FMU-C RAPTOR PROTOCOL

FMU-C RAPTOR TEAM

MSDT(f10) - UNSTOLENABLE PASSWORD - FLOATING PASSWORD,
AZ ELLOPHATATLAN JELSZÓVAL MŰKÖDŐ REJTJELEZŐ SZOFTVER

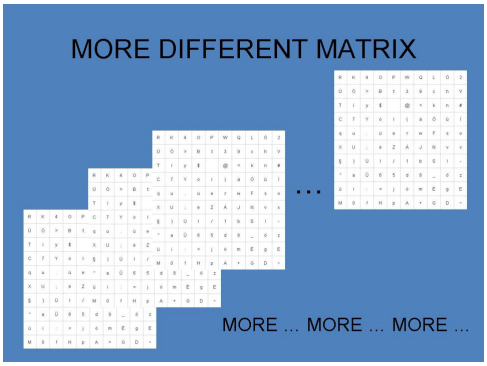
MSDT(f10) / Maximum Safety Data Transfer © fmu-c MSDT v7.0 és v7.3 (DCM)
rejtjelező szoftver leírása és használati utasítása

A jelenleg használt jelszavak, (pl. az email-belépésnél, vagy honlapra bejelentkezésnél) **mindig rögzítésre kerülnek valamely szerveren**, mivel a belépéskor össze kell hasonlítani a regisztrálás és a bejelentkezés felhasználónevét és jelszavát. Az ilyen rögzített jelszavakat a felhasználótól (netről behatolás, kémprogramok, keylogger …stb) , az interneten történő megtett csatornákon útközben, de akár a szerverről is meg lehet szerezni. (Ne csak betörésre gondoljunk, ott is emberek dolgoznak, és a hatóságoknál is, akik kikérhetik a jelszavainkat…) Az ilyen jelszó biztonságban van? Az ilyen jelszó titok a továbbiakban?

Az áttörés a kriptográfiában: az új generációs MSDT(f10) rejtjelező szoftver olyan jelszavakat használ a titkosításhoz, amelyek **nem kerülnek ki a felhasználójától!** Ez a **LEBEGŐ-JELSZÓ, FLOATING PASSWORD (UNSTOLENABLE PASSWORD = ELLOPHATATLAN JELSZÓ)!** Ennek a jelszónak a biztonsága (a használati utasítás betartása esetén) kizárólag a felhasználójára van bízva! **A LEBEGŐ-JELSZÓ nincs rögzítve sem a programban, sem a kódolt anyagban, sem hitelesítő-szerveren,** (működése leírva később). A **LEBEGŐ-JELSZÓ** használatával lehetővé vált az **új kulcsok interneten történő cseréje,** mely során a régi kulccsal, és jelszóval kódolunk egy új, cserélni kívánt szimmetrikus kulcsot.

A Kulcskészítés:

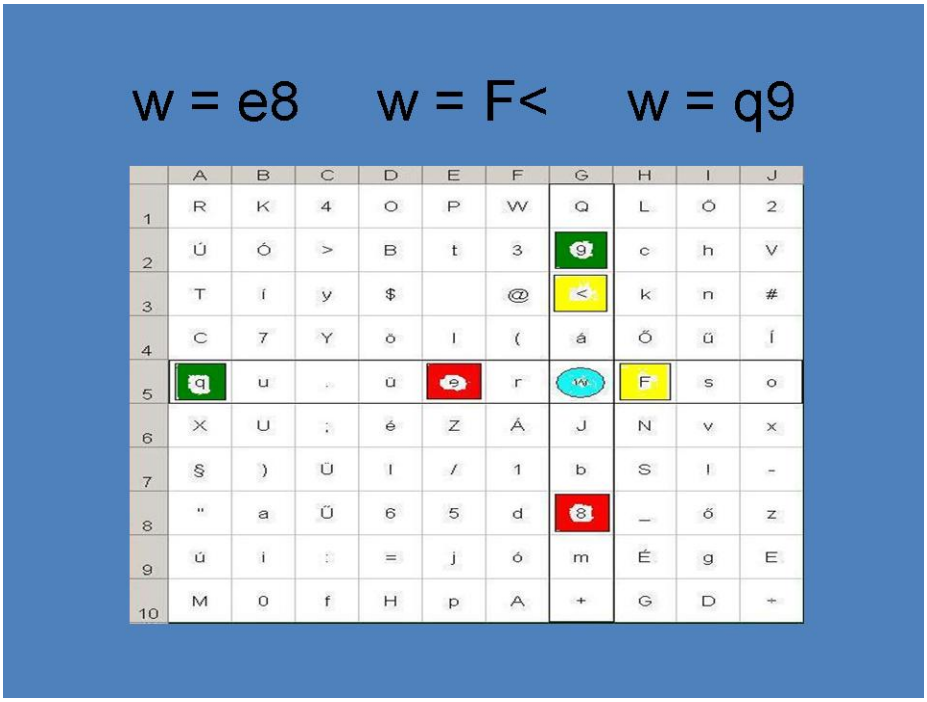
A program jelenleg 256 különböző karaktert használ. A karaktereket 16x16-os mátrixban véletlenszerűen összekeverve kapjuk az első mátrixot, majd ebből újra keverve lesz a második mátrix, és így tovább akár 10.000 mátrixig, mindig újrakeverve a következő mátrixot.



A mátrixok halmaza lesz a kulcs a kódoláshoz (KEY MDT). A karakterkészlet UNICODE feltöltése, az egyedi véletlenszám-generálás (RND), a karakterek összekeverése, az ASCII 0-255 közötti értékek elhelyezése, és a 256 faktoriális a Működtetés részleteiben leírva.

A Kódolás:

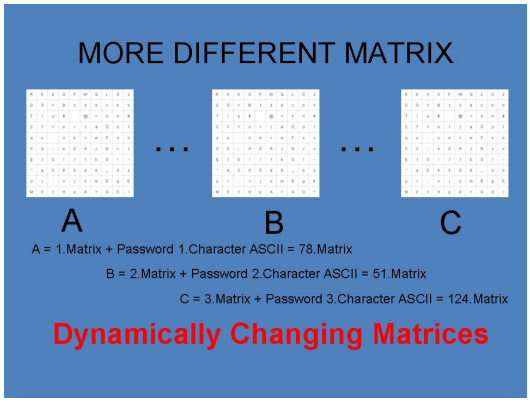
A felhasználó beállítja kulcskészítéskor, hogy hány mátrixból álljon a kulcs. Ez jelenleg 100 és 10.000 közötti mátrixból állhat, (melyet a forráskódban korlátlanul lehet fokozni). Az eredeti állomány (clear-text) első karakterét úgy kódolja az ALGORITMUS, (mely a HONLAP letöltés bemutató videójában látható), hogy **az eredeti karakter mátrixban történő elhelyezkedése az a metszéspont, amely kijelöl egy sort és egy oszlopot. A metszéspontban lévő karaktert helyettesítjük a mátrix így kijelölt sorában, majd az oszlopában lévő karakterek közül véletlenszerűen kiválasztott sor, majd oszlopkarakterrel.**



	A	B	C	D	E	F	G	H	I	J
1	R	K	4	O	P	W	Q	L	Ő	2
2	Ú	Ó	>	B	t	3	9	c	h	V
3	T	í	y	$		@	<	k	n	#
4	C	7	Y	ö	I	(	á	Ő	ű	Í
5	q	u	.	ü	e	r	w	F	s	o
6	X	U	;	é	Z	Á	J	N	v	x
7	§	)	Ü	I	/	1	b	S	I	-
8	"	a	Ű	6	5	d	8	_	ő	z
9	ú	i	:	=	j	ó	m	É	g	E
10	M	0	f	H	p	A	+	G	D	÷

A második eredeti karaktert már a második mátrixban kódolja ugyanígy, és így tovább, minden karaktert a következő mátrixban. Így a kulcsban lévő összes mátrixra szükség van a folyamatos kódoláshoz. Az utolsó mátrixban történő kódolás után az első mátrixban kezdődik ismét a folyamat.

