Belarusian State Technological University
Department of Information Systems and Technology

Pavel Urbanovich

INFORMATION PROTECTION.

Part 6: ELEMENTS OF CRYPTOSYSTEMS AND MATHEMATICAL FOUNDATIONS OF CRYPTOGRAPHY

pav.urb@yandex.by, p.urbanovich@belstu.by

Cryptography. Basic Concepts

Cryptography is closely related to the disciplines of **cryptology** and **cryptanalysis**.

•The history of **cryptography** is long and goes back at least 4,000 years to the Egyptians, who used hieroglyphic codes for inscription on tombs.
•Since then many cryptosystems, also called ciphers, have been developed and used.

•Meaning of cryptography:
Cryptography is the field that offers techniques and methods of managing secrets.
Cryptography is about communication in the presence of an adversary

Cryptography is a method of protecting information and communications through the use of codes so that only those for whom the information is intended can read and process it.
The pre-fix "*crypt*" means "*hidden*" or "*vault*"
and the suffix "*graphy*" stands for "*writing*."

It encompasses many problems:

- encryption,
- authentication,
- key distribution to name a few.

The field of modern cryptography provides a theoretical foundation based on which we may understand what exactly these problems are, how to evaluate protocols that purport to solve them, and how to build protocols in whose security we can have confidence.

The most ancient and basic problem of cryptography is secure communication over an insecure channel.

A rigorous theory of perfect secrecy based on information theory was developed by C. Shannon in 1943 (see C. E. Shannon. A mathematical theory of communication. Bell Sys. Tech. J., 27:623–656, 1948.)

In this theory, the adversary is assumed to have unlimited computational resources.

Cryptography. Basic Concepts

Purposes of Cryptography:

- to preserve **confidentiality**,
- to **authenticate** senders and receivers of messages,
- to facilitate message **integrity**,
- to ensure that the sender will not be able to deny transfer of message (**non-repudiation**).

There are **two main goals in cryptography**:

Secrecy and **Authentication**

• secrecy is where one makes sure that only the legitimate parties can read some particular information
and
• authentication is where one can verify the authenticity of some data or the authenticity of an entity, which could be a person or a computer.

Cryptography and Information Protection

Basic Definitions

Definition 1. Cryptography (from gr.κρςπτός – cryptos or "confidential, hidden" and γράφω – gráfo or "write") is a field of mathematics (mathematical techniques) related to information security aspects such as confidentiality, authentication, integrity (indivisibility).

Definition 2. Confidentiality - is to share information only with those who are authorized to have access to it. There are many ways to ensure confidentiality, from physical protection to mathematical algorithms that make data incomprehensible (for uninitiated).

Definition 3. Authentication - is the process of determining whether someone or something is, in fact, who or what it is declared to be. Logically, authentication precedes **authorization** (although they may often seem to be combined). The two terms are often used synonymously but they are two different processes.

Definition 4. Authorization is the function of specifying access rights/privileges to resources related to information security and computer security in general and to access control in particular.

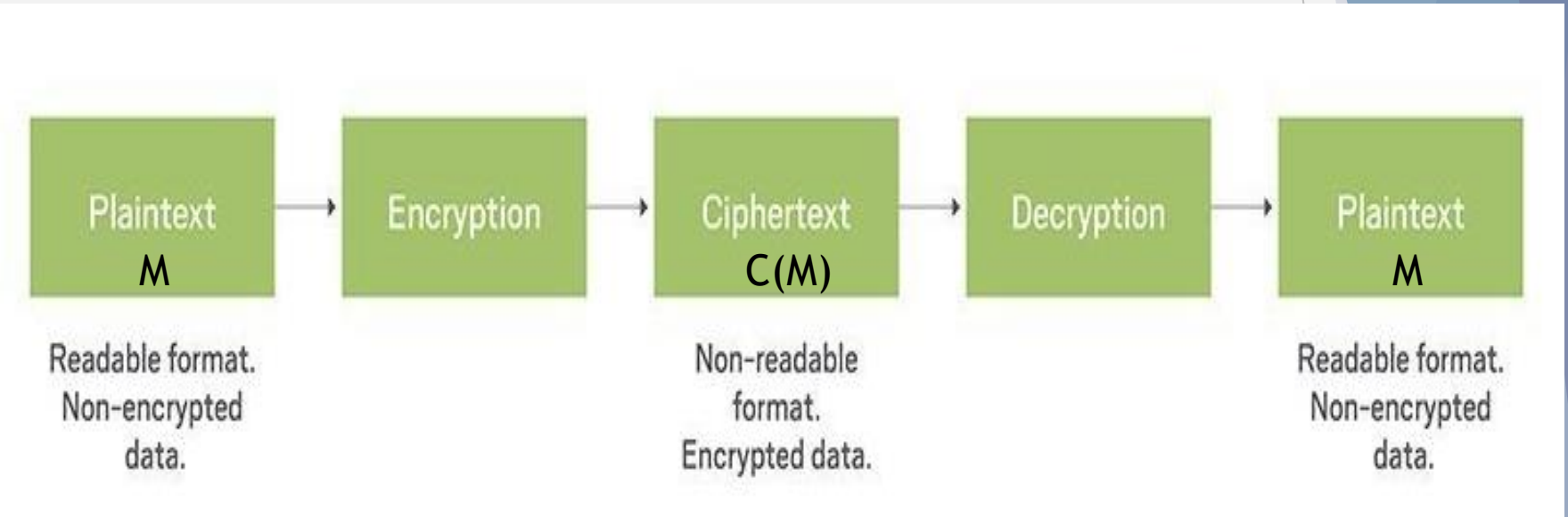
Definition 5. Cryptology (from κρςπτός –cryptos -"hidden" and λόγοσ -logos -"word") is a science covering **cryptography** and **cryptanalysis**.

