网络安全技术

刘振

上海交通大学 计算机科学与工程系

电信群楼3-509

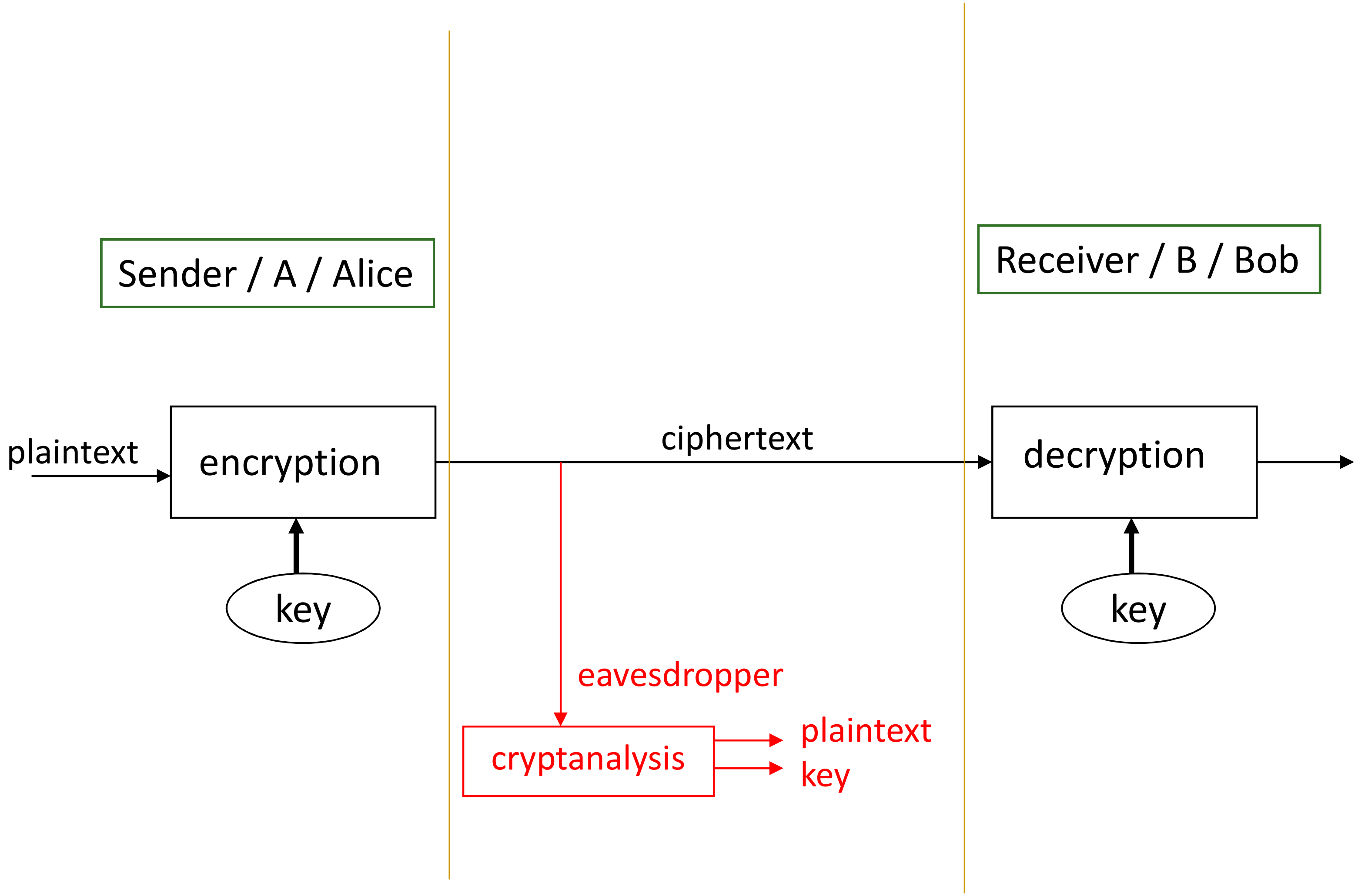
liuzhen@sjtu.edu.cn

Symmetric Key Encryption

Crypto – a brief introduction

- **Cryptology** — The art and science of making and breaking "secret codes"
- **Cryptography** — making "secret codes"
  - ychrpyaprtgo
  - $C = M \oplus K$
- **Cryptanalysis** — breaking "secret codes"
  - ychrpyaprtgo is cracked to _______________, QED.
- **Crypto** — all of the above (and more)
  - More on non-repudiation (signature), authentication, identification, zero-knowledge, commitment, and more…
  - Any reference books?... Bruce Schneier, HAC

A cipher or cryptosystem is used for encrypting/decrypting a plaintext/ciphertext



Cryptanalysis

Basic assumption

- Known as **Kerckhoffs Principle**
- The system is completely known to the attacker
- Only the key is secret
- Crypto algorithms are not secret
- No “security through obscurity”

Objective of an attacker

- Identify secret key used to encrypt a ciphertext
- (OR) recover the plaintext of a ciphertext without the secret key

Examples of (Classical) Symmetric Key Encryption Algorithms – Classical Cryptography