A **DCM (v7.3)** verzióban a REF mezőbe beírt **LEBEGŐ-JELSZÓ** karakterei ASCII kódjainak számértéke hozzáadódik az aktuális mátrixok sorszámához, (az eredeti kódolás minden következő mátrixa egyenként van eltolva). Viszont **a jelszó használata során nem egyenként, hanem a jelszó egymást követő karaktereinek ASCII kódjai számértékével vannak eltolva a mátrixok**, így a jelszótól függően (mely most 200 karakterből állhat, de korlátlanul növelhető a forráskódban), dinamikusan változó mátrixokban (= DCM) történik a kódolás.



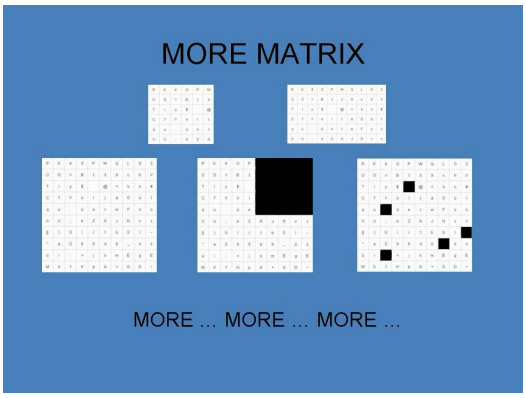
A Dekódolás:

A sor és oszlop azonosítók alapján nyert metszéspontból visszaállítjuk az eredeti karaktert. A program a kódolásnak megfelelően mindig a következő mátrixból állítja vissza a következő eredeti karaktert, a jelszó használatakor a dinamikus mátrixokat veszi figyelembe a program a dekódoláskor.

Fejlesztési lehetőségek: A kulcsot tartalmazó mátrixok mérete, alakja rugalmas lehet, valamint tartalmazhat úgynevezett fekete lyukakat. A kódolásból kizárt mezők helyének meghatározásához jelszót párosíthatunk.

A Működtetés részletei:

A kulcskészítéshez, kódoláshoz és dekódoláshoz jelenleg alapértelmezetten a 256 karakterből álló UNICODE karakterkészletet használjuk. A forráskódban **a karakterkészletet lehet cserélni** oly módon is, hogy pl. több karakterrel töltjük fel, így nagyobb mátrixokkal is dolgozhat a program. **A mátrixok a karakterek cseréjén túl különböző alakúak is lehetnek** (pl. T vagy L alakú), de ha négyzet, vagy téglalap alakú mátrixokat készítünk, akkor pl. keresztrejtvény formátumához hasonlóan „fekete lyukak" is lehetnek bennük. A T vagy L alakú, vagy a „fekete lyukakkal" feltöltött mátrixokban az üres helyeket eldobja a program, újraszámol az RND-vel, amíg aktív helyet nem talál. Ezekkel lassul a működés, de a „valamit valamiért" alapon ezzel érzékelhető a korlátlan szabadságfok.



A véletlenszám-generálás (RND) a rendszeridő másodpercének tizedesvessző után lévő tized-század-ezred…stb másodpercei helyéből a forráskódban meg van határozva 10 hely, pl. a tizedesvessző utáni 8-ik helytől a 17-ik helyig lévő futó számok. Ezen helyekből a rendszer egyjegyű RND száma határozza meg a futtatás pillanatában, hogy melyik szám jelölje ki az első sort. Pl. a rendszer RND-je a 2-es számot adta meg az indítás pillanatában, akkor a kijelölt, tizedesvessző utáni 8-17. hely közül a 9. helyen akkor futó szám jelöli ki a sort. A második generálás ad egy másik számot (mely, ha ugyanaz, mint az első, akkor eldobja), ez a szám egy másik sort jelöl ki, és először ez a két sor, a benne lévő karakterek lesznek megcserélve. A karakterek teljes összekeverése úgy történik, hogy a leírt módszer a mátrixnak 1-1 sorát cseréli meg, majd ezt követően 1-1 oszlop lesz kijelölve, és azok megcserélve. Utána ismét 1-1 sor, majd 1-1 oszlop, és így tovább. A forráskódban meghatározható, hogy hányszor cserélődjenek a sorok és oszlopok. Majd újabb keverésekhez a rendszeridő másik meghatározott száma következik. Így **véletlenszerűen fognak elhelyezkedni a karakterek a mátrixban.**

