

Lecture 1: Encryption

(First step towards security)

- 1.1: Definitions: Encryption/decryption/keys
- 1.2: Classical ciphers + illustration of attacks
- 1.3: Modern ciphers + recommended key length
- 1.4: Attacks on cryptosystem implementations
- 1.5: Kerckhoffs' principle vs security through obscurity
- 1.6: Interesting historical facts

How Important is Encryption?

Hadi Partovi, co-founder of Code.org:

“Encryption is at least as foundational as photosynthesis”



“We don’t teach biology or chemistry to kids because they’re going to become surgeons or chemists.

We teach them about photosynthesis and that water is H₂O, or how lightbulbs work, just to understand the world around us.

You don’t use any of it, but you **do on a day-to-day basis use**
public-key encryption”

Increasing Data Protection Need

The
Economist

Topics ▾

Current edition

More ▾

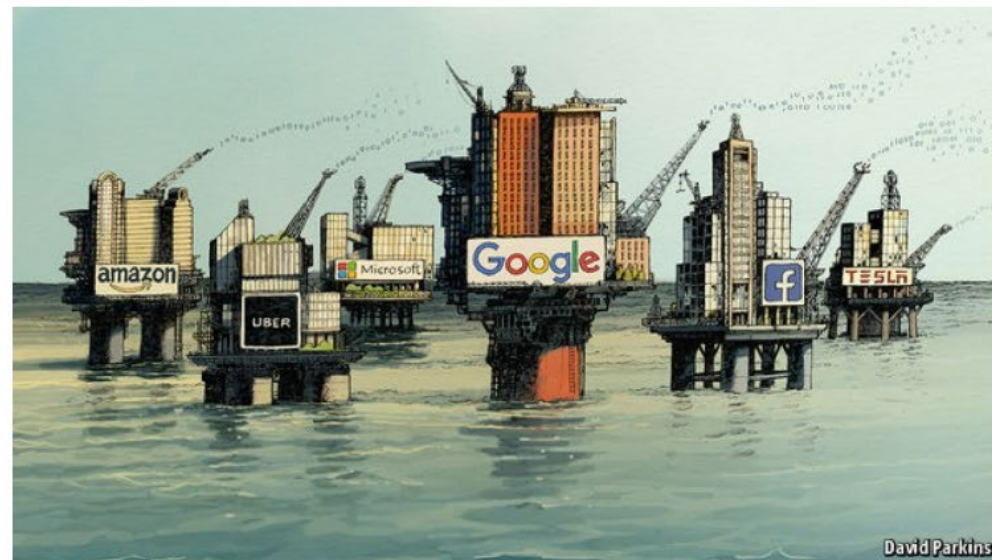
Subscribe

Regulating the internet giants

The world's most valuable resource is no longer oil, but data

The data economy demands a new approach to antitrust rules

Ref:
<https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data>



Print edition | Leaders >

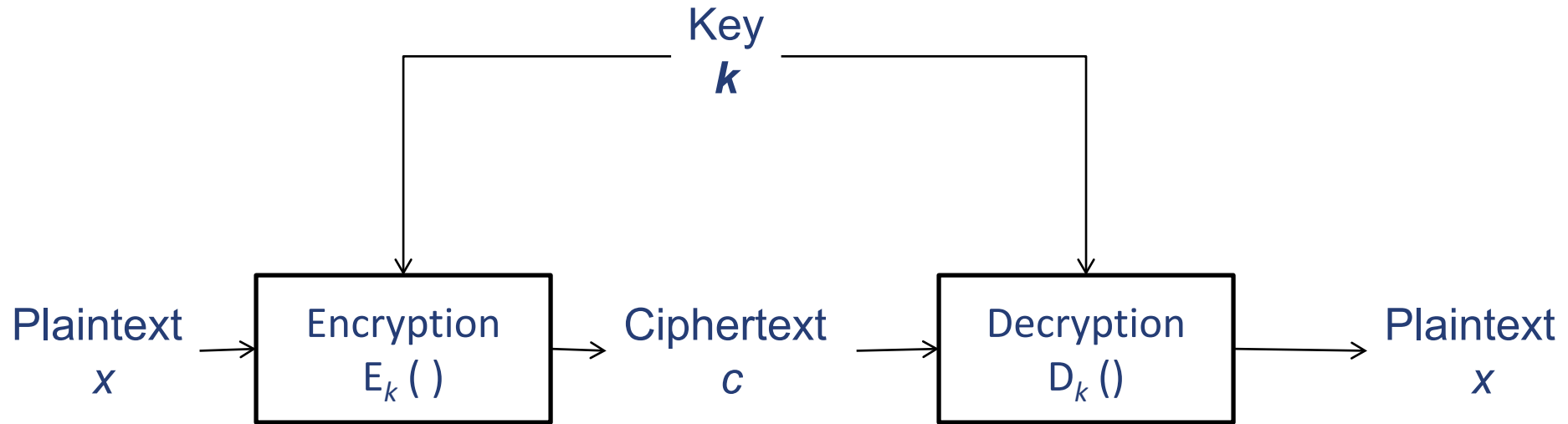
May 6th 2017



1.1 Definitions

Encryption

An **encryption scheme** (also known as **cipher**) consists of two algorithms:
encryption and decryption



Two requirements:

Correctness: For any plaintext x and key k , $D_k(E_k(x)) = x$

Security: Given the ciphertext, it is “difficult” to derive useful information of the key k and the plaintext x . The ciphertext should resemble a sequence of random bytes.

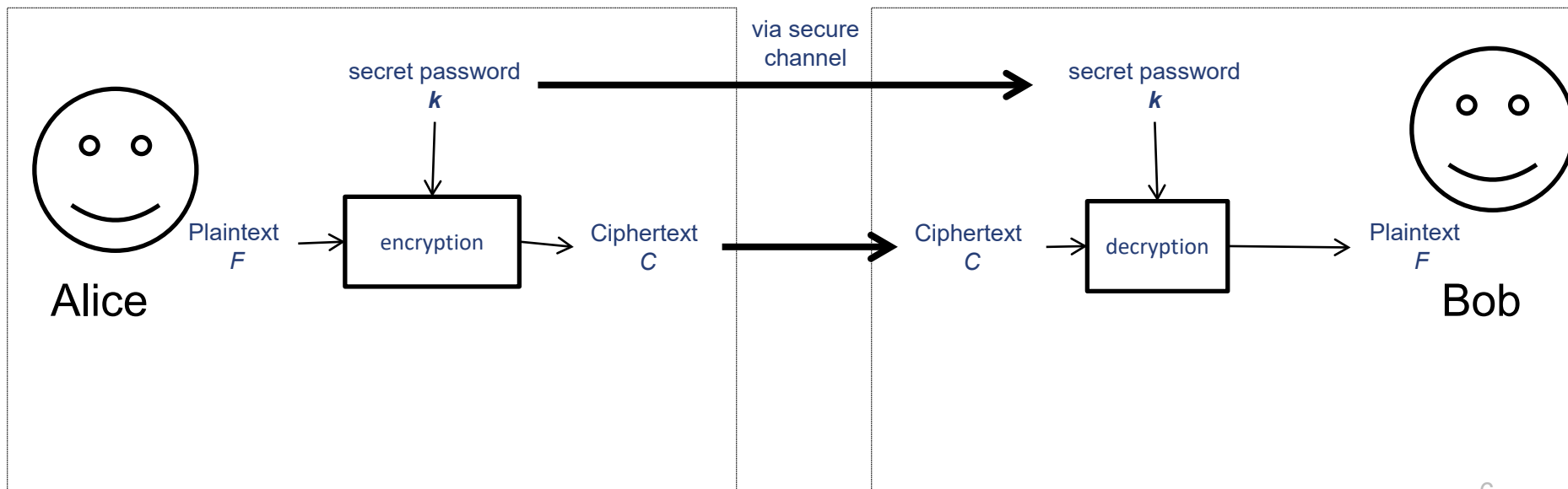
(There are many refined formulations of security requirements, e.g. *semantic security*.)

In this module, we will not go into details.

There’s also a performance requirement: the encryption and decryption processes can be efficiently computed.)

A Simple Application Scenario

- Alice had a large file F (say an Excel file containing information of her bank accounts and financial transactions).
- She “encrypted” the file F using Winzip with a password “13j8d7wjnd”, and obtained the ciphertext C .
- Next, she called Bob to tell him the password, and subsequently sent the ciphertext to Bob via email attachment.
- Later, Bob received C , and decrypted the ciphertext with the password to recover the plaintext F .



A Simple Application Scenario

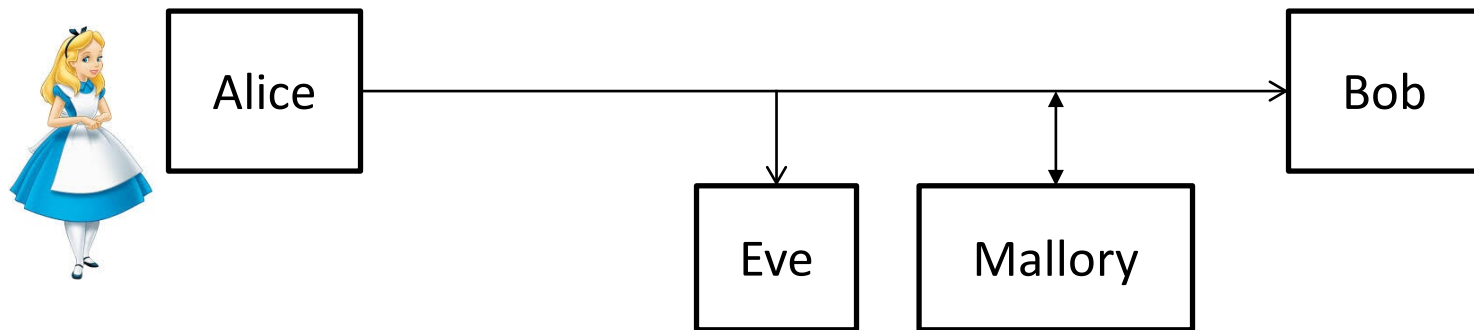
- Anyone who has obtained C , without knowing the password, is unable to get any information on F
- Although C indeed contains information of F , the information is “hidden”
- To someone who doesn’t know the secret, C is just a **sequence of random bits**
- *Remark:*
Winzip is **not** an encryption scheme.
It is an application that employs standard encryption schemes, such as AES.

Cryptography (vs Cryptology?)

- **Cryptography** is the study of techniques in securing communication in the presence of *adversaries* who have access to the communication
- Although cryptography is commonly associated with encryption, encryption is *just one* of the primitives in cryptography
- Others include cryptographic hash, digital signature, etc.
- *How about cryptology?*

Characters in Cryptography

- (Terminology) Common placeholders used in cryptography:
 - Alice: usually the originator of message
 - Bob: usually the recipient
 - Eve: eavesdropper, can only listen to sent messages
 - Mallory: malicious, can also modify sent messages



- See the interesting list of crypto characters in:
https://en.wikipedia.org/wiki/Alice_and_Bob
- Depending on context, Alice may not be a human:
she could be the machine that encrypts the message

In this module,
“read”: Part of the teaching materials. Read it.
“see”: Info that is good to know.
“optional”: Optional information.

