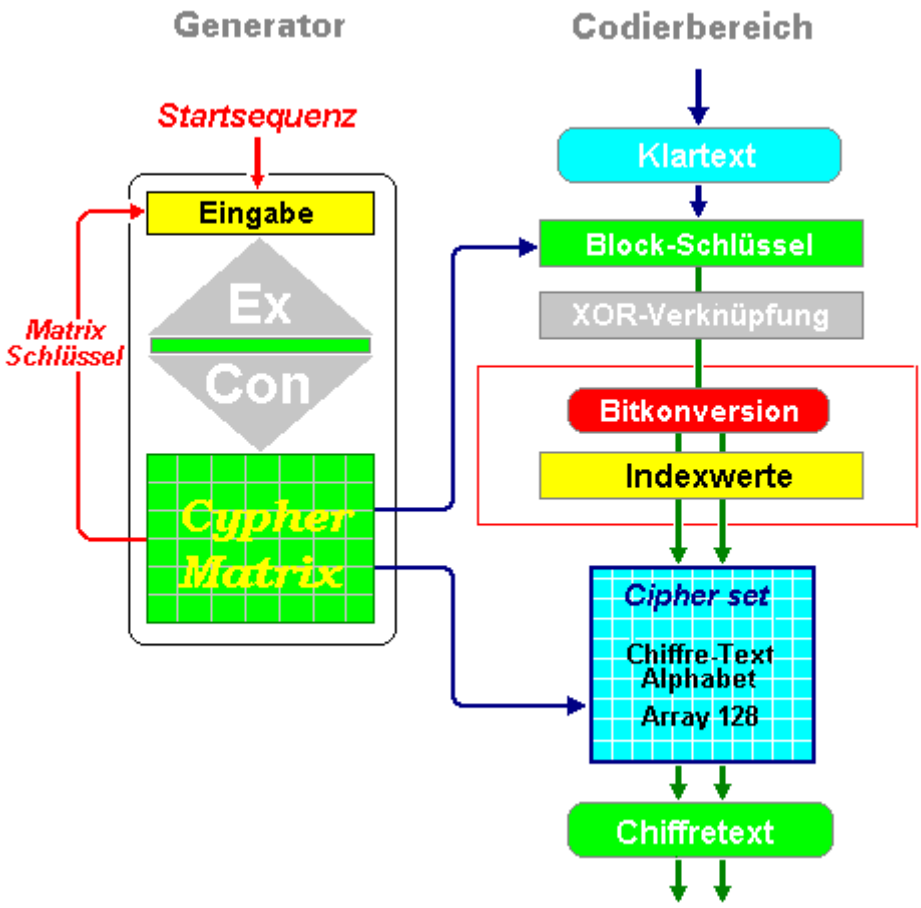
Sicherheit des „CypherMatrix“ Verfahrens

(Ernst Erich Schnoor)

Das CypherMatrix Verfahren teilt sich in zwei Bereiche: **Generator** zur Erzeugung aller notwendigen Bestimmungsdaten und **Codierbereich** für die Durchführung der verbundenen Anwendung (z.B: Verschlüsselung).



Beide Bereiche werden kombiniert, können aber auch getrennt und eigenständig verwendet werden. Einzelheiten sind unter telecypher.net/CypherMatrix.pdf dargelegt [#1]. Entsprechend der Teilung in zwei Bereiche ist auch die Sicherheit für jeden Bereich gesondert zu betrachten. Über allem als Grundfunktion steht die „**Bit-Konversion**“.

1 Bit-Konversion

Bitkonversion ist die Umwandlung einer Bitfolge von einem Bitsystem in ein anderes Bitsystem. Dabei bleiben die Anzahl der Bits und ihre Reihenfolge gleich. Kein Bit wird hinzugefügt und kein Bit wird weggelassen. Nur die Anzahl der Bits in einer Einheit (Unit) ändert sich, und damit die Struktur der Bitfolge. Die dezimalen Werte der neuen Einheiten sind Indexwerte für das zugeordnete System-Alphabet.

