Cryptography Basics

Part 1: Concepts

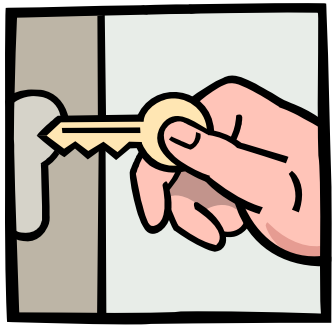
Contents

- Cryptography goals
- Encryption principles
- Encryption quality
- Public key cryptography

Next week:

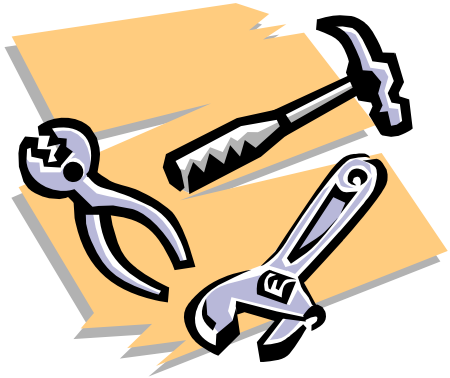
- Example algorithms
  - DES, AES, AES
- Encrypting larger messages
- `Provably secure' crypto

Cryptology:



Cryptography

The art of making



Cryptanalysis

The art of breaking

Security Goals and Cryptography

- Confidentiality
- Authenticity
- Data integrity
- Non-repudiation
- *Privacy*
- *Availability*

Example: What's the message

Greetings to all at Oxford. Many thanks for your letter and for the summer examination package all Entry forms and Fess Forms should be ready for final dispatch to the Syndicate by Friday 20th or at the very latest, I'm told, by the 21st. Admin has improved here, though there's room for improvement still; just give us all two or three more years and we'll really show you! Please don't let these wretched 16+ proposals destroy your basic A and O pattern. Certainly this sort of change, If implemented immediately would bring chaos.

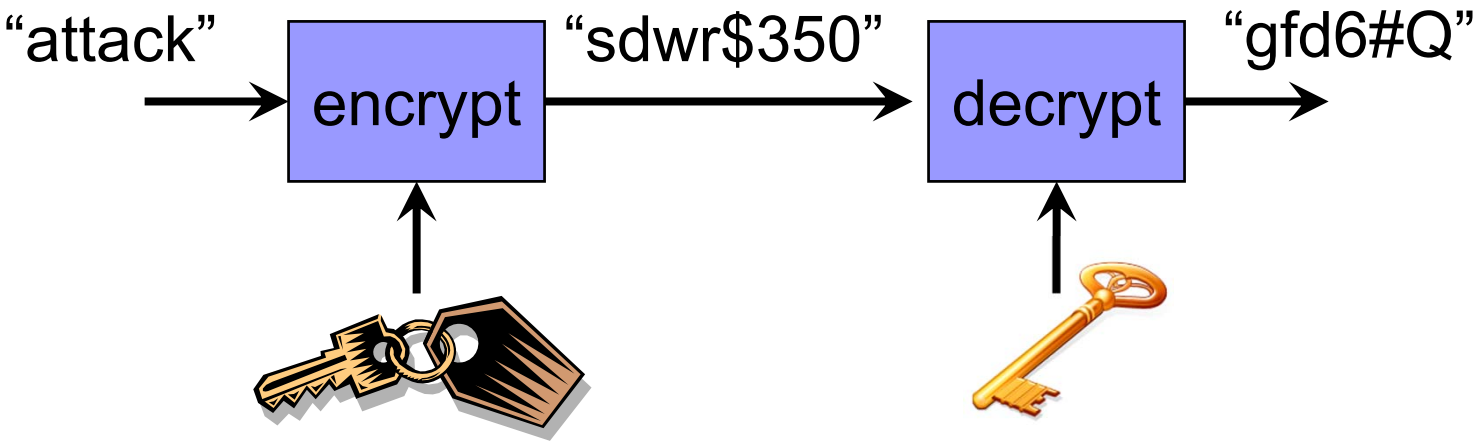
Another Example: What's the message

Welcome back to Oxford. Thanks again, this letter explains the winter examination method and its related forms. Early submission does guarantee full and early feedback but does not influence the grading of the quality of the work done. A full grade report will be available once the deadline for submissions has passed. In it the evaluation is explained. The evaluation is final as the criteria for the work are now known.

Final Example: What's the message

A final greeting to our Oxford graduates. Though with a slight delay, we hope this letter finds you well. The new variation in the forms attached shows how our alumni will continue to play a key role in our school and will not be forgotten. Instead we hope that you continue to work with us, and any contribution that you can bring, either directly or indirectly, will be appreciated.

Algorithms + keys



Cipher (aka cryptosystem)

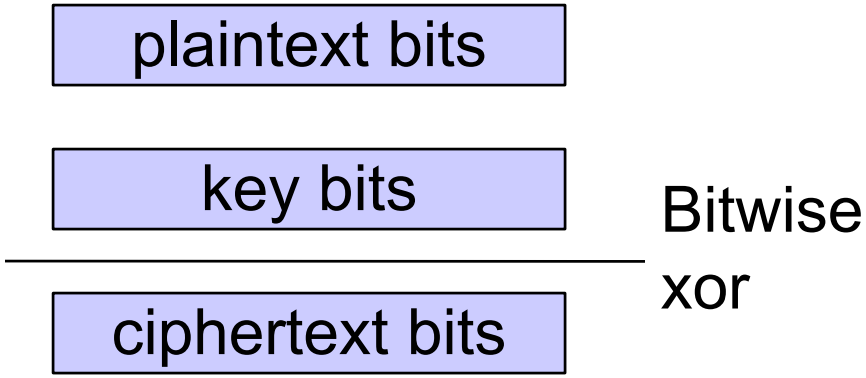
“Public” algorithm + Secret keys (Kerckhoffs’ principle)

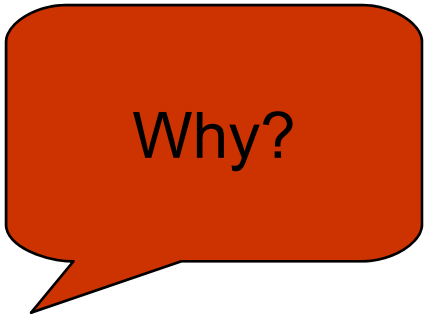
When is message `safe'?

- Suggestion 1: `cannot know the message'.
  - *Kill the king with a @#$%~!.*
- Suggestion 2: `cannot know even a single bit'.
  - 99% chance "Kill the king", 1% "Drink coffee"...
- *... lets find a definition...*
- For ciphertext each plaintext equally likely
  - Can this be done?

Yes(*)!: One time pad

- Vernam's one time pad is information theoretically secure





Note: random key equally long as message

XOR

In 1	In 2	Out
0	0	0
0	1	1
1	0	1
1	1	0

- XOR truth table:
- Addition modulo 2
- Property: (c + k) + k = c
  - Repeat operation to `undo'.
- If k `random'
  - (c+k) random
  - independent of c (!)

Some History: the Caesar cipher

- Monoalphabetic substitution
- Replace letter by letter 3 places further

Plaintext	A	B	C	D	E	F	G	H	…
Ciphertext	D	E	F	G	H	I	J	K	…

- Example:

"attackatdawn" → "dwwdfndwgdzq"

- Letter frequency undisturbed
- Nr of keys: 26 (25)

A=1, B=2, C=3, …
Encrypt: C = P+3
Decrypt: P = C-3

Vigenere cipher

- Polyalphabetic substitution
- Key is keyword
- Encrypt: Add keyword (letter by letter)
  - Modulo 26 with A=0, B=1, etc.
- Decrypt: Subtract keyword
- Example

```
  wearediscoveredsaveyourself
  deceptivedeceptivedeceptive
+ ---------------------------
  ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```

Cryptanalysis – plaintext structure

- (English) Text
  - Distribution of characters known
  - Distribution of bi-graphs also known:
- Data
  - Format known

E: 12%
T: 9 %
A,I,N,O,R: 8%

TH: 3.2%
HE: 3.1 %
ER: 2.1%

