Classical Ciphers

密碼學與應用

海洋大學資訊工程系
丁培毅

Classical Cryptography

- Monoalphabetic ciphers: letters of the plaintext alphabet are mapped into unique ciphertext letters
- Polyalphabetic ciphers: letters of the plaintext alphabet are mapped into ciphertext letters depending on the context of the plaintext
- Stream ciphers: a key stream is generated and used to encrypt the plaintext

Classical Cryptosystem: Shift Cipher

- Shift Cipher
  - Letters of the alphabet are assigned unique numbers

a 0	b 1	c 2	d 3	e 4	f 5	g 6	h 7	i 8	j 9	k 10	l 11	m 12
n 13	o 14	p 15	q 16	r 17	s 18	t 19	u 20	v 21	w 22	x 23	y 24	z 25

- Algorithm:
  - Let $P = C = K = Z_{26}$ and $x \in P$, $Y \in C$, $k \in K$
  - *Encryption:* $E_k(x) = x + k \bmod 26$.
  - *Decryption:* $D_k(Y) = Y - k \bmod 26$.

Shift Cipher

- *Caesar Cipher* : shift cipher with $k = 3$
- **Example:** Let the key $k = 17$
  - Plaintext: X = a t t a c k = (0, 19, 19, 0, 2, 10)
  - Ciphertext : Y = (0+17 mod 26, 19+17 mod 26, …)
    = (17, 10, 10, 17, 19, 1) = R K K R T B
- Attacks
  - Ciphertext only:
    - Exhaustive Search: Try all possible keys. $|K|=26$.
      Nowadays, for moderate security $|K| \geq 2^{80}$,
      for recommended security $|K| \geq 2^{100}$.
    - Letter frequency analysis (Same plaintext maps to same ciphertext

Frequency Analysis

- In most languages, letters occur in texts with different frequencies
- single, double, triple letter frequencies

Single	Frequency	Double	Triple
E	.127	TH	THE
T	.091	HE	ING
A	.082	IN	AND
O	.075	ER	HER
I	.070	AN	ERE
N	.067	RE	ENT
S	.063	ED	THA
H	.061	ON	NTH

Letter Frequency Analysis

- Method 1: Find the most frequent cipher character, make a guess as E_k('e'), solves k. Use this k to decrypt ciphertext and see if it is a reasonable guess. Otherwise, find the second frequent cipher character, make a guess as E_k('e').
- Method 2: correlation
  A_0=[.082 .015 .028 .043 .127 .022 .020 .061 .070 .002
  .008 .040 .024 .067 .075 .019 .001 .060 .063 .091
  .028 .010 .023 .001 .020 .001]
  A_i is obtained by circularly shift right A_0 i elements
  e.g. A_2=[.020 .001 .082 .015 .028 .043 ...
- correlation = $A_i \cdot A_j$ is the usual dot product between A_i and A_j
- let A be the frequency of the ciphertext paragraph
- calculate correlation between A and A_i, choose the maximum

Shift Cipher

- **Known plaintext:** You can deduce the key if you know one letter of the plaintext along with its corresponding ciphertext. Ex. t(=19) encrypts to D(=3), then the key is $k \equiv 3 - 19 \equiv -16 \equiv 10 \pmod{26}$
- **Chosen plaintext**: choose the letter ‘a’ as the plaintext, the ciphertext is the key
- **Chosen ciphertext**: choose the letter ‘A’ as ciphertext, the plaintext is the negative of the key

Shift Cipher

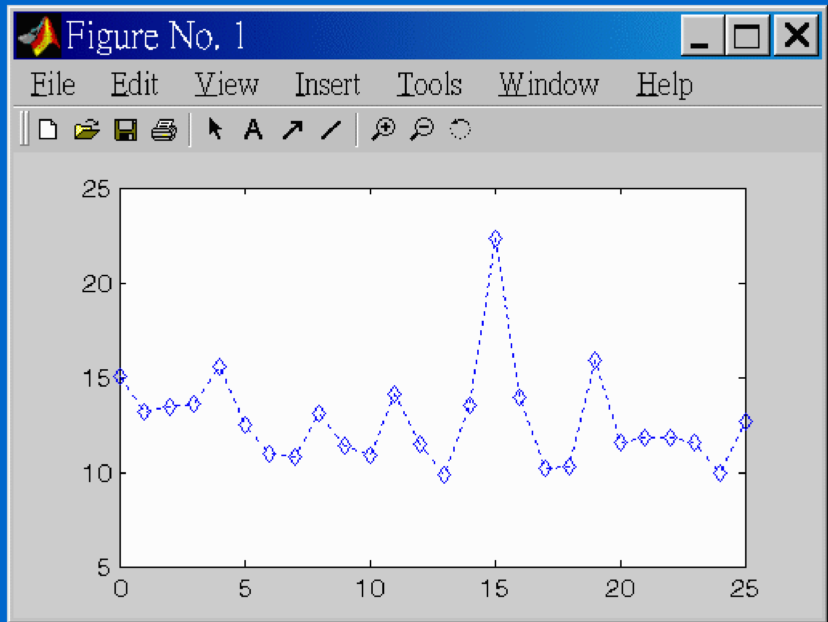
- One time pad can be considered as a shift cipher with modulus 2 and a changing key sequence, in which each key is used only for one plaintext character and never repeated.
- A shift cipher as defined is therefore perfectly secure if the key keeps changing and is used for one character only.