Zu jedem Bitsystem gehört ein sinnvolles „System-Alphabet“. Das System-Alphabet ist der wichtigste Bestandteil der Computertechnik. Es ist die Grundlage für die Visualisierung des Inhalts der Bitfolgen. Ohne die sachgerechte Definition eines Alphabets im jeweiligen Bitsystem könnte mit dem Computer gar nicht gearbeitet werden. Im Bitsystem zur Basis 8 ist das System-Alphabet der ASCII-Zeichensatz in seiner jeweiligen Ausprägung.

1.1 Systematik der Bitfolgen

Die Systematik der Bitfolgen zeigt folgende Übersicht:

Systembasis	System-alphabet	Bitfolgen in Einheiten Indizierung	Längen-verhältnis
Bitsystem zur Basis 1	2	0 1 0 0 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 0.	1:8
Bitsystem zur Basis 2	4	01 00 11 10 01 10 11 11 01 11 01 00 01 10 00 01 01 10 00. 1 0 3 2 1 2 3 3 1 3 1 0 1 2 0 1 1 2 0	1:4
Bitsystem zur basis 3	8	010 011 100 110 111 101 110 100 011 000 010 110 001 ... 2 3 4 6 7 5 6 4 3 0 2 6 1	1:2,66
Bitsystem zur Basis 4	16	0100 1110 0110 1111 0111 0100 0110 0001 0110 0010 ... 4 14 6 15 7 4 6 1 6 2	1:2
Bitsystem zur Basis 5	32	01001 11001 10111 10111 01000 11000 01011 00010 01.. 9 25 23 23 8 24 11 2	1:1,6
Bitsystem zur Basis 6	64	010011 100110 111101 110100 011000 010110 001001 ... 19 38 61 52 24 22 9	1:1,33
Bitsystem zur Basis 7	128	0100111 0011011 1101110 1000110 0001011 0001001 ... 39 27 110 70 11 9	1:1,143
Bitsystem zur Basis 8 *Bereiche*	256	01001110 01101111 01110100 01100001 01100010 011.. 78 111 116 97 98 4E 6F 74 61 62 N o t a b ... **Stromchiffre, Blockchiffre mit Längenkongruenz, Feistel-Netze ECB, CBC, CFB, OFB, DES, IDEA, AES, RSA, differenzielle und lineare Kryptanalyse, fast alle weiteren Programme**	1:1
Bitsystem zur Basis 9	512	010011100 110111101 110100011 000010110 001001100 156 445 419 22 76	1:1,79
Bitsystem zut Basis 10	1024	0100111001 1011110111 0100011000 0101100010 01100 313 759 280 354	1:1,6
Bitsystem zur Basis 11	2048	01001110011 01111011101 00011000010 11000100110 ... 627 989 194 1574	1:1,46
Bitsystem zur Basis 12	4096	010011100110 111011110100 011000010110 001001100 1 1254 3956 1558 613	1:1,33
Bitsystem zur Basis xx	*div*	x x x x x	1: xxx

Im Längenverhältnis kommen die Längen von Klartext und Geheimtext zum Ausdruck. Im Bitsystem zur Basis 8 sind Klartext und Chiffretext gleich lang, in allen anderen Fällen nicht.

1.2 Buchstaben-Verschlüsselung

Bei Einführung der digitalen Techniken traten an die Stelle der Buchstaben zwar die Bits, aber im Ergebnis wurden die Grundsätze der Buchstaben-Verschlüsselung beibehalten. „Verschlüsselt wird nach wie vor gemäß den Grundsätzen der Substitution und Transposition“ [#10].

