Kriptográfia 2.

TibiV 2002

IT alapok

Klasszikus titkosító rendszerek

- **A helyettesítő titkosítók (S-boxok)**

  A nyílt szöveg betűit (esetleg nagyobb blokkjait) egyesével bijektív módon a titkosított szöveg betűivel helyettesítjük.

- **A keverő titkosítók (P-boxok)**

  A titkosított szöveg a nyílt szöveg karaktereinek permutációja.

- **A produkciós titkosítók**

  keverés-helyettesítés (többszörös) egymás utáni alkalmazása

Titkosítók generációi

- **Első generáció:** XVI-XVII. századig, főleg egyábécéshelyettesítések (pl. Caesar)
- **Második generáció:** XVI-XIX században, többábécés helyettesítések (pl. Vigenére)
- **Harmadik generáció:** XX sz. elejétől Mechanikus és elektromechanikus eszközök (pl. Enigma, Hagelin, Putple, Sigaba)
- **Negyedik generáció:** a XX. század második felétől, produkciós titkosítók, számítógépekkel (pl. DES, Triple DES, Idea, AES)
- **Ötödik generáció**: kvantumelvű titkosítások, sikeres kisérletek vannak rá, de gyakorlati alkalmazásuk ma még futurisztikus ötletnek tűnhet

A helyettesítő titkosítók

- **Karakterekkel:** a nyílt szöveg betűi (jelei, betűcsoportjai) sorra más jelekkel helyettesítődnek
- **Bitenként:** ha bitenként tekintjük a szövegeket, akkor rögzített hosszú (pl. 64 bit) bitcsoportokat ugyanolyan hosszú bitcsoportokra cserélünk
- Minden esetben a jelek pozíciója változatlan marad

A helyettesítő titkosítók fajtái

- **Monoalfabetikus helyettesítés**

A monoalfabetikus helyettesítésnél ugyanaz a karakterek csakis ugyanazzal a karakterrel helyettesítődik.

Biztonság: Ezt titkosítási fajtát gyakoriságanalízissel viszonylag egyszerűen (papírral-ceruzával) fel lehet törni.

- Ebbe a fajtába tartoznak: - Ceasar-féle kódolási módszer
  - Átváltási táblázatos kódolási módszer

- **Poligrafikus helyettesítés**

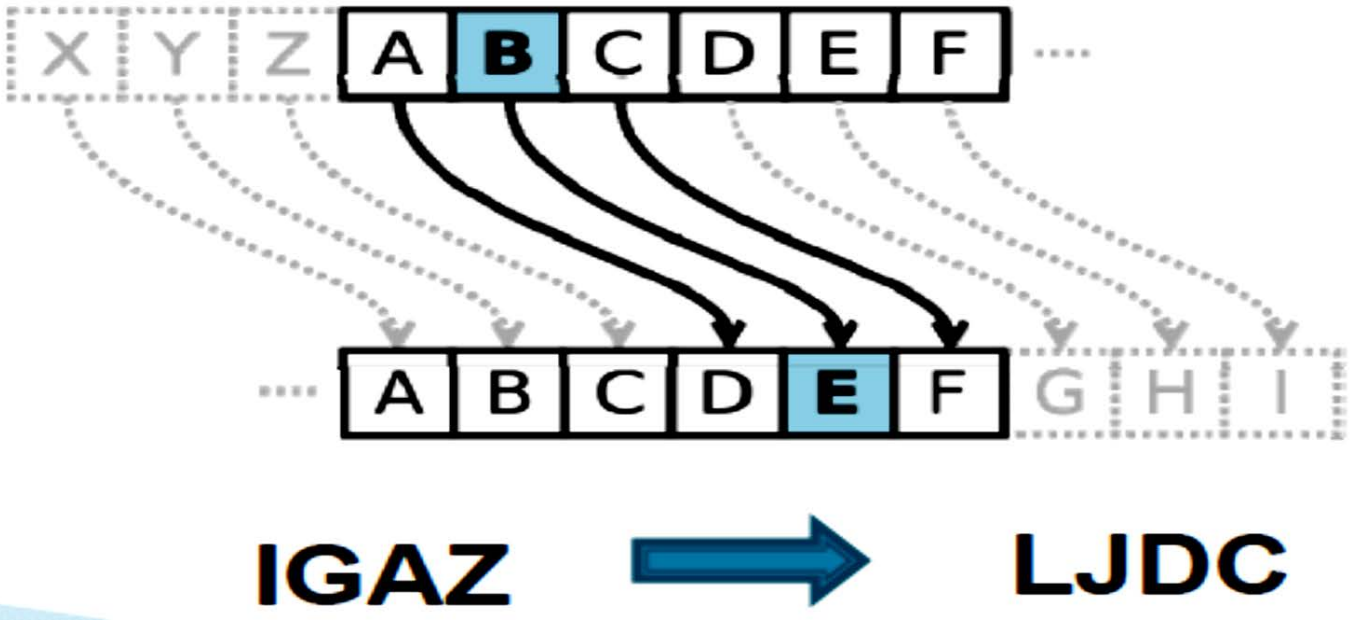
Ennél a fajtánál egy karaktersort egy másik karaktersorral helyettesítenek.

Biztonság: Ez a módszer megnehezíti a gyakoriságanalízises kódfejtést.

- Ebbe a fajtába tartoznak: - Porta-féle kódolás

Caesar titkosító (Caesar chiper)

- Az első bizonyítottan használt háborús alkalmazása a helyettesítő titkosításnak
- Helyettesítsünk minden betűt az ábécé rendben után a következő harmadik betűvel



Eltoló/léptető titkosító (Shift cipher)

- helyettesítsünk minden betűt az ábécé rendben utána következő k-dik betűvel
- Caesar a k=3 kulcsot használta, Augustus k=2-t
- Pl.: k=3 –ra a helyettesítés:

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

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

- a matematikai leíráshoz a betűket számokkal azonosíthatjuk:

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

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 nyílt szöveg betűi kisbetűk,
- a titkosított szövegéi nagybetűk
- a magyar szövegeket is ékezet nélkül tekintjük

A eltoló rendszer kriptoanalízise

- csak 26 lehetséges kulcs van:◦ „a” képe lehet A,B,…,Z
- ezek sorra kipróbálhatók, azaz teljes kipróbálással feltörhető
- csak ismert kódszöveg típusú támadással is
- persze ehhez fel kell tudni ismerni a nyíltszöveget

Egy ábcés – monoalfabetikus helyettesítés

- az ábécé betűinek egyszerű **eltolása helyett** tetszőlegesen össze is rendezhetjűk a betűket
- így minden nyílt betűt egy titkossal helyettesítünk
- különböző nyílt betűket különbözőekkel
- ekkor e kulcs a 26 betű egy sorrendje:

KULCS: abcdefghijklmnopqrstuvwxyz
DKVQFIBJWPESCXHTMYAUOLRGZN

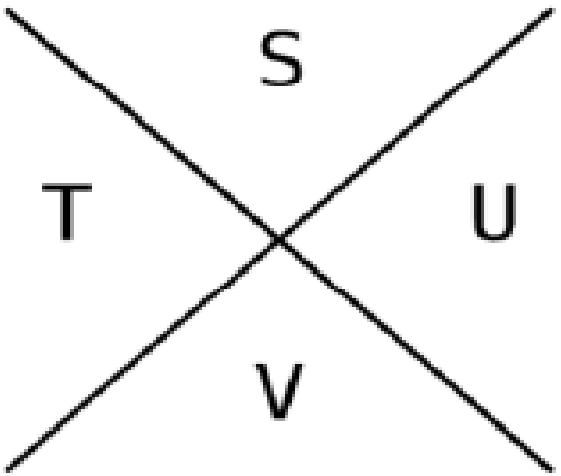
Nyílt szöveg : ifwewishtoreplaceletters

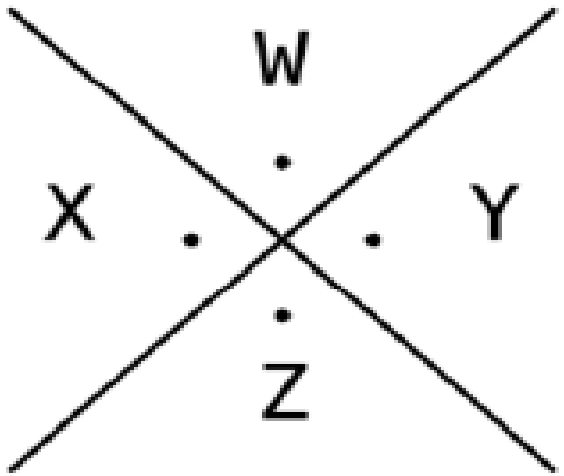
Titkos szöveg: WIRFRWAJUHYFTSDVFSFUUFYA

Pig -pen titkosítás

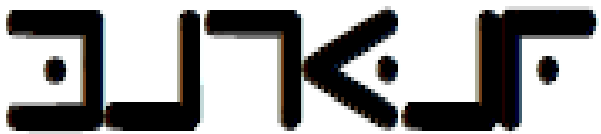
A	B	C
D	E	F
G	H	I

J	K	L
M	N	O
P	Q	R





MAGYAR



Az egyábcés helyettesítés kriptóanalízise

- a kulcstér most 26! ≈ 4 x 1026 ≈ 288.4 elemű
- ez biztonságosnak látszik
- de ez csak a teljes kipróbálás ellen véd

  NEM BIZTONSÁGOS !!!
- a kriptoanalízis a nyílt szöveg nyelvének nyelv statisztikai sajátosságain alapszik
- **A gyakorlatban egy kb. 50 betűs szöveg már feltörhető!**

Tikosított szövegek statisztikai elemzése

A titkosított szövegek feltörésének egyik módszere a statisztikai elemzés:

- az emberi nyelvek redundánsak
- pl. a magánhangzók elhagyhatók:
- „mndn zldsg s gymlcs fnm"
- nem minden betű egyformán gyakori
- az angolban: E, T,A,O,I,N,S,H,R....,J,X,Q,Z
- magyarban (ékezetekkel)
    - a leggyakoribb az "E", "A" és "T"
    - majd "L", "N", "S", "K", "O", "R"
    - igen ritka: "Ő", "W", "X", "Q"

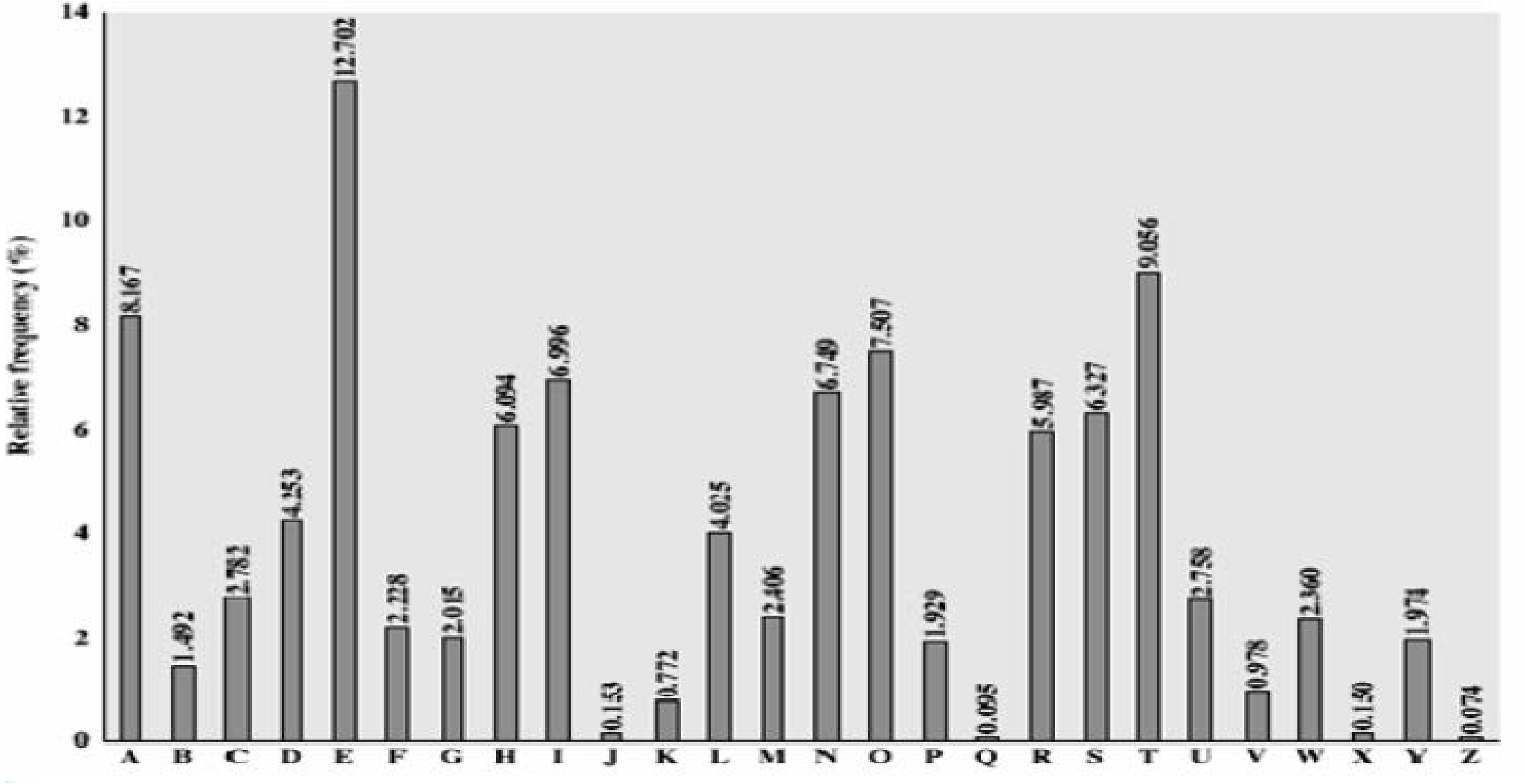
Statisztikai elemzés

- hasonlóan lehet a betűpárok (digram), hármasok (trigram) gyakoriságát vizsgálni pl.

  ”SZ”, ”TT”, "THE"

- ezek a statisztikák a nyelvekre jellemzőek, segítségükkel a nyílt szöveg nyelve azonosítható
- MÁS MÓDSZER: gyakori / jellemző szavak keresése: pl. pénzügyi szövegben m**illi**ó =
- ABBA minta, időjárásjelentésben: eső

Betű gyakoriság az angolban



Betű gyakoriság a magyarban

q	0,003
x	0,027
w	0,047
ö	0,092
ü	0,177
ú	0,343
í	0,439
ű	0,589
c	0,849

ó	0,871
p	0,934
u	1,043
f	1,098
j	1,179
h	1,808
ő	1,830
b	1,887
v	1,922

d	2,284
y	2,305
é	2,979
á	3,159
g	3,527
i	3,667
z	3,721
m	3,819
r	4,346

o	4,364
k	4,586
s	5,212
n	5,671
l	6,523
t	9,213
a	9,278
e	10,189

Kiszámolható a titkosított szöveg betűinek/betűpárjainak gyakorisága

Az egyábécés helyettesítés nem változtatja meg a betűgyakoriságot!