Matlab Example

- dir, cd, help
- path(path, 'c:\lcwMatlabCode')
- k = 20
  plain = 'hellothisisashiftcipherexample'
  plain_i = text2int(plain)
  cipher_i = mod(plain_i + k, 26)
  cipher = int2text(cipher_i)
  recovered_i = mod(cipher_i - k, 26)
  recovered = int2text(recovered_i)
- cipher = shift(plain, k)
  recovered = shift(cipher, -k)

Matlab letter frequency analysis

- sci=
['themethodusedforthepreparationandreadingofcodemessagesissimplei', …
'ntheextremeandatthesametimeimpossibleoftranslationunlessthekeyi', …
'sknowntheeasewithwhichthekeymaybechangedisanotherpointinfavorof', …
'theadoptionofthiscodebythosedesiringtotransmitimportantmessages', …
'withouttheslightestdangeroftheirmessagesbeingreadbypoliticalorb', …
'usinessrivalsetc'];
- cipher=shift(sci, 15);
- freq=frequency(cipher);
- correlation=corr(freq);
- plot(0:25,correlation,'bd:')



Affine Cipher

- **Algorithm:** Let $P = C = Z_{26}$ and $x \in P$, $Y \in C$
  - *Encryption:* $E_k(x) = Y = \alpha \cdot x + \beta \bmod 26$
  - The key $k = (\alpha, \beta)$ and $\alpha, \beta \in Z_{26}$
  - ex. $\alpha=13$, $\beta=4$

    input = (8, 13, 15, 20, 19) $\Rightarrow$ (4, 17, 17, 4, 17) = ERRER

    alter = (0, 11, 19, 4, 17) $\Rightarrow$ (4, 17, 17, 4, 17) = ERRER
  - There is no one-to-one mapping between plaintext and ciphertext. What's wrong?
  - *Decryption:* $D_k(Y) = x = \alpha^{-1} \cdot (Y - \beta)$

    $= \alpha' \cdot Y + \beta' \bmod 26$

Affine Cipher

- **Key Space:**
  - β can be any number in Z_{26} . 26 possibilities
  - Since α^{-1} is required to exist, we can only select integers in Z_{26} s.t. $\gcd(\alpha, 26) = 1$. Candidates are {1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25}
  - Therefore, the key space has $12 \cdot 26 = 312$ candidates.
- **Attack types:**
  - *Ciphertext only*: exhaustive search or frequency analysis

Letter Frequency Analysis

- Consider the ciphertext

  FMNVEDKAPHFERBNDKRX
  RSREFMORUDSDKDVSHVU
  FEDKAPRKDLYEVLRHHRH

- Letter frequency of the ciphertext:

Letter	# of Occurrences
R	8
D	6
E	5
H	5
K	5
V	4
F	4

Letter Frequency Analysis

- Make a guess: choose two potential candidate letters
  e.g. 1st guess R $\rightarrow$ e and D $\rightarrow$ t
- Try to show the guess make sense by solving
  (α, β) from $E_k(x) = Y = \alpha \cdot x + \beta \bmod 26$
  e.g. $4\,\alpha + \beta = 17 \bmod 26$ and $19\,\alpha + \beta = 5 \bmod 26$
  $\Rightarrow \alpha = 6, \beta = 19$, which is illegal since gcd(6,26)>1
- 2nd guess: R $\rightarrow$ e and E $\rightarrow$ t ….. $\Rightarrow \alpha = 13$, still illegal
- 3rd guess: R $\rightarrow$ e and H $\rightarrow$ t ….. $\Rightarrow \alpha = 3, \beta = 5$
  i.e. $E_k(x) = 3 \cdot x + 5 \bmod 26$
  $D_k(x) = 9 \cdot x - 19 \bmod 26$

Letter Frequency Analysis

- Better Solution: correlation
  - Enumerate 312 possible keys, ex. (3,2)
  - Let A_0= [.082, .015, .028, .043, .127, .022, .020, .061, .070, .002, .008, .040, .024, .067, .075, .019, .001, .060, .063, .091, 028, .010, .023, .001, .020, .001]
  - Let the i-th key be (3,2), which maps plaintexts [0, 1, 2, 3, 4 …, 25] to ciphertexts [2, 5, 8, 11, 14, 17, 20, 23, …]
  - Calculate a vector A_i with the k-th element being $A_0(E_{3,2}(k))$, ex. A_i = [$A_0(2)$, $A_0(5)$, $A_0(8)$, $A_0(11)$, $A_0(14)$, $A_0(17)$, $A_0(20)$, $A_0(23)$, $A_0(0)$, …]
  - Perform correlation $A \cdot A_i$ and find the maximum

Affine Cipher

- **Attack types:**
  - *Known plaintext*: two letters in the plaintext and corresponding ciphertext letters would suffice to find the key.