1.2 Classical Ciphers

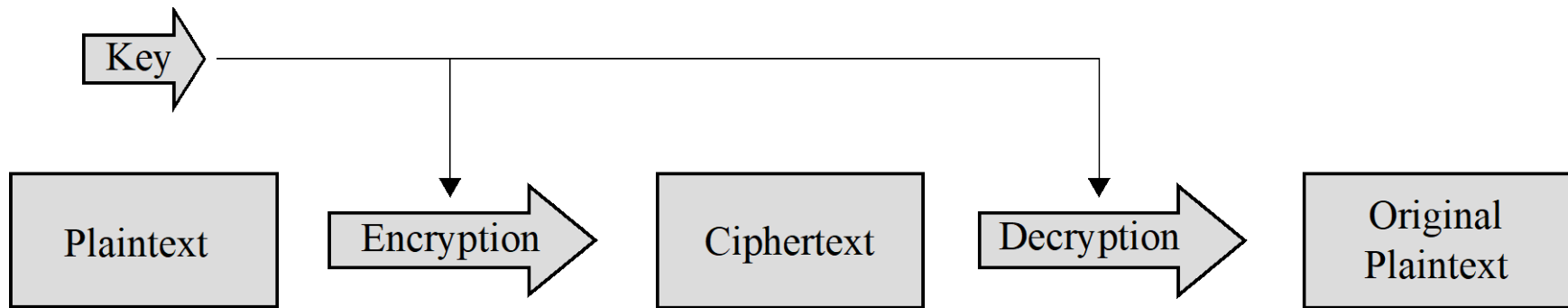
For illustration, we will look into a few classical ciphers.
Classical ciphers are **not** secure in the computer era.
(The exception: the “unbreakable” one-time-pad).

(See <http://ciphermachines.com/index>

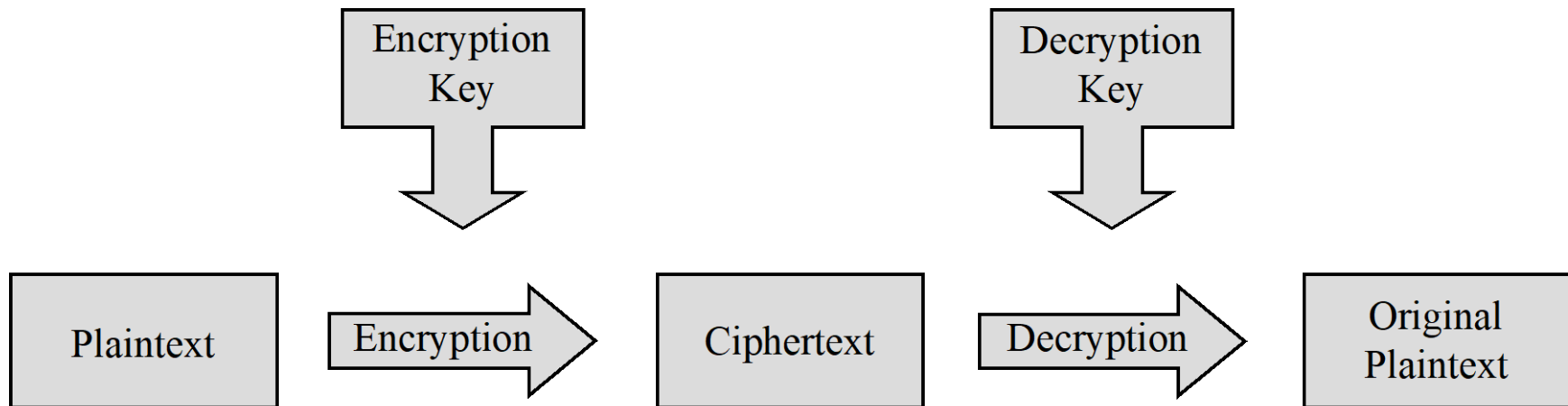
for a good listing of classical ciphers and cipher machines used during WWII.)

- 1.2.1. Substitution Cipher
- 1.2.2. Permutation Cipher
- 1.2.3. One Time Pad

Note: Symmetric vs. Asymmetric Cryptosystems



(a) Symmetric Cryptosystem



(b) Asymmetric Cryptosystem

From *Security in Computing, Fifth Edition*, by Charles P. Pfleeger, et al. (ISBN: 9780134085043). Copyright 2015 by Pearson Education, Inc. All rights reserved.

1.2.1 Substitution Cipher

Substitution Cipher

- **Plaintext** and **ciphertext**: a string over a set of symbols U

E.g.

Let $U=\{“a”, “b”, “c”, \dots, “z”, “_”\}$.

Example of plaintext: “hello_world”

- **Key**: a substitution table S ,
representing an 1-1 onto function from U to U

E.g.

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	_
g	v	w	b	n	e	f	h	d	a	t	l	u	c	q	m	z	i	r	s	j	x	o	y	k	_	p

$S(a) = g, S(b) = v, \dots$

The inverse of S :
 $S^{-1}(g)=a, S^{-1}(v) = b$

Substitution Cipher

Some terms:

- The ***key space***: the set of all possible keys
- The ***key space size*** or ***size of key space***: the total number of possible keys
- The ***key size*** or ***key length***: the number of bits required to represent a particular key
- For substitution cipher:
 - The key space?
 - The key space size: $27!$
 - The key size: at least $\log_2(27!) \approx 94$ bits

Substitution Cipher: Encryption/Decryption

Encryption: Given a plaintext of length n , which is a string $X = x_1 x_2 x_3 \dots x_n$, and the key S , output the ciphertext

$$E_S(X) = S(x_1) S(x_2) S(x_3) \dots S(x_n)$$

Example:

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	_
g	v	w	b	n	e	f	h	d	a	t	l	u	c	q	m	z	i	r	s	j	x	o	y	k	_	p

plaintext: h e l l o _ w o r l d

ciphertext: h n l l q p o q i l b

Decryption: Given a string of ciphertext of length n $C = c_1 c_2 c_3 \dots c_n$ and the key S , output the plaintext

$$D_S(C) = S^{-1}(c_1) S^{-1}(c_2) S^{-1}(c_3) \dots S^{-1}(c_n)$$

Attacks on a Cipher

- In general, the attacker's goal is:
 - To find the key: if the key can be found, then the plaintext can be obtained
(How about the converse?)
 - To obtain some information of the plaintext
- Before commencing an attack, the attacker needs access to some information, such as:
 - A large number of ciphertexts that are all encrypted using the same key → **"ciphertext only"**; or
 - Pairs of ciphertext and the corresponding plaintext → **"known plaintext"**

Exhaustive Search (aka Brute-Force Search) Attack

- A *simple (brute-force) attack* is to exhaustively search the keys:
 - i.e. examine all possible keys one by one
- For most schemes, exhaustive search is *infeasible*
- Surprisingly, for some modern ciphers e.g. DES, it is feasible to break it using exhaustive search
- More sophisticated attacks exploit the properties of the encryption scheme to speedup the process

Exhaustive Search (aka Brute-Force Search) Attack

- Consider the substitution cipher (with table size of 27)
- Suppose the attacker somehow has access to a ciphertext C and a plaintext X ,
how difficult for him to find the key using exhaustive search?
- Let \mathcal{S} be the set of all possible substitution table
- Given X, C :
 1. For each S in \mathcal{S}
 2. Compute $X' = D_S(C)$;
 3. If $(X' == X)$ then break;
 4. end-for
 5. Display (“The key is ”, S);

Exhaustive Search (aka Brute-Force Search) Attack

- The running time depend on the size of the key space S
- Since a key can be represented by a sequence of 27 symbols, the size is key space is $27!$

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	_
g	v	w	b	n	e	f	h	d	a	t	l	u	c	q	m	z	i	r	s	j	x	o	y	k	_	p

- Eventually, exhaustive search will find the key
- However, in the worst case, the exhaustive search needs to carry out $27! \approx 2^{94}$ loops, and on average $\approx 2^{93}$ loops. This is infeasible using current computing power (see Tutorial 1).
- *Can we attack substitution cipher more efficiently?*

Attack on Substitution Cipher: Known-Plaintext Attack

- ***“Known plaintext attack scenario”:***
when an adversary has access to pairs of ciphertexts and their corresponding plaintexts, and try to guess the key
- The attacker doesn’t need to carry out exhaustive search.
Given a plaintext and ciphertext, e.g.

plaintext: h e l l o _ w o r l d
ciphertext: h n l l q p o q i l b

The attacker can figure out the entries in the key

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	_
			b	n			h				l			q			i					o				p

- For a sufficiently long ciphertext, the full table can be determined

Attack on Substitution Cipher: Known-Plaintext Attack

- If the adversary can successfully derive the key, we say that the scheme is:

“insecure under known plaintext attack” or

“broken under known plaintext attack”

- Hence, substitution cipher is *not secure* under known plaintext attack!

Some Remarks on Known-Plaintext Attack

- To carry out a known-plaintext attack, the attacker needs to obtain at least a pair of ciphertext and its corresponding plaintext
- *Is this requirement reasonable??*
- In many cases, the attacker *doesn't need* to know the full plaintext: only *the first few bytes* of the plaintext are sufficient
- These first few bytes of the plaintext can sometimes be guessed:
 - Email data: certain words in its header are fixed, such as “From”, “Subject”, etc.
 - Many network protocols have fixed headers, or only a few choices in their first few bytes of data packets
 - During WWII, cryptologists exploited commonly-used words like “weather” and “nothing to report” as the known plaintext to guess the secret keys
- (Optional: Read more about the “Enigma Machine”.)

Attack on Substitution Cipher: Ciphertext-Only Attack

- Suppose the attackers have access to ciphertext only (i.e. without the corresponding plaintext), and knows that the plaintext are English sentences.
- Can he successfully find the key using exhaustive search?
- *Yes!*
- Let \mathcal{S} be the set of all possible substitution table
Given C :
 1. For each S in \mathcal{S}
 2. Compute $X = D_S(C)$;
 3. If X contains words in the English dictionary, then break;
 4. end-for
 5. Display (“The key is ”, S);

Attack on Substitution Cipher: Ciphertext-Only Attack

- Eventually, the exhaustive search will find the key
- However, the attack is also infeasible due to the large key space size
- Note: There is a very small probability that the above algorithm finds a wrong key.
Yet, for a sufficiently long ciphertext, say 50 characters long, the probability that a wrong key will give a meaningful English sentence is very low (treated as “*negligible*”).
- Is there an effective ciphertext-only attack technique on substitution cipher?
- *Yes!*