(már az arabok is felfedezték a IX. században)

Ha a szóközöket és az írásjeleket meghagyjuk, sokkal könnyebb az egyábécés helyettesítés feltörése

Mit lehet tenni? – A helyettesítés változatai

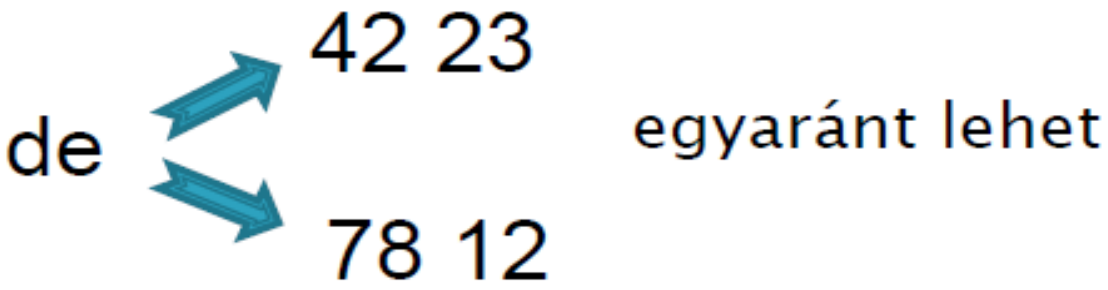
- Mint láttuk az egyábécés helyettesítés könnyen feltörhető, mert a kódszöveg megtartja a nyílt szöveg betűgyakoriságait.
- Ezen több módon lehet javítani, nehezebbé (de nem lehetetlenné!) téve a kriptoanalízist:

    1. homofónok használata
    2. nagyobb egységek pl. betűpárok helyettesítése pl. ilyen a Playfair titkosító
    3. több ábécés helyettesítések pl. Enigma

Homofónok használata

- A gyakoribb betűknek több kódot feleltessünk meg.

```
d -> 42, 59, 78, 91
e -> 12, 23, 32, 48, 66, 73, 88, 89, 97
  ...
x -> 15
```

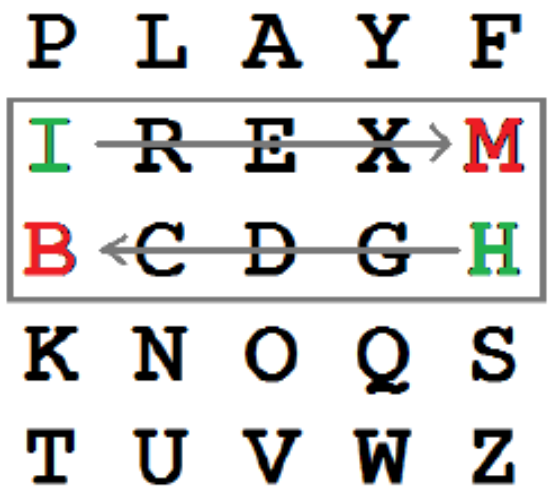


- gyakoribb betűknek több képük van, ezzel "elrejthetjük" ugyan betűgyakoriságokat
- de a több-betűs minták gyakorisága továbbra is megmarad

Következő lépés : Poligrafikus helyettesítés

Poligrafikus helyettesítés: Ne betűket, hanem karaktercsoportokat cseréljünk.

Playfair-titkosító:



A kulcs: **PLAYFAIR EXAMPLE**

IH ⟹ MB

Playfair titkosító

- betűpárok betűpárokkal való helyettesítésén alapul
- Charles Wheatstone találta fel 1854-ben, de a barátjáról Baron Playfairről nevezte el
- az angol hadsereg széles körben használta az I. világháborúban de még előfordult a II-ban is (Kennedy későbbi elnök, 1943)
- előnye, hogy egy személy eszköz segítsége nélkül papíron használhatja

Playfair kulcs mátrix

- egy 5X5-ös mátrix melynek első betűit a kulcsszó határozza meg
- a kulcsszó betűinek csak az első előfordulását vesszük
- a mátrix többi részét kitöltjük az abc maradék betűivel
- Pl. ha a kulcsszó MONARCHY:

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

A nyílt szöveg betű párokra osztjuk, ha egy kódolandó pár egy betű ismétlése, akkor egy elválasztó betűt, mondjuk 'X’-et teszünk közéjük és ezután kódoljuk.
Pl > ba lx lo on

Playfair titkosítás és megfejtés

1. ha a két betű egy sorban van, helyettesítsük őket a tőlük közvetlenül jobbra lévő betűkkel (a sor vége után a sor első betűjére ugorva) Pl. ar -> RM
2. ha a két betű egy oszlopban van, helyettesítsük őket a tőlük közvetlenül alattuk lévő betűkkel (az oszlop alja után a legfelső betűre ugorva) Pl. mu -> CM
3. különben a betűk kódja a saját sora és a másik betű oszlopának metszetében álló betű. Pl. hs -> BP, ea -> IM

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

A playfair kriptóanalízise

- jóval erősebb az egyábécés helyettesítésnél mivel 26 x 26 = 676 betűpár van
- a gyakoriság táblázathoz így 676 gyakoriságérték kell (szemben a 26 betűvel) így hosszabb titkos szövegre van szükségünk
- de fel lehet törni néhány száz betűs szöveg esetén is, mivel a nyílt szöveg struktúrájából még mindig sok tükröződik a titkosított szövegben
- Mit tehetünk? -> többábécés helyettesítések

Több ábécés helyettesítések

Ne egy ábécét, azaz helyettesítést, használjunk, hanem többet, valamilyen rendszer szerint váltogatva.

Több ábécés tikosítók (Polyalphabetic Ciphers)

Két közös jellemzőjük:

1. Betűnként más-más (egymással összefüggő) ábécét, pontosabban egyábécés helyettesítést használnak
2. Az hogy mikor melyik ábécé kerül sorra, a kulcs határozza meg
   - általában a kulcs véget érése után a használt ábécék ciklikusan ismétlődnek
   - minél több az ábécé, annál jobban kiegyenlítődik a betűgyakoriság megnehezítve ezzel a kriptoanalízist

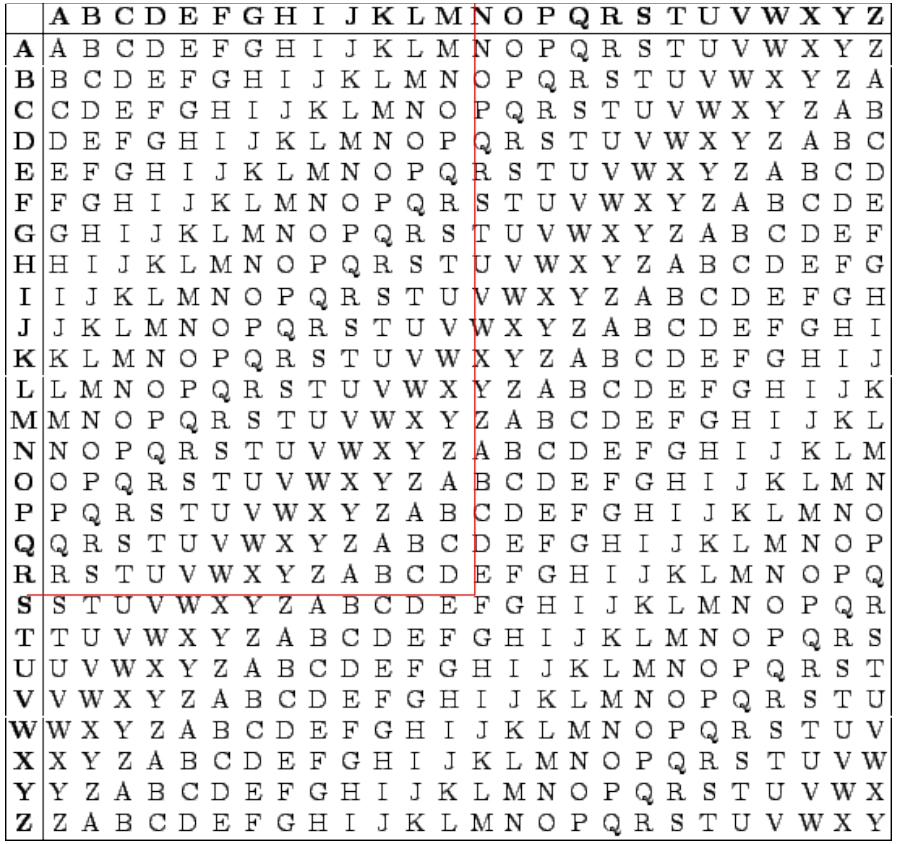
Vigenere titkosítás

- Titkosítandó szöveg
- Titkosító kulcs

Pl:

- Szöveg: REJTJEL
- Kulcs: MACSKA
- Titkosított szöveg: DELLTEX

Egy kulcs segítségével a kódolt szöveget létrehozzuk. A nyílt szöveget kulcs hosszúságú darabokra osztjuk, majd a darabok alá a kulcsot írva az egyes betűket egy táblázat megfelelő rovatából keressük ki.



	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

A tabula recta

	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

A kriptográfiában a tabula recta egy négyzet alakú tábla, amelyen a betűk minden egyes sorban ábécébe vannak rendezve, de minden sorban az eggyel feljebb levő sorhoz képest eggyel balra eltolódva.

A Vigenere kriptóanalízise

- A kódot a számtalan variációs lehetőség miatt sokáig feltörhetetlennek gondolták, azonban Leonhard Euler felfedezte, hogy a kulcsnál lényegesen hosszabb szöveg esetén periodikus ismétlődések lesznek a kódolt szövegben.
- Az összes periódus legnagyobb közös osztója lesz a kulcs hossza.
- Ekkora darabokra bontva a szöveget a visszafejtés lényegesen egyszerűbbé válik.
- Ha a kulcs azonos hosszúságú a szöveggel, akkor a fejtés nagyom nehéz.

A Vigenère-rejtjel Blaise de Vigenère (a képen) után kapta a nevét, bár Giovan Battista Bellaso hamarabb találta fel ezt a kódot. Vigenère egy erősebb autokulcs-kódot talált fel.

Vernam (XOR) titkosítás

INPUT		OUTPUT
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

- Alapja Vigenere titkosítás ahol :
- Ideális estben a kulcs ugyanolyan hosszú, mint a nyílt szöveg
- Ezt Gilbert Vernam (AT&T) javasolta 1918-ban

Az ő rendszere bitenként dolgozik:

$c_i = p_i$ XOR k_i

Ahol pi = a nyílt szöveg i-dik bitje

k_i = a kulcs i-dik bitje

c_i = a titkosított szöveg i-dik bitje

XOR = a kizáró vagy művelet,

0 XOR 1 = 1 XOR 0 = 1

0 XOR 0 = 1 XOR 1 = 0

Példa:

Nyílt szöveg: 00 10 11 01 10

Kulcs: 10 11 01 10 11

Titk. szöveg: 10 01 10 11 01

Kulcs: 10 11 01 10 11

Nyílt szöveg: 00 10 11 01 10

Egyszeri hozzáadásos titkosító (one-time pad)

- Ha a kulcs valóban véletlen és ugyanolyan hosszú, mint a nyílt szöveg a titkosító nem törhető fel (=feltétlenül biztonságos)
- Ezt a két feltétel azonban szigorúan be kell tartani, például nem szabad ugyanazzal a kulccsal még egyszer üzenetet titkosítani (innen az egyszeri név)
- Ezt hívják egyszeri hozzáadásos módszernek One-Time pad: OTP
- A OTP azért feltörhetetlen mert a titkosított szövegnek nincs statisztikai kapcsolata a nyílt szöveggel
- A gyakorlatban két nehéz probléma van vele:
  - **valóban véletlen kulcsgenerálás**
  - **a kulcselosztás és tárolás problémája**

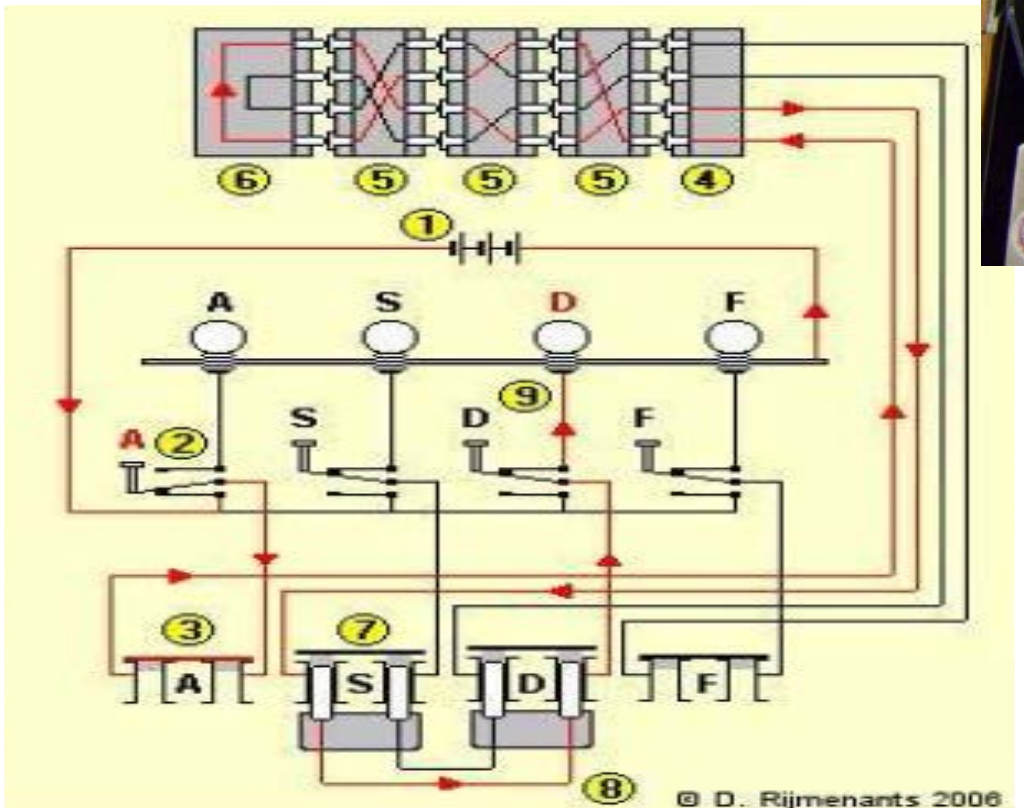
Gilbert Vernam

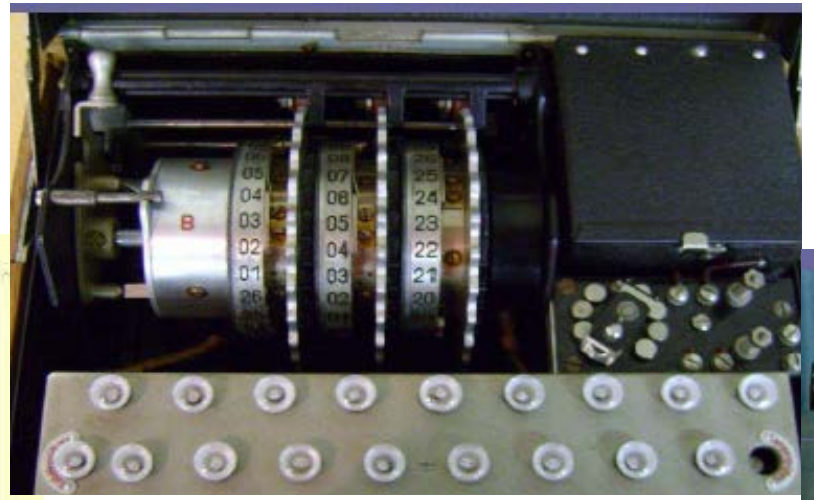
Rotoros gépek (Rotor Machines)

