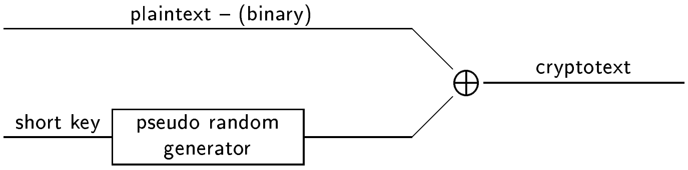
13. FROM CRYPTO-THEORY TO CRYPTO-PRACTICE

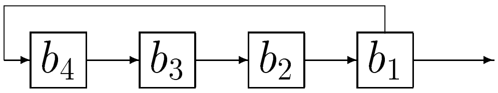
SHIFT REGISTERS

The first practical approach to ONE-TIME PAD cryptosystem.

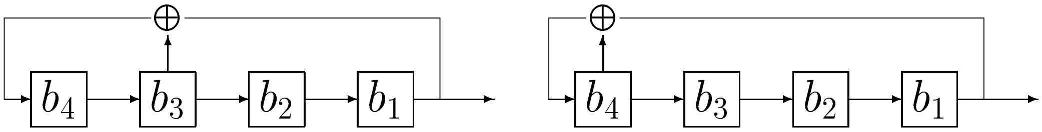
Basic idea: to use a short key, called “seed” with a pseudorandom generator to generate as long key as needed.



Shift registers as pseudorandom generators



linear shift register



period

b_4	b_3	b_2	b_1
0	1	0	0
1	0	1	0
0	1	0	1
0	0	1	0
0	0	0	1
1	0	0	0

b_4	b_3	b_2	b_1
1	1	0	0
1	1	1	0
1	1	1	1
0	1	1	1
1	0	1	1
0	1	0	1
1	0	1	0
1	1	0	1
0	1	1	0
0	0	1	1
1	0	0	1
0	1	0	0
0	0	1	0
0	0	0	1
1	0	0	0

period

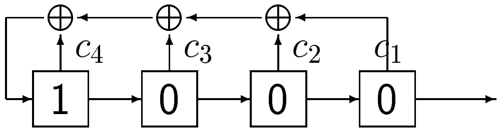
Theorem For every $n > 0$ there is a linear shift register of maximal period $2^n - 1$.

CRYPTOANALYSIS of linear feedback shift registers

Sequences generated by linear shift registers have excellent statistical properties, but they are not resistant to a known plaintext attack.

Example Let us have a 4-bit shift register and let us assume we know 8 bits of plaintext and cryptotext. By XOR-ing these two bit sequences we get 8 bits of the output of the register, say 00011110

We need to determine c_4, c_3, c_2, c_1 such that the above sequence is outputed by the shift register



state of cell 4	state of cell 3	state of cell 2	state of cell 1
c_4	1	0	0
$c_4 \oplus c_3$	c_4	1	0
$c_2 \oplus c_4$	$c_4 \oplus c_3$	c_4	1
$c_1 \oplus c_3 \oplus c_4 \oplus c_3 \cdot c_4$	$c_2 \oplus c_4$	$c_4 \oplus c_3$	c_4

$$\begin{array}{c} c_4 = 1 \\ c_4 \oplus c_3 = 1 \\ c_2 \oplus c_4 = 1 \\ c_1 \oplus c_3 \oplus c_4 \oplus c_3 \cdot c_4 = 0 \end{array} \quad \Rightarrow \quad \begin{array}{c} c_4 = 1, \\ c_3 = 0, \\ c_2 = 0, \\ c_1 = 1 \end{array}$$

How to make cryptoanalysts' task harder?

Two general methods are called diffusion and confusion.

Diffusion: dissipate the source language redundancy found in the plaintext by spreading it out over the cryptotext.

Example 1: A permutation of the plaintext rules out possibility to use frequency tables for digrams, trigrams.

