CS5285
Tutorial 2

- A secret key (in Simple Substitution) is a *random permutation* of the alphabetic characters.
- E.g.

a	b	c	d	e	f	g	h	i	j	k	l	m
X	N	Y	A	H	P	O	G	Z	Q	W	B	T

n	o	p	q	r	s	t	u	v	w	x	y	z
S	F	L	R	C	V	M	U	E	K	J	D	I

- Each permutation is a potential candidate of the secret key
- A Ceasar cipher is a special substitution cipher with a fixed shift of n

Q1

- Ceasar Cipher
  - What is the plaintext?
  - Little bit of frequency analysis and knowing all letter shift by same amount
  - There are 3xL and 3xH (So which one is E or I)
- L is I, H is E

A constant shift of 3 (i.e. A > D, etc.), thus the plaintext becomes:

I CAME I SAW I CONQUERED

Q2

a) The ciphertext of the plaintext phrase : ‘ecommerce’
Using the given alphabet the ciphertext becomes EFYLLESFE

b) What is the plaintext of the ciphertext: JKLLEPSTF
The given ciphertext can be decrypted to: ‘symmetric’

One-time Pad Encryption

Encryption: Plaintext ⊕ Key = Ciphertext

The key is same length as plaintext! Could combine in different ways…
XOR is the most common!

We use ASCII to represent the text (shown as hexadecimal numbers)
eXclusive OR (⊕) binary operation (1⊕1=0; 1⊕0=1; 0⊕0=0)

	h	e	l	l	o	a	l	i	c	e
Plaintext:	68	65	6C	6C	6F	61	6C	69	63	65
Key:	FF	0A	B2	5D	C7	C3	EE	22	3F	68
Ciphertext:	97	6F	DE	31	A8	A2	82	4B	5C	0D

Q3

a) Alice wants to send the message HELLO to Bob using a one-time pad with the key ABCDE. What is the ciphertext?

Encryption: Plaintext XOR Key = Ciphertext.

Plaintext M	H	E	L	L	O
	0111	0100	1011	1011	1110
One-time pad K	A	B	C	D	E
	0000	0001	0010	0011	0100
$M \oplus K$	0111	0101	1001	1000	1010
Ciphertext C	**H**	**F**	**J**	**I**	**K**

Q3

b) What is the plaintext you get if you decrypt the ciphertext from 3(a) with the key FOBEL?

Decryption: Ciphertext XOR Key = Plaintext.

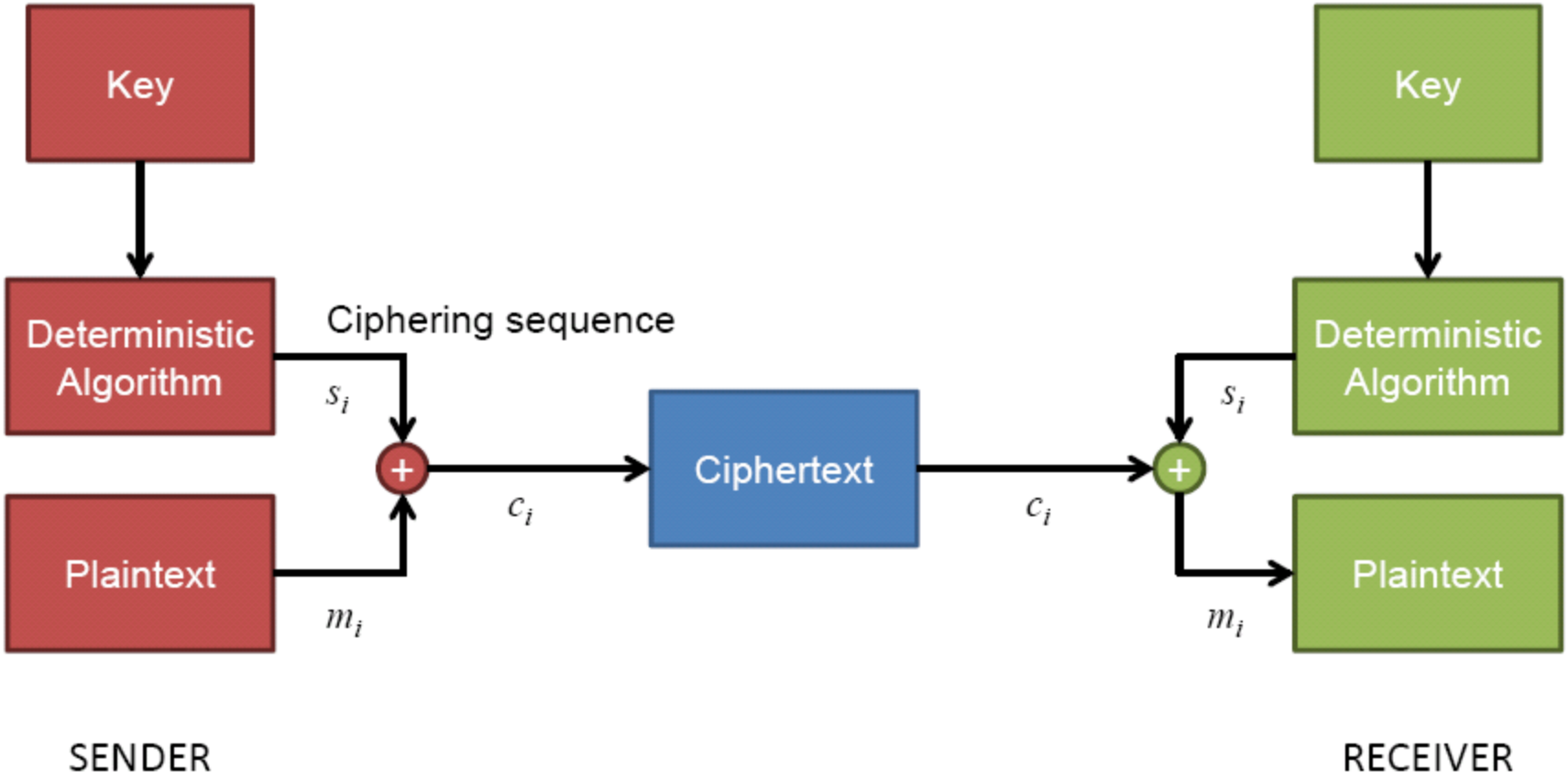
Ciphertext C	H	F	J	I	K
	0111	0101	1001	1000	1010
One-time pad K	F	O	B	E	L
	0101	1110	0001	0100	1011
$C \oplus K$	0010	1011	1000	1100	0001
Plaintext M	**C**	**L**	**I**	**M**	**B**