Definition 6. Cryptanalysis is a science of mathematical techniques dealing with overcoming of cryptographic methods of information protection.

Definition7. A cryptanalyst is a person who deals with cryptanalysis.

Definition 8. Cryptosystem is a concept related to the collection of cryptographic algorithms used to protect information.

Cryptosystem



Encryption and Decryption

Definition 9. ***M*** denotes a **set** of the messages.
This set consists of strings of symbols of the alphabet (they can be binary strings, text in polish, computer code, etc.).
An element $\mathbf{m}_i$ of ***M*** is called a plaintext:

$$M=m_1, m_2, \ldots, m_n$$

Definition 10. ***C*** is a set of cryptograms.
It consists of alphabet symbols, which may, however, differ from the alphabet for set **M**.
An element $\mathbf{c}_i$ set of the set ***C*** is called a ciphertext:

$$C=c_1, c_2, \ldots, c_n$$

Definition 11. **K** is a **set** of keys (an element of this set is the key). Each element $e \in K$ uniquely defines the bijection of the set M to the set C (we denot it as E_e):

$$E_e : M \rightarrow C.$$

E_e is an encrypted function that must be a *bijection* – to make the process reversible.

Definition 12. For every $d \in K$, D_d denotes the bijection of the set C to the set M, called the decryption function:

$$D_d : C \rightarrow M$$

Definition 13. A process of calculating of the value of function E_e for the argument $m \in M$ is called the encryption of plaintext m.

Definition 14. Similarly, a process of calculating of the value of function D_d for the argument $c \in C$ is called the decryption of ciphertext c.

Definition 15. An encryption scheme consists of the set $\{E_e : e \in K\}$ of the encrypted function and the corresponding to E_e set $\{D_d : d \in K\}$ of decryption functions with the property that for every key $d \in K$ there exists a unique key $d \in K$, such that $D_d = (E^{-1})_e$.

So $D_d(E_e(m))=m$
for any $m \in M$.

Cryptosystem: $CS = (M, K, C, E, D)$

We use the following notation:

- M is the set of messages or plaintexts
- K is the set of keys
- C is the set of ciphertexts
- $E_k(m)$ is the encryption of plaintext $m \in M$ using the key $k \in K$
- $D_k(c)$ is the decryption of ciphertext $c \in C$ using the key $k \in K$