Example 2: Make each letter of cryptotext to depend on so many letters of the plaintext as possible

Illustration Let letters of English are given by integers from $\{0, \ldots, 25\}$. Let the key $k = k_1, \ldots, k_s$ be a sequence of such integers.
Let

$$p_1, \ldots, p_n$$

be a plaintext.
Define for $0 \leq i \leq s, p_{-i} = k_{s-i}$ and construct the cryptotext by

$$c_i = \left(\sum_{j=0}^{s} p_{i-j} \right) \bmod 26, \quad 1 \leq i \leq n.$$

Confusion make the relation between the cryptotext and plaintext as complex as possible.

Example polyalphabetic substitutions.

History of DES

15. 5. 1973 National Burea of Standards published a solicitation for a new cryptosystem.

This lead to the development of

Data Encryption Standard

DES was developed at IBM, as a modification of an earlier cryptosystem Lucifer.

17. 3. 1975 DES was first published.

After heated public discussion DES was adopted as a standard on 15. 1. 1977.

DES has been reviewed by NBS every 5 years.

DES cryptosystem –
– Data Encryption Standard – 1977

A revolutionary step in secret-key cryptography:

Both encryption and decryption algorithms were made public.

Preprocessing A secret 56-bit key k_{56} is chosen.
A fixed+public permutation ϕ_{56} is applied to get $\phi_{56}(k_{56})$.
The first (second) part of the resulting string is taken to get a 28-bit block C_0 (D_0).
Using a fixed+public sequence $s_1, \ldots, s_{16}$ of integers 16 pairs of 28-bit blocks (C_i, D_i), $i = 1, \ldots, 16$ are obtained as follows:

C_i (D_i) is obtained from C_{i-1} (D_{i-1}) by s_i left shifts.

Using a fixed+public order 48-bit block K_i is created from C_i and D_i.

Encryption A fixed+public permutation ϕ_{64} is applied to a plaintext w to get $w' = L_0R_0$, where each L_0, R_0 has 32 bits. 16 pairs of 32-bit blocks $L_i, R_i, 1 \leq i \leq 16$, are designed using the recurrence:

$$L_i = R_{i-1},$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i),$$

where f is a fixed and public and easy-to-implement function.

The cryptotext $c = \Pi_{64}^{-1}(L_{16}R_{16})$.

Decryption $\phi_{64}(c) = L_{16}R_{16}$ is computed and then the recurrence

$$R_{i-1} = L_i$$
$$L_{i-1} = R_i \oplus f(L_i, K_i)$$

is used to get $L_i, R_i, i = 15, \ldots, 1, 0, w = \Pi_{64}^{-1}(L_0R_0)$

How fast is DES?

200 megabits can be encrypted per second using a special hardware.

How safe is DES?

Pretly good.

How to increase security using DES?

1. Use two keys for a double encryption.
2. Use three keys, k_1, k_2 and k_3 to compute

$$c = DES_{k_1}(DES_{k_2}^{-1}(DES_{k_3}(w)))$$

3. How to increase security when encripting long plaintexts.

$$w = m_1 m_2 \dots m_n$$

where each m_i has 64-bits.
Choose a 56-bit key k and a 64-bit block c_0 and compute

$$c_i = DES(m_i \oplus c_{i-1})$$

for $i = 1, \dots, m$.

The DES contraversy

1. There have been suspisions that the design of DES might contain hidden "trapdoors" what allows NBS to decrypt messages.

2. The main criticism has been that the size of the keyspace 2^{56} is too small to be really secure.

3. In 1977 Diffie+Hellamn sugested that for 20 milions$ one could build VLSI chip that could search the entire key space within 1 day.

4. In 1993 M. Wiener sugested a machine of the cost 100.000$ that could find the key in 1.5 days.

DES modes of operation

ECB mode: to encode a sequence

$$x_1, x_2, x_3, \ldots$$

of 64-bit plaintext blocks each x_i is encrypted with the same key.

CBC mode: to encode a sequence

$$x_1, x_2, x_3, \ldots$$

of 64-bit plaintext blocks a y_0 is choosen and
each x_i is encrypted by $y_i = e_k(y_{i-1} \oplus x_i)$.

