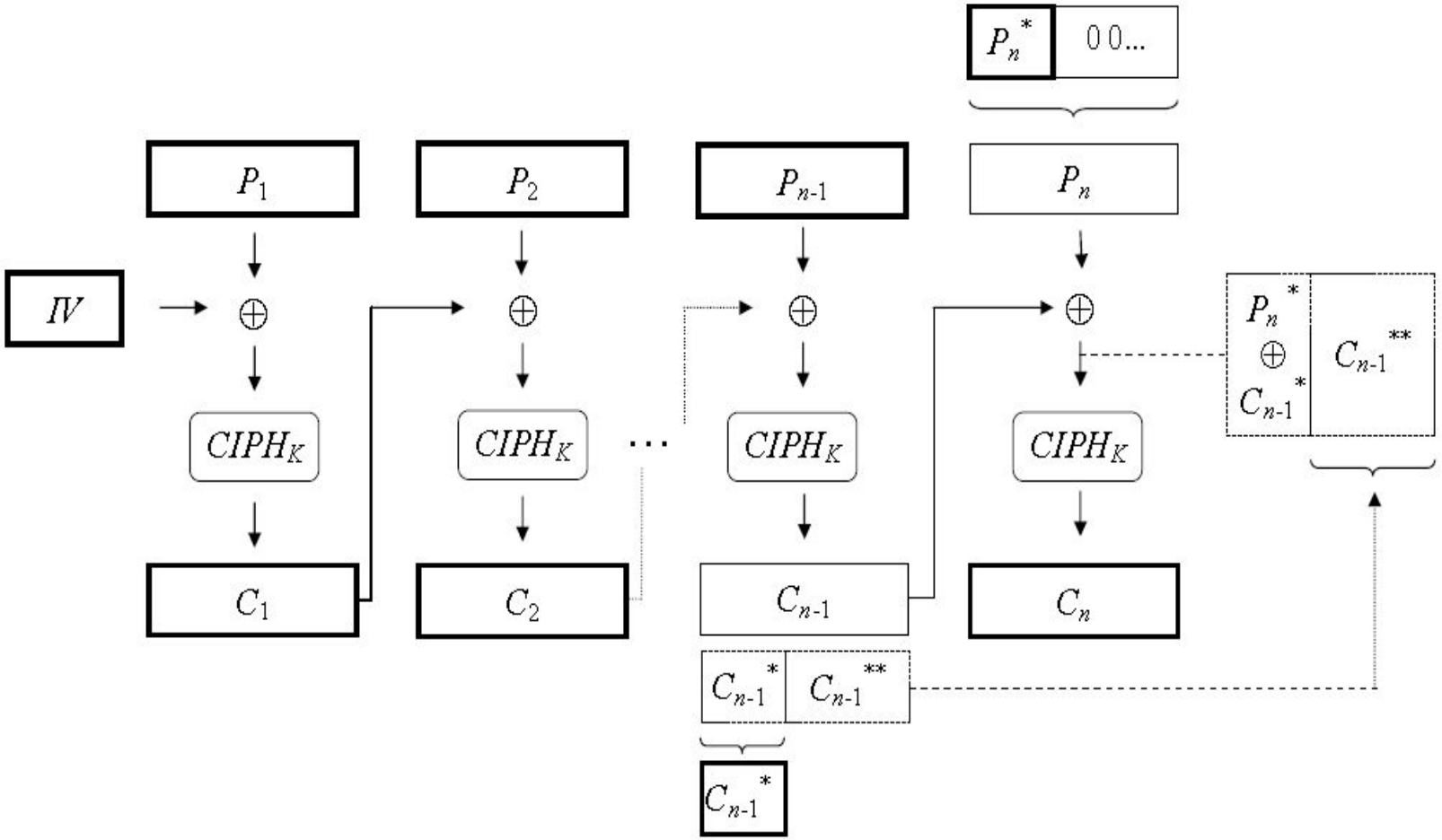
MORE ON CBC

- message must be padded to a multiple of the cipher block size
  - one way to handle this issue is ciphertext stealing
- a plaintext can be recovered from just two adjacent blocks of ciphertext
  - as a consequence, decryption can be parallelized
  - usually a message is encrypted once, but decrypted many times