    Ex. plaintext ‘if’=(8, 5) and ciphertext ‘PQ’=(15, 16)

    $8 \cdot \alpha + \beta \equiv 15 \bmod 26$

    $5 \cdot \alpha + \beta \equiv 16 \bmod 26 \Rightarrow \qquad \alpha = 17 \text{ and } \beta = 9$

    What happens if we have only one letter of known plaintext?

    still have great reduction in candidates

Affine Cipher

- **Attack types:**
  - *Chosen plaintext*: Choose a and b as the plaintext. The first character of the ciphertext will be equal to $0 \cdot \alpha + \beta = \beta$ and the second will be $\alpha + \beta$.
  - *Chosen ciphertext* : Choose A and B as the ciphertext. The first character of the plaintext will be equal to $0 \cdot \alpha' + \beta' = \beta'$ and the second will be $\alpha' + \beta'$, $\alpha = (\alpha')^{-1}$ and $\beta = -\alpha \cdot \beta'$

Matlab Example

- a = 3, b = 5,   ap = 9, bp = -19;
- plain = 'matlabaffinecipherencryptionexample';
- cipher = affinecrypt(plain, a, b)
- recovered = affinecrypt(cipher, ap, bp)

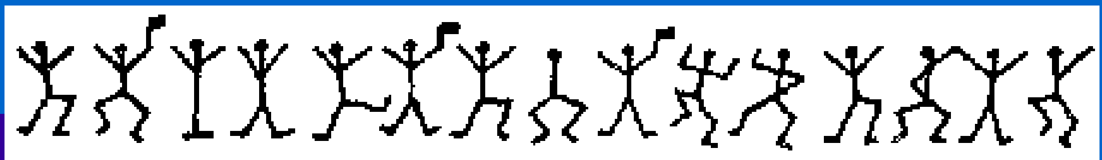
Substitution Ciphers

– Each letter in the alphabet is replaced (substituted) by another letter. More precisely, a permutation of the alphabet is chosen and applied to the plaintext.

– Shift ciphers and affine ciphers are special cases of substitution ciphers.

– Since ciphertext preserves the statistic of the language used in the plaintext, the "frequency analysis" is an effective way of breaking substitution ciphers with only ciphertext.

– The Adventure of the Dancing Men by Arthur Conan Doyle http://www.sherlockian.net/canon/stories/danc.html



Vigenère Cipher

- **Algorithm:** Let $P = C = Z_{26}$ and $x \in P$, $Y \in C$
  - *Encryption*: $Y = E_k(x) \equiv x + k_i \pmod{26}$
  - The key $k = (k_1, k_2, k_3, \ldots, k_n)$, $k_i \in Z_{26}$, neither the key or the length n is known to adversary
  - *Decryption*: $x = D_k(Y) \equiv Y - k_i \pmod{26}$
- ex. key='danger'

plaintext:	h	e	l	l	o	t	h	i	s	i	s	a
keys:	d	a	n	g	e	r	d	a	n	g	e	r
ciphertext:	K	E	Y	R	S	K	K	I	F	O	W	R

- Attacks:
  - Ciphertext Only:
    - Finding the key length
    - Finding the key

Vigenère Cipher

- Finding the key length:
  - Friedman's Test uses Index of Coincidence: Let $I_c(x)$ be the probability that two random elements of the n-letter string *x* are identical
  - Let $f_0, f_1, \ldots, f_{25}$ be the number of occurrence of A, B, …Z, respectively in the n-letter string *x*

$$I_c(x) = \frac{\sum_{i=0}^{25} \binom{f_i}{2}}{\binom{n}{2}} = \frac{\sum_{i=0}^{25} f_i(f_i - 1)}{n(n-1)}$$

Vigenère Cipher

- The letter frequency of English is
  - A_0= [.082 .015 .028 .043 .127 .022 .020 .061 .070 .002 .008 .040 .024 .067 .075 .019 .001 .060 .063 .091 .028 .010 .023 .001 .020 .001]
- The expected value of $I_c(x)$ is
  - for English Text:

    $$I_c(x) = A_0 \cdot A_0 = (.082)^2 + (.015)^2 + \ldots = 0.666$$
  - for Random String:

    $$I_c(x) = 26 \cdot (1/26)^2 = 0.038$$
  - for shifted English Text(the first letter shifted by k_i and the second letter shifted by k_j):

$\lvert i-j \rvert$	1	2	3	4	5	6	7	8	9	10	11	12	13
$I_c(x) = A_i \cdot A_j$	.039	.032	.034	.044	.033	.036	.039	.034	.034	.038	.045	.039	.042