Q3

c) Give one disadvantage of one-time pad encryption.

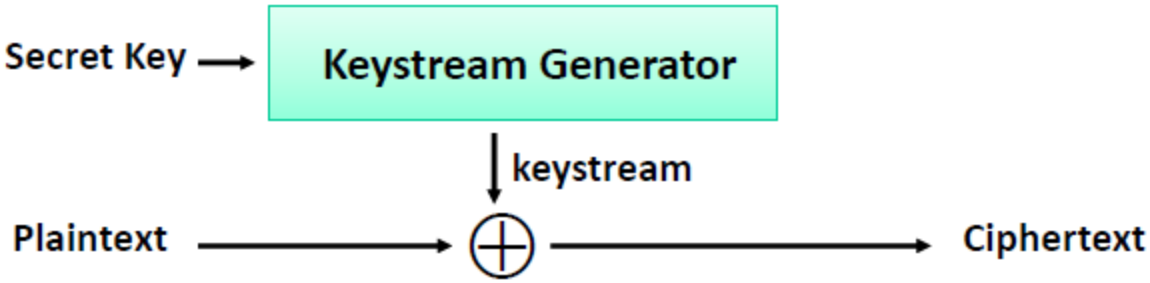
Key management - the keys needs to be the same length as the data, you must somehow give this key to the receiver, and each message need a new, random key.

Stream Ciphers



- How does this improve on stream cipher?
- **Key set up once (key then used to generate ever changing keystream**
- **Key not the same length as message**

Stream Ciphers



- Secret key length: 128 bits, 256 bits, etc.
- Maximum plaintext length: usually can be arbitrarily long.
- **Security:** Given a "long" segment of keystream (e.g. 2^{40} bits), the secret key cannot be derived AND the subsequent segment of the keystream cannot be deducted.

Q4

a)How can I make money from this scheme?

$C = (account||amount||pwd) \oplus KS$ (KeyStream)

For Account field, $KS_{acc} = C_{acc} \oplus account$ *(account is is known!)*

Now XOR on $your\_account$ number with KS_{acc} to get C'.

New message $E(your\_account,\ amount, pwd)$.

b) What extra service do we need?

Integrity/Data origin authentication

Q4

Worked example of stream cipher issue.

Attacker knows account number 1234, his account number is 5674. I send $ 6500 to account number 1234, my PIN is 9141. The message is: 650012349141 (each character is 4 bits)

The generated keystream is 012345678901

P 650012349141 -> 0110 0101 0000 0000 0001 0010 0011 0100 1001 0001 0100 0001

XOR XOR

KS 012345678901 -> 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 0000 0001

C 642357531840 -> 0110 0100 0010 0011 0101 0111 0101 0011 0001 1000 0100 0000

Attacker knows the account number 1234 so it takes the ciphertext 5753 and XOR it with 1234

C_acc = 0101 0111 0101 0011

XOR

P_acc = 0001 0010 0011 0100

KS_acc = 0100 0101 0110 0111

KS_acc = 0100 0101 0110 0111

XOR (recovered KS to attack account)

P'_acc = 0101 0110 0111 0100 (5674)

C'_acc = 0001 0011 0001 0011

C' 642313131840 -> 0110 0100 0010 0011 0001 0011 0001 0011 0001 1000 0100 0000

XOR XOR

KS 012345678901 -> 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 0000 0001

P' 650056749141 -> 0110 0101 0000 0000 0101 0110 0111 0100 1001 0001 0100 0001

The end!

?

Any questions…