Alternative Verfahren der digitalen Kryptographie

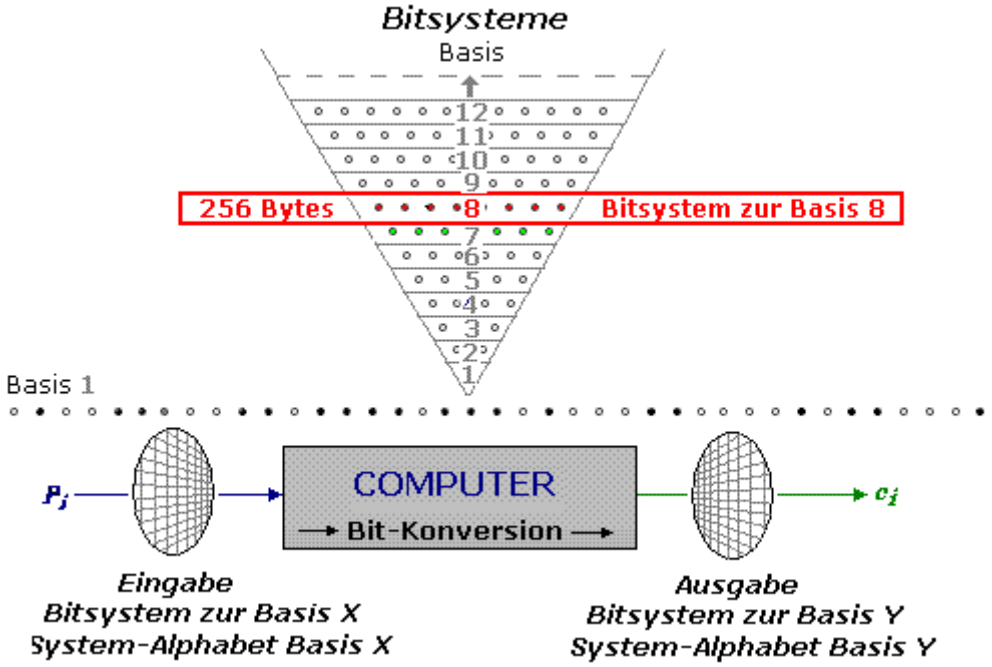
(Ernst Erich Schnoor)

In der klassischen Kryptographie war Gegenstand der Verschlüsselung das einzelne Zeichen. Mit Einführung der Computer haben sich die Verfahren der Verschlüsselung auf die digitale Technik verlagert. Aber Computer können grundsätzlich nur zwei Zustände unterscheiden: „vorhanden“ (eine Spannung) oder „nicht vorhanden“ (keine Spannung), in Ziffern des Zahlensystems zur Basis 2: „eins“ oder „null“. Dieser Vorgang wird bekanntlich als „**Bit**“ definiert.

Eine Information ist mehrschichtig. Sie muss aus mindestens zwei Bit bestehen, allgemein aus einer Folge von Bits. Die Bits sind in ihrer Menge unbegrenzt (1 bis ∞). Um mit Bitfolgen systematisch zu arbeiten, müssen sie systematisiert, d.h. skaliert und in feste Abschnitte (Units) geteilt werden. Es liegt ein vergleichbares Phänomen vor, wie bei der Menge aller Zahlen. Wie in der Zahlentheorie lassen sich auch die Bitfolgen in einem Stellenwertsystem ordnen. Für 8-bit Folgen kann dann beispielsweise vom „**Bitsystem** zur **Basis 8** gesprochen werden. Auszugsweise:

				System-Alphabet
Bitfolgen	1-bit	= Bitsystem zur Basis 1	= 2^1 Zeichen =	2 Units
	3-bit	= Bitsystem zur Basis 3	= 2^3 Zeichen =	8 Units
	6-bit	= Bitsystem zur Basis 6	= 2^6 Zeichen =	64 Units
	7-bit	= Bitsystem zur Basis 7	= 2^7 Zeichen =	128 Units
	8-bit	**= Bitsystem zur Basis 8**	**= 2^8 Bytes =**	**256 Bytes**
	11-bit	= Bitsystem zur Basis 11	= 2^11 Zeichen =	2048 Units
	13-bit	= Bitsystem zur Basis 13	= 2^13 Zeichen =	8192 Units
	16-bit	= Bitsystem zur Basis 16	= 2^16 Zeichen =	65536 Units

Die Zusammenhänge sind im „whitepaper“ www.telecypher.net/ParadigmaDe.pdf [#1] ausführlich erläutert. Der Aufbau des Stellenwertsystems wird am besten mit der kopfstehenden System-Pyramide dargestellt.