OFB mode: to encode a sequence

$$x_1, x_2, x_3, \ldots$$

of 64-bit plaintext blocks a z_0 is choosen and $z_i = e_k(z_{i-1})$ computed and
each x_i is encrypted by $y_i = x_i \oplus z_i$.

CFB mode: to encode a sequence

$$x_1, x_2, x_3, \ldots$$

of 64-bit plaintext blocks a y_0 is choosen and
each x_i is encrypted by $y_i = x_i \oplus z_i$ where $z_i = e_k(y_{i-1})$.

AES CRYPTOSYSTEM

On October 2, 2000, NIST selected, as the proposed **A**dvanced **E**ncryption **S**tandard, the cryptosystem Rijndael, designed in 1998 by Joan Daemen and Vincent Rijmen.

The main goal has been to develop a new federal cryptographic standard that could be used to encrypt sensitive governmental information securely well into the next century.

AES is expected to be used obligatory by U.S. governmental institution and, naturally, voluntarily, but as a necessity, also by private sector.

AES is to encrypt 128-bit blocks using a key with 128, 192 or 256 bits. In addition, AES is to be used as a standard for authentication, (MAC), hashing and pseudorandom numbers generation.

Motivations and advantages:

- Short code and fast implementations.
- Simplicity and transparency of the design
- Variable key length
- Resistance against all known attacks.

ARITHMETICS in $\mathbf{GF}(2^8)$

The basic data structure of AES is a byte

$$a = (a_7, a_6, a_5, a_4, a_3, a_2, a_1),$$

where a_i's are bits, which can be conviniently represented by the polynomial

$$a(x) = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0.$$

Bytes can be conviniently seen as elements of the field

$$F = GF(2^8)/m(x), \text{ where } m(x) = x^8 + x^4 + x^3 + x + 1.$$

In the field F the addition is the bitwise-XOR and multiplication can be elegantly expressed using polynomial multiplication modulo $m(x)$.

$$c = a \oplus b; \quad c = a \bullet b \text{ where } c(x) = [a(x) \bullet b(x)] \bmod m(x).$$

MULTIPLICATION in GF(2^8)

Multiplication

$$c = a \bullet b \text{ where } c(x) = [a(x) \bullet b(x)] \bmod m(x).$$

in GF(2^8) can be easily peformed using a new operation

$$b = \text{xtime}(a)$$

that corresponds to the polynomial multiplication

$$b(x) = [a(x) \bullet x] \bmod m(x),$$

as follows

set $c = 00000000$ and $p = a$;
for i $=$0 **to** 7 **do**
$c \leftarrow c \oplus (b_i \bullet p)$
$p \leftarrow \text{xtime}(p)$

Hardware implementation of multiplication requires tehrefore one circuit for operation *xtime* and two 8-bit registers.

Operation $b = \text{xtime}(a)$ can be implemented by one step (shift) of the following shift register:

EXAMPLES

- $$`53' + `87' = `D4'$$

  because, in binary,

  $$`01010011' \oplus `10000111' = `11010100'$$

  what means

  $$(x^6 + x^4 + x + 1) + (x^7 + x^2 + x + 1) = x^7 + x^6 + x^4 + x^2$$

- $$`57'' \bullet `83' = `C1'.$$

  Indeed,

  $$(x^6+x^4+x^2+x+1)(x^7+x+1) = x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1$$

  and

  $$(x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1) \bmod (x^8+x^4+x^3x+1) = x^7+x^6+1$$

- $$`57' \cdot `13' = (`57' \cdot `01') \oplus (`57' \cdot `02') \oplus (`57' \cdot `10') = `57' \oplus `AE' \oplus `07' = `FE'$$

  because

  $$`57' \bullet `02' = \mathsf{xtime}(57) = `AE'$$
  $$`57' \bullet `04' = \mathsf{xtime}(AE) = `47'$$
  $$`57' \bullet `08' = \mathsf{xtime}(47) = `8E'$$
  $$`57' \bullet `10' = \mathsf{xtime}(8E) = `07'$$

POLYNOMIALS over GF(2^8)

Algorithms of AES work with 4-byte vectors that can be represented by polynomials of the degree at most 4 with coefficients in GF(2^8).