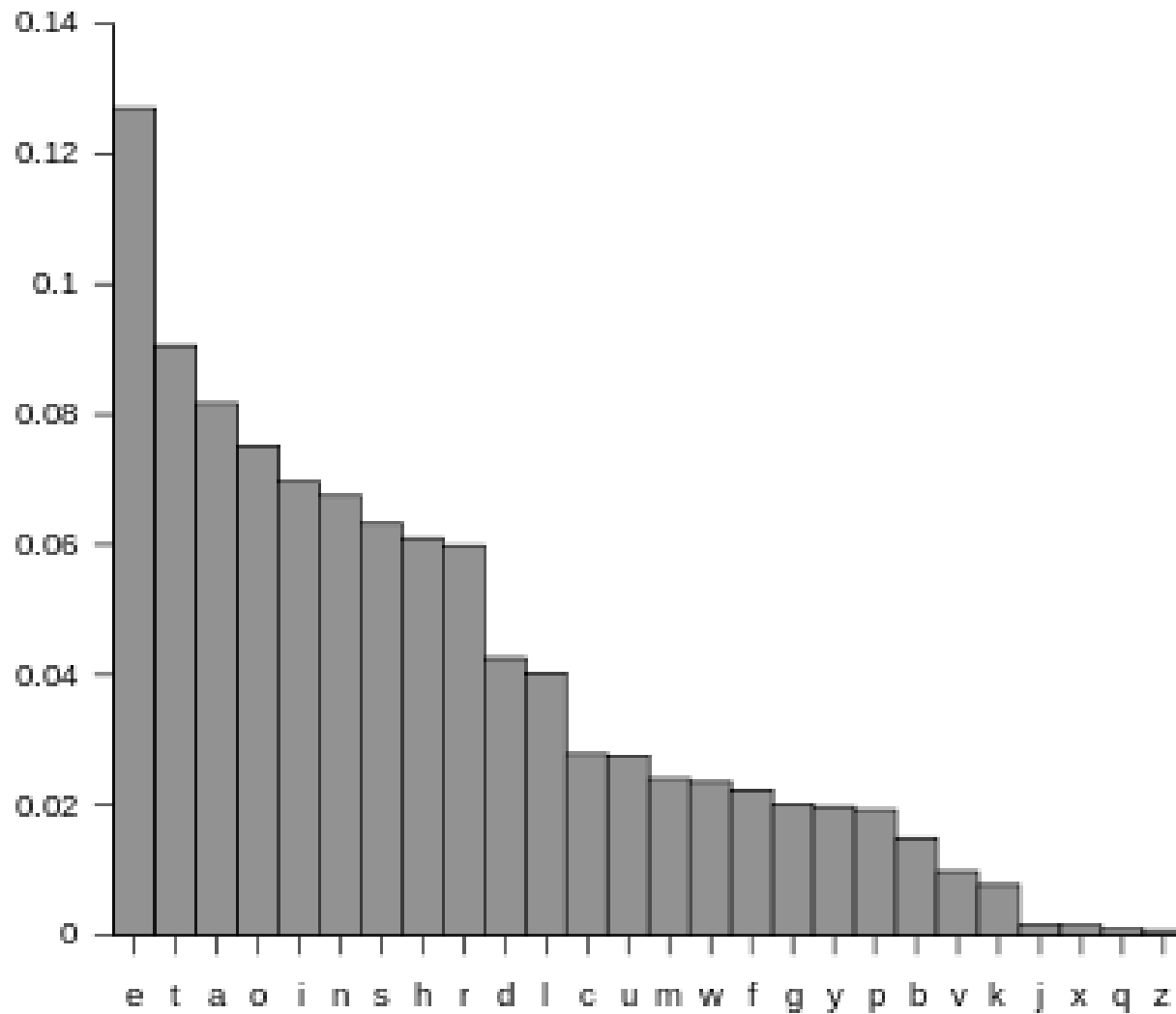
Frequency Analysis

- Substitution cipher is vulnerable to *frequency analysis*
- Note that in the hello_world example below, symbol “o” appears 2 times in the plaintext, whereas the corresponding “q” also appears 2 times in the ciphertext

plaintext:	h	e	l	l	o	_	w	o	r	l	d
ciphertext:	h	n	l	l	q	p	o	q	i	l	b

- Suppose the plaintexts are English sentences.
The **letter frequency distribution** in English text is ***not*** uniform, for e.g. “e” is more commonly used than “z”.
- How can an adversary apply frequency analysis and break substitution cipher?

Letter Frequency Distribution in English Text



From http://en.wikipedia.org/wiki/Letter_frequency

Frequency Analysis on Substitution Cipher

- If adversary is aware that the plaintexts are English sentences, given a sufficiently long ciphertext (say around 50 characters), then an adversary may be able to guess the plaintext by:
 - Mapping the *frequently-occurring letters in the ciphertext* to the *frequently-occurring letters of English*.
- Frequency analysis can be successfully carried out!
- Hence, substitution cipher is ***not secure under ciphertext-only attack*** either, when the plaintexts are English sentences
- In fact, the attack is effective on any language

LumiNUS Forum Challenge

- “**Breaking substitution cipher**” challenge in LumiNUS forum
- You’ll be given a ciphertext
- Do break the substitution cipher by finding the correct corresponding plaintext
- The **first person** who can post the correct plaintext will get **3 (three) extra marks** for assignment!
- The challenge will be posted this evening at **~8pm**

Ongoing NUS Bug Bounty Challenge!

- <https://nusit.nus.edu.sg/its/announcements/nus-bug-bounty-challenge/>



NUS
National University
of Singapore

Information
Technology

NUS IT presents

BUG BOUNTY CHALLENGE

STAND A CHANCE TO WIN

- CASH PRIZES up to USD1500 per bug • EXTRA MARKS for eligible course modules • SPONSORSHIP for security conference
- A place in the HALL OF FAME

AUG 14th–26th
2019

Exclusively for NUS Students
Sign up now at https://hackerone.com/users/sign_up
Registration is open to students with NUS email account

NEW TO HACKING? Learn the ropes at <https://www.hacker101.com>

For more information, please contact cceits@nus.edu.sg | or visit <https://nusit.nus.edu.sg/its/>

T & Cs apply

Ongoing NUS Bug Bounty Challenge!

- I'll give **10 (ten) extra marks** for CS2107 assignment to a Bounty Winner: no double claim in AY19/20

OTHER REWARDS

Modules	Description
CS2107	Intro to Information Security
CS3235	Computer Security
CS4238	Computer Security Practice
CS4239	Software security
CS5321	Network Security
CS5331	Web Security

IVLE's Cryptanalysis Challenge on Substitution Cipher

- The substitution table used:

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	_
p	x	c	o	l	g	y	b	h	_	q	d	w	m	s	e	f	k	v	a	j	z	r	i	t	n	u

- Mapping from plaintext to ciphertext:

i_can_break_this_substitution_cipher_to_show_that_
hucpmuxklpquabhvuvjxvahaajahsmucheb1kuasuvbsruabpau

it_is_insecure_and_i_can_get_three_extra_marks
hauhvuhmv1cjk1upmouhucpmuylauabk11uliakpuwppkqv

Some Useful Heuristics

- What's the most-frequently occurring character?
 - e, _ (space)?
- Spaces must break up the sentence reasonably well
- Single-letter words:
 - a: an indefinite article
 - i: the first (singular) person
- Digraphs:
 - **to, it, is**, do, on, in, at, of, or, an, he, ...
- Trigraphs:
 - **can, and, get**, not, for, the, has, one, man, men, ...
- You can also utilize some available online tools!

Quiz

- Suppose a substitution cipher works over a set of symbols U with $|U| = 50$
- What is:
 - Its key space size?
 - (The lower bound of) its key size/length?

Caesar cipher

- A type of substitution cipher
- Each letter in the plaintext is “shifted”
a certain number of places down the alphabet
- Example: with a shift of **1**, A would be replaced by B, B would become C,, Z would be replaced by ?
- How about its:
 - Key space size?
 - Key size (lower bound)?

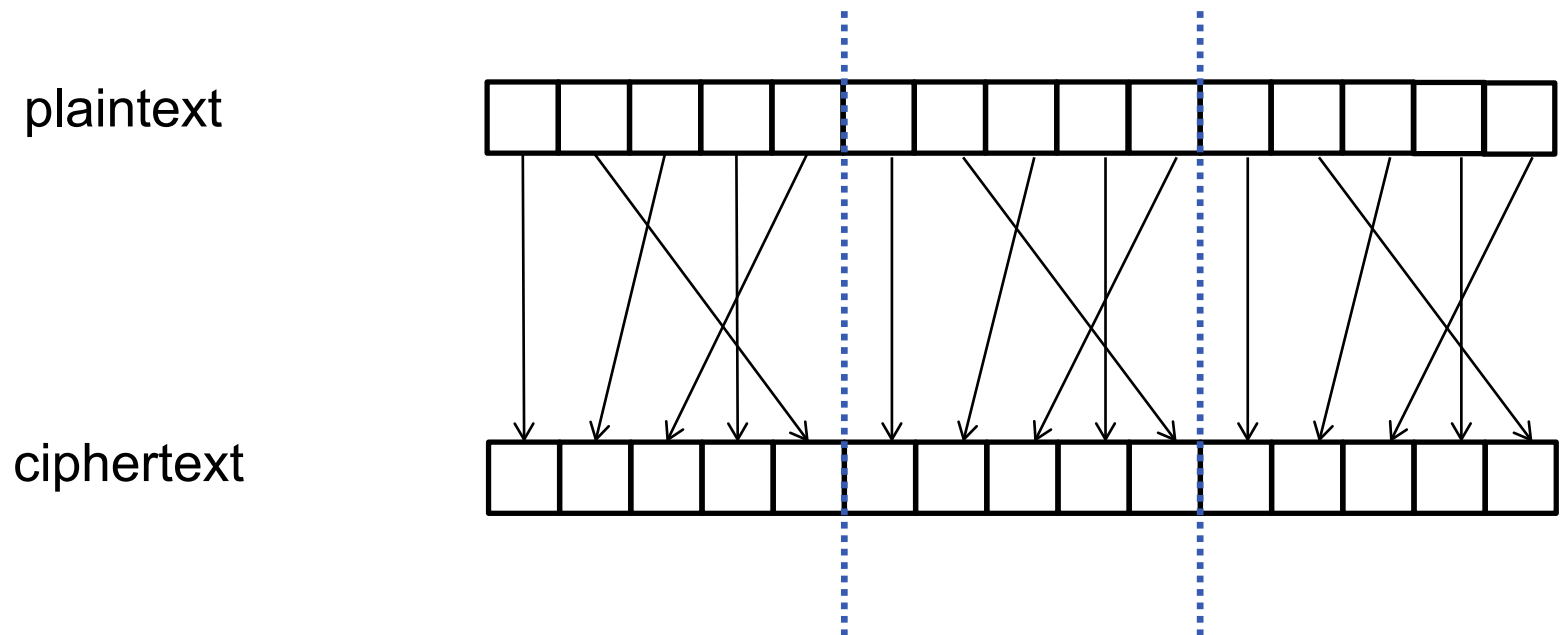
1.2.2 Permutation Cipher

Permutation Cipher

- Also known as **transposition** cipher
- It first groups the plaintext into blocks of **t** characters, and then applied a secret “permutation” to the characters in **each** block by shuffling the characters
- The key is the secret “permutation”:
an 1-1 onto function e from $\{1, 2, \dots, t\}$ to $\{1, 2, \dots, t\}$
- We can write the permutation **p** as a sequence
$$\mathbf{p} = (p_1, p_2, p_3, \dots, p_t),$$
which shifts the character at position/index i within the block to the position p_i
- The block size **t** could be part of the key:
 t is also kept secret

Permutation Cipher

- Example:
Given the plaintext and the key $t=5$, $p=(1,5,2,4,3)$:



Cryptanalysis (Known-Plaintext Attack)