```
<account>87539</account
>
<amount>1234</amount>
```

Transposition cipher

- Change order of letters in the message

“meet me after the toga party”

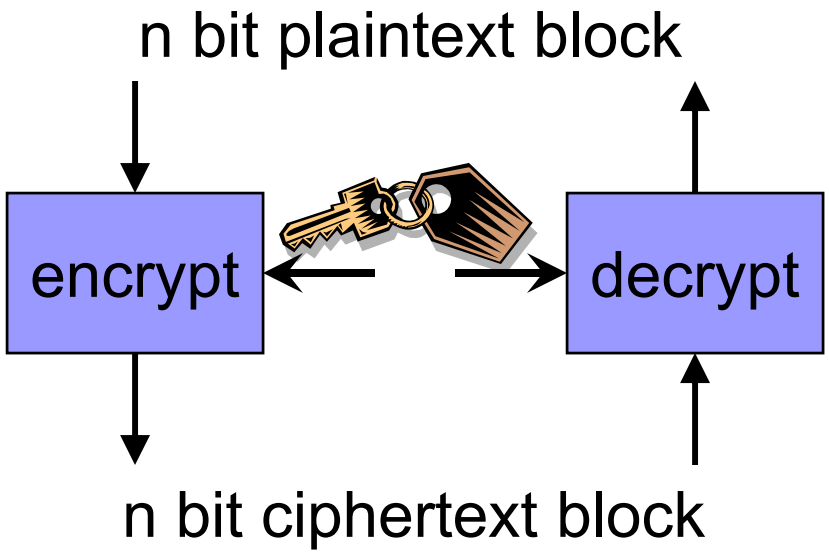
M e m a t r h t g p r y

e t e f e t e o a a t

“mematrhtgpryetefeteoaat”

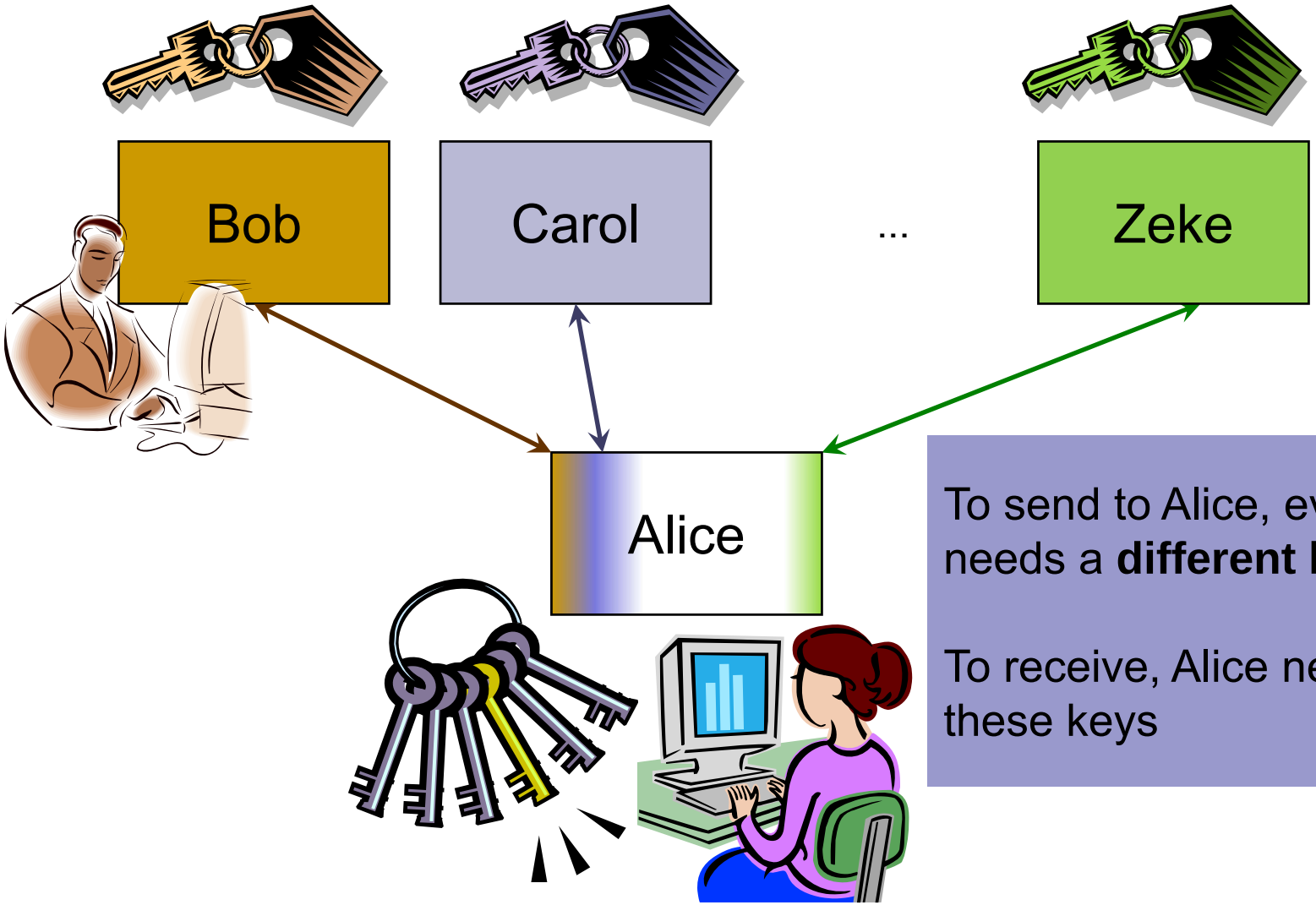
Modern Block Cipher

- Principle: Combine
  - Confusion (substitution)
  - Diffusion (transposition)
- Design: Iterate a round function
- Two common types:
  - Feistel network (e.g. DES)
  - Substitution-permutation network (e.g. AES)



More on this next week – Now first: asymmetric (public key) cryptography

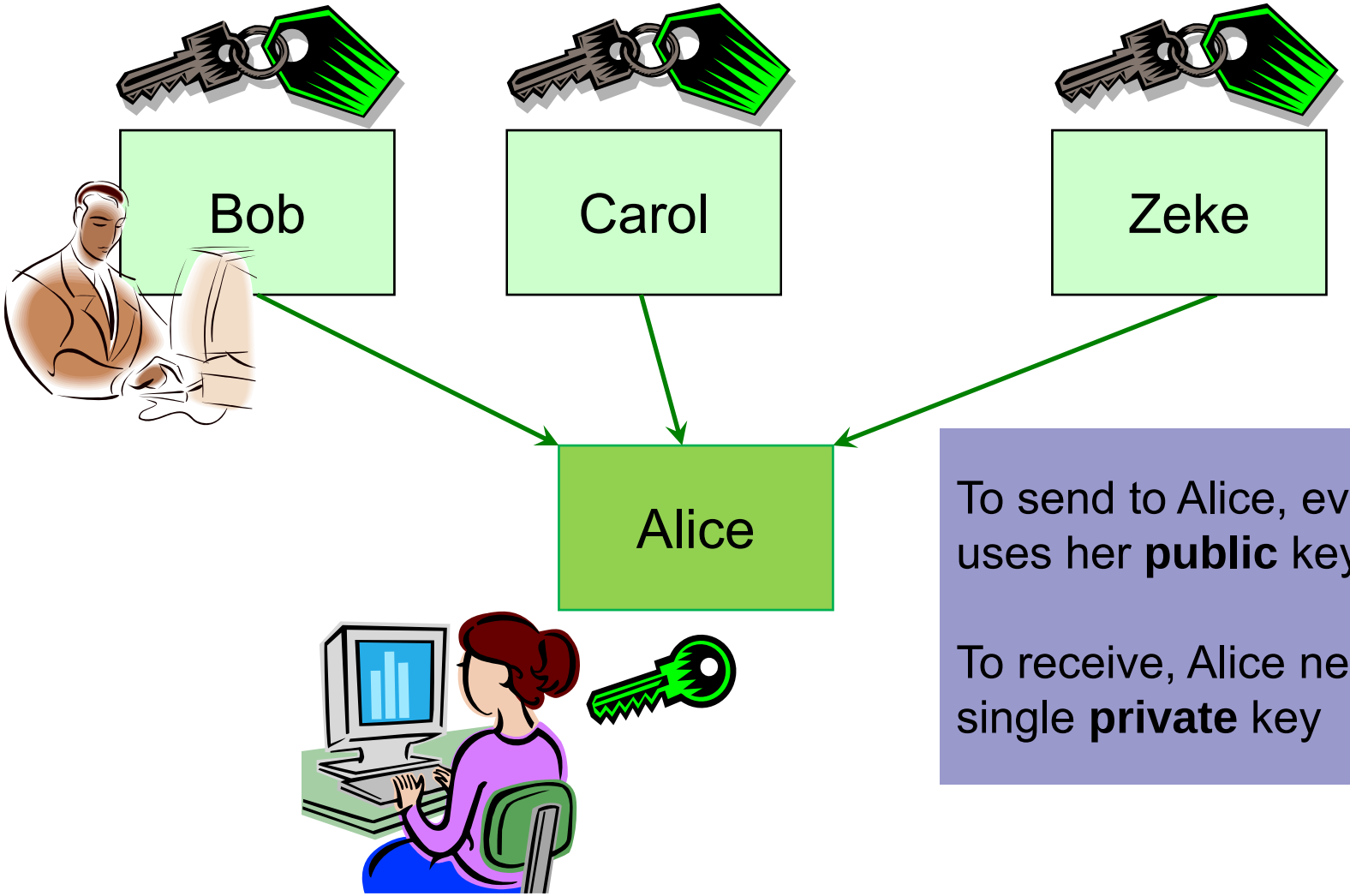
Many symmetric keys needed



To send to Alice, everyone needs a **different key**

To receive, Alice needs all these keys

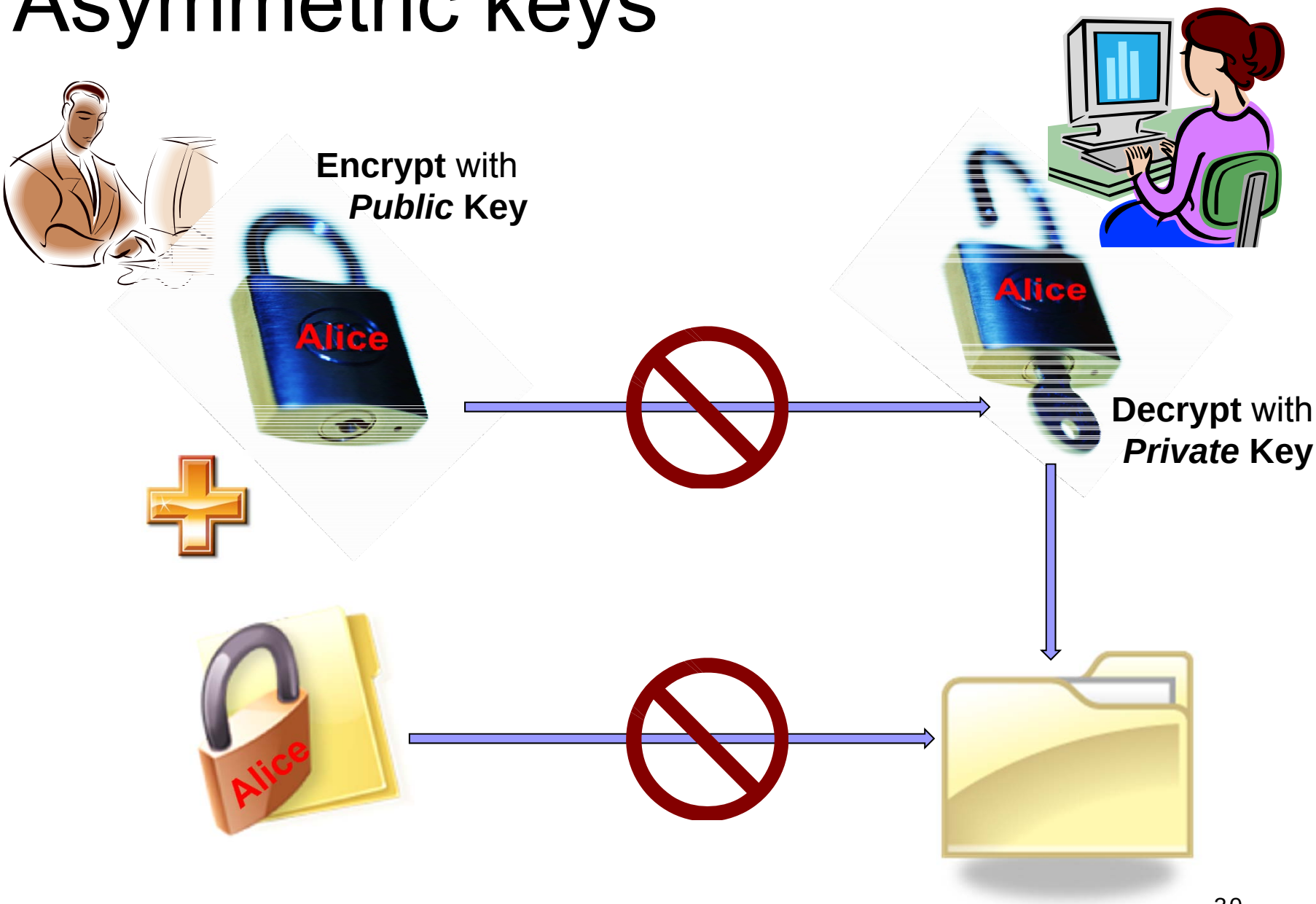
Asymmetric (public) key



To send to Alice, everyone uses her **public** key

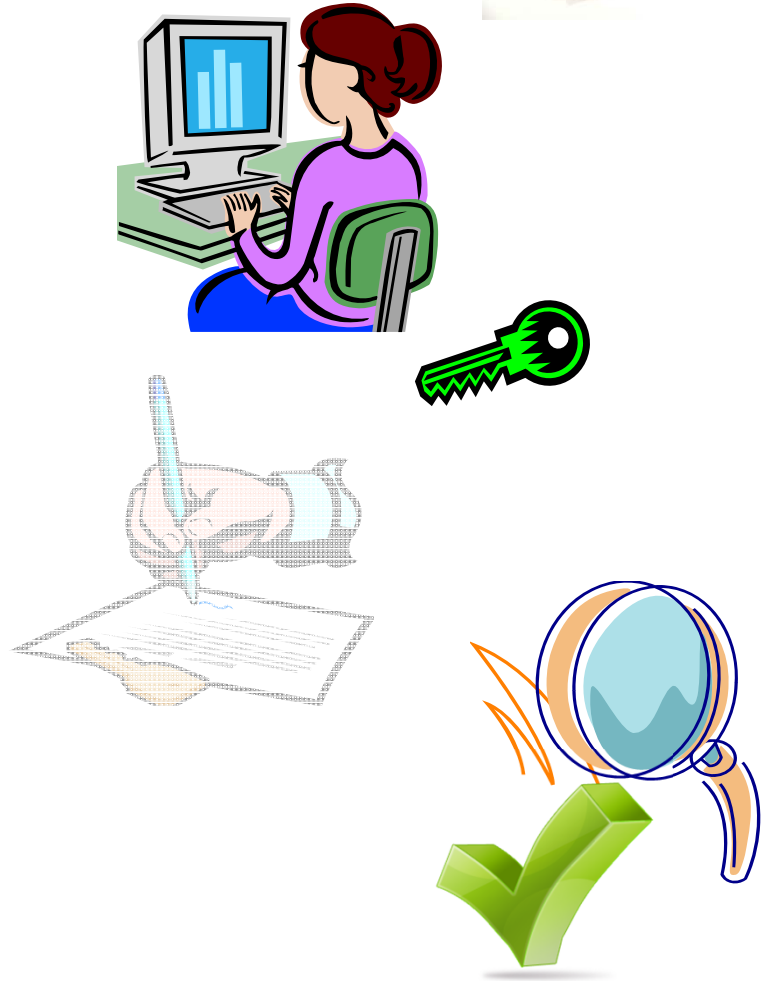
To receive, Alice needs a single **private** key

Asymmetric keys



Authenticity - Symmetric

Authenticity - ASymmetric



All can `sign' only Alice check: message *for* Alice.

Authenticity - ASymmetric

Digital signature: reverse role encryption – decryption

Alice can Sign, All can check: is a message *from* Alice

Key

	Asymmetric	Symmetric
Distribution 	Obtain public key - Authenticity	Establish shared key - Confidentiality
Storage 	Public keys - Tampering Private keys - Confidentiality	Many keys - Confidentiality - Tampering
Revocation 	Don't know where - Check key status	Bilateral

Diffie Hellman key exchange (1976)

$$a^x \bmod p \longrightarrow r$$

$$r \longleftarrow a^y \bmod p$$

$$key = r^x \bmod p$$

$$key = r^y \bmod p$$

Basis: Discrete log is hard

(for large numbers – e.g. 1024 bits)

$$a,\ b,\ \ b = a^i \bmod\ \ p$$

DH - Soundness and Security

- Alice key equals

$$r^x \bmod p = \left(a^y \bmod p\right)^x \bmod p = a^{xy} \bmod p$$

- Bobs key equals

