Computer Science Journal of Moldova, vol.**2**, no.3(6), 1994

Effective software–oriented cryptosystem in complex PC security software

A.Moldovyan N.Moldovyan P.Moldovyan

Abstract

To ensure high encryption rate and good data security, an organization of an encipherement program in the form of two modules was proposed. The first module is used for customizing the second one, the latter being the resident of the program, wich maintains all application calls about encryption procedures. This approach is shown to be perspective for the elaboration of the cryptosystems with indefinite cryptalgorithm. Several typical software–oriented cryptoschemes are considered. The developed cryptomodules have high encipherement rate (2–10 Mbps for Intel 386) and secure high information protection level. Organization of a new computer security software complex COBRA is considered. High enciphering rate and good data protection are provided by the resident cryptomodule using less than 1 kbyte of the main memory and working in dynamic encryption mode.

1 Introduction

Modern information technologies widely used in State, military, commercial, financial and industrial activities have raised the problem of information protection from unauthorized access and modification. One of the important problems of the PC protection is to maintain the integrity of technological programs and data. Another one is the information secrecy. Different types of hardware or a combination of hardware and software are used in information protection systems [1, 2]. One of the safe and universal methods of data protection is the cryptographical conversion.

The majority of the known cryptalgorithms providing high data protection level, such as the USA standard DES [3], the USSR standard GOST 28147–89 and Japanese cipher FEAL–4 [4]. are hardware–oriented. Soft realization of these cryptoschemes does not allow one to create cryptomodules having high enciphering rate and minimal size. Creation of effective software–oriented cryptalgorithms promotes solving numerous practical problems without utilizing any additional hardware resouces, the following advantages being obtained:

- shortening of the new system creation terms;
- reduction of the financial expenditures;
- possibility of rapid customizing for concrete problem.

Working on the complex software for the PC security maintenance, we have elaborated a software–oriented cryptoschemes on the basis of which a new promissing computer protection system COBRA has been created.

This paper deals with the construction ideology and concrete practical schemes of software–oriented cryptosystems and describes briefly the organization of the complex computer security software COBRA.

2 Soft cryptomodule organization

To create effective software–oriented cryptosystems a division of the encipherement program into two modules was proposed [5, 6]. The first module functioning consists in customizing the second one, the latter executing encryption–decryption procedures. The essence of such organization is obvious in cases when encipherement program is started once to maintain a great many of cryptoconversion procedures. Thus, all critical characteristics are determined by the second module which is the resident of the cryptosystem. On the other hand, tuning subprogram does not impose limitations on used main memory capacity and the run time. Moreover, the customizing stage is an effective barrier counteracting attacks by means of the password search because to check a pass key one must execute the tuning subprogram.

The classical single key cryptosystem organization is based on fixed cryptoconversion operations, parameters of which are controlled by the key. Two–stage mechanism of the soft cryptomodule construction can be used for elaborating cryptosystems which significantly differ from the classical ones. Namely, during the tuning phase, the password–controlled generation of the cryptalgorithm of the resident module can be foreseen. The customization subroutine job can include the following general tasks, each of which is controlled by the password:

- formation of the cryptographical keys and of both substitution and permutation tables;
- formation of the unique set of key immeidiate constants used by the second subroutine;
- selection of the unique set of operations;
- generation of the retrieval queue of the conversion of the input data block symbols.

The systems with indefinite conversion procedures can secure both the unique key and the unique cryptalgorithm for every user. If the number of realizable cryptographical mechanisms is about or over 10^5 , the conversion procedure indefiniteness will effectively counteract cryptanalytical attacks. Though the theoretical question about the evidence for the new cryptosystems resistibility is very complicated, the practical reasons in favour of their application are

- experimental testing;
- possibility of combining classical and new cryptalgorithms in the multipass schemes;
- possibility of checking the quality of the formed algorithm during the tuning phase.

