FMU-C RAPTOR PROTOCOL

FMU-C RAPTOR TEAM

MSDT(f10) - UNSTOLENABLE PASSWORD - FLOATING PASSWORD,
Unstolenable password cryptographic software / New Encryption System

---

MSDT(f10) / Maximum Safety Data Transfer©fmu-c MSDT v7.0 / v7.3 (DCM)
Introduction of the encryption software and instructions

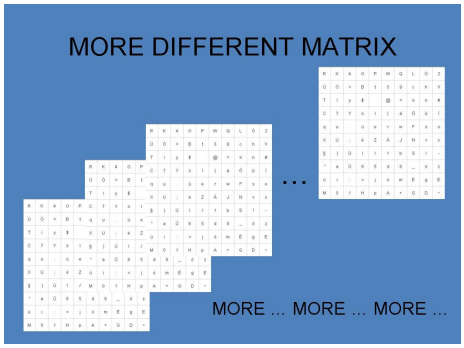
Is your password safe?

The currently used passwords are always registered and are all stored in a server when logging into email, or a website in order to compare the user's name and password to the registered ones. These so called registered passwords can be obtained in the course of the logging in procedure or from the servers directly through net penetration, spy programs, key-logger…etc. (Please do not think only about criminals; state authorities have legal access to obtain your password as well.) **Now, is your password safe? Is your password your exclusive property? These are the real questions nowadays.**

The revolutionary break-through in the cryptographic industry, the MSDT(f10) encryption program uses passwords that always remain in the user's possession. This is the so called FLOATING PASSWORD GENERATED BY THE MSDT (f10) ENCRYPTION PROGRAM IN AN ULTIMATE SECURE WAY! The user is the only responsible person for the security of the password. The FLOATING PASSWORD has not been registered/stored in any program, coded material or server. The floating password enables to exchange the encryption keys via the internet by using the old key and password for coding a new symmetric one.

Create an encryption key:

Currently, the program uses 256 different characters. The first matrix is being created by mixing the characters randomly in a 16x16 matrix from which we create the second matrix by mixing the characters again and so on up to even 10 000 matrixes by always mixing the next matrix.

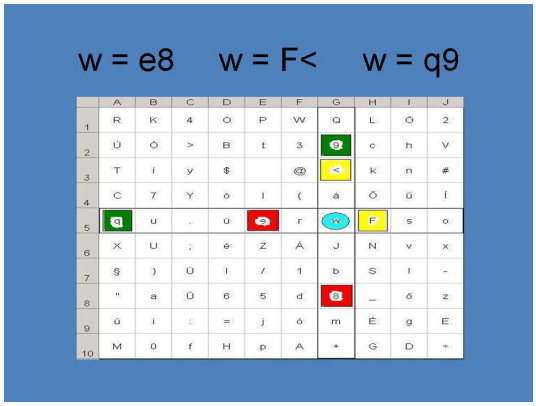


The set of matrixes will be the encryption key for the coding procedure (KEY MDT). The UNICODE uploading of the charset, the unique random number generation (RND), mixing of the characters, placing the values between ASCII 0-255 and the 256 factorials, please see "OPERATION in DETAILS" below.

The coding procedure:

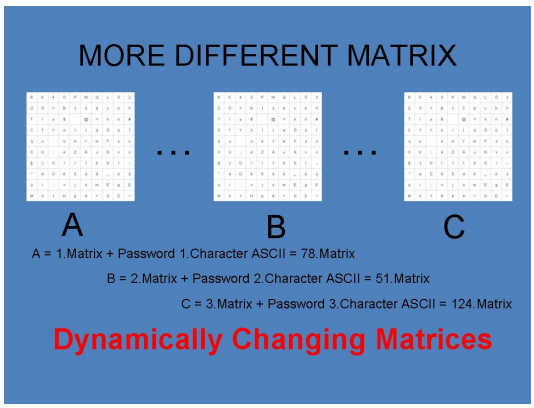
The user should decide the number of matrixes when creating the key. Currently, it could be made up of 100 to 10 000 matrixes (which can be amplified boundlessly in the source-code). The first character of the original (clear) text will be coded by the ALGORITHM (demonstrated on our website under

"Downloading" section) in a way when the designated place of the clear (original) character in the matrix is that intersection which selects a row and a column. The character in the intersection will be replaced by the randomly selected row and then the column characters of the matrix.