$$r^y \bmod p = \left(a^x \bmod p\right)^y \bmod p = a^{xy} \bmod p$$

- Eavesdropper sees

$$a^x \bmod p$$

$$a^y \bmod p$$

- Vulnerable to man-in-the-middle attack

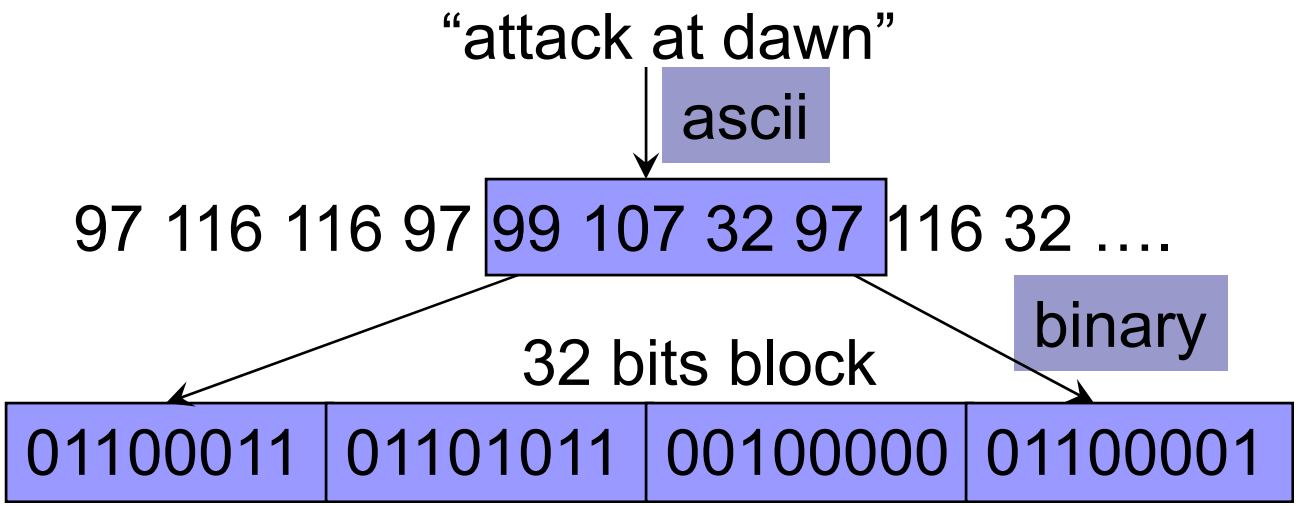
A		E		B
$\rightarrow$	a^x	$\rightarrow$	$a^{x'}$	$\rightarrow$
$\leftarrow$	$a^{y'}$	$\leftarrow$	a^y	$\leftarrow$

Cryptography Basics

Part 2: Block Modes

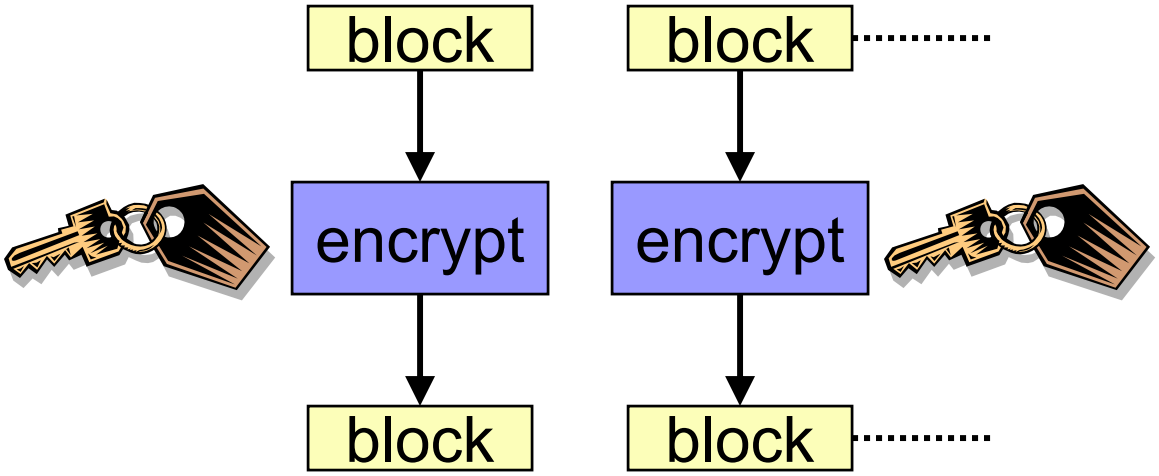
Encrypting Larger messages

- Seen methods to encrypt block
- Split into blocks (padding to fill last block)
- Treat blocks separately?



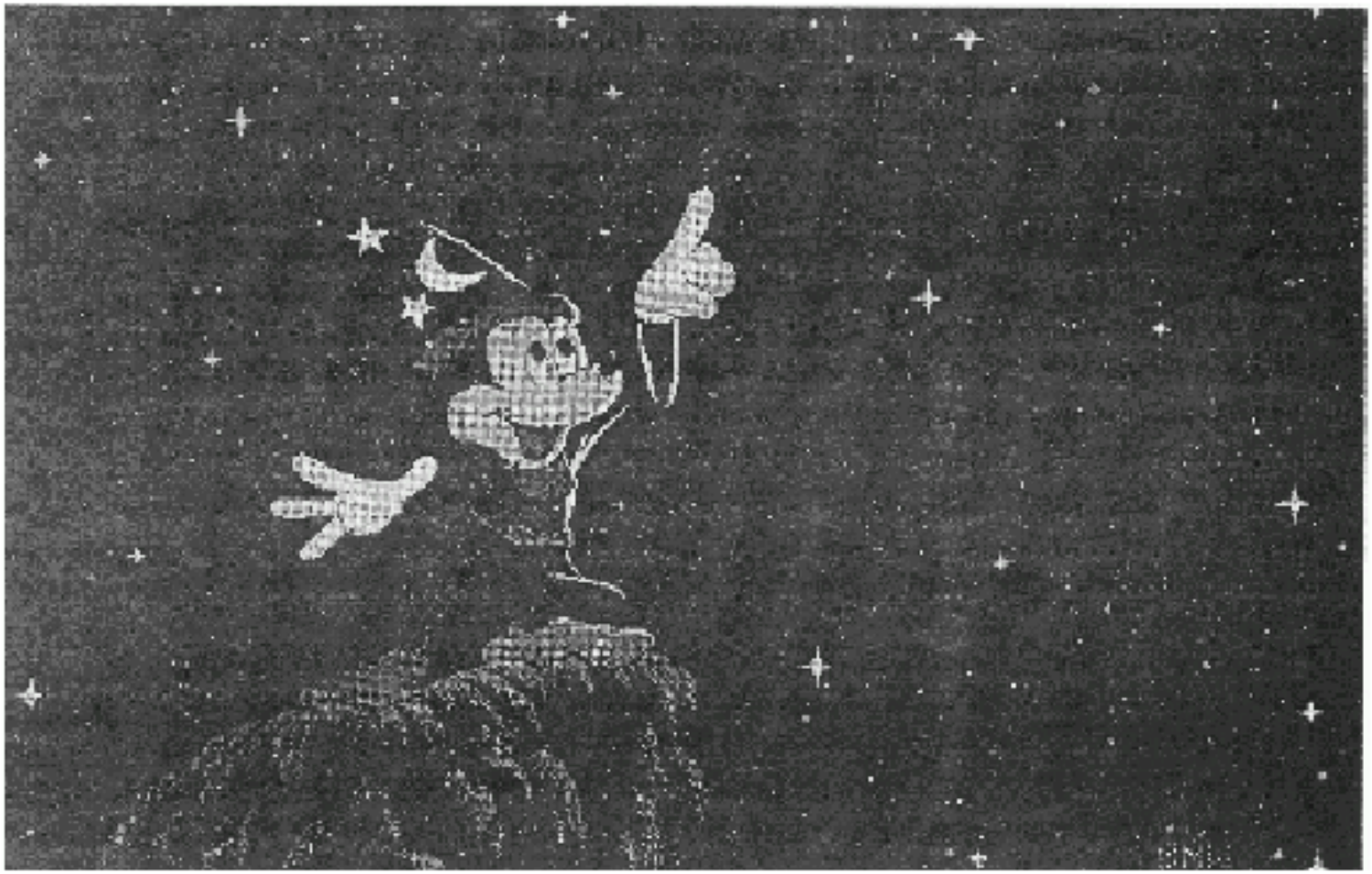
Block representation of text

ECB mode



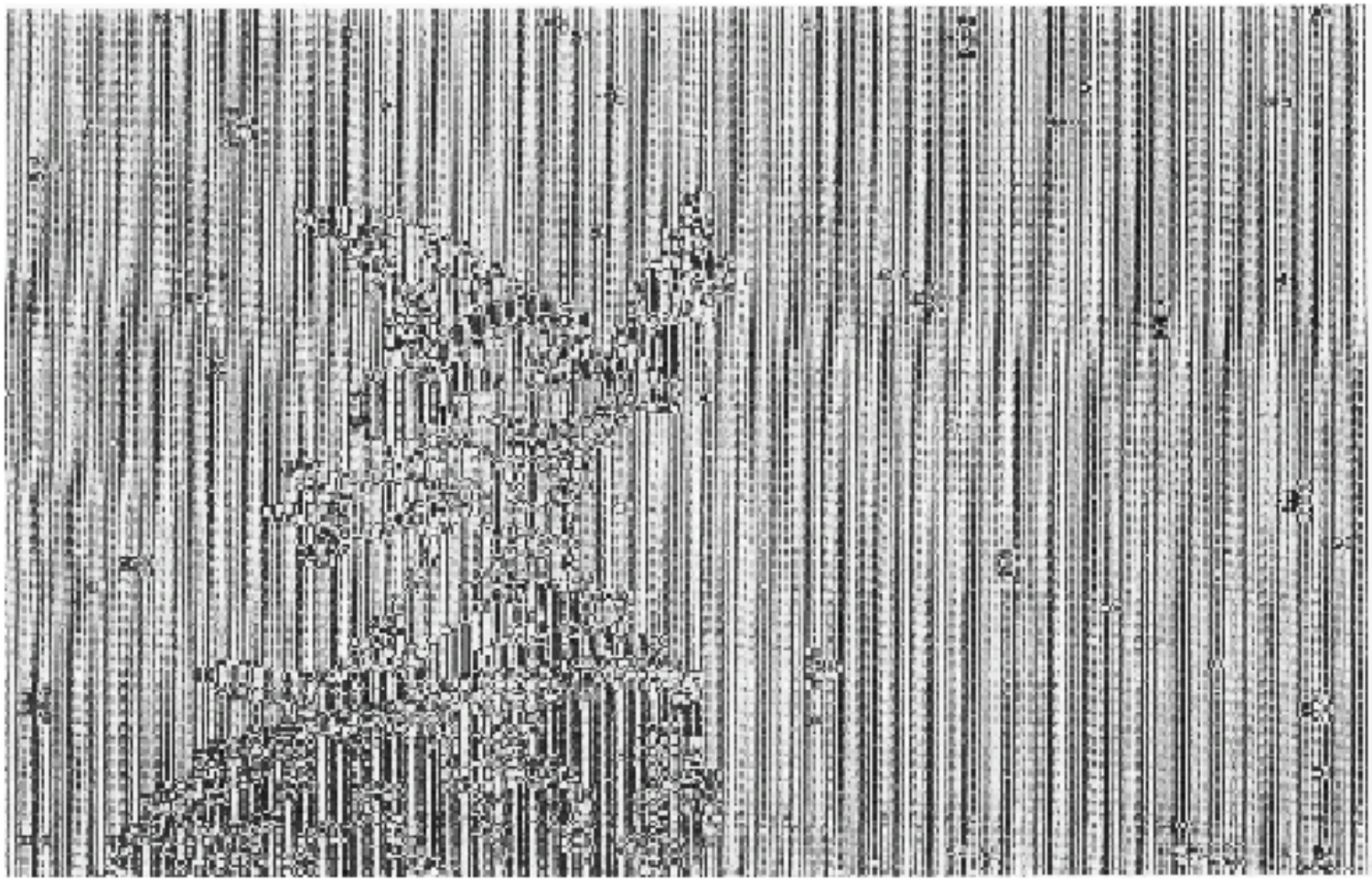
- Same plaintext block maps to same ciphertext block
  - Reordering, replacing possible
- No error propagation
  - Bit changes only
  - Bit deletions/omissions are a problem

Example: Mickey Mouse



- Original picture

Example: Mickey Mouse

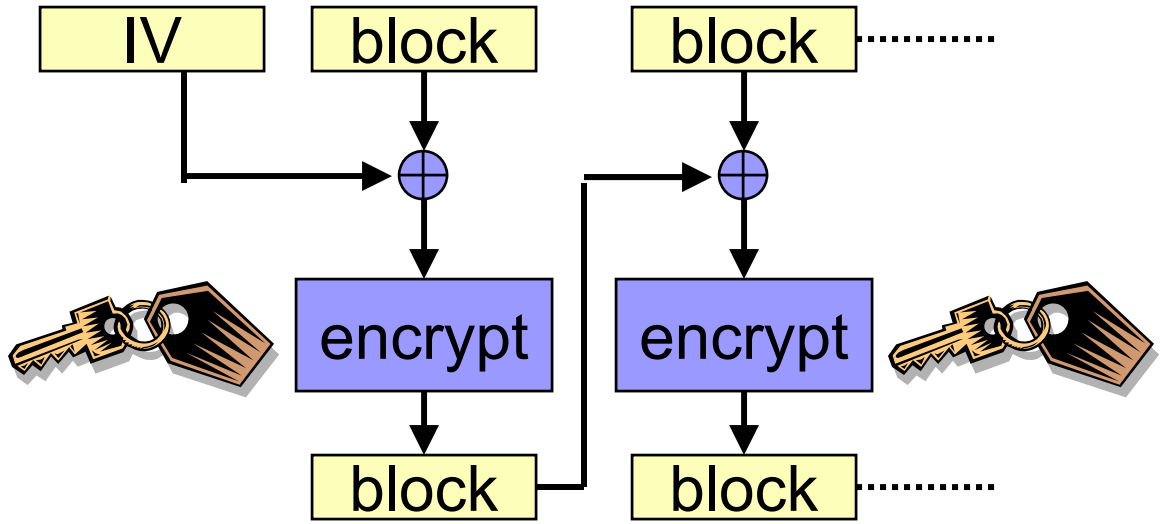


- Encrypted in ECB mode

Encrypting larger messages

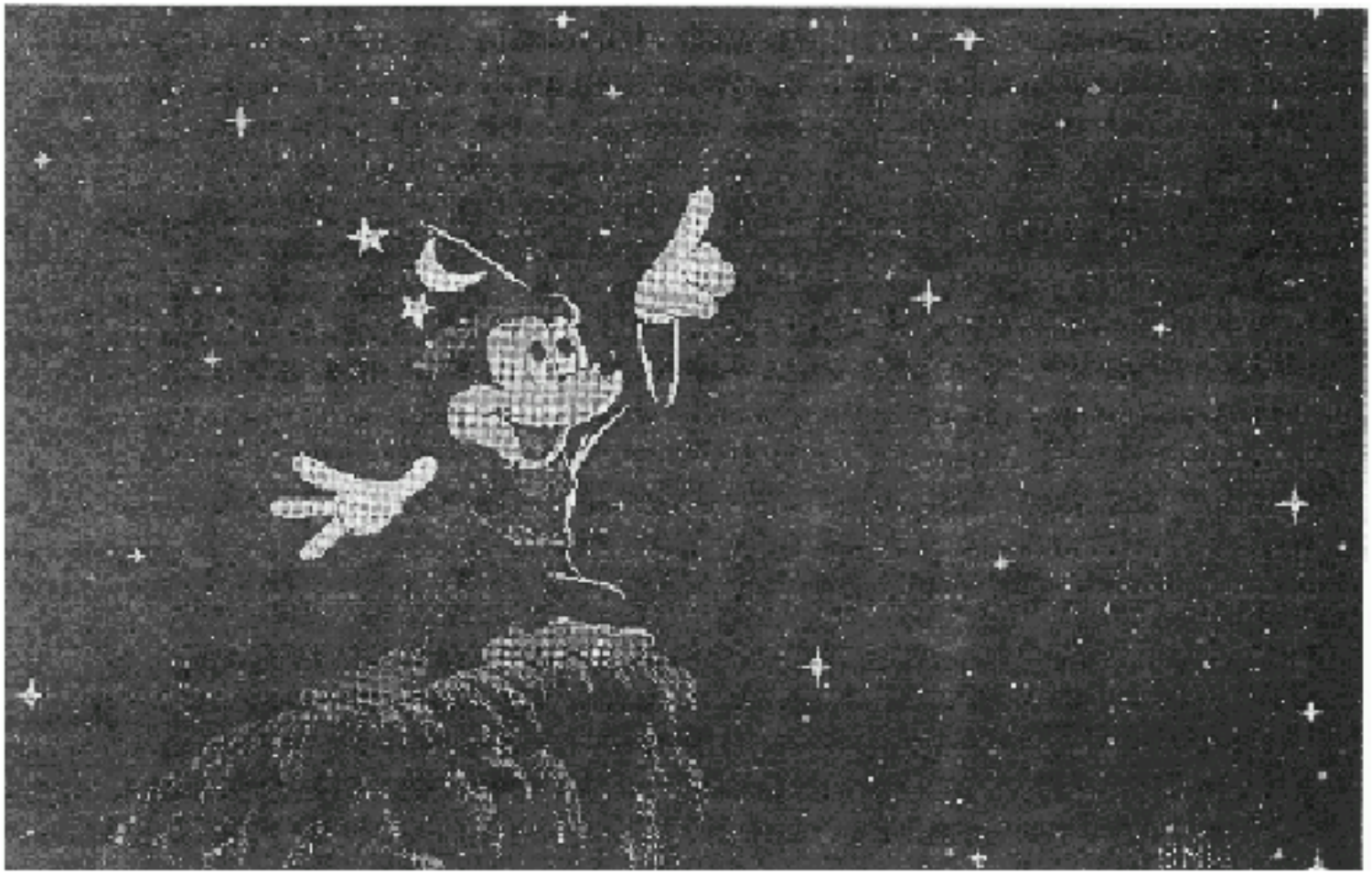
- Operation modes
  - Electronic codebook (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)

CBC mode



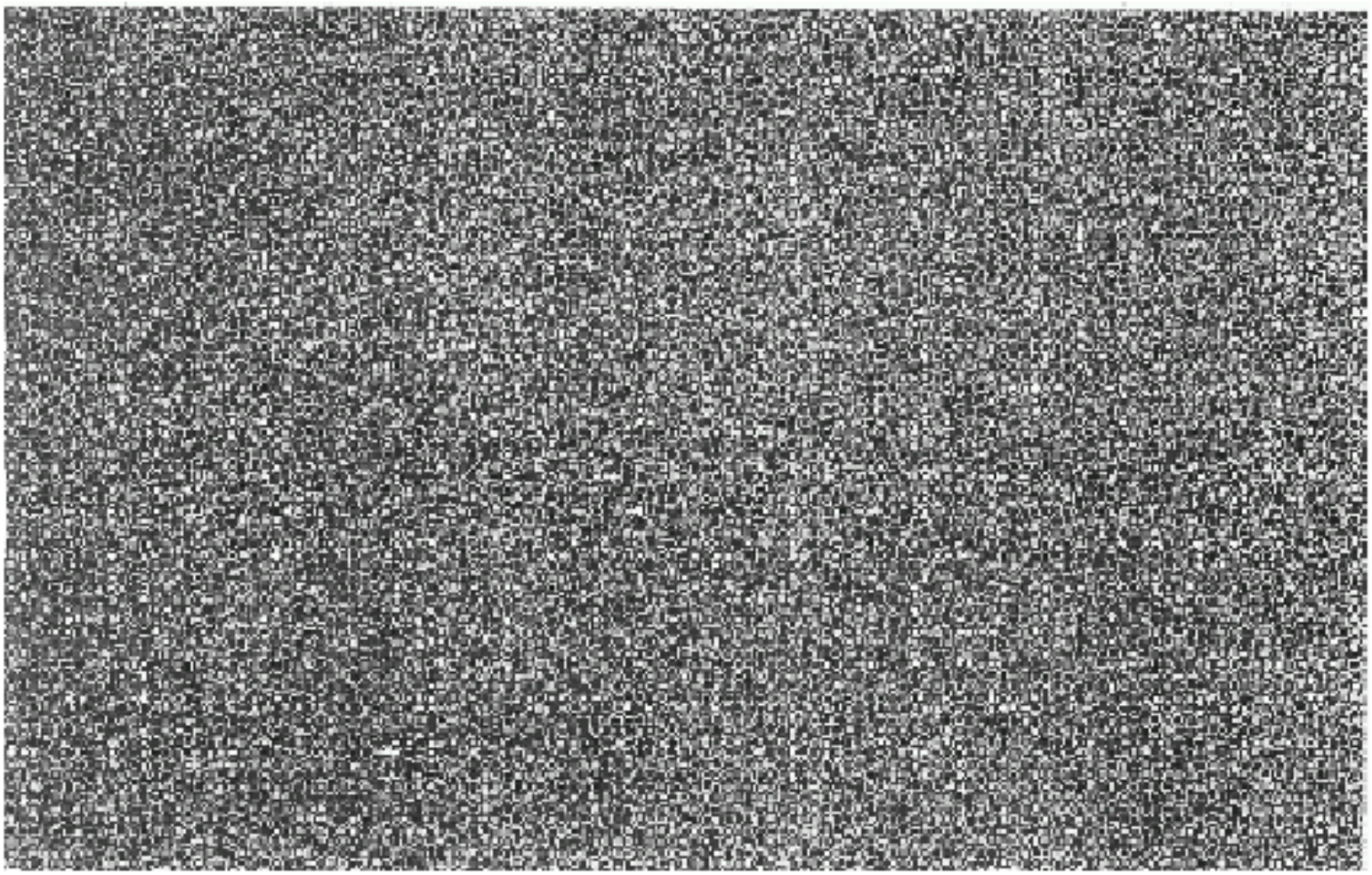
- **Same plaintext block maps to different ciphertext block**
  - *Reordering, replacing not possible*
    - *Depending on previous block*
- **Limited error propagation**
  - *Affects only current and next block*

Example: Mickey Mouse



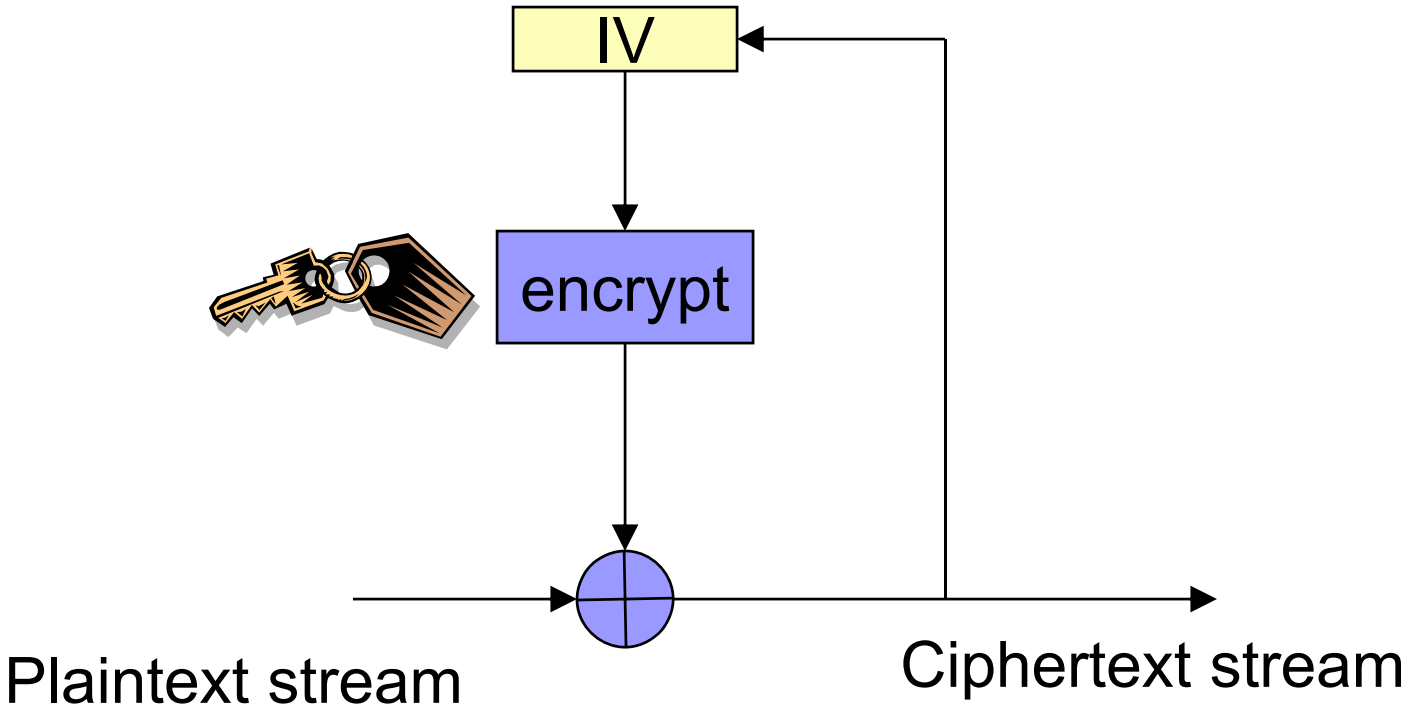
- Original picture

Example: Mickey Mouse



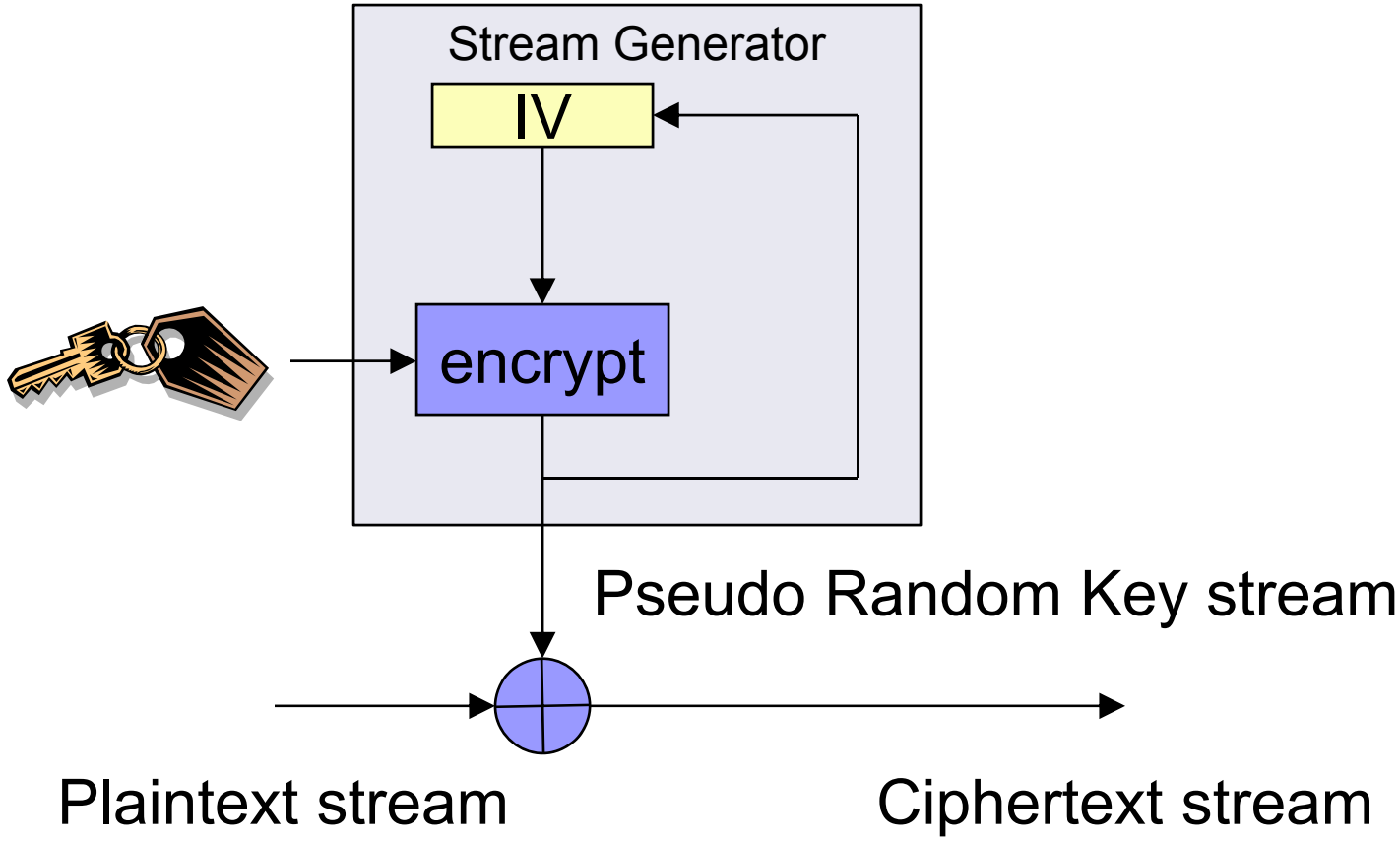
- Encrypted in CBC mode

CFB mode



- Self-synchronizing

Stream Ciphers and OFB mode



Stream ciphers

- Fast and `easy' in hardware
- (Almost) no buffering
- No error propagation

- Most stream ciphers are confidential
  - GSM A5/1 -- broken!
  - Military

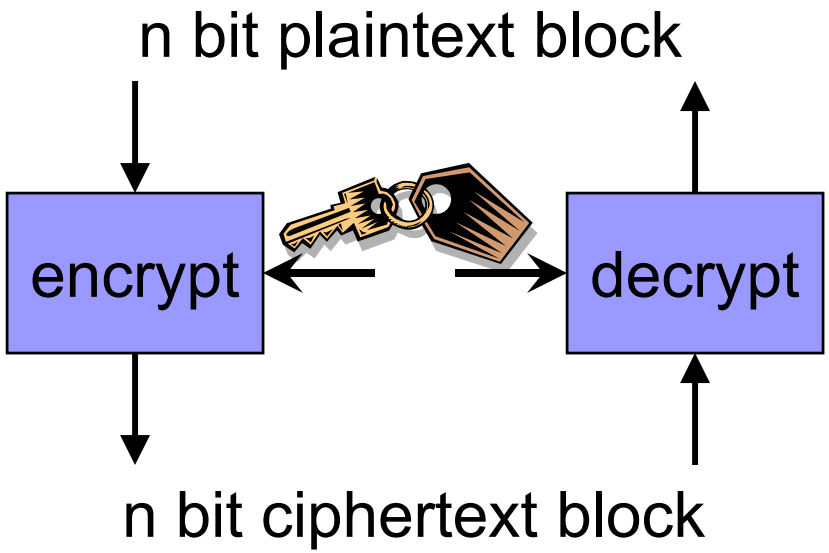
- Related: Random number generation

Cryptography Basics

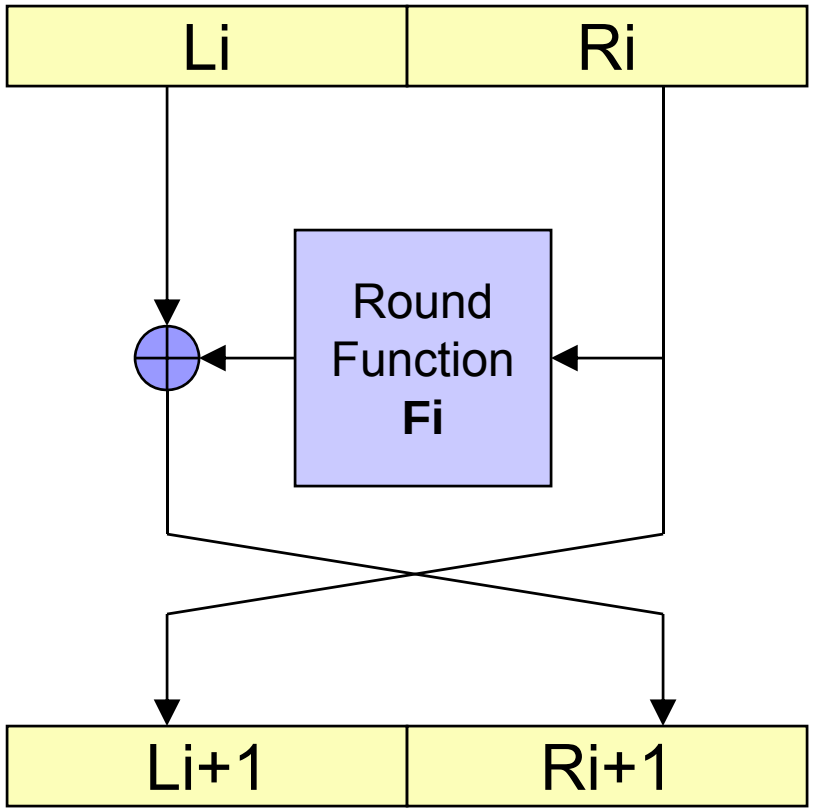
Part 3: Algorithms