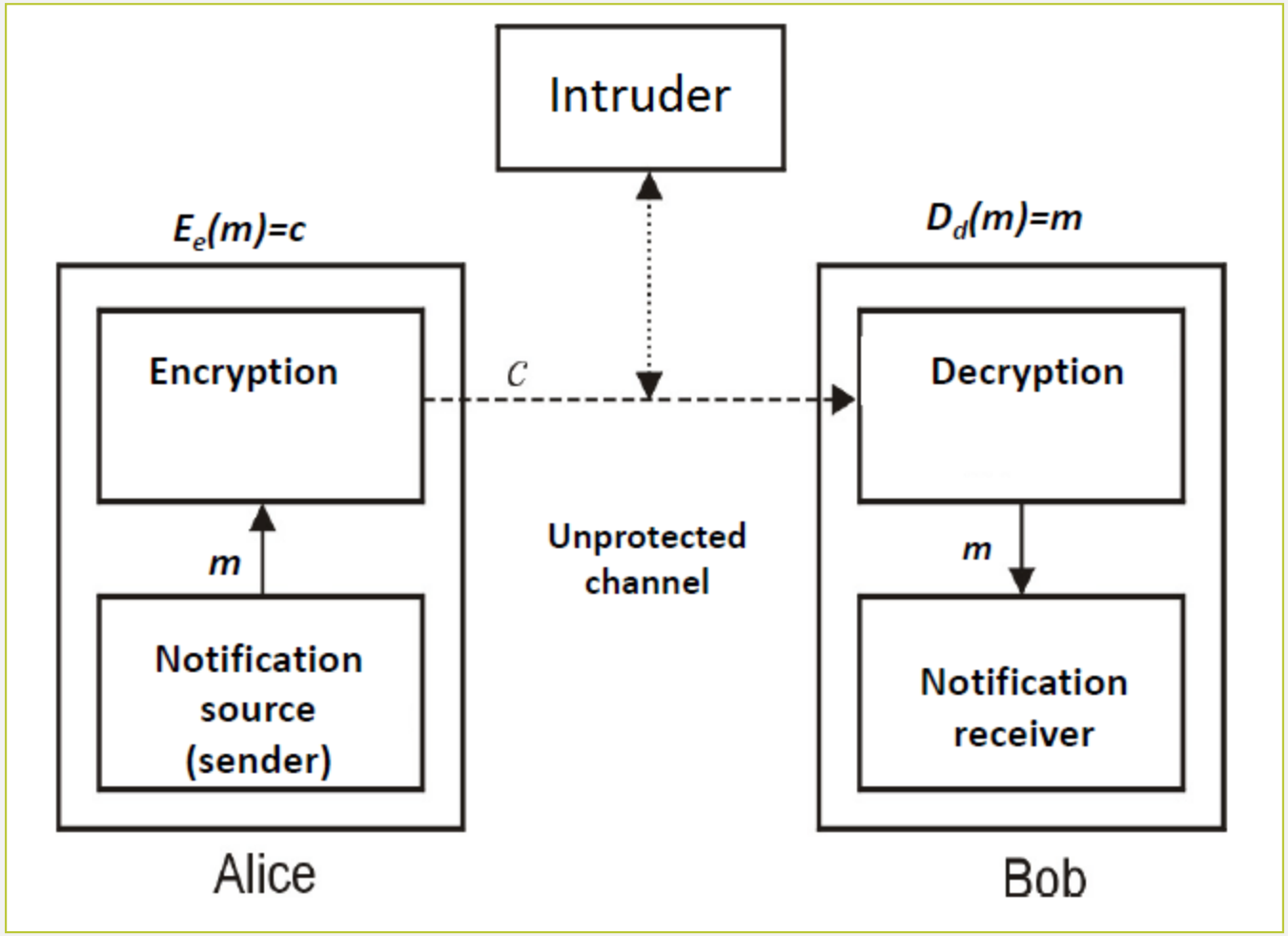


Fig. 1. Diagram of communication between two parties with the use of cryptography

Communication Participants

- A **Subject** (party) is someone (or something) who (or what) sends, receives or manipulates information. It can be person, terminal, etc.;
- A **Sender** (Notification source) is a **Subject** that is the actual transmitter of the information;
- A **Receiver** (Notification receiver) is an **Subject** which is intended for the information of a sender;
- An **Intruder** (intruz) is an subject that is neither a sender nor a receiver who is trying to defeat the security of information transmitted between a sender and a receiver, under which often masquerades;
- An **Unprotected channel** is a channel in which information can be manipulated by any subject that is neither a sender nor a receiver;
- A **Protected channel** is a channel where an intruder is unable to manipulate the transmitted information. This channel can be protected both physically and by cryptographic methods.

Classification of Cryptographic Systems

I.

a) **Substitution Cipher** - replace units of plaintext with units of ciphertext (a simple example is the **Caesar** cipher, the **Vigenère** cipher is a simple example which substitutes plaintext values for ciphertext values using a series of Caesar ciphers which are defined by a keyword),

b) **Permutation Cipher** - attempt to hide information from an adversary by rearranging the plaintext so that it can no longer be recognized.

Example.

In the **Rail Fence Cipher** the plain text is written downwards and diagonally on successive "rails" of an imaginary fence, then moving up when we reach the bottom rail.
When we reach the top rail, the message is written downwards again until the whole plaintext is written out.
The message is then read off in rows.

For example, if we have 3 "rails" and a message (***M***) of 'Informatics', the cipherer writes out:

I --O --A --C
-N --R --T --S
--F --M -I

Then reads off to get ciphertext: ***C***= 'IOACNRTSFMI'

II.

a) **Block Cipher** is a deterministic algorithm operating on fixed-length (***k***) groups of bits, called a ***block***, with an unvarying transformation that is specified by a key.
Length ***k*** can be from a few dozen to a few thousand of bits.

A classic example is a block algorithm **DES** (for more details about it - below).

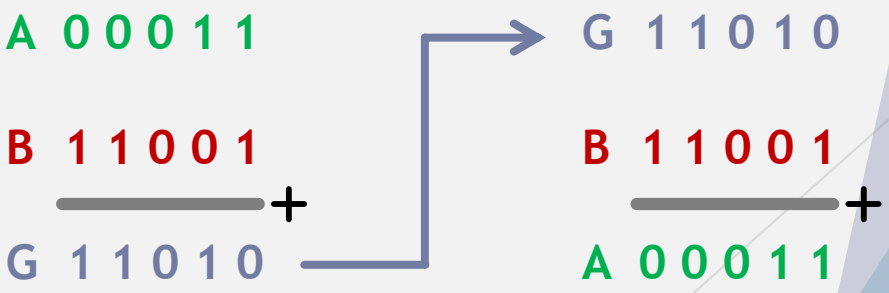
b) **Stream Cipher** is a method of encrypting text (to produce ciphertext) in which a cryptographic key and algorithm are applied to **each binary digit** in a data stream, one bit at a time.