Vigenère Cipher

- find the coincidences in the ciphertext

'vvhqwvvrhmusgjgthkihtssejchlsfcbgvwcrlryqtfsvgahwkcuhwauglq'
'hnslrljshbltspisprdxljsveeghlqwkasskuwepwqtwvspgoelkcqyfnsv'

'wljsniqkgnrgybwlwgoviokhkazkqkxzgyhcecmeiujoqkwfwvefqhkijrc'
'lrlkbienqfrjljsdhgrhlsfqtwlauqrhwdmwlgusgikkflryvcwvspgpmlk'
'assjvoqxeggveyggzmljcxxljsvpaivwikvrdrygfrjljslveggveyggeia'
'puuisfpbtgnwwmuczrvtwglrwugumnczvile'

- coincidences: (by shift and count)

shift →				
14	14	16	14	24
12	13	13	7	14
13	19	13	15	26
11	13	14	11	20
17	14	15	16	21

⇒ Key length is 5

Vigenère Cipher

- Finding the Key:
  - To find the first element of the key, count the frequencies of the letters in the 1st, 6th, 11th … positions of the ciphertext
    V = (0,0,7,1,1,2,9,0,1,8,8,0,0,3,0,4,5,2,0,3,6,5,1,0,1,0)
  - Divide by number of letters counted, 67
    W = (0, 0, .1045,.0149,.0149,.0299,…,.0149,0)
  - Compute $W \cdot A_i$ for $0 \leq i \leq 25$

0.0250	0.0391	0.0713	0.0388	0.0275	0.0380	0.0512	0.0301	0.0325
0.0430	0.0338	0.0299	0.0343	0.0446	0.0356	0.0402	0.0434	0.0502
0.0392	0.0296	0.0326	0.0392	0.0366	0.0316	0.0488	0.0349	

$\Rightarrow$ first key is ‘c’

Vigenère Cipher

– Known plaintext:
  - if enough (plaintext, ciphertext) pairs are known

$$k_i = Y - x$$

– Chosen plaintext:
  - choose plaintext aaaaa…

$$k_i = Y$$

– Chosen ciphertext:
  - choose ciphertext AAAAA…

$$k_i = -x$$

Matlab Example

- Encrypt/decrypt
  - key = 'vigenere';
  - key_i = text2int(key);
  - plain = 'matlabaffinecipherencryptionexample';
  - cipher=vigenere(plain, key_i)
  - recovered=vigenere(cipher, -key_i)

Matlab Example

- Ciphertext only attack:
  - ciphertexts
  - for i=1:25,
  - a(i) = coinc(vvhq, i);
  - end
  - first = choose(vvhq, 5, 1)
  - V = frequency(first)
  - W = V / length(first)
  - corr(W)

finding key length

finding first key

Block Ciphers

- In the substitution ciphers, changing one letter in the plaintext changes exactly one letter in the ciphertext.
- This greatly facilitates finding the key using frequency analysis.
- Block ciphers prevent this by encrypting a block of letters simultaneously.
- Many of the modern (symmetric) cryptosystems are block ciphers. DES operates on 64 bits of blocks while AES uses 128 bits of blocks (optionally 192 and 256 bits blocks).

Hill Cipher

- The key is an $n \times n$ matrix whose entries are elements in Z_{26}
- Ex. Let n=3, the key matrix be

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 11 & 9 & 8 \end{pmatrix}$$

and the plaintext be abc = (0, 1, 2) then the encryption operation is a vector-matrix multiplication

$$(0,1,2) \times \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 11 & 9 & 8 \end{pmatrix} \equiv (0,23,22) \bmod 26 \Rightarrow \text{AXW (ciphertext)}$$

In order to decrypt, the inverse of the key matrix M is:

$$N = \begin{pmatrix} 22 & 5 & 1 \\ 6 & 17 & 24 \\ 15 & 13 & 1 \end{pmatrix}$$

Hill Cipher (cont'd)

- If we change one letter in the plaintext, all the letters of the ciphertext will be affected.
- Let the plaintext be bbc instead of abc then the ciphertext

$$(1,1,2)\times\begin{pmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 11 & 9 & 8\end{pmatrix}\equiv(1,25,25)\bmod 26\Rightarrow \text{BZZ (ciphertext)}$$

General Design Principle

- Claude Shannon, in *Communication theory of secrecy systems* Bell Systems Technical Journal 28, (1949), 656-715, introduced properties that a good cryptosystems should have:
  - **Diffusion:** one character changes in the plaintext should effect as many ciphertext characters as possible, and vice versa.
  - **Confusion:** The key should not relate to the ciphertext in a simple way.