CIPHERTEXT STEALING

- general method that allows for processing of messages that are not evenly divisible into blocks
  - without resulting in any expansion of the ciphertext
  - at the cost of slightly increased complexity
- consists of altering processing of the last two blocks of plaintext, resulting in a reordered transmission of the last two blocks of ciphertext (and no ciphertext expansion)
- suitable for ECB and CBC
- from NIST website http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/ciphertext%20stealing%20proposal.pdf

CIPHERTEXT STEALING SCHEME



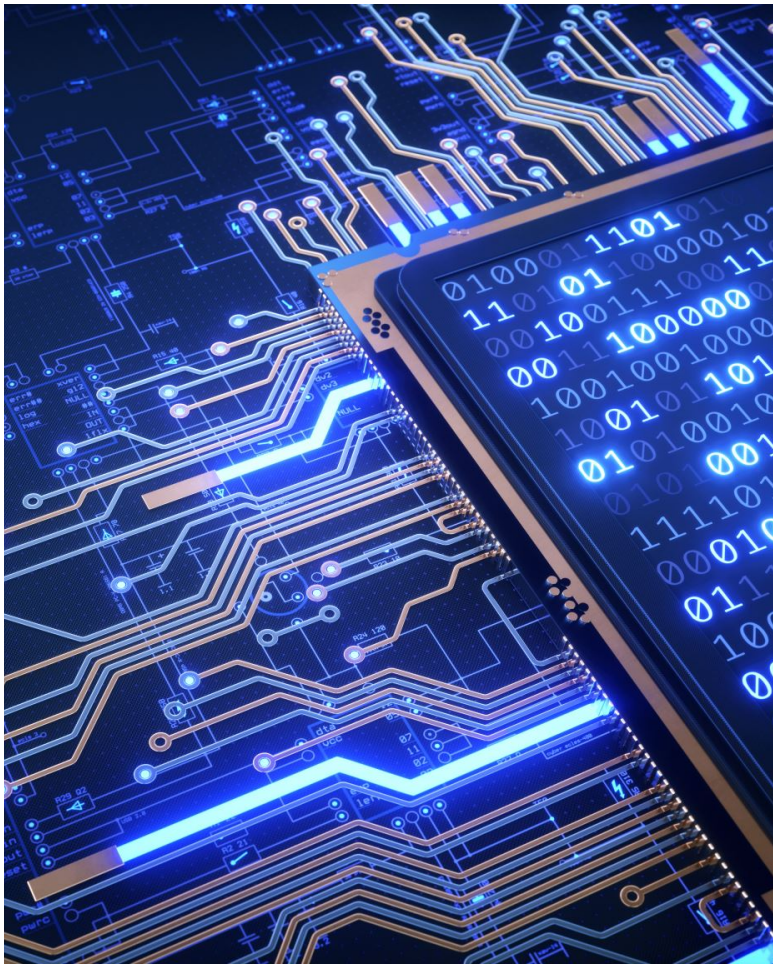
ENCRYPTION/DECRYPTION

Encryption procedure

- If the plaintext length is not a multiple of the block size, pad it with enough zero bits until it is.
- Encrypt the plaintext using the Cipher Block Chaining mode.
- Swap the last two ciphertext blocks.
- Truncate the ciphertext to the length of the original plaintext.