Additon of such polynomials is done using component-wise and bit-wise XOR. Multiplication is done modulo $M(x) = x^4 + 1$. (It holds $x^j \bmod (x^4 + 1) = x^{j \bmod 4}$.)

Multiplication of vectors

$$(a_3x^3 + a_2x^2 + a_1x + a_0) \otimes (b_3x^3 + b_2x^2 + b_1x + b_0)$$

can be done using matrix multiplication

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ a_1 & a_2 & a_3 & a_0 \\ a_2 & a_3 & a_0 & a_1 \\ a_3 & a_0 & a_1 & a_2 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix},$$

where additions and multiplications ($\bullet$) are done in GF(2^8) as described before.

Multiplication of a polynomial $a(x)$ by x results in a cyclic shift of the coefficients.

BYTE SUBSTITUTION

Byte substitution $b = \mathsf{SubByte}(a)$ is defined by the following matrix operations

$$\begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} (a^{-1})_7 \\ (a^{-1})_6 \\ (a^{-1})_5 \\ (a^{-1})_4 \\ (a^{-1})_3 \\ (a^{-1})_2 \\ (a^{-1})_1 \\ (a^{-1})_0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.$$

This operation is computationally heavy and it is assumed that it will be implemnted by a precomputed substitution table.

ENCRYPTION in AES

Encryption and decryption is done using state matrices

A	E	I	M
B	F	J	N
C	G	K	O
D	H	L	P

elements of which are bytes.

A byte-matrix with 4 rows and $k = 4, 6$ or 8 colums is also used to write down a key with $D_k = 128, 192$ or 256 bites.

ENCRYPTION ALGORITHM

1. KeyExpansion
2. AddRoundKey
3. **do** $(k+5)$-times:
   (a) SubByte
   (b) ShiftRow
   (c) MixColumn
   (d) AddRoundKey
4. Final round
   (a) SubByte
   (b) ShiftRow
   (c) AddRoundkey

The final round does not contain MixColumns procedure. The reason being is to be able to use the same hardware for encryption and decryption.

KEY EXPANSION

The basic key is written into the state matrix with 4, 6 or 8 columns. The goal of the key expension procedure is to extend the number of keys in such a way that each time a key is used a new key is used.

The key extension algorithm generates new columns W_i of the state matrix from the columns W_{i-1} and W_{i-k} using the following rule

$$W_i = W_{i-k} \oplus V,$$

where

$$V = \begin{cases} F(W_{i-1}), \text{if } i \bmod k = 0 \\ G(W_{i-1}), \ \text{if } i \bmod k = 4 \text{ and } D_k = 256 \text{ bit} \\ W_{i-1}, \ \text{otherwise} \end{cases},$$

where the function G performs only the byte-substitution of the corresponding bytes. F function is defined in a little more complicated way.

STEPS of ENCRYPTION

AddRoundKey procedure adds byte-wise and bit-wise current key to the current contents of the state matrix.

ShiftRow procedure cyclicaly shifts i-th row of the state matrix by i shifts.

MixColumns procedure multiplies columns of the state matrix by the matrix

$$\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

DECRYPTION in AES

Steps of the encryption algorithm map an input state matrix into an output matrix.

All encryption operations have inverse operations. Decryption algorithm applies in the oposite order as at the encryption the inverse versions of the encryption operations.

DECRYPTION

1. Key Expansion
2. AddRoundKey
3. **do** $k + 5$-times:
   (a) InvByteSub
   (b) InvShiftRow
   (c) InvMixColumn
   (d) AddInvRoundKey
4. Final round
   (a) InvByteSub
   (b) InvShiftRow
   (c) AddRoundKey

SECURITY GOALS

The goal of the authors was that Rijndael (AES) is **K-secure** and **hermetic** in the following sense:

Definition A cryptosystem is K-secure if all possible attack strategies for it have the same expected work factor and storage requirements as for the majority of possible cryptosystems with the same dimension.