A classic example is the **Vernam Cipher** (**1917**).

The Vernam Cipher is based on the principle that each **plaintext** character from a message is 'mixed' with one character from a **key stream**. If a truely **random** key stream is used, the result will be a truely 'random' **ciphertext** which bears no relation to the original plaintext. In that case the cipher is similar to the unbreakable **One-Time Pad (OTP).** As it was generally used with teleprinters and 5-level punched tape, the system is also known as **One-Time Tape** or **OTT**.

The **ciphertext** is generated by applying the logical **XOR** operation to the individual bits of **plaintext** and the **key stream**. The advantage of using the XOR operation for this, is that it can be undone by carrying out the same operation again:

plaintext + key = ciphertext ⇒ ciphertext + key = plaintext



Vernam Cipher Security

The above procedure is 100% safe if, and only **if,** the following conditions are all met:

- there are only two copies of the key-tape,
- both sides of the communications link have the same key-tape,
- the key-tape is used only once,
- the key-tape is destroyed immediately after use,
- the key-tape contains truly random characters,
- the key tape was not compromised during transport.

III.

a) **Symmetric Key Cryptography** - is an cryptographic system in which the sender and receiver of a message share a ***single, common key*** that is used to encrypt and decrypt the message.
-Symmetric key systems are simpler and faster (compared with the asymmetric systems), but their main disadvantage is that the *two parties must somehow exchange the key in a secure way*.
-Symmetric key cryptography is also called as *Secret-key Cryptography*.

The most popular symmetric key cryptography is "Data Encryption Standard", ***DES***.

b) **Asymmetric Key Cryptography** - is an cryptographic system that uses **pairs of keys**: public keys which may be disseminated widely, and private keys which are known only to the owner.
This accomplishes two functions:
•authentication, which is when the public key is used to verify that a holder of the paired private key sent the message,
•encryption, whereby only the holder of the paired private key can decrypt the message encrypted with the public key.

One of the most popular asymmetric key cryptography is ***RSA***.

Security of Cryptographic Algorithms

Depends on key length, ***L*** (one of the A. Kerckhoffs's principle).

Examples.

Let L = **8** bits, there are 2^8 keys; L = **56** bits - 2^{56} keys.

If a PC in 1 s analyzes 1 million keys, then analysis of 2^{56} keys needs **2285** years,

with L = **64** bit - **585000** years;

with L = **2048** bit - work 1 mln PC in 10^{597} years.

Modern cryptography is based primarily on the theory of numbers (large).

Number	Physical equivalent
2^{34} years	Age of the Universe
2^{170}	Number of a planet's atoms
$2^{1024} - 2^{2048}$	The number of all possible keys in a modern cryptographic system

Mathematical Foundations of Cryptography

- A bit ***b*** can take the two values, **0** and **1**.
- Two bits b_0b_1 can together take ***four values***.
- ***n*** bits $b_0b_1 \ldots b_{n-1}$ can together take 2^n ***values***.

- For ***x*** and ***y*** bytes, $x \oplus y$ is the **exclusive-or** of ***a*** and ***b***.
- If $x = x_0x_1x_2 \ldots x_7$ and $y = y_0y_1y_2 \ldots y_7$,

then $z = x \oplus y$ where $z_i = x_i + y_i \bmod 2$.

Definition 16. The integers

The integers are the natural numbers, their negatives and zero

$$\mathbb{Z} = \ldots. -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, \ldots$$

Definition 17. Division algorithm

Assume a, b are integers and $a > b$.
Write $a = qb + r$, where q, r are integers and $0 \leq r < b$.

q is **quotient** and
r is **remainder** of a divided by b.

Example.
With ***a*** = 67 and ***b*** = 9, one can write ***a*** = **7** · ***b*** + **4**, where ***q*** = **7**, ***r*** = **4**.

Definition 18. Prime numbers (or primes) - a prime number is a positive integer not divisible (without a remainder) by any positive integer other than itself and one.
Example.
The first ten primes are 2, 3, 5, 7, 11, 13, 17, 19, 23, 29.

There are about 10^{151} primes length of 1 to 512 bits.

Definition 19. Relatively prime - two *numbers* are "*relatively prime*" when they have no common factors other than 1.

Definition 20. Unique factorization - a positive integer is either a prime number or it can be written as a **product** of prime numbers.

If these prime number factors are ordered (pol. czynniki liczbowe są uporządkowane, e.g., listed from smallest to largest), this *factorization* is **unique**.

Example.

Consider the integer **420**. The integers 2, 3, 5, and 7 divide 420. The unique factorization of 420 is $2^2 \cdot 3 \cdot 5 \cdot 7$.

Fundamental theorem of arithmetic. Every natural number **n**, except 1, can be represented as the product of simple factors:
$n = p_1 * p_2 * p_3 * \ldots * p_m, \quad m > 1.$

Example.

N = 1554985071 = 3 × 3 × 4463 × 38713;
N = 39 616 304 = 2 × 13 × 7 × 2 × 23 × 13 × 2 × 13 × 2 × 7 = 2 × 2 × 2 × 2 × 7 × 7 × 13 × 13 × 13 × 23.