A program a kódoláshoz az **ASCII 0-255 közötti értékek** elhelyezése alapján már csak számokat használ, illetve hexadecimális értékekkel is dolgozik. Mivel jelenleg 16x16-os mátrixban dolgozik a program, ezért, hogy minden sor és oszlop cseréjére sor kerülhessen, a kijelölésre kerülő sorok és oszlopok felváltva az alsó, majd a felső, illetve a bal, majd a jobb széltől lesznek számolva.

A 256 különböző karakter kombinációja a **256 faktoriális érték**. Ez 507 db számból álló kombinációs lehetőség egyetlen mátrix kikeverésére. Ez a kombináció még csak a kódolt állomány első karakterére vonatkozik, minden kódolt karakterre másik mátrix, ismét egy ekkora kombináció vonatkozik. Ezt össze lehet hasonlítani a Googol-számmal, (http://hu.wikipedia.org/wiki/Googol), mely nagyjából egyenlő 70 faktoriálissal, és egyértelmű, hogy mennyi ideig tartana a kulcsból egyetlen mátrixot kikeverni ezekkel a kombinációkkal. Tehát aki a kulcs nélkül próbálja feltörni a kódolt állományt, az összes kombinációs lehetőséget (szűrni kell, hogy mind különböző legyen), azon túl az összes kombinációt lefuttatni a programmal, és értelmes szövegre szűrni. Persze fájlkódoláskor nincs értelmes szöveg... A feltörés szöveg-kódoláskor azért sem lehetséges, mert a használati utasításban leírt **MXM2 verzió**ban, a kétszeri kódolás köztes állapota nem hoz értelmes szöveget (gyakoriság-elemzés kilőve).

Ezt követi a **DCM (v7.3) verzió, mely egy LEBEGŐ-JELSZÓ ASCII-kódjaitól függően dinamikusan változtatja a mátrixokat.** Ezután jön a legfontosabb, a **vezérlő-algoritmus szerepe**, mely alapján az eredeti állomány (nem csak szöveges anyag, hanem bármilyen más fájl (exe, avi, jpg, mp3, rar…stb)) **minden egyes új kódolása után más-más kódolt anyagot kapunk. Ez a NEM EGY-EGYÉRTELMŰ MEGFELELTETÉS** (a fent leírtakkal együtt) **teszi feltörhetetlenné a titkosítást az MSDT(f10)-programmal!**

MSDT(f10) © fmu-c MSDT v7.0 és v7.3 (DCM) rejtjelező
szoftver használati utasítása

Alapszabályok:

Az MSDT v7.0 program telepítést nem igényel, portable verzió, memóriakártyáról, pendrive-ról üzemeltethető biztonságosan. Ha internetre kötött gépen dolgozunk a programmal, bármikor kilophatják a neten keresztül a kulcsot vagy a jelszót. Minden hivatásos rejtjelező program külön **internet nélküli munkagépen fut**, és csak a kódolt állományt adják át az internet-kapcsolattal rendelkező gépnek (pl. egy másik pendrive-on). A munkagépben nem hagyjuk a programot, és a kulcsot tartalmazó pendrive-ot, arra külön vigyázunk! Ezért **a programot mindenki csak a saját felelősségére használhatja, mindenki úgy vigyáz a kulcsára és jelszavára, ahogy akar!**

Itt lép be a **BIZALMI HÁLÓ,** mely szerint mi hiába dolgozunk az MSDT-vel biztonságosan, ha a partner, akivel pl. levelezünk, internetes gépen használja a programot. Ha tőle kilopják a közösen használt kulcsot vagy jelszót, akkor olyan, mintha mi sem vigyáznánk rá! **A kulcs és a jelszó megőrzése, közvetítése nem a program feladata**, hanem az azt használó személyeké.