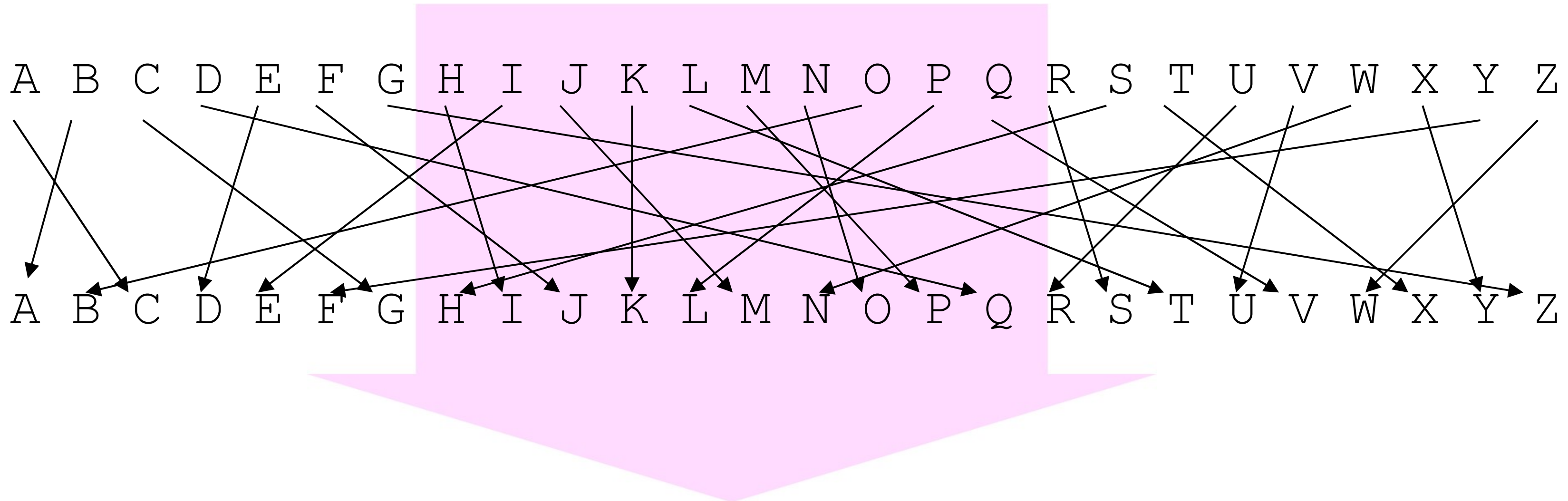
Ciphertexts:

1. IRXUVFRUHDAGVHYHABHDUVDIR
2. VSRQJHEREVTXDUHSDQWU

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Simple Substitution: each plaintext letter is substituted by a distinct ciphertext letter

EIMBULJIWLNYANJMVLIURAHIWAI



DEPARTMENTOFCOMPUTERSCIENCE

- What's the ciphertext of "**solutionstofinalexam**"?

An example of simple substitution…

"Encryption software is expensive…so we just rearranged all the letters on your keyboard."

An Example

Ciphertext (encrypted using simple substitution)

PBFPVYFBQXZTYFPBFEQJHDXXQVAPTPQJKTOYQWIPBVWLXTOXBTFXQWAX
BVCXQWAXFQJVWLEQNTOZQGGQLFXQWAKVWLXQWAEBIPBFXFQVXGTVJ
VWLBTPQWAEBFPBFHCVLXBQUFEVWLXGDPEQVPQGVPPBFTIXPFHXZHVFA
GFOTHFEFBQUFTDHZBQPOTHXTYFTODXQHFTDPTOGHFQPBQWAQJJTODX
QHFOQPWTBDHHIXQVAPBFZQHCFWPFHPBFIPBQWKFABVYYDZBOTHPBQP
QJTQOTOGHFQAPBFEQJHDXXQVAVXEBQPEFZBVFOJIWFFACFCCFHQWAUV
WFLQHGFXVAFXQHFUFHILTTAVWAFFAWTEVOITDHFHFQAITIXPFHXAFQHEF
ZQWGFLVWPTOFFA

Question: how secure is Simple Substitution?

Let's do some analysis…

- 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

- Question: how many distinct permutations are there? (in other words, how many distinct secret keys are in the key space?)

- Total number of possible permutations

26!

- 26! = 403,291,461,126,605,635,584,000,000 (27 digits) $\approx 2^{88}$
- Maybe… write a computer program to try all the possible keys exhaustively… (so-called **Brute-force Attack**)
- **Calculation**: suppose we have one million 3GHz PCs which can try 3 billion permutations per second, the machines will take **4,263 years** to try all the 26! permutations…
  - Not so efficient
- Question: any better cracking algorithm?

Statistical Attack / Character Frequency Attack

- An interesting observation on simple substitution: the relative letter frequencies do not change during encryption

- letters in an alphabet are not equally common
- in English, **e** and **t** are by far the most common letters
- Probability distribution of the 26 English letters (Beker and Piper, 1982)

letter	probability	letter	probability
A	.082	N	.067
B	.015	O	.075
C	.028	P	.019
D	.043	Q	.001
E	.127	R	.060
F	.022	S	.063
G	.020	T	.091
H	.061	U	.028
I	.070	V	.010
J	.002	W	.023
K	.008	X	.001
L	.040	Y	.020
M	.024	Z	.001

Basic Approach of Statistic Attack:

1.Identify possible encryptions of letter ‘e’ (the most common English letter)

2.Identify possible diagrams starting/finishing with letter ‘e’

3.Use trigrams (e.g. find ‘the’)

4.Identify word boundaries

Iterate the above for the 2nd most common English letter and so on.