Definition 21. The Greatest Common Divisor (*GCD*) of two or more integers, which are not all zero, is the largest positive integer that divides each of the integers.

Example.
What is the greatest common divisor of 54 and 24?

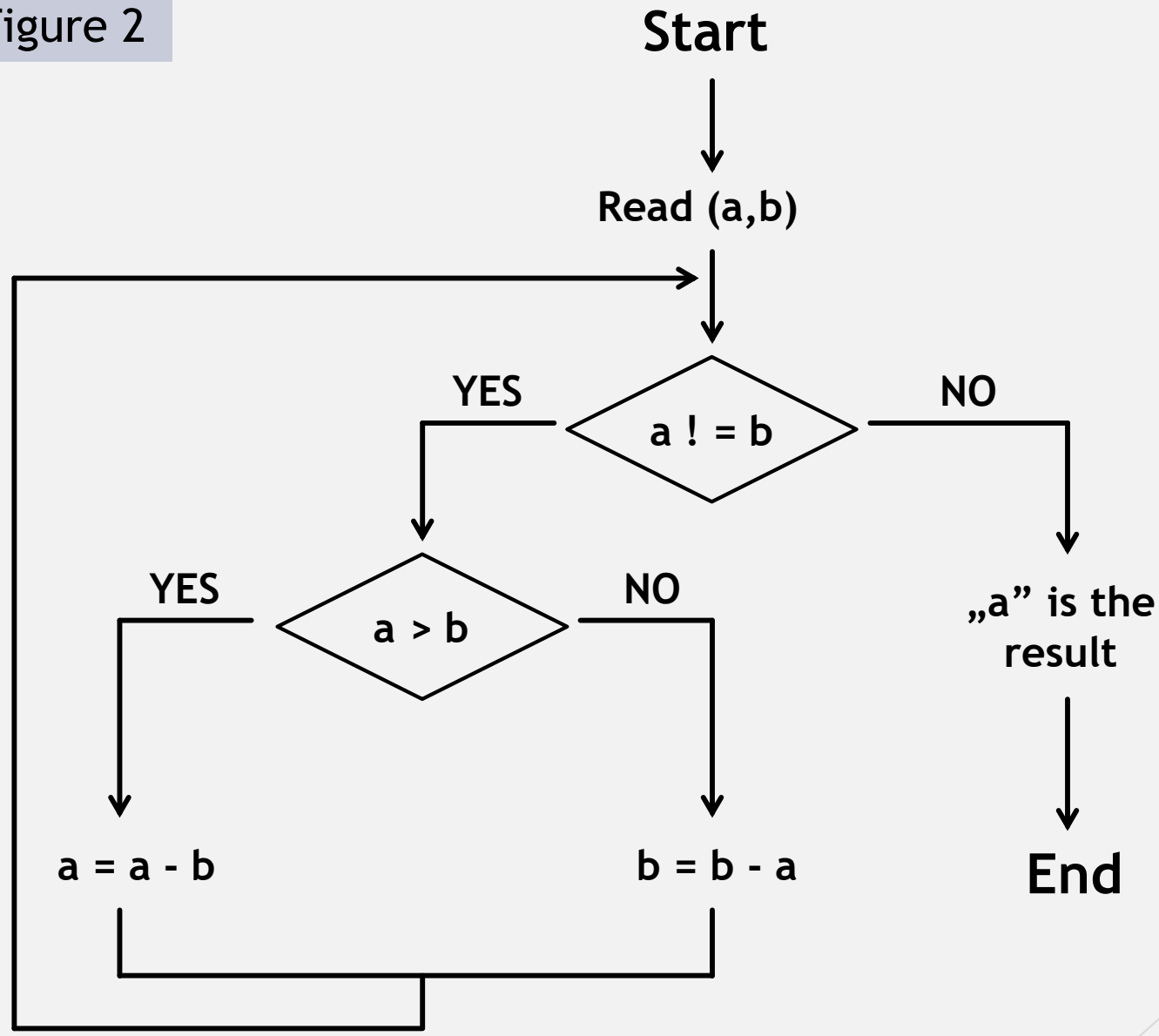
The number 54 can be expressed as a product of two integers in several different ways: $54 \times 1 = 27 \times 2 = 18 \times 3 = 9 \times 6$.

Thus the **divisors of 54** are: 1 , 2 , 3 , 6 , 9 , 18 , 27 , 54.
Similarly, **the divisors of 24** are: 1 , 2 , 3 , 4 , 6 , 8 , 12 , 24.

The numbers that these two lists share in common are the **common divisors** of 54 and 24: 1 , 2 , 3 , 6.
The greatest of these is 6. That is, the **greatest common divisor** of 54 and 24: *GCD (54 , 24) = 6*.

Euclid gave a simple and effective algorithm for search ***GCD***, it consists of 4 steps, as in Figure 2.