A program Windows Xp operációs rendszerben, Visual Basic 6 nyelven lett fejlesztve, tehát az Xp-n, vagy fölötte futtatható. A program a működtetése során nyomtatható, és vezérlő karaktereket is használ. Mivel a felhasználók eltérő hardver- és szoftver-környezetekben használják a programot, minden személynek a használatba vétel előtt le kell tesztelni a megfelelő beállításokat. A felhasználók ezért is **csak a saját felelősségükre használhatják a programot!**

Amennyiben nem indul a program: egyetlen oka lehet. Mivel a portable verziót nem kell telepíteni, így a futtató környezet sincs feltelepítve, azt nekünk kell az első használat előtt kialakítani. Mivel nem minden operációs rendszer tartalmazza a VB teljes futtató környezetét, azt emiatt bemásoljuk a megfelelő helyre. A letöltött program tartalmaz egy „**OCX_INS.exe”** komponens fájlt, ezt kell elindítanunk, így automatikusan feltelepül a gépünkre a futtató környezet, a megfelelő az ocx-ek. Ezután már indul a program. Ha nem, akkor az már nem a program hibája.

Egy példa: az RND, mely több véletlen generátorból áll, speciális, újított kialakítását nem ismerik a víruskeresők, amelyek főleg, ha érzékenyre vannak állítva, tekinthetik vírusos anyagnak a programot. Természetesen a munkakönyvtárban a letöltött REG-fájlnak is ott kell lennie, hogy a program induljon.

A program működtetése:

Miután a munkagépbe helyezzük a pendrive-ot, az „**MSDT70.exe”** fájlt indítjuk. Jelen fejlesztés során magyar vagy angol nyelven futtatható a program. Alapesetben a program már rendelkezik egy alap-kulccsal, de anélkül is megkaphatjuk, akkor mi készítjük el az alap-kulcsot. Amennyiben a munkakönyvtárban látható a **„KEY.MDT”** fájl, akkor van alap-kulcsunk. A program az aktuális kulcsot mindig alul középen mutatja. Ha nincs alap-kulcs, akkor az alábbiak szerint készíthetjük el.

Átkattintunk a második (Feladat) fülre, látható a felső sorban a 3. helyen egy lakat szimbólum. Arra kattintva a program elkészíti az alap-kulcsot, és ezt követően látható is lesz a munkakönyvtárban. Ha már volt alap-kulcsunk, akkor, ha ismét rákattintunk a lakatra, megjelenik egy ablak, amelyben alul **Key** név szerepel a fájlnév mellett. Ezt kiegészíthetjük, vagy helyette írhatunk be egy egyedi, általunk választott nevet az új kulcsnak. (Pl. ha ezt a kulcsot Kiss Lászlóval kívánjuk használni internetes levelezésünk során, akkor célszerű **kisslaci1**-et adni a kulcsnak, mert amikor már több kulcsot használunk, akkor tudjuk, hogy melyik kulcs kihez tartozik.) Természetesen a Kiss László csak akkor tudja elolvasni a részére megküldött anyagot, ha ő is rendelkezik ezzel a kulccsal. A kulcs nagyságát lakattól jobbra lévő csúszkával állíthatjuk be, jelen fejlesztésben 100 és 10.000 érték között. Ez azt jelenti, hogy hány darab mátrix lesz a kulcsban. Minden mátrixot használ a program oly módon, hogy minden karaktert a következő mátrixban kódolja, tehát az egész kulcs használva van. Egy nagy teljesítményű gép a legnagyobb kulcsot is néhány másodperc alatt elkészíti, lassú géppel (ha nem akarunk több percet várni, akkor) érdemes kisebb kulcsot készíteni. **A forráskódban korlátlan szabadságfokkal rendelkezik a program** minden tekintetben, így a mátrixok feltöltéséhez használt karaktereket, a mátrixok nagyságát, alakját, formáit, egy kulcson belüli mátrixok darabszámát (**MXM** = more matrix = matrix more) bárki megváltoztathatja.