Ciphertext:

PBFPVYFBQXZTYFPBFEQJHDXXQVAPTPQJKTOYQWIPBVWLXTOXBTFXQWAX
BVCXQWAXFQJVWLEQNTOZQGGQLFXQWAKVWLXQWAEBIPBFXFQVXGTVJV
WLBTPQWAEBFPBFHCVLXBQUFEVWLXGDPEQVPQGVPPBFTIXPFHXZHVFAG
FOTHFEFBQUFTDHZBQPOTHXTYFTODXQHFTDPTOGHFQPBQWAQJJTODXQH
FOQPWTBDHHIXQVAPBFZQHCFWPFHPBFIPBQWKFABVYYDZBOTHPBQPQJT
QOTOGHFQAPBFEQJHDXXQVAVXEBQPEFZBVFOJIWFFACFCCFHQWAUVWFL
QHGFXVAFXQHFUFHILTTAVWAFFAWTEVOITDHFHFQAITIXPFHXAFQHEFZQW
GFLVWPTOFFA

Ciphertext frequency counts:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
21	26	6	10	12	51	10	25	10	9	3	10	0	1	15	28	42	0	0	27	4	24	22	28	6	8

Question: How to build a symmetric key cryptosystem which is secure against statistical attack?

One-time Pad Encryption

Encryption: Plaintext ⊕ Key = Ciphertext

e=000 h=001 i=010 k=011 l=100 r=101 s=110 t=111

	h	e	i	l	h	i	t	l	e	r
Plaintext:	001	000	010	100	001	010	111	100	000	101
Key:	111	101	110	101	111	100	000	101	110	000
Ciphertext:	110	101	100	001	110	110	111	001	110	101
	s	r	l	h	s	s	t	h	s	r

e=000 h=001 i=010 k=011 l=100 r=101 s=110 t=111

Decryption: Ciphertext ⊕ Key = Plaintext

	s	r	l	h	s	s	t	h	s	r
Ciphertext:	110	101	100	001	110	110	111	001	110	101
Key:	111	101	110	101	111	100	000	101	110	000
Plaintext:	001	000	010	100	001	010	111	100	000	101
	h	e	i	l	h	i	t	l	e	r

- Pad must be random, **used only once**
- Pad has the same size as message

Questions: What are the current symmetric key cryptosystems?

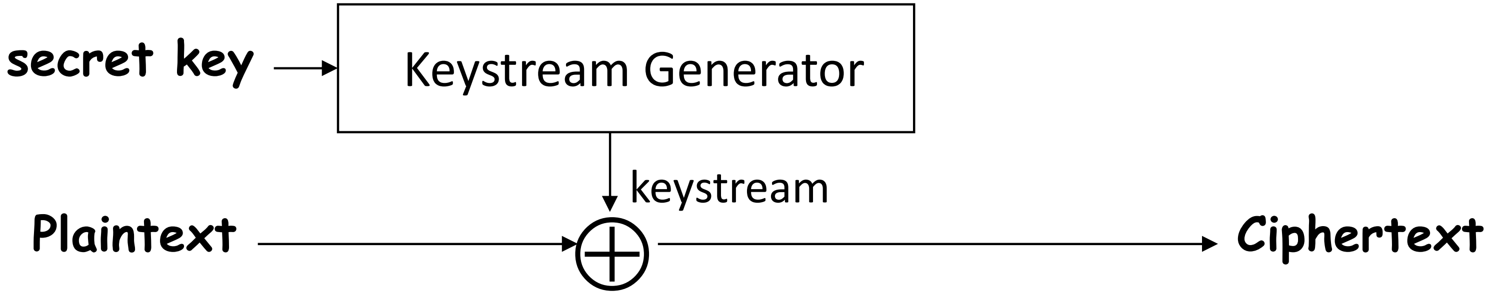
There are many…

They can be categorized into two types:

1. Stream Cipher

2. Block Cipher

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.

RC4

- A stream cipher
- **R**on's **c**ode version **4** (Ronald Rivest)
- Stream ciphers are generally faster than block ciphers
- RC4
  - Stage 1: RC4 initialization
  - Stage 2: RC4 keystream generation

RC4 Initialization

- Setup:

```
byte key[N];    // secret key (e.g. N = 16, i.e. 128-bit key)
byte K[256];   // keying material
byte S[256];   // internal states
```

- Initialization:

```
for i = 0 to 255
        S[i] = i
        K[i] = key[i (mod N)]
j = 0
for i = 0 to 255
        j = (j + S[i] + K[i]) mod 256
        swap(S[i], S[j])
i = j = 0
```