1 Systembasis der aktuellen Verfahren

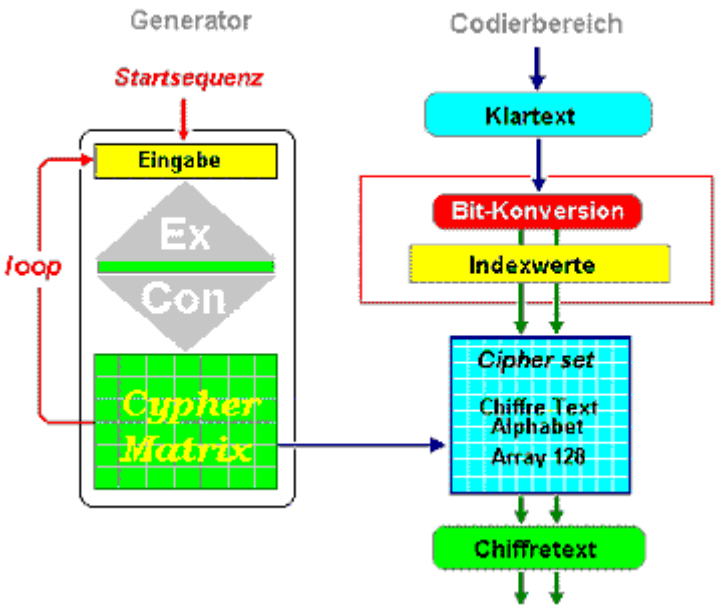
In der aktuellen Kryptographie vollziehen sich fast alle Operationen im Bitsystem zur **Basis 8.** Sowohl Eingaben werden im Bitsystem zur Basis 8 mit dem System-Alphabet von 256 Zeichen (00 bis FF) verarbeitet als auch verschlüsselte Ergebnisse (Chiffretext) im Bitsystem zur Basis 8 und System-Alphabet mit 256 Zeichen ausgegeben. Dass zwischen Eingabe und Ausgabe vielfach Bitfolgen anderer Länge manipuliert werden (Codierfunktion), ist für das Ordnungsprinzip nicht entscheidend. Insoweit vollziehen sich alle Verschlüsselungsoperationen – unabhängig von der Anzahl der zwischenzeitlich manipulierten Bits – in einem einheitlichen **Ordnungsystem**, im Bitsystem zur **Basis 8**.

Mit diesen Zusammenhängen und der Forderung, Klartext und Chiffretext müssten gleich lang sein [#2,#9], wird der **Aktionsraum** der aktuellen Kryptographie auf den Bereich des Bitsystems zur **Basis 8** mit dem System-Alphabet (Chiffre-Alphabet) von **256 Bytes** eingeengt. Diese Enge kann mit einer neuen Alternative überwunden werden – dem CypherMatrix Verfahren (Bezeichnung vom Autor) [#3].

2 Alternative Verfahren

Im CypherMatrix Verfahren – erläutert im Artikel: "Grundlagen des CypherMatrix Verfahrens" - ist die Verschlüsselung einer Information relativ einfach:

> Ein **Generator** erzeugt das erforderliche **System-Alphabet** und
> mit der **Bit-Konversion** im **Codierbereich** wird der Chiffretext geschrieben.



2.1 “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 könnte mit dem Computer gar nicht gearbeitet werden.

2.2 „Bit-Konversion“

Bit-Konversion 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. Eine Bit-Konversion kann für alle Bitsysteme von zur Basis 1 bis zur Basis 14 (und höher) durchgeführt werden.