Stream Cipher

- plaintext alphabets P
  ciphertext alphabets C
  key stream alphabet L
  key stream generator $F = \{f_1, f_2, \ldots.\}$
  $$f_i: K \times P^{i-1} \rightarrow L$$
  $$\ell_i = f_i(k, x_1, \ldots, x_{i-1})$$ k is the seed
- Encryption:
  – for plaintext $x_1, x_2, \ldots$ ciphertext $c_1 = E_{\ell_1}(x_1), c_2 = E_{\ell_2}(x_2), \ldots$
- Decryption:
  – for ciphertext $c_1, c_2, \ldots$ recovered plaintext $x_1 = D_{\ell_1}(c_1), x_2 = D_{\ell_2}(c_2), \ldots$
- For each $\ell \in L$, E_ℓ, D_ℓ satisfy $\forall x \in P$, $D_\ell(E_\ell(x)) = x$

Autokey cipher

- Key stream generator: $\ell_i = x_{i-1}$, $\ell_1 = k$, k is an initial seed
  Encryption: $E_\ell(x) = x + \ell \mod 26$
  Decryption: $D_\ell(y) = y - \ell \mod 26$
- Ex: k = 8, plaintext: 'rendezvouz'

plaintext:	r	e	n	d	e	z	v	o	u	s	
	17	4	13	3	4	25	21	14	20	18	
keys:	8	17	4	13	3	4	25	21	14	20	18
ciphertext:	25	21	17	16	7	3	20	22	8	12	
	Z	V	R	Q	H	D	U	J	I	M	
keys:	8	17	4	13	3	4	25	21	14	20	18
plaintext:	17	4	13	3	4	25	21	14	20	18	

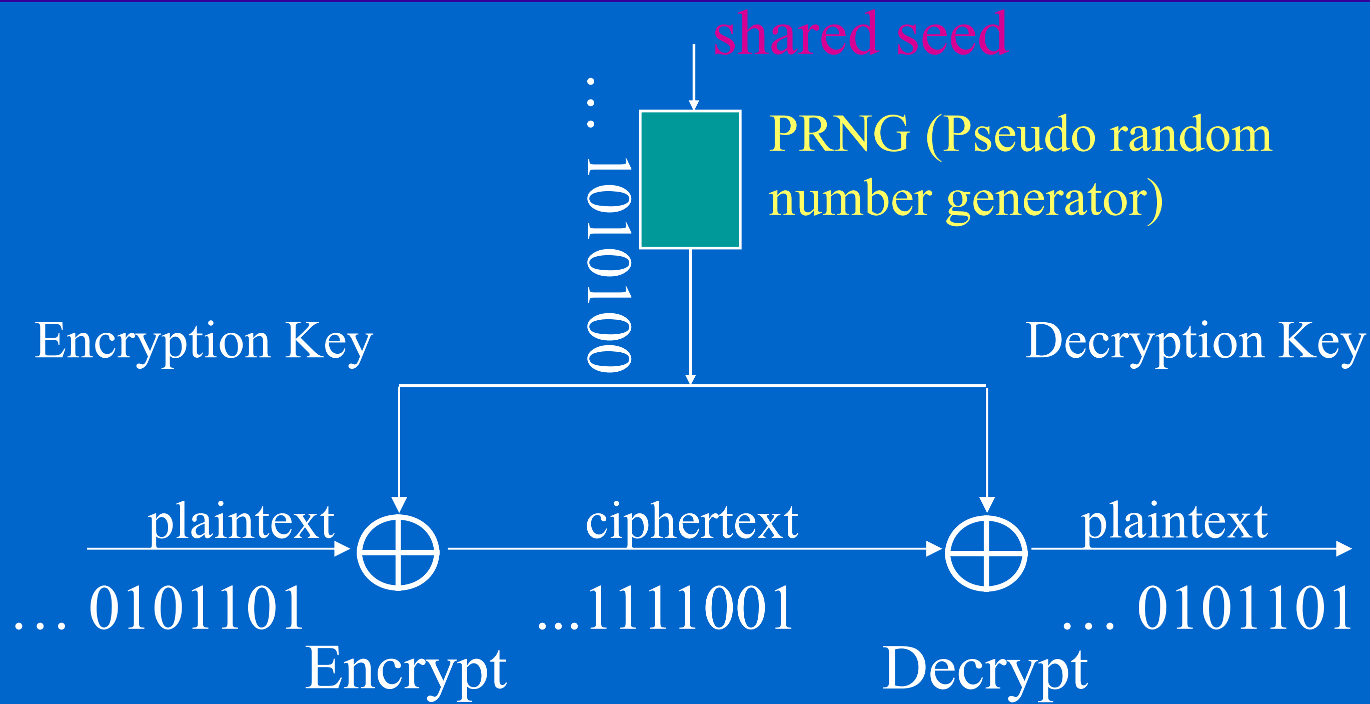
Stream Cipher

- Block ciphers are special cases of stream ciphers where the key stream is constant.
- A stream cipher is synchronous if the key stream is independent of the plaintext.
  - Both sender and receiver must be synchronized.
  - Resynchronization can be needed.
  - No error propagation (if the deciphered plaintext is incorrect).
  - Active attacks can easily be detected.
- A stream cipher is periodic with period d if $\ell_{i+d} = \ell_i$, for all $i \geq 1$.