Figure 2



The **Euclidean algorithm** can be represented in the form of the equations:

$$a = b * q_1 + r_1, \quad 0 < r_1 < b,$$
$$b = r_1 * q_2 + r_2, \quad 0 < r_2 < r_1,$$
$$r_1 = r_2 * q_3 + r_3, \quad 0 < r_3 < r_2,$$
$$r_2 = r_3 * q_4 + r_4, \quad 0 < r_4 < r_3,$$
$$\ldots\ldots$$
$$r_{t-3} = r_{t-2} * q_{t-1} + r_{t-1}, \quad 0 < r_{t-1} < r_{t-2},$$
$$r_{t-2} = r_{t-1} * q_t + r_t, \quad 0 < r_t < r_{t-1},$$
$$r_{t-1} = r_t * q_{t+1}, \quad r_{t+1} = 0$$

Then r_t *(last remainder)* is the greatest common divisor of the numbers ***a*** and ***b***.

Example.

Let **a** = 525, **b** = 231. Use the Euclidean algorithm:

525 = 231 * 2 + 63;

231 = 63 * 3 + 42;

63 = 42 * 1 + **21**;

42 = 21 * 2.

And ***GCD*** (525, 231) = ***21***

Modular Arithmetic (or residue arithmetic)

Modular Arithmetic is also known as "clock arithmetic".

We all know the 24-hour clock, where the clock is reset to zero every day at midnight.

On a dial, the same position of the hands corresponds, for example, to 1 o'clock and 13 o'clock or 0 and 12 and 24 o'clock (***n***=12)

Lesser known is perhaps "***the integers modulo n***", where **n** is a positive integer. Here we subtract **n** from an integer if it is equal to n or if it exceeds n.

We say that n is "***congruent to zero modulo n***" and that **n+1** is "***congruent to 1 modulo n***" etc.

Formally,

Definition 22. Let ***a*** and ***b*** be integers and let **n** be a positive integer.
We shall write $a \equiv b \pmod{n}$ if m divides ($b - a$).
We say that "*a is congruent to b modulo n*".

In the operation $a \equiv b \pmod{n}$,
b is called the residue modulo ***n***.

Example.
Let $n = 5$, then $2 \equiv 17 \pmod{5}$, since 5 divides (17−2).
Also $2 \equiv -3 \pmod{5}$ since 5 divides (−3 − 2).

- Note that for positive n and any a it holds that $a \equiv a + n \pmod{n}$ and also that $kn \equiv 0 \pmod{n}$ for any integer k.

Take $a \in \mathbf{Z}$, *then we can write:*

$$a = qn + r,$$

where $0 \le r < n$.
Also

$$a \equiv r \pmod{n}$$

and we say *r* is "a reduced modulo n".

We shall also use the notation "*a mod n*" *for r.*
Negative numbers are allowed, e.g.,

$$-1 \equiv n - 1 \pmod{n}$$

Example.
The integers ***a*** = 23 *and* ***r*** = *11* are equal modulo ***n*** = 12:
*23 ≡ 11 mod 12 (23 = 1 * 12 + 11)*

(23 hours and 11 hours can mean *one and the same time*, as we noted earlier*)* – "*clock arithmetic*"

Rules of Modular Arithmetic

$(a + b) \bmod n = ((a \bmod n) + (b \bmod n)) \bmod n$
$(a - b) \bmod n = ((a \bmod n) - (b \bmod n)) \bmod n$
$(a * b) \bmod n = ((a \bmod n) * (b \bmod n)) \bmod n$
$(a * (b+c)) \bmod n = (((a * b) \bmod n) + ((a*c) \bmod n)) \bmod n$
$a+0=a$, $a+(\textit{not } a)=1$

Example.
Calculate $a^8 \bmod n$
A simple solution: $((a^2 \bmod n)^2 \bmod n)^2) \bmod n$

Example.
Calculate $a^{25} \bmod n$
solution: $a^{25} \bmod n = (a * a^{24}) \bmod n = ((((a^2\ a)^2)^2)^2\ a) \bmod n =$
$(((((((a^2 \bmod n)\ a) \bmod n)^2 \bmod n)^2 \bmod n)^2 \bmod n)\ a) \bmod n$

Calculate: $34+45 \bmod 5$=…. $34*45 \bmod 33$=……

Fermat's Little Theorem

If ***n*** *is a prime number, then for any integer* ***a****, the number* $a^n - a$ *is an integer multiple of* ***n****.*
In the notation of modular arithmetic, this is expressed as

$$a^n \equiv a \pmod{n}.$$

Example.
If $a = 2$ and $n = 7$, $2^7 = 128$, and $128 - 2 = 7 \times 18$ is an integer multiple of 7.

If ***a*** *is not divisible by* ***n***, Fermat's little theorem is equivalent to the statement that $a^{n-1} - 1$ is an integer multiple of **n**, or in symbols:

$$a^{n-1} \equiv 1 \pmod{n}.$$

Example.
If $a = 2$ and $n = 7$ then $2^6 = 64$ and $64 - 1 = 63$ is thus a multiple of 7.