Modern Block Cipher

- Principle: Combine
  - Confusion (substitution)
  - Diffusion (transposition)
- Design: Iterate a round function
- Two different types:
  - Feistel network (e.g. DES)
  - Substitution-permutation network (e.g. AES)



One Feistel round



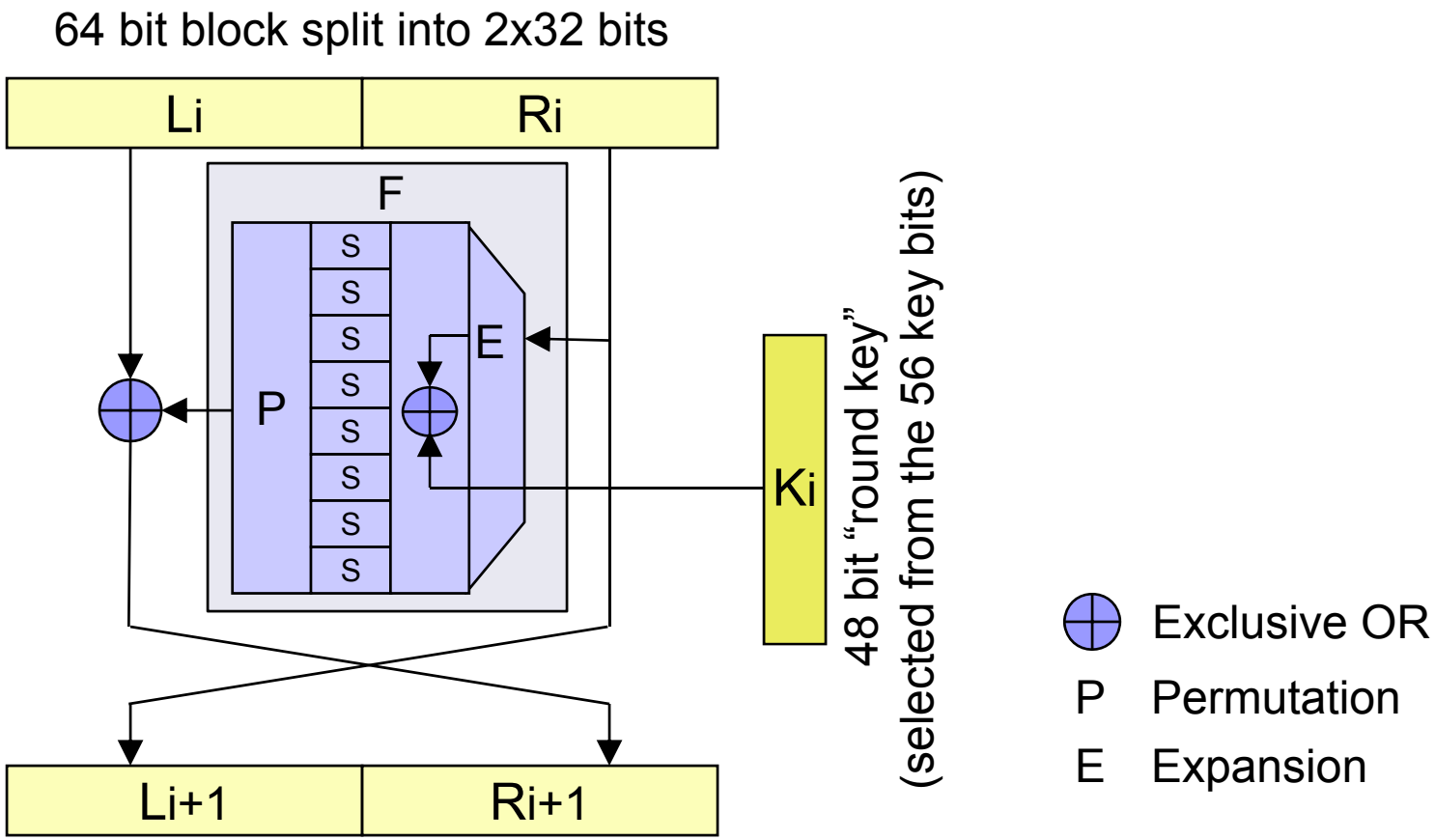
DES

- Data Encryption Standard
  - published by NIST as FIPS PUB 46 in 1977
- Based on Lucifer by IBM
- NSA changed the design
  - Fear of weaknesses
- Used extensively by banks
  - E.g. ATM
- With whitening in Win2K encrypted FS
- Becoming less common (move towards AES)

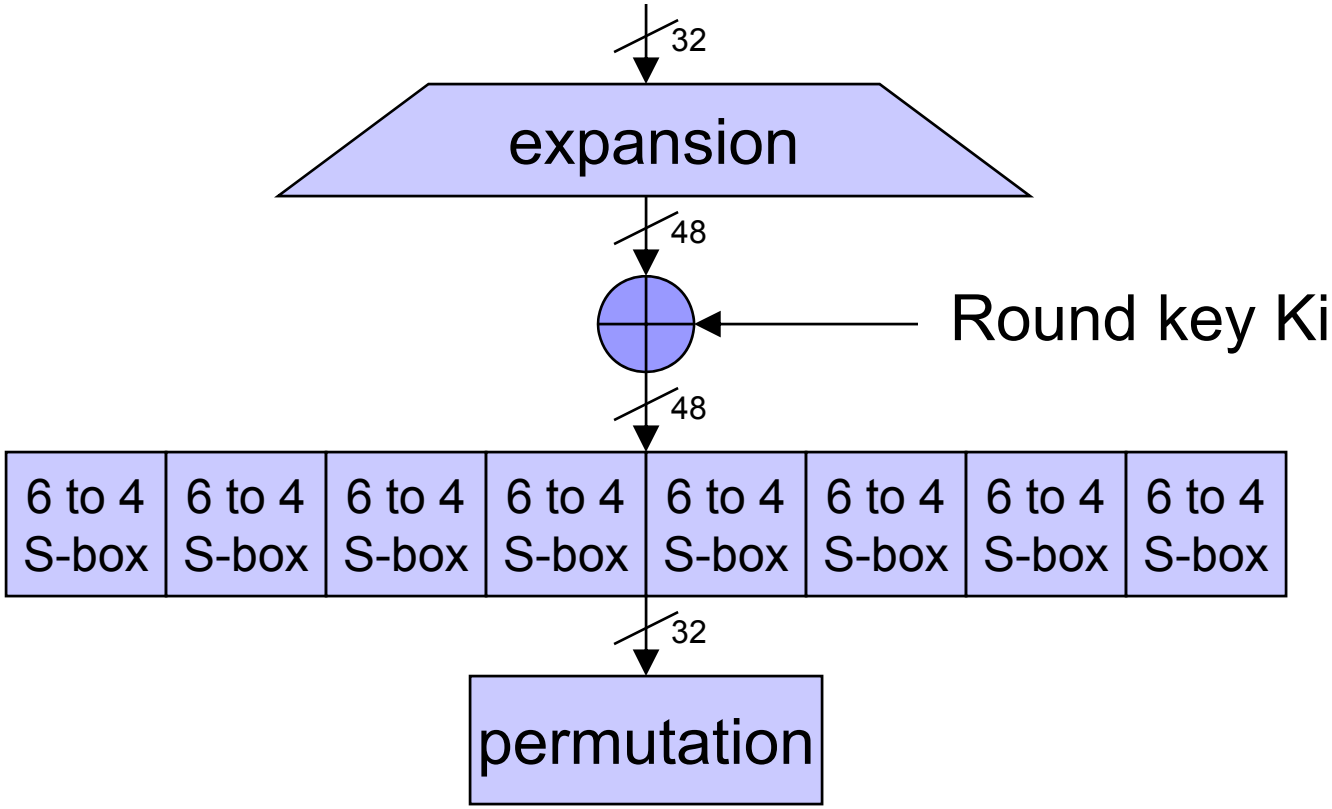
DES properties

- Block size 64 bit
- Key size 64 bit
  - 56 bit real key data
  - Remaining 8 bits are parity bits
- 16 rounds Feistel network
- Complement property:
  - $E(k,x^c) = E(kc,x)^c$

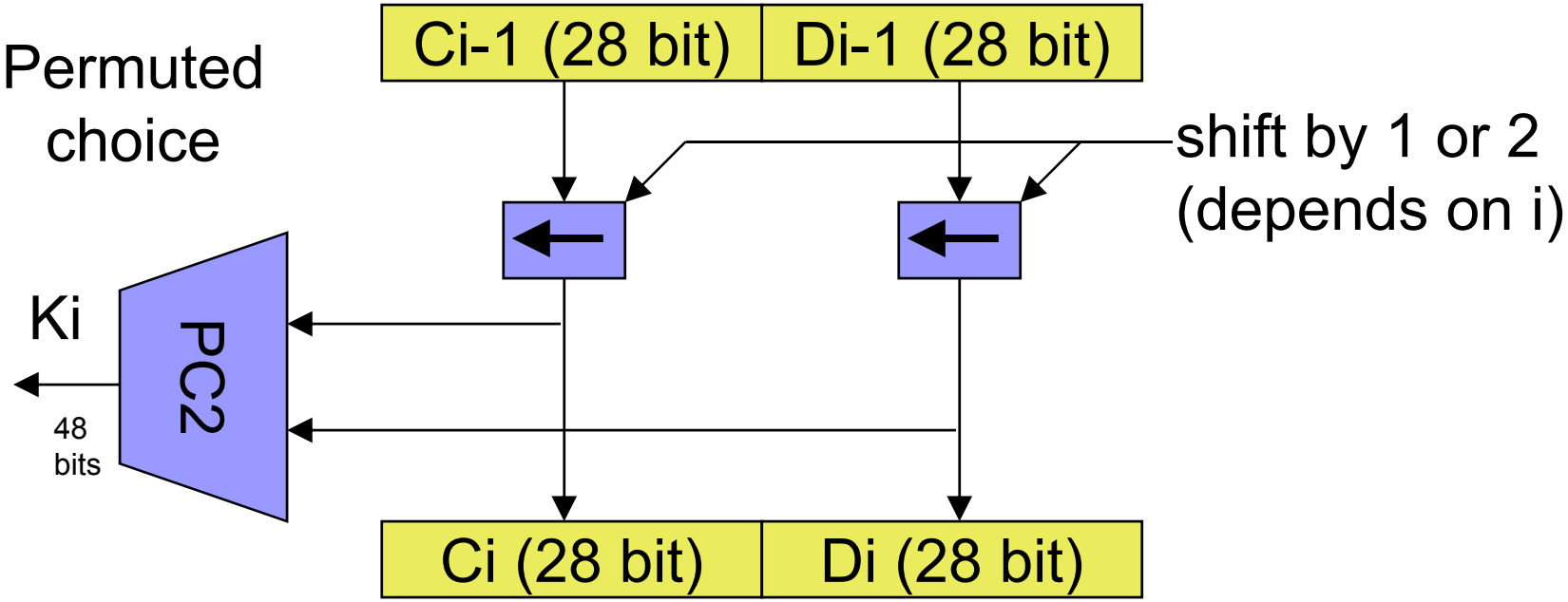
One Feistel round



DES Round function (F)



Key-schedule

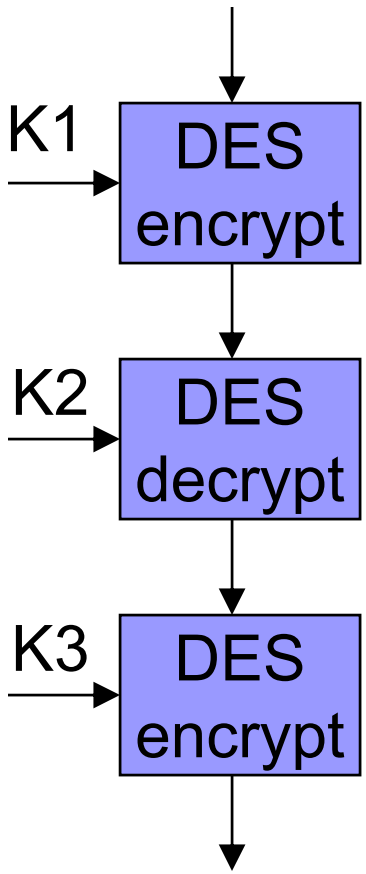


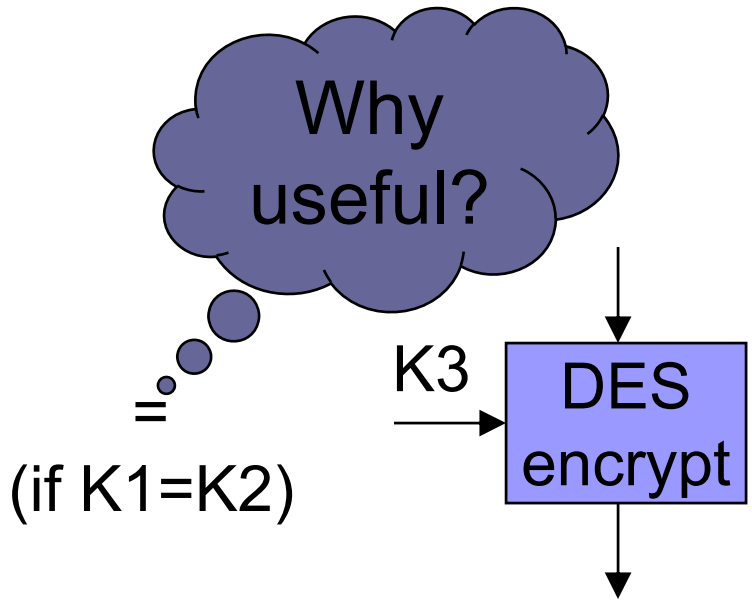
DES: discussion

- Extensively studied
  - No severe weaknesses found
- However, 56 bit key too short
  - 3DES
  - AES as new standard

3DES

(ANSI X9.17, ISO 8732 standard)





RSA

- By Rivest, Shamir and Adleman in 1978
- First “public” public key system
- Most popular
- Patent expired September 2000
- Large keys (1024 bits or more)

RSA preliminaries

- Euler Totient Function φ
- $\varphi(n) = \# \{ i \mid i < n, i \text{ relatively prime with } n \}$
- $\varphi(p * q) = (p - 1)(q - 1)$ for p, q prime
- $a^{\varphi(n)} \bmod n = 1$
  - If a,n relatively prime
  - For $n=p*q$ also without a,n relatively prime
- Inverse modulo n `easy' to find.

RSA Key generation

- Pick two large primes p,q and set n = p * q
  - p =/= q
- Pick e,d such that
  - ed = 1 mod φ(n)
  - i.e. ed = 1 mod (p-1)(q-1)
- Destroy p,q
- Public key: (e, n)
- Private key: (d, n)

RSA Encryption, Decryption

- Encrypt P: $C = P^e \bmod n$
- Decrypt C: $P = C^d \bmod n$

Public key: (e, n)