Stream Cipher

- The Vigenère cipher with keyword length m is a periodic stream cipher with period m.
- Stream ciphers are often described in binary 0, 1 alphabets. ex. one-time pad
- Perfectly Secure: One-time pad
- Examples of practical stream ciphers
  - Autokey Cipher
  - One-time pad with Pseudo-random Bit Generation
  - Linear Feedback Shift Register (LFSR)
  - DES in Counter Mode or CFB Mode
  - Feistel Cipher

OTP with PRNG



- Security? not provably secure??
  - Random number sequence is used as key!! What if it is not so random??
- What is a random number?

Randomness

random

pseudo random, PRNG

- Randomness? ex. flipping a fair coin, thermal noise
  - Uniformly distributed string sequences
  - a string s is Komogorov-random if its length equals the length of the shortest program producing s

    ex. 01010101010101010101
  - Statistical approach: pass some statistical tests: ex. 0/1 bits appear equally, number of 0/1 bits are equal, any two bits are uncorrelated, Maurer’s Universal Test, Chi-Square Test, Kolmogorov-Smirnov Test ...
  - Computational approach:
    - indistinguishable from any uniformly distributed sequences
    - unpredictable by any poly-time algorithm (the probability to predict the next bit is no better than 1/2)

Pseudorandom Number Generator

- Existence? one way function assumption
- Poor implementation for cryptographic usage:
  - linear congruential generator rand() in the standard C/UNIX library

    $x_n = a\, x_{n-1} + b \mod m$, x_0 is the initial seed

    - a, b, m can be discovered from the x_n sequence
    - therefore x_n is completely predictable (key is know to everybody!!)
    - any polynomial congruential generator is cryptographically insecure
    - can be used only for the purpose of statistical experiments

Pseudorandom Number Generator

- Fairly good implementation for cryptographic purpose:
  - Method 1: based on one-way function candidates (DES, SHA..)
    - one-way function f: $y = f(x)$, given y, it's hard to compute x

      $x_j = f(s+j)$, $j=1,2,3,\ldots$ s is the seed

      let the random bit sequence b_j be the LSB of x_j,
    - PRNG in the OpenSSL toolkit is based on SHA
  - Method 2: Blum-Blum-Shub (BBS, 1984)
    - $p \equiv 3 \pmod 4$, $q \equiv 3 \pmod 4$, $n = p \cdot q$, seed k
    - $x_0 \equiv k^2 \pmod n$, $x_j \equiv x_{j-1}^2 \pmod n$,

      let the random bit sequence b_j be the LSB of x_j

BBS example

- Let p = 24672462467892469787 q = 396736894567834589803

  n = 9788476140853110794168855217413715781961

  take k = 873245647888478349013

  $x_0 \equiv k^2 \pmod{n} \equiv 8845298710478780097089917746010122863172$

  $x_1 \equiv x_0^{\,2} \pmod{n} \equiv 7118894281131329522745962455498123822408$

  $x_2 \equiv x_1^{\,2} \pmod{n} \equiv 3145174608888893164151380152060704518227$

  ….

  $b_1 = 0$

  $b_2 = 1$,….

- slow for practical application,

  take k ($\le \log_2\log_2 n$) LSB bits of x_j

Maple example in Matlab

```
maple('p := 246724624678924697 87')
```

maple('p := 24672462467892469787')

maple('q := 39673689456783458980 3')

maple('n := p*q')

maple('x := 87324564788847834901 3')

maple('x0 := x&^2 mod n')

maple('x1 := x0&^2 mod n')

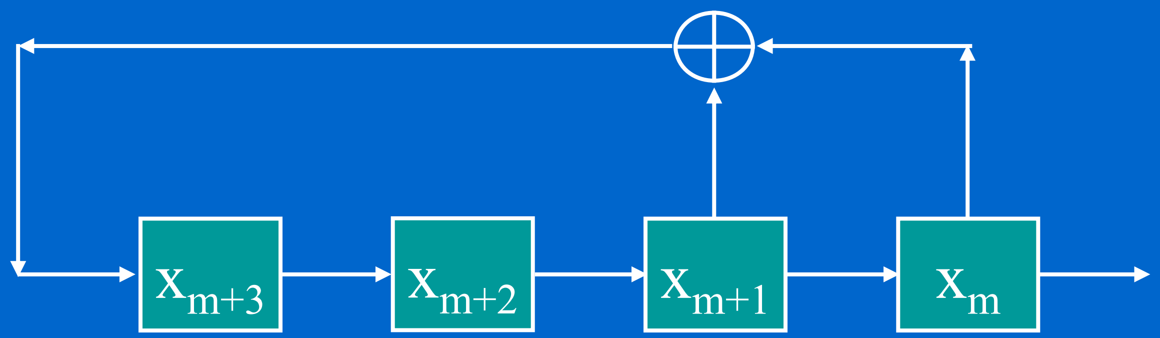
maple('x2 := x1&^2 mod n')