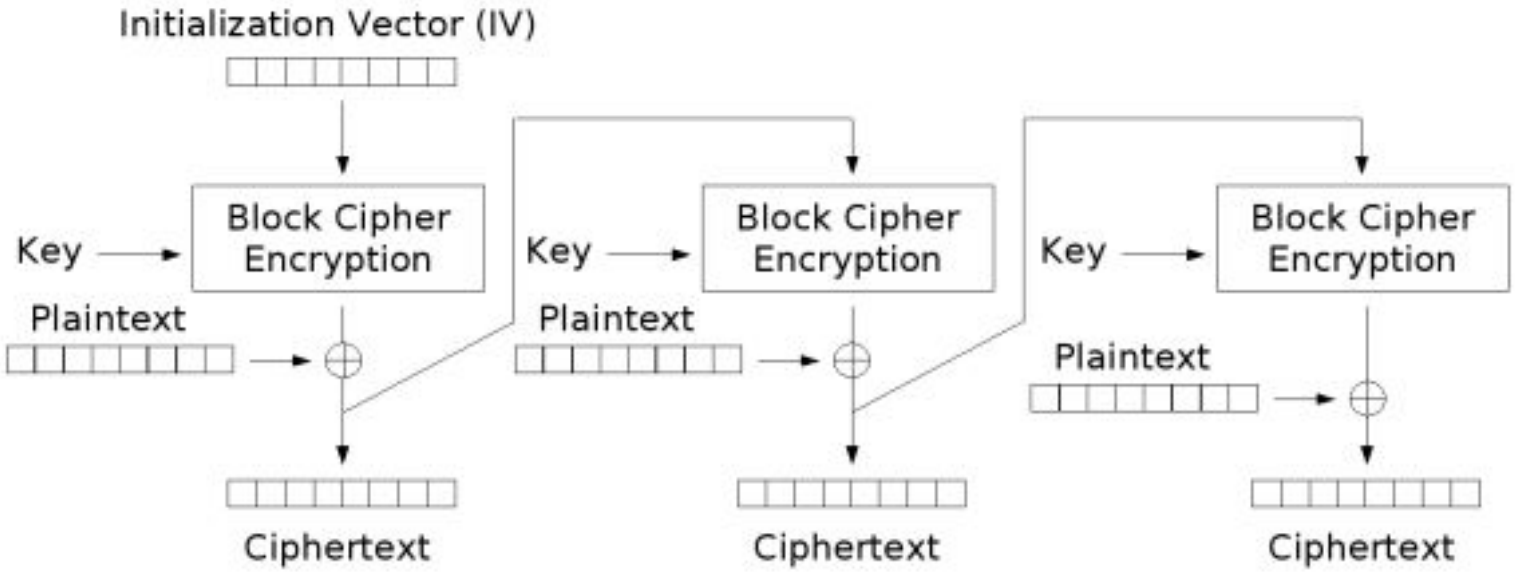
Decryption procedure

- If the ciphertext length is not a multiple of the block size, say it is n bits short, then pad it with the last n bits of the block cipher decryption of the last full ciphertext block.
- Swap the last two ciphertext blocks.
- Decrypt the ciphertext using the Cipher Block Chaining mode.
- Truncate the plaintext to the length of the original ciphertext.



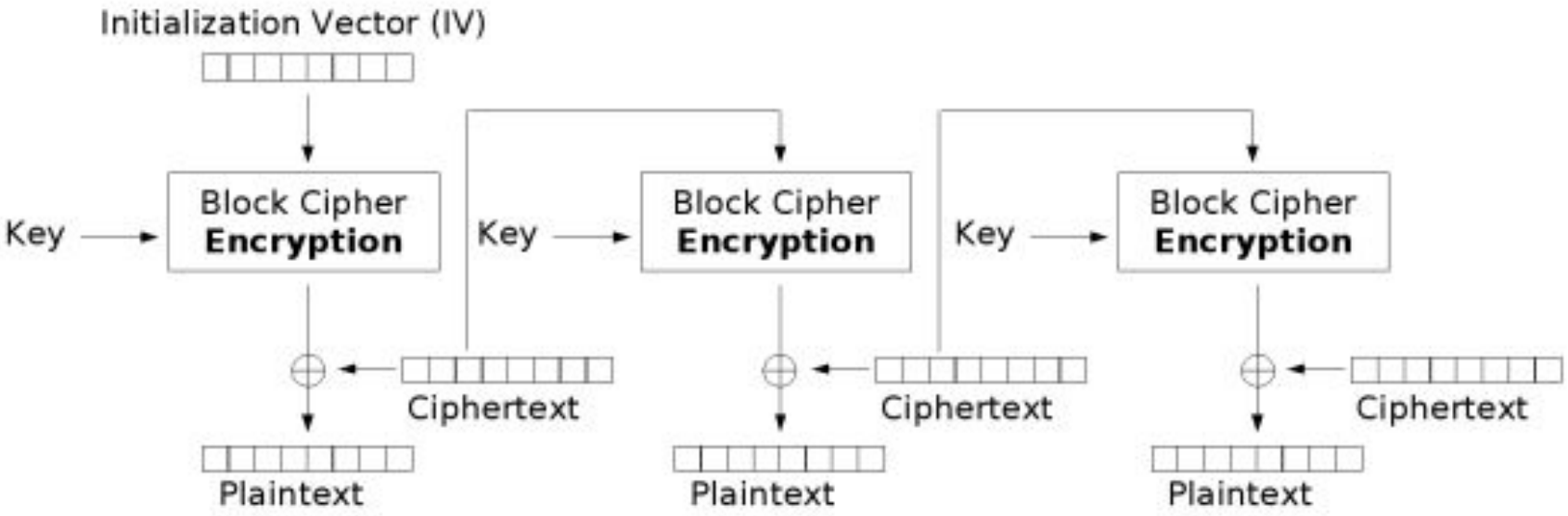
CIPHER FEEDBACK (CFB)