The first important task of the tuning phase is to convert the password of arbitrary length into a pseudorandom key sequence (key area)

of relatively large size. Algorithm of such conversion must be composed as an one–way function mechanism to prevent any attempt to attack the system on the basis of the detection of the correlation between password and generated key sequence. Key area generation can be fulfilled in accordance with the Algorithm 1 which uses two source random sequences $\{a_j\}$ $(j = 0, 1, 2, \ldots, 1023)$ and $\{b_i\}$ $(i = 0, 1, 2, \ldots, 511)$.

Algorithm 1.

Ganeration of pseudorandom key sequence

INPUT: a password $\{p_l\}$, $l = 0, 1, 2, \ldots, L$ $(L < 128)$.

1. The pasword is repeated several times to obtain the 1024–byte sequence $\{p_j\}$ in which every member p_j is the binary representation of the corresponding character of the password.
2. Two 512–byte sequences $\{t_i\}$ and $\{h_i\}$ are formed, where $i = 0, 1, 2, \ldots, 511$; $t_i = a_i + p_i \, mod \, 256$; and $h_i = a_{i+512} + p_{i+512} \, mod \, 256$.
3. Using the cryptographical mechanism of the second phase (Algorithm 2) and the $\{h_i\}$ sequence as the key area, one executes the encryption of the $\{t_i\}$ sequence, the 512–byte ciphertext $\{c_i\}$ being obtained.
4. Using the module 2 addition operation $*$, the 512–byte sequence $\{d_i = b_i * c_i\}$ is formed.
5. Using Algorithm 3 and sequence $\{d_i\}$ as key area convert sequence $\{h_i\}$ in the terminal sequence $\{k_i\}$.

OUTPUT: 512–byte pseudorandom key sequence $\{k_i\}$ wich is used by the resident cryptomodule.

3 Resident module cryptoscheme

Functioning of the resident cryptomodule mechanism is explained in Fig. 1, where pointers Y and U represent the numbers of bytes in the

upper and lower keys. Cryptographical conversion procedures are executed in accordance with the Algorithms 2 and 3 which use additional 8–bit key parameters K_1, K_2, K_3, K_4, K_5 and K_6.

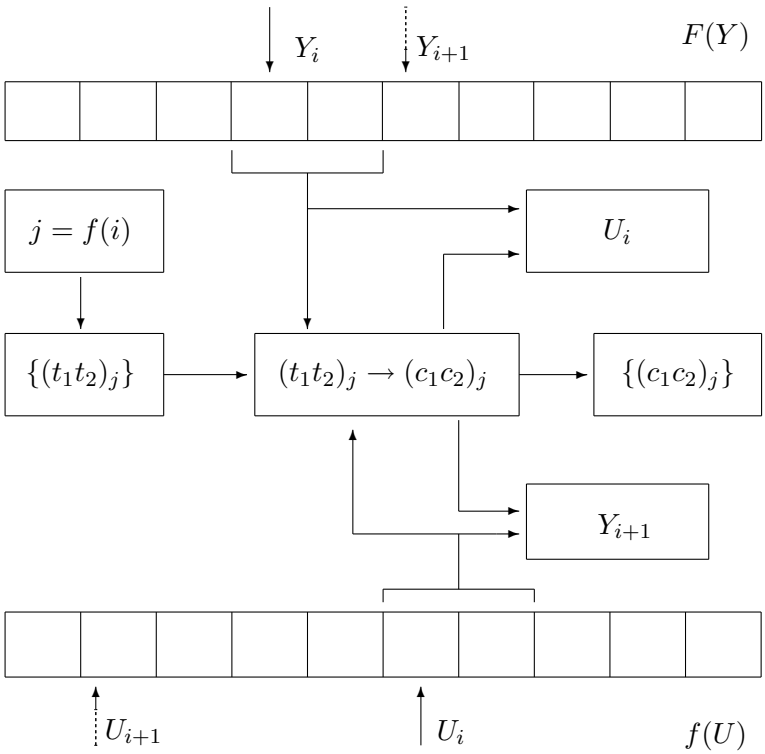

Figure 1: Resident module encryption scheme

Algorithm 2.