…

mhelp intro
mhelp mod
mhelp ^
mhelp &
mhelp :=

Linear Feedback Shift Register (LFSR)

- Hardware-oriented implementation: sacrifice security to obtain encryption speed



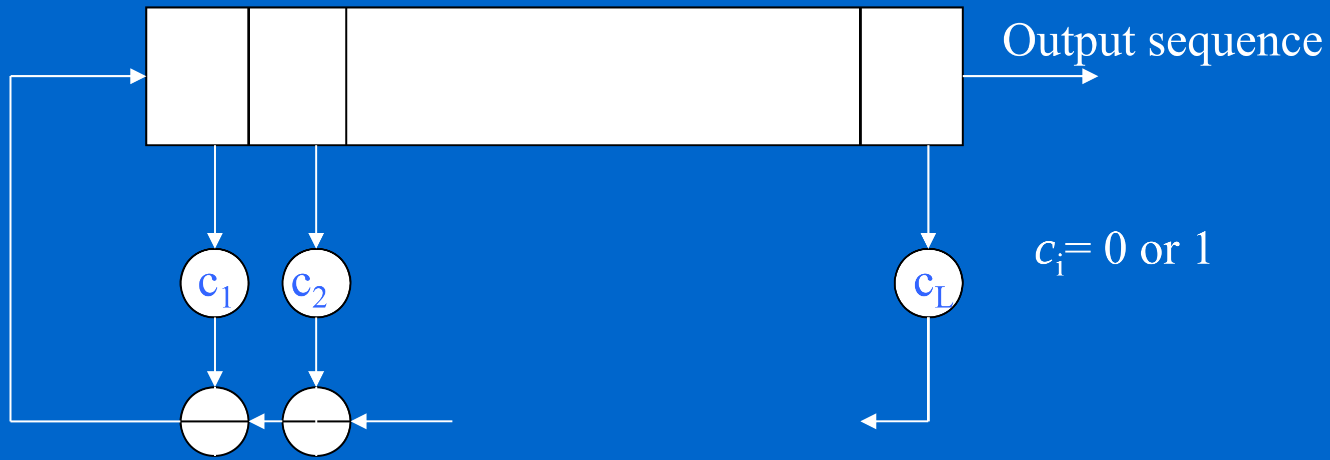
$$x_{m+3} = x_{m+1} + x_m$$

- in general:

$$x_{n+m} = c_0 x_n + c_1 x_{n+1} + \ldots + c_{m-1} x_{n+m-1} \pmod 2$$

with initial values $x_1, x_2, \ldots, x_m$

Linear Feedback Shift Register (LFSR)



$$C(x) = c_1 x + c_2 x^2 + \cdots + c_L x^L$$

- If $C(x)$ is primitive, LFSR is called *maximum-length LFSR*, and the output sequence is called *m-sequence* and its period is $T = 2^L$-1.
- *m-sequences* have good statistical properties.
- However, they are predictable.

Linear Feedback Shift Register (LFSR)

- For a length m linear recurrence relation, the period of the sequence is at most 2^m-1.
  - Any m consecutive terms of the sequence determine the complete sequence. As soon as there are more than 2^m-1 terms, some string of length m must occur twice.

  - ex. $x_{n+31} \equiv x_n + x_{n+3}$, with any **nonzero initial vector**, will produce a sequence that has period 2^{31}-1

Linear Feedback Shift Register (LFSR)

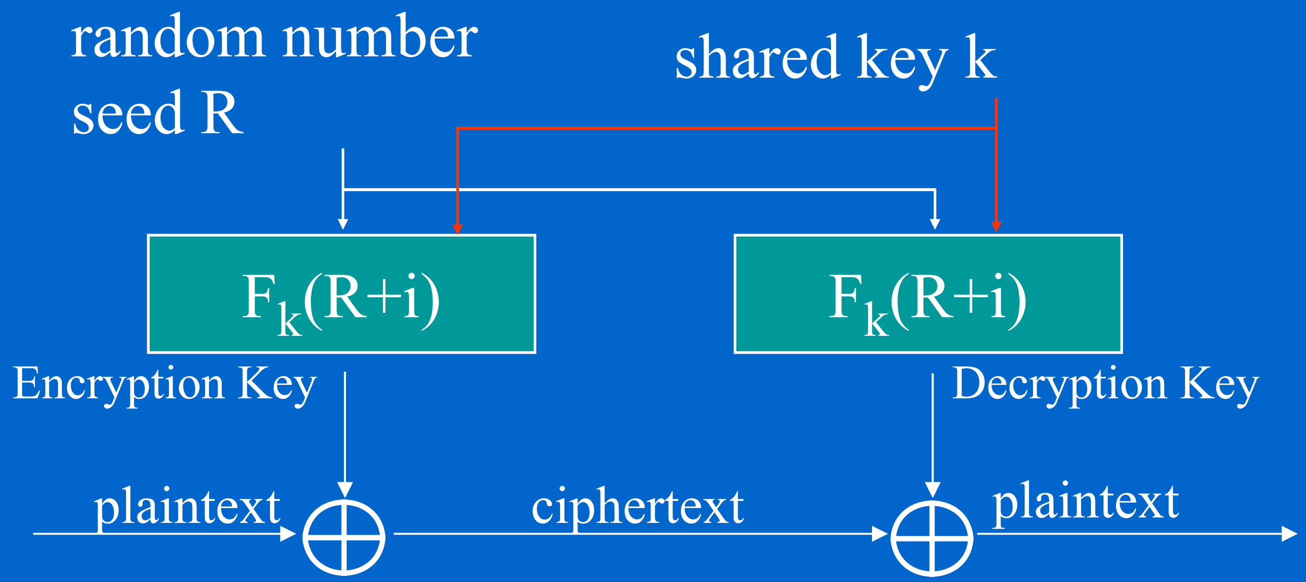
- Given a segment 011010111100 of a LFSR sequence, it is possible to deduce the length of the recurrence and the coefficients. (If you find a segment of 2m-bit plaintext and the corresponding ciphertext, you discover the corresponding segment of the key sequence.)
- The general solution: solve coefficients c_i from