3 Umwandlung der Bitsysteme

Eine Umwandlung setzt voraus, dass die Länge der umzuwandelnden Bitfolge sowohl durch die Anzahl der bisherigen Einheiten als auch durch die Anzahl der angestrebten Einheiten teilbar ist, da sonst einzelne Bits verloren gehen. Die Umwandlung von Basis 8 nach Basis 7 beispielsweise vollzieht sich wie folgt:

Bisherige Einheiten:
01001110 01101111 01110100 01100001 01100010 01100101 01101110 = 7 Einheiten
Neue Einheiten:
0100111 0011011 1101110 1000110 0001011 0001001 1001010 1101110 = 8 Einheiten

Die Länge der Bitfolge mit 56 Bits ist sowohl durch 7 als auch durch 8 teilbar. Infolge der Umwandlung verlängert sich der Chiffretext im Verhältnis zum Klartext um den Faktor **8/7**, d.h. aus einem Klartextzeichen entstehen **1,143** Chiffretextzeichen. Chiffretext und Klartext sind nicht mehr gleich lang.

3.1.1 Bit-Konversion von Basis 1 nach Basis 8

Historisch gesehen wurde dieser Vorgang bereits 1963 am Anfang der digitalen Bearbeitung durchgeführt und eine 7-Bit-Zeichencodierung ASCII als Standard-Code für Schriftzeichen entwickelt [Wikipedia]. Als das 128 Zeichen umfassende Alphabet zu eng wurde, kam der 8-Bit-Code mit dem erweiterten ASCII-Zeichensatz von 256 Elementen, der noch heute weitgehend als Standard verwendet wird.

Bei genauer Betrachtung stellt dieser Vorgang bereits eine **Verschlüsselung** dar. Sie wird allerdings kaum als solche wahrgenommen, sondern eher als Grundlage der digitalen Kommunikation (Standard Code) gesehen. Im Einzelnen geschieht die Bit-Konversion von Basis 1 nach Basis 8 wie folgt:

Bitfolge Basis 1:
01100010011010010111010001100110011011110110110001100111011001010110111 0
Bitfolge Basis 8:
01100010 01101001 01110100 01100110 01101111 01101100 01100111 01100101 . . . . .
Index: 98 105 116 102 111 108 103 101
System-Alphabet (ASCII-Zeichensatz):
b i t f o l g e

System-Alphabet (ASCII): dezimal hexadezimal

Alphabet$(98)	=	b	62
Alphabet$(101)	=	e	65
Alphabet$(102)	=	f	66
Alphabet$(103)	=	g	67
Alphabet$(105)	=	i	69
Alphabet$(108)	=	l	6C
Alphabet$(111)	=	o	6F
Alphabet$(116)	=	t	74

Index (hexadezimal) = System-Alphabet (hexadezimal):

62 69 74 66 6F 6C 67 65

In der Klartextausgabe lautet die Bitfolge: „**bitfolge**“
Die hexadezimale Ausgabe lautet: „**626974666F6C6765**“

Die zugrunde liegende Bitfolge im Bitsystem zur Basis 1 bleibt unverändert (kein Bit wird hinzugefügt und kein Bit wird weggelassen). Das Beispiel verdeutlicht, das nur das System-Alphabet gewechselt werden muss, während die nicht strukturierte Bitfolge (Anzahl und Reihenfolge) bei jeder Konversion grundsätzlich gleich bleibt.

3.1.2 Variation des System-Alphabets

Mit dem **Generator** und der Startsequenz „**Leonardo erobert Florenz mit Schneekanonen**“ wird ein permutiertes System-Alphabet erzeugt, das zu folgender Variation der Basis-Verschlüsselung führt:

1	□#–¾—#Žñ□#Pä#ž#°□□º@ÎNB#¥ó#˜™²½ë„òc+úÒ¼ùàfS1\,H2Üöˆsaå6ká9Ýôêûõ3	64
65	R#û…šœl>ü€Þ³-T□Êxhtvì.´XâY#Æ!j**Z**Ÿ **¤÷¡**D**y**z#**§**¢{**£**|¨ý}**©**]#†#‡‰ŠÐ#7À‹^eÏ	128
129	gª_b«mÔ4¬8þ";#æãn<Œµ®Aç¿¯ÇÕl#±·¸¹»ÁÂÖ####ºò#ôþãa®}#Îl V~Æx#™\ï¢	192
193	æÀk¡…ˆÊ÷h#yõ£Bp#EqË9#ÃHlÒ<Á#U*ƒ*cR·¤É„u#!YS†ý#J#ÂÚP-¨W×€å—?`CÄçdøÞ	256