Encryption/decryption of the 512–byte data block

INPUT: source 512–byte data block in wich two–byte words $(t_1t_2)_j$ are numbered by index $j = 0, 1, 2, \ldots, 255$.

1. Set counter $i = 0$ and initial values $Y_0 = K_1$, $U_0 = K_2$, and define conversion mode $E = 1$ (encryption) or $E = 0$ (decryption).
2. Calculate index $j = (K_3 * i) + K_4 \, mod \, 256$.
3. If $E = 1$ jump to step 4, otherwise jump to step 7.
4. Execute conversion

$$\begin{aligned}(c_1c_2)_j \;\; = \;\; & \{(t_1t_2)_j * [256F(Y_i + 1 \, mod \, 256) + F(Y_i)]\} \\ & + [256f(U_i + 1 \, mod \, 256) + f(U_i)] \, mod \, 256^2,\end{aligned}$$

   where $(c_1c_2)_j$ and $(t_1t_2)_j$ are the current pairs of characters of the output and input data blocks.
5. Set the next pointer positions

$$\begin{aligned}& Y_{i+1} = Y_i + (c_1)_j * f(U_i) \, mod \, 256 \\ & \text{and} \\ & U_{i+1} = U_i * (c_2)_J + F(Y_i) \, mod \, 256.\end{aligned}$$

6. Jump to step 9.
7. Execute conversion

$$\begin{aligned}(c_1c_2)_j = \;\; & (t_1t_2)_j - [256f(U_i + 1 \, mod \, 256) \\ & + f(U_i)] * [256F(Y_i + 1 \, mod \, 256) \\ & + F(Y_i)] \, mod \, 256^2.\end{aligned}$$

8. Set the next pointer positions

$$\begin{aligned}& Y_{i+1} = Y_i + (t_1)_j * f(U_i) \, mod \, 256 \\ & \text{and} \\ & U_{i+1} = U_i * (t_2)_J + F(Y_i) \, mod \, 256.\end{aligned}$$

9. If $i < 255$ then increment i and jump to step 2, otherwise STOP.

OUTPUT: 512–byte data block $\{c_i\}, i = 0, 1, 2, \ldots, 511$.

Algorithm 3.

Modified en(de)cryption of the 512–byte data block

INPUT: source 512–byte data block in wich two–byte words $(t_1t_2)_j$ are numbered by index $j = 0, 1, 2, \ldots, 255$.

1. Set counter $i = 0$, index $j = -1$, initial values $Y_{-1} = K_1,\ U_{-1} = K_2,\ (c_1)_{-1} = (t_1)_{-1} = K_5,\ (c_2)_1 = (t_2)_{-1} = K_6$ and define conversion mode $E = 1$ (encryption) or $E = 0$ (decryption).
2. If $E = 1$ jump to step 3, otherwise jump to step 7.
3. Calculate values

$$\begin{aligned} Y_i' = &\ Y_{i-1} * [256f(U_{i-1} + 1\, mod\, 256) \\ &\ f(U_{i-1})] + (c_1)_j\ mod\ 256^2 \end{aligned}$$

and

$$\begin{aligned} U_i' = &\ U_{i-1} + [256F(Y_{i-1} + 1\, mod\, 256) \\ &\ F(Y_{i-1})] * (c_2)_j\, mod\, 256^2. \end{aligned}$$

4. Calculate index $j = 255 - [(K_3 * i) + K_4\, mod\, 256]$.
5. Execute conversion

$$(c_1c_1)_j = [(t_1t_2)_j + Y_i'] * U_i'\, mod\, 256^2$$

6. Jump to step 12.
7. Calculate values

$$\begin{aligned} Y_i' = &\ Y_{i-1} * [256f(U_{i-1} + 1\, mod\, 256) \\ &\ +f(U_{i-1})] + (t_1)_j\, mod\, 256^2 \end{aligned}$$

and

$$\begin{aligned} U_i' = &\ U_{i-1} + [256F(Y_{i-1} + 1\, mod\, 256) \\ &\ +F(Y_{i-1})] * (t_2)_j\, mod\, 256^2 \end{aligned}$$