Annyi kulcsot készíthetünk, amennyire szükségünk van, illetve később is készíthetünk bármennyit (jelen esetben, pl. használjuk a **kisslaci1** kulcsot). **A lakatra bármikor kattintva választhatjuk ki, hogy melyik kulccsal kívánunk dolgozni.**

A kulcs kiválasztása után dolgozhatunk a programmal. 3 üzemmódban (rezsimben) lehet használni a programot. Az 1. rezsim a fájlkódolás, ebben pl. exe-fájlt, megírt dokumentumot, képet, hangfájlt…stb kódolunk. Ez a legegyszerűbb, leggyorsabb és legbiztonságosabb üzemmódja, nem függ a karakterkészletek különbözőségétől. A 2. rezsimben kézzel írjuk be a kódolandó szöveget, de egy külön fájlt kódol belőle a program. Ez kissé lassabb, mivel nekünk kell begépelni a programba a szöveget (de betehetjük vágólapról is), ez egy köztes állapot az 1. és 3. rezsim között. A leglassabb a 3. rezsim, mert oda mi gépeljük be a szöveget (ez az üzemmód, a 2.-hoz hasonlóan csak szöveg kódolására és dekódolására alkalmas), és a dekódolt állomány is szöveges formátumú lesz, igaz, hogy mások számára értelmezhetetlen karakterek halmaza csak. Mivel a 3. rezsimben látható **a nem egy-egyértelmű megfeleltetés a halmazok között**, ezért ezzel kezdjük a bemutatót.

A harmadik rezsim:

Kódolás: A FELADAT ablakban a felső sor 1. szimbóluma KÓDOLÁS KÜLDÉSHEZ nevű bal oldali borítékra kattintunk. (A megfelelő kulcsot kiválasztjuk a lakattal.) Alatta megjelenik aktívan három sáv, amelyek közül választhatunk. A felső a közvetlen beírást teszi lehetővé, a középső egy fájl konvertálását (kódolását), míg az alsó egy kiválasztott mappában lévő összes kijelölt fájl egy folyamat során történő konvertálását.

Kezdjük a közvetlen beírással. Arra kattintva megjelenik a Transfer feliratú ablak, melynek a felső részébe kattintunk az egérrel, és már írhatjuk is be a kódolandó szöveget. (Mindig az az ablak az aktív, amelyikbe belekattintunk az egérrel.) Ennek rövidebb szöveg begépelésénél van előnye, ha pl. oldalakat akarunk kódolni, akkor ne írjuk be, hanem a már kész szöveget a fájl-konvertálással kódoljuk. Jelen esetben írjuk be a szövegünket, majd, ha készen vagyunk, akkor az ablak alsó részén lévő folyamat indítására kattintunk. Ekkor a kódolás megtörténik, és az alsó ablakban látható a kódolt állomány hexadecimális megjelenítése. A program ezt a HEXA kódokból álló állományt vágólapra teszi. A vágólap tartalmát beilleszthetjük üres dokumentumokba.

Ezzel készen van a program egyik típusú kódolása. Ez a hexadecimális megjelenítésű kódolt anyag dupla „„””-ek között van. Ha többször kódoljuk ugyanazt a szöveget, látni fogjuk, hogy mindig más-más karakterekből áll. Ez az egyik legfőbb erénye a programnak, hogy mindig másként kódol, ezért nem lehet visszafejteni. Ezt a HEXA karakterekből álló halmazt (a dupla „„””-el együtt) küldhetjük e-mailben. Vagy archiválhatjuk, ha saját részre készítjük, ekkor nem is adunk a kulcsból másnak.