- like CBC, makes a block cipher into an asynchronous stream cipher
  - i.e., supports some re-synchronizing after error, if input to encryptor is given thru a shift-register



Cipher Feedback (CFB) mode encryption

CFB DECRYPTION



Cipher Feedback (CFB) mode decryption

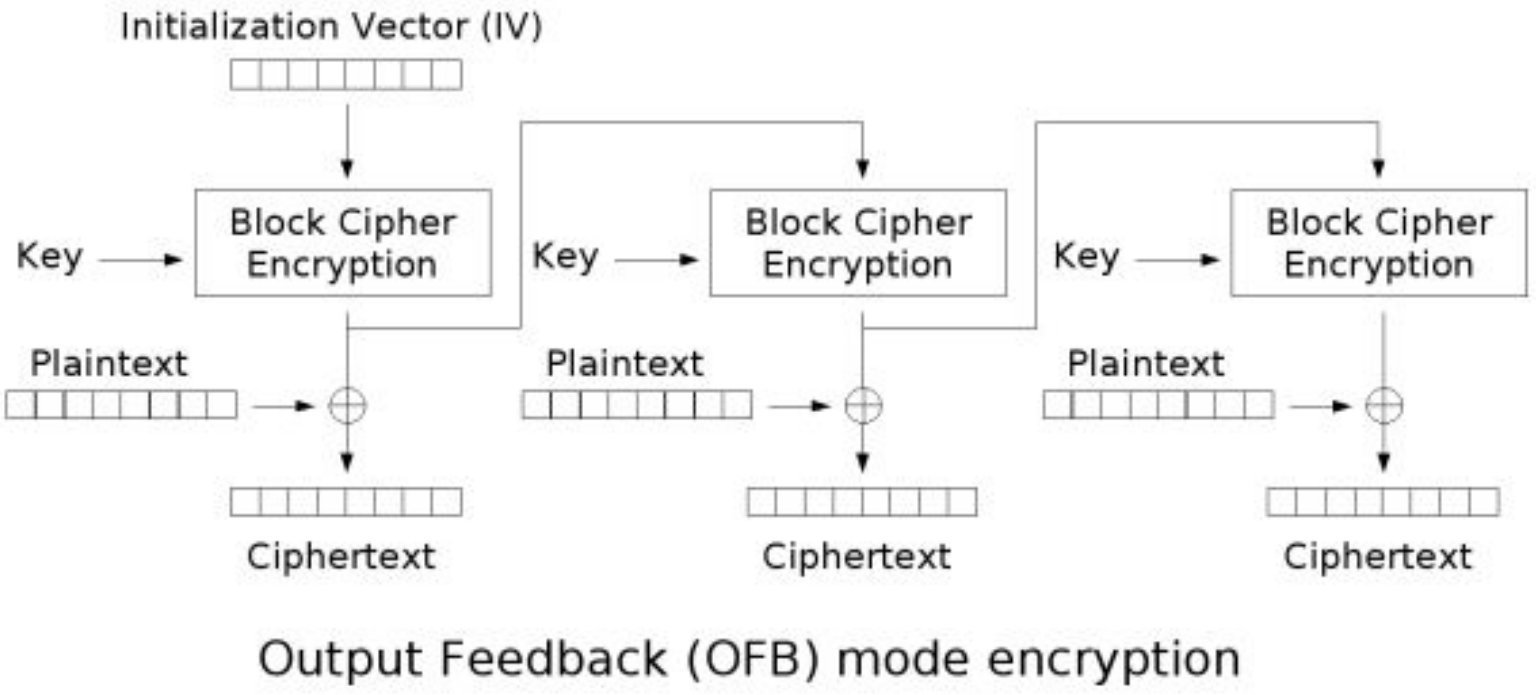
CFB

- Like CBC mode, changes in the plaintext propagate forever in the ciphertext, and encryption cannot be parallelized.
  - Also, like CBC, decryption can be parallelized.
- When decrypting, a one-bit change in the ciphertext affects two plaintext blocks: a one-bit change in the corresponding plaintext block, and complete corruption of the following plaintext block. Later plaintext blocks are decrypted normally.
- CFB shares two advantages over CBC mode: the block cipher is only ever used in the encrypting direction, and the message does not need to be padded to a multiple of the cipher block size.