Private key: (d, n)

- Why it works:

$$
\begin{aligned}
C^d \bmod n &= (P^e \bmod n)^d \bmod n \\
&= P^{ed} \bmod n \\
[ed = 1 \bmod \varphi(n)] &= P \times P^{\varphi(n)*k} \bmod n \\
[P^{\varphi(n)} \bmod n = 1] &= P
\end{aligned}
$$

RSA Key generation Example

- Choose p,q: p=7 and q=17
- Gives n=119 and $\varphi(n) = 6 * 16 = 96$

- Pick e relatively prime with 96, e.g. e=5
- Compute d with ed = 1 mod 96.
  - Result: d=77
  - Verify: 77 * 5 = 385 = 4*96 + 1
- Public key: (5,96) Private key (77,96)

RSA Encrypt/Decrypt Example

- Public key: (5,96)
- Encrypting P=19:
  - $19^5 = 2476099 = 20807 * 119 + 66$
  - Ciphertext is 66

- Private key (77,96)
- Decrypting 66
  - $66^{77} = 19 \bmod 96$

RSA: Setup and Security

- Given p,q, it is easy to find e,d such that

$$ed = 1 \bmod \phi(n) = 1 \bmod (p-1)(q-1)$$

- Without p,q
  - computing φ(n) is hard
  - finding d given e as hard as finding p,q
  - finding private key as hard as factoring

RSA Special properties

- E(m*m')=E(m)*E(m') mod n
  - Add redundancy to sign messages
- Blinding with a random r

$$E(mr^e) = (mr^e)^d \bmod n = m^d r \bmod n$$

  - Hide message from signer
    - Application: Anonymous money
- RSA can be used to sign or encrypt
  - signing = decrypting
  - use separate key pairs

RSA: choices & requirements

- e = 3, e = 7 or e = 65537 (= 2^{16} +1)
  - Salt append random bits (e.g. 64) to plaintext
  - Otherwise attacks exist to find private key and
  - encryption small m less than n; easily recovered
- All users must pick distinct modulus n
  - Any e,d with ed = 1 mod φ(n) allows factoring n
  - Easy to compute any d' from e'

RSA: choices & requirements (2)

- d roughly the same size as n
  - Otherwise it can be found efficiently from e and n
- factoring n must be hard
  - p,q sufficiently big
  - p,q roughly the same size
  - still p-q sufficiently large

RSA vs DES performance

- RSA ~ 1000 slower in hardware
- RSA ~ 100 time slower in software
