Cryptography & Security

1. Introduction to Cryptography and Security

Aureliu Zgureanu,
PhD, Associate Professor

Rough Road-map of the Course (9 Topics)

Art & Science

- Intro to Cryptography
- Classical ciphers

Encryption

- Symmetric/ Secret key Enc.
  - Stream cyphers
  - Block cyphers
- Asymmetric/ Public key Enc.

Integrity/Authenticity

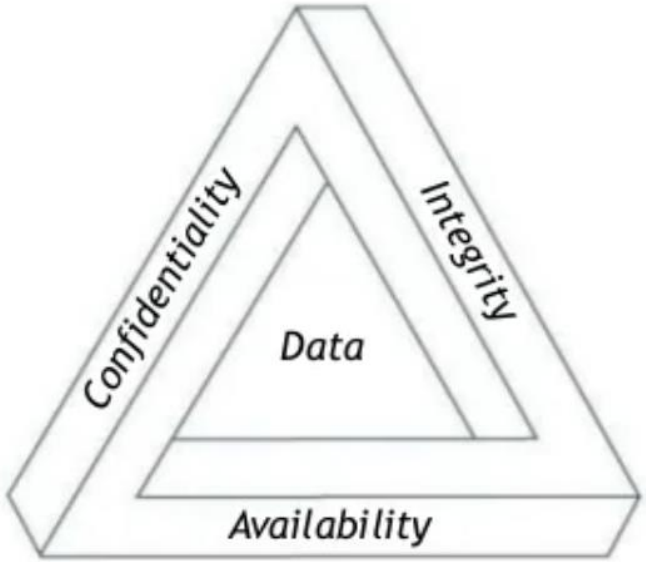
- One-way Function
- Message authentication code
- Digital signatures

Additional algorithms

- Modular calculation
- Pseudo-random Generator
- Pseudo-random Function
- ...

Key Establishment

- Certificates
- Public-Key Infrastructures



CIA Security Model

Confidentiality, Integrity & Availability

What is the CIA Model?

- A simple but widely-applicable security model is the CIA triad standing for:
  - Confidentiality
  - Integrity
  - Availability
- These are the three key principles which should be guaranteed in any kind of secure system.
- This principle is applicable across the whole subject of Security Analysis, from access to a user's internet history to security of encrypted data across the internet.
- If any one of the three can be breached it can have serious consequences for the parties concerned.

Confidentiality

- Confidentiality is the ability to hide information from those people unauthorised to view it.
- It is perhaps the most obvious aspect of the CIA triad when it comes to security; but correspondingly, it is also the one which is attacked most often.
- Cryptography and Encryption methods are an example of an attempt to ensure confidentiality of data transferred from one computer to another.

Integrity

- The ability to ensure that data is an accurate and unchanged representation of the original secure information.

- One type of security attack is to intercept some important data and make changes to it before sending it on to the intended receiver.

Availability

- It is important to ensure that the information concerned is readily accessible to the authorised viewer at all times.

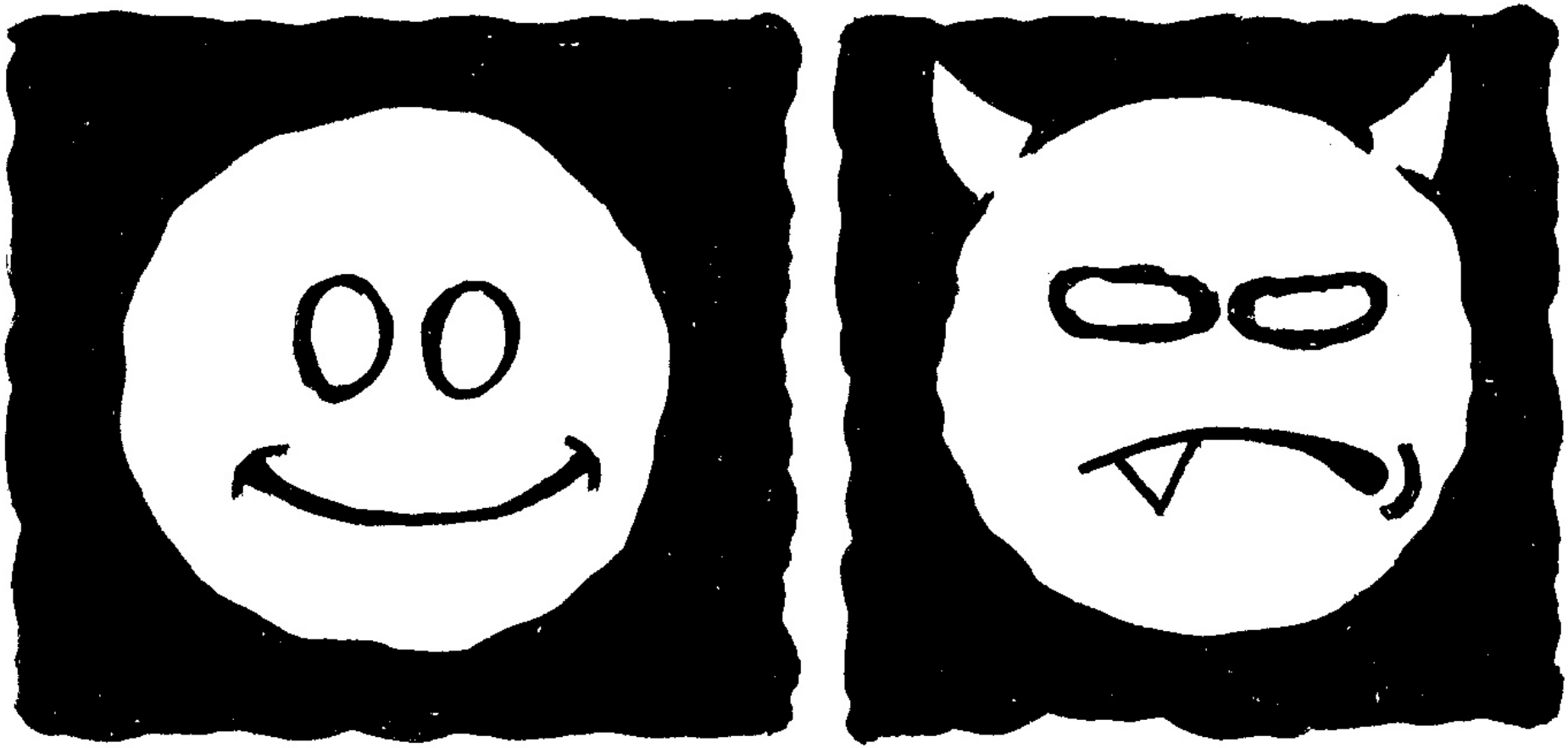
- Some types of security attack attempt to deny access to the appropriate user, either for the sake of inconveniencing them, or because there is some secondary effect.

- For example, by breaking the web site for a particular search engine, a rival may become more popular.

Nonrepudiation:

- prevents either sender or receiver from denying a transmitted message.
- When a message is sent, the receiver can prove that the alleged sender in fact sent the message.
- When a message is received, the sender can prove that the alleged receiver in fact received the message.

Why Cryptography?



Goal of an Ethical Cryptographer: Protect Good from Bad

Prerequisite: Being able to distinguish between Good and Bad.
High Ethical values.

What is Cryptography?

- The study of mathematical techniques for securing digital information, systems, and distributed computations against adversarial attacks.

-Tools to control access to information;
-Tools to enforce policies who can learn/influence information

Crypto: Past and Present

Classical Crypto

Scope

Scientific Basis

Consumer/End-user

Modern Crypto

Message Communication

Authenticated Message Communication

E-cash

E-election, E-auction

Activism with Safety

Secure Storage, Secret Sharing

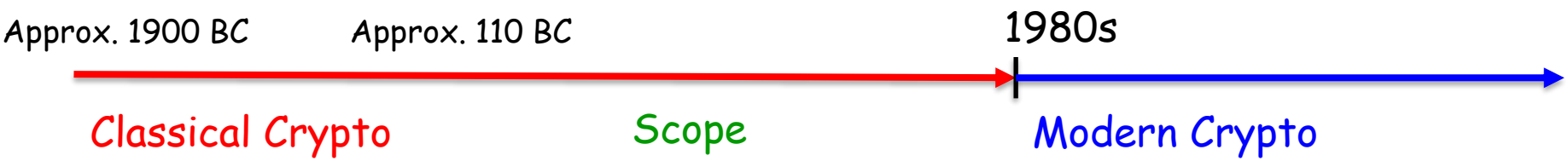
Secure Information Retrieval

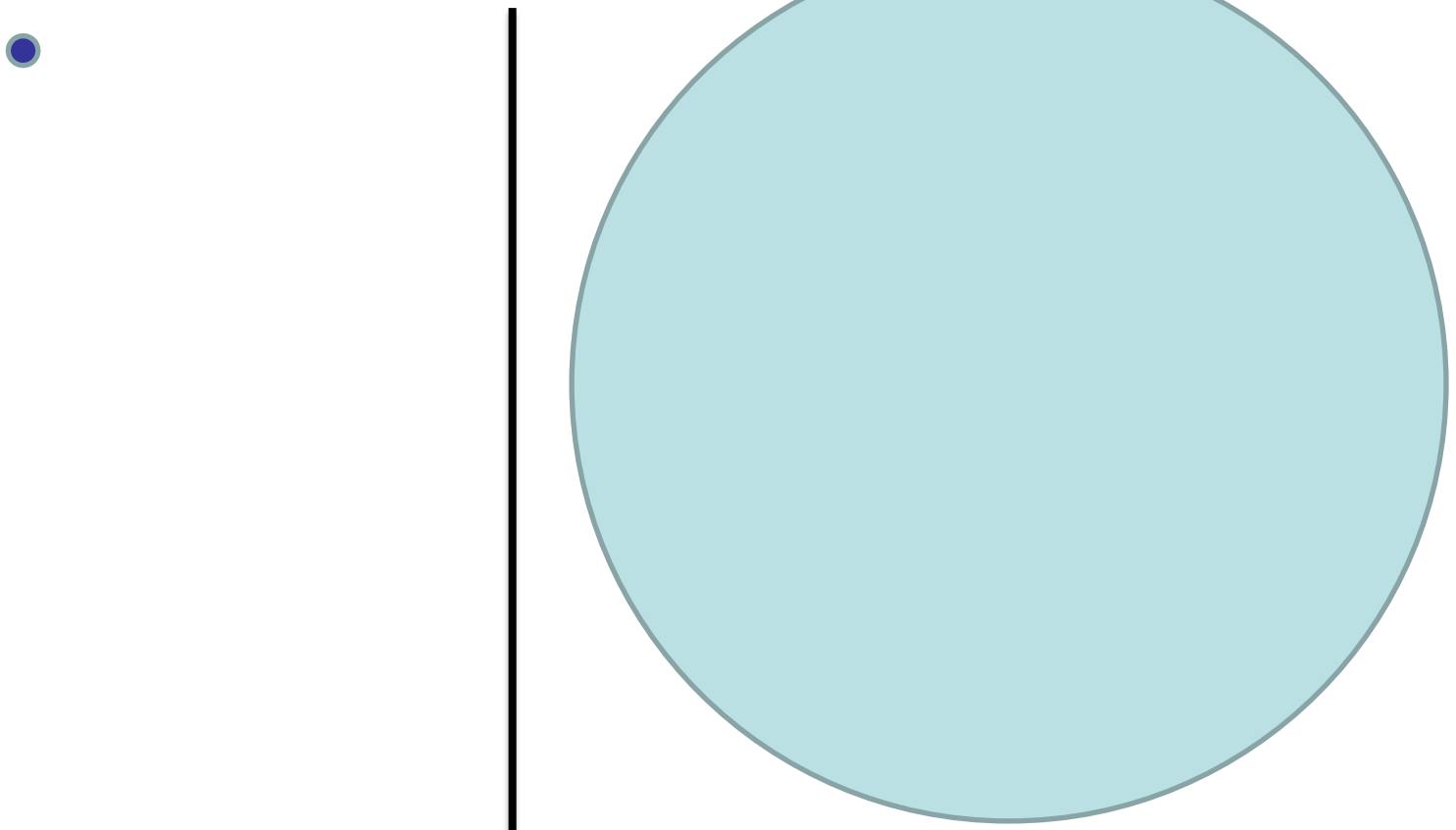
Secure outsourcing to Cloud

Secure Computation: Holy-grail

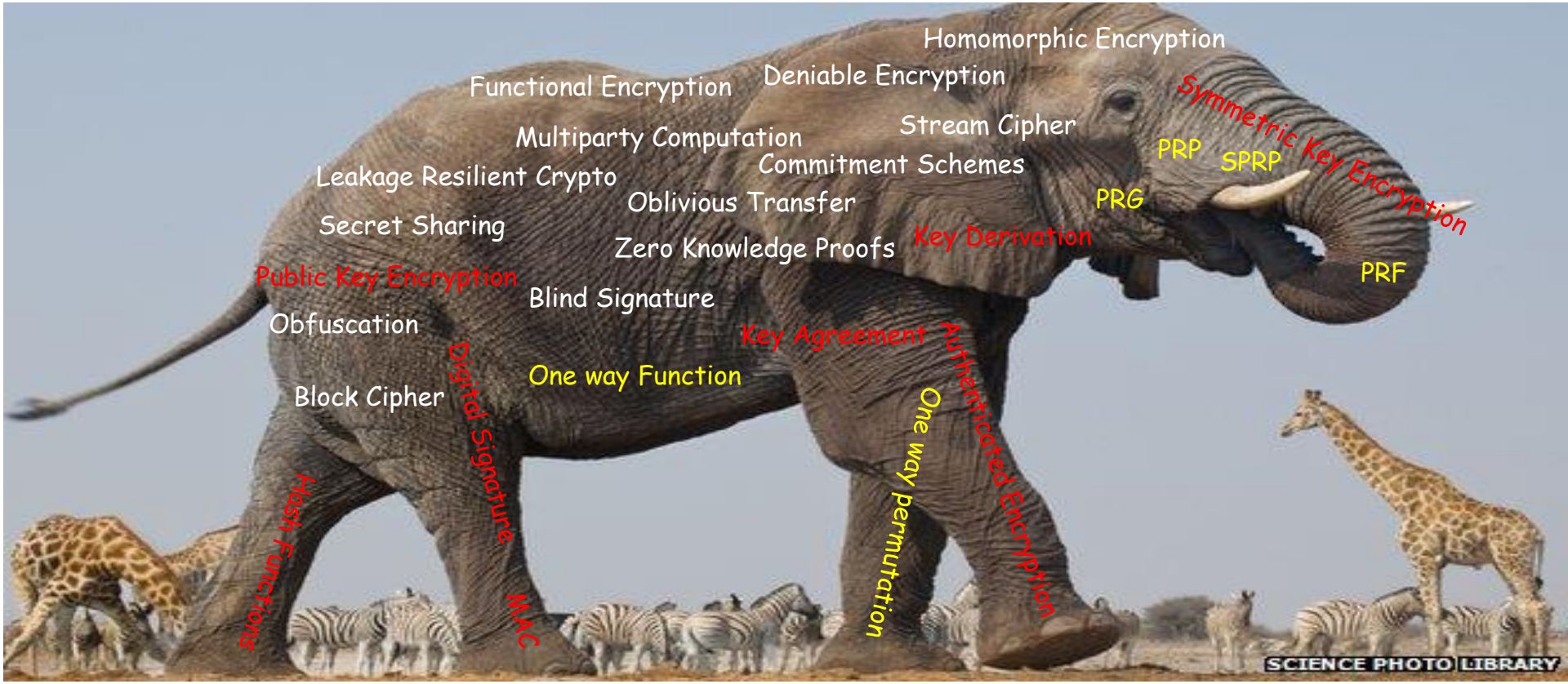
…….and the list goes on

Crypto: Past and Present





Crypto Elephant



Formal Definition of Security

Two components of a security definition:

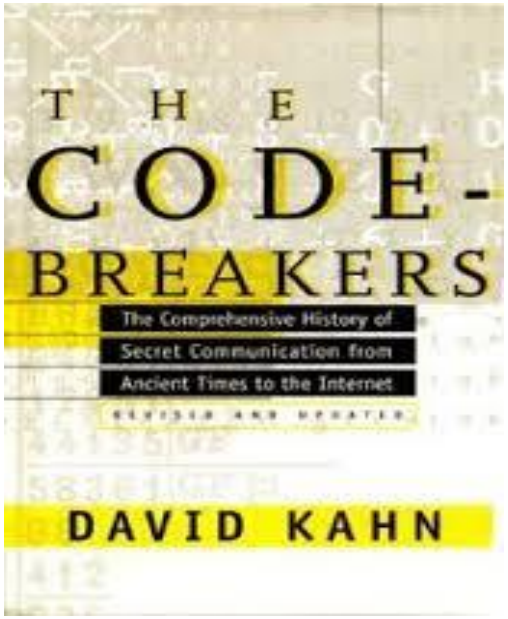
Threat:
- >> Who is your threat?
- >> Who do you want to protect from?
- >> Cultivate your enemy a.k.a adversary in crypto language.
- >> Look out in practical scenarios / be an adversary
- >> Unless you know your adv, no hope of defeating him

Break:
- >> What are you afraid of losing?
- >> What do you want to protect?
- >> If you don't know what to protect then how to do you when or if you are protecting it?
- >> Means different for different task

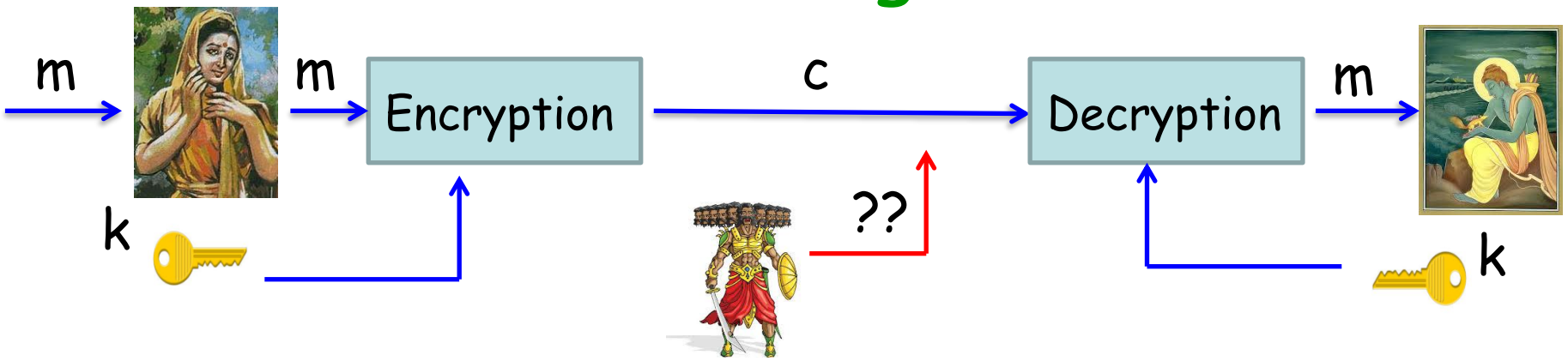
Before entering the world of Modern Crypto…

› Let us Learn From Classical Crypto about the classic blunders and golden rules

› A wonderful account of ciphers, right from pre-historic era to modern era



Secure Communication in Private Key Setting



- Secret key k shared in advance (by "some" mechanism)
- m is the plain-text
- c is the cipher-text (scrambled message)
- Symmetry: same key used for encryption and decryption

Syntax of Secret/Private/Symmetric Key Encryption

1. Key-generation Algorithm (Gen()):

   - Outputs a key k chosen according to some probability distribution.
   - MUST be a Randomized algorithm

2. Encryption Algorithm ($Enc_k(m)$); m from {0,1}*:

   - c ← $Enc_k(m)$ when randomized and c:=$Enc_k(m)$ when deterministic
   - Deterministic/Randomized algorithm

3. Decryption Algorithm ($Dec_k(c)$):

   - Outputs m:= $Dec_k(c)$
   - Usually deterministic

Syntax of SKE

Any cipher defines the following three space (sets):

1. **Key space ($\mathcal{K}$):**
   - Set of all possible keys output by algorithm Gen
   - Usually Gen selects a key k uniformly at random from $\mathcal{K}$
2. **Plain-text (message) space ($\mathcal{M}$):**
   - Set of all possible "legal" message (i.e. those supported by Enc)
3. **Cipher-text space ($\mathcal{C}$):**
   - Set of all cipher-texts output by algorithm Enc
   - The sets $\mathcal{K}$ and $\mathcal{M}$ together define the set $\mathcal{C}$

Any cipher is defined by specifying (Gen, Enc, Dec) and $\mathcal{M}$

Julius Ceaser's Cipher

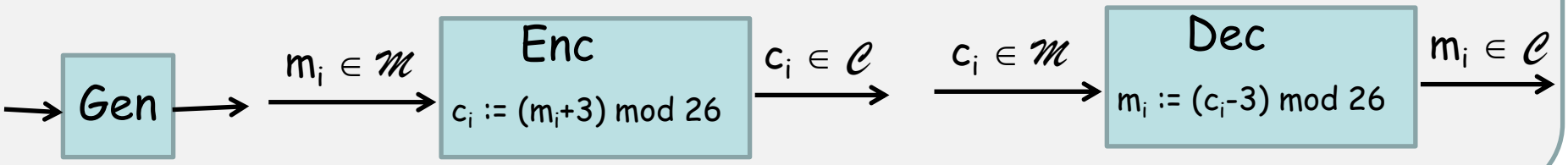
- One of the oldest recorded ciphers
- Encryption algorithm :

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	…	w	x	y	z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕		↕	↕	↕	↕
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R		Z	A	B	C

  - Plain-text --- begintheattacknow
  - Cipher-text --- EHJLQWKHDWWDFNQRZ

- Formally, interpret the alphabet set {a, b , …, z} as set {0, 1, …, 25}

  $\mathcal{M} = \mathcal{C} =$ set of strings over {0, 1, …, 25}



- Trivial to break
  - No secret key
  - Encryption/decryption algorithm must remain private

Keys and Kerckhoffs' Principle

- To maintain security key k should be definitely a secret
- What about Enc and Dec algorithm ?
  - More security by keeping them private too ?

Kerckhoffs' Principle:

"The cipher method must not be required to be secret and it must be able to fall into the hands of the enemy without any inconvenience "

a.k.a

Security rely solely on the secrecy of the key

19th century Dutch cryptographer

Arguments for Kerckhoffs' Principle

P1: Maintaining the privacy of a "short" key is easier than maintaining the privacy of a "large" algorithm

- Key ≈ 100 bits
- Program: 1000 times larger

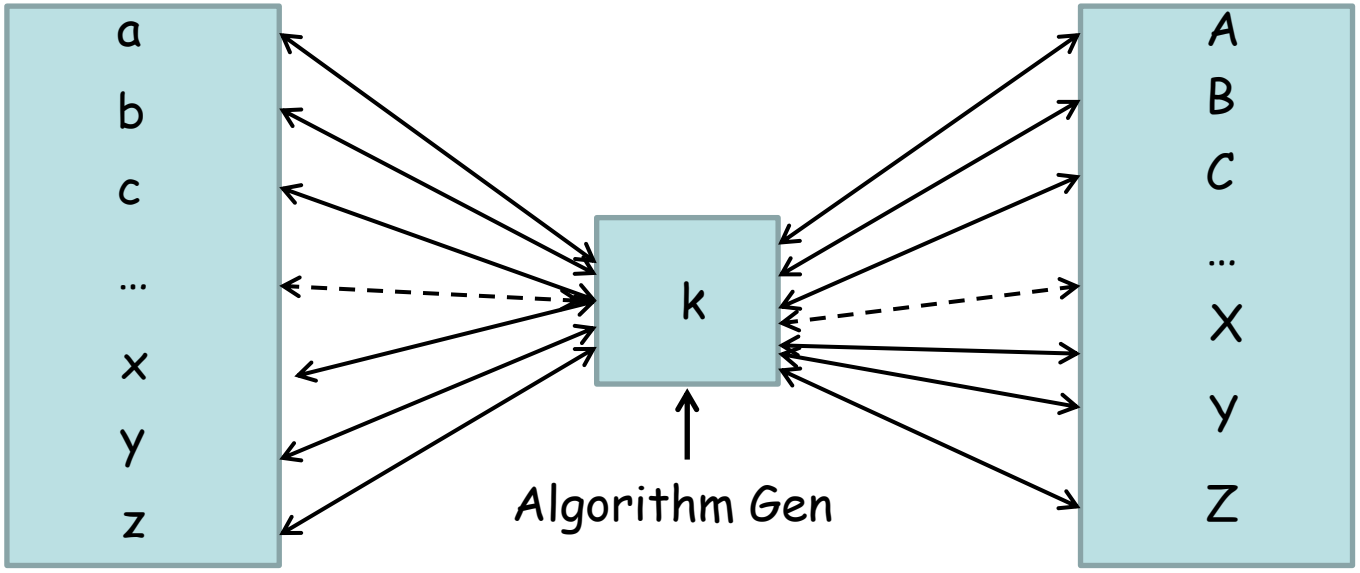
P2: Easy to replace a key than a whole program when exposed

P3: Infeasible to imagine a secret pair of algorithms for every pair of communicating parties

Shift Cipher

"Generalization" of Caesar's cipher

- k is a key randomly chosen from {0,1,…25}
- Plain-text characters "shifted" by k positions "forward"
- Cipher-text characters "shifted" k positions "backward"



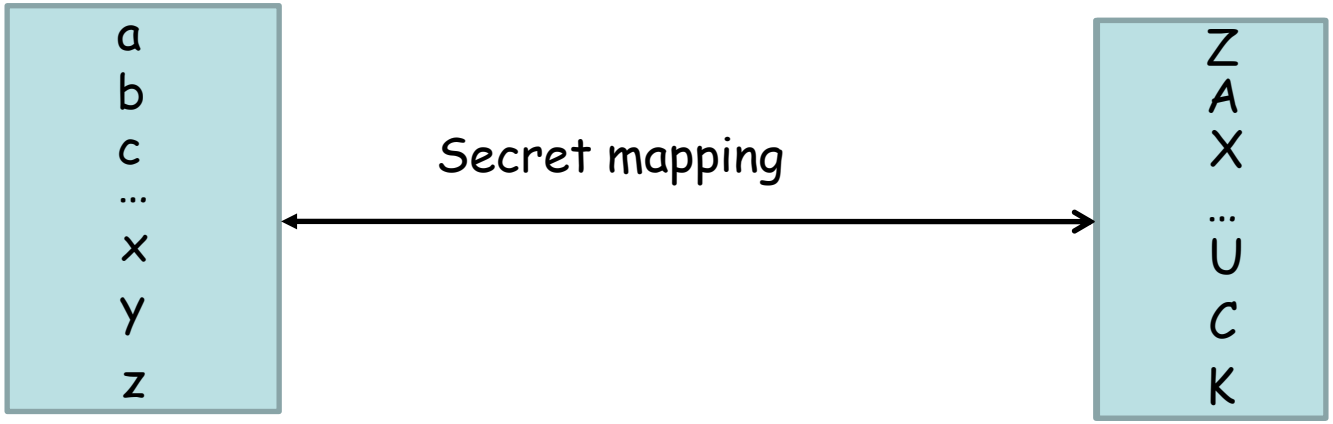
Easy break: Brute-force attack / Exhaustive search (Try all possible 26 keys)

Sufficient key-space principle

Mono-alphabetic Substitution

› Key k: a secret random permutation

Ex:



› infeasible to search $\mathcal{K}$ exhaustively in one's life-time $\mathcal{K} = 26! \approx 2^{88}$

› Easy Break: Frequency/statistical Analysis (exploiting statistical pattern of the English Language)

Works because the mapping from a plain-text character to the corresponding cipher-text character is fixed

Vigenere (Poly-alphabetic Shift) Cipher

Idea: each instance of plain-text character is mapped to different cipher-text characters