The second original character will be coded in the second matrix in the same way and so forth…the next character in the next matrix. So, the coding procedure needs all the matrixes available in the key. When reaching the last matrix, the process starts the process in the first matrix again.

The characters of the FLOATING PASSWORD entered the REF-field in the **DCM (v7.3)** version the numbers of the ASCII codes will be added to the serial number of the actual matrix (the next matrix of the original coding is shifted one by one). But in case of using the **password the matrixes are shifted with numbers of the ASCII codes of the password's consecutive characters and not one by one**, so the coding procedure can take place in dynamically changing matrixes (DCM) depending on the password (which can consist of 200 characters presently but can be enlarged boundlessly in the source code).



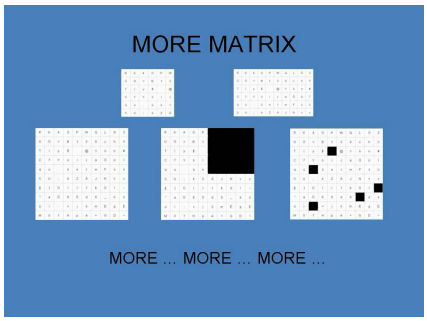
The decoding procedure:

The original character will be restored from the intersection obtained from the row and column identifiers. The program always restores the next original character from the next matrix similar to the coding process and in case of using a password it operates according to dynamically changing matrixes (DCM). **More options:** The size and shape of the matrixes can be flexible as well as it can contain so called "black holes". We could add password to the fields excluded from the coding procedure.

Operation in details:

The program uses 256 default characters of the UNICODE charset for creating key, coding and decoding procedures. It is possible to **change the charset** in the source-code by uploading more characters in order to use bigger matrixes. Over to this option, the matrixes can be of different shapes (L-, T-shaped for example). If we create square- or rectangular-shaped matrixes, we can generate so called "black holes" within the matrixes (similar to a crossword-puzzle). The program discards the

empty parts in the T-, L-shaped, or “black-holed” matrixes, starts recounting with the RND as long as it finds an active field. Of course, this makes the program slower but it offers boundless freedom.



The random-number generation (RND): There is in the source-code selected ten places of the tenths, century, millennium…etc. seconds after the decimal point of the system time’s second. For example: There are running numbers from 8th to 17th places. The one-digit RND-number of the system determines as to which number should select the first row when the system runs. For instance: The RND of the system gives number 2 when it starts operating. Now the row will be selected by the number running at the 9th place from the 8th-17th places after the decimal point. The second generation gives another number (if it is the same as the first one was the system discards it); this number selects another row and these two rows, the characters in them will be interchanged. Mixing the characters entirely takes place in a way when the system interchanges 1-1 row then 1-1 column will be designated and interchanged. Then come the next 1-1 row and 1-1 column again and so forth. You could determine in the source-code as to how many times the rows and columns could be interchanged. New mixing needs new determined number of the system time so the characters in the matrix will get an accidental place.

The program uses only numbers for the coding procedure according to the placing of the value between **ASCII 0-255** and uses hexadecimal values as well. Because the program operates with 16x16 matrix in the default mode and to reach the exchange of all rows and columns the designated rows and columns will be counted alternately, first from the bottom up to the top then from the left to the right.