- A számítógépek és ezzel a modern titkosítók megjelenése elött a rotoros gépek voltak a legelterjedtebb komplex titkosító eszközök
- Széles körben használták a II. világháborúban:
  - németek: Enigma,
  - szövetségesek: Hagelin,
  - japánok: Purple
- Igen bonyolult többábécés helyettesítések forgó korongok (rotorok) segítségével, melyek egy-egy egyszerű helyettesítést kódoltak, de minden betű titkosítása után számlálószerűen különbözı sebességgel forogtak
- Pl. egy 3 rotoros gép 263=17576 ábécével dolgozott
- Működés: http://enigmaco.de/enigma/enigma.html

Az Enigma

- Minden betű titkosítása után a rotorok számláló szerűen forognak, ez 263 = 17576 ábécés titkosítás





A Hagelin (amerikai) és a Purlpe (japán) változat

Keverő titkosítók - Transposition Ciphers

- a helyettesítés mellett a másik alap titkosítási módszer a keverés (pemutációk)
- a szöveg egységek (betűk/bájtok/bitek/bitcsoportok) megmaradnak
- csak a sorrendjük változik meg
- alkalmazásuk felismerhető, mert a jelek gyakoriságát nem változtatják meg.

Kerítés rács elrendezés (**Rail Fence cipher**)

- írjuk le az üzenetet átlósan lefelé több sorba, majd olvassuk el soronként balról jobbra haladva
- Példa: (This is a secret message)

Plan text: T H I S I S A S E C R E T M E S S A G E

KEY =2

T		I		I		A		E		R		T		E		S		G	
	H		S		S		S		C		E		M		S		A		E

Tikos szöveg: T I I A E R T E S G H S S S C E E M S A E

KEY=4

T						A						T						G	
	H				S		S				E		M				A		E
		I		I				E		R				E		S			
			S						C						S				

Tikos szöveg: T A T G H S S E M A E I I E R E S S C S

Soronként cserélő titkosítók (Row Transposition Ciphers)

- bonyolultabb keverést kapunk, ha az üzenetet soronként adott számúoszlopba írjuk
- majd az oszlopokat a kulcs által megadott sorrendben olvassuk össze felülről lefelé

Példa: A szöveg “the simplest possible transpositions”

- Hozzunk létre 5 oszlopos táblázatot
- A szöveget soronként írjuk be
- Az üres helyeket töltsük fel
- Kulcs legyen: 15342

1	2	3	4	5
T	H	E	S	I
M	P	L	E	S
T	P	O	S	S
I	B	L	E	T
R	A	N	S	P
O	S	I	T	I
O	N	S	X	X

Oszlopok felcserélése a kulcs értékében

1	2	3	4	5
T	H	E	S	I
M	P	L	E	S
T	P	O	S	S
I	B	L	E	T
R	A	N	S	P
O	S	I	T	I
O	N	S	X	X

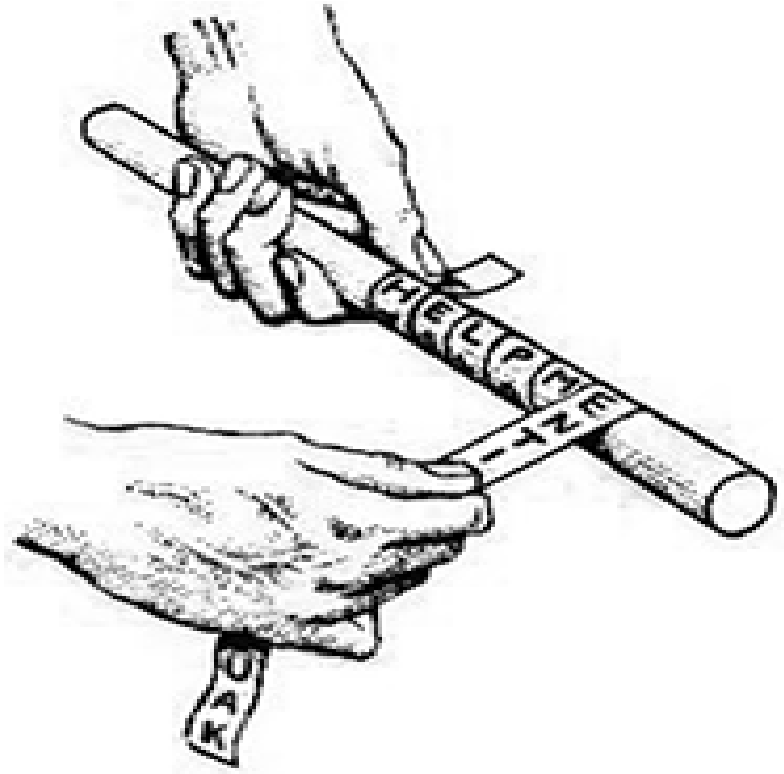
1	5	3	4	2
T	I	E	S	H
M	S	L	E	P
T	S	O	S	P
I	T	L	E	B
R	P	N	S	A
O	I	I	T	S
O	X	S	X	N

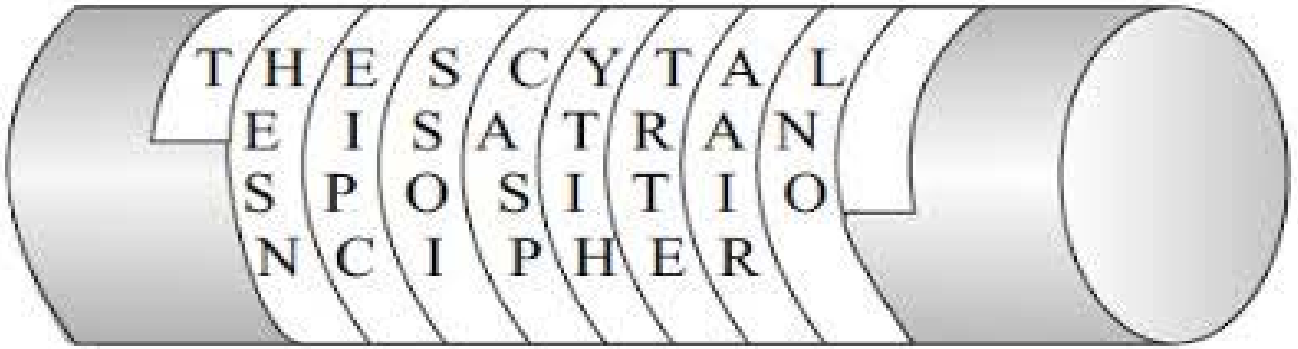
A kulcs : 15342

A titkosított szöveg: TIESH MSLEP TSOSP ITLEB RPNSA OIITS OXSXN

Skitlai (scytale)

- Spártaiak használták katonai célokra
- A kulcs a bot átmerője, csak a megfelelő átmérőn olvasható az üzenet





Duplán keverő titkosító

- Még biztonságosabb titkosításhoz jutunk, ha az előzőkeverést kétszer végezzük el, különbözőkulcsokkal (azaz permutációkkal)
- A kulcsok által meghatározott permutációja az oszlopoknak különböző elemszámú véletlen betűkkel töltjük ki az üzenet végét, hogy teljes sorokat kapjunk
- A permutációkat jelszavak segítségével is elő lehet állítani.
- Cryptool Permutation/Transposition Cipher

Produkciós titkosítók (Product Ciphers)

- Sem a helyettesítő, sem a keverő titkosítók nem biztonságokat, a nyelv jellegzetességei miatt
- Az ötlet: alkalmazzuk őket egymás után, hogy erősebb titkosításhoz jussunk, de:
  - két helyettesítés eredménye egy újabb (általában komplexebb) helyettesítés
  - két keverés egymásutánja továbbra is egy újabb keverés
  - de ha a keveréseket és a helyettesítéseket egymás után váltogatjuk (esetleg többször) valóban erősebb titkosításhoz jutunk
- A különbözı elvű titkosítások keverése vezet a modern szimmetrikus módszerekhez (DES, AES, stb.)

Modern Blokk titkosítók

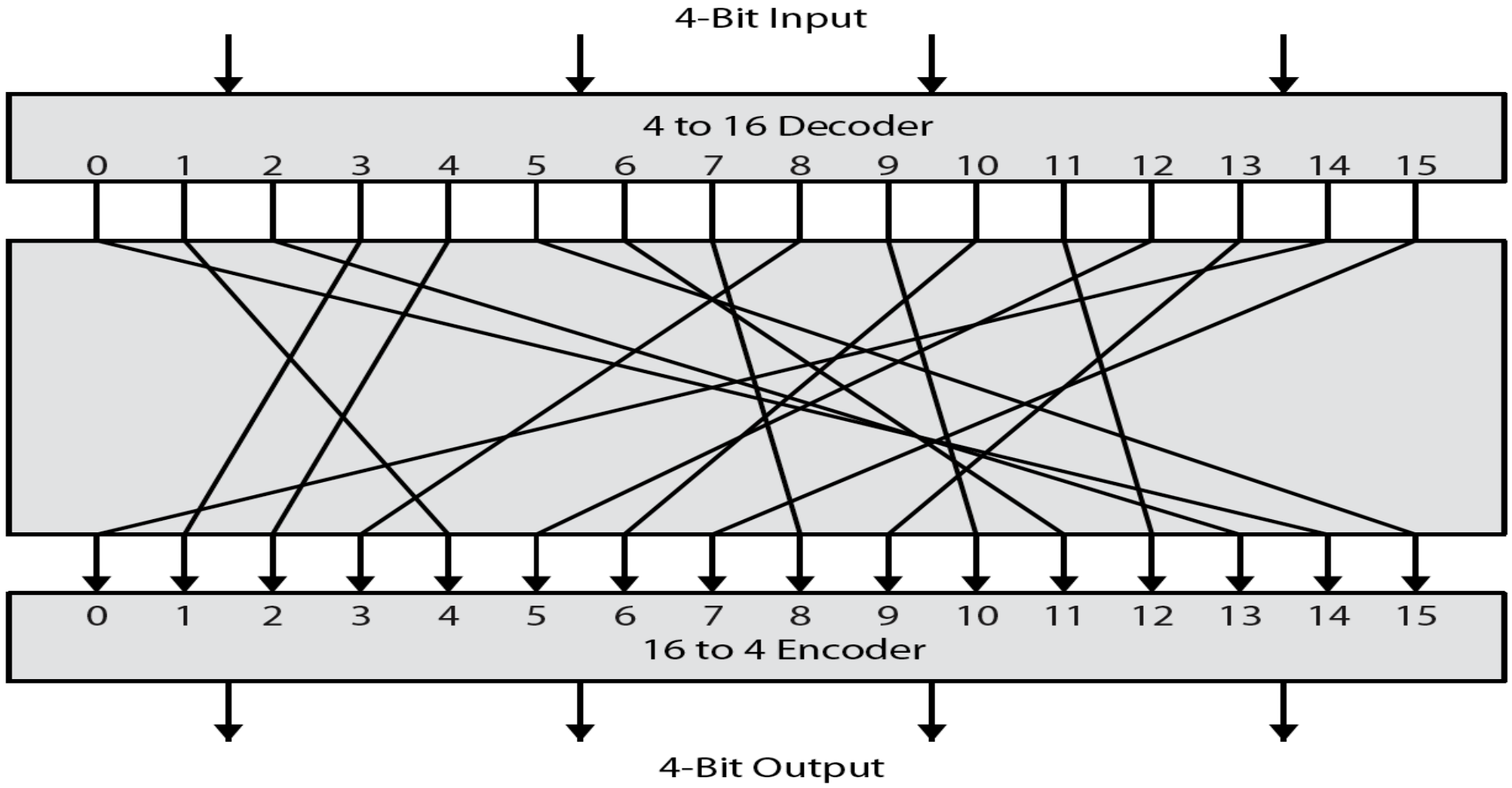
Ha az eddig módszerek mind feltörhetőek akkor hogyan titkosítsunk?

- A modern (gépi) módszerek következnek
- Ma is a szimmetrikus blokktitkosítók a legelterjedtebbek (gyorsak, sokat vizsgáltak)
- Ezek titkosságot / hitelességet biztosítanak

Block vagy folyam titkosítók ?

- a folyam titkosítók (stream ciphers) az üzenetet bitenként vagy bájtonként titkosítják / fejtik meg
- ekkor nem kell egy blokknyi adatot összevárni a titkosítás megkezdéséhez
- számos mai titkosító blokk titkosító
- szélesebb körben alkalmazottak, mint a folyam titkosítók
- blokkonként lehet velük adatfolyamot is továbbítani

Az ideális blokktitkosító



Blokk titkosítók alapelvei

- a blokk titksosítók hatalmas ábécék feletti helyettesítések
- de szükség van rá, hogy a helyettesítés injektív legyen, sőt effektíven kiszámítható
- általános esetben 64 biten $2^{64}!$ darab helyettesítés definiálható
- a legtöbb szimmetrikus blokk titkopsító ún.