Aim: "smoothing out" the probability distribution of cipher-text characters

- Key a random word of length t

  Ex: cafe (t = 4)

- Enc: Divide plain-text into blocks of t and "shift" each block according to the key

  Ex: plain-text --- tell him about me

  tell hima bout me

  cafe cafe cafe ca

  Cipher-text --- WFRQ KJSF EPAY PF

- Break: Se shift cipher)

  - Crypto must not adhere to adhocism.
  - Need Solid Scientific Framework
  - Towards Modern Crypto

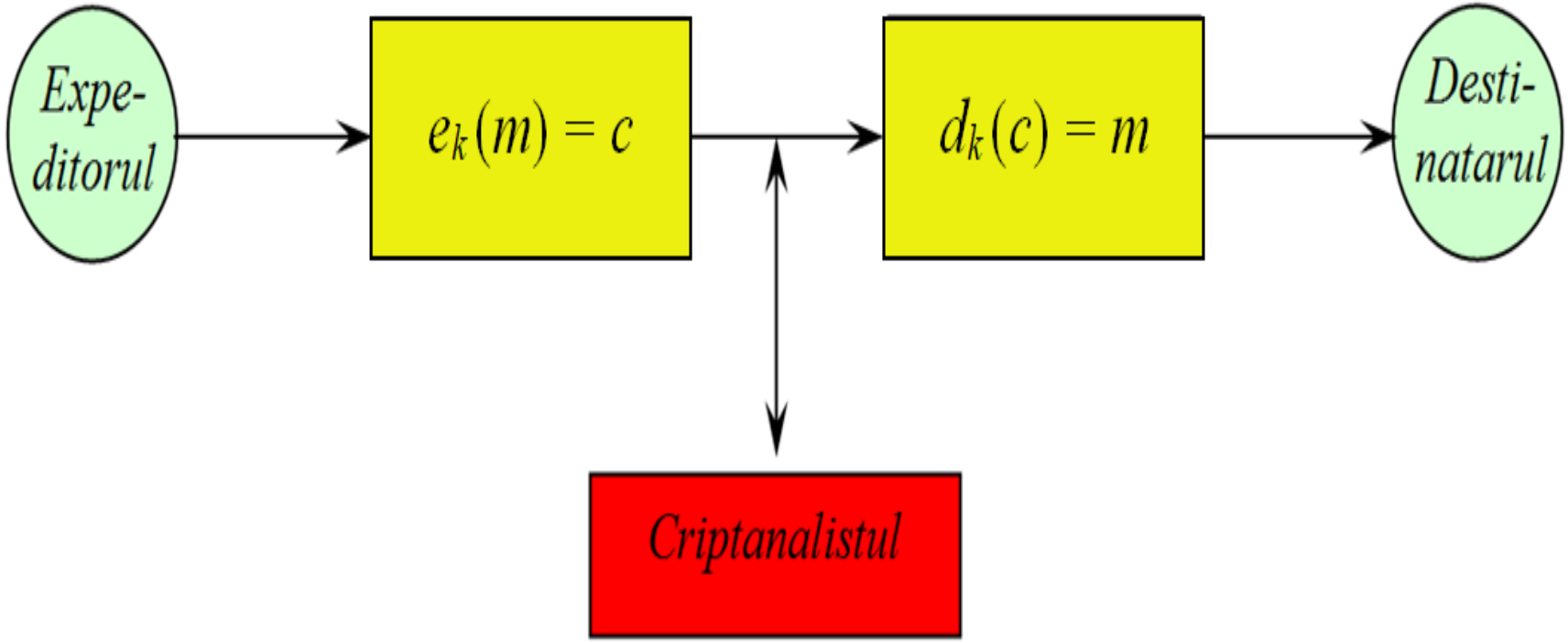
Thank You!

Cryptography & Security

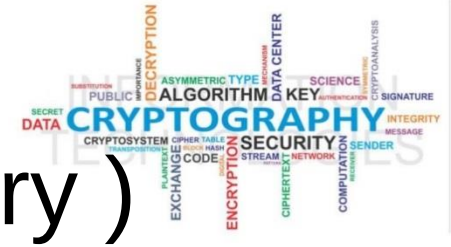
2. Classical ciphers

Aureliu Zgureanu,
PhD, Associate Professor

Classical cipher scheme



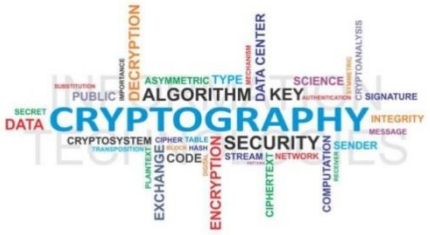
Encryption criterias
(Francis Bacon, 17th-century)

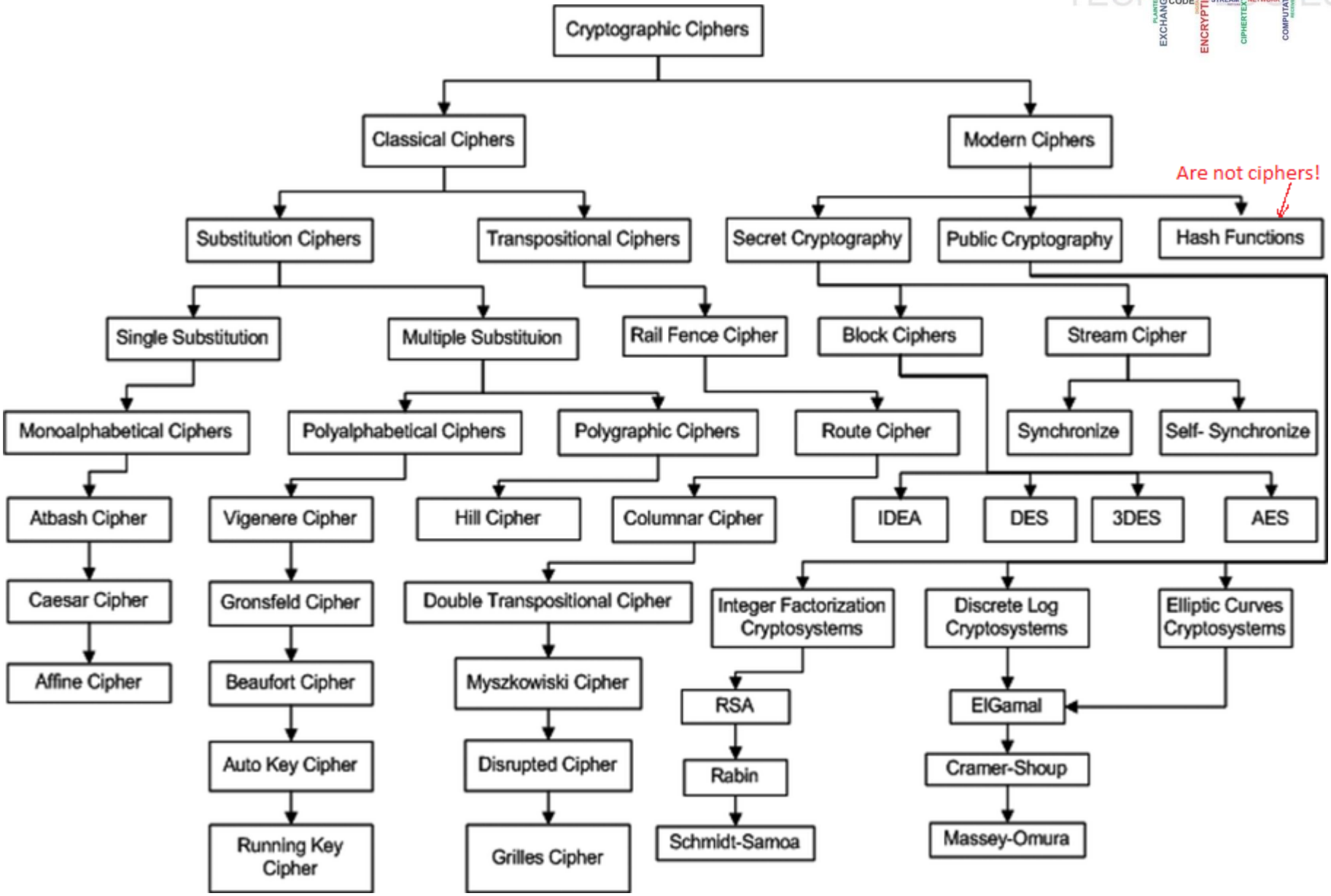


1. Having e_k, ***k, a***nd ***m*** – it is easy to calulate $e_k(m)$.
2. Having d_k, ***k, a***nd ***c*** – it is easy to calulate $d_k(c)$.
3. It is **imposible** to find out ***m*** from ***c***, without knowing ***k***.

Now **imposible** is changed by very **difficult!**

Classification Of Ciphers





Substitution Ciphers

- Great tool for understanding brute-force vs. analytical attacks
- Encrypts letters rather than bits (like all ciphers until after WW II)

Idea: replace each plaintext letter by a fixed other letter.

Plaintext		Ciphertext
A	→	k
B	→	d
C	→	w
	....	

for instance, ABBA would be encrypted as kddk

- Example (ciphertext):

iq ifcc vqqr fb rdq vfllcq na rdq cfjwhwz hr bnnb hcc hwwhbsqvqbre hwq vhlq

- How secure is the Substitution Cipher?

Monoalphabetic Cipher and Polyalphabetic Cipher

SR.NO	Monoalphabetic Cipher	Polyalphabetic Cipher
1	Monoalphabetic cipher is one where each symbol in plain text is mapped to a fixed symbol in cipher text.	Polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets.
2	The relationship between a character in the plain text and the characters in the cipher text is one-to-one.	The relationship between a character in the plain text and the characters in the cipher text is one-to-many.
3	Each alphabetic character of plain text is mapped onto a unique alphabetic character of a cipher text.	Each alphabetic character of plain text can be mapped onto 'm' alphabetic characters of a cipher text.

Julius Ceaser's Cipher

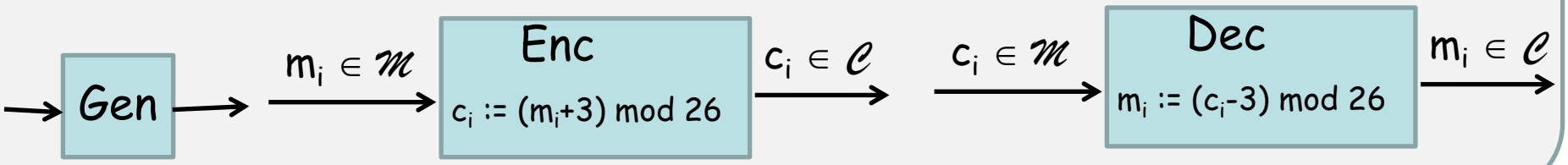
- One of the oldest recorded ciphers
- Encryption algorithm :

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	…	w	x	y	z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕		↕	↕	↕	↕
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R		Z	A	B	C

  - Plain-text --- begintheattacknow
  - Cipher-text --- EHJLQWKHDWWDFNQRZ

- Formally, interpret the alphabet set {a, b , …, z} as set {1, …, 25}

  $\mathcal{M} = \mathcal{C} =$ set of strings over {0, 1, …, 25}



- Trivial to break
  - No secret key
  - Encryption/decryption algorithm must remain private

Modular Arithmetic

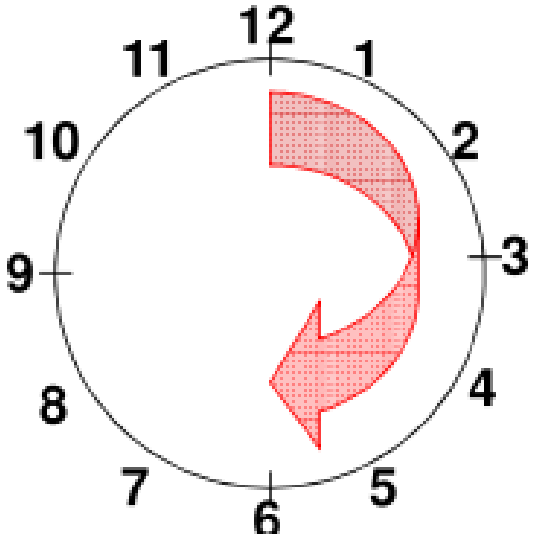
Why do we need to study modular arithmetic?

- **Extremely important for asymmetric cryptography (RSA, elliptic curves etc.)**
- **Some historical ciphers can be elegantly described with modular arithmetic (cf. Caesar and affine cipher later on).**

Most cryptosytems are based on **sets of numbers** that are

1. **discrete** (sets with integers are particularly useful)
2. **finite** (i.e., a finitely many numbers)

Even though the numbers are incremented every hour we never leave the set of integers
{1, 2, 3, … 11, 12}



Modular Arithmetic

- A system which allows to **compute** in finite sets of integers like the 12 integers we find on a clock (1,2,3, … ,12).
- It is crucial to have an operation which „keeps the numbers within limits", i.e., after addition/multiplication they never leave the set (i.e., never larger than 12).

Definition: Modulus Operation

Let a, r, m be integers and $m > 0$. We write $a \equiv r \bmod m$

if $(a-r)$ *is divisable by* m.

- "m" is called the **modulus;** "r" is called the **remainder**

Examples for modular reduction.

- a=12 and m=9 : $12 \equiv 3 \bmod 9$; a=34 and m=9: $34 \equiv 7 \bmod 9$
- a= -7 and m=9: $-7 \equiv 2 \bmod 9$

(verify that the condition „m divides $(a-r)$" holds in each case)

Modular Arithmetic

- **The remainder is not unique**

$$12 \equiv 3 \bmod 9; \quad 12 \equiv 21 \bmod 9; \quad 12 \equiv -6 \bmod 9$$

By convention, we agree on the **smallest positive integer** r as remainder. This integer can be computed as

$$a = q\ m + r \qquad \text{where } 0 \le r \le m-1$$

- Example: $a=12$ and $m=9$

$$12 = 1 \times 9 + 3 \quad \rightarrow \quad r = 3$$

Remark: This is just a convention. Algorithmically we are free to choose any other valid remainder to compute our crypto functions.

$$(a+b) \bmod m = (a \bmod m + b \bmod m) \bmod m$$

$$(a \times b) \bmod m = \{(a \bmod m) \times (b \bmod m)\} \bmod m$$

Modular Arithmetic

- **How do we perform modular division?**

rather than divide, we prefer to multiply by the inverse. Ex:

$$b / a \equiv b \times a^{-1} \bmod m$$

The inverse a^{-1} of a number a is defined such that: $a\, a^{-1} \equiv 1 \bmod m$

Ex: What is $5 / 7 \bmod 9$?