Neues System-Alphabet :

Alphabet$(98)	=	Z
Alphabet$(101)	=	¤
Alphabet$(102)	=	÷
Alphabet$(103)	=	¡
Alphabet$(105)	=	y
Alphabet$(108)	=	§
Alphabet$(111)	=	£
Alphabet$(116)	=	©

Der Klartext „Bitfolge“ ergibt verschlüsselt den Chiffretext: „**Zy©÷£§¡¤ „**

3.2 Bit-Konversion von Basis 8 nach Basis 13

Der Umwandlung liegt das Programm **System13.exe** zugrunde. Bei der Bit-Konversion vom Bitsystem zur Basis 8 zum Bitsystem zur Basis 13 bleibt die Anzahl der Bits und ihre Reihenfolge gleich. Es wird kein Bit weggelassen und kein Bit hinzugefügt. Nur die Abstände der Einheiten ändern sich. Aus 8-bit Sequenzen werden 13-bit Abschnitte. Die dezimalen Werte der neuen Einheiten sind die Indexwerte für die Zeichen im System-Alphabet zur Basis 13. Dafür sind im vorliegenden Fall 2^13 Zeichen = 8192 Zeichen erforderlich. Da aber für diesen Umfang keine Einzelzeichen mehr zur Verfügung stehen, werden die Ziffern des **Zahlensystems** zur **Basis 128** – das sind 16384 Ziffern – als **Doppelzeichen** verwendet. Das System-Alphabet umfasst einen permutierten Abschnitt ab Kappa+1, d.h. von der Ziffer 6601 bis 14793 = 8192 Doppelzeichen.

Bitfolge Basis 8:
01100010 01101001 01110100 01100110 01101111 01101100 01100111 01100101 . . . . .
Index: 98 105 116 102 111 108 103 101
Bitfolge Basis 13:
3011000100110 1001011101000 1100110110111 1011011000110 0111011001010 . . . . .
Index: 3149 1489 4919 5830 3786
Basis 13 ëL #P …¬ ¥E {I

Das System-Alphabet (Basis 13):

Alphabet$(3149) = ëL
Alphabet$(1489) = #P
Alphabet$(4919) = …¬
Alphabet$(5830) = ¥E
Alphabet$(3786) = {I

Im Bitsystem zur Basis 13 wird „bitfolge“ wie folgt umgewandelt: **ëL#P…¬¥E{I**

3.3 Bit-Konversion von Basis 8 nach Basis 7

Die Bit-Konversion von Basis 8 nach Basis 7 ist der haupt Anwendungsfall für Verschlüsselungen. Im Folgenden wird die Umwandlung am Beispiel des Programms **Crypto07.exe** erläutert.
Bitfolge Basis 8:
01100010 01101001 01110100 01100110 01101111 01101100 01100111 01100101 . . . . .
Index: 98 105 116 102 111 108 103 101
System-Alphabet (ASCII):
b i t f o l g e
Bitfolge Basis 7:
0110001 0011010 0101110 1000110 0110011 0111101 1011000 1100111 0110010 . . . . .
Index:49 26 46 70 51 61 88 103 50
(+1) 50 27 47 71 52 62 89 104 51
Chiffre-Alphabet:
e ¡ À ® g þ E & Ï

„bitfolge“ wird im Bitsystem zur Basis 7 wie folgt dargestellt: **e¡À®gþE&Ï**

Das System-Alphabet zur Basis 7 mit 128 Units wurde mit dem Generator und der Start-sequenz: „Leonardo erobert Florenz mit Schneekanonen“ erzeugt. Die Indexwerte sind um (+1) erhöht, da das System-Array den Index „0“ nicht erkennt.