8. Calculate index $j = 255 - [(K_3 * i) + K_4\, mod\, 256]$.
9. Execute conversion

$$(c_1c_1)_j = [(t_1t_2)_j * U_i'] - Y_i'\, mod\, 256^2.$$

10. If $i < 255$ then set the next pointer positions $Y_i = Y_i' mod\, 256$, $U_i = U_i' mod\, 256$, increment i, and jump to step 2, otherwise STOP.

OUTPUT: 512–byte data block $\{(c_1c_1)_j\},\ j = 0, 1, 2, \ldots, 255$.

On the basis of the Algorithms 2 and 3 the high–speed (5–10 Mbps for microprocessor Intel 80386/40) resident cryptomodule for the system COBRA have been composed, its size being about 1 kbyte. The cryptomodules having the encryption rate 5 Mbit/s or more can be used to maintain two– or three–pass conversion mode. In the COBRA system it is foreseen that the number of rounds depends on the password size. The pass key having the size of 10 or more bytes defines the two–pass mode (Algorithm 2 + Algorithm 3). By choosing the 9 or less byte password user can define the one–pass mode (Algorithm 3). Possibly it is reasonable to foresee the choice of the three– and four–pass encryption modes in the next version of the COBRA system. Experimental cryptomodules combining cryptoscheme described above with the substitution and permutation methods show good characteristics as regards conversion speed, size, and cryptoresistibility.

All cryptomodules elaborated transform different type cleartexts into cryptogramms, the latter being the pseudorandom sequences of bytes having values from 0 to 255. The frequency of all symbols of the ciphertext is about the same. Numerous cryptogramms were checked by several spectral and compression tests which proved pseudorandomness of the symbol distribution.

The evaluation of the system cryptoresistibility have been made fore the Algorithm 2 fulfilling one–pass encryption. As regards cryptoresistibility it does not seem to be better than Algorithm 3. It is supposed that cryptanalist knows the algorithm, the cryptogramm and the V size part of the cleartext. When the V value is large enough, there is the theoretical possibility of revealing and calculating the $[F(Y), f(U)]$ key element pairs used more than once while encypting the V size text. It means that some critical value $V = V_c$ exists in principle.

Quantitative evaluation of the data protection safety can be carried out assuming the pseudorandom character of the current Y and U

positions. Two key elements for one elementary conversion $(t_1t_1)_j \rightarrow (c_1c_1)_j$ are used. Guessing the value of one of them one can calculate the second key element. Whether the current trial variant is erroneous can become clear when the Y and U positions are repeated simultaneously. For this model we have obtained the following formula for the minimal search resistibility

$$W = 256^{48-N},$$

where W is the minimal number of trial variants; N is the quantity of the key element pairs which can be theoretically calculated from the known plaintext of the $V < V_c$ size. It was experimentally found that V_c equals about 4 kbyte. This formula gives the minimal number of variants which are to be tried after the cryptanalist having succeeded in dertermining N pairs of the key elements.

One can expect the W and V_c values for the two– and three–pass encipherement schemes to be much greater, as compared with the considered scheme. It is easy to show that one–pass encryption scheme (Algorithm 2 or Algorithm 3) is sensitive for chosen plaintext attack, but two– (e.g. Algorithm 2 + Algorithm 3) and multipass schemes resist it well. Cryptoresistibility calculations for two– and multiround systems represent a difficult problem requiring a special investigation.

4 Multipass cryptoscheme with indefinite algorithm

Using high–speed cryptalgorithms one can execute the multipass encryption of the plaintext. The indefiniteness of the conversion algorithm is set by the password–controlled selection of the concrete encryption methods from a given batch (library) of n algorithms. Chosen algorithms are queued according to some key parameters generated during tuning phase (Fig. 2).Thus in m–pass system the tuning module fulfils pseudorandom selection of m algorithms from the library and activates them. The number of realizable algorithms is $S = n^m$. For $m = 4$ and $n = 20$ the S value is more than 10^5. A set of suitable soft–oriented mechanisms has been described previously [5, 7].