- S[] is the permutation of 0,1,...,255

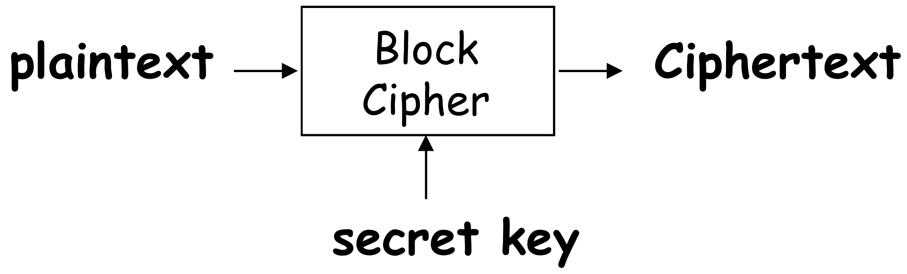
RC4 Keystream Generation

- To output a keystream byte, swap table elements and select a byte
- Keystream generation:

```
i = (i + 1) mod 256
j = (j + S[i]) mod 256
swap(S[i], S[j])
t = (S[i] + S[j]) mod 256
KeyStreamByteSelected = S[t]
```

- Use the KeyStreamByteSelected to do XOR with one byte of plaintext, then iterate the keystream generation steps above for getting another byte of keystream
- **Note:** Some research results show that the first 256 bytes must be discarded, otherwise attacker may be able to recover the key.