$$\begin{pmatrix} x_1 & x_2 & \cdots & x_m \\ x_2 & x_3 & \cdots & x_{m+1} \\ \vdots & \vdots & \vdots & \vdots \\ x_m & x_{m+1} & \cdots & x_{2m-1} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{m-1} \end{pmatrix} \equiv \begin{pmatrix} x_{m+1} \\ x_{m+2} \\ \vdots \\ x_{2m} \end{pmatrix}$$

Linear Feedback Shift Register (LFSR)

- Computation in GF(2^n) can be quickly implemented in hardware with linear-feedback shift registers.
- Computation in GF(2^n) (eg. exponentiation and discrete log) is often quicker than computation over GF(p).
  - E. R. Berlekamp, Algebraic Coding Theory, Aegean Park press 1984
  - T. Beth et. al, "Architectures for Exponentiation in GF(2^n)," Crypto 86

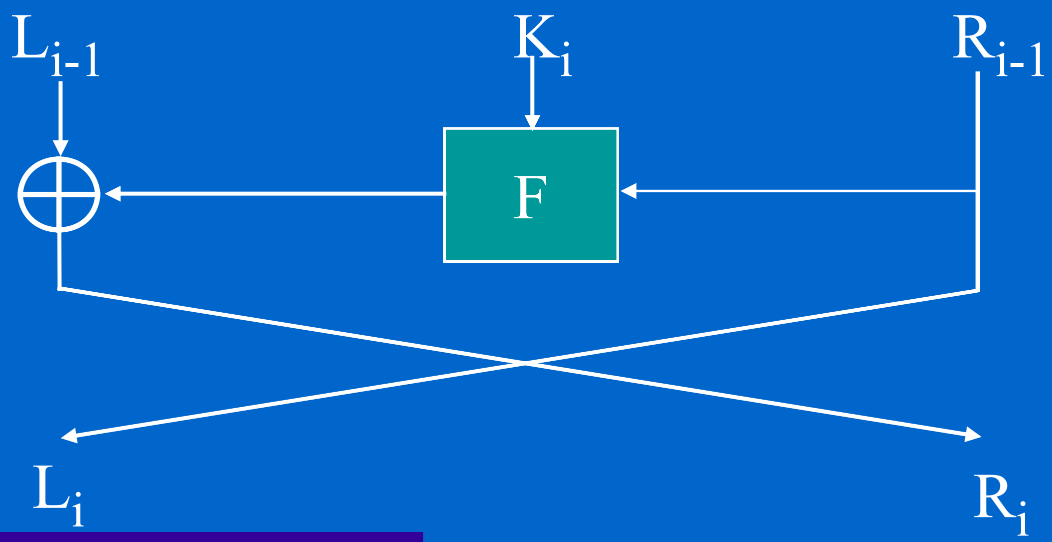
DES in Counter Mode



- pseudo one-time pad
- has better security properties than CBC, CFB, OFB encryption modes

Feistel Cipher

- Horst Feistel, 1973 IBM LUCIFER
- a common block encryption structure used in many symmetric encryption schemes that maximize the effects of Shannon's "Confusion" and "Diffusion"



Enigma

- German Enigma cipher machine in World War II. The Enigma had been broken by the Allies in World War II. The capture of the German U-505 submarine in David Kahn's book.
- U-571, 2000 movie; Enigma, 2002 movie
- see John J. G. Savard, A Cryptographic Compendium
  – http://home.ecn.ab.ca/~jsavard/crypto/entry.htm
- Codes throughout history
  – http://codebreaker.dids.com/fhistory.htm