  Feistel titkosító stuktúrán alapszik

Horst Feistel ötlete a 70-es évek elején

Shannon (S-P hálózatos) produkciós titkosítós ötletéből

könnyen invertálható titkosítók általános terve

Helyettesítő-keverő titkosítók (Substitution-Permutation Ciphers)

- Claude Shannon vezette be a helyettesítő-keverő hálózatokat (S-P networks) 1949-ben
- a modern blokktitkosítók rajtuk alapulnak
- az S-P hálózatok pedig a két korábbi alap kriptográfiai műveletből épülnek fel:
  - *helyettesítés (S-doboz) / substitution* (S-box) /
  - *keverés (P-doboz) / permutation* (P-box) /
- ezek biztosítják a titkos szövegnek mind a nyílt szövegtől, mind a kulcstól való minél komplexebb, de könnyen megadható függését

Diffúzió és konfúzió (Confusion and Diffusion)

- a titkosítónak a nyílt szöveg statisztikai jellemzőit a felismerhetetlenségig el kell rejtenie a titkosított szövegben
- a gyakorlatban erre Shannon az **S és P dobzok** szolgálnak kombinálását javasolta, a két cél:
- **diffúzió (szétterjesztés)** – szétoszlatja a nyílt szöveg statisztikai struktúráit egy terjedelmes méretű titkos szöveg részekben
- **konfúzió (összekeverés)** – a titkos szöveg és a kulcs kapcsolatát minél komplexebbé teszi

Szimmetrikus kulcsú titkosítás

A folyamat :

- A titkosítandó szöveget a közös titkosítási kulcsot felhasználva átalakítjuk, az így kapott információt továbbítjuk.
- A fogadó fél ugyanazt a közös titkosító kulcsot használva fejti azt meg.
- Az aszimmetrikus titkosításnál a szimmetrikus kulcsú titkosításkor a használt titkosítási kulcs különleges eljárást, biztonságot igényel.
- Ezt speciális kulcskezelési rendszerek támogatják. Problémát okoz, hogy a kulcs egyértelműen feloldja a védett információt.
- Speciális feladat a titkos kulcs küldő és fogadó fél közötti cseréjének a megoldása.
- További probléma a titkosító kulcsok biztonságának védelme a helyi számítógépeken.

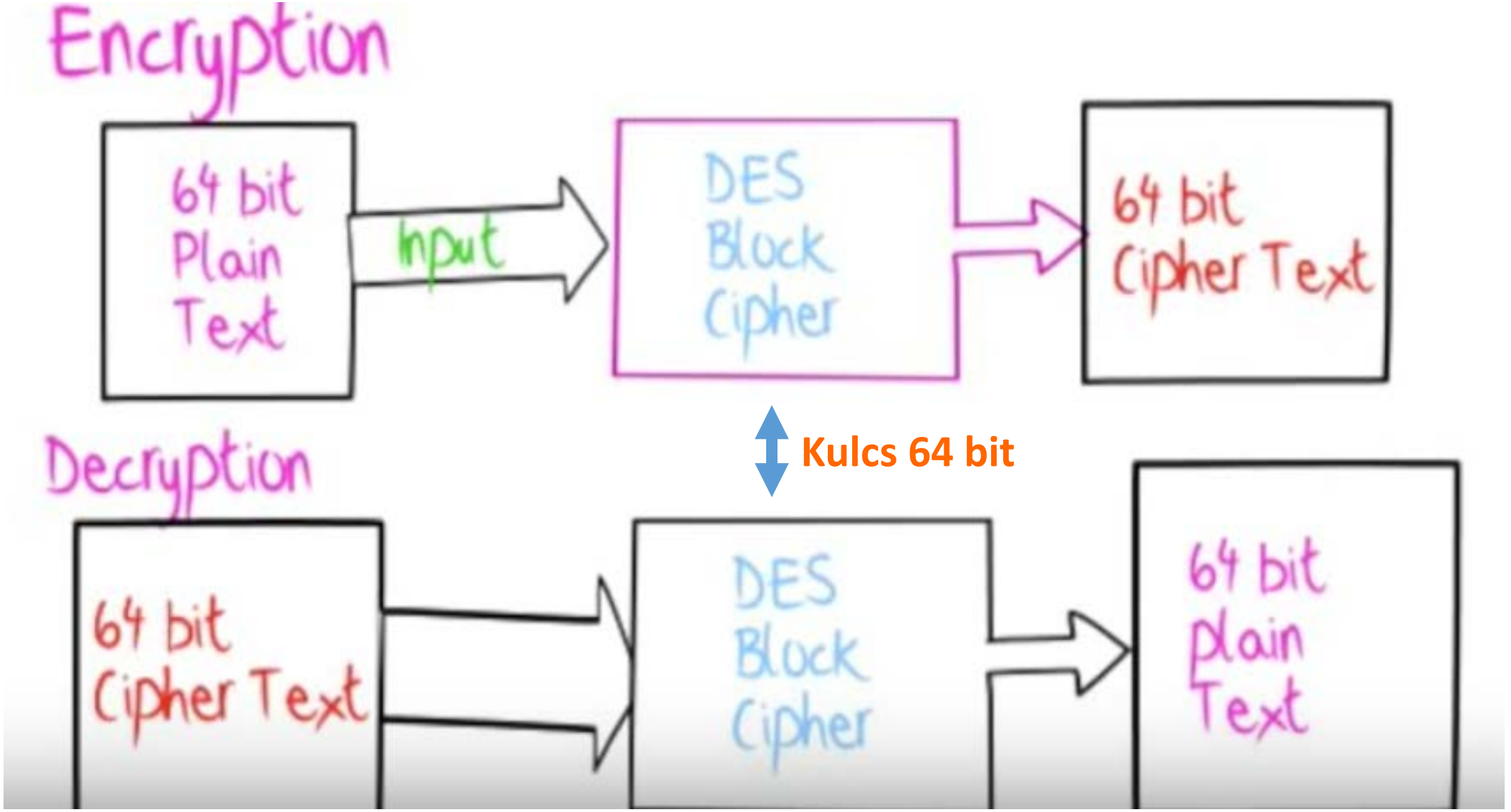
DES

- A **DES (Data Encryption Standard**) szimmetrikus kulcsú titkosítási eljárást az Amerikai Egyesült Államok Szabványügyi Hivatala (NBS) 1976-ban nyilvánította szövetségi szabvánnyá.
- A felhívást egy szabványos titkosítási algoritmus elkészítésére 1973-ban adták ki.
- A hivatal az IBM által Lucifer kódnéven benyújtott algoritmust fogadta el. Ezt követően az NBS és a NSA 3 éven át vizsgálta és finomította az eljárást és az 1976-ban vált az USA-ban szövetségi szabvánnyá.
- A DES szabvány előírása szerint a DES algoritmust 5 évente ismételt biztonsági felülvizsgálat alá kellett vetni. 1983-ban a felülvizsgálat sikeres volt, azonban az NSA 1987-ben már nem tartotta kellően megbízhatónak.
- Ennek ellenére, mivel más megfelelő alternatíva nem állt rendelkezésre egészen 1993-ig

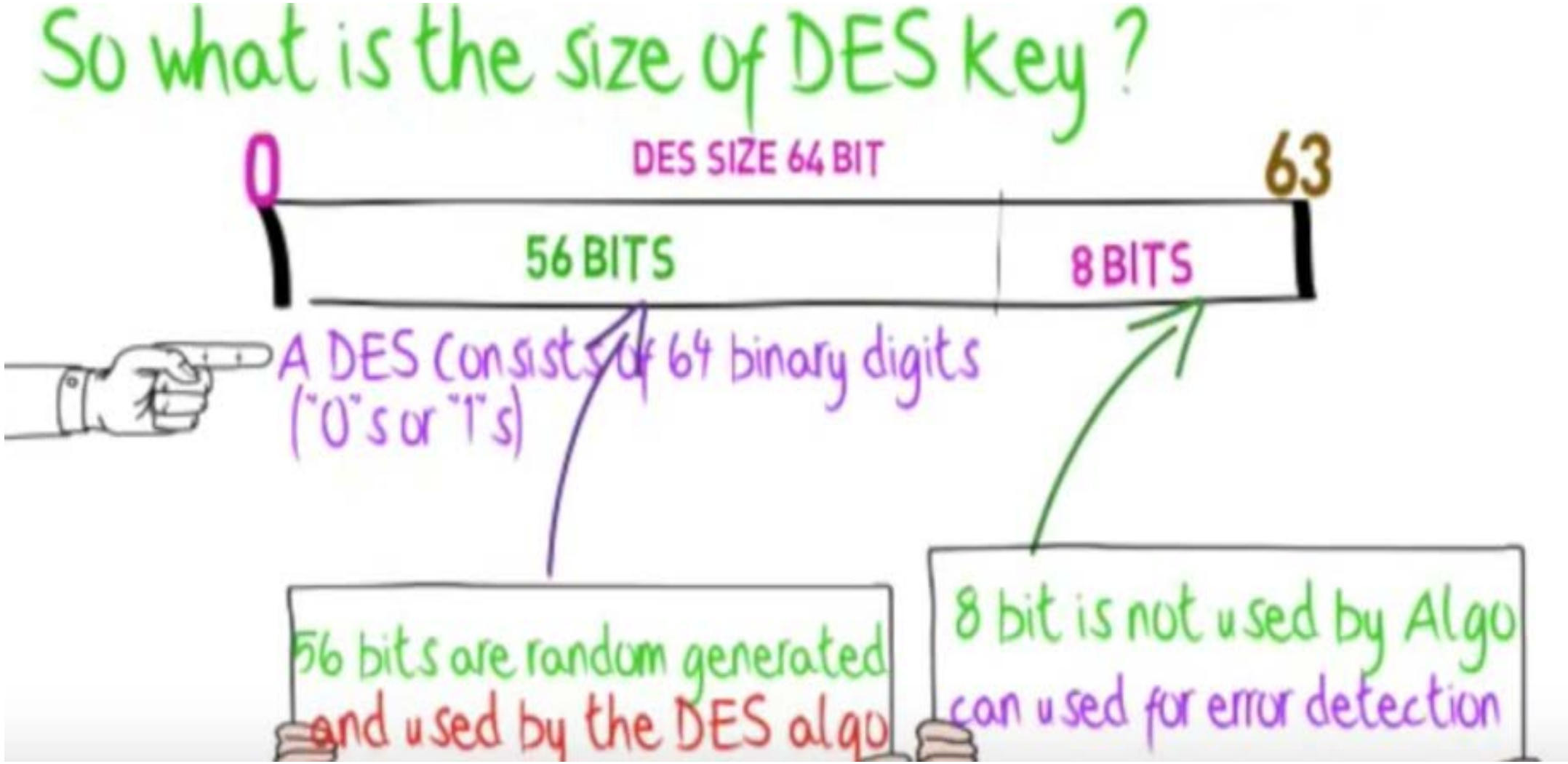
DES

- A **DES** egy úgy **nevezett blokkrejtjelező** (blokk alapú) titkosító eljárás (block cipher), ez azt jelenti, hogy a bemenő adatokat **meghatározott méretű blokkokra** osztja, tipikusan az **utolsó blokk kiegészítésével**, hogy az is elérje a szükséges blokkméretet
- A DES estében ez a **blokkméret 64** bit azaz a DES **64 bites input** blokkokat fogad be és **64 bites titkosított** szöveget bocsát ki.
- Ezt az eljárást ismétlik meg minden 64 bites blokkon.
- A DES titkosító kulcsának hossza **eredetileg 56 bit volt**, és ugyan azt a kulcsot, valamint ugyan azt az algoritmust, használta a titkosításhoz és a megfejtéshez is.
- A DES kulcs hossza 64 bit lett amiből 8 hiba javító bit