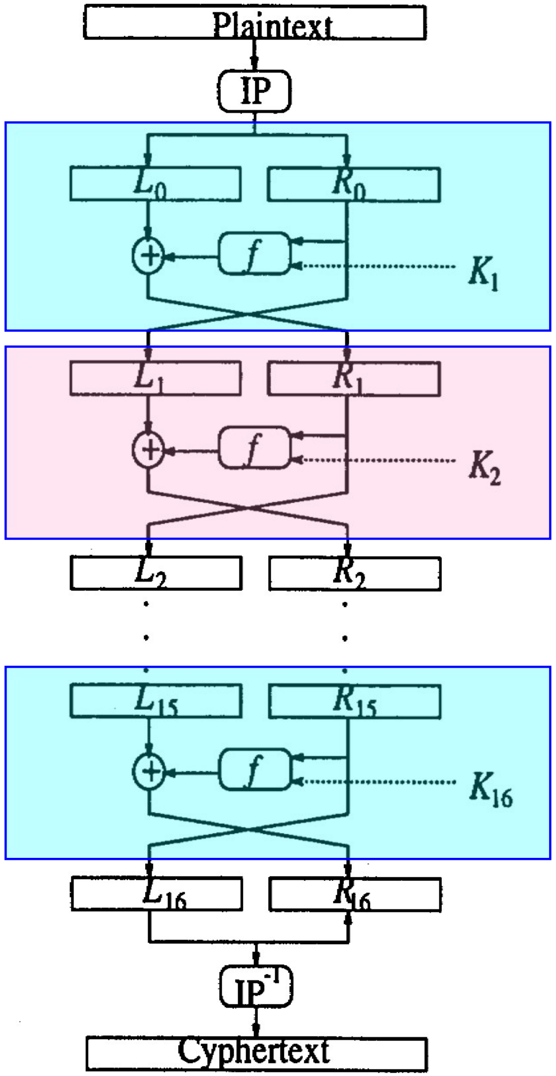
Block Ciphers

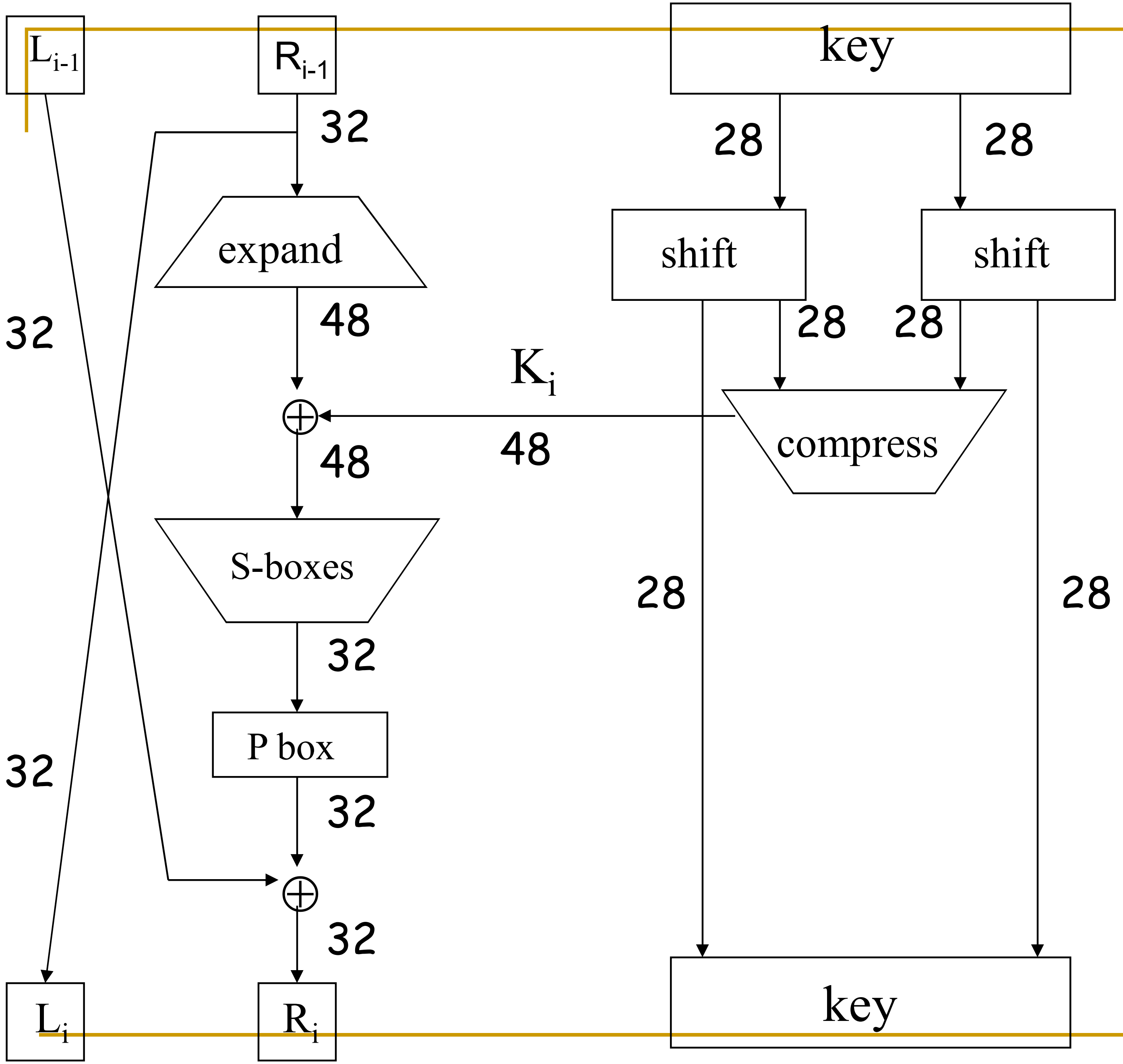


- A block cipher takes a *block* of **plaintext** and a **secret key**, produces a *block* of **ciphertext**.
- The key is **reused** for different plaintext blocks
- Typical block sizes: 64 bits, 128 bits, 192 bits, 256 bits
- Key sizes: 56 bits (DES), 128/192/256 bits (AES)
- Popular block ciphers: DES, 3DES, AES, Twofish, Serpent

(Iterated) Block Cipher

- Ciphertext obtained from plaintext by iterating a **round function**
- Input to round function consists of key and the output of previous round
- DES: 16 rounds of Feistel round function
  - Block size: 64 bits
  - Key size: 56 bits





One Round of DES

Security of DES

- Security of DES depends solely on the internals of *f*
- Thirty years of intense analysis has revealed no “back door”
- The most effective attack today against DES is still the exhaustive key search (aka bruteforce attack)

Exhaustive Key Search

- Given a plaintext x and corresponding ciphertext y, every possible key would be tested until a key K is found such that

  $$E(K, x) = y$$

  Note: there may be more than one such key K.
- For DES, total number of keys = $2^{56} \approx 7.2 \times 10^{16}$ keys
- Assume at the speed of 10^6 encryptions per second, it would need more than 1000 years to break DES.
- Diffie and Hellman postulated in 1977 that a DES cracking machine with 10^6 processors, each could test 10^6 keys per second, could be built for about US$20M.
  - This machine can break DES in about 10 hours.

Exhaustive Key Search

- In 1993, Michael Wiener presented a pipelined chip which tests 5×10^7 DES keys per second.
  - Each chip could cost US$10 and a frame of 5760 chips would cost about $100K.

Machine Unit Cost	Expected Time
$100,000 (1 frame)	35 hours
$1,000,000 (10 frames)	3.5 hours
$10,000,000 (100 frames)	21 minutes

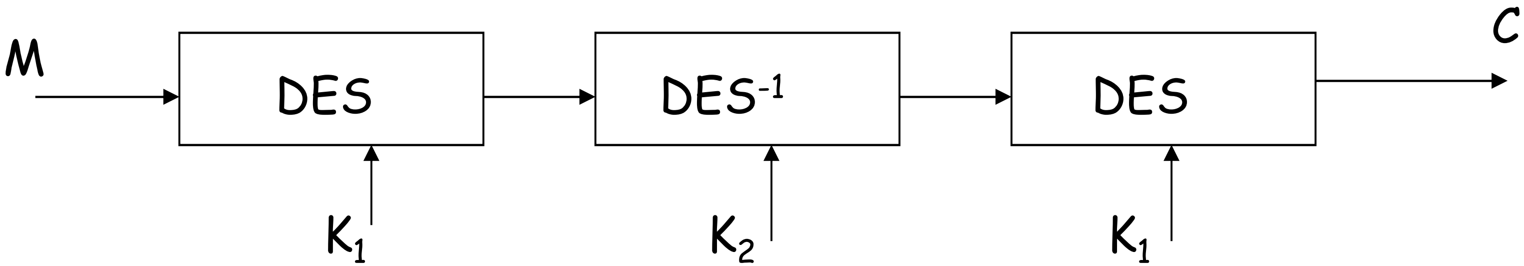
- In 1998, DES cracker (nicknamed "Deep Crack" http://en.wikipedia.org/wiki/EFF_DES_cracker) was built by the Electronic Frontier Foundation (EFF).
  - It performs 2^{56} DES operations in 56 hours.
  - Cost: US$250K (first piece), US$50K - $75K (duplicates).
- Software version of DES cracking effort can be found at http://www.distributed.net/des/

What Should We Use Today?