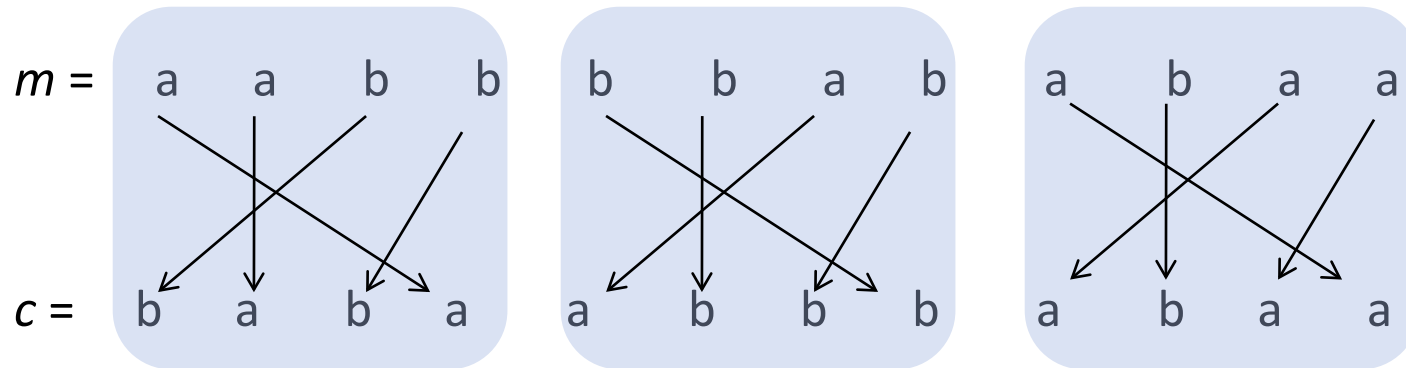
- Permutation cipher fails miserably under known-plaintext attack.
- Given a plaintext and a ciphertext, it is very easy to determine the key.
- Example:

$m =$ a a b b b b a b a b a a

$c =$ b a b a a b b b a b a a

- Question:
 - In the above, what is the block size t ?
 - What is the permutation?

Solution



- Permutation cipher can be easily broken under ciphertext-only attack if the plaintext is an English text

1.2.3 One Time Pad

One-Time-Pad

- Encryption:

Given an n -bit plaintext: $x_1x_2\dots x_n$ and n -bit key: $k_1k_2\dots k_n$,
output the ciphertext:

$$C = (x_1 \text{ xor } k_1) (x_2 \text{ xor } k_2) (x_3 \text{ xor } k_3) \dots (x_n \text{ xor } k_n)$$

- Decryption:

Given an n -bit ciphertext: $c_1c_2\dots c_n$ and n -bit key: $k_1k_2\dots k_n$,
output the plaintext:

$$X = (c_1 \text{ xor } k_1) (c_2 \text{ xor } k_2) (c_3 \text{ xor } k_3) \dots (c_n \text{ xor } k_n)$$

- In short:

Encryption: plaintext \oplus key \rightarrow ciphertext

Decryption: ciphertext \oplus key \rightarrow plaintext

Xor (Exclusive Or) Operation

- Xor operation: $A \oplus B = (A+B) \bmod 2$

- Xor table:

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

- Some interesting properties:
 - Commutative: $A \oplus B = B \oplus A$
 - Associative: $A \oplus (B \oplus C) = (A \oplus B) \oplus C$
 - Identity element: $A \oplus 0 = A$
 - Self-inverse: $A \oplus A = 0$

One-Time-Pad Example

decryption	PlainText	0	0	1	0	1	1	0	encryption
	Key	1	1	0	0	1	1	1	
	Ciphertext	1	1	1	0	0	0	1	

- Why does its decryption process works?
- For any x, k :
 $(x \oplus k) \oplus k$
 $= x \oplus (k \oplus k)$ [by associativity]
 $= (x \oplus 0)$ [by self-inverse]
 $= x$ [by identity element]

Security of One-Time-Pad

- From a pair of ciphertext and plaintext, yes, the attacker can derive the key.
- However, such a key is useless, since it *will not* be used any more!
- Note that it is not clear how to apply exhaustive search on one-time pad
- In fact, It can be shown that one-time-pad leaks **no information** of the plaintext, even if the attacker has an arbitrary running time (unlimited computing power)
- Hence, it is sometime called “*unbreakable*” (provided that a good “random” key is used)

Some Remarks

- Yes, one-time-pad can be shown to be “unbreakable”
- However, the length of the key is the same as plaintext’s, which is useless in many applications
- Nevertheless, it is practical in some scenarios

(See <http://ciphermachines.com/otp>)



<http://ciphermachines.com/otp>

- Yet, also check “*The Venona Story*”, where one-time-pads fails

(Optional: https://www.nsa.gov/about/files/cryptologic_heritage/publications/coldwar/venona_story.pdf)



1.3 Modern Ciphers

Modern ciphers generally refer to schemes that **use computer** to encrypt/decrypt.

Examples are: RC4, DES, A5, AES, RSA

Modern Ciphers

- Designs of modern ciphers take into considerations of known-plaintext-attack, frequency analysis, and other known attacks
- Examples: DES (Data Encryption Standard, 1977)
 - RC4 (Rivest's Cipher 4, 1987)
 - A5/1 (used in GSM, 1987)
 - AES (Advanced Encryption Standard, 2001)
- They are supposed to be “secure”, so that any successful attack does not perform noticeably better than exhaustive search

(**Optional:** Nevertheless, RC4 is broken in some adoptions, A5/1 is vulnerable, and DES's key length is too short. Wiki pages on RC4, A5/1 and DES give quite good descriptions. AES is believed to be secure, and classified as “Type 1” by NSA
https://en.wikipedia.org/wiki/NSA_cryptography#Type_1_Product.)

Exhaustive Search and Key Length (See Work factor [PF2.3page91])

- If the key length is 32 bits, there are 2^{32} possible keys. Hence, the exhaustive search has to loop for:
 - 2^{32} times in the worst case
 - 2^{31} times on average
- We can quantify the security of an encryption scheme by the *length of the key*
- Example:
 - Comparing a scheme *A* with 64-bit keys and a scheme *B* with 54-bit keys, then scheme *A* is more secure w.r.t. exhaustive search

Exhaustive Search and Key Length (See Work factor [PF2.3page91])

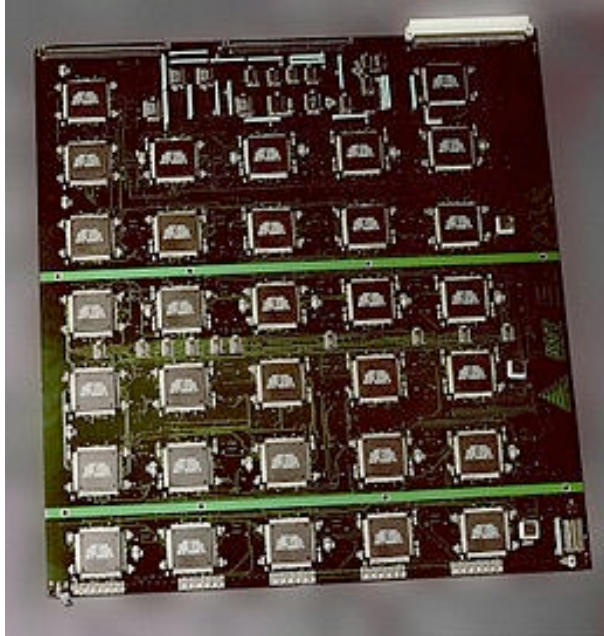
- ***Additional Note:*** Some schemes, e.g. RSA, have known attacks that are more efficient than exhaustively searching all the keys
- In those cases, we still want to quantify the security by the **equivalent exhaustive search**
- For example, in the best known attack on a 2048-bit RSA, roughly 2^{112} searches are required:
 - Hence, we treat its security as equivalent to 112 bits
 - So, we say that the 2048-bit RSA has key strength of 112 bits
- Question: How **many bits** are considered “secure”?
- Answer: See Tutorial 1
- Also *read* NIST Recommended key length for AES
<http://www.keylength.com/en/4/>

Exhaustive Search on DES

- Key length of DES is 56 bits
- While exhaustive search on 56 bits seemed infeasible in the 70s, very soon, it was possible using distributed computing or specialized chip
- *RSA Security* hosted a few challenges on DES:
 - **DES Challenge II-1:** *"The secret message is: Many hands make light work."*
(Found in 39 days using distributed computing, early 1998)
 - **DES Challenge II-2:** *"The secret message is: It's time for those 128-, 192-, and 256-bit keys."*
(Found in 56 hours using a specialized hardware, 1998)
- (Note: *RSA* is an encryption scheme, whereas *RSA Security* is a company)

Exhaustive Search on DES

- EFF's DES cracking machine ("*Deep Crack*"):



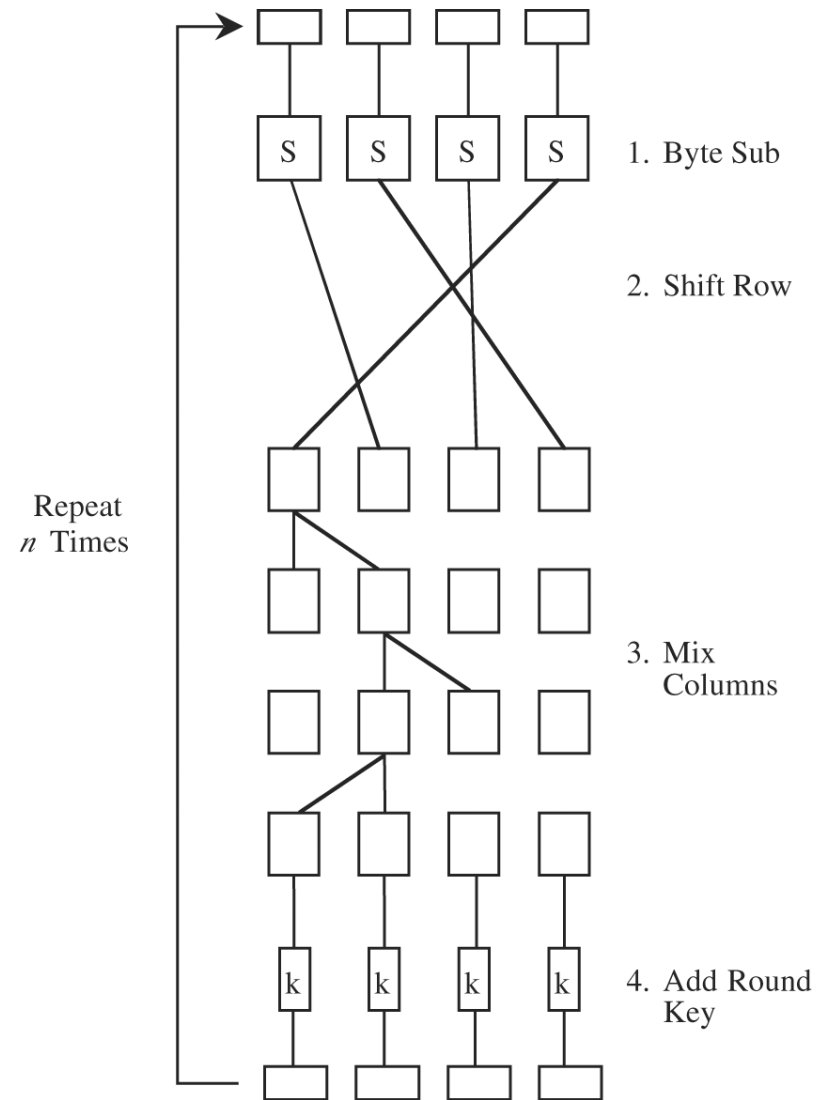
Optional:

https://en.wikipedia.org/wiki/EFF_DES_cracker

- A question is: why would an agency designed a scheme that could be broken in the near future?
- Many believed that it was intentional

AES: Advanced Encryption System

- Symmetric block cipher
- Developed in 1999 by independent Dutch cryptographers
- Still in common use



DES vs AES

	DES	AES
Date designed	1976	1999
Block size	64 bits	128 bits
Key length	56 bits (effective length); up to 112 bits with multiple keys	128, 192, 256 (and possibly more) bits
Operations	16 rounds	10, 12, 14 (depending on key length); can be increased
Encryption primitives	Substitution, permutation	Substitution, shift, bit mixing
Cryptographic primitives	Confusion, diffusion	Confusion, diffusion
Design	Open	Open
Design rationale	Closed	Open
Selection process	Secret	Secret, but open public comments and criticisms invited
Source	IBM, enhanced by NSA	Independent Dutch cryptographers

From *Security in Computing, Fifth Edition*, by Charles P. Pfleeger, et al. (ISBN: 9780134085043). Copyright 2015 by Pearson Education, Inc. All rights reserved.