Definition A block cryptosystem is **hermetic** if it does not have weaknesses that are not present for the majority of cryptosystems with the same block and key length.

MISCELANEOUS

- Pronounciation of the name *Rijndael* is as "Reign Dahl" or "rain Doll" or "Rhine Dahl".
- AES proposal of Rinndael can be found on http://csrc.nist.gov/encryption/aes/

Key management

Secure methods of key management are extremely important. In practice, most of the attacks on public-key cryptosystems are likely to be at the key management levels.

Problems: How to obtain securely an appropriate key pair? How to get other people public keys? How to get confidence in the legitimacy of other's public keys? How to store keys? How to set, extend, ... expiration dates of the keys?

Who needs a key? Anyone wishing to sign a message, to verify signatures, to encrypt messages and to decrypt messages.

How does one get a key pair? Each user should generate his/her own key pair. Once generated, a user must register his/her public-key with some central administration, called a **certifying authority**. This authority returns a certificate.

Certificates are digital documents attesting to the binding of a public-key to an individual or institutions. They allow verification of the claim that a given public-key does belong to a given individual. Certificates help prevent someone from using a phony key to impersonate someone else. In their simplest form, certificates contain a public-key and a name. In addition they contain: expiration date, name of the certificate issuing authority, serial number of the certificate and the digital signature of the certificate issuer.

How are certificates used

The most secure use of authentification involves enclosing one or more certificates with every signed message. The receiver of the message verifies the certificate using the certifying authority's public-key and, being confident of the public-key of the sender, verify the message's signature. There may be more certificates enclosed with a message, forming a hierarchical chain, wherein one certificate testifies to the authentificity of the previous certificate. At the top end of a certificate hierarchy is a top-level certifying-authority to be trusted without a certificate.

Example According to the standards, every signature points to a certificate that validates the public-key of the signer. Specifically, each signature contains the name of the issuer of the certificate and the serial number of the certificate.

How do certifying authorities store their private keys?

It is extremly important that private-keys of certifying authorities are stored securely. One method to store the key in a tamperproof box called a **Certificate Signing Unit**, CSU.

The CSU should, preferably, destroy its contents if ever opened. Not even employees of the certifying authority should have access to the private-key itself, but only the ability to use private-key in the certificates issuing process.

CSU are for sells.

What is PKI?

- PKI (Public key infrasture) is an infrastructure that allows to handle public-key problems for the community that uses public-key cryptography.
- Structure of PKI

Security policy that specifies rules under which PKI can be handled.

Products that generate, store, distribute and manipulat keys.

Procedures that define methods how

- to generate and manipulate keys
- to generate and manipulate certificates
- to distribute keys and certificates
- to use certificates.

Authorities that take care that the general security policy is fully performed.

PKI users and systems

- Certificate holder
- Certifcate user
- Certification authority (CA)
- Registration authority (RA)
- Revocation authority
- Repository (to publish a list of certicates, of revocated certificates,...)
- Policy management authority (to create certification policy)
- Policy approving authority

SECURITY of CA and RA

PKI system is so secure how secure are systems for certificate authorities and registration authorities.

The basic principles to follow to ensure necessary security of CA and RA.

- Privite key of CA has to be stored in a modul that is secure against intentional professional attacks.
- Steps have to be made for renovation of the private key in the case of a collapse of the system.
- Access to CA/RA tools has to be maximally controlled.
- Each requirement for certifiction has to be authorized by several independent operators.
- All key transactions of CA/RA have to be loged to be available for a possible audit.
- All CA/RA systems and their documentation have to satisfy maximal requirements for their reliability.

PUBLIC-KEY INFRASTRUCTURE PROBLEMS

Public-key cryptography has low infrastructural overhead, it is more secure, less truthful and with better geographical reach. However, this is due to the fact that public-key users bear a substantial administrative burden and security advantages of the public key cryptography rely excessively on the end-users' security discipline.