- 3DES (or Triple DES)
- AES (or Rijndael)
- Other candidates
  - Twofish
  - RC6
  - Serpent

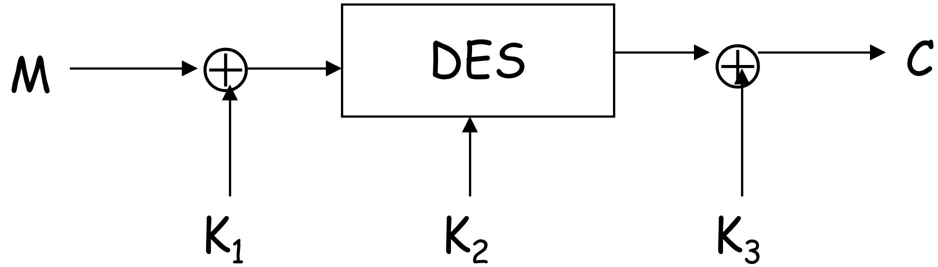
Triple DES and DESX

- Triple DES: two 56-bit keys



- DESX: three keys

$$C = K_3 \oplus DES(K_2, M \oplus K_1)$$



- Similar security to DES using differential cryptanalysis and linear cryptanalysis, which are theoretical attacks
- But much harder to break using exhaustive key search than DES.

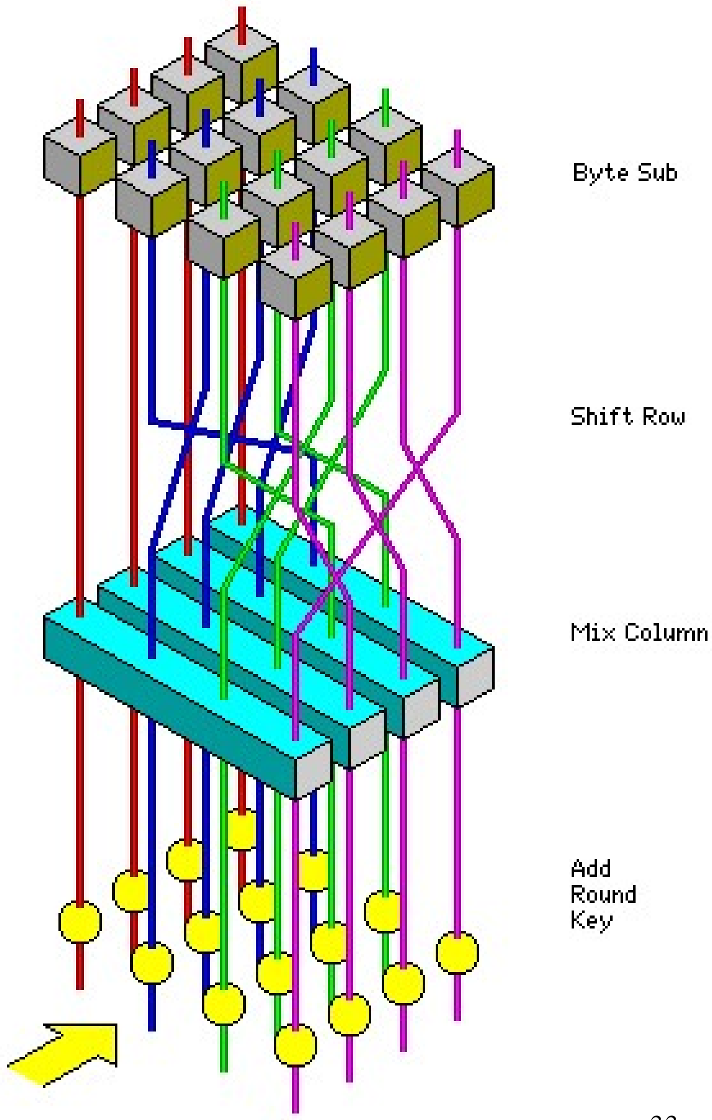
What are the sizes of K1, K2, and K3?

Advanced Encryption Standard

- Replacement for DES
- AES competition (late 90's)
  - NSA openly involved
  - Many strong algorithms were proposed and cryptanalyzed publicly
  - Rijndael Algorithm was ultimately selected
    - Pronounced like "Rain Doll" or "Rhine Doll"
- Iterated block cipher (like DES)
- Not using Feistel round function (unlike DES)

AES (Advanced Encryption Standard)

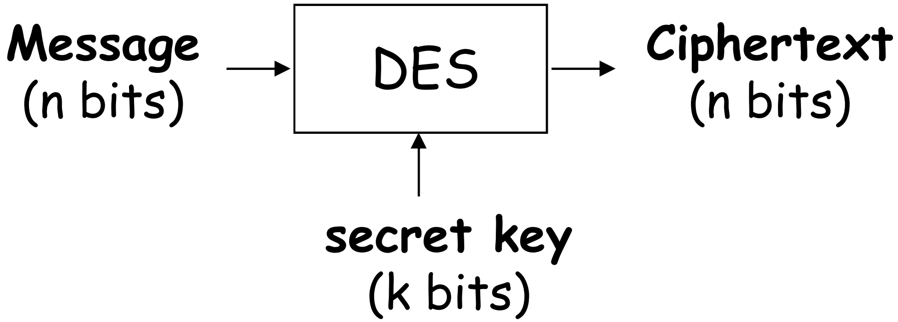
- **Replacement of DES**
- **Block size:** 128 bits
- **Key length:** 128, 192 or 256 bits (independent of block size)
- 10 to 14 rounds (depends on key length)
- Each round uses 4 functions (in 3 "layers")
  - ByteSub (nonlinear layer)
  - ShiftRow (linear mixing layer)
  - MixColumn (nonlinear layer)
  - AddRoundKey (key addition layer)