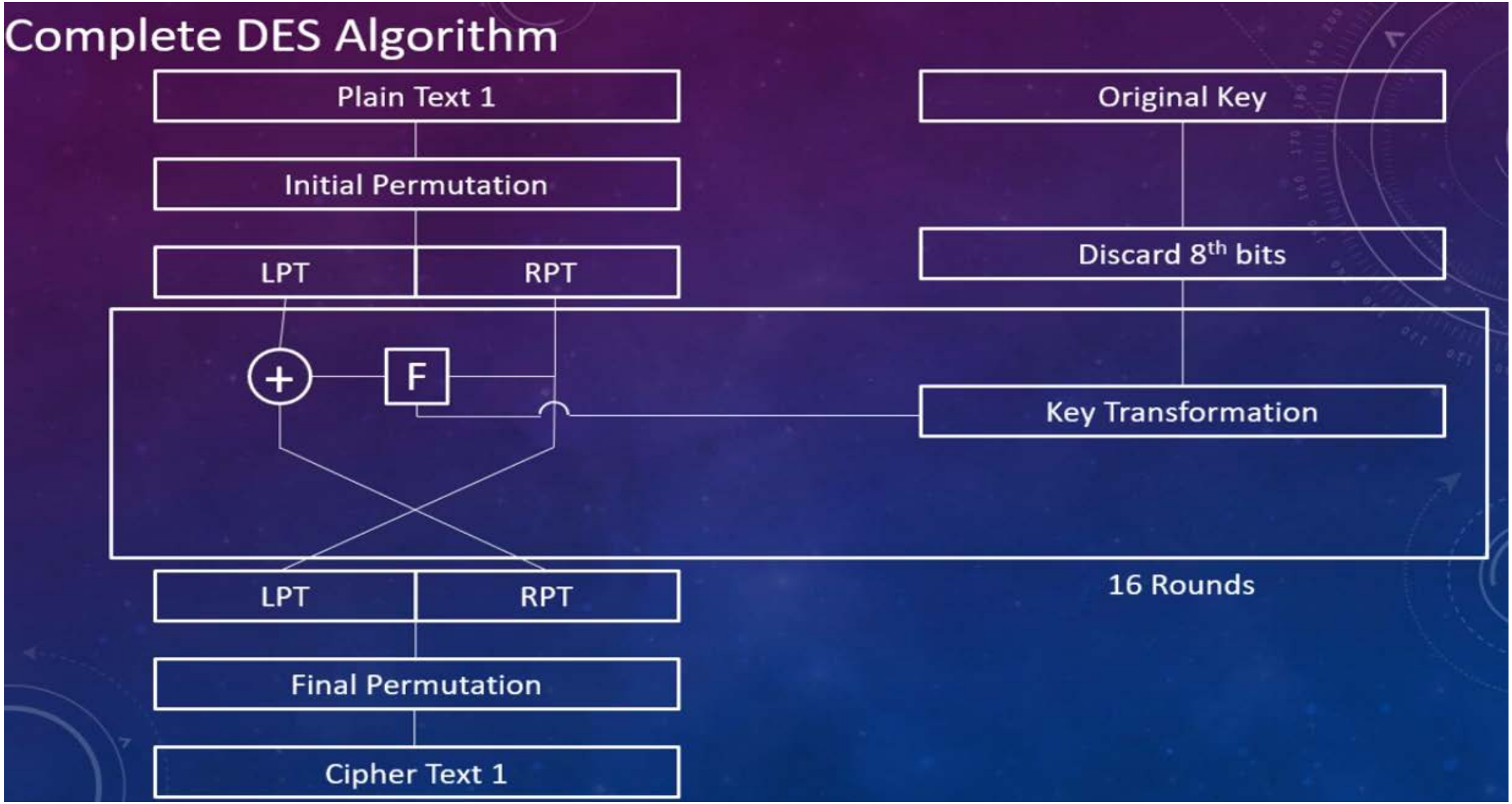
DES elvi működés blokk vázlat



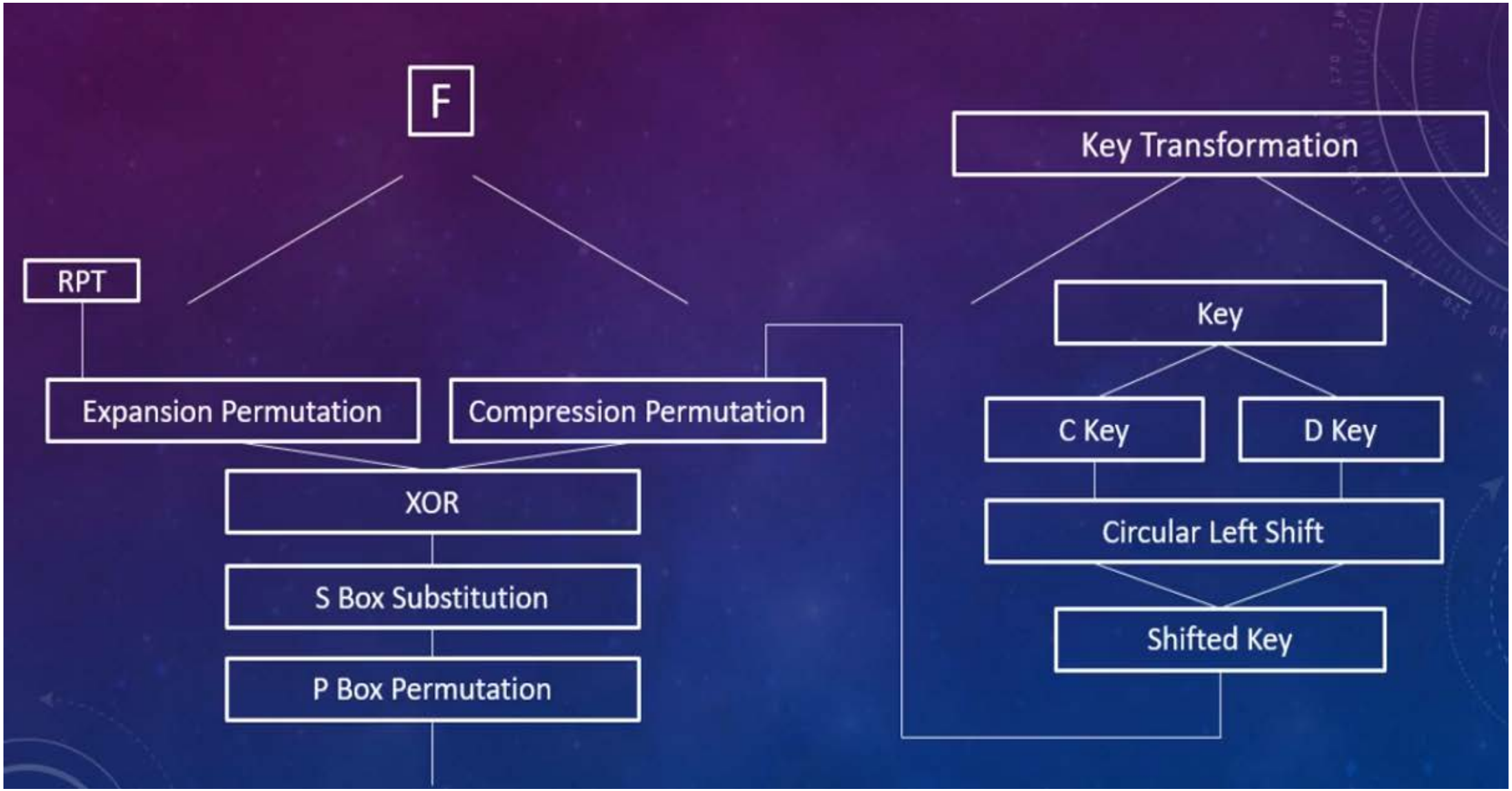
DES kulcs


So what is the size of DES key ?
0
DES SIZE 64 BIT
63
56 BITS
8 BITS
A DES consists of 64 binary digits
("0"s or "1"s)
56 bits are random generated
and used by the DES algo
8 bit is not used by Algo
can used for error detection

DES permutációs algoritmus



DES permutáció és helyettesítés algoritmus



Az AES blokk titkosító

- a DES szabványt le **kellett** váltani, mert
  - az 56 bites kulcs brute-force módon való feltörését több kisérlet is igazolta (distributed.net, Deep Crack)
  - kriptoanalízisen alapuló támadások is ismertek rá (igaz csak elméletiek)
- a TDES helyettesítheti, de lassú, és a blokkméret továbbra is kicsi (64 bit)
- az USA-ban a NIST újabb pályázatot hirdetett 1997-ben
  AES = Advanced Encryption Standard-re
  (Továbbfejlesztett titkosítási szabványra)

Az AES követelményei

- titkos kulcsú szimmetrikus blokk titkosító
- 128-bites blokk, 128/192/256-bites kulcs
- erősebb és gyorsabb legyen a TDES-nél
- hatékony megvalósíthatóság
  - számítási teljesítmény
  - kód és adatmemória szükséglet
  - előkészítő számítások (pl. kulcsszervezés)
- flexibilitás az egyes platformok és későbbi fejlesztések tekintetében
- aktív működés 20-30 évre (+ archív használat)

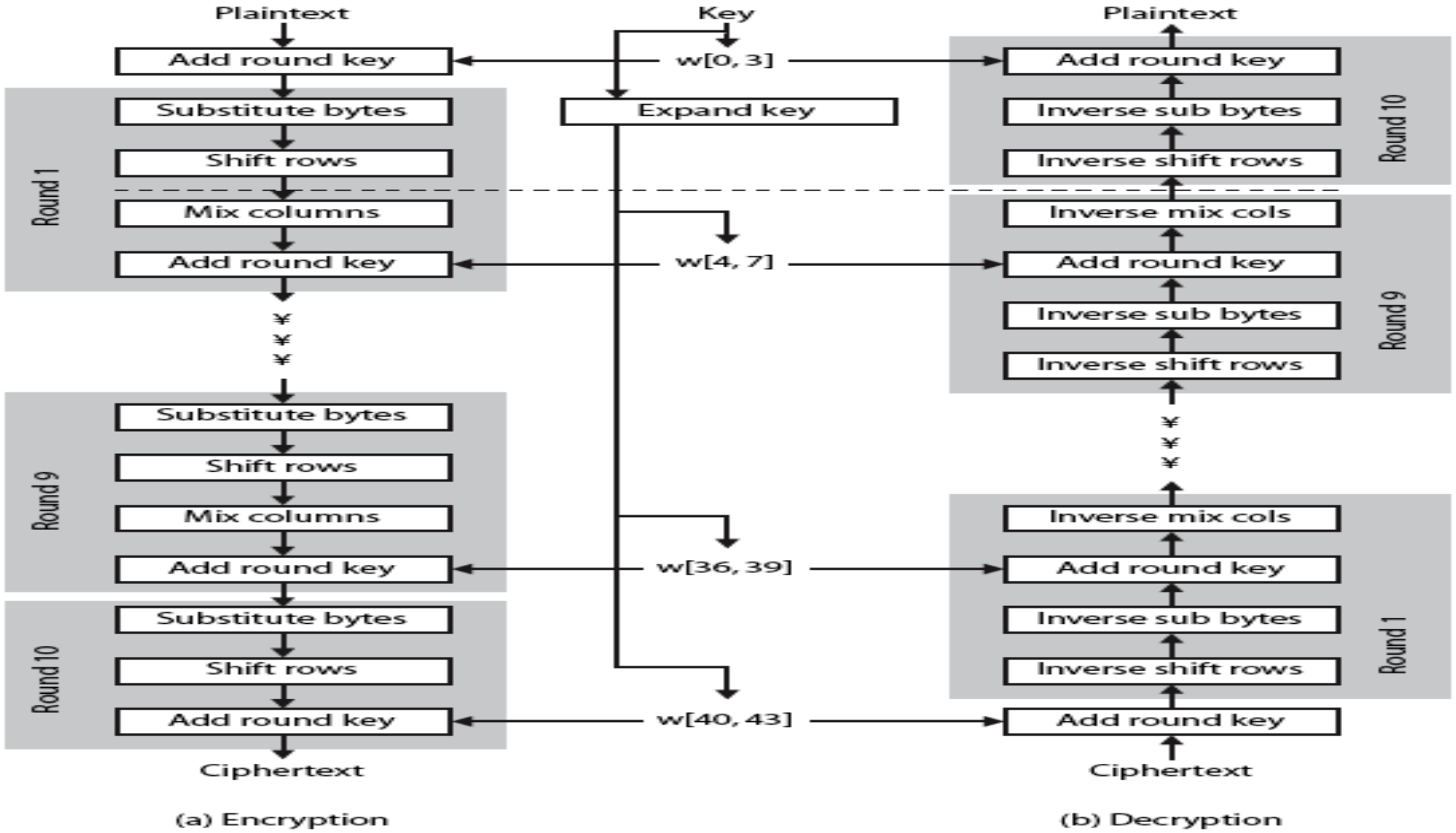
Az AES titkosító - Rijndael

- tervezői: Joan Rijmen és Vincent Daemen (Belgium)
- a Rijndaelben mind az adatblokk mind a kulcshossz 128-256 bit között 32 bitenként változtatható
- AES szabványos kulcsok: 128/192/256 bit,
- AES szabványos adatblokk 128 bit
- **iteratív** S-P hálózat, de nem **Feistel** struktúrás
  - az adatokat 4 sor x 4 oszlopos adatblokkokban tárolja, melyekben egy-egy bájtot tárol
  - minden körben az egész adatblokkon dolgozik
- úgy tervezték, hogy:
  - minden ismert támadásnak ellenálljon
  - gyors és tömör kód írható rá sok processzor típuson
  - átlátható terve legyen

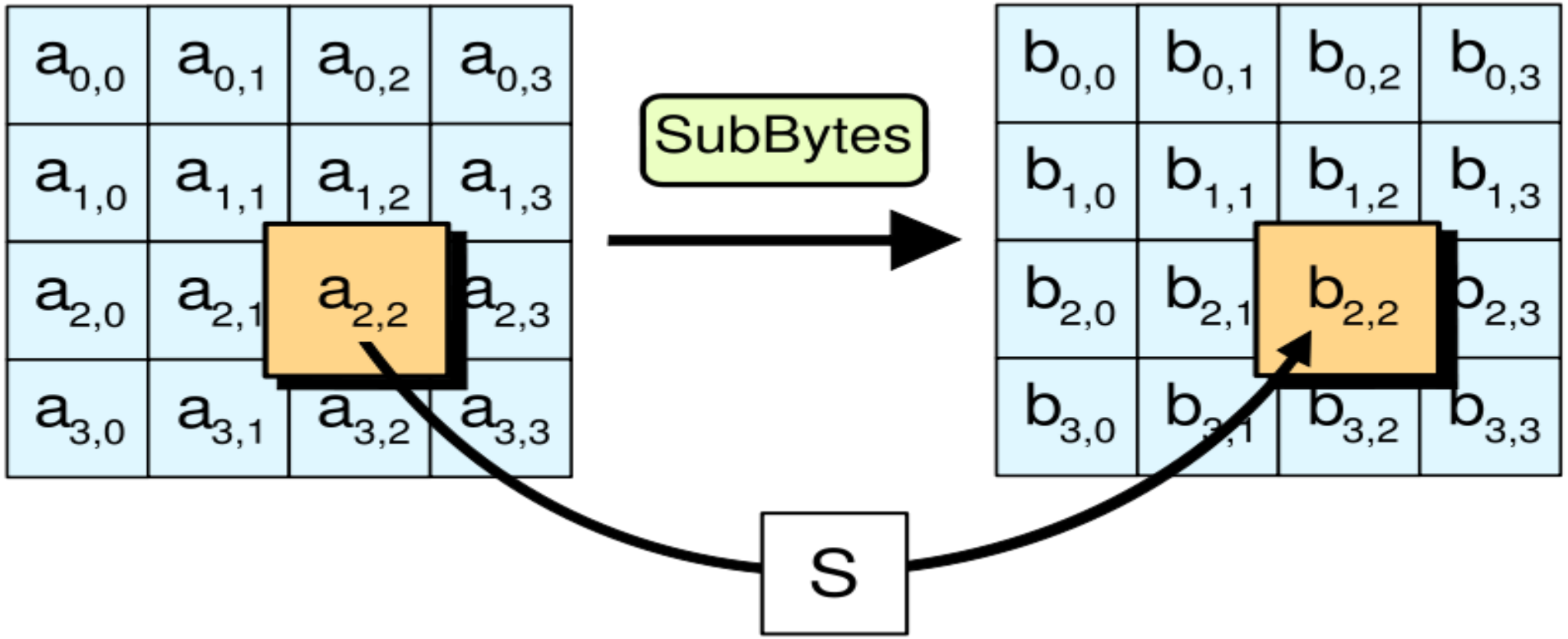
Az AES

- az 4 x 4 bájtos adatblokk egy tartalmát **állapotnak** (state) hívjuk = 4 db 32 bites **szó** = 128 **bit**
- a kulcsot is 32 bites szavak oszlopaira bontjuk ki melyekből körönként szintén 4 db-ot használunk fel
- **Lépések (kulcsméret függvényében) 10/12/14 körben:**
  - byte substitution: (1 közös S-dobozos a bájthelyettesítés)
  - shift rows: (a sorok elforgatása különböző mértékben)
  - mix columns: (az oszlopok átalakítása mátrixszorzással)
  - add round key: (XOR művelet bájtonként a körkulccsal)
- az első kör előtt még egy **add round key** van
- az utolsó körben a **mix columns** kimarad
- könnyen invertálható és implementálható XOR-ok és műveleti táblázatok segítségével