Properties of Ciphers: Confusion and Diffusion

- Two properties of an encryption system: confusion and diffusion
- **Confusion**: an attacker should *not* be able to predict what will happen to the ciphertext when **one character** in the plaintext changes
- This means:
 - The input (i.e. plaintext and key pair) undergoes *complex* transformations during encryption
- A cipher with **good confusion**: it has a “*complex functional relationship*” between the plaintext/key pair and the ciphertext

Properties of Ciphers: Confusion and Diffusion

- ***Diffusion***: a change in the plaintext will affect *many parts* of the ciphertext
- This means:
 - Information from the plaintext is *spread over* the entire ciphertext
 - The transformations *depends equally* on all bits of the input
- A cipher with **good diffusion**:
it requires an attacker to access *much of* the ciphertext in order to infer the encryption algorithm
- Block cipher: high diffusion
- Stream cipher: low diffusion

1.3.2 Stream Cipher and IVs

Stream Cipher

- Stream cipher is “inspired” by one-time-pad
- Suppose the plaintext is 2^{20} bits, but the secret key is only 256 bits
- Stream cipher generates a 2^{20} -bit *sequence* from the key, and takes the generated sequence as the “*secret key*” in one-time-pad
- The generator has to be carefully designed, so that it gives ***(cryptographically-secure) pseudorandom sequence***

Stream Cipher

- Visually:

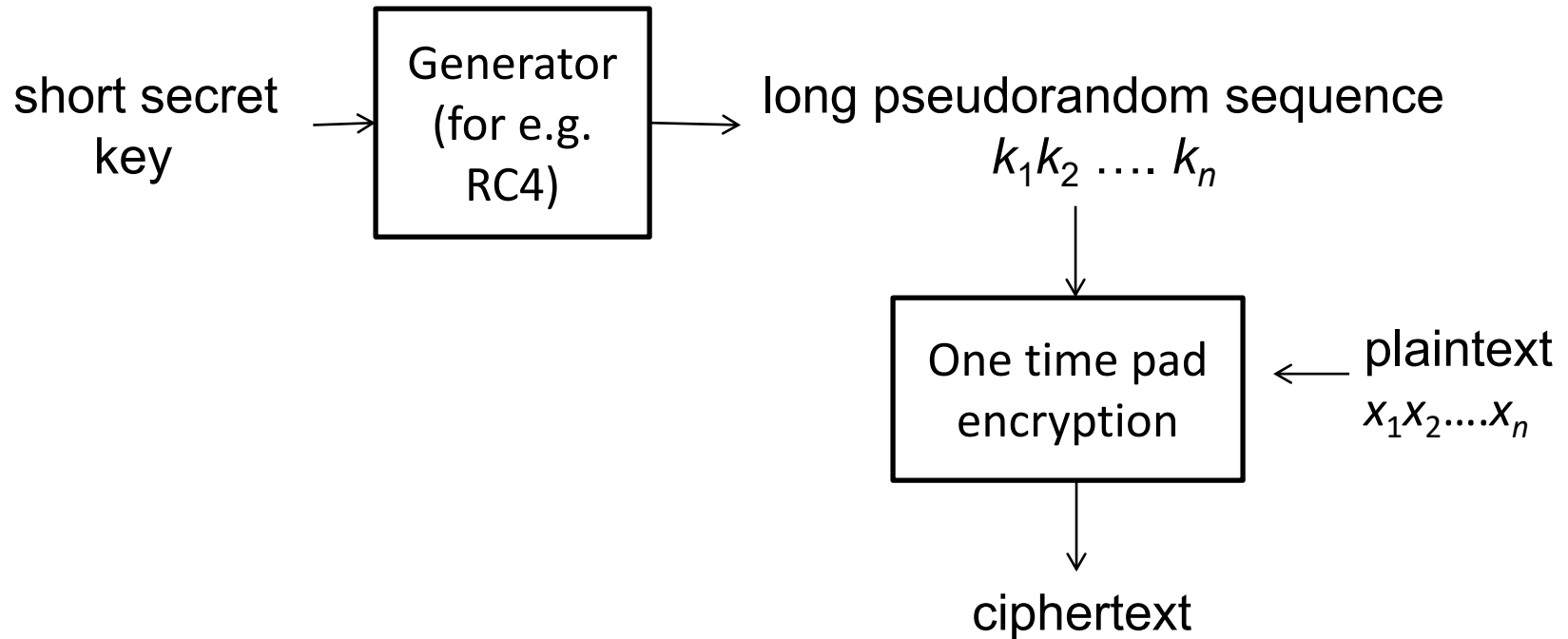
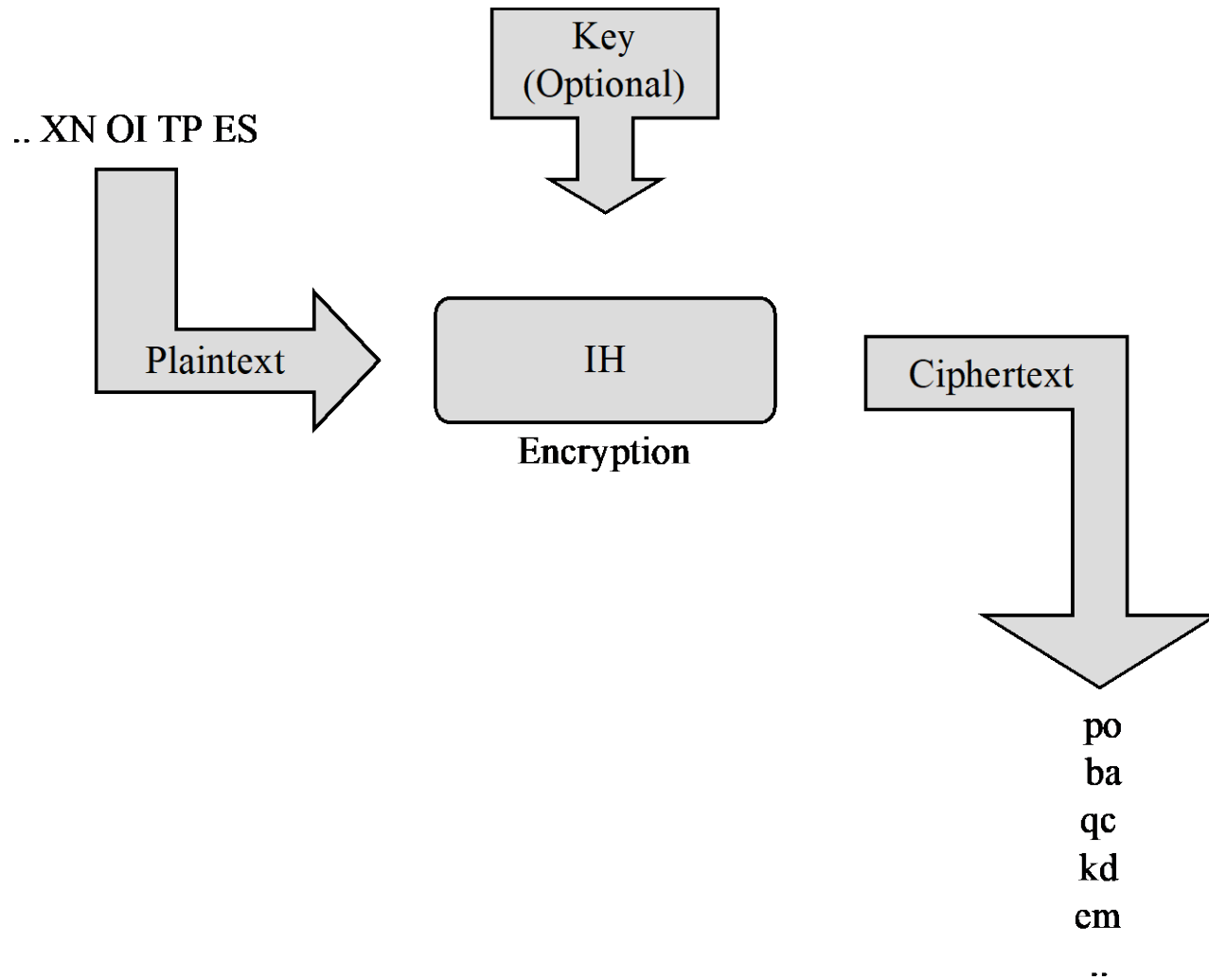
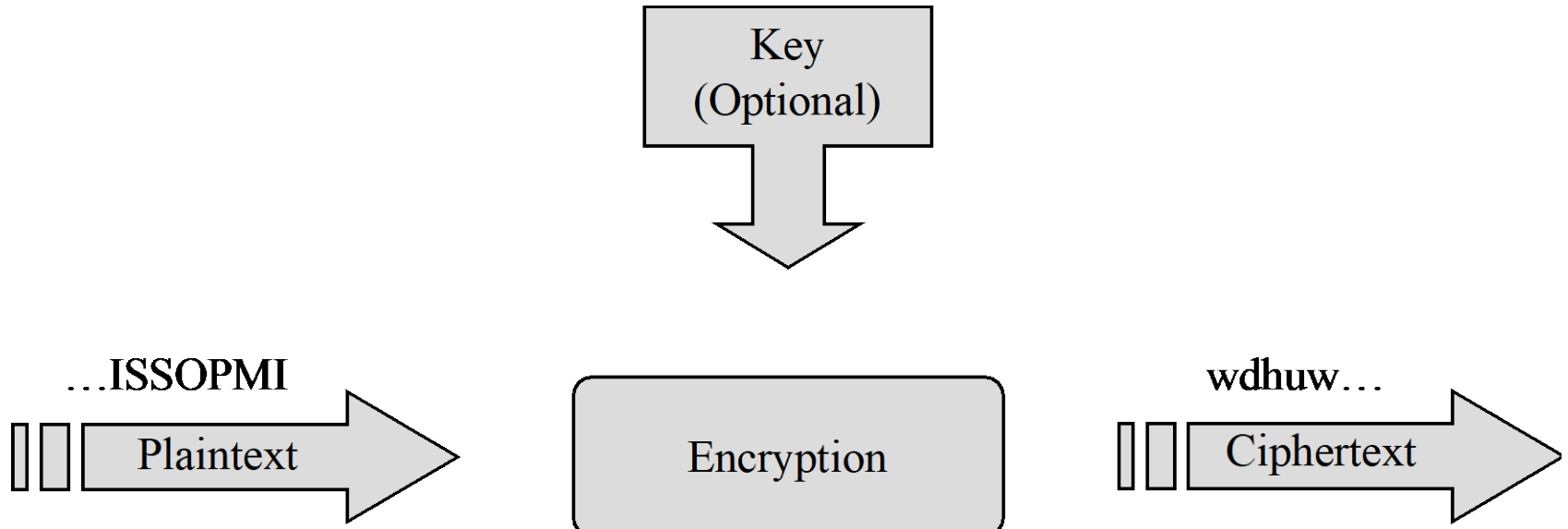


Illustration of a Block Cipher



From *Security in Computing, Fifth Edition*, by Charles P. Pfleeger, et al. (ISBN: 9780134085043). Copyright 2015 by Pearson Education, Inc. All rights reserved.