The inverse of $7 \bmod 9$ is 4 since $7 \times 4 \equiv 28 \equiv 1 \bmod 9$, hence:

$$5 / 7 \equiv 5 \times 4 = 20 \equiv 2 \bmod 9$$

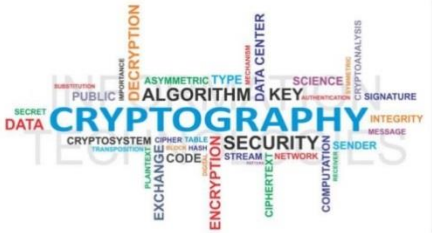
- **How is the inverse computed?**

The inverse of a number $a \bmod m$ only exists if and only if: gcd $(a, m) = 1$

(above gcd(7, 9) = 1, so that the inverse of 7 exists modulo 9)

For now, the best way of computing the inverse is to use exhaustive search. The Euclidean Algorithm computes an inverse for a given number and modulus

Julius Ceaser's Cipher



Mathematical model

- $k \in \{1, 2, 3, \ldots, n-1\}$,
- $c = e_k(x) = x + k \bmod n$,
- $m = d_k(y) = y - k \bmod n$.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
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

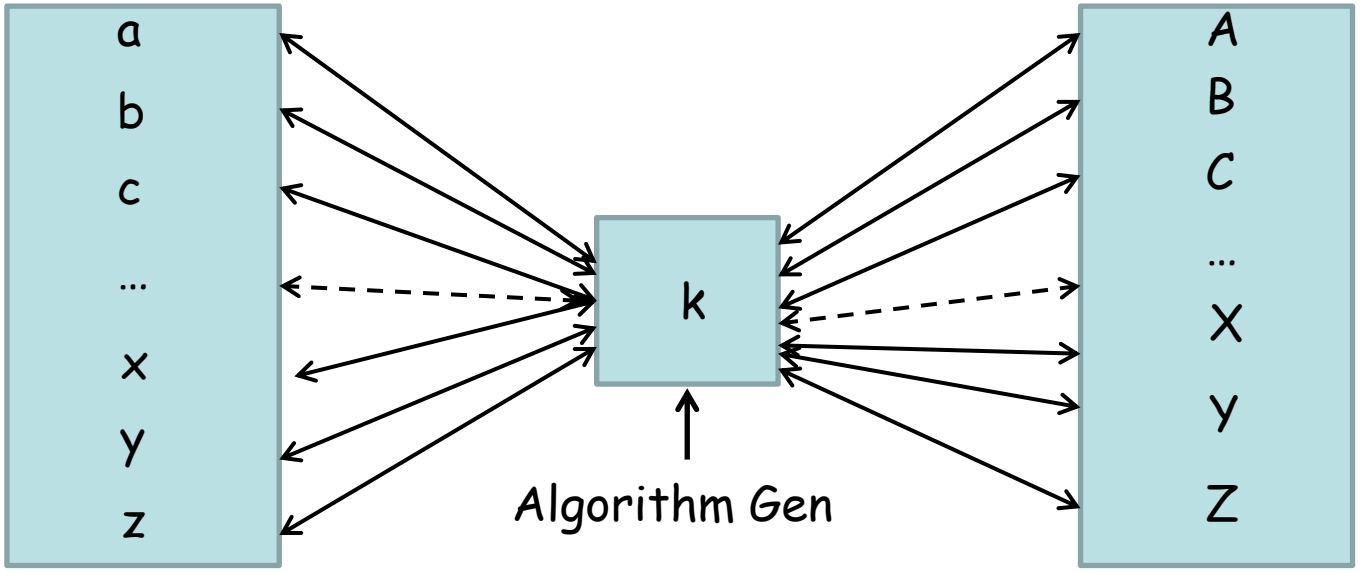
A Julius Ceaser's Cipher Tool

- Shift/Caesar Cipher - rotate each letter of the plaintext by a fixed amount
- Example:
  - Plaintext - SEND HELP
  - Key - rotate up by 13
  - Ciphertext - FRAQ URYC

Shift Cipher

"Generalization" of Caesar's cipher

- k is a key randomly chosen from {1,…25}
- Plain-text characters "shifted" by k positions "forward"
- Cipher-text characters "shifted" k positions "backward"



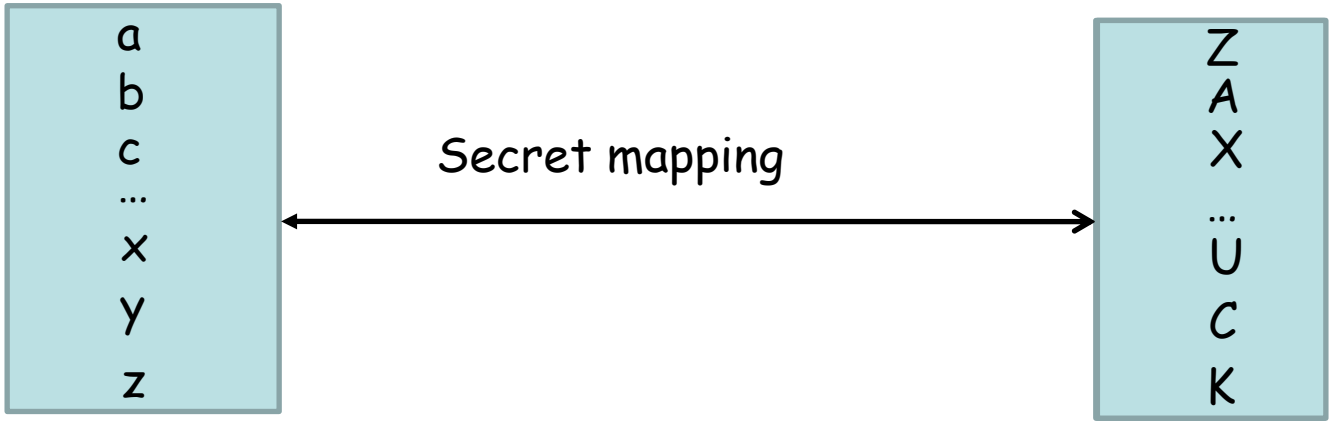
Easy break: Brute-force attack / Exhaustive search (Try all possible 26 keys)

Sufficient key-space principle

Mono-alphabetic Substitution

› Key k: a secret random permutation

Ex:



› infeasible to search $\mathcal{K}$ exhaustively in one's life-time $\mathcal{K} = 26! \approx 2^{88}$

› Easy Break: Frequency/statistical Analysis (exploiting statistical pattern of the English Language)

Works because the mapping from a plain-text character to the corresponding cipher-text character is fixed

Affine Cipher

- Extension of the shift cipher: rather than just adding a key to the plaintext, we also multiply
- We use for this a key consisting of two parts: k = (a, b)

  Let k, x, y ε {0, 1, …, 25}, **a, b ≠ 0, a ≠ 1**

  - Encryption: $y = e_k(x) \equiv (a\,x + b) \bmod 26$
  - Decryption: $x = d_k(x) \equiv a^{-1}(y - b) \bmod 26$

- Since the inverse of *a* is needed for inversion, we can only use values for *a* for which: $\gcd(a, 26) = 1$

  There are 12 values for *a* that satisfy this condition.

- The key space is only 12 x 26 = 312
- Again, several attacks are possible, including:
  - Exhaustive key search and letter frequency analysis, similar to

Finding a modular inverse
Euclidian extended alg.

i	x	y	r	q
1	1	0	n	
2	0	1	a	$q_2 = \left[\frac{r_1}{r_2}\right]$
3	$x_i = x_{i-2} - q_{i-1} \cdot x_{i-1}$	$y_i = y_{i-2} - q_{i-1} \cdot y_{i-1}$	$r_i = r_{i-2} \bmod r_{i-1}$	$q_3 = \left[\frac{r_2}{r_3}\right]$
…	…	…	…	…
k	$x_k = x_{k-2} - q_{k-1} \cdot x_{k-1}$	$y_k = y_{k-2} - q_{k-1} \cdot y_{k-1}$	1	$q_k = \left[\frac{r_{k-1}}{r_k}\right]$

Polybius square

Πολύβιος, Polýbios;
c. 200 – c. 118 BC

	m	*o*	*u*	*s*	*e*
m	A	B	C	D	E
u	F	G	H	I/J	K
s	L	M	N	O	P
c	Q	R	S	T	U
a	V	W	X	Y	Z

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

	a	*b*	*c*	*d*	*e*
a	A	B	C	D	E
b	F	G	H	I	J
c	K	L	M	N	O
d	P	R	S	T	U
e	V	W	X	Y	Z

Some attacks against the Substitution Cipher

1. Attack: Exhaustive Key Search (Brute-Force Attack)

- Simply try every possible subsititution table until an intelligent plaintext appears (note that each substitution table is a key).
- How many substitution tables (= keys) are there?

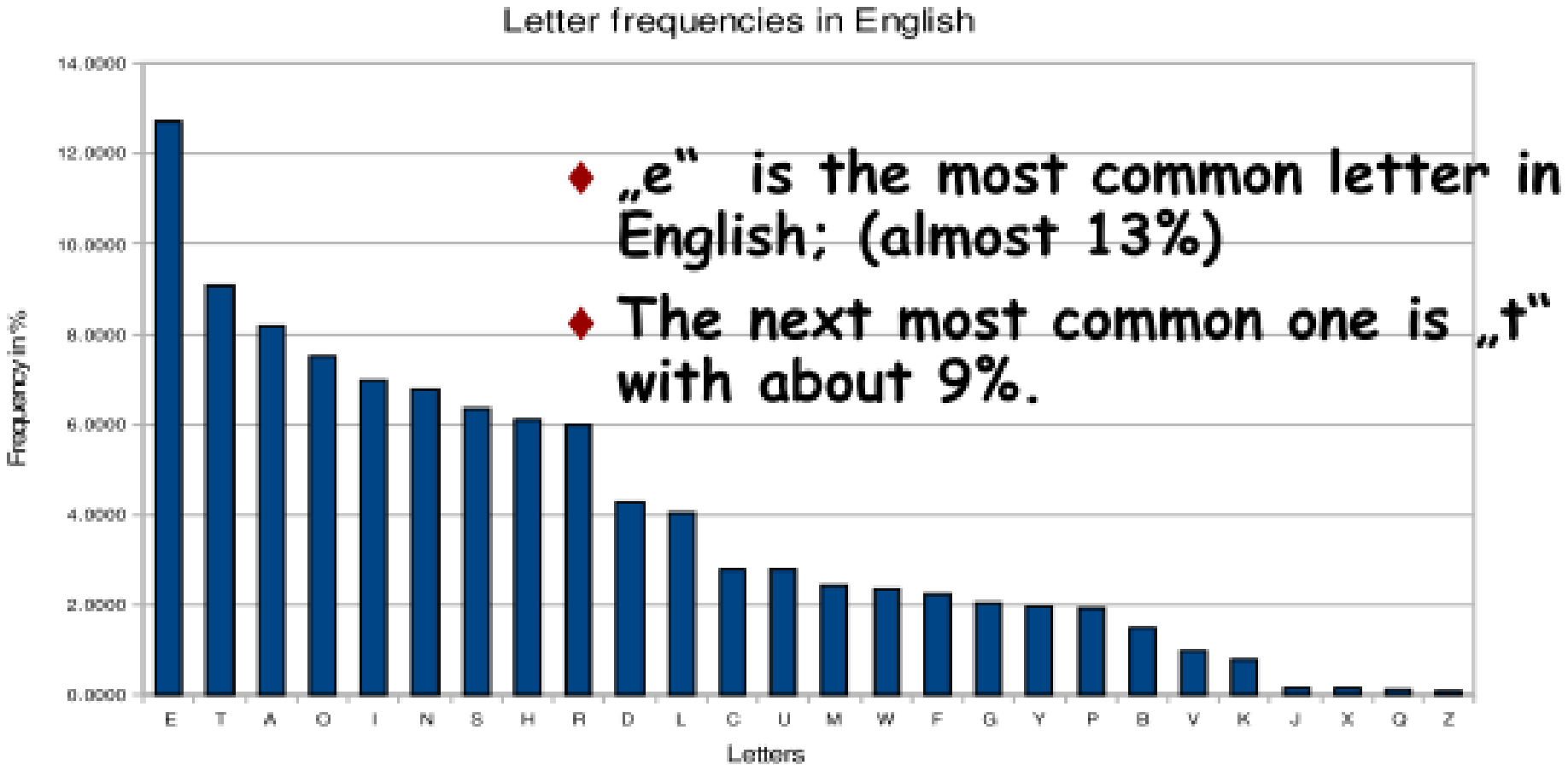
$$26 \times 25 \times \ldots \times 3 \times 2 \times 1 = 26! \approx 2^{88}$$

Search through 2^{88} keys is completely infeasible with today's computers

- Q: Can we now conclude that the substitution cipher is secure since a brute-forece attack is not feasible?
- A: No - We have to protect against all possible attacks.

2. Letter Frequency Analysis

- Letters have very different frequencies in English
- Moreover: frequency of plaintext letters is preserved in the ciphertext.



- „e" is the most common letter in English; (almost 13%)
- The next most common one is „t" with about 9%.