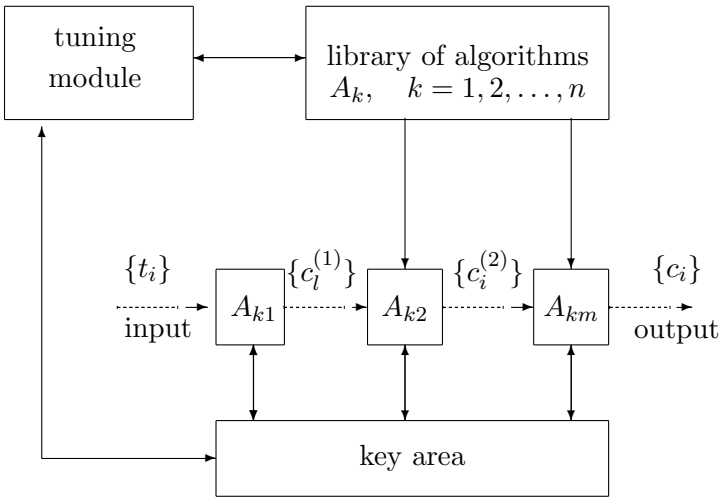

Figure 2: Resident module mechanism with indefinite cryptalgorithm

Independently of the selection of algorithms, one can also use the password–controlled pseudorandom retrieval of symbols from both plaintext $\{t_i\}$ and intermediate cryptogramms $\{c_i^{(1)}\}, \{c_i^{(2)}\}, \ldots, \{c_i^{(m-1)}\}$, $i = 0, 1, 2, \ldots, L$, where L is the length of input data blocks. In this case the number of potentially realizable encipherement mechanisms equals $S = n^m A_w^m$, where w is the number of different retrieval sequences provided by a special mechanism. For system processing 512–byte blocks the following formula for calculating number of the current pair of bytes j defines very large w value:

$$j = [(K_1 * i) + K_2] * K_3,$$

where K_1, K_2, K_3 are 8–bit key parameters.

Such multipass cryptosystems can secure very high resistibility, but their resident program is enlarged and data conversion speed decreases as compared with the two–pass cryptoschemes. A five–pass experimental cryptomodule having encryption speed about 2 Mbps and size of the resident subprogramm of about 3 kbyte has been composed. This cryptosystem is planned to be used in special modifications of the security complex COBRA.

5 Software complex COBRA

The computer security system COBRA has been used for about several years in different institutions of Moldova and Russia. It has shown itself as an effective, reliable and comfortable computer security tool. COBRA functions in the MS DOS, PC DOS or DR DOS environments together with WINDOWS, SuperStor, dBase, FoxPro, Clipper and other software. This PC security system **imposes no restriction on utilization of any software.** When the most powerful protection mode of this system is installed the lost of data processing rapidity does not exceed 3%.

The system COBRA includes several subsystems the main ones being the access control subsystem and the soft environment integrity maintenance subsystem. The latter is oriented to

- modification detection of the soft environment, caused by an intruder, computer viruses or inconsiderate action of the user,
- typing–out the information about modified component,
- automatic re–establishing of the modified component.

The access control subsystem performs the following functions:

- user identification,
- formation of the user authentification list,

- protection of the computer from the bootstrap loading from a floppy disk, and
- cryptographical conversion in the dynamic mode.

Resident cryptomodule executes automatically

- data encryption when they are transfered from the main memory to the magnetic one, and
- data decryption when they are transfered from the magnetic memory to the main storage.