A simple conclusion from the FLT is that if for ***N*** being a natural number we find natural ***a*** and such that

i. GCD **(a, n) = 1**, i.e. ***a*** and ***n*** are relatively prime,
ii. $\mathbf{a^{n-1} \equiv 1 \bmod n,}$

than ***N*** is a complex number.

In a general sense: **a ≡ b mod n,** if **a=b+kn** i **k** – integer**.**

In modular arithmetic the equation

$$a * x \equiv 1 \bmod n$$

is equivalent to finding such values **x** and **k**, that **a * x = n*k +1**
General task: find such **x**, that
$1 \equiv (a*x) \bmod n$
or **$a^{-1} \equiv x \bmod n$**

The last equation has only one solution,
if **a** and **n** - are relatively primes.

Example.
If a=5 , n=14,

than **x=3 : $5^{-1} \equiv 3 \bmod 14$,**

because
$(5*3) \bmod 14 \equiv 1$
a^{-1} - the **inverse value modulo**.

A modular multiplicative inverse of *a* modulo *n* can be found by using the extended Euclidean algorithm.
Consider this in more detail:

If GCD (a,n) =1, then

$$a^{-1}a = 1 \ mod \ n,$$

a^{-1} - the inverse of *a* modulo *n*.

Also true:

if $x^{-1} = y \ mod \ n$, then $y^{-1} = x \ mod \ n$.

If $yx = 1 \ mod \ n$ and GCD(x,n) =1, GCD(y,n) =1,
Then it's true

$$y\text{-}1 = x \bmod n,$$
$$x\text{-}1 = y \bmod n.$$

The last equations can be written in another form:

$$xy + kn = 1,$$

k – integer (the result of dividing xy/n).

The last of several equations produced by the algorithm may be solved for this GCD.
Then, using a method called "back substitution", an expression connecting the original parameters and this GCD can be obtained.

Example. Solve the equation using the extended Euclidean algorithm:
$7x = 1 \bmod 20$ or **$x^{-1} = 7 \bmod 20$.**
The first step. Find GCD(20, 7), using the **Euclidean algorithm** (see sl.26):
20=7*2+6 or **6=20-7*2,**
7=6*1+1 or **1=7-6*1.**
The second step. We use a "back substitution" method:
1=7-6*1 = 7- (20-7*2)*1= 7*3 + 20*(-1) = yx + kn =1,
or **y=7, x=3, k=-1.**
We found the root of the equation: **x=3.**
Example. Solve the equation **$7y = 1 \bmod 40$** or **$y^{-1} = 7 \bmod 40$.**
Find GCD(7,40):
40=7*5+5,
7=5*1+2,
5=2*2+1.
back substitution: **1= 5-2*2 = 5-2(7-5*1) = 5*3+7(-2)=(40-7*5)3+7(-2) = 40*3+7(-17) = kn + xy=1 mod n;** or **7(-17) = 7y,** because **-17 mod 40 = 23,** and **y=23,** those 23 is the inverse number 7 mod 40

Listing 1 shows the implementation of the extended Euclidean algorithm in C ++.

```
#define isEven(x) ( (x & 0x01) == 0)
# define isOdd(x) (x & 0x01)
# define swap(x,y) (x^= y, y^= x, x^= y)
void ExtBinEuclid(int *u, int *v, int *ul, int *u2, int *u3) {
// warning: u and v will be rearranged if u <v
int k, tl, t2, t3;
if (*u < *v) swap(*u,*v);
for (k = 0; isEven(*u) && isEven(*v); ++k) {
                *u >>= 1; *v >>= 1 ;
}
*ul = 1; *u2 = 0; *u3 = *u; tl = *v; t2 = *u - 1; t3 = *v;
do {
do {
if (isEven(*u3)) {
if (isOdd(*ul) || isOdd(*u2)) {
*ul += *v; *u2 += *u;
}
*ul >>= 1; *u2 >>= 1; *u3 >>= 1;
}
if (isEven(t3) || *u3 < t3) {
swap(*ul,tl); swap(*u2, t2); swap(*u3, t3);
}
} while (isEven(*u3));
while (*ul < tl || *u2 < t2) {
                *ul += *v; *u2 += *u;
}
*ul -= tl; *u2 -= t2; *u3 -= t3;
} while (t3 > 0);
```

Listing 1

```
while(*ul>=*v&&*u2>=*u){
*ul-=*v;*u2-=*u;
                                }
                *ul<<=k;*u2<<=k;*u3<<=k;
}
main(intargc,char**argv) {
inta,b,gcd;
if(argc<3) {
cerr<<“Using:xeucliduv"<<endl;
return-1;
                }
intu=atoi(argv[l]);
intv=atoi(argv[2J);
if(u<=0||v<=0) {
cerr<<"Argumentsshouldbepositive!"<<endl;
return-2;
}
//warning: u and v will be rearranged if u<v ExtBinEuclid(&u,&v,&a,&b,&gcd);
cout<<a<<"*"<<u<<"+(-"
<<b<<")*<<"<<v<<"="<<gcd<<endl;
                if(gcd==1)
cout<<"InverseValue"<<v<<"mod"<<u<<"equalto
                "
                                        <<u-b<<endl;
return0;
}
```

Listing 1. Ending

Source: Bruce Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition , 1996

Discrete Logarithm Problem

Definition 23. The primary (residual) root (pierwiastek pierwotny) modulo ***n*** is a such number, that its powers give all possible modulo ***n*** residues which are relatively prime of ***n***.