Attack: Letter Frequency Analysis

- Let's retun to our example and identify the most frequent letter:

iq ifcc vqqr fb rdq vfllcq na rdq cfjwhwz hr
bnnb hcc hwwhbsqvqbre hwq vhlq

- We replace the ciphertext letter q by E and obtain:

iE ifcc vEEr fb rdE vfllcE na rdE cfjwhwz hr
bnnb hcc hwwhbsEvEbre hwE vhlE

- By further guessing based on the frequency of the remaining letters we obtain the plaintext:

WE WILL MEET IN THE MIDDLE OF THE LIBRARY AT
NOON ALL ARRANGEMENTS ARE MADE

Attack: Letter Frequency Analysis

- In practice, not only frequencies of individual letters can be used for an attack, but also the frequency of letter pairs (i.e., „th" is very common in English), letter triples, etc.
- cf. Problem 1.1 in *Understanding Cryptography* for a longer ciphertext you can try to break

Important lesson: Even though the substitution cipher has a sufficiently large key space of appr. 2^{88}, it can easily be defeated with analytical methods. This is an excellent example that an encryption scheme must withstand all types of attacks.

Vigenere (Poly-alphabetic Shift) Cipher

Idea: each instance of plain-text character is mapped to different cipher-text characters

Aim: "smoothing out" the probability distribution of cipher-text characters

- Key a random word of length t

  Ex: cafe (t = 4)

- Enc: Divide plain-text into blocks of t and "shift" each block according to the key

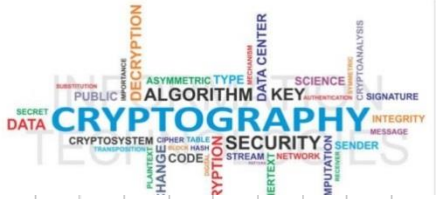
  Ex: plain-text --- tell him about me

  tell hima bout me

  cafe cafe cafe ca

  Cipher-text --- WFRQ KJSF EPAY PF

Polyalphabetic Ciphers
Vigenère cipher



- $c_i = m_i + k_i \pmod{26}$
- $m_i = c_i - k_i \pmod{26}$
- $k = (k_1, k_2, \ldots, k_p)$

If p=5 – there are $26^5 = 11\,881\,376$ different keys

	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
A	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	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	A
C	C	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
D	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
E	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	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

Vigenère cipher

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
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

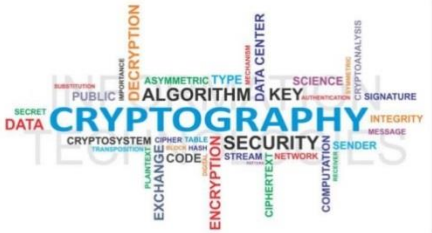
Encryption

Clear text M	P	E	R	A	S	P	E	R	A	A	D	A	S	T	R	A
Key K	**S**	**U**	**P**	**E**	**R**	S	U	P	E	R	S	U	P	E	R	S
Clear text M	15	4	17	0	18	15	4	17	0	0	3	0	18	19	17	0
Key K	18	20	15	4	17	18	20	15	4	17	18	20	15	4	17	18
$M+K$ (mod 26)	7	24	6	4	9	7	24	6	4	17	21	20	7	23	8	18
Ciphertext C	H	Y	G	E	J	H	Y	G	E	R	V	U	H	X	I	S

Decryption

Ciphertext C	H	Y	G	E	J	H	Y	G	E	R	V	U	H	X	I	S
Key K	S	U	P	E	R	S	U	P	E	R	S	U	P	E	R	S
Ciphertext C	7	24	6	4	9	7	24	6	4	17	21	20	7	23	8	18
Key K	18	20	15	4	17	18	20	15	4	17	18	20	15	4	17	18
$M-K$ (mod 26)	15	4	17	0	18	15	4	17	0	0	3	0	18	19	17	0
Clear text M	P	E	R	A	S	P	E	R	A	A	D	A	S	T	R	A

Homophonic cipher

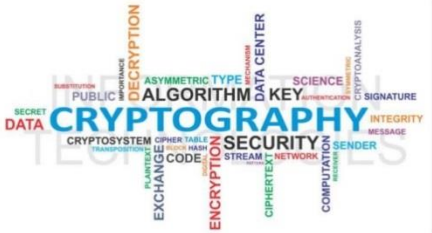


- $a \in M \rightarrow H(a) \subset c$:
- $H(a) \cap H(b) = \varnothing \Leftrightarrow a \neq b$;
- If ***a*** appears more frequently than ***b*** in clear text, then card ($H(a)$) ≥ card(H(b)).

Homophonic cipher

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
8	2	3	4	12	2	2	6	6	1	1	4	2	6	7	2	1	6	6	9	3	1	2	1	2	1
09	48	13	01	14	10	06	23	32	15	04	26	22	18	00	38	94	29	11	17	08	34	60	28	21	02
12	81	41	03	16	31	25	39	70			37	27	58	05	95		35	19	20	61		89		52	
33		62	45	24			50	73			51		59	07			40	36	30	63					
47			79	44			56	83			84		66	54			42	76	43						
53				46			65	88					71	72			77	86	49						
67				55			68	93					91	90			80	96	69						
78				57										99					75						
92				64															85						
				74															97						
				82																					
				87																					
				98																					

Playfair cipher



k= KEYWORD

m= "THE BIG WHEEL"→ TH EB IG WH EE LO ???

↓

"TH EB IG WH EQ EL" !!!

c=„ VF WD LH YJ WN OG"

K	E	Y	W	O
R	D	A	B	C
F	G	H	I J	L
M	N	P	Q	S
T	U	V	X	Z

Transposition cipher

M= Securitatea este asigurată

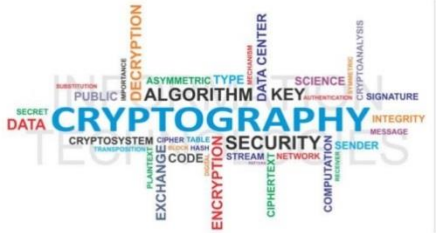
M=Securi tateae steasi gurată xyztwu

	1	2	3	4	5	6
1	*S*	*e*	*c*	*u*	*r*	*i*
2	*t*	*a*	*t*	*e*	*a*	*e*
3	*s*	*t*	*e*	*a*	*s*	*i*
4	*g*	*u*	*r*	*a*	*t*	*ă*
5	*x*	*y*	*z*	*t*	*w*	*u*

	1	2	3	4	5	6
5	*x*	*y*	*z*	*t*	*w*	*u*
3	*s*	*t*	*e*	*a*	*s*	*i*
4	*g*	*u*	*r*	*a*	*t*	*ă*
1	*S*	*e*	*c*	*u*	*r*	*i*
2	*t*	*a*	*t*	*e*	*a*	*e*

C=*xsgstytueazercttaauewstrauiăie*

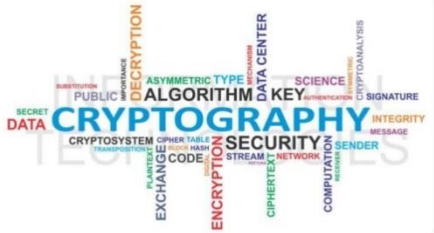
Transposition cipher



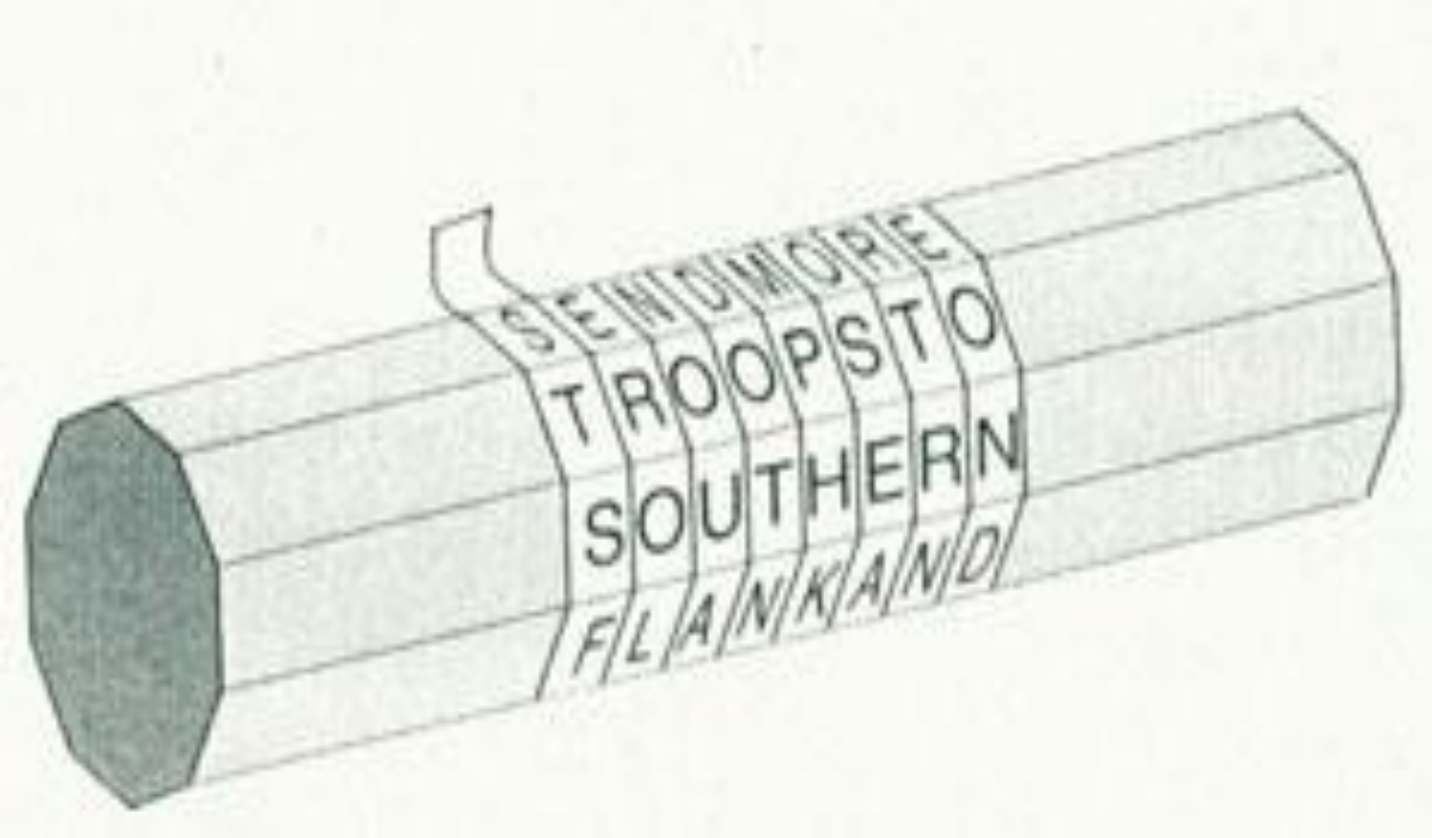
		1	2	3	4	5	6
b	**1**	*S*	*e*	*c*	*u*	*r*	*i*
u	**4**	*t*	*a*	*t*	*e*	*a*	*e*
t	**3**	*s*	*t*	*e*	*a*	*s*	*i*
u	**5**	*g*	*u*	*r*	*a*	*t*	*ă*
c	**2**	*x*	*y*	*z*	*t*	*w*	*u*

	1	2	3	4	5	6
1	*S*	*e*	*c*	*u*	*r*	*i*
2	*x*	*y*	*z*	*t*	*w*	*u*
3	*s*	*t*	*e*	*a*	*s*	*i*
4	*t*	*a*	*t*	*e*	*a*	*e*
5	*g*	*u*	*r*	*a*	*t*	*ă*

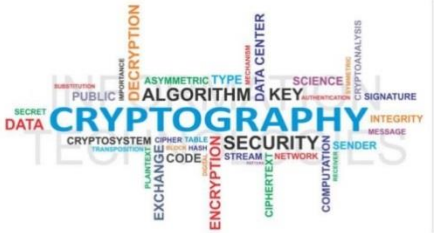
Encryption tools



- Scytale

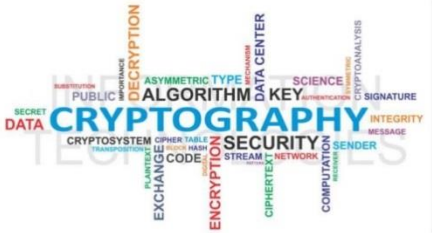


Encryption tools



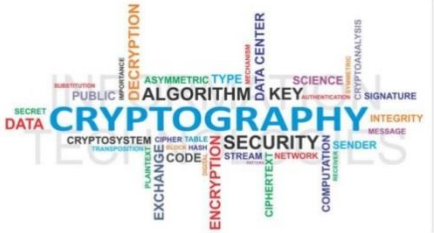
- The Alberti Cipher disk

Encryption tools



- *Jefferson disk*

https://www.youtube.com/watch?v=ybkkiGtJmkM



Encryption tools

- *Enigma machine*

About
158 962 555 217 826 350 000
different combinations

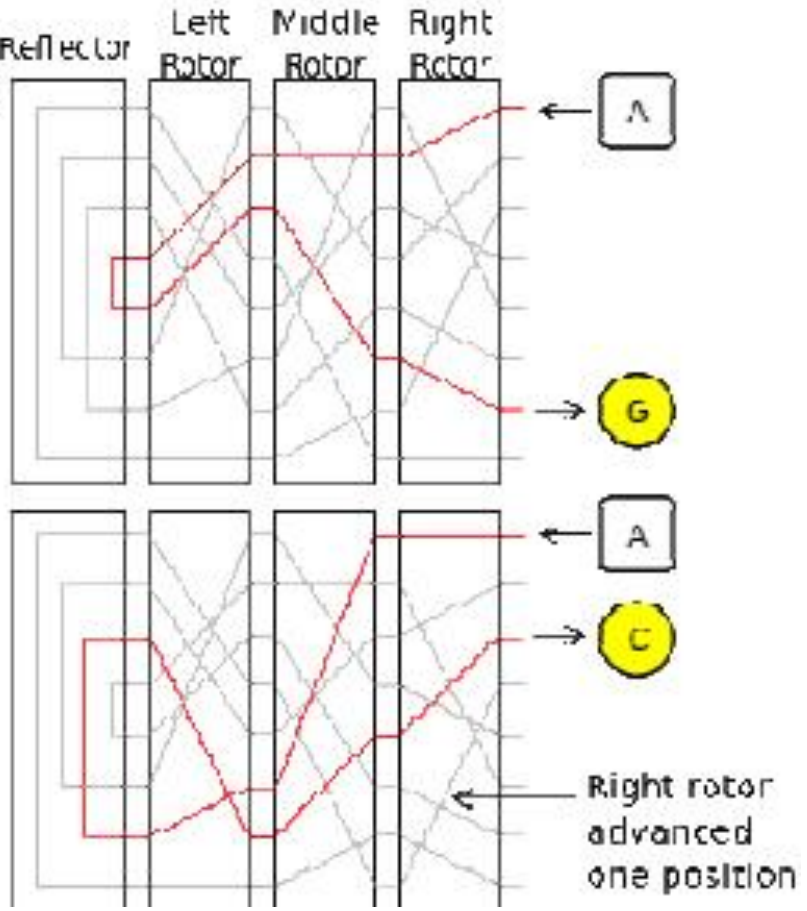
https://www.youtube.com/watch?v=ybkkiGtJmkM

Enigma machine

- Invented in Germany in 1920 and then improved and used by the military in WWII
- Five rotors and the plugboard provided 1.6×10^20 settings
- Poly-alphabetic substitution cipher
- Initial setting of rotors and plugboard changed everyday
- Product of permutations

$$PR_1R_2R_3FR_3^{-1}R_2^{-1}R_1^{-1}P^{-1}$$

- The Enigma code was broken by a team headed by Alan Turing




DECRYPTION
ASYMMETRIC TYPE
PUBLIC
ALGORITHM
KEY
SCIENCE
DATA CENTER
SIGNATURE
SECRET
DATA
CRYPTOGRAPHY
INTEGRITY
MESSAGE
CRYPTOSYSTEM
CIPHER
SECURITY
SENDER
EXCHANGE
CODE
STREAM
NETWORK
ENCRYPTION
CIPHERTEXT
COMPUTATION
CRYPTOANALYSIS

Questions???