The combination of 256 different characters is the **factorial value**. This offers a combination option composed of 507 numbers for mixing a single matrix. This combination option regards only the first character of the coded feature. The next character has the same option in a different matrix. Compared to the GOOGOL number (http://en.wikipedia.org/wiki/Googol) which is equal 70 factorials so it is obvious as to how long would it take to mix a single matrix from these combinations. If someone tries to break up the coded material (without a key) needs to use all the combination options (take into account that every of them must be different) moreover to filter every option to get a clear text not to forget that converting a file does not result in any clear text. So, there is no way of breakup in case of text coding because the interim stage in the **MXM2 version** outlined in the Instructions part does not enable to display a clear text in any way (frequency analysis is excluded!).

There is an additional **DCM (v7.3) version, which is able to change the matrixes dynamically according to ASCII- codes of the FLOATING PASSWORD.** And now it is the icing on the cake: the role of the **control- algorithm**: Every clear text coding procedure results in different coded material should it be text, file (exe, avi, jpg, mp3, rar…etc) i.e. no matter how many times you repeat the coding procedure you will get **different encrypted material (text, file or whatever you want)** at the end of the day. **There is no one-to-one correspondence between sets in the MSDT program. This none-one-to-one correspondence between sets in the program makes the MSDT(f10) program unbreakable.**

MSDT(f10)©fmu-c MSDT v7.0 és v7.3 (DCM) encryption program instructions:

Basic rules:

The MSDT v7.0 program needs no installation; it is a portable encryption version that can operate via memory card, pendrive (USB stick) or memory card in a secure way.

If you work on a computer connected to the internet your keys and passwords are in danger; they can be stolen at any time. Every coding procedure needs an **independent computer** which is not and cannot be linked to the net in any way. Just the coded feature is allowed to be furnished via internet through another computer and different pendrive. Do not keep the program and the key-pendrive on your independent computer after you fished the coding procedure! They must be kept in a secure place! The program can be used at your own risk, so take care of your key and the password. Coding procedure is a question of confidence; your work is in vain if your partner at the other end of the line carries out the decoding procedure through a computer connected to the net: both the program and the keys and passwords can be stolen from your encryption partner as well. The program is unable to safeguard the key and password; these are the user's responsibility.

The program was written in Visual Basic 6 for Windows XP or above. The program, when in operation, uses printable and commander characters alternately. Before you use the encryption program please check whether or not the program works on your computer. There are different soft and hardware systems so it is highly recommended to test your configuration first. **That is the reason why the program can be used at your own risk!**

If the program does not start, it has only one reason: Because the portable version does not need any installation so there is no background for running the program; it must be created by you before you start using the program.
Since not all operating systems include full VB runtime environment we need to make a copy of it and put it to the right place.

The downloaded program contains an „OCX_INS.exe" component file we need to start. The runtime background, the needed ocx' will be installed automatically. Now the program is ready to start. (If it does not start it is not the program's dysfunction.)

It means that the files needed for operation are not available. However, the MSDT program contains them. You may copy them from the work library. There are four files in the OCX folder on „C:\WINDOWS\system32\" subroutine (class) or you may start the OCX_INS.exe file that will automatically copy the OCX files.

For example: The new, special feature of RND, which includes many random number generators, cannot be detected even by the sensitive virus killers that could harm the program. Naturally, the downloaded REG file must be available in the library for getting started the program.

How the program operates:

Insert the pendrive into your work computer and click on „MSDT70.exe" file. At the present stage it operates in Hungarian and the English language. Essentially, the program already has the so-called basic key, but you can also create it yourself. If you can see the file „KEY.MDT" in the library you already have a basic key. If there is no basic key, follow the steps below. The actual key is being shown below in the middle. If there is no basic key follow the steps further down.

Click on the second tab TASK. You will then see a lock symbol at the third place above. Click on this symbol and the program generates the basic key, which then appears in the work library. If you already have the basic key, click on the lock symbol again and a window with Key will appear below, next to the file name. You may extend it or create a new name for the key. (For example: If you want to use this key with your partner, whose name on the internet is John Smith, then it is wise to name this key johnsmith1 to avoid confusing the keys you use in communication with other partners.)