Dekódolás: Kiss László a kapott anyagot vágólapra teszi a dupla „„””-el együtt, és a vágólapról másolja be a DEKÓDOLÁS – VÁGÓLAPRÓL alsó ablakába, (miután az aktuális kulcs már ki van választva), majd az állomány végéhez kattint, (így tudja a program, hogy ez az ablak az aktív), majd alul a folyamat indítása gombra kattintva olvasható felül a dekódolt levél. A „ ! „ fül a CHECK funkciót tartalmazza a legújabb (v7.3) fejlesztésben, mely a vágólap tartalmát fordítja oda-vissza.

A második rezsim:

Kódolás: A harmadik rezsimhez hasonlóan járunk el addig, amíg begépeljük a kívánt szöveget, és amikor a folyamat indítását követően az alsó ablakban megjelenik a kódolt állomány, azt egy fájlba mentjük le. Mivel az interneten történő küldés, a megfelelő szoftverkörnyezet, a tűzfalak, víruskeresők különböző érzékenységet mutatnak a kiterjesztésekre is, ezért az eredeti MDT kiterjesztésből az egyik legáltalánosabb kiterjesztést, **a „JPG” kiterjesztést adtuk meg a forráskódban**. (Ezen is lehet változtatni későbbi igény szerint.) A mentést úgy érjük el, hogy amikor a folyamat indítása után az alsó ablakban megjelent a kódolt állomány hexadecimális alakzatban, nem a vágólapra tett állománnyal foglalkozunk, hanem az ablak alsó részén jobb oldalon lévő 4 gomb közül (az 1. a mentés, a 2. a nyomtatási funkció, a 3. a törlés, és a 4. a kilépés a programból), most az elsőre (MENTÉS) kattintunk. Ekkor felkínálja a program az új fájl nevét, mely „SelfTextC_130458”, melyből a számok a mentés idejét jelentik. Ezt így is menthetjük, vagy a felkínált név mögé még beírhatjuk, pl. hogy kisslaci001, hogy teljesen egyedi neve legyen a fájlnak, mert sok kódolás után jó, ha tudjuk, hogy kinek írtuk ezt a levelet. **A „SelfTextC_”-nek meg kell maradnia**, (a dekódoláshoz szükséges eléréshez), de az időt jelző számot lehet cserélni bármire. Ez a fájl „JPG” kiterjesztéssel a munkakönyvtárba kerül, amelyet lehet küldeni.

Dekódolás: Megkapja Kiss László email-ben a „JPG” kiterjesztésű anyagot, akkor a DEKÓDOLÁS FOGADÁS – FÁJLBÓL megnyitása után dekódolhat, a programmal előzőleg helyes kulcs behívása után.

Az első rezsim:

Kódolás: Ebben az esetben pl. egy exe-fájlt, kész szöveges dokumentumot, képet, mp3 fájlt…stb, vagy tömörített anyagokat (ZIP, RAR), közvetlenül kódolunk, oly módon, hogy a program megnyitása után a KÓDOLÁS KÜLDÉS – FÁJL KONVERTÁLÁSA megnyomása után felkínálja a program az elérési útvonalat a kódolandó állomány megkeresésére. Amint kijelöljük, hogy mit kívánunk kódolni, arra az állományra kétszeri kattintás után megjelenik a folyamat indítása. Erre kattintva kódolunk, és a kész anyag a munkakönyvtárban lesz.

Dekódolás: A dekódolás úgy történik, hogy a DEKÓDOLÁS FOGADÁS – FÁJLBÓL kiválasztása után kétszer rákattintunk a kódolt anyagra, majd a folyamatot indítjuk, így a program visszakódolja az eredeti anyagra, és azt elhelyezi a munkakönyvtárban.