Thank You!

Cryptography & Security

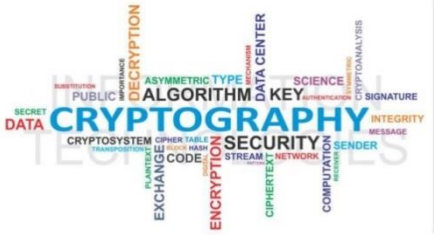
3. Stream ciphers

Aureliu Zgureanu,
PhD, Associate Professor

Content of this part

- Intro to stream ciphers
- Random number generators (RNGs)
- One-Time Pad (OTP)
- Linear feedback shift registers (LFSRs)
- Trivium: a modern stream cipher

Classification Of Ciphers

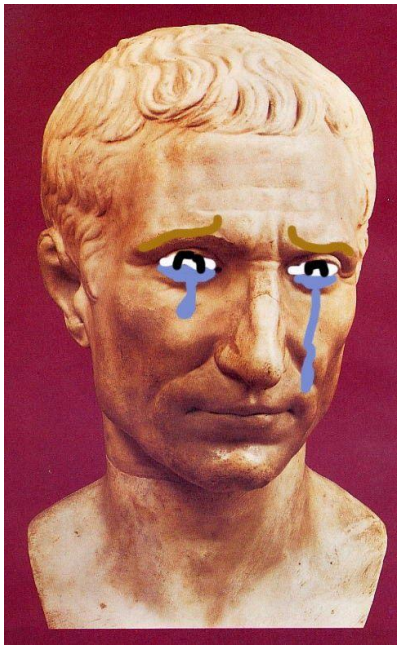


- Cryptographic Ciphers
  - Classical Ciphers
    - Substitution Ciphers
      - Single Substitution
        - Monoalphabetical Ciphers
          - Atbash Cipher
            - Caesar Cipher
              - Affine Cipher
      - Multiple Substituion
        - Polyalphabetical Ciphers
          - Vigenere Cipher
            - Gronsfeld Cipher
              - Beaufort Cipher
                - Auto Key Cipher
                  - Running Key Cipher
        - Polygraphic Ciphers
          - Hill Cipher
          - Columnar Cipher
            - Double Transpositional Cipher
              - Myszkowiski Cipher
                - Disrupted Cipher
                  - Grilles Cipher
    - Transpositional Ciphers
      - Rail Fence Cipher
      - Route Cipher
  - Modern Ciphers
    - Secret Cryptography
      - Block Ciphers
        - IDEA
        - DES
        - 3DES
        - AES
      - Stream Cipher
        - Synchronize
        - Self- Synchronize
    - Public Cryptography
      - Integer Factorization Cryptosystems
        - RSA
          - Rabin
            - Schmidt-Samoa
      - Discrete Log Cryptosystems
        - ElGamal
          - Cramer-Shoup
            - Massey-Omura
      - Elliptic Curves Cryptosystems
        - ElGamal
    - Hash Functions

Are not ciphers!

Recap

- Caesar Cipher, Shift Cipher, Substitution Cipher, Vigenere Cipher
- All historical ciphers have fallen



Private Key Encryption Syntax

- Message Space: $\mathcal{M}$
- Key Space: $\mathcal{K}$
- Three Algorithms
  - $\mathrm{Gen}(R)$ (Key-generation algorithm)
    - Input: Random Bits R
    - Output: Secret key $\mathrm{k} \in \mathcal{K}$
  - $\mathrm{Enc}_{\mathrm{k}}(m)$ (Encryption algorithm)
    - Input: Secret key $\mathrm{k} \in \mathcal{K}$ and message $\mathrm{m} \in \mathcal{M}$
    - Output: ciphertext *c*
  - $\mathrm{Dec}_{\mathrm{k}}(c)$ (Decryption algorithm)
    - Input: Secret key $\mathrm{k} \in \mathcal{K}$ and a ciphertex c
    - Output: a plaintext message $\mathrm{m} \in \mathcal{M}$

Typically picks $\mathrm{k} \in \mathcal{K}$ uniformly at random

Trusted Parties (e.g., Alice and Bob) must run Gen in advance to obtain secret k.

Assumption: Adversary does not get to see output of Gen

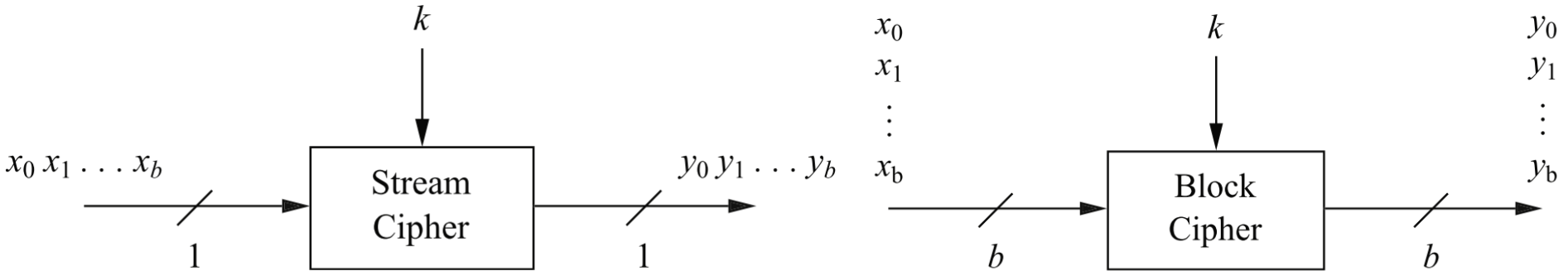
- Invariant: $\mathrm{Dec}_{\mathrm{k}}(\mathrm{Enc}_{\mathrm{k}}(\mathrm{m}))=\mathrm{m}$

Shannon's Theorem

Theorem: Let (Gen,Enc,Dec) be an encryption scheme with $|\mathcal{K}| = |\mathcal{M}| = |\mathcal{C}|$. Then the scheme is perfectly secret if and only if:

1. Every key k ∈ $\mathcal{K}$ is chosen with (equal) probability $1/|\mathcal{K}|$ by the algorithm Gen, and
2. For every m ∈ $\mathcal{M}$ and every c ∈ $\mathcal{C}$ there exists a unique key k ∈ $\mathcal{K}$ such that $Enc_k(m)=c$.

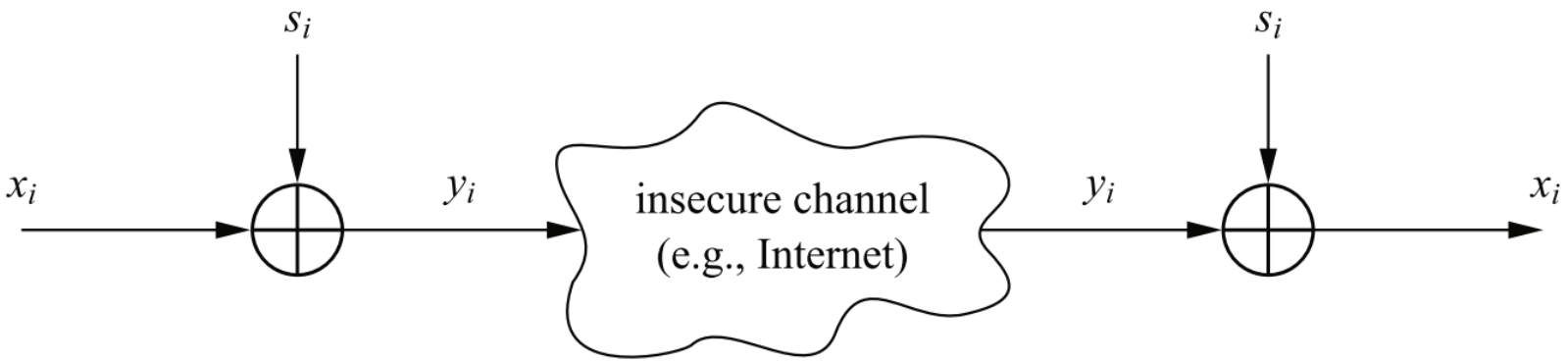
Stream Cipher vs. Block Cipher



- **Stream Ciphers**
  - Encrypt bits individually
  - Usually small and fast → common in embedded devices (e.g., A5/1 for GSM phones)
- **Block Ciphers:**
  - Always encrypt a full block (several bits)
  - Are common for Internet applications

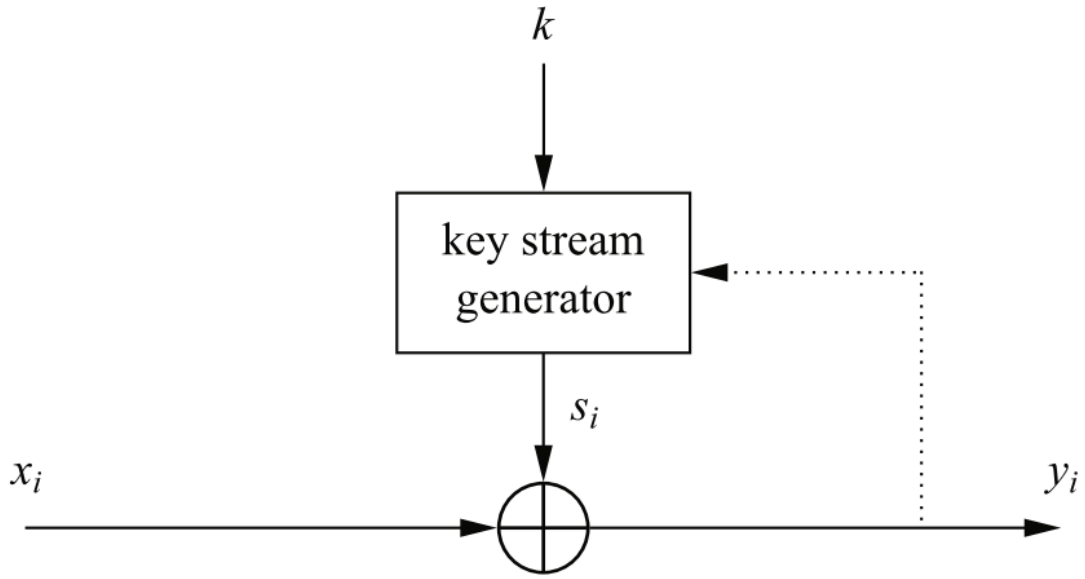
Encryption/decryption with Stream Ciphers

Plaintext x_i, ciphertext y_i and key stream s_i consist of individual bits



- Encryption and decryption are simple additions modulo 2 (XOR)
- Encryption and decryption are the same functions
- Encryption: $y_i = e_{s_i}(x_i) = x_i + s_i \bmod 2$
  $x_i , y_i , s_i \in \{0,1\}$
- Decryption: $x_i = e_{s_i}(y_i) = y_i + s_i \bmod 2$

Synchronous vs. Asynchronous Stream Cipher



- Security of stream cipher depends entirely on the key stream s_i:
  - Should be **random**, i.e., $Pr(s_i = 0) = Pr(s_i = 1) = 0.5$
  - Must be **reproducible** by sender and receiver
- **Synchronous Stream Cipher**
  - Key stream depends only on the key (and possibly an initialization vector IV)
- **Asynchronous Stream Ciphers**
  - Key stream depends also on the ciphertext (feedback enabled)

Why is Modulo 2 Addition a Good Encryption Function?