AES titkosítás/megfejtés

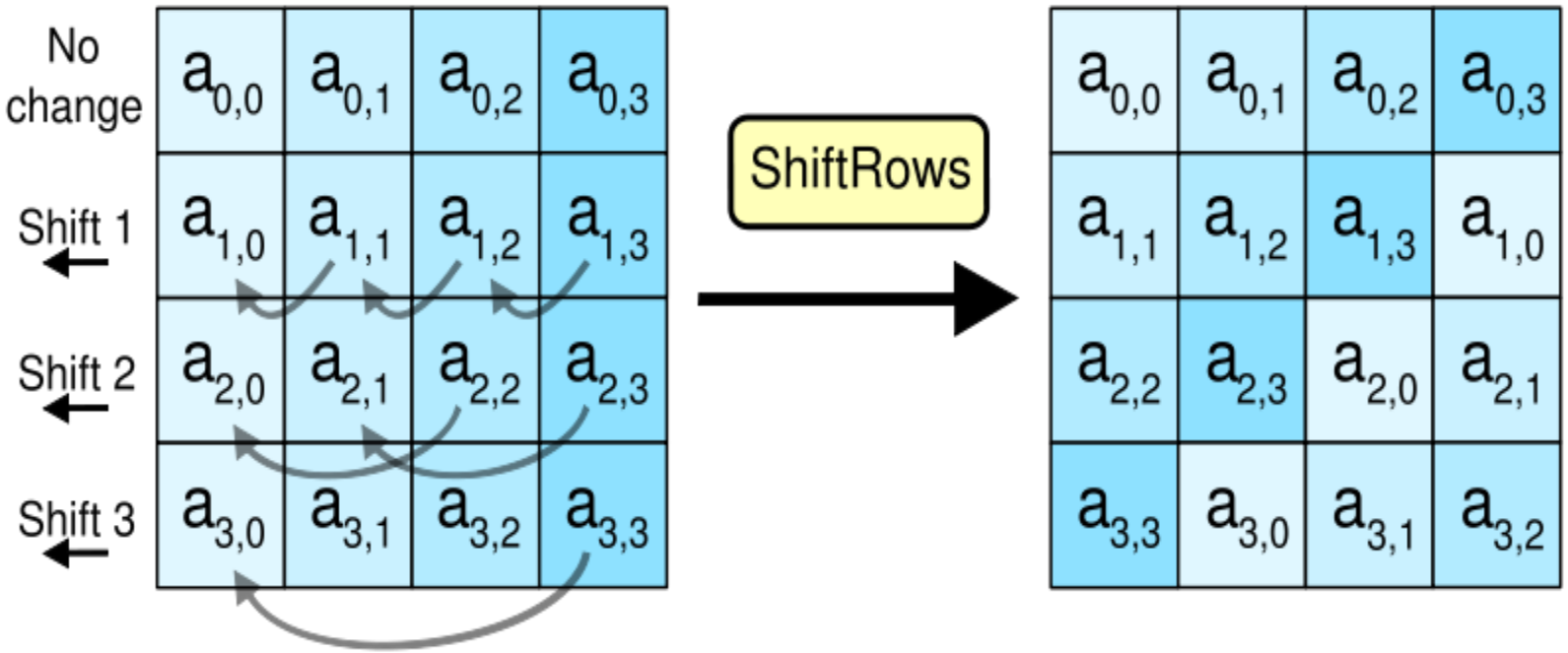


1. SubBytes() (bájt helyettesítés)



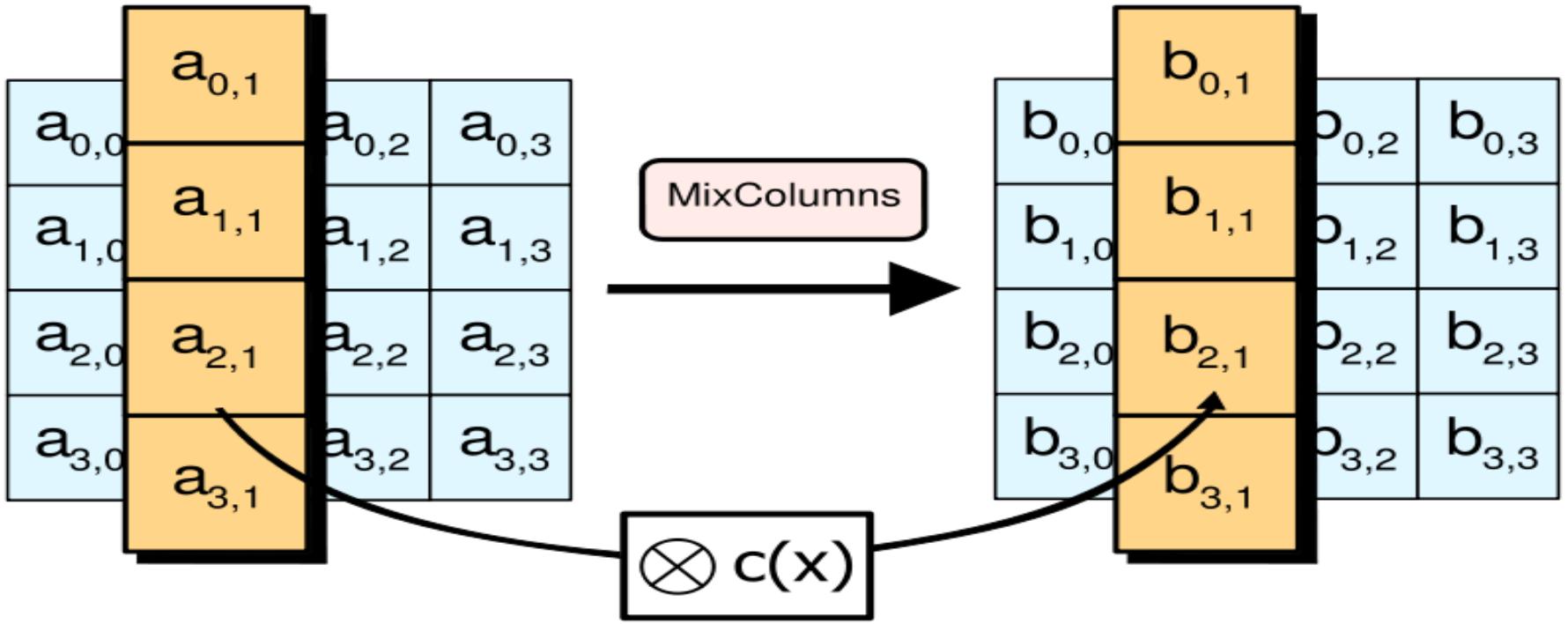
Minden bájt helyettesítése egy közös (8 bit -> 8 bites)
S-doboz segítségével.

2. Shift Rows() sorok elforgatása



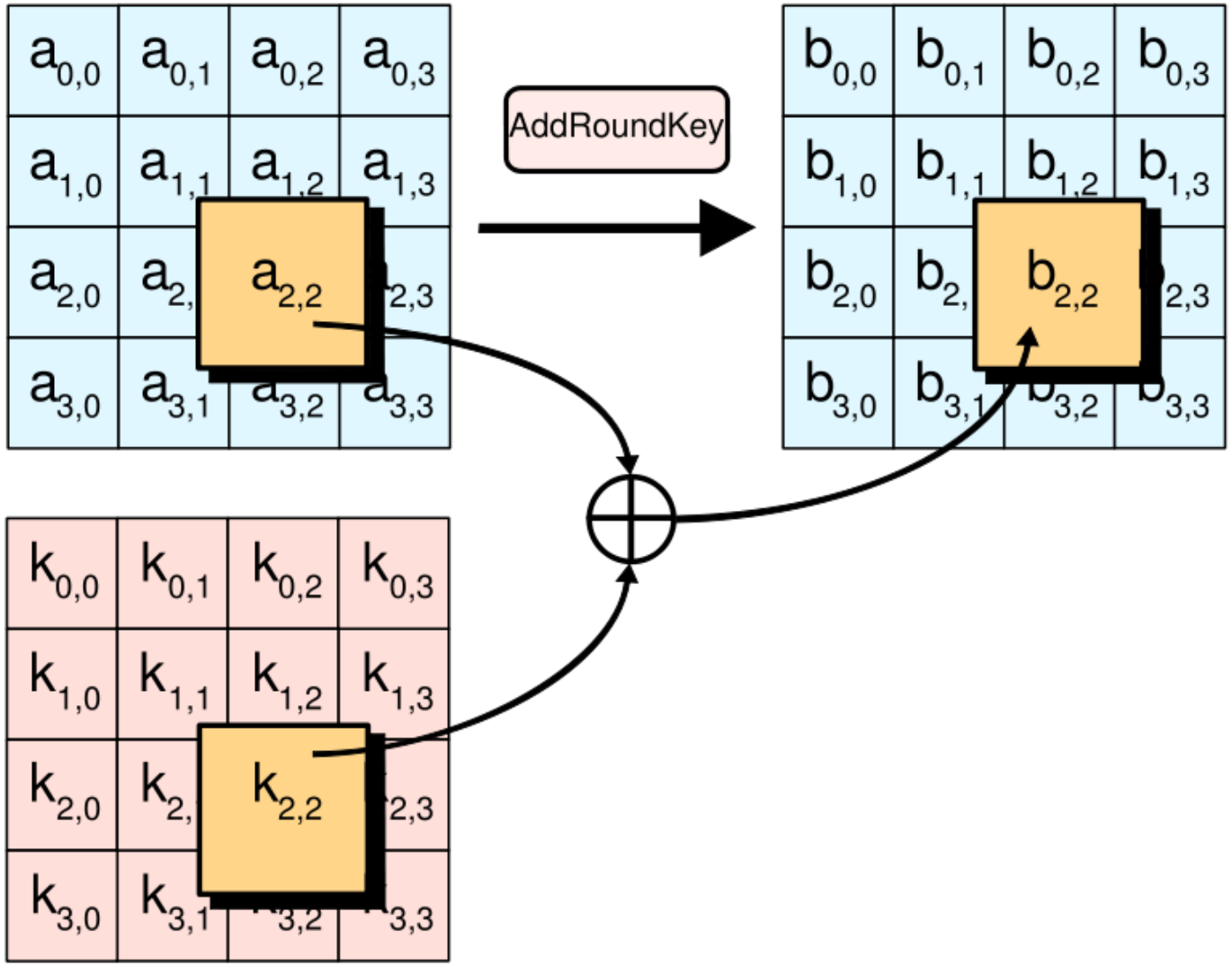
- permutációs lépés (keverés)
- a bájtok körkörös forgatása balra
- a megfejtéskor egyszerűen jobbra forgatunk
- mivel az állapotokban a bájtokat oszloponként tekinthetjük, ez a lépés az oszlopok adatait keveri össze
- minden oszlop kap bájtot minden oszlopból

3. MixColumns() (oszlopok összekeverése)



- az oszlopokat külön-külön helyettesítjük
- minden helyettesített bájt a keverendő oszlop mind a 4 bájtjának függvénye

4. AddRoundKey() (körkulcs hozzáadása)



4. AddRoundKey() (körkulcs hozzáadása)

- annyira egyszerű, amennyire csak lehet
- XOR az állapot és a 128 bites körkulcs között
- szintén oszloponként haladva (persze bájt szinten párhuzamosítható)
- inverze a megfejtéskor ugyanez
- mivel a XOR önmaga inverze, csak a kulcsok sorrendjét kell megfordítani

Az AES biztonsága

- a módszer feltörésének minősül minden olyan módszer, ami pl. a tejes kipróbálás átlagos 2^{127} titkosítási műveleténél kevesebbel fejti meg a 128 bites kulcsot
- mondjuk egy 2^{120} műveletigényű, 2^{100} választott nyíltszöveget igénylő módszer is, ami a gyakorlatban biztos, hogy kivitelezhetetlen
- jelenleg 10 vagy több körös megoldásban **nem ismert AES törés** csupán az AES kevesebb körrel rendelkező változatatira van ilyen:
  - ismert nyílt szövegű támadás
    - 7 körös AES-128-ra
    - 8 körös AES-192-re és AES-256-ra
  - hasonló kulcsos támadás (related key attack)
    - 9 körös AES-256-ra (2009. aug, 2^{39} !!! ,2 hasonló kulccsal)
  - 2002-ben ugyan napvilágot látott egy spekulatív „XSL attack” nevezetű támadás de még nyitott kérdéses, hogy valóban alkalmazható-e elméletileg is az AES-re

Titkos kulcsú kriptográfia

- a hagyományos: szimmetrikus/titkos/egy kulcsú kriptográfiában egy titkosító/megfejtő kulcs van
- ha nem is szó szerint egyezik meg a kettő, a titkosító és a megfejtő kulcs, egymásból könnyen kiszámítható
- a kulcsot csak a feladó és a címzett ismeri
- a kulcs titokban tartásán alapszik a biztonság
- a feleknek előzetesen kommunikálni kell egymással a titkos kulcsot
- ez szimmetrikus, a felek szerepe egyenrangú: mindketten tudnak titkosítani és megfejteni is
- ezért nem védi a feladót a címzettel szemben attól, hogy a címzett a kapott üzenetet meghamísítva azt állítsa, hogy az a feladótól jött

Nyilvános kulcsú kriptográfia (Public-Key Cryptography)

- talán a legjelentősebb találmány a kriptográfia 3000 éves történetében
- **két kulcs van**
  - egy **nyilvános (public key)**
  - egy **magán (private key)** /néhol: **saját kulcs** v. **titkos kulcs**/
- **a nyilvános kulccsal lehet titkosítani**
- de az üzenetet **csak a magánkulccsal lehet megfejteni**
- így például maga a küldő sem tudja visszafejteni az üzenetet, ha mondjuk elfelejtette, hogy mit titkosított

Nyilvános kulcsú kriptográfia II (Public-Key Cryptography)

- **a nyilvános kulcsot nyilvánosságra lehet hozni** és legalább a küldő számára nyilvánosságra kell hozni (de ez nem igényeli hogy biztonságos kommunikáció legyen)
- bárki lehet feladó, aki a nyilvános kulcsot megkapja
- a számelmélet számítási szempontból ,,egyik irányban nehéz -- másik irányban könnyű'' problémáin alapszik (pl. faktorizáció, diszkrét log.)
- kiegészíti és nem helyettesíti a titkosított kulcsú kriptográfiát

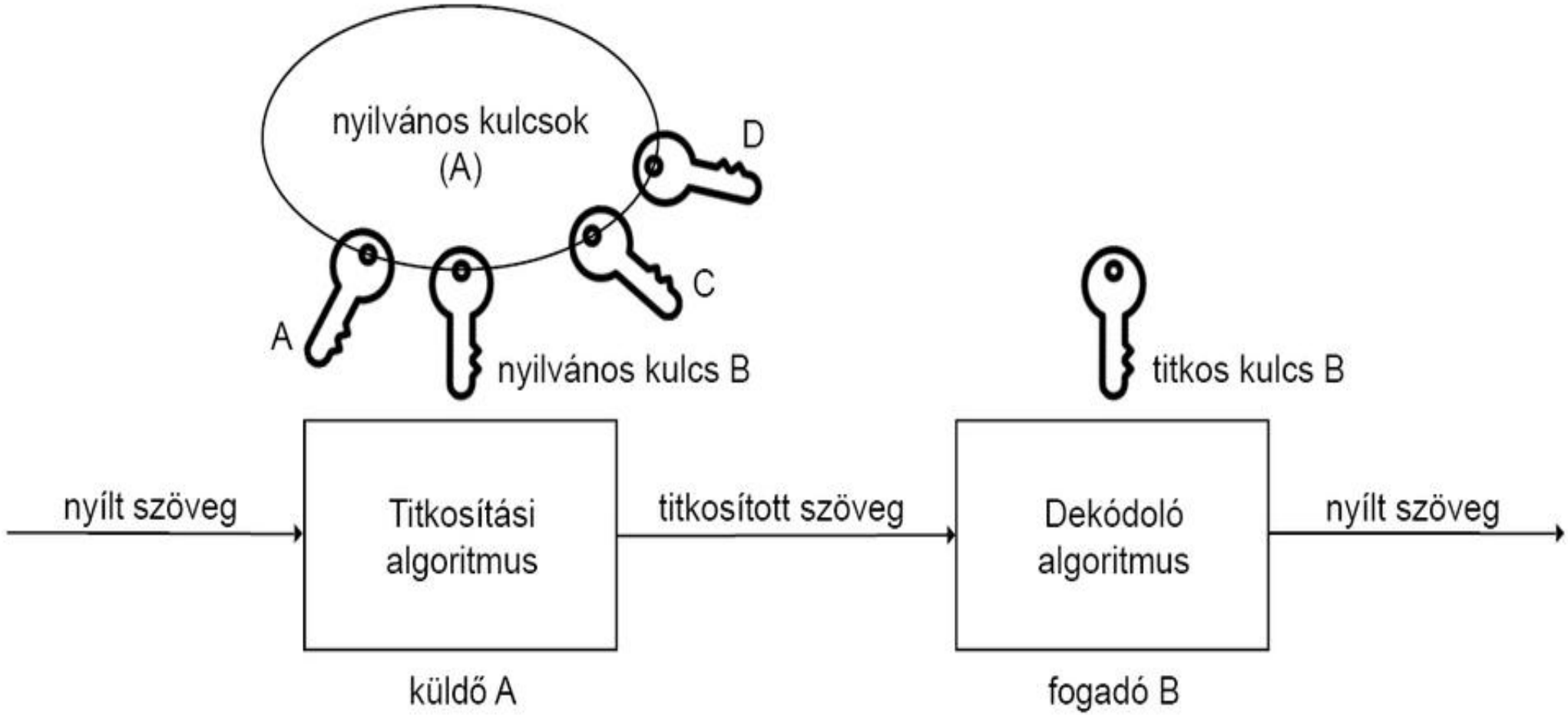
Miért jó a nyilvános kulcsú kriptográfia?