Key Space

- The Key Space of a cipher is the set of all possible and distinct secret keys
  - E.g. The key space of DES is all distinct 56-bit binary strings
  - E.g. The size of the key space of simple substitution for case-insensitive English alphabet is 26!
- What's the key space size of AES?
- What's the key space size of RC4?

Multiple Blocks



- How to encrypt multiple blocks?
- A new key for each block?
  - As bad as (or worse than) the one-time pad!
- Encrypt each block independently?
- Make encryption depend on previous block(s), i.e., "chain" the blocks together?
- How to handle partial blocks?

Modes of Operation

- Many modes of operation — we discuss three
- Electronic Codebook (**ECB**) mode
  - Obvious thing to do
  - Encrypt each block independently
  - There is a serious weakness
- Cipher Block Chaining (**CBC**) mode
  - Chain the blocks together
  - More secure than ECB
- Counter Mode (**CTR**) mode
  - Acts like a stream cipher
  - Popular for random access

ECB Mode

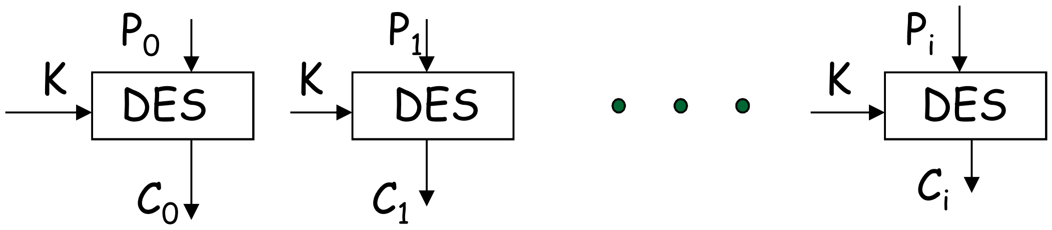
- Notations: $C=E(K, P)$ $P=D(K,C)$
- Given plaintext P = $P_0,P_1,\ldots,P_m,\ldots$ (in blocks)
- Obvious way of using a block cipher is to encrypt plaintext blocks independently

Encrypt

$C_0 = E(K, P_0)$,
$C_1 = E(K, P_1)$,
$C_2 = E(K, P_2),\ldots$

Decrypt

$P_0 = D(K, C_0)$,
$P_1 = D(K, C_1)$,
$P_2 = D(K, C_2),\ldots$



ECB Cut and Paste Attack

- Suppose plaintext is

    `Alice digs Bob. Trudy digs Tom.`

- Assuming 64-bit blocks and 8-bit ASCII:

    P_0 = "`Alice di`", P_1 = "`gs Bob. `",

    P_2 = "`Trudy di`", P_3 = "`gs Tom. `"

- Ciphertext: C_0,C_1,C_2,C_3
- Trudy cuts and pastes: C_0,C_3,C_2,C_1
- Decrypts as

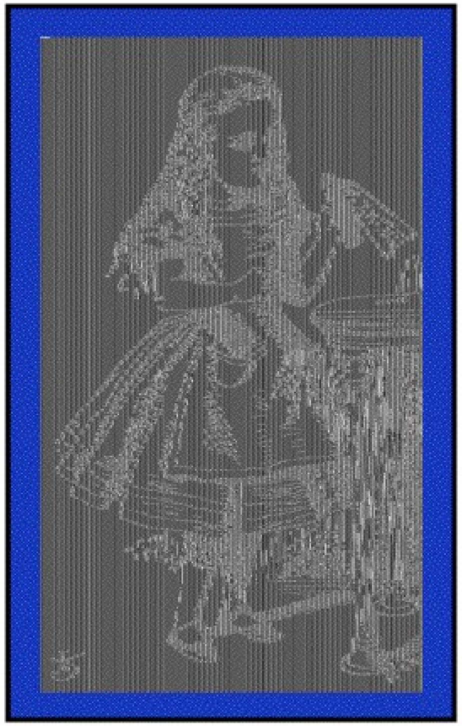
    `Alice digs Tom. Trudy digs Bob.`

ECB Weakness

- Suppose $P_i = P_j$
- Then $C_i = C_j$ and Trudy knows $P_i = P_j$
- This gives Trudy some information, even if she does not know P_i or P_j
- Is this a serious issue?

Alice Hates ECB Mode

- Alice’s uncompressed image, Alice ECB encrypted



- Why does this happen?
- Same plaintext block ⇒ same ciphertext!