Chiffre-Alphabet (Basis 7)

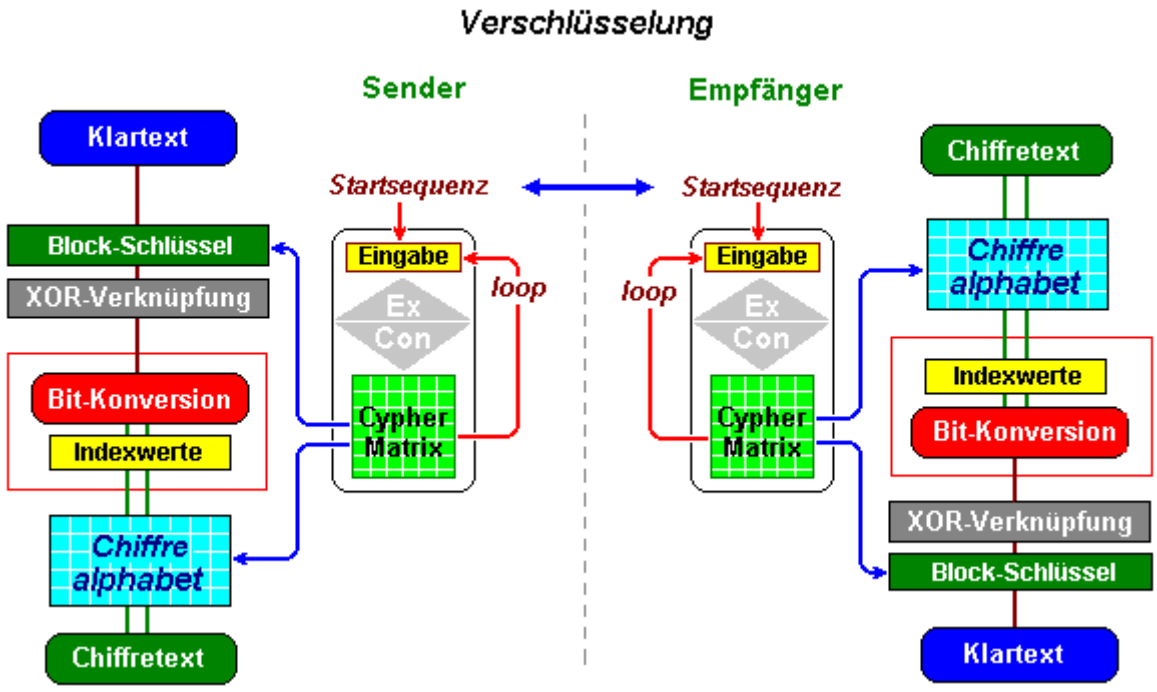
1	> ü € ³ – T ¥ Ê x h t V ì . ´ X â Y Æ ! j Z Ÿ ¤ ÷ ¡ D y z § ¢	*32*
33	{ £ | ¨ ý } ©] † ‡ ‰ Š Đ 7 À ‹ ^ e Ï g ª _ b « m Ô 4 ¬ 8 þ ; æ	*64*
65	ã n < Œ µ ® A ç ¿ ¯ Ç I · ¸ ¹ » Á Â Ö # $ ' C E J Q o í O È ?	*96*
97	K å è Ë :) □ & ƒ É % , = Ì U Ú Ñ u / › î ` p [Ã ¦ w ï ø ’ ì Ä	*128*

Chiffre-Alphabet (hex)

1	3E	FC	80	B3	2D	54	9D	CA	78	68	74	56	EC	2E	B4	58	16
17	E2	59	C6	21	6A	5A	9F	A0	A4	F7	A1	44	79	7A	A7	A2	32
33	7B	A3	7C	A8	FD	7D	A9	5D	86	87	89	8A	D0	37	C0	8B	48
49	5E	65	CF	67	AA	5F	62	AB	6D	D4	34	AC	38	FE	3B	E6	64
65	E3	6E	3C	8C	B5	AD	AE	41	E7	BF	AF	C7	49	B7	B8	B9	80
81	BB	C1	C2	D6	23	24	27	43	45	4A	51	6F	ED	4F	C8	3F	96
97	4B	E5	E8	CB	3A	29	7F	26	83	C9	25	82	3D	CC	55	DA	112
113	D1	75	2F	9B	EE	60	70	5B	C3	A6	77	EF	F8	92	8D	C4	128