Problem 1: With public-key cryptography users must constantly be careful to validate rigorously every public-key they use and must take care for secrecy of their private secret keys.

Problem 2: End-users are often unwilling or unable to manage keys diligently.

User's behaviour is the weak link in any security system, but public-key security is unable to reinforce this weakness.

Problem 3: Only sophisticated users, like system administrators, can realistically be expected to meet fully the demands of public-key cryptography.

Main components of public-key infrastructure

• The Certification Authority (CA) signs user's public-keys.
(There has to be a hierarchy of CA, with a root CA on the top.)

• The Directory is a public-access database of valid certificates.

• The Certificate Revocation List (CRL) – a public-access database of invalid certificates.
(There has to be a hierarchy of CRL).

Stages at which key management issues arise

• Key creation: user creates a new key pair, proves his identify to CA. CA signs a certificate. User chooses his passpharse to encrypt his private key.

• Single sign-on: decryption of the private key, participation in public-key protocols.

• Authenticating others: to get others keys and certificates, to consult CRL for notice of certificate's revocation, validation of CA signatures.

• Key revocation: CRL should be checked every time a certificate is used. If a user's passpharse or his secret key is comprommised, CRL administration has to be notified.

MAIN PROBLEMS

• **Authenticating the users:** How does a CA authenticate a distant user, when issuing the initial certificate?

(Ideally CA and the user should meet. Consequently, properly authenticated certificates will have to be expensive, due to the laber cost in a face-to-face identity check.)

• **Authenticating the CA:** Public key cryptography cannot secure the distribution and the validation of the Root CA's public key.

• **Certificate revocation lists:** Timely and secure revocation presents big scaling and performance problems. As a result public-key deployment is usually proceeding without a revocation infrastructure. (Revocation is the classical Achilles' Heel of public-key cryptography.)

• **Private key management:** The user must keep his long-lived secret key in memory during his login-session: There is no way to force a public-key user to choose a good password.

(Lacking effective password-quality controls, most public-key systems are volnurable to the off-line guessing attacks.)

LIFE CYCLE of CERTIFICATES

Issuing of certificates

- registration of applicants for certificates;
- generation of pairs of keys;
- creation of certificates;
- delivering of certificates;
- dissemination of certificates;
- backuping of keys;

Using of certificates

- receiving a certificate;
- validation of the certificate;
- key backup and recovery;
- automatic key/certificate updating

Revocation of certificates

- expiration of certificates validity period;
- revocation of certificates;
- archivation of keys and certificates.

Pretty Good Privacy

In June 1991 Phil Zimmermann, made publically available software that made use of RSA cryptosystem very friendly and easy and by that he made strong cryptography widely available.

Starting February 1993 Zimmermann was for three years a subject of FBI and Grand Jurry investigations, being accused of illegal exporting arms (strong cryptography tools).

William Cowell, Deputy Director of NSA said: "If all personal computers in the world – approximately 200 milions – were to be put to work on a single PGP encrypted message, it would take an average an estimated 12 milion times the age of universe to break a single message".

Heated discussion whether strong cryptography should be allowed keep going on. September 11 attack brought another dimension into the problem.

Patentability of cryptography

- Cryptographic systems are patentable

- Many secret-key cryptosystems have been patented

- The basic idea of public-key cryptography are contained in U.S. Patents 4 200 770 (M. Hellman, W. Diffie, R. Merkle) – 29. 4. 1980 U.S. Patent 4 218 582 (M. Hellman, R. Merkle)

The exclusive licensing rights to both patents are held by “Public Key Partners” (PKP) which also holds rights to the RSA patent.

All legal challenges to public-key patents have been so far settled before judgment.

Some patent applications for cryptosystems have been blocked by intervention of us: intelligence or defence agencies.

All cryptographic products in USA needed export licences from the State department, acting under authority of the International Traffic in Arms Regulation, which defines cryptographic devices, including software, as munition.

Export of cryptography for authentication has not been restricted, problems were only which cryptography for privacy.