- Gets worse with longer keys

- How long a key is needed?
  - Estimate effort needed by attacker

Hypotheses

- 56 bit DES key was strong enough in 1982
  - Breaking it requires 500,000 Mips Years
    - 1 Mips Year = 20 hours on 450Mhz Pentium II
- Computing per $ doubles every 18 months
  - Variant of Moore's law
  - Every 10 years, 100x computing power per $
- Budget of organisations doubles every 10 years
- Algorithmic improvement
  - Computation required halves every 18 months

Overview

Year	DES	RSA	DSA	EC	Mips years
1982	56	417	102	…	$5x10^5$
2002	72	1028	127	139	$2x10^{10}$
2012	80	1464	141	165	$4x10^{12}$
2022	87	1995	154	193	$8x10^{14}$

Cryptography Basics

Part 4: (Provable) Security

Definitions of security

- Information theoretical (aka unconditional)
  - Possible in public key setting ?
  - Public key known: Anyone can encrypt.
  - Try all possible private keys
    - (Recall why is this not possible for one time pad...)
- Computational
  - Breaking cipher is mathematically hard problem

What is a hard problem (1)

- Algorithm for short or long instances
  - Running time depends on length of instance
  - E.g.: Sorting 10 numbers takes less time than sorting 10.000 numbers
- For some problems minimum number of steps for any algorithm known
  - Sorting n numbers takes at least n log n steps
  - Very hard to prove

What is a hard problem (2)

- `Hard' problem: requires at least an exponential number of steps to solve
  - I.e. nr of steps more than any polynomial.
  - in size of problem (= *security parameter*)

- No hard problems in NP known

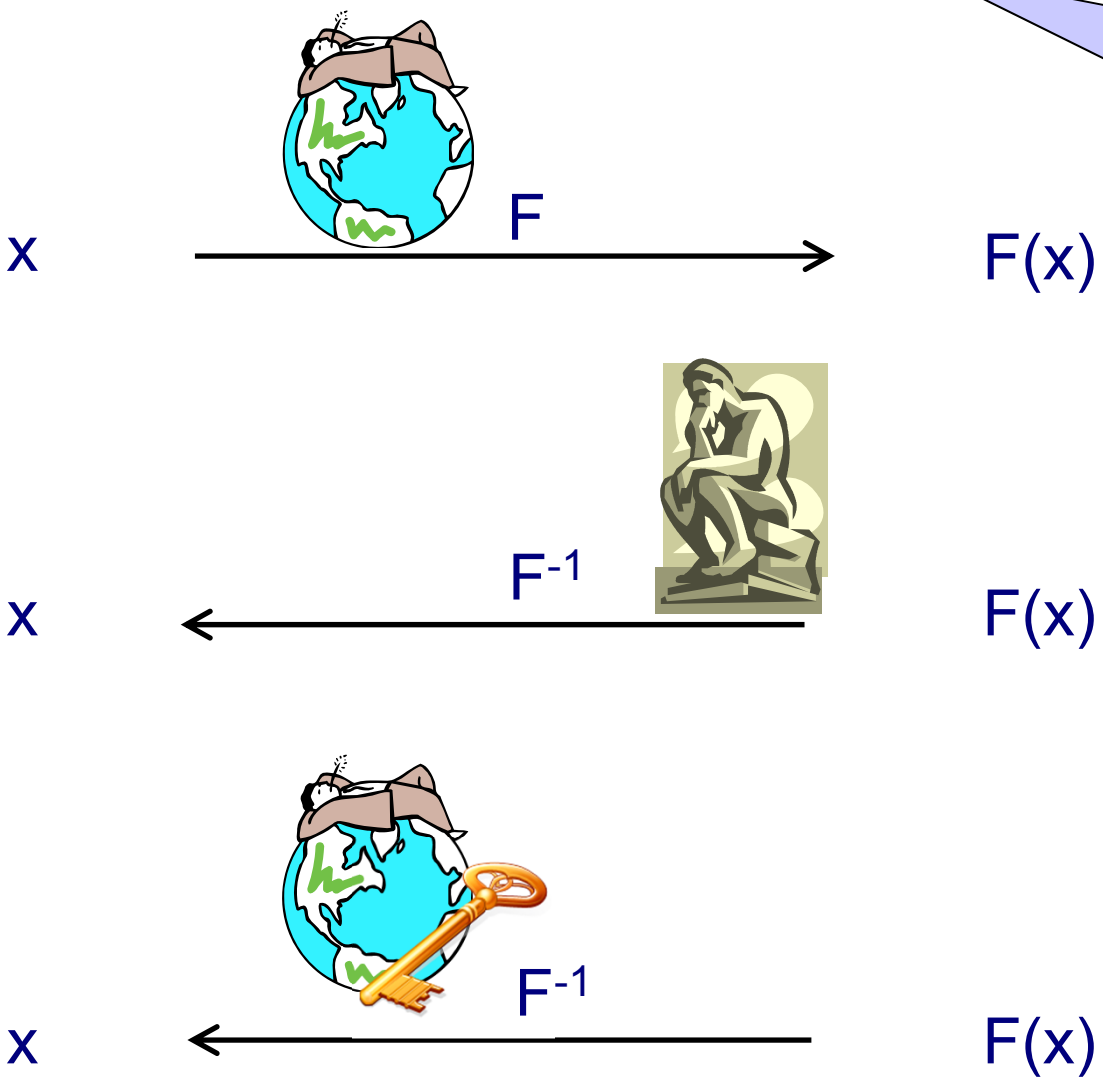
- *Known* solutions take exponential time:
  - Factoring a product of two primes
  - Computing the discrete logarithm

P vs NP

- P
  - Solving takes polynomial time
- NP
  - Solution can be checked in polynomial time
    - But finding solution may take exponential time
- NP contains P
- It is unknown whether P = NP

Trapdoor function: F