In fast allen derzeitigen Verschlüsselungsverfahren (DES, AES u.a.) ist für jedes Klartext-zeichen ein bestimmtes Chiffrezeichen vorhanden. Beide Bereiche – Eingaben und Ausgaben – arbeiten im selben Chiffrealphabet (erweiteter ASCII-Zeichensatz) und damit

auch im gleichen Bitsystem zur Basis 8. Nur so können Wiederholungsmuster, Wortkombinationen, Häufigkeitsanalysen und Bigramme im Chiffretext zu vergleichbaren Merkmalen im Klartext führen [#2]. Die bekanntesten Angriffe, wie Strukturanalyse, „known plaintext attack“, „chosen plaintext attack“ und auch „differenzielle“ und „lineare“ Kryptoanalyse u.a. können nur Erfolg haben, wenn für jedes Chiffrezeichen auch ein bestimmtes Klartextzeichen vorhanden ist, also Klartext und Geheimtext die gleiche Länge haben [#3,#4,#5]. Insoweit besteht ein einheitliches Ordnungssystem zur Basis 8.

1.3 Wegfall des einheitlichen Ordnungssystems

Im Gegensatz zu bisherigen Verfahren werden im CypherMatrix Verfahren nur die folgenden Techniken angewendet: Startsequenz mit optimal 42 Zeichen, laufende Erzeugung von Block-Schlüssel, Matrix-Schlüssel und System-Alphabete sowie System-Wechsel durch Bitkonversion in jeder Runde (z.B: Block von 42 Zeichen). Alle weiteren in aktuellen Verschlüsselungen angewendeten Schritte zur Abwehr von Angriffen sind nicht erforderlich. Das einheitliche Ordnungssystem zur Basis 8 fällt fort und beide Bereiche, Klartext und Chiffretext, können nicht mehr sinnvoll verglichen werden. Damit entfällt auch die Basis für alle bekannten Angriffszenarien und wir können sie vergessen.

2 Der Generator

Der Generator wird für vielfache Aufgaben der Kryptographie eingesetzt (Verschlüsselungen, Hashwertberechnungen, Signaturverfahren, Zufallszahlen u.a.).

Die **Startsequenz** (Passphrase) zum Initialisieren des Generators kann zwischen 36 und 64 Zeichen (optimal 42 Bytes) festgelegt werden. Bei Eingabe über die Tastatur mit etwa 100 Zeichen umfasst die Startsequenz folgenden Schlüsselraum:

36 Bytes →	100^36 = 1E+ 72	Länge: 240 Bit	Entropie: 239.188
42 Bytes →	100^42 = 1E+ 84	Länge: 280 Bit	Entropie: 279.042
64 Bytes →	100^64 = 1E+128	Länge: 426 Bit	Entropie: 425.207

Die Start-Sequenz wird nur einmal zum Start des Verfahrens verwendet und nicht auf der Festplatte gespeichert.

Der **Matrix-Schlüssel** aus der CypherMatrix (256 Zeichen) mit 42 Zeichen entnommen, entfaltet maximal folgenden Schlüsselraum:

42 Bytes →	256^42 = 1.4E+101	Länge: 336 Bit	Entropie: 336.000

Bei einem Angriff auf den Matrix-Schlüssel mit 42 Bytes Länge ergibt sich eine Entropie von 336 und eine exponentielle Komplexität von O(2^336) = **1.4E+101** [#6].
Angenommen in einem „brute force“ Angriff dauere eine Exhaustionsrunde 1 nano sec, dann würden 5E+89 Jahre notwendig sein, den Schlüssel mit Sicherheit zu finden.

Die für die laufende Sicherheit des Generators wichtigen Schritte sind vor allem:

a) Vermeidung von Kollisionen,
b) erste Einwegfunktion: „Expansion“ zur Hashfunktionsreihe und
c) zweite Einwegfunktion: „Kontraktion“ zur BASIS VARIATION.

Mit Einbindung dieser Funktionen ist eine rückwärts gerichtete Bestimmung vorhergehender Daten nicht möglich.