OFB Mode (Output FeedBack)

- Makes a block cipher into a synchronous stream cipher: it generates keystream blocks, which are then XORed with the plaintext blocks to get the ciphertext.
- Flipping a bit in the ciphertext produces a flipped bit in the plaintext at the same location. This property allows many error correcting codes to function normally even when applied before encryption.

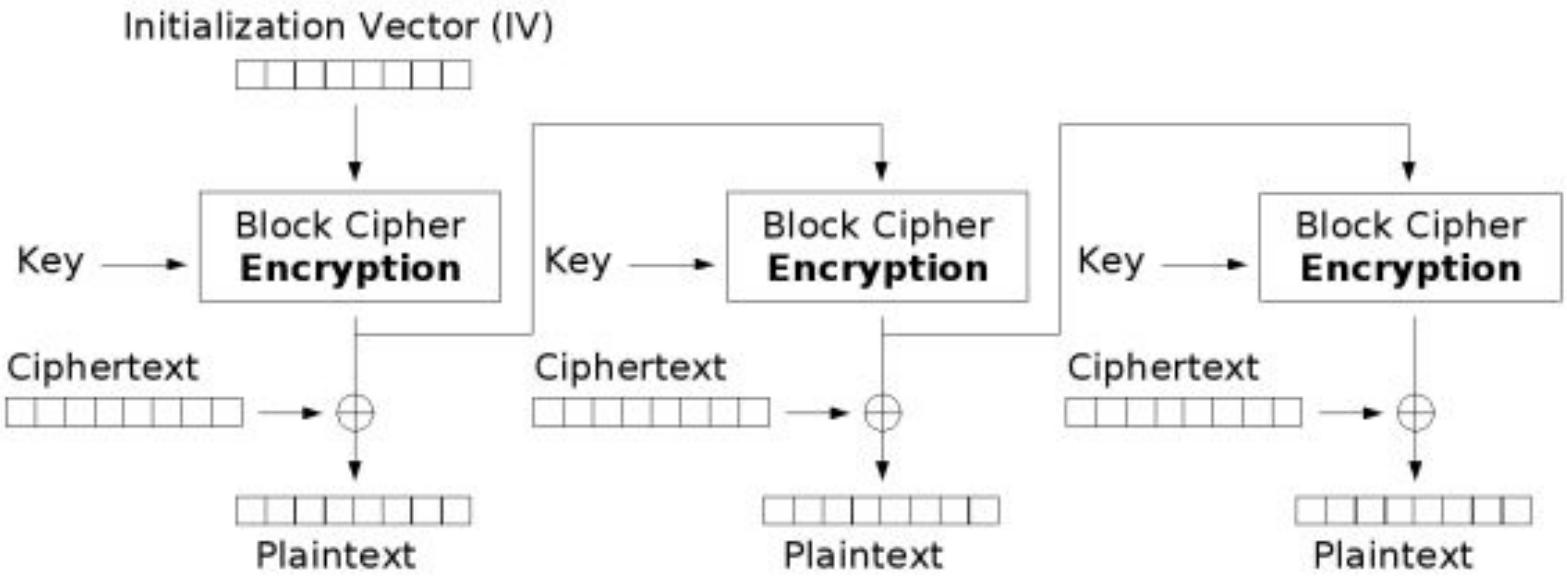
OFB SCHEME



Output Feedback (OFB) mode encryption

An initialization vector IV is used as a "seed" for a sequence of data blocks

OFB DECRYPTION



Output Feedback (OFB) mode decryption

OFB PROPERTIES

Because of the symmetry of the XOR operation, encryption and decryption are exactly the same

$$C_i = P_i \oplus O_i$$
$$P_i = C_i \oplus O_i$$
$$O_i = E_k(O_{i-1})$$
$$O_0 = \text{IV}$$

OFB MODE

Discussion

- If E_k is public (known to the adversary) then initial seed must be encrypted (why?)
- If E is a cryptographic function that depends on a secret key, then initial seed can be sent in clear (why?)
- Initial seed must be modified for EVERY new message - even if it is protected and unknown to the adversary (in fact if the adv knows a pair message, initial seed then he can encrypt every message - why?)
- Extension: it can be modified in such a way that only k bits are used to compute the ciphertext (k-OFB)