Example:
Multiply 2 primes
Factoring hard
...unless 1 known



Can we prove crypto is `good'?

- Argued that it is hard to find private key
- Is this sufficient to show message `safe' ?

- Show breaking solves `hard' problem
- Security game expresses exact property
- Probabilistic; always tiny chance to guess
  - Attacker advantage; better than pure guess

Security Game 1 (IND-CPA)

Indistinguishable under chosen-plaintext attack

1. Opponent picks two plain texts
2. We randomly pick one, encrypt it & give cipher text to opponent
3. Opponent guesses which text was encrypted

Opponent advantage: | **P**(correct guess) - 1/2 |

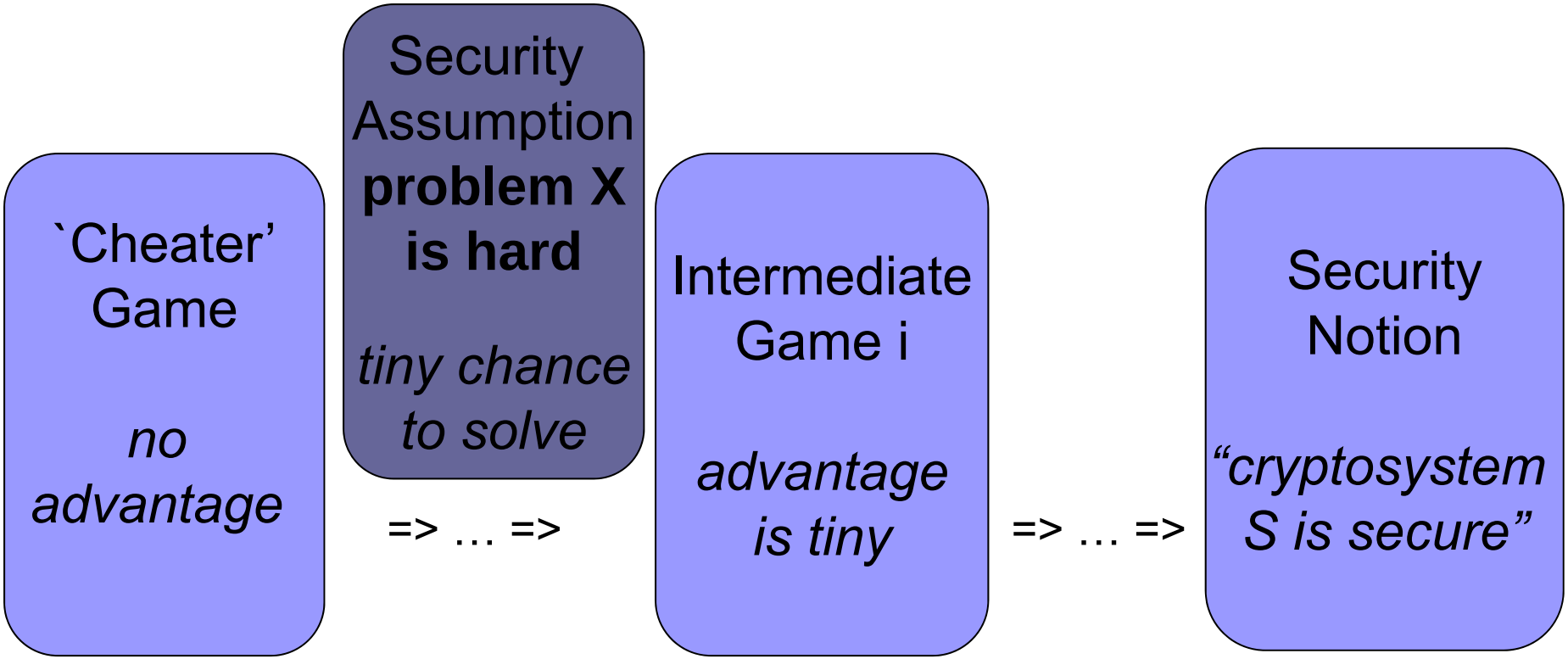
Good cipher: opponent advantage small

Example: ElGamal

- Multiplicative Group $Zq = \{1 \ldots q-1\}$
  - Multiply, divide, exponentiation easy, log hard
- Key creation: sample x from Zq,
  - x is the private key, g^x is the public key
- Encryption: sample **y** from Zq (salt)
  - $enc(m,g^x) = (c,k) = (m*g^{xy}, g^y)$
- Decryption:
  - $dec((c,k), x) = c / k^x$

$$
\begin{aligned}
&dec(enc(m,g^x),x) \\
&\quad = dec((m*g^{xy}, g^y),x) \\
&\quad = m*g^{xy} / g^{yx} \\
&\quad = m
\end{aligned}
$$

Game stepping (reduction)



Tasks:

1. Show security of basic game
2. Show correctness of implications (games steppings)

`Cheater' Game for ElGamal

- Opponent picks two plain texts
- We randomly pick one and encrypt it.
- Opponent gets g^z for random z
- Opponent guesses which text encrypted.

- Opponent advantage:

  | **P**(correct guess) - 1/2 |
- Information opponent independent of choice
  - no opponent advantage possible

Security Assumption
(the `hard' problem)

- Decisional Diffie Hellman (DDH):

  "no effective attacker can distinguish between
  (g^x, g^y, g^z) and (g^x, g^y, g^{xy})"

- Exists ε such that, for any attacker, any q:

  Random x,y,z in Zq;
  guess1 = Attacker(g^x, g^y, g^z);
  guess2 = Attacker(g^x, g^y, g^xy)
  | **P**(guess1) - **P**(guess2) | < ε(q)