4 Verschüsselungen

Die Verschlüsselung – das Schreiben und Lesen von geheimen Informationen – findet ausschließlich im Codierbereich statt. Mit Eingabe der gleichen Startsequenz, sowohl beim Sender als auch beim Empfänger, werden im gesamten Verfahren ein identischer Verlauf und identische Steuerungsparameter erzeugt. Das folgende Schema zeigt die Zusammenhänge:



Die Verschlüsselung wird in folgenden Alternativen durchgeführt:

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

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

Nach den vorstehenden Grundsätzen 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).

Die Programme können einzeln oder in Gruppen mit oder ohne Quellcode beim Autor per e-mail angefordert und im Rahmen der **CMLizenz** getestet oder weiter entwickelt werden. Alle Programme sind DOS-Programme und laufen nur noch unter Windows XP. Sie müssen auf **C#** umgeschrieben werden, wie dies bereits unter Leitung von Prof. **Bernhard Esslinger** (Uni Siegen) und seinem CrypTool-Team (insbesondere: **Michael Schäfer**) mit den Programmen **DataCode** und **DynaCode** geschehen ist [#4]. Alle weiteren Programme müssen noch umgeschrieben werden. Eine Zusammenstellung der Programme kann aus dem Internet unter: www.telecypher.net/CypherMatrix.exe herunter geladen werden.

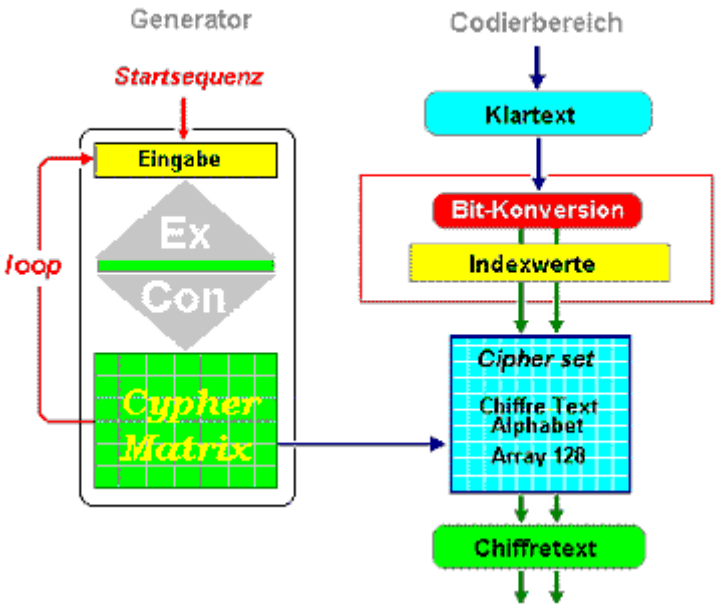
4.1 Basis-Coding

Im „Basis-Coding“ Modus erfolgt die Bit-Konversion direkt vom bisherigen Bitsystem zum angestrebten Bitsystem. Zum Beispiel von Basis 8 nach Basis 7:

Klartext	N	o	t	a	b	e	n	e	
	01001110	01101111	01110100	01100001	01100010	01100101	01101110	01100101	
	0100111	0011011	1101110	1000110	0001011	0001001	1001010	1101110	0110010	1
Indexe	39	27	110	70	11	9	74	110	50	

Die Umwandlung vom Klartext zum codierten Text geschieht mit zwei Funktionen:

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



Als Beispiel für **Basis-Coding** wird ein Gedicht von Matthias Claudius verschlüsselt mit dem Programm: **System12.exe** und der Start-Sequenz: „Die weiße Elster fließt in das schwarze Meer“ (Datei: **Claudius.txt** / Umfang: 215 Bytes).

Der Mond ist aufgegangen,
Die goldnen Sternlein prangen
Am Himmel hell und klar;
Der Wald steht schwarz und schweiget,
und aus den Wiesen steiget
Der weiße Nebel wunderbar.