Naturally, John Smith will be able to read the clear text if he was provided with the deciphering key before sending coded messages to him. When you click on the lock symbol you can see the partners' keys and select the appropriate one.

The size of the key can be set by the slider next to the lock symbol on the right from 100 to 10,000 at the present stage. These figures indicate the number of the matrixes the key contains. The program makes use of every matrix in a way that every character is being coded in the next matrix i.e. the whole key is in use when operating. A computer of higher capacity is able to create a large key within

seconds but it is wise to create a smaller key if you have a PC of lower capability. The program has unlimited autonomy within the source code in every aspect, i.e. the characters needed to upload the matrixes, size, shape, form and number of the matrixes can everyone change (MXM = more matrix = matrix more).

You can generate as many keys as you need or want initially or at a later time (johnsmith1 is now in use).

You can select the key at any time by clicking on the lock symbol.
After you have selected the key, you can work in three different modes (regimes) of the program.

1. **File-coding mode:** The file could be an exe, a written document, image or an audio file etc…This is the simplest, fastest and most secure mode. The difference in the characters does not impact this mode in any way.

2. **Manual-mode:** You type the open text into the window and the program encrypts it by making a different (coded) file of it. This is slower to encrypt due to the typing. (But you can use "cut and paste" as well.) This is a 'halfway mode' between the regimes 1 and 3.

3. **"Visible" mode:** This is the slowest mode. Coding of text as described in 2 (used only for coding and decoding of texts). The encrypted text appears in the form of sets of indecipherable figures. Because there is no one-to-one correspondence between sets in the program and it can be displayed only through this "visible" mode we will start the demonstration with it.

The Third ("Visible") Mode:

Coding process: Click on the icon of the left envelope named CODING TO SEND (the 1st symbol in the upper line in the TASK window). The appropriate key is being selected through the lock symbol. It will appear three active zones under the icon. You can select the first (upper) that enables the direct text typing, the second one for file-coding, and the third one for coding of selected files within a folder.

Select direct typing and click. The window Transfer will open. Click on the upper part of the window and you can type the text to be coded into it. (The window can be activated when you click on it.) This kind of operation is preferable if the text or message is short. If you want to encrypt longer texts (several pages), the file-coding mode is recommended.

You have now finished typing the short text. Click on the sign PROCESS at the bottom of the window. The encrypting procedure will be done automatically. You can see the coded section immediately in the window below in hexadecimal form. The program put this feature made up of HEXA codes on the CLIPBOARD and then PASTE it into the blank document.

You have now completed the type of encrypting process that consists only of hexadecimal figures that are between „„"" (double quotation marks). No matter how many times you repeat the encryption procedure of the original text, you will always get new and different figures of different extensions. This is the advantage of the program. It always uses different ways of coding and only the person(s) with access to the key can decrypt it. You can now send these HEXA sets of figures (do not forget the double „„"" at the beginning and end of the figures) via email or file the document for your coded archive. In this case you need to create only one key.

To be able to see and send the encrypted material, click twice (double click) on the bottom window. The program converts it to figures using ASCII codes and puts it on the clipboard. Now you can paste it to a blank document.

Decoding process: John Smith puts the received coded material on the clipboard (selects all from "" to"") and copies the material from the clipboard to the lower window of the program, DECODING FROM CLIPBOARD (after the actual key has already been selected). Click at the end of the message to let this window know that it is being activated. Click on PROCESS below and the decoded text appears above.

The „ ! „ quotation mark contains of the CHECK function in the newest version (v7.3) which swaps the content of the clipboard back and forth.

The Second Mode:

Coding process: Follow the steps of the Third Mode and type the text. Click on PROCESS. When the coded message appears in the window below, save it. Since the spam filters, antivirus programs, different software environments and firewalls could scan the extensions as well the program contains the most common **"JPG" Extension** in its source code (which could be change on demand.)