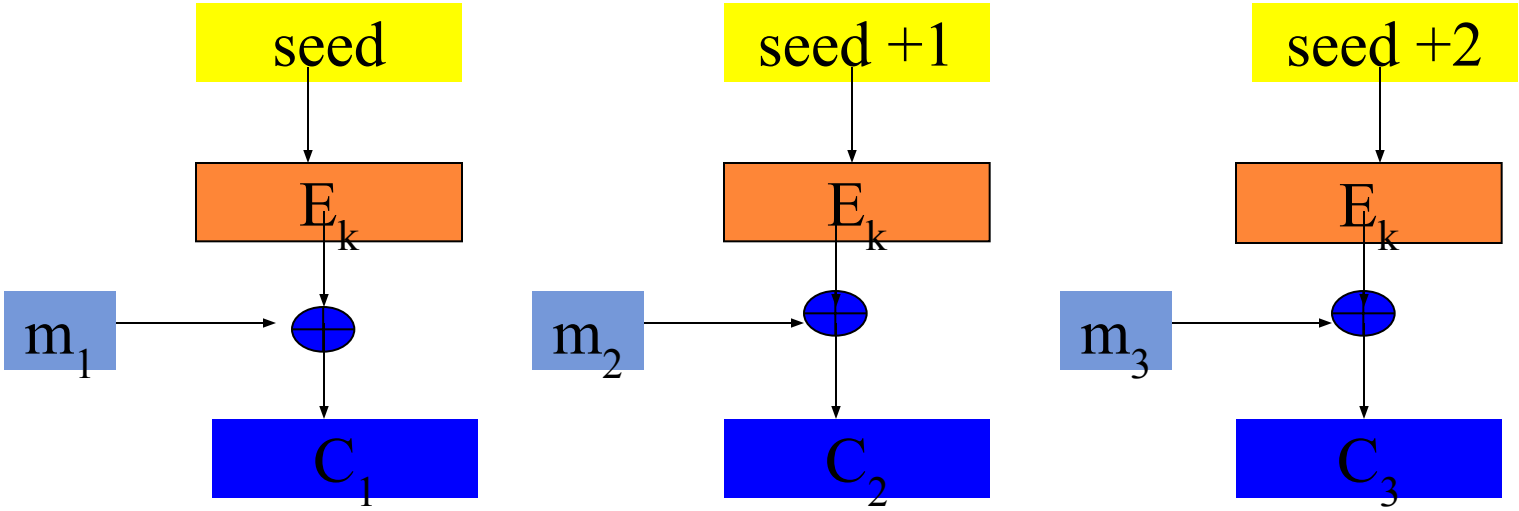
Properties of OFB

- Synchronous stream cipher
- Errors in ciphertext do not propagate
- Pre-processing is possible
- Conceals plaintext patterns
- No parallel implementation known
- Active attacks by manipulating plaintext are possible

CTR (Counter Mode)

- also known as Integer Counter Mode (ICM) and Segmented Integer Counter (SIC) mode
- turns a block cipher into a stream cipher: it generates the next keystream block by encrypting successive values of a "counter"
  - counter can be any function which produces a sequence which is guaranteed not to repeat for a long time, although an actual counter is the simplest and most popular.
- the usage of a simple deterministic input function raised controversial discussions
- has similar characteristics to OFB, but also allows a random access property during decryption
- well suited to operation on a multi-processor machine where blocks can be encrypted in parallel

CTR (Counter Mode)



Similar to OFB

There are problems in repeated use of same seed (like OFB) -

CTR vs OFB: using CTR you can decrypt the message starting from - block i for any i (i.e. You do not need to decrypt from the first block as in OFB)

Initialization vector (IV)

- Most modes (except ECB) require an initialization vector, or IV
  - sort of "dummy block" to kick off the process for the first real block, and also to provide some randomization for the process.
  - no need for the IV to be secret, in most cases, but it is important that it is never reused with the same key.
- For CBC and CFB, reusing an IV leaks some information about the first block of plaintext, and about any common prefix shared by the two messages.
- In CBC mode, the IV must, in addition, be unpredictable at encryption time
  - see TLS CBC IV attack
- For OFB and CTR, reusing an IV completely destroys security.