- For perfectly random key stream s_i , each ciphertext output bit has a 50% chance to be 0 or 1

  → Good statistical property for ciphertext

- Inverting XOR is simple, since it is the same XOR operation

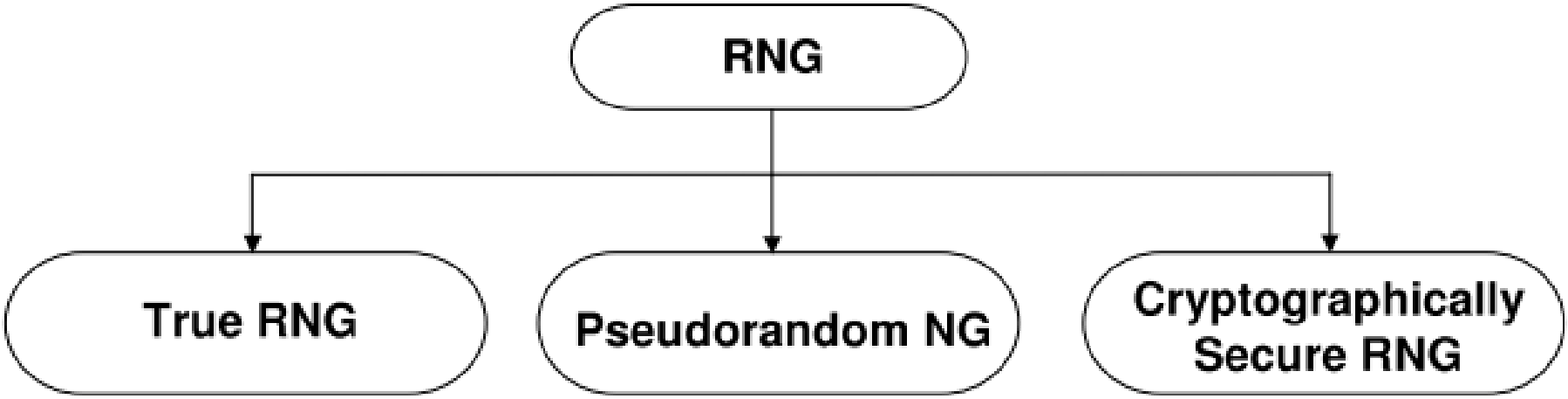
x_i	s_i	y_i
0	0	0
0	1	1
1	0	1
1	1	0

Stream Cipher: Throughput

- Intel® Core™ i7-2600 (3.40 GHz) processor
- Intel® Q65 Express 4 GB 1333 MHz DDR3 (RAM)
- Ubuntu 12.04 LTS operating System

File Type	Size (In MB)	Encryption Time in Millisecond						
		AES	DES	3-DES	RC2	Blowfish	Skipjack	RC4
		128	56	112	40	32	80	40
BMP	10.7	101	272	788	238	133	381	40
	50	455	1253	3804	1095	614	1729	198
	100	909	2595	7628	2189	1223	3505	372
FLV	50	456	1268	3810	1112	629	1731	196
	100	918	2586	7631	2224	1267	3515	360
	482	4518	12529	35654	11038	6087	16941	1972

Random number generators (RNGs)



True Random Number Generators (TRNGs)

- Based on physical random processes: coin flipping, dice rolling, semiconductor noise, radioactive decay, clock jitter of digital circuits
- Output stream s_i should have good statistical properties: $Pr(s_i = 0) = Pr(s_i = 1) = 50\%$ (often achieved by post-processing)
- Output can neither be predicted nor be reproduced

Typically used for generation of keys, nonces (used only-once values) and for many other purposes

Pseudo-random Number Generator (PRNG)

- Generate sequences from initial seed value
- Typically, output stream has good statistical properties
- Output can be reproduced and can be predicted

Often computed in a recursive way:

$$s_0 = seed$$

$$s_{i+1} = f(s_i, s_{i-1}, \ldots, s_{i-t})$$

Example: *rand() function* in ANSI C:

$$s_0 = 12345$$

$$s_{i+1} = 1103515245 s_i + 12345 \bmod 2^{31}$$

Most PRNGs have poor cryptographic properties

Cryptanalyzing a Simple PRNG

Simple PRNG: **Linear Congruential Generator**

$$S_0 = seed$$

$$S_{i+1} = AS_i + B \bmod m$$

Assume

- **unknown *A*, *B* and S_0 as key**
- **Size of *A*, *B* and S_i = 100 bit**
- **300 bit of output are known, i.e., S_1, S_2 and S_3**

Solving

$$S_2 = AS_1 + B \bmod m$$

$$S_3 = AS_2 + B \bmod m$$

directly reveals A and B. All S_i can be computed easily

Bad cryptographic properties due to the linearity of most PRNGs

Cryptographically Secure Pseudorandom Number Generator (CSPRNG)

- Special PRNG with additional property:
  - Output must be unpredictable

More precisely: Given n consecutive bits of output s_i the following output bits s_{n+1} cannot be predicted (in polynomial time)

- Needed in cryptography, in particular for stream ciphers

- Remark: There are almost no other applications that need unpredictability, whereas many (technical) applications need PRNGs

One-Time Pad (OTP)

Invented by
Frank Miller, 1882

Patented by
Gilbert Vernam, 1917

Unconditionally secure cryptosystem:

- **A cryptosystem is unconditionally secure if it cannot be broken even with *infinite* computational resources**
- **One-Time Pad**
  - **A cryptosystem developed by Mauborgne that is based on Vernam's stream cipher**
- **Properties:**

  Let the plaintext, ciphertext and key consist of individual bits $x_i, y_i, k_i \in \{0,1\}$.

Encryption: $e_{k_i}(x_i) = x_i \oplus k_i$.
Decryption: $d_{k_i}(y_i) = y_i \oplus k_i$

OTP is unconditionally secure if and only if the key k_i is only used once

One-Time Pad (OTP)

Unconditionally secure cryptosystem:

$$y_0 = x_0 \oplus k_0$$
$$y_1 = x_1 \oplus k_1$$
$$\vdots$$

Every equation is a linear equation with two unknowns

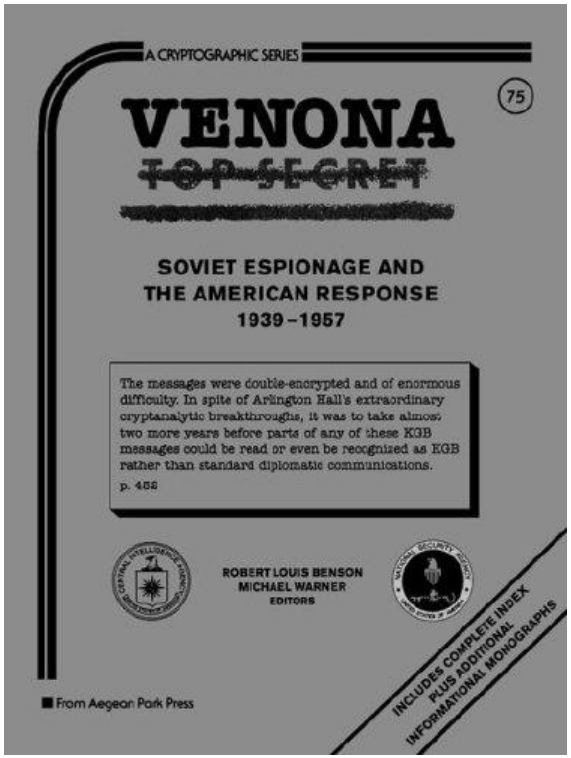
$\Rightarrow$ for every y_i : $x_i = 0$ and $x_i = 1$ are equiprobable

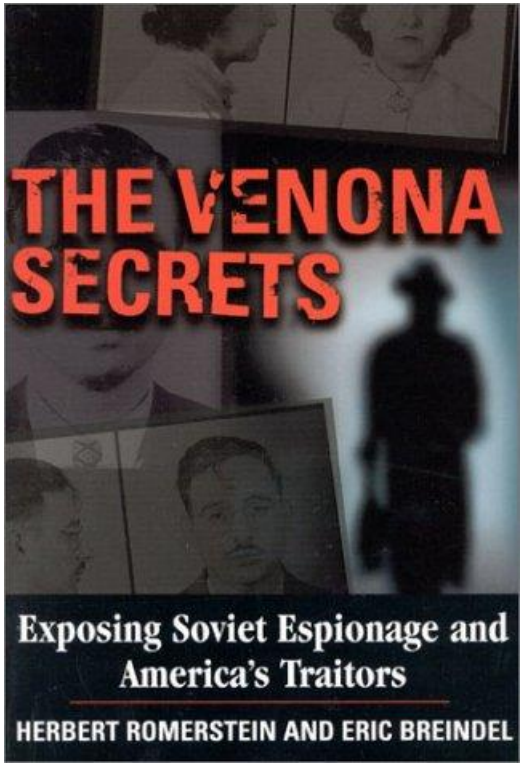
$\Rightarrow$ This is true iff $k_0, k_1, \ldots$ are independent, i.e., all k_i have to be generated truly random

$\Rightarrow$ It can be shown that this systems can *provably* not be solved.

Disadvantage: For almost all applications the OTP is impractical since the key must be as long as the message! (Imagine you have to encrypt a 1GByte email attachment.)

VENONA project





VENONA project (US + UK)

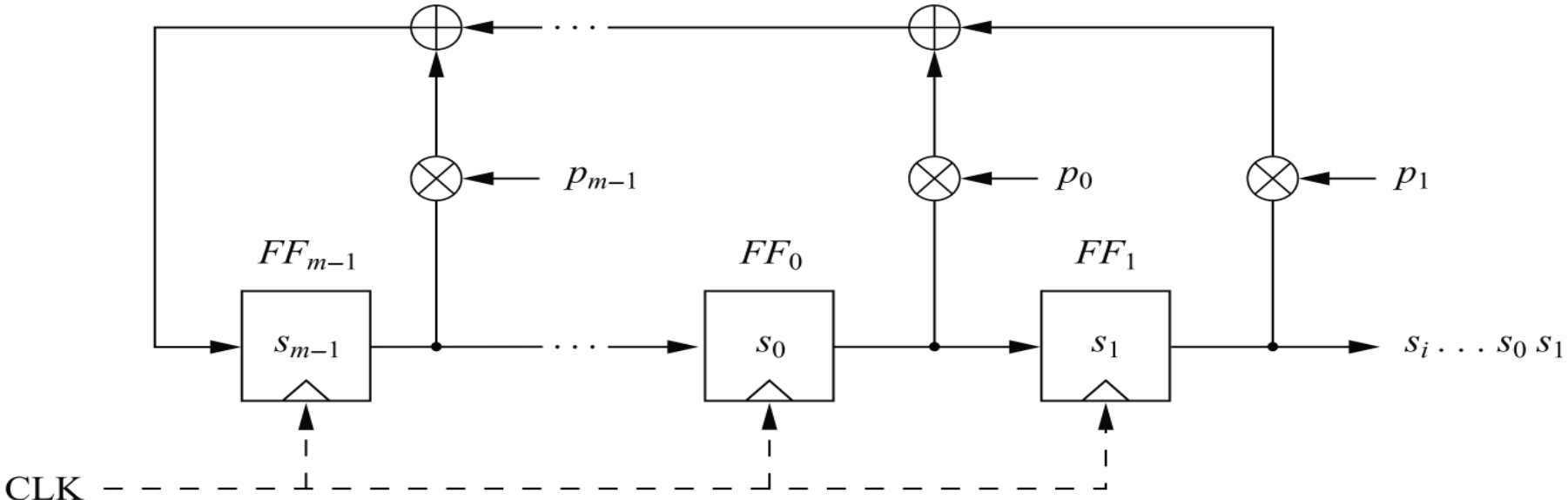
Decrypt ciphertexts sent by Soviet Union which were mistakenly encrypted with portions of the same one-time pad over several decades

$$c \oplus c' = (m \oplus k) \oplus (m' \oplus k) = m \oplus m'$$

One-Time Pad (OTP)

Cold War

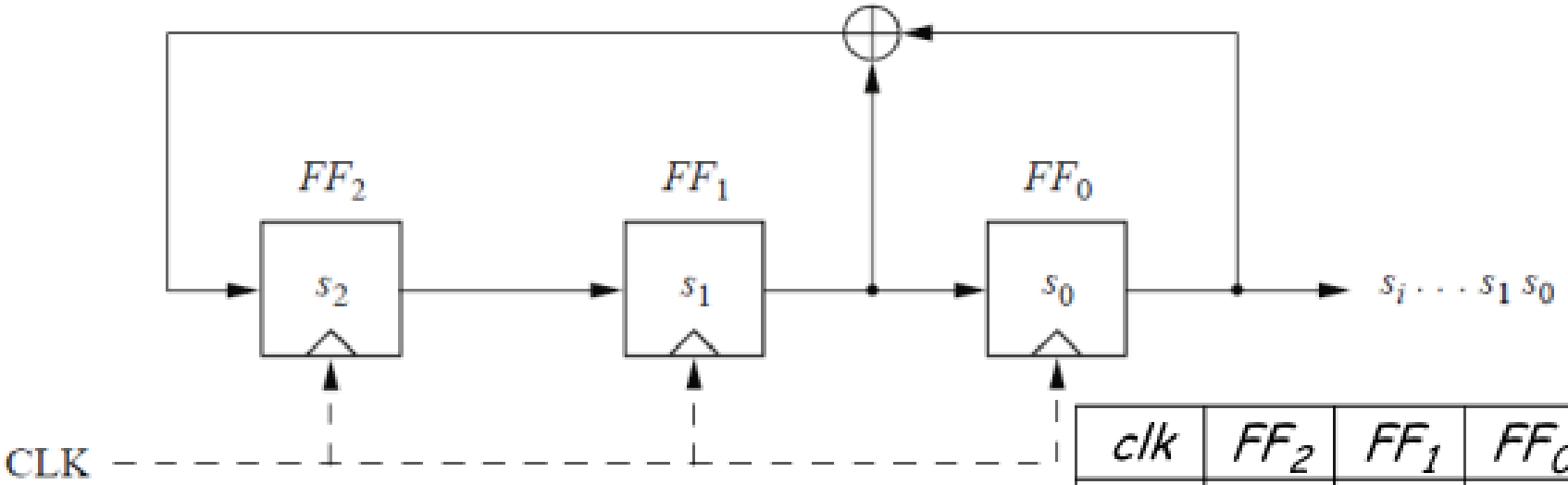
Linear Feedback Shift Registers (LFSRs)



$$s_m \equiv s_{m-1}p_{m-1} + \cdots + s_1 p_1 + s_0 p_0 \bmod 2$$

- A shift register with a feedback path computing new value by XOR of certain state bits
- *Degree m* given by number of storage elements
- If $p_i = 1$, the feedback connection is present ("closed" switch), otherwise there is no feedback
- Output sequence repeats periodically
- Maximum output length: $2^m - 1$