The coded feature will appear in hexadecimal form in the lower window. To continue the procedure, click on SAVE (There are four possibilities on the right of the lower window: SAVE, PRINT, DELETE, EXIT. Disregard the coded material on the clip board just focus on SAVE). The program will offer a new file name: SelfTextC_130458 (the numbers show the time at which it was saved). You may save it with the program name that appears automatically or you may add johnsmith001 to the filename to personalize it and avoid mixing up the message with other correspondence but **„SelfTextC_" must be kept** (which is needed to the decoding process). The program puts this "JPG" extended file into the work library and now it can be sent.

Decoding process: John Smith receives the "JPG" extension message through email and he can decode the message from the DECODING FROM RECEIVING-FILE after it opens it and selects the right key.

The First Mode:

Coding process: In this process, you may directly code exe-file, text, image, mp3 etc. files or even compressed documents (ZIP, RAR). On opening the program, the menu displays CODING SEND - CONVERSION OF FILE and it offers the way to find the text to be coded through BROWSE, SELECT. Select the text and double click on it and the process starts immediately. Click and the coded message is available in the work library.

Decoding process: To decrypt the message click on DECODING FROM RECEIVING - FILE. Double click on the coded text and the clear text will be out into the work library.

Converting a Folder:

Coding Process: With this mode, you can directly code every material within the folder by using the first mode. There is no need to code every material for the archive or to be sent separately; the program proceeds automatically and codes the whole content through the same key.

Decoding Process: The coded message can be decrypted in one procedure by using the first mode.

Enhanced Safety System (= MXM2)

The so-called Enhanced Safety System, the MXM2 is a new application, which is similar **to the double-keyed vault system applied by the banks**: If the material is coded with two different keys in one process (one after another), the keys must be kept at two secure locations or by two different individuals. In this case, you will be able to get access to the superior-coded material in the event of the two conditions being fulfilled. MXM2 program is recommended in case of sensitive governmental, nuclear energy, cyber security and sensitive business safekeeping issues.

Development of a DLL function group:

A new, so called DLL-function group was developed from the control algorithm that can be **embedded into any informatics process similar** to a simple door lock replacement; it is not necessary to remove the whole door if you want to rig up a new and better piece (MSDT(f10) DLL function group). Old system works perfectly with the new, encryption program.

Predicted programming theory against quantum machine hacking

By using our encryption keys, that works with 10.000 X 256 (!) factorials, (which can be unlimitedly upgraded) and that are orders of magnitude larger than the Googol-number (10100 in decimal representation used by mathematicians only for education), the coding process is easy even with any regular computer of the present time.

Compared to our current MSDT (f10) version, we are now entering the unlimited degree of freedom of encryption. By using the floating password (i.e. it is not registered/stored in any program, coded material or server), the next encryption key is send inserted into the coded material (conferring the principle of “opposite combs”). The distance, the size and attaching technics, fragment files can be varied in M (manual), A (automatic), RND (random) and INTAC (interactive) modes. In this process, you may also directly code not only clear text but also images, audio and video or compressed documents etc. as well. In addition, the software is able to manage biometrics (face recognition, fingerprint, voice tag, iris, etc.) in an interactive way.

Exploiting unlimited degrees of freedom and converting the software into open source code, re-coding, altering or replacing the operational modes mentioned above, it can be continuously upgraded so that we include the advantages of the planned capacity of the quantum machines in a way of pre-programming the coding and decoding procedures. Moreover, we can obtain QUANTUM-SUPERIORITY continuously with M / A / RND / INTAC modes or attaching ANY DIGITAL DATABASE selected freely, as well as permitting the Artificial Intelligence (AI) only partially (i.e. through the none-one-to-one correspondence and with the MXM2 program together).

FMU-C RAPTOR PROTOCOL  FMU-C RAPTOR TEAM