Illustration of a Stream Cipher



From *Security in Computing, Fifth Edition*, by Charles P. Pfleeger, et al. (ISBN: 9780134085043). Copyright 2015 by Pearson Education, Inc. All rights reserved.

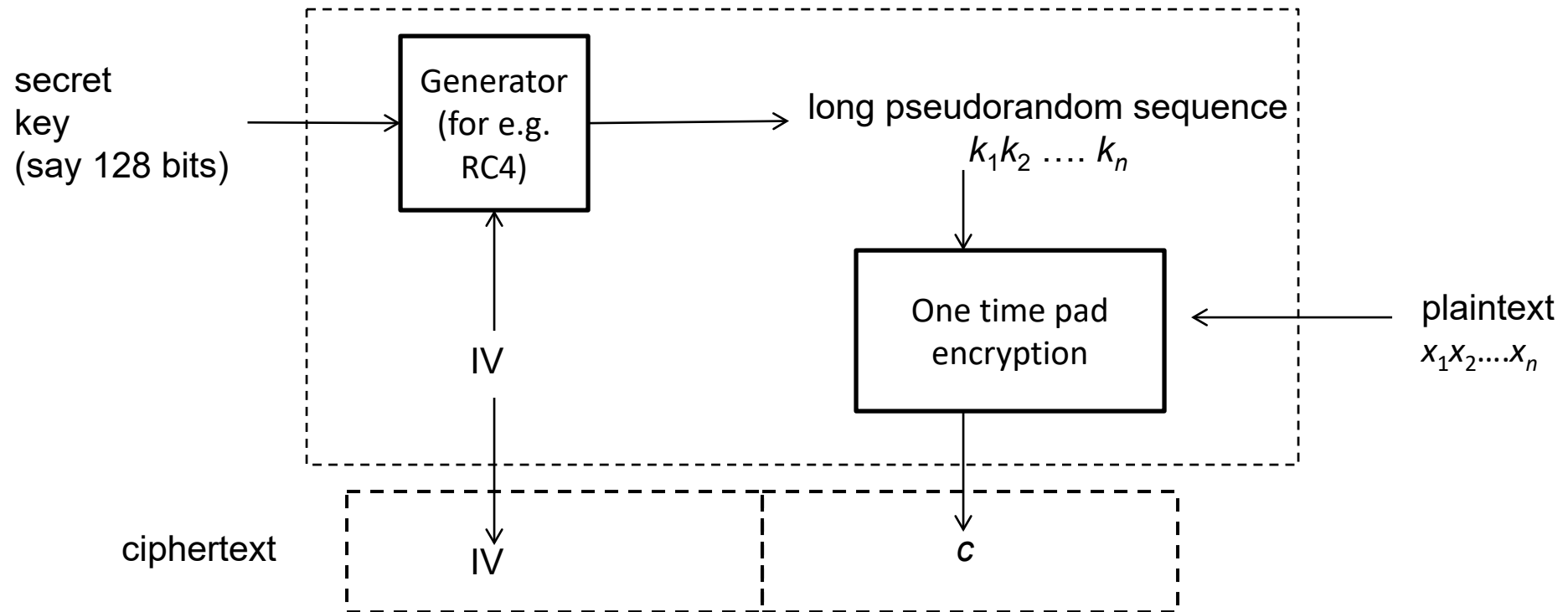
Stream vs Block Ciphers

	Stream	Block
Advantages	<ul style="list-style-type: none">• Speed of transformation• Low error propagation	<ul style="list-style-type: none">• High diffusion• Immunity to insertion of symbol
Disadvantages	<ul style="list-style-type: none">• Low diffusion• Susceptibility to malicious insertions and modifications	<ul style="list-style-type: none">• Slowness of encryption• Padding• Error propagation

Stream Cipher with Initial Value (IV)

- Stream cipher typically has an *Initial Value or Initialization Vector* (IV)
- The IV can be randomly chosen or from a counter
- A long pseudorandom sequence is generated from the secret key together with the IV
- The final ciphertext contains the IV, followed by the output of the one-time-pad encryption
- *Note:* In some documents, the term “ciphertext” does not include the IV. In any case, the IV **must** appear in clear, and every entity can see it.
- For decryption:
 - The IV is extracted from the ciphertext
 - From the IV and the key, the same pseudorandom sequence can be generated and thus obtain the plaintext

Stream Cipher with Initial Value (IV)



Important question: What if IV is omitted in the ciphertext?

Example

Encryption:

Given:

15-bit plaintext $X = 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0$

short key = $0\ 1\ 0\ 1$

Step 1: Randomly generate an IV, say: $IV = 0\ 0\ 0\ 1$

Step 2: From the short key and IV, generate a 15-bit sequence

$K = 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0$

Step 3: Output IV, followed by K xor X

$0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0$

Decryption:

Given the short key and the ciphertext with the IV

Step 1: Extract the IV from the ciphertext

Step 2: From the short key and IV, generate the long sequence K

Step 3: Perform xor to get the plaintext

Why IV?

- Suppose there isn't an IV
(or set the IV to be always a string of 0's)
- Consider the situation where the same key is used to encrypt two different plaintexts:

$$\mathbf{X} = x_1 x_2 x_3 x_4 x_5 \quad \text{and}$$

$$\mathbf{Y} = y_1 y_2 y_3 y_4 y_5$$

- Further, suppose that an attacker eavesdrops and obtains the two corresponding ciphertexts \mathbf{U}, \mathbf{V}

Why IV?

- The attacker can now compute:

$$\mathbf{U} \oplus \mathbf{V} = (\mathbf{X} \oplus \mathbf{K}) \oplus (\mathbf{Y} \oplus \mathbf{K})$$

- By associative and commutative properties of xor:

$$(\mathbf{X} \oplus \mathbf{Y}) \oplus (\mathbf{K} \oplus \mathbf{K}) = \mathbf{X} \oplus \mathbf{Y}$$

- So, from ciphertexts \mathbf{U} and \mathbf{V} , the attackers can obtain information about $\mathbf{X} \oplus \mathbf{Y}$, i.e. the following sequence:

$$(x_1 \oplus y_1) (x_2 \oplus y_2) (x_3 \oplus y_3) (x_4 \oplus y_4) (x_5 \oplus y_5)$$

- Question:** What's the big deal about revealing $\mathbf{X} \oplus \mathbf{Y}$?

What's the big deal about revealing $X \oplus Y$?

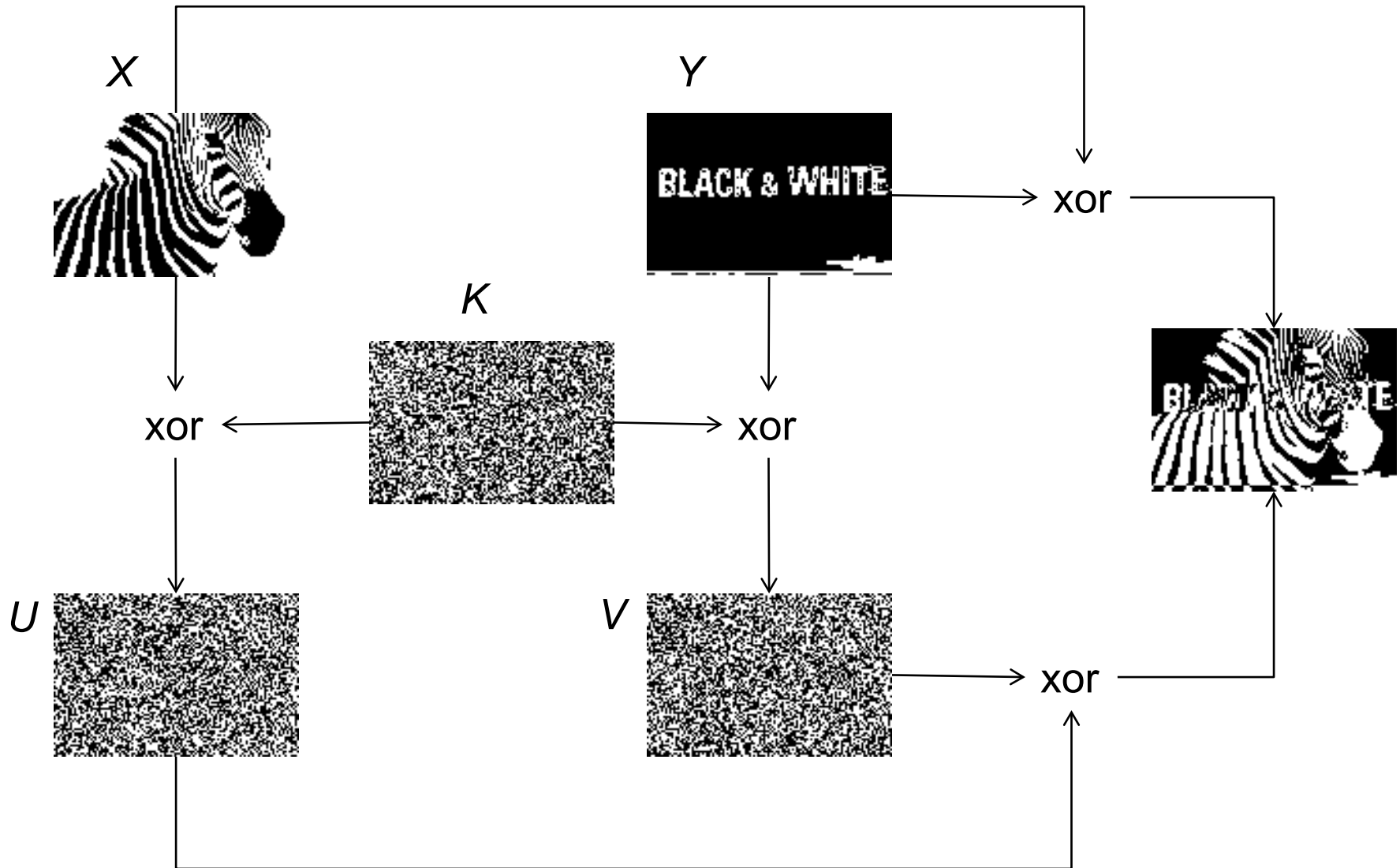
- Suppose X is a 80x120 image of an animal. Each pixel is either black (0) or white (1). The image can be represented as a (80x120)-bit sequence, where each bit corresponds to a pixel.
- Y is another 80x120 pixels image rendering two words, which is similarly represented as a sequence.