LFSR: Example with m=3



$$s_3 \equiv s_1 + s_0 \bmod 2$$

- LFSR output described by recursive equation:

$$s_{i+3} = s_{i+1} + s_i \bmod 2$$

- Maximum output length (of $2^3-1=7$) achieved only for certain feedback configurations, e.g., the one shown here.

clk	FF_2	FF_1	$FF_0=s_i$
0	1	0	0
1	0	1	0
2	1	0	1
3	1	1	0
4	1	1	1
5	0	1	1
6	0	0	1
7	1	0	0
8	0	1	0

Polynomial representation of LFSRs

LFSRs typically described by polynomials:

$$P(X) = X^m + p_{m-1}X^{m-1} + \ldots + p_1 X + p_0$$

Degree 3 LFSR from previous slide: $P(X) = X^3 + X + 1$

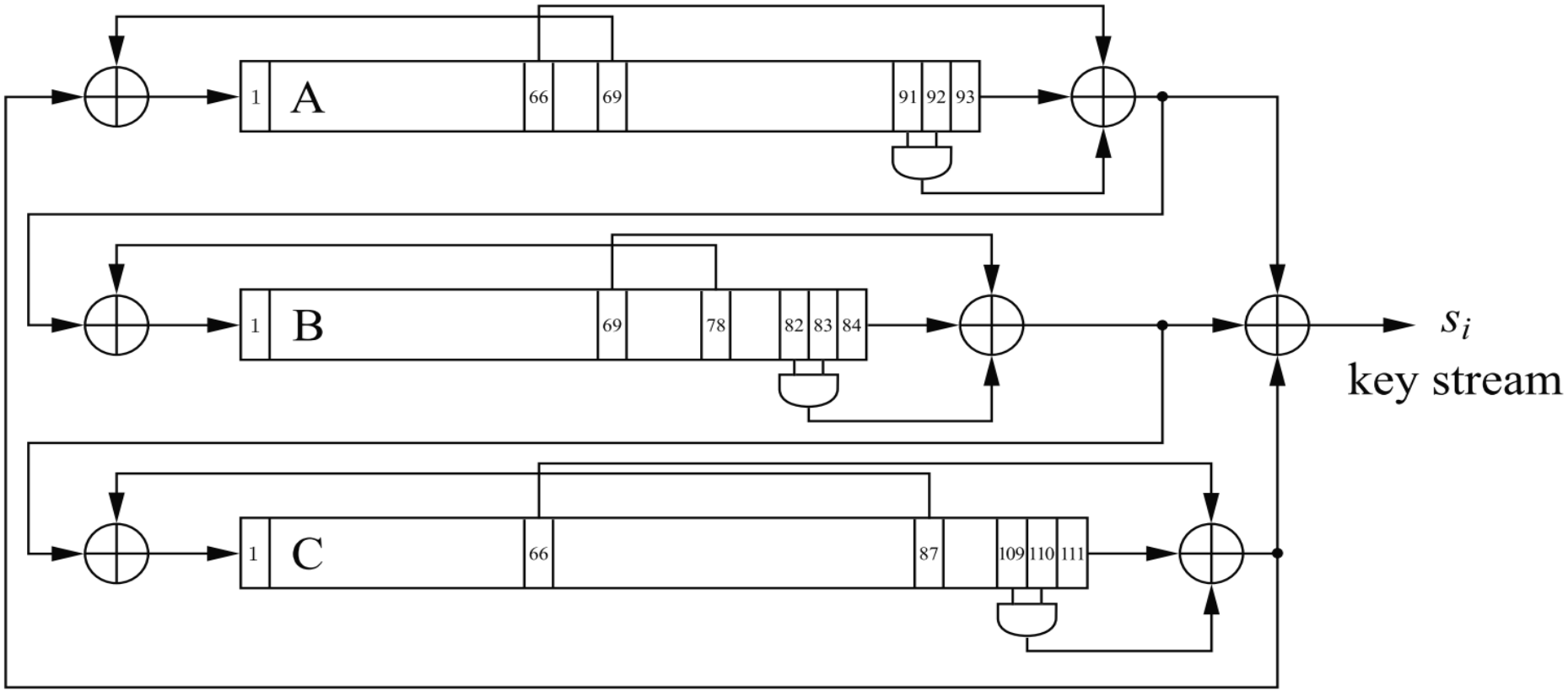
- Not all LFSR generate maximum length sequence
- Example: m=3, $p_2 = p_1 = p_0$:
- LFSR has maximum length *iff* the polynomial P(x) is a primitive polynomial, special kind of irreducible polynomial – divisible only by itself and 1
- A list of known primitive polynomials is available

clk	FF_2	FF_1	FF_0
0	1	0	0
1	1	1	0
2	0	1	1
3	0	0	1
4	1	0	0
5	1	1	0
6	0	1	1

Security of LFSRs

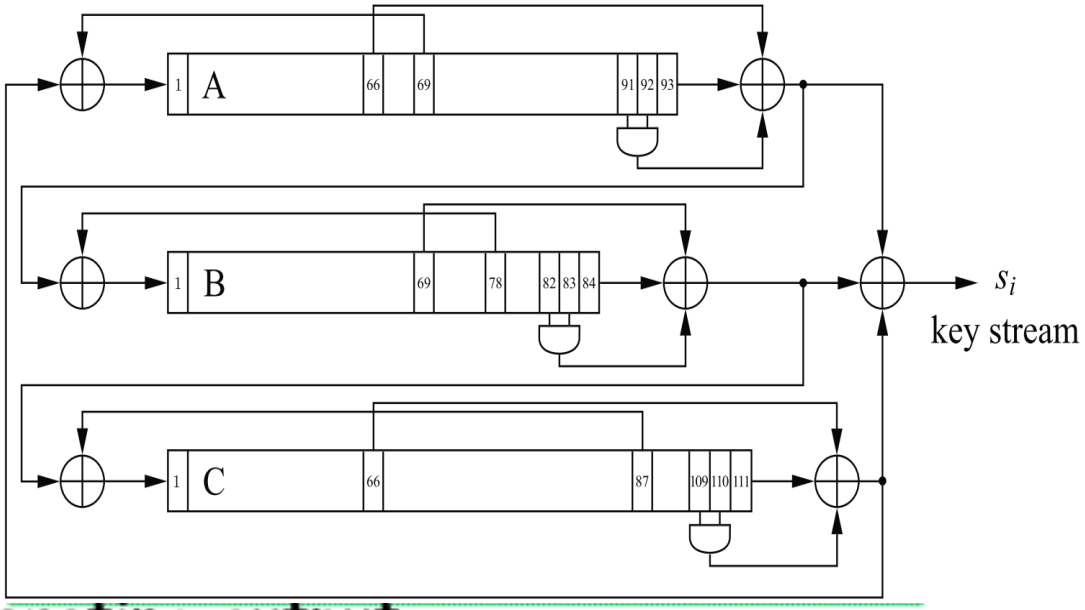
- If an LFSR is used to generate the stream of bits s_i so that $y_i = x_i + s_i \bmod 2 = x_i \oplus s_i$
  - The coefficients p_i can constitute the secret key
- If $2m$ output bits of an LFSR of degree m are known, the feedback coefficients p_i of the LFSR can be found by solving a system of linear equations
  - Assume Oscar has the plaintext x_i and the ciphertext y_i and knows the degree m (not necessary)
  - $s_i = x_i \oplus y_i$ and for m,m+1,…,2m $s_m \equiv p_{m-1}s_{m-1} + \ldots + p_1 s_1 + p_0 s_0 \mod 2$
- Because of this many stream ciphers use combinations of LFSRs
  - A5/1 for GSM phones uses 3 LFSRs

A Modern Stream Cipher - Trivium



- Three *nonlinear* FSRs (NLFSR) of length 93, 84, 111
- XOR-Sum of all three NLFSR outputs generates key stream s_i
- Small hardware:
  - Total FF count: 288; Non-linearity: 3 AND Gates
  - 7 XOR Gates (4 with three inputs)

Trivium



Initialization:

- Load 80-bit IV into A
- Load 80-bit key into B
- $c_{109}, c_{110}, c_{111}$=1, other=0

Warm-Up: Clock 4x288 =1152 times without generating output

IV: a randomizer with a new value for every encryption (nonces – values used only once); no need to keep secret

Without IV the sequence S_i will be deterministic

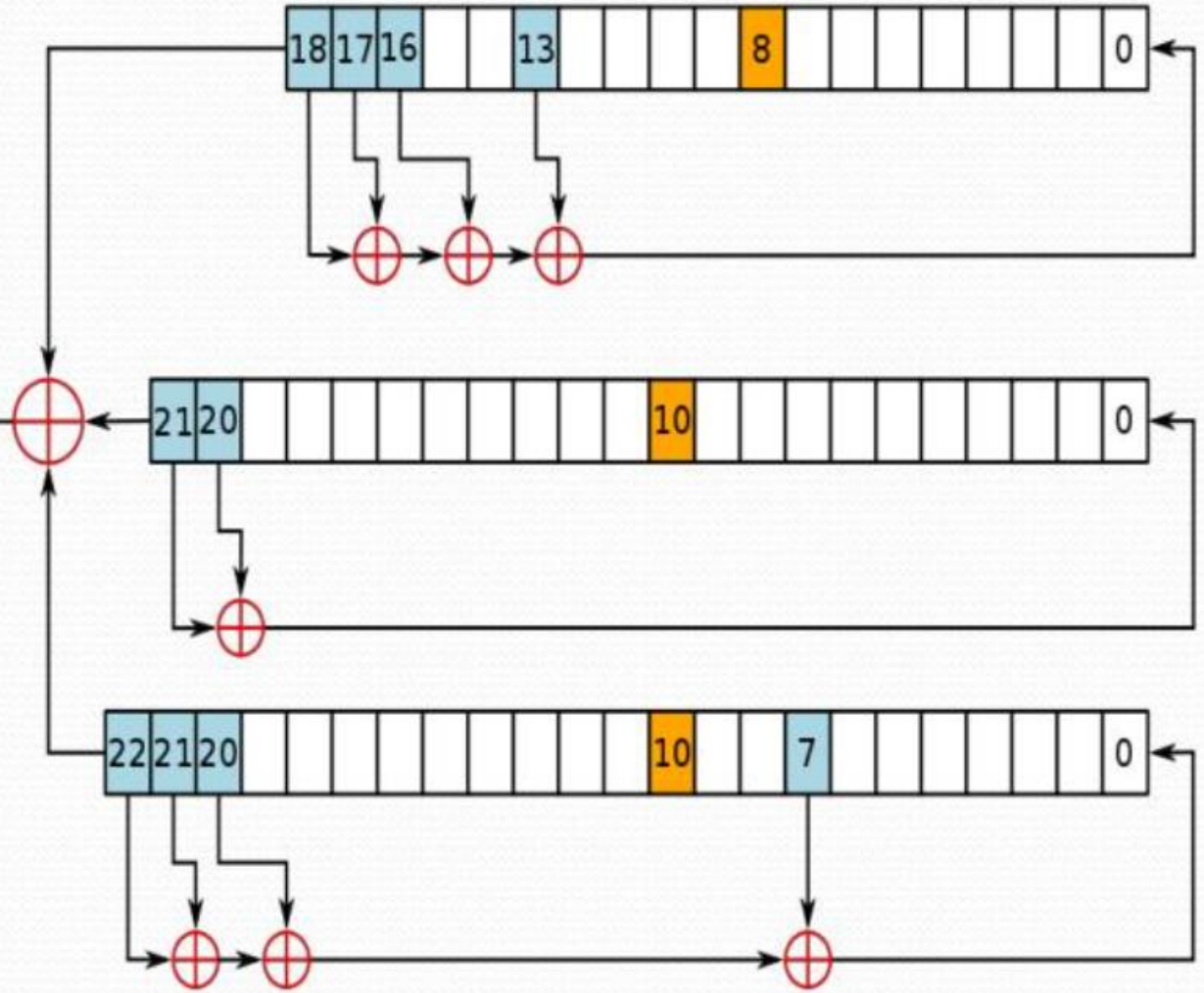
It is yet unknown how secure is Trivium

	Register length	Feedback bit	Feedforward bit	AND inputs
A	93	69	66	91, 92
B	84	78	69	82, 83
C	111	87	66	109, 110

A5/1 Stream Cipher

Consists of 3 LFSRs of different lengths

- 19 bits
  - $x^{18} + x^{17} + x^{16} + x^{13} + 1$
  - clock bit 8
  - tapped bits: 13, 16, 17, 18
- 22 bits
  - $x^{21} + x^{20} + 1$
  - clock bit 10
  - tapped bits 20, 21
- 23 bits
  - $x^{22} + x^{21} + x^{20} + x^{7} + 1$
  - clock bit 10
  - tapped bits 7, 20, 21, 22



A register is clocked if its Clocking bit is equal to at least another clocking bit

Lessons Learned

- Stream ciphers are less popular than block ciphers in most domains such as Internet security. There are exceptions, for instance, the popular stream cipher RC4.

- Stream ciphers sometimes require fewer resources, e.g., code size or chip area, for implementation than block ciphers, and they are attractive for use in constrained environments such as cell phones.

- The requirements for a *cryptographically secure pseudorandom number generator* are far more demanding than the requirements for pseudorandom number generators used in other applications such as testing or simulation

- The One-Time Pad is a provable secure symmetric cipher. However, it is highly impractical for most applications because the key length has to equal the message length.

- Single LFSRs make poor stream ciphers despite their good statistical properties. However, careful combinations of several LFSR can yield strong ciphers.

Thank You!


DECRYPTION
ASYMMETRIC TYPE
PUBLIC
ALGORITHM
KEY
SCIENCE
DATA CENTER
SIGNATURE
SECRET
DATA
CRYPTOGRAPHY
INTEGRITY
MESSAGE
CRYPTOSYSTEM
CIPHER
SECURITY
SENDER
EXCHANGE
CODE
STREAM
NETWORK
ENCRYPTION
CIPHERTEXT
COMPUTATION
CRYPTOANALYSIS

Questions???