Example.

- The next residues modulo 5 of $\mathbf{2}^{\mathbf{i}}$ are: 2, 4, 3, 1
  (they give all possible residues).
  Number 2 is a **primary (residual) root** modulo 5.

- The next residues modulo 7 of $\mathbf{2}^{\mathbf{i}}$ are: 2, 4, 1, 2, ...
  (they do not give all possible residues).
  Number 2 is not a **primary (residual) root** modulo 7.

- The next residues modulo 17 of $\mathbf{3}^{\mathbf{i}}$ are:

  3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1

  (they give all possible residues).
  Number 3 is a **primary (residual) root** modulo 17.

In the equation

$$y = a^x \bmod n$$

we use the module ***n*** of the primary, eg. 17, and determine the **primary (residual) root** of 17, in this case **3**. It has the important property that when we use different powers (a^i), the solution will spread evenly from 0 to 16.
Number 3 is the generator.

Reverse procedure is much more difficult.
The Reverse - that is: for the known ***a***, ***n***, ***y*** we need to count ***x***.
This is called a discrete logarithm problem.

- And we have a *one-way function* that is easy to do and difficult to reverse!
- With small numbers - simple, but if you use a few hundred digit module, the solution becomes impractical.
- Even if you had access to all the computing power in the world, exploring all the possibilities would take thousands of years.
- Thus, the power of a one-way function depends on the time it takes to reverse it.

Definition 24. For an integer ***a*** and a positive integer ***n*** with **gcd(*a*,*n*) = 1**, the **order** of **a** modulo **n** is the smallest positive integer **k** with $a^k \equiv 1 \pmod{n}$.

Example

For $\alpha = 2$: $\alpha^i \bmod 17$ for $i = 1, \ldots, 16$ are

$$2, 4, 8, 16, 15, 13, 9, 1,\ \ 2, 4, 8, 16, 15, 13, 9, 1.$$

The order of 2 modulo 17 is 8.

Example

For $\alpha = 3$: $\alpha^i \bmod 17$ for $i = 1, \ldots, 16$ are
$3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1.$
The order of 3 modulo 17 is 16.

Example

For $\alpha = 4$: $\alpha^i \bmod 17$ for $i = 1, \ldots, 16$ are
$4, 16, 13, 1,\ \ 4, 16, 13, 1,\ \ 4, 16, 13, 1,\ \ 4, 16, 13, 1.$
The order of 4 modulo 17 is 4.

Elements in $\mathbb{Z}_{17}^*$ and their order modulo 17.

i	1	2	3	4	5	6	7	8
order of i mod 17	1	8	16	4	16	16	16	8
i	9	10	11	12	13	14	15	16
order of i mod 17	8	16	16	16	4	16	8	2

For good security it is recommended to choose primes of at least 1536 bits, popular is 2048 bits, that is $\mathbf{n > 2^{2047}}$.

Example a 2048-bit prime *n*:

16158503035655036503574383443349759802220513348577420160651727137623
27569433945446598600705761456731844358980460949009747059779575245460 5
47544076193224141560315438683650498045875098875194826053398028819192 0
33784138396109321309878080919047169238085235290822926018152521443787 9
45770532904303776199561965192760957166694834171210342487393282284747 4
28088017663161029038902829665513096354230157075129296432088558362971 8
01859230928678799175576150822952201848806616643615613562842355410104 8
62578550863465661734839271290328348967522998634176499319107762583194 7
18667771801067716614802322659239302476074096777926805529798117247

→ http://www.keylength.com

References:

1. Bruce Schneier. Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition , 1996 (URL: http://www.cryptomuseum.com/crypto/vernam.htm)

2. Харин Ю.С., Берник В.И., Матвеев Г.В. Математические основы криптологии.– Минск: БГУ, 1999. – 319 с.

3. Урбанович, П. П. Защита информации методами криптографии, стеганографии и обфускации : учеб.-метод. пособие для студ.. - Минск : БГТУ, 2016. - 220 с.
(URL: http://elib.belstu.by/handle/123456789/23763)

4. J. Pieprzyk, T. Harjono, J. Seberry. Fundamentals of Computer Security, Springer-Verlag, 2003

5. MIT Open Cource Ware: Cryptography and Cryptanalysis, [Electronic Resource], URL: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-875-cryptography-and-cryptanalysis-

6. Greatest_common_divisor, [Electronic Resource], URL: https://en.wikipedia.org/wiki/Greatest_common_divisor

7. Shafi Goldwasser, Mihir Bellare. Lecture Notes on Cryptography, [Electronic Resource], URL: http://cseweb.ucsd.edu/~mihir/papers/gb.pdf