- Here is $X \oplus Y$:

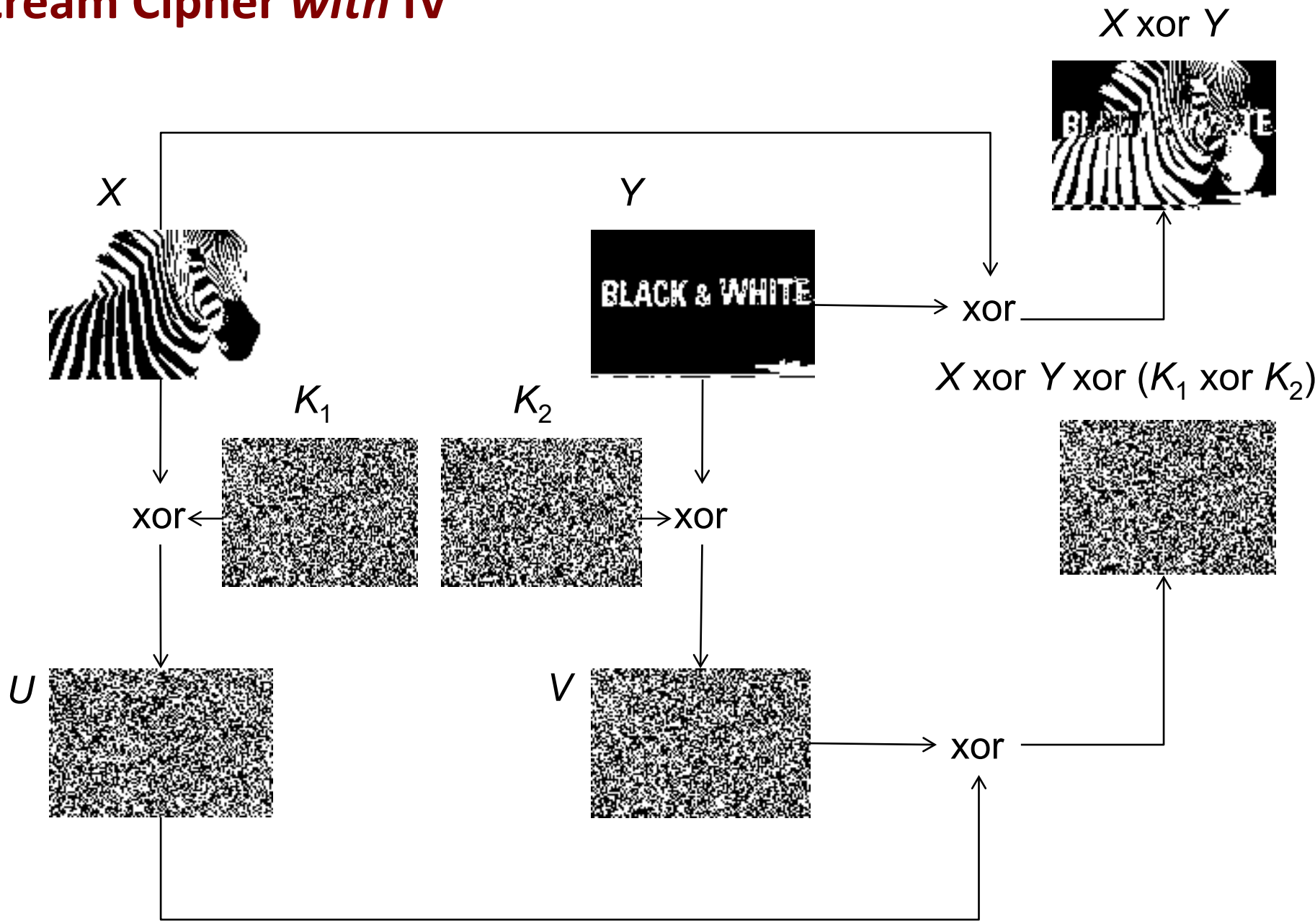


- So, what is X , Y ?

Stream Cipher *Without IV*



Stream Cipher with IV



Note: On this slide, the ciphertexts/keys are “artist’s impression”

Role of IV

- During 2 different processes of encryption of the same plaintext:
 - The IVs are likely to be different
 - The 2 pseudorandom sequences are thus likely to be different
 - The ciphertexts are also likely to be different
- Furthermore, the xor-ing of two ciphertexts would **not** cancel the effect of the pseudorandom sequences, and reveal information of the plaintexts!
- Note that IV is not only used in stream cipher, but also adopted in other ciphers

(**Note:** In many documents, the description of ciphers (such as AES, RSA) does not include the IV, and thus doesn't specify how the IV is to be chosen. Typically, the IV is mentioned during discussion of implementation. E.g. for AES, the IV is only included in the “**mode of operation**” of AES.)

1.4 Attacks on Cryptosystem Implementations

A secure encryption scheme can be vulnerable if it is **not implemented properly**.

This section gives some examples:

- 1.4.1 – Reusing IV and one-time-pad key in encryption

- 1.4.2 – Predictable secret-key generation

- 1.4.3 – Designing your own cipher

(**To be studied later:** Using encryption for the wrong purpose, e.g. using encryption scheme to ensure *message integrity*)

1.4.1 Reusing IV, One-Time-Pad Key

Mishandling of IV

- Some applications overlooked IV generation.
As a result, under some situations, the same IV is **reused**.
- E.g. To encrypt a file F , the IV is derived from *the filename*.
It is quite common to have files with the same filename.

(Read “**Schneier on Security, Microsoft RC4 Flaw**”:

https://www.schneier.com/blog/archives/2005/01/microsoft_rc4_f.html
<http://eprint.iacr.org/2005/007.pdf>)

- E.g. When using AES under the “CBC mode”, the IV has to be **unpredictable** to prevent a certain type of attack.
(Hence, it is vulnerable to choose IV as 1, 2, 3,....).

The well known BEAST attack exploits this:

(Optional: <http://resources.infosecinstitute.com/ssl-attacks/>)

Reusing One-Time-Pad Key


- The Venona project is a classic example on such failure

(Optional: https://www.nsa.gov/about/files/cryptologic_heritage/publications/coldwar/venona_story.pdf)

95

VENONA

~~TOP SECRET~~



USSR

Ref. No: 3/NBF/T1799

Issued: /13/7/1966

Copy No: 201.

FRAGMENTARY PRAGUE TEXT (1948)

From: MOSCOW

To: PRAGUE

No: 36

1 March 48

To MIKES [MIKESh][i]

[3 groups unrecovered] LI...[a]

[62 groups unrecoverable]

meeting with TEREZIE [TEREZIYa][ii] and [2 groups unrecovered].

No. 1865

DIRECTOR

Note: [a] ~~Possibly~~ the beginning of a proper name.

Comments: [i] MIKES: Unidentified; a Czech surname presumably used as a covername.

[ii] TEREZIE: Unidentified; presumably a covername. Also occurs in MOSCOW-PRAGUE No. 37 of 1st March 1948 (3/NBF/T1868).

DISTRIBUTION

1.4.2 Predictable Secret-Key Generation

Random Number Generation

- **Scenario 1:**
 - You are coding a program for a **simulation system**, for e.g. to simulate road traffic
 - In the program, you need a sequence of random numbers, for e.g. to decide the speed of the cars
 - *How to get these random numbers?*
- **Scenario 2:**
 - You are coding a program for a **security system**
 - In the program, you need a random number, for e.g. you need to generate a random number as a temporary **secret key**
 - *How to get these random numbers?*

To be Discussed in Tutorial

- In Java, what is the difference between the following?
 - `java.util.Random`
 - `java.security.SecureRandom`
- In C, what is the difference between using the following:

```
#include <time.h>
#include <stdlib.h>

srand(time(NULL));
int r = rand();
```

and a more complicated version below?

```
int byte_count = 64;
char data[64];
FILE *fp;

fp = fopen("/dev/urandom", "r");
fread(&data, 1, byte_count, fp);
fclose(fp);
```

1.4.3 Designing Your Own Cipher

Caution!

- **Don't** design your own cryptosystem, or even make a slight modification to existing scheme, unless you has an in-depth knowledge of the topic!
- Read “Don't roll your own crypto”:
<http://security.stackexchange.com/questions/2202/lessons-learned-and-misconceptions-regarding-encryption-and-cryptology/2210#2210>

1.5 Kerckhoffs' Principle vs Security through Obscurity

Kerckhoffs' Principle

- “A system should be secure even if everything about the system, *except the secret key*, is a public knowledge”

Security through Obscurity

- To hide the design of the system in order to achieve security
- *Is it good or bad??*

Examples (*Against* Obscurity)

- RC4:
 - Was introduced in 1987 and its algorithm was a **trade secret**
 - In 1994, a description of its algorithm was **anonymously posted** in a mailing group.
 - See <http://en.wikipedia.org/wiki/RC4>
- MIFARE Classic:
 - A contactless smartcard widely used in Europe employed a set of proprietary protocols/algorithms
 - However, they were **reverse-engineered** in 2007
 - It turned out that the encryption algorithms were already known to be weak (using only 48bits) and breakable
 - See <http://en.wikipedia.org/wiki/MIFARE>
 - Optional: Presentation video by the researcher who reverse-engineered it: <http://www.youtube.com/watch?gl=SG&hl=en-GB&v=QJyxUvMGLr0>.
The algorithm was revealed at 14:00.)

Examples (*Supporting Obscurity*)

- Usernames:
 - They are not secrets
 - However, it is *not* advisable to publish all the usernames
- Computer network structure & settings:
 - E.g. location of firewall and the firewall rules
 - These are not secrets, and many users within the organization may already know the settings
 - Still, it is *not* advisable to them
- The actual program used in a smart-card:
 - It is not advisable to publish it
 - If the program is published, an adversary may be able to identify vulnerability that was previously unknown, or carry out side-channel attacks
 - A sophisticated adversary may be able to reverse-engineer the code nevertheless

So, Should We Use Obscurity???