AES Proposed Modes

- CTR (Counter) mode (OFB modification): Parallel implementation, offline pre-processing, provable security, simple and efficient
- OCB (Offset Codebook) mode - parallel implementation, offline preprocessing, provable security (under specific assumptions), authenticity
- also see Block Cipher Modes, by NIST http://csrc.nist.gov/groups/ST/toolkit/BCM/index.html 2001 - 2010

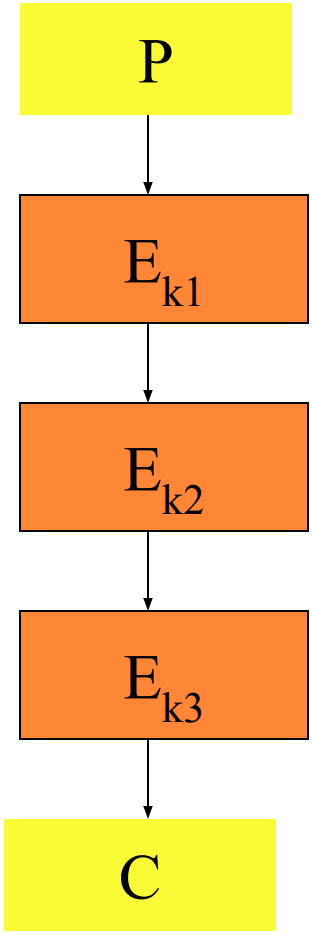
Strengthening a Given Cipher

- Design multiple key lengths – AES
- Key Whitening - the DES-X idea
  - Key whitening consists of steps that combine the data with portions of the key (most commonly using a simple XOR) before the first round and after the last round of encryption.
  - First use by Ron Rivest for strengthen DES, in 1984

$$\text{DES-X}(M) = K_2 \oplus \text{DES}_K(M \oplus K_1)$$

  - apparently key size becomes 184 (= 56 + 64×2), but its strength is 119 ($|K_1| = |K_2| = 64$)
- Iterated ciphers – Triple DES (3-DES), triple IDEA and so on

Triple Cipher - Diagram



Iterated Ciphers

- Plaintext undergoes encryption repeatedly by underlying cipher
- Ideally, each stage uses a different key
- In practice triple cipher is usually

  $C = E_{k1}(E_{k2}(E_{k1}(P)))$ [EEE mode] or

  $C = E_{k1}(D_{k2}(E_{k1}(P)))$ [EDE mode]

  EDE is more common in practice

Two or Three Keys

- Sometimes only two keys are used in 3-DES
- Identical key must be at beginning and end
- Legal advantage (export license) due to smaller overall key size
- Used as a KEK (key-encrypting key) in the BPI (Baseline Privacy Interface) protocol which secures the DOCSIS cable modem standard
  - DOCSIS: international standard that permits the addition of high-speed data transfer to an existing Cable TV system. It is employed by many cable television operators to provide Internet access over their existing hybrid fiber coaxial infrastructure

Adversary's goal

- Final goal: find the secret key
- Partial goals:
  - Reduce the # of possible keys
  - Detect patterns in the text
  - Decode part of the text
  - Modify the ciphertext obtaining a plausible text (even without breaking the cipher; even without knowing which modifications)

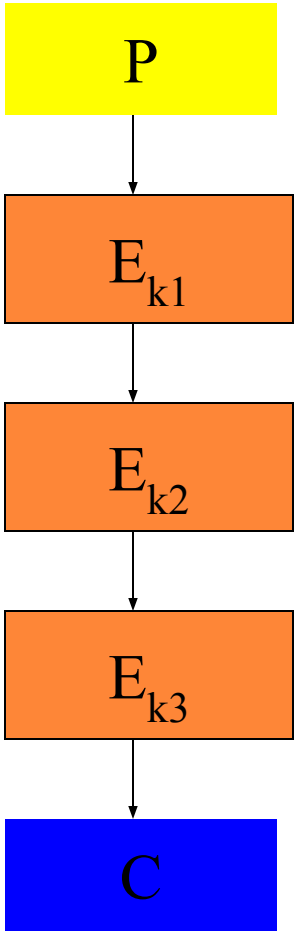
Double DES: meet-in-the-middle attack

Cipher twice with two different keys? NO!