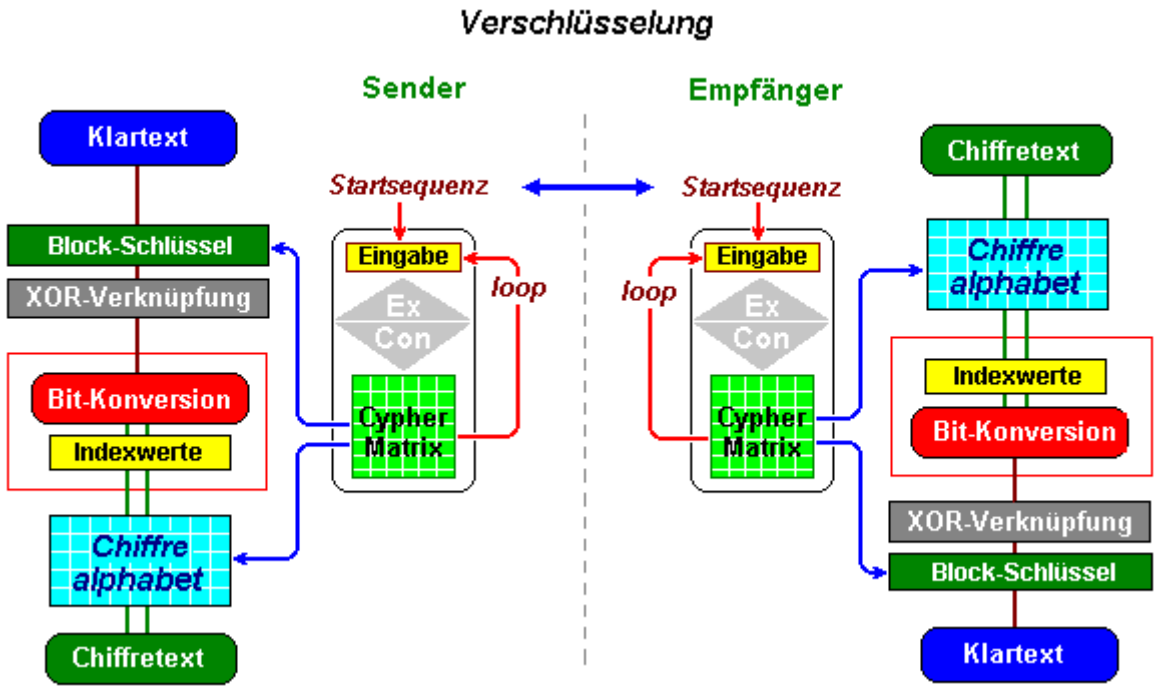
Die **CypherMatrix** (16x16 Zeichen) wird in jeder Runde neu generiert. Eine Wiederholung der gleichen Matrix tritt erst nach 256! (Fakultät) = 8.57E+506 Fällen auf (2*10^1684).

Das **System-Alphabet** mit 128 Zeichen nimmt 212 Bytes aus der CypherMatrix von 256 Zeichen. 44 Zeichen bleiben unberücksichtigt.

212 Bytes → 212^128 = 5.9E+297 Umfang: 989 Bit Entropie: 989,173

3 Codierbereich

Das folgende Schema zeigt die Zusammenhänge:



Die Verschlüsselung wird in folgenden Alternativen durchgeführt:

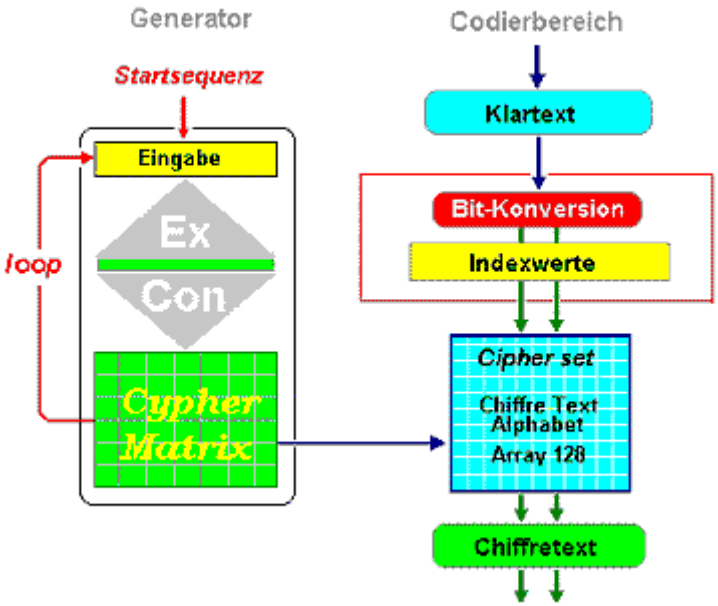
Basis-Coding: Bitkonversion allein ohne weitere Operationen oder
Verbund-Coding: Bitkonversion mit zusätzlichen Operationen, und zwar:

a) mit XOR-Verknüpfung (voran- oder nachgestellt) oder
b) mit weiteren Operationen verbunden.

3.1 Basis-Coding

Im „Basis-Coding“ Modus erfolgt die Bit-Konversion direkt vom bisherigen Bitsystem zum angestrebten Bitsystem ohne Zwischenschritte. Das Schreiben des Chiffretextes im Codierbereich geschieht mit zwei Schritten:

1. Bit-Konversion
   8-bit Klartextwerte → 7 bit Indexwerte (0 ...127)
2. Bestimmung des Chiffretexts
   7-bit Indexwerte → Chiffre-Alphabet (0...127) → Chiffretext.



3.2 „Verbund-Coding“

Beim Verbund-Coding werden verschiedene Operationen seriell verbunden, z.B. XOR-Verknüpfung mit Bit-Konversion und weiteren Operationen (dyn24, exchange).

Mit vorgeschalteter XOR-Verknüpfung vollzieht sich die Verschlüsselung in drei Funktionen:

1. Partielles dynamisches „One-time-pad“
   Klartext-Block → Block-Schlüssel → 8-bit XOR-Verknüpfung
2. Bit-Konversion
   8-bit XOR-Verknüpfung → 7 bit Indexwerte (0 ...127)
3. Bestimmung des Chiffretexts
   7-bit Indexwerte → Chiffre-Alphabet (0...127) → Chiffretext.

Beim „Verbund-Coding“ besteht außerdem für jeden Block-Schlüssel ein partielles „**one-time-pad**“ [#7]. Klartextblock und Block-Schlüssel sind gleich lang und der Schlüssel wird auch nicht wiederholt. In jeder Runde wird ein anderer Schlüssel aus der betreffenden CypherMatrix entnommen. Das ergibt für den gesamten Verschlüsselungsvorgang eine Kette als zusammenhängende „one-time-pad“-Funktion. Nach derzeitiger Auffassung wird dadurch eine absolute Sicherheit erreicht [#8].

3.3 „brute force“ Angriff

Immerhin bleibt noch die Möglichkeit einer „brut force“ Attacke [#9]. Einem Angreifer sind grundsätzlich nur der Chiffretext und das CypherMatrix Verfahren bekannt. Das jeweilige Programm und die einzelnen Steuerungsparameter, einschließlich der Startsequenz, kennt er nicht. Ihm bleiben die folgende Angriffszenarien:

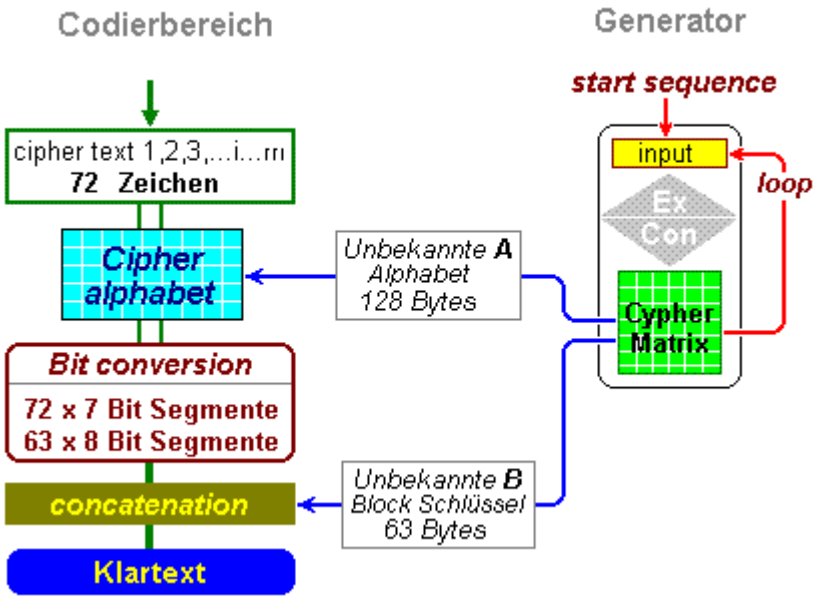
1. Am Beginn die „Startsequenz“ zu suchen,
2. in das laufende Verfahren einzusteigen oder
3. vom Ende her den Weg vom Chiffretext zum Klartext zu finden.

Der Datenumfang für einen Angriff auf die Startsequenz ist bereits oben dargestellt.

Ein Versuch, aus dem laufenden Verfahren vom Chiffretext auf die Startsequenz oder den Matrix Schlüssel zu schließen, muss aussichtslos bleiben. Zwischen beiden gibt es keine Verbindung und beide werden für die Verschlüsselung nicht verwendet.

Weiterhin bleibt die Möglichkeit, vom Chiffretext auszugehen und den Vorgang vom Ende her aufzurollen. Dazu müsste das letzte System-Alphabet im Bitsystem zur Basis 7 und die Verteilung der 128 Alphabet-Zeichen herausgefunden werden. Die richtige Verteilung könnte nur durch eine Iteration gesucht werden. Für die Verteilung besteht ein Kombinationsraum von 128! (Fakultät) = **2.85E+215** Möglichkeiten.

Wenn für eine bestimmte Runde, aus welchen Gründen auch immer, das System-Alphabet zur Basis 7 dennoch bekannt werden sollte, bleibt die Aufgabe, den jeweiligen Block-Schlüssel herauszufinden. Die Entschlüsselung geschieht in jedem Durchgang wie folgt:



Das Verfahren enthält drei Funktionen:

1. Klartextblock --> **Block-Schlüssel** --> -8 bit XOR-Sequenzen
2. 8-bit XOR Sequenzen --> 7-bit Index-Werte
3. 7-bit Index-Werte --> **Chiffre-Alphabet (128)** --> Chiffretext

In diesen Funktionen sind die Parameter **Block-Schlüssel** und **Chiffre-Alphabet** zwei voneinander unabhängige Variable. Es gelten:

cm = f [f1 (an, **k1**), f2 (b1, b2), f3 (b2, **k2**)]
an = f [f3 (cm, **k2**), f2 (b2, b1), f1 (b1, **k1**)]

fx = funktionale Verbindung
an = Klartext
k1 = **Block-Schlüssel**
b1 = 8-bit Sequenz
b2 = 7-bit Index-Wert
k2 = **Chiffre-Alphabet (128)**
cm = Chiffretext

Die Ermittlung des Chiffretextes „**cm**“ und die retrograde Suche nach dem Klartext „**an**“ zeigen sich somit als Gleichungen mit zwei unbekannten Veränderlichen: **k1** und **k2**. Das führt bekanntlich nur dann zu einer eindeutigen Lösung, wenn eine Unbekannte aus der anderen abgeleitet werden kann oder wenn zwei Gleichungen mit denselben Unbekannten vorhanden sind.

Aber zwischen dem jeweiligen Block-Schlüssel = k1 und dem in derselben Runde generierten Chiffre-Alphabet (128) = k2 gibt es keine Verbindung. Beide sind zwar aus der aktuellen CypherMatrix entnommen, haben aber keine funktionale Beziehung: (k1 --> (**Hk MOD 169**)**+1**) und k2 --> (**Hk +Hp**) **MOD 255**)**+1**). Die Runden CypherMatrix selbst ist aus der ursprünglichen Start-Sequenz hergeleitet. Dahin führt jedoch kein Weg zurück (zwei Einwegfunktionen stehen dagegen). Insoweit kann auch hier kein erfolgversprechender Weg gefunden werden.