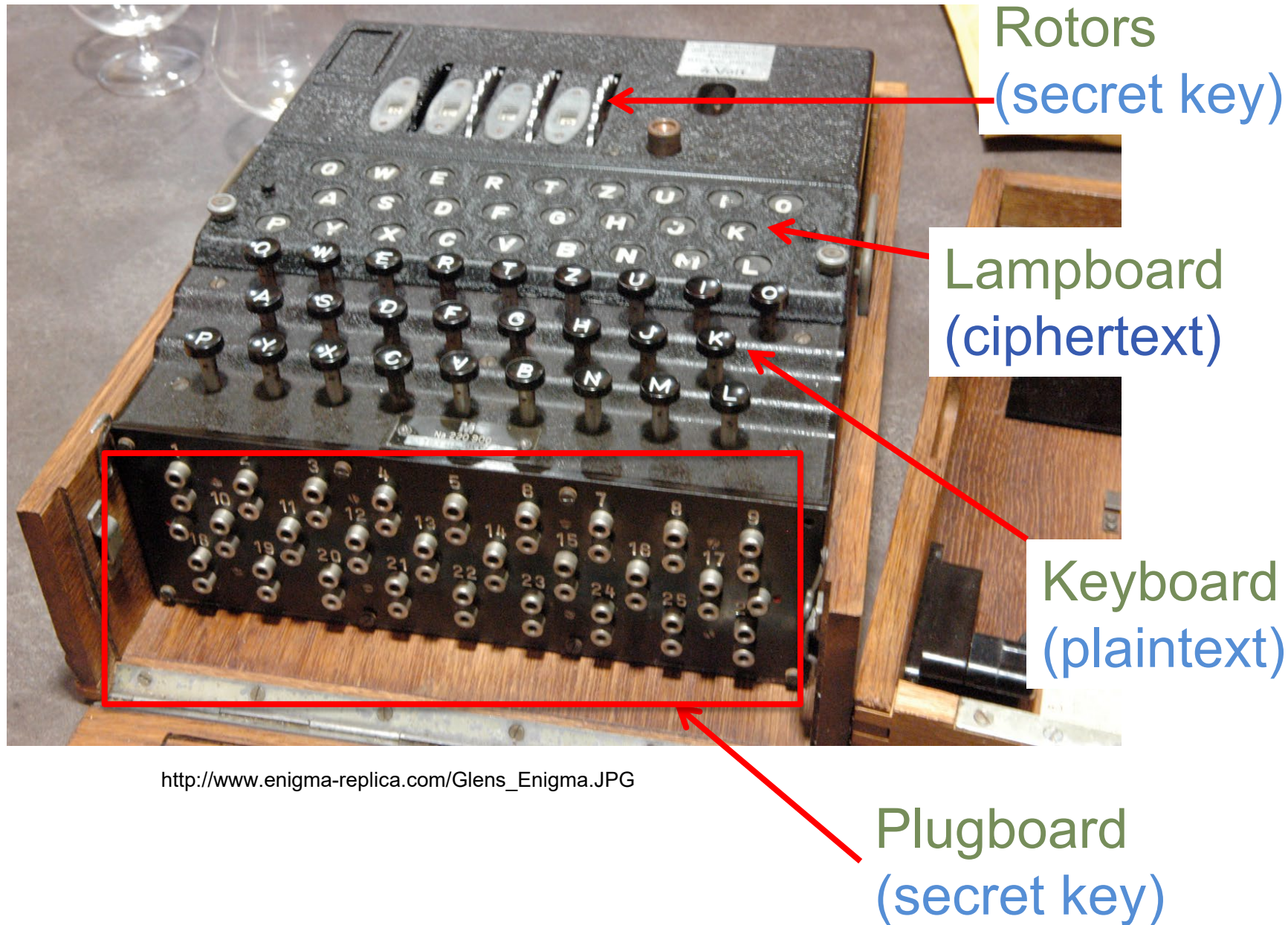
- In general, obscurity can be used as one layer in a ***defense in depth*** strategy
- It could deter or discourage novice attackers, but is ineffective against attackers with high skill and motivation
- The system **must remain secure** even if everything about it, *except its secret key*, becomes known
- In this module, we always assume that the attackers **know** the algorithms
- See:
 - <http://technet.microsoft.com/en-us/magazine/2008.06.obscurity.aspx>
 - http://en.wikipedia.org/wiki/Security_through_obscurity

1.6 Some Historical Facts

Cryptography: History

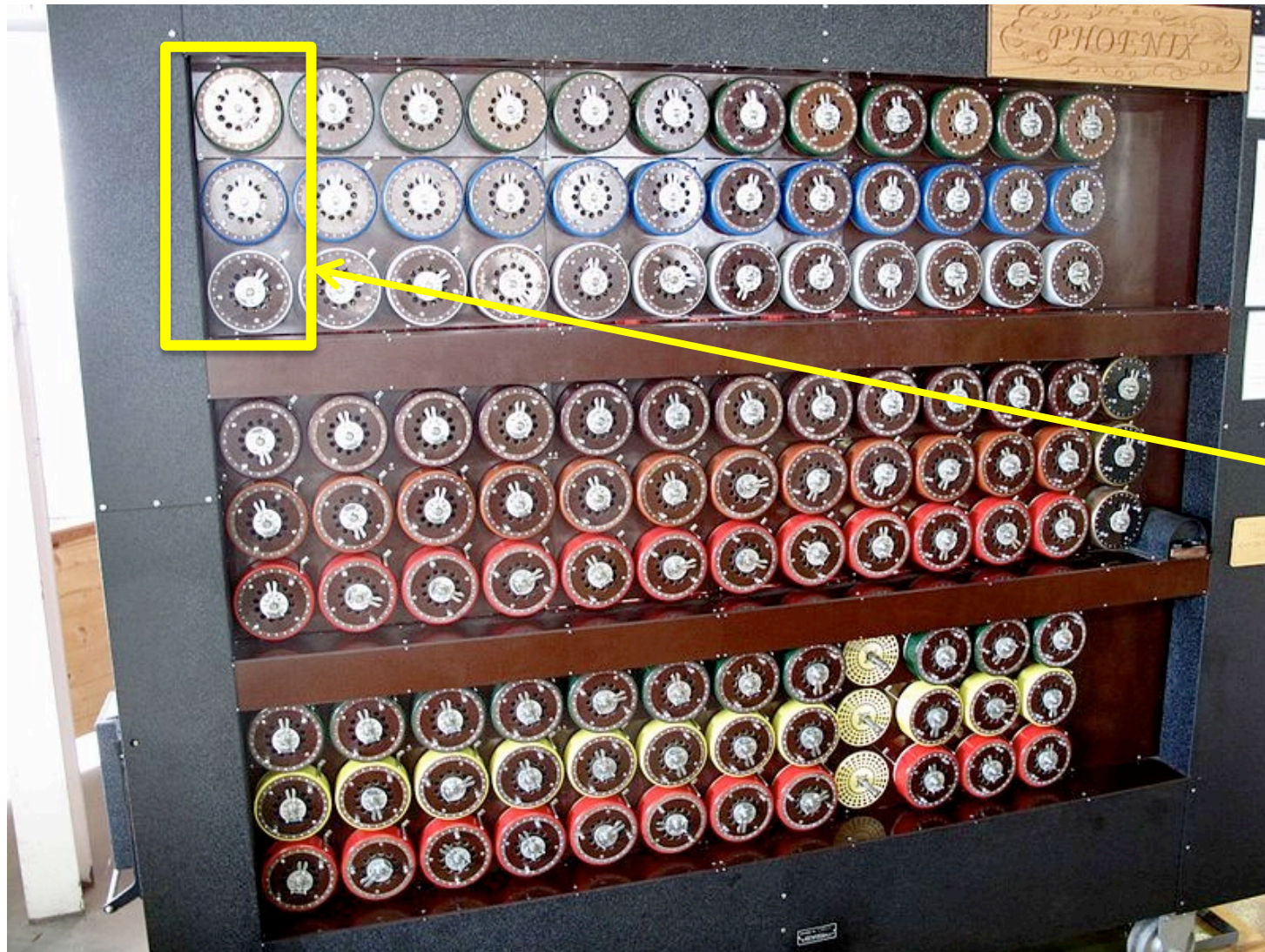
- Cryptography is closely related to warfare and can be traced back to ancient Greece
- Its role became significant when information is sent over the air
- WWII: Famous encryption machines include the Enigma, and the Bombe (that helped to break Enigma)

Enigma (Replica)



http://www.enigma-replica.com/Glens_Enigma.JPG

Working Rebuilt Bombe at Bletchley Park Museum



Simulates
the 3
rotors
in one
Enigma
machine

http://en.wikipedia.org/wiki/Cryptanalysis_of_the_Enigma#Crib-based_decryption

Modern Ciphers

DES (Data Encryption Standard):

- 1977: DES, 56 bits

(During cold war, cryptography, in particular DES was considered as “munition”, and subjected to export control. Currently, export of certain cryptography products is still controlled by US.

Read the crypto law survey's section on Singapore at <http://www.cryptolaw.org/cls2.htm>)

- 1998: A DES key broken in 56 hours
- Triple DES (112 bits) is still in used

AES (Advanced Encryption Standard):

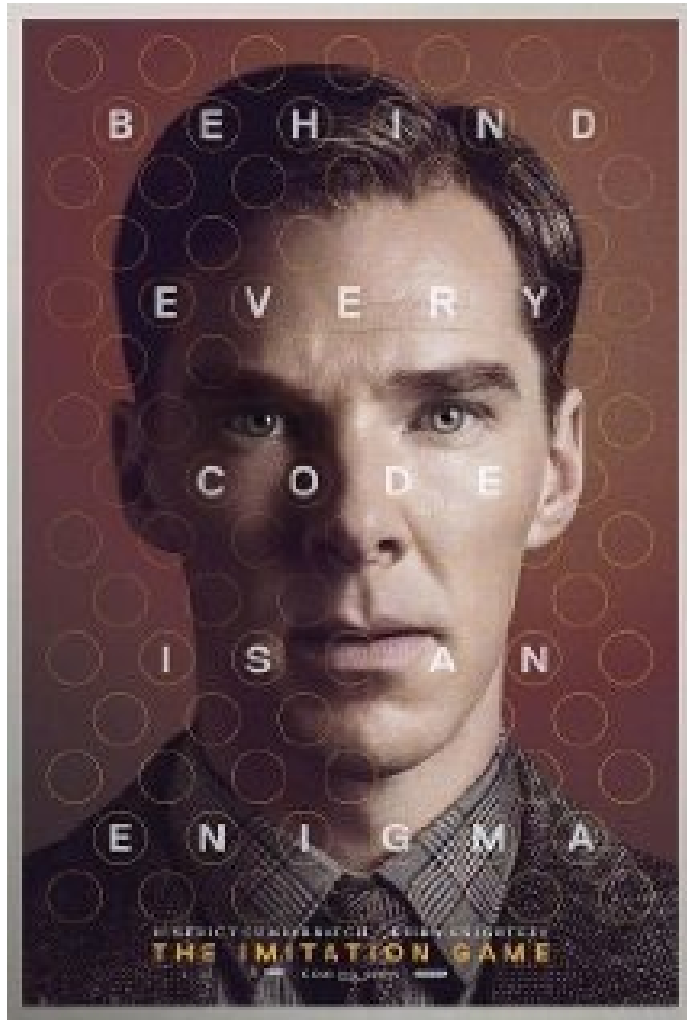
- 2001: NIST. 128, 192, 256 bits

Modern Ciphers

RC4:

- 1987: Designed by Ron Rivest (RSA Security), initially a trade secret
- 1994: Algorithm leaked in
- 1999: Used in widely popular **WEP** (for WiFi);
WEP implementation has 40 or 104-bit key
- 2001: A weakness in how WEP adopts RC4 is published by Fluhrer, Mantin, Shamir
- 2005: A group from FBI demonstrated the attack
- Afterward: Industry switched to WPA2
(with WPA as an intermediate solution)

Movie About Encryption



“The Imitation Game”:

During World War II, mathematician **Alan Turing** tries to crack Enigma with help from fellow mathematicians

[\(http://www.imdb.com/title/tt2084970/\)](http://www.imdb.com/title/tt2084970/)

Sample Tutorial Questions

Question:

Bob encrypted a video file using Winzip, which employs the 256-bit key AES. He choose a 6-digit number as password.

Winzip generated the 256-bit key from the 6-digit password using a “hash” function, say SHA1.

Alice obtained the ciphertext.

Alice also knew that Bob used a 6-digit password.

Given a “guess” of the 256-bit key, Alice can determine whether the key can successfully decrypted the file.

How many guesses Alice really needed to make in order to get the video?