Matthias Claudius, Reinfeld 1812

Als ersten Block des zu verschlüsselnden Klartextes werden 36 Bytes eingegeben:

Der Mond ist aufgegangen,♪¤Die goldn

44 65 72 20 4D 6F 6E 64 20 69 73 74 20 61 75 66 67 65 67 61 6E 67 65 6E 2C
0D 0A 44 69 65 20 67 6F 6C 64 6E

Der Klartext im Bitsystem zur **Basis 8** (36x8=288) wird in Abschnitte des angestrebten Bitsystems zur **Basis 12** (288:12=24) umgewandelt.

Basis 8: D e r M o n d
01000100 01100101 01110010 00100000 01001101 01101111 01101110 01100100 . . . .
Basis 12:
010001000110 010101110010 001000000100 110101101111 011011100110 0100 . . . .
Index: 1094 1394 516 3439 1766
(+1) 1095 1395 517 3440 1767
System-Alphabet Basis 12:

Alphabet$(517) = ‰s
Alphabet$(1095) = ì¤
Alphabet$(1395) = Éa
Alphabet$(1767) = “O
Alphabet$(3440) = X

Chiffretext: ì¤ Éa ‰s X “O

Der Chiffretext lautet wie folgt:

ì¤Éa‰s X“Oì|’èŒc‰uˆd’O”N’eˆW’e‰WŠžŸyì¤˜N‰u”X’¤ŽWx–êqvêt¥zwë0xá2pRm6–
xgê§x–êXmuo1pO#ªy–è¥y,l©xç0äWë7ì‡á4í‹ãBçè‡Zè‹ëBäUî¦í‹èvîïè«íWèvîîç1ïFãBïx–
áEVnP—W1OpWàSqUàS#NBh�WXklL{K@VªlIUXj—SàamW1SvLO#ejk&ekZ�S§a|
yf¥SíejluefSbjWyfPSígjuxekWvdšW5Tdl�Gkf—
T{NžSWHªJERXPšSXPªKëFRXLPšSXPªKëFˆRŒX¤RœP¢RëFxMDHyG†Zg

```
8D A4 90 61 89 73 A0 58 93 4F 8D 7C 92 E8 8C 63 89 75 88 64 92 4F
94 4E 92 65 88 57 92 65 90 57 8A 9E 9F 79 8D A4 98 4E 89 75 94 58
92 A4 8E 57 78 96 EA 71 76 EA 74 A5 7A 77 EB 30 78 96 E1 32 70 52
6D 36 78 67 EA A7 78 96 EA 58 6D 75 6F 31 70 4F 23 AA 79 96 E8 A5
79 82 6C A9 78 96 E7 30 E4 57 EB 37 EC 87 E1 34 ED 8B E3 42 E7 E8
87 5A E8 8B EB 42 E4 55 EE A6 ED 8B E8 76 EE EF E8 AB ED 57 E8 76
EE EE E7 31 EF 46 E3 42 EF 78 E1 45 56 6E 50 97 57 31 4F 70 57 E0
53 71 55 E0 53 23 4E 42 68 81 57 58 6B 6C 4C 7B 4B 40 56 AA 49 6C
55 58 6A 97 53 E0 61 6D 57 31 53 76 4C 7C 4F 23 65 6A 6B 26 65 6B
```

```
5A 90 53 A7 61 79 66 A5 53 ED 65 6A 6C 75 65 66 53 62 65 6A 57 79
66 50 53 ED 67 6A 75 78 65 6B 57 76 64 9A 57 35 54 64 6C 90 47 6B
66 97 54 7B 4E 9E 53 57 48 AA 4A 45 4C A2 52 58 50 9A 53 58 50 AA
4B EB 46 88 52 8C 58 A4 52 9C 50 A2 52 EB 46 78 4D 44 48 79 47 86
5A 67
```

Infolge der Bit-Konversion entfallen auf 12 Zeichen im Bitsystem zur Basis 8 insgesamt 96 Bits, die dann auf 8 Zeichen im Bitsystem zur Basis 12 verteilt werden. Insoweit ergibt sich wegen der Doppelzeichen ein Längenverhältnis von 1:1,333 .