- a titkos kulcsú kriptográfia két alap problémájára ad választ:
  - **kulcselosztás**
  - **elektronikus aláírások**
- az első **nyilvános** publikációja:
  Whitfield Diffie és Martin Hellman (Stanford), 1976
  - ismert volt, bár titokban tartották 1999-ig: James Ellis (UK), 1970
  - sőt állítólag az NSA már a 60-as évek közepén ismerte

Public-Key Cryptography

- nyílt kulcsú/két kulcsú/aszimmetrikus titkosításnak is nevezik
- a kulcsok szerepe:
  - a nyilvános kulcsot **titkosításra** és a magánkulccsal készített **aláírás ellenőrzésére** lehet használni
  - a magánkulccsal (amit csak a címzett ismer) a **megfejteni** lehet, és **aláírást készíteni** természetesen másik irányú titkos vagy nem titkos üzenetküldéshez
- **a felek szerepe aszimmetrikus**:
  a nyilvános kulcs tulajdonosa (a feladó)
  - **csak titkosítani és aláírást ellenőrizni tud**,
  - **megfejteni vagy aláírni nem**
- ezért aláíráshoz magánkulcs kell, de üzenet titkosításához elég a küldő nyilvános kulcsát ismerni

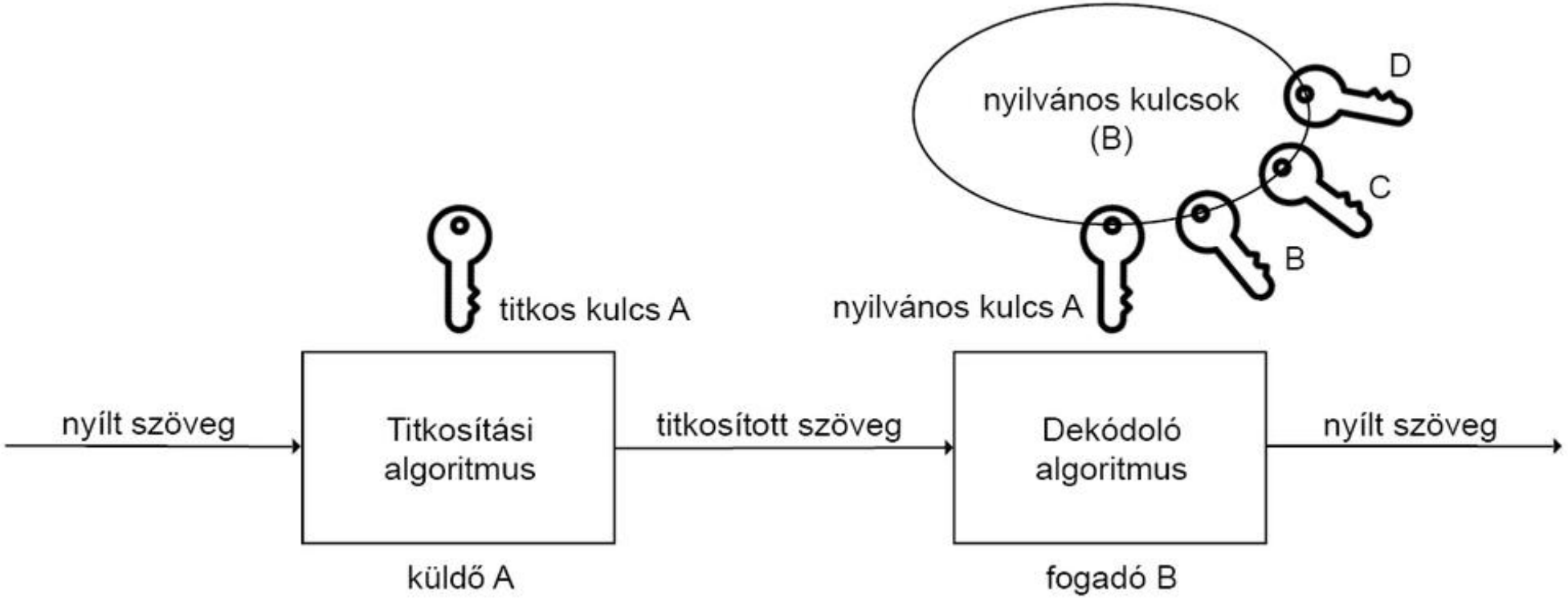
A nyilvános kulcsú titkosítás vázlata



A két kulcs viszonya

- Feltételek a nyilvános kulcsú titkosítás működéshez:
  - a nyilvános kulcs ismeretében hatékonyan lehet titkosítani
  - a magánkulcs ismeretében hatékonyan lehet üzenetet megfejteni
  - jelenlegi algoritmusainkkal reménytelenül sok ideig tart a nyilvános kulcsból a magánkulcsot kiszámítani
    (a titkosító/megfejtő algoritmust ismeretét persze feltételezzük)
  - a magán kulcs ismerete nélkül szintén reménytelen számítási feladat az üzenet megfejtése
  - hatékonyan tudunk véletlen nyilvános-magán kulcspárokat generálni
- néhány algoritmusnál (pl. RSA) hasznos, hogy
  - a magánkulccsal is lehet titkosítani, ami csak a nyilvános kulccsal fejthető meg (ezen alapul az digitális aláírás)

Nyilvános kulcsú aláírás (elvi vázlat)



A nyilvános kulcsú rejtjelezésen belüli **azonosítás** inverz módon Az azonósítás (fordítottan) történik. Amennyiben a nyílt szöveget az A küldő titkosítja saját titkos kulcsával, a megfejtés csak az A nyílt kulcs használatával lehetséges, ami azt jelenti, hogy a szöveget az A küldő rejtjelezte.

A nyilvános kulcsú kriptográfia alkalmazásai

- 3 kategóriába osztható:
  - **titkosítás/megfejtés** (bizalmasságot ad)
  - **elektronikus aláírások** (hitelesítést ad)
  - **kulcscsere**
    (kapcsolatkulcsok (session keys) cseréjére)
- néhány algoritmus mindhárom feladatra alkalmas, mások csak egy-két célra használhatók

A nyilvános kulcsú rendszerek biztonsága I

- nem biztonságosabbak vagy kevésbé biztonságosak a titkos kulcsú rendszereknél, a biztonság a kulcs hosszától is függ
- mint a titkos kulcsú rendszereknél itt is a **teljes kipróbálás** (brute force) feltörés legalább is elméletben mindig lehetséges
- de a gyakorlatban használt kulcsok a teljes kipróbálás meghiúsításánál jóval hosszabbak (> 512 bitesek)
- mert a titkosítás alapját képező számelméleti probléma nehéz irányának (pl. faktorizáció) kiszámítása ellen kell védekeznünk
- pl. 512-bit RSA ≈ 64-bit DES, 1024-bit RSA ≈ 80-bit DES

A nyilvános kulcsú rendszerek biztonsága II

- a biztonság a „könnyű irány” (titkosítás) és a „nehéz irány” (feltörés) közötti **elég nagy számítási különbségen** alapszik
- pl. RSA esetében
  - könnyű irány = szorzás, (illetve hatványozás mod p)
  - nehéz irány = faktorizáció (prímtényezőkre bontás)
- általánosságban a ,,nehéz irány’’ is algoritmussal megoldható, de **elég nehézzé kell tennünk** ahhoz, hogy a gyakorlatban kivitelezhetetlen legyen
- ehhez nagy (több százjegyű) számokra van szükség
- ezért a nyilvános kulcsú kriptográfia **jóval lassabb a titkos kulcsúnál**

RSA

- **R**ivest, **S**hamir & **A**dleman (MIT), 1977
- a legismertebb és legelterjedtebb nyilvános kulcsú algoritmus
- véges test feletti hatványozáson alapszik valamilyen **n** modulusra nézve $a^b \pmod n$ kiszámításának időigénye $O((\log n)^3)$ ez **polinomiális** a bemenet hosszának ($\log n$) függvényében /könnyű/
- a modulus nagy szám (pl. 1024 bit)
- a biztonságát a faktorizáció nehézsége adja
  - erre ma csak **superpolinomiális algoritmusok ismertek** /nehéz/, pl. GNFS időigénye

$$O\left(\exp\left(\left(\frac{64}{9}n\right)^{\frac{1}{3}}(\log n)^{\frac{2}{3}}\right)\right)$$ ahol n a bemenet hossza

Az RSA biztonsága

- lehetséges támadási típusok ellene:
  - a kulcsok teljes kipróbálása (kivitelezhetetlen a számok nagysága miatt)
  - matematikai támadások
  - időméréses támadások (a megfejtő algoritmuson)
  - választott titkos szöveg alapú támadások

Kulcsgondozás (Key Management)

- a nyilvános kulcsú titkosítás segít megoldani a kulcselosztás problémáját

Ennek két aspektusa van:

- I. a nyilvános kulcsok szétosztása
  - azaz eljuttatása mindazokhoz, akik üzenetet küldhetnek a titkos kulcs birtokosának
- II. a nyilvános kulcsú titkosítás használata titkos kulcsok cseréjére
  - hogy az így biztonságosan eljuttatott titkos kulccsal gyors szimmetrikus titkosítással lehessen kommunikálni

A nyilvános kulcsok szétosztása (Distribution of Public Keys)

- **A nyilvános kulcsú titkosítás, semmit sem ér, ha a küldő nem győződik meg arról, hogy a nyilvános kulcs hiteles**, azaz valóban a címzetté.
- Mert egy támadó a címzett nyilvános kulcsát a saját nyilvános kulcsára hamisítva elolvashatja az üzenetet.
- Sőt, ha elfogja és megfejti az üzenetet, utána még a címzett valódi kulcsával titkosítani is tudja, majd továbbítani a címzettnek. Így még észrevétlen is maradhat.
- De pont azért szeretnénk a nyilvános kulcsú rendszert használni, hogy ne kelljen a nyilvános kulcsot előre titokban kommunikálni.
- Hogyan oldható ez meg mégis? Kemény dió.

A nyilvános kulcsok szétosztása (Distribution of Public Keys)

Különböző technikák vannak használatban, melyek az alábbi 4 csoportba sorolhatók:

1. nyilvános kihirdetés (public announcement)
2. nyilvános hozzáférésű katalógus (publicly available directory)
3. nyilvános kulcsszolgáltató (public-key authority)
4. nyilvános kulcs tanúsítványok (public-key certificates)

1. Nyilvános kihirdetés

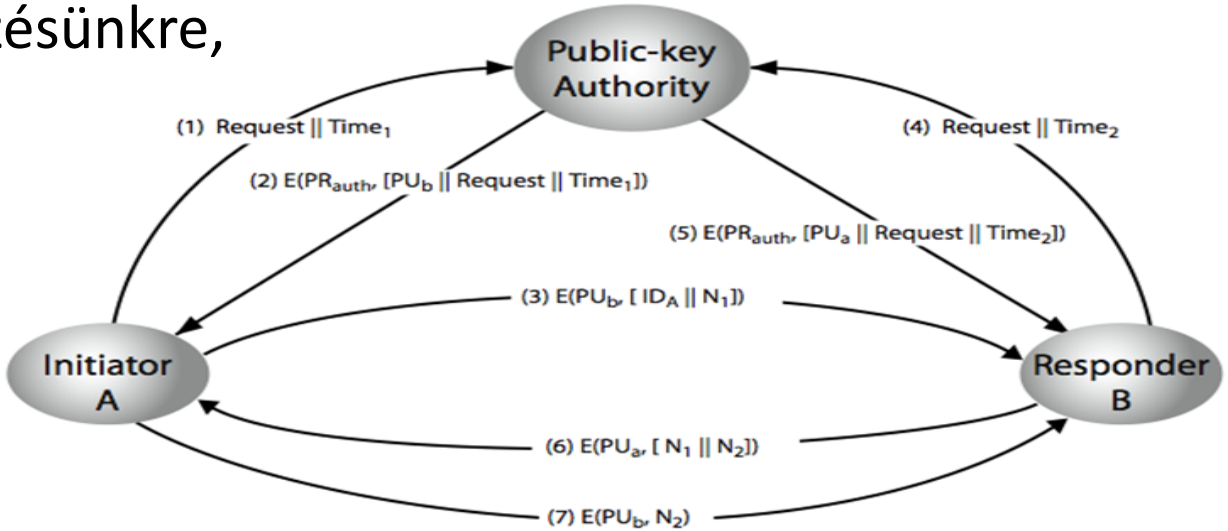
- a felhasználók közhírré teszik a nyilvános kulcsukat a várható feladóknak, vagy nagyobb közösségeknek pl.
  - e-mailhez csatolva
  - hírcsoportokra, levelezési listákra küldve
  - (személyes) weblapra feltéve, stb.
- a legnagyobb hátránya, hogy **könnyen hamisítható**:
  - bárki könnyen készíthet nyilvános-magán kulcspárt, amit más nevében adhat ki
  - amíg a csalást a címzett észre nem veszi, és a feladókat nem figyelmezteti, a támadó szabadon olvashatja a neki küldött üzeneteket