Transformation

- Use property of * to conclude

Random x,y,z in Zq

guess-game1 = Attacker(g^x,g^y,m * g^z);

guess-game2 = Attacker(g^x,g^y,m * g^xy)

| **P**(guess-game1) - **P**(guess-game2) | < ε(q)

- If can tell difference m * a and m * b then can tell difference a, b

Difference Cheat - Security Game

- In real game attacker gets
  - Public key: g^x
  - From the cipher text: g^y and $m * g^{xy}$
- In the basic game the opponent
  - Public key: g^x
  - From the cipher text: g^y and $m * g^z$
- If the attacker can distinguish then also between

  (g^x,g^y,g^z) and (g^x,g^y,g^xy)
  - Play security game with this input.
  - For first will be basic game, for second security game

Security Game 2 (IND-CCA2)

- Opponent has **Enc**, **Dec** oracle
- Opponent picks two plain texts
  - Can use **Enc/Dec** as wanted before choosing
- We randomly pick one and encrypt it
- Opponent gets cipher text (challenge)
- **Decryption** oracle not for challenge
- Opponent guesses which text encrypted

Math

- Computing modulo n
- Groups
  - Generator g (e.g. 2 in the multiplicative group Z^*_{13} below)
- (Possibly…) hard problems
  - Factoring an integer
  - Computing the discrete logarithm

i	0	1	2	3	4	5	6	7	8	9	10	11	12
2^i	1	2	4	8	3	6	12	11	9	5	10	7	1

Attacks

- Ciphertext only
- Known plaintext

passive

- Chosen plaintext
- Adaptive chosen plaintext
- Chosen ciphertext
- Adaptive chosen ciphertext

active

- Aim to increase `attacker advantage'.
  - More on this next week

Attacks

- Brute force key search
  - passwords
- Timing attack
- Differential cryptanalysis
- Birthday attack
  - Collision likely to happen when # inputs in order of square root of size outcome space.
- Side channel attack

Further study

- Exercises available (see server)

- ***Security Engineering, Chapter 5***
  - ***www.cl.cam.ac.uk/~rja14/book.html***
- ***Handbook of applied cryptography***
  - ***www.cacr.math.uwaterloo.ca/hac***
- ***Courses on cryptography***