4 Beispiele zum Testen

Im Anhang A sind einige vom Autor entwickelte Programme aufgeführt. Für jedes einzelne Programm kann der Quellcode per e-mail beim Autor angefordert werden (eschnoor@multi-matrix.de).

Um die Sicherheit des Verfahrens einmal selbst zu testen, können Sie sich das Programm **telecypher.net/securita.7z** herunter laden und alles ausprobieren.

Die Klartextdatei *„Beispiel.txt“* wird mit dem Programm **„DataCode.exe“** verschlüsselt. Ein Teil vom Anfang des Klartextes ist bekannt. Außer einem „*ciphertext only*“ Angriff können auch ein *„chosen-plaintext“* Angriff und weitere Analysen vorgenommen werden. Die Startsequenz und der restliche Klartext sind zu suchen. Im Erfolgsfall erbittet der Autor eine entsprechende Nachricht per e-mail (eschnoor@multi-matrix.de).

5 Hinweise

[#1] „CypherMatrix“ Verfahren, www.telecypher.net/CypherMatrix.pdf
[#2] Bauer, F. L., Entzifferte Geheimnisse, Berlin Heidelberg New York 1995, S.186ff.
[#3] Swoboda, Joachim, Spitz St., Pramateftakis M., Kryptographie und IT-Sicherheit, Wiesbaden 2008, S.22

[#4] Schmeh, Klaus, Safer Net, Kryptographie im Internet und Intranet, Heidelberg 1998, S. 61
[#5] Paar, Christof und Pelzl, Jan, Understanding Cryptography, Berlin-Heidelberg, 2010, S.124
[#6] Schneier, Bruce, Angewandte Kryptographie (dt.Ausgabe), Bonn … 1996, S.278
[#7] Schneier B., a.a.O., S.17
[#8] Schneier B., a.a.O., S. 275
[#9] Schneier, B., a.a.O.,, S.177
[#10] Singh, Simon, Geheime Botschaften, München 2004, S.298

Anhang A

Nach den Grundsätzen der „Codiergraphie“ hat der Autor eine Reihe von Programmen entwickelt, die in der folgenden Liste zusammengestellt sind.

System Basis	System Alphabet	Basis-Coding (ohne XOR-Funktion) einfache Matrix	Verbund-Coding (mit XOR-Funktion) einfache Matrix	Längen-verhältnis
1	2	Crypto01	MonoCode	1:8
2	4	Crypto02	ZweiCode	1:4
3	8	Crypto03	DreiCode	1:2,66
4	16	Crypto04	VierCode	1:2
5	32	Crypto05	QuinCode	1:1,6
6	64	Crypto06	CM64Code	1:1,33
7	128	Crypto07	DataCode DynaCryp CodeData[1)] QuadCode[2)]	1:1,143 1:1,143 1:1,143 1:1,143
8	256	Crypto08 CMCode8D System08	PlanCode MyCode08	1:1 1:1 1:1
9	512	Crypto09 System09	NeunCode MyCode09	1:1,79 1:1,79
10	1024	Crypto10 System10	ZehnCode MyCode10	1:1,6 1:1,6
11	2048	Crypt11A Crypt11B System11	ElvaCode MyCode11	1:1,46 1:1,46
12	4096	Crypto12 System12	MegaCodA MegaCodB MyCode12	1:1,33 1:1,33 1:1,33
13	8192	System 13	MyCode13	1:1,23
14	16384	System14	MyCode14	1:1,143

[1)] Programm mit drei Operationen (XOR – bit conversion - exchange),
[2)] Programm mit vier Operationen (dyn24 – XOR – bit conversion – exchange).

Der Quell-Code der einzelnen Programme kann beim Autor per e-mail angefordert werden (eschnoor@multi-matrix.de).