2. Nyilvános hozzáférésű katalógus (Publicly Available Directory)

- nagyobb biztonság érhető el, a kulcsok nyilvános katalógusba történő regisztrációjával
- a katalógust működtető egyénben/szervezetben a résztvevőknek meg kell bízni
- a katalógus:
  - **{név, nyilvános kulcs}** bejegyzéseket tartalmaz
  - a résztvevők személyesen v. titkosan regisztrálják kulcsaikat a katalógusba
  - a katalógusban tárolt kulcsot bármikor ki lehet cserélni
  - a katalógust időnként közzéteszik
  - a katalógus elektronikusan is biztonságos kommunikációval elérhető
- hátrány: lehetséges hamisítás és a katalógus feltörése

3. Nyilvános kulcsszolgáltató (Public-Key Authority)

- a biztonságosság növelhető a kulcsok katalógusból történő kiadásának szigorúbb ellenőrzésével
- tulajdonságai ua.-ok mint a katalógus esetében
- a felhasználóknak ismerni kell a katalógus nyilvános kulcsát
- a kért nyilvános kulcsok a katalógusból biztonságos interaktív kapcsolattal kérhetők le
  - de csak akkor szükséges valós idejű hozzáférés a katalógushoz, ha a kulcsok még nem állnak rendelkezésünkre,

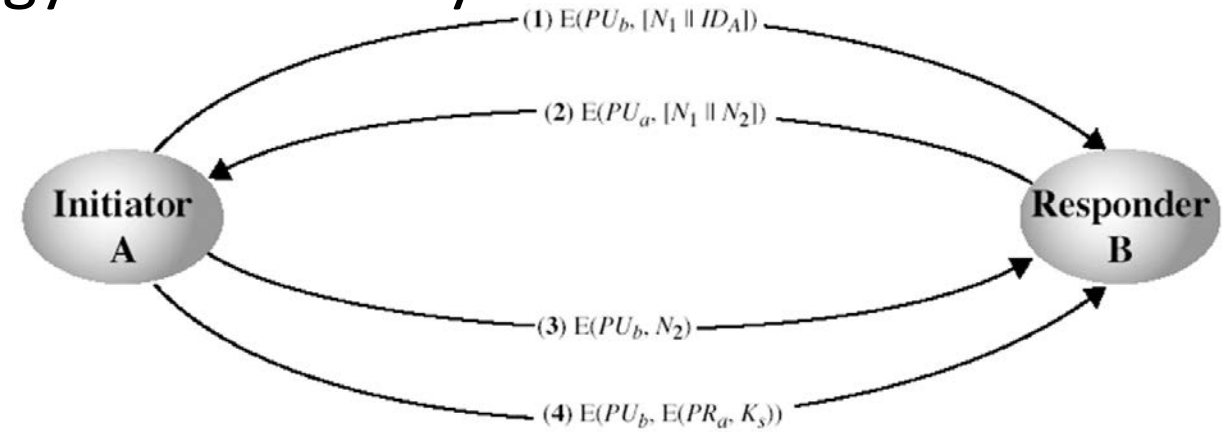


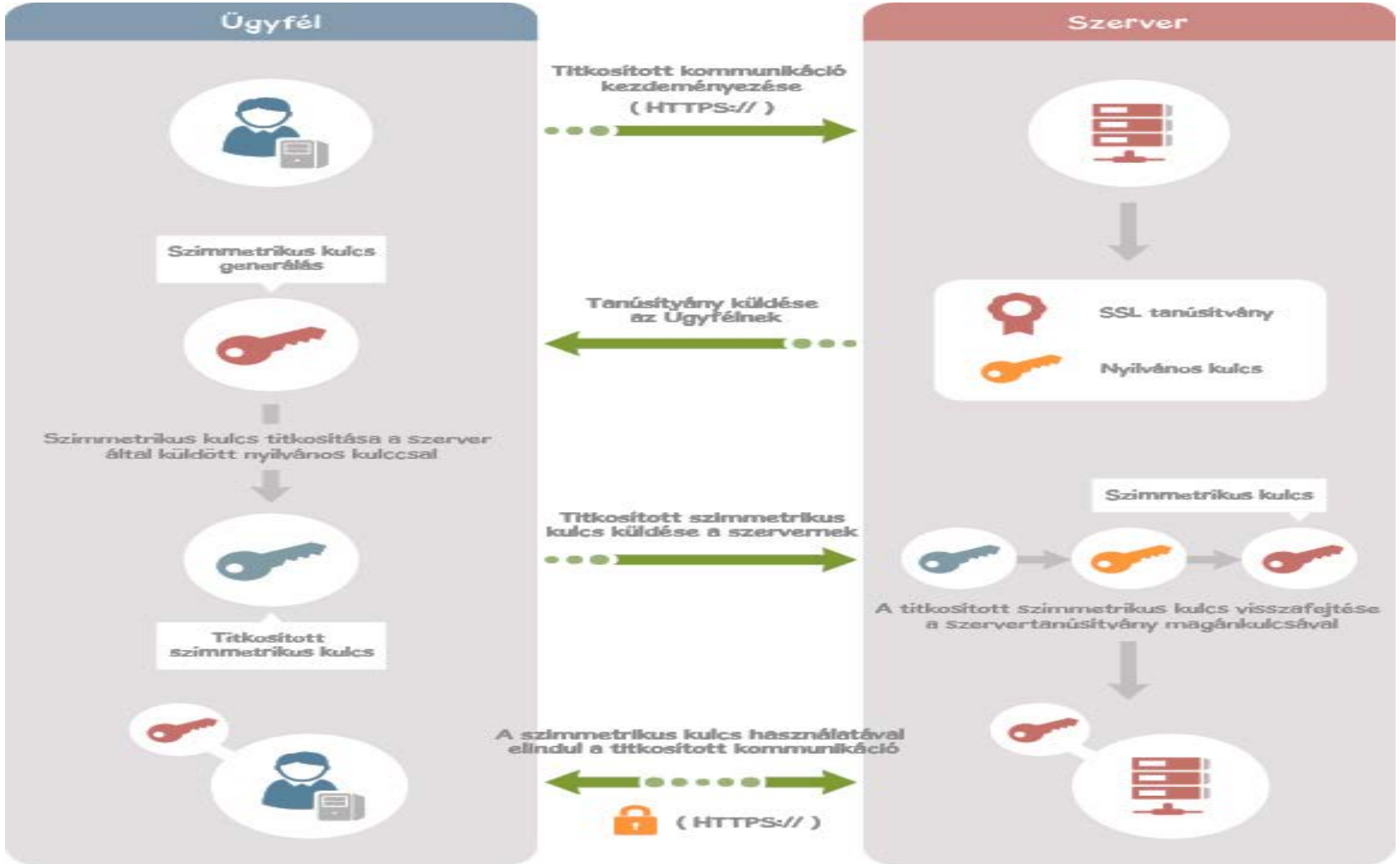
4. Nyilvános kulcs tanúsítványok (Public-Key Certificates)

- a kulcsszolgáltató használata szűk kereszt-metszet lehet: elérhetőség, feltörhetőség
- a **tanúsítvány** biztosítja a kulcs-cserét a kulcsszolgáltató valós idejű elérése nélkül
- a tanúsítvány a **felhasználói azonosító** és a **nyilvános kulcs** összetartozását igazolja
- a tanúsítványt **aláírja** a nyilvános kulcs szolgáltató vagy hitelesítés-szolgáltató Certificate Authority (**CA**)
- bárki ellenőrizheti, aki ismeri a szolgáltató nyilvános kulcsát
- Például az **X.509**-es tanúsítvány szabványt használja az IPSec, SSL, SET, S/MIME (ld. később)

II. A nyilvános kulcsú titkosítás használata titkos kulcsok cseréjére

- az előző módszerekkel megszerzett nyilvános kulcsot felhasználhatjuk titkosításra vagy hitelesítésre
- de a nyilvános kulcsú algoritmusok lassúak
- ezért inkább a gyors szimmetrikus módszerekkel akarjuk magát az üzenetet titkosítani
- ehhez egy **kapcsolatkulcsra** (session key) **van szükség**
- ezt cserélik ki a felek egymás közt a nyilvános kulcsú titkosítás használatával

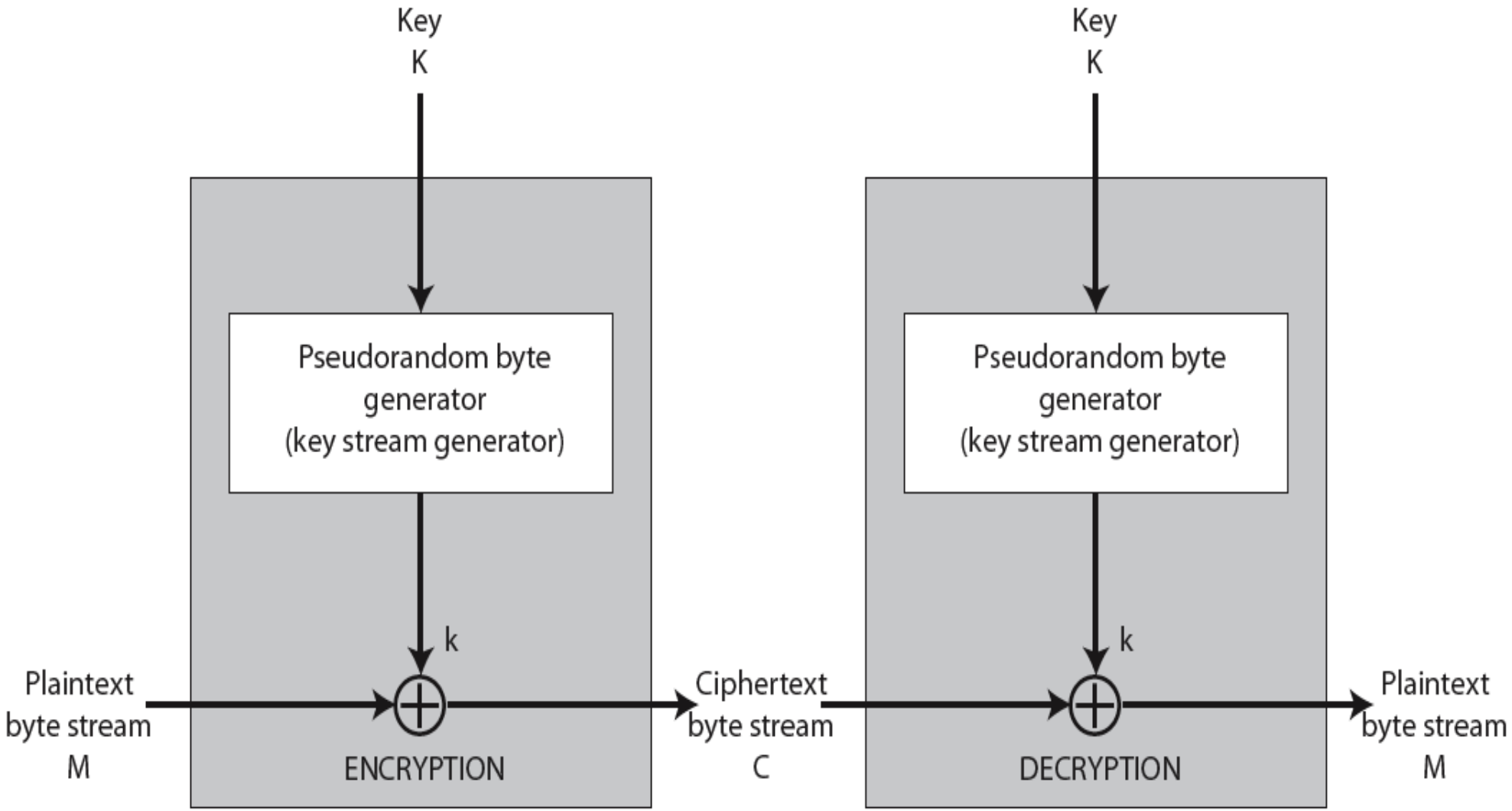



Ügyfél
Szerver
Titkosított kommunikáció kezdeményezése
(HTTPS://)
Szimmetrikus kulcs generálás
Tanúsítvány küldése az Ügyfélnek
SSL tanúsítvány
Nyilvános kulcs
Szimmetrikus kulcs titkosítása a szerver által küldött nyilvános kulccsal
Szimmetrikus kulcs
Titkosított szimmetrikus kulcs küldése a szervernek
A titkosított szimmetrikus kulcs visszafejtése a szervertanúsítvány magánkulcsával
Titkosított szimmetrikus kulcs
A szimmetrikus kulcs használatával elindul a titkosított kommunikáció
(HTTPS://)

Folyam titkosítók (Stream Chiphers)

- az üzenetet bitenként (bájtonként) dolgozza fel adatfolyamként
- ehhez a kulcsból egy álvéletlen (pseudo random) kulcsfolyamot (keystream) állít elő
- majd XOR műveletet végez vele bitenként
- a kulcsfolyam véletlensége teljesen megsemmisíti a nyílt szöveg statisztikai jellemzőit
  - $C_i = M_i \ XOR \ StreamKey_i$
- de ugyanazt a kulcsot tilos többször alkalmazni, különben feltörhető (nem úgy mint a blokktitkosítóknál.)

A folyam titkosítók általános felépítése



A folyam titkosítók tulajdonságai

Néhány tervezési elv:

- a kulcsfolyam hosszú periódusú legyen hosszabb ismétlődések nélkül
- állja ki a véletlenség statisztikai próbáit
  - ekkor a rejtjelezett szöveg is véletlennek látszó lesz
- egy elég hosszú kulcstól függjön (pl. ma >= 128 bit)
- magas lineáris kompexitású legyen

- az alkalmasan tervezett folyam titkosító ugyanolyan biztonságos lehet, mint az azonos kulcshosszú blokk titkosító (csak brute force technikával)
- de általában egyszerűbb (pár soros kód!) és gyorsabb

Az RC4 folyam titkosító

- tervezte Ron Rivest (RSA Security, 1987)
- az RSA üzleti titokként kezelte a kódját, de valaki elküldte a lev. listára 1994-ben
- nagyon egyszerű, így rendkívül hatékony
- változtatható kulcsméretű, bájtonként dolgozik
- az egyik legelterjedtebb (SSL/TLS, WPA, WEP) folyam titkosító
- a kulcsfolyamhoz a 0..255 számok (ál)véletlen permutációit használja
- a kulcsból előállít egy álvéletlen permutációt, majd
- az aktuális, permutáción kever még egyet, majd generál belőle egy bájtot, amivel XOR-olva titkosítja az aktuális bájtot

Az RC4 biztonsága

- az eddigi kutatások alapján biztonságos az ismert támadásfajtákkal szemben
  (persze elég hosszú kulcs esetén, >= 128 bit)
  - van néhány gyakorlatban kivitelezhetetlen kriptoanalízisen alapuló támadás
- nagyon nem lineáris titkosító
- mivel RC4 folyam titkosító, ezért **tilos a kulcsot újra felhasználni**
- ezzel van a fő probléma a WEP esetén,
- de ezt sérülékenységet a kulcsgenerálás gyengesége okozza és nem magát az RC4-et érinti