CBC Mode

- Blocks are “chained” together
- A random initialization vector, or IV, is required to initialize CBC mode
- IV is random, but is not a secret

Encryption

$C_0 = E(K, IV \oplus P_0)$,

$C_1 = E(K, C_0 \oplus P_1)$,

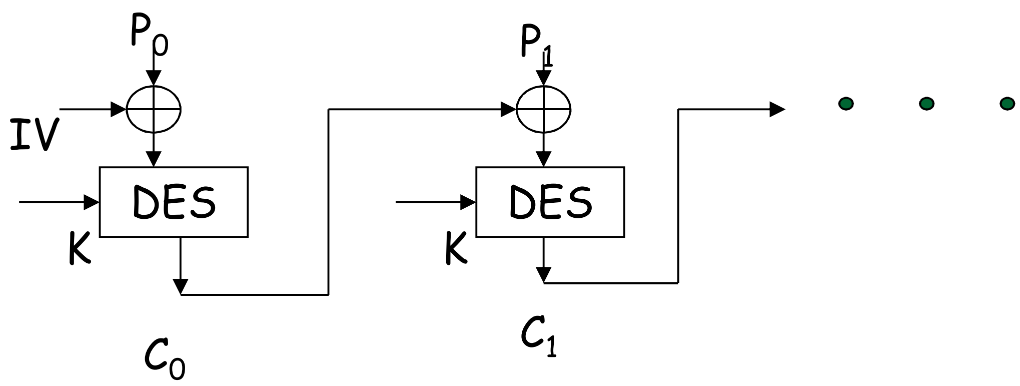
$C_2 = E(K, C_1 \oplus P_2), \ldots$

Decryption

$P_0 = IV \oplus D(K, C_0)$,

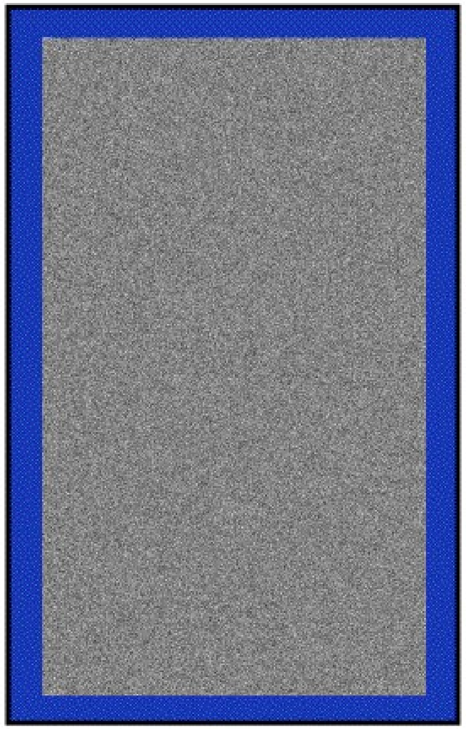
$P_1 = C_0 \oplus D(K, C_1)$,

$P_2 = C_1 \oplus D(K, C_2), \ldots$



Alice Likes CBC Mode

- Alice's uncompressed image, Alice CBC encrypted



- Why does this happen?
- Same plaintext yields different ciphertext!

Counter Mode (CTR)

- Use block cipher like stream cipher

Encryption

$C_0 = P_0 \oplus E(K, IV)$,

$C_1 = P_1 \oplus E(K, IV+1)$,

$C_2 = P_2 \oplus E(K, IV+2), \ldots$

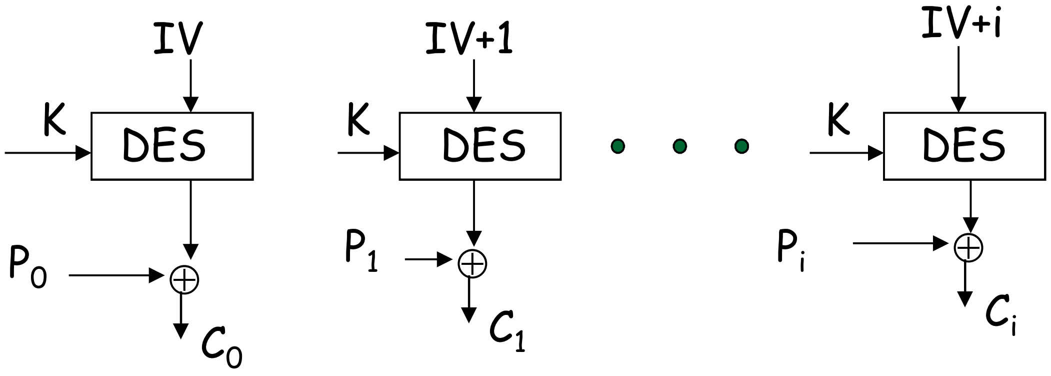
Decryption

$P_0 = C_0 \oplus E(K, IV)$,

$P_1 = C_1 \oplus E(K, IV+1)$,

$P_2 = C_2 \oplus E(K, IV+2), \ldots$

- CTR is good for random access (both READ and WRITE)
- CBC is good for random READ only, but not WRITE



Summary

- **Kerckhoffs Principle**
- **Simple Substitution Encryption and statistical attack**
- **One-time Pad Encryption**
- **Stream Cipher: RC4**
- **Block Cipher: DES, AES**
- **Key Space**
- **Modes of Operation**