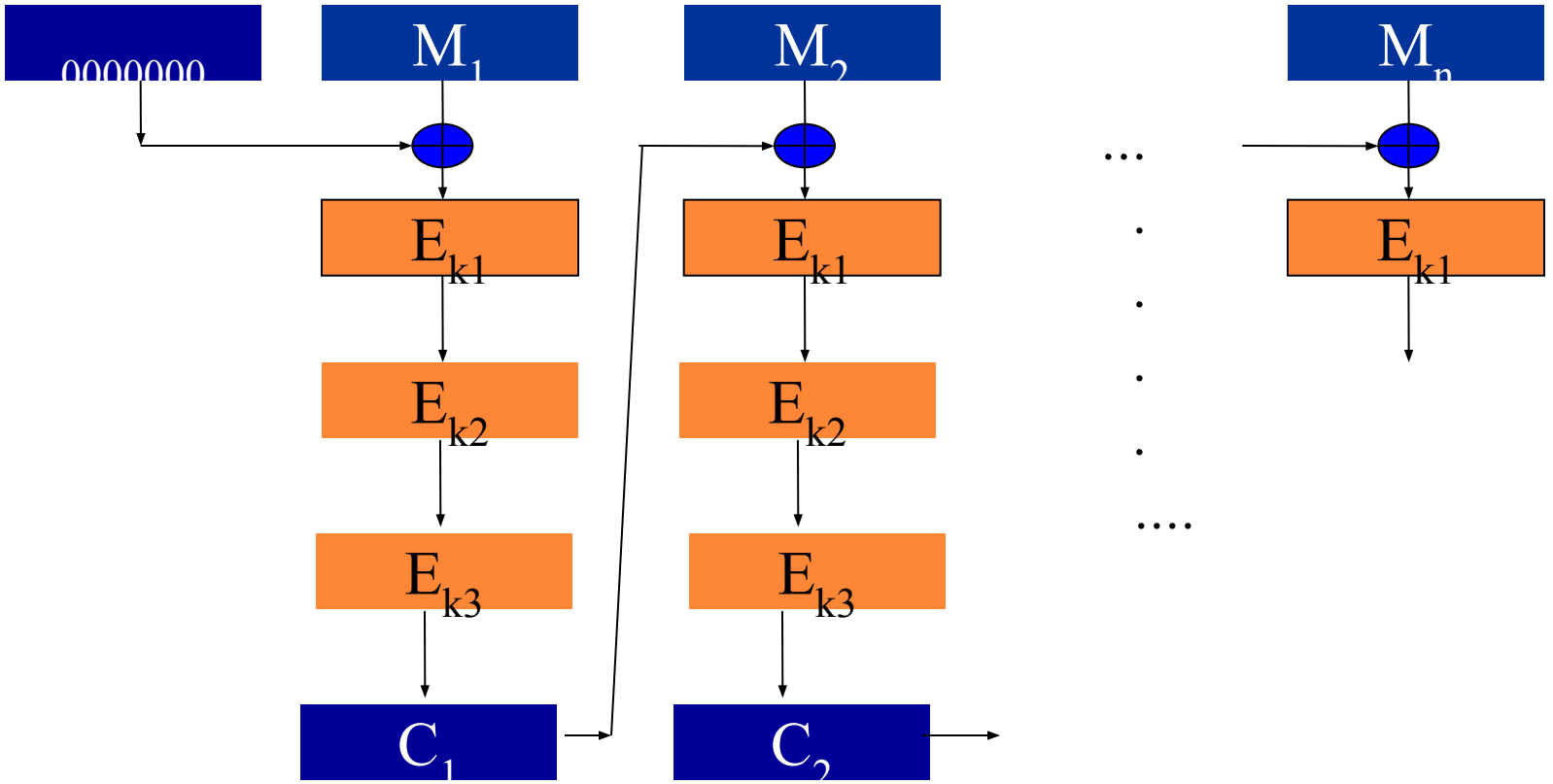
Meet-in-the-middle attack. Requirements

- Known plaintext/ciphertext pairs
- 2^n encryptions + 2^n decryptions (2 keys of n bit), instead of 2^{2n} brute-force
- 2^n memory space
- Idea: try all possible 2^n encryptions of the plaintext and all possible 2^n decryptions of the ciphertext. Encryptions stored into a lookup table.
- Check for a pair of keys that transform the plaintext in the ciphertext. Test pair on other pairs plaintext/ciphertext
- Note: the method can be applied to all block codes

Triple encoding



Triple encoding and CBC



In the picture: External CBC: code (using triple encoding) each block ; then concatenate

Other possibility: Internal CBC (the concatenation is internally made, before 1st encryption and after decryption). More secure, but less used