The cryptomodule is started in the dynamic mode, i.e. in accordance with each application call about reading or writing a definite portion of information. This cryptoconversion mode provides writing confidential information down on a magnetic disk only in the form of a ciphertext. Such organization of the protection system **does not change user's routine work technology**. High cryptographical conversion rate makes the system **transparent (imperceptible) for the user**. Minimal size of the system rezident (less than 3 kbyte) makes COBRA also transparent for the application programs. The presence of the complex **COBRA does not change the data processing rapidity** and is perceptible only when the computer is switched on and the system requests the pass key.

Setting a special mode one can completely encrypt the hard disk, including master boot record. In this case no information on hard disk can be read without a special key diskette.

Software COBRA is complemented with the file encryption subsystem SAFE which is intended for automatization of the confidential information processing. This program is a powerful and comfortable tool for reliable and quick encryption and decryption of information. Any group of files, a directory with subdirectories, a group of directories, a logical partition of the hard disk, or a diskette, any of these can be processed as a single input block. The program SAFE includes the internal command interpreter whose language allows one to automatize the file encryption/decryption activities fulfilled frequently. The subsystem has an easy, clear and comfortable interface.

Program US is the special user's shell and gives the possibility to control access to the function buttons of this subsystem (F1 ... F10), to files, to directories, and to command line.

Subsystem LOCK allows one to point out the user's activities for journaling and to install the real–time encryption mode when working with assigned files or directories.

6 Conclusion

The two–stage principle of the cryptographycal conversion organization makes it possible to create high–speed soft cryptomodules of minimal size, providing safe information protection, as well as cryptosystems with indefinite algorithm. An encryption program based on this approach has been composed and used in the complex computer security software COBRA. The efficiency of the resident cryptomodule allows one to realize the dynamic encipherement mode. Elaboration of cryptoschemes with indefinite algorithm seems to be a promissing trend aimed at creating effective soft cryptomodules securing enhanced resistibility.

Being extremely comfortable for the user and providing high level of information protection, the computer security complex COBRA is perspective for mass utilization on personal, lap–top and note–book computers.

References

[1] Johnson D.B., Dolan G.M., Kelly M.J., Le A.V., Matyas S.M. Common Cryptographic Architecture Cryptographic Application Programming Interfase/IBM Systems Journal. 1991, Vol.30, No.2, pp.130–150.

[2] Yeh P.C., Smith, Sr. R.M. ESA/390 Integrated Cryptographic Facility: An overview/IBM Systems Journal. 1991, Vol.30, No.2, pp.192–205.

[3] Garon G., Outerbridge R. DES watch: an examination of the sufficiency of the Data Encryption Standard for financial institution in the 1990's/Cryptologia. 1991, Vol.15. No.3, pp.177–193.

[4] Murphy Sean. The cryptanalysis of FEAL-4 with 20 chosen plaintexts/J. Cryptol. 1990, Vol.2, No.3, pp.145–154.

[5] Moldovyan A.A., Moldovyan N.A., Moldovyan P.A. A new method of cryptographical transformations for modern computer security systems/Upravlyayuschie sistemy i mashiny. 1992, No.9/10, pp.44–50 (in Russian).

[6] Moldovyan A.A., Moldovyan N.A. A new principle of the cryptographical module organization in a computer security system/ Kibernetica i sistemny analiz. 1993, No.5, pp.42–50 (in Russian).

[7] Moldovyan A.A., Moldovyan N.A. A method for constructing efficient programmed small-capacity cryptomodule/Upravlyayuschie sistemy i mashiny. 1993, No.3, pp.84–88 (in Russian).

Received 30 November, 1994

A.Moldovyan
Institute of Modelling and Intellectualization
of Complex Systems, Prof. Popov Street, 5,
St-Petersburgh 197376, Russia
ph. 7-812-2349094 fax.7-812-2349093

N.Moldovyan
Institute of Applied Physics
Kishinev, 277028, Moldova,
Academiei str., 5

P.Moldovyan
Computer department,
of the bank "Intreprinzbanca",
Banulescu-Bodoni str., 45
Kishinev, 277012, Moldova
e-mail: *peter@ibank.moldova.su*
ph./fax 373-2-225170