A Mappa konvertálása és visszaállítása:

Kódolás: Kiválasztjuk a mappát, amelyben bármennyi kódolásra szánt anyag található, és azokat az első rezsimnek megfelelően lekódoljuk. Ezzel saját archiválásra szánt, vagy küldendő anyagokat nem kell egyenként kódolni, azt a program egymás után ugyanazzal a kulccsal végrehajtja.

Dekódolás: A kiválasztott kódolt anyagokat mappa visszaállítással egyetlen folyamatban az első rezsimnek megfelelően dekódoljuk.

Emelt szintű biztonsági Rendszer (= MXM2)

Új alkalmazási mód az ún. emelt szintű biztonsági rendszer, az MXM2, **a bankok által alkalmazott dupla kulcsos széfekhez hasonlóan**. Amennyiben az állományt egymás után két különböző kulccsal kódoljuk, a két kulcsot különböző helyen őrizzük. Vagy a kulcsokat két különböző személyre bízzuk, akkor csak a két személy együtt tudja megnyitni az emelt szintű kódolt anyagot. Ez pl. a kiemelt kormányzati, atomenergetikai, kiberbiztonsági vagy üzleti titkok biztonságát képes úgy szavatolni, amikor nem akarjuk egy személyre bízni a biztonsági anyagot.

DLL függvény csoport kifejlesztése:

Kifejlesztésre került a vezérlő algoritmusból egy DLL függvény csoport, amely **bármely jelenleg is működő informatikai eljárásba integrálható**. Hasonlatképpen: a jelenleg használt ajtók zárjaiból csupán a zárbetétet kell kivenni, és helyére egy új zárbetétet (az MSDT(f10) DLL függvényét) behelyezni, és nem kell egész ajtót cserélni, ezzel a régi rendszerek az új eljárás alapján működtethetők.

A kvantumgépek feltörései elleni előtervezett programozás elmélete

A Googol értéknél (amelyet a matematikusok is csak oktatásra használnak), nagyságrendekkel nagyobb 10.000 X 256 (!) faktoriálissal dolgozó kulcsunkat (mely korlátlanul fejleszthető), és az azzal történő kódolást könnyedén végezzük a jelenlegi gyenge gépekkel.

A most kidolgozott MSDT(f88h) verzióval az eddigi (f10) változathoz képest az alábbiakkal lépünk tovább a korlátlan szabadságfokú üzemmódba. A lebegő jelszót (internetről lekapcsolt) munkagépeken használva a következő kulcsokat a kódolt anyagban (szemben álló fésűk elvén) befűzve küldjük. A töredék állományok távolsága, nagysága és befűzési módja is variálható minden M (manual) / A (automatikus) / RND (random) / INTAC (interaktív) módon. Ehhez is, mint a clear-texthez, nem csak szöveget, hanem képet, hangot, videót, tömörített állományt…stb használhatunk mintavételnek. Továbbá interaktív módon a felhasználók biometrikus (arcfelismerés, ujjlenyomat, hangminta, írisz…stb) állományából is mintát vehet a szoftver.

A korlátlan szabadságfokokat kihasználva, a szoftvert nyílt forráskódúvá alakítva, újrakódolva, kiegészítve vagy cserélve az eddig leírt üzemmódokat úgy lehet fejleszteni folyamatosan, hogy a későbbi kvantumgépek kapacitásainak tervezett előnyeit beszámítjuk azzal, hogy már kvantumra programozzuk a kódolást és dekódolást. Továbbá a M / A / RND / INTAC konfigurációkkal, illetve a felhasználók által szabadon választható BÁRMILYEN DIGITÁLIS ADATBÁZIS HOZZÁFŰZÉSÉVEL, valamint a MI (Mesterséges Intelligencia) csak részben történő engedélyezésével, (a nem egy-egyértelmű megfeleltetéssel, és az MXM2-vel együtt) folyamatos KVANTUMFÖLÉNYRE teszünk szert!

FMU-C RAPTOR PROTOCOL  FMU-C RAPTOR TEAM