Für das System-Alphabet zur Basis 12 werden 4096 Zeichen benötigt, die als Einzelzeichen nicht mehr zur Verfügung stehen. Sie werden als Doppelzeichen mit Hilfe der Ziffern des **Zahlensystems** zur **Basis 128** – das sind 16384 Ziffern - generiert:

Im Quellcode:

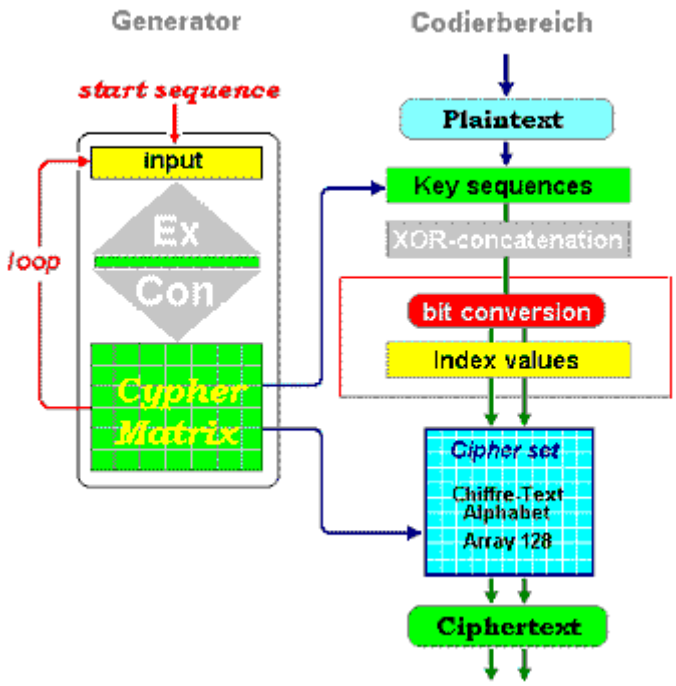
```
SUB Alphabet
      SHARED Alphabet$(),  Kappa
       Kappa = 11441, 6035, 8344, 2250  (in jeder Runde neu berechnet)

       FOR C=1 TO 4096
         X## = C + Kappa
         CALL DezNachSystem (128, X##, Zeichen$)
         Digit$ = „00“+Zeichen$
         Digit$ = RIGHT$(Digit$,2)
         Alphabet$(C) = Digit$
      NEXT C
END SUB
```

4. 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.



Im Quellcode werden die Module nacheinander durchlaufen:

```
. . . . . .
CALL  XORGenerator (InputData$,TransData$)          für „XOR“
CALL BitConversion (TransData$,DataTrans$)          für „Bit-Konversion“
CALL Substitution (DataTrans$,ReportData$)          für „dyn24“
CALL ByteWechsel (ReportData$,OutputData$)          für „exchange“
. . . . . .
```

4.2.1 Verschlüsselung von Basis 8 nach Basis 7

Als Beispiel werden die Verschlüsselungsschritte im Programm **DataCode.exe** und der Startsequenz: „**Im Elbsandsteingebirge gibt es keinen Sand**“ für die Datei **„Garten.txt“** erläutert (573 Bytes) .

Ein Steingarten ist eine Zierde für jeden Garten. Schon allein die Steine,
aus denen sich eine solche Anlage aufbaut -ob es sich nun um Kalksteine,
Sandsteine, Schiefer, Granit, Basalt, Tuffsteine oder auch nur um große
Kieselsteine handelt - verleihen dem Garten einen besonderen Akzent.
Man kann sie im rohen Zustand verwenden, gerade so, wie man sie vielleicht
auf einer Wanderung durch bergige oder gebirgige Gegenden findet, doch eine
ebenso große Wirkung kann man natürlich mit behauenen Steinen erzielen.

Steingartenpflanzen, ADAC-Ratgeber, Stuttgart 1961

In jeder Runde wird ein Klartext-Block von 42 Bytes (42x8 = 336 Bits) mit einem Block-Schlüssel von gleicher Länge XOR-Funktion verknüpft. Als ersten Klartext-Block liest das Programm folgende 42 Bytes ein:

Ein Steingarten ist eine Zierde für jeden□ (42 Bytes)

45 69 6E 20 53 74 65 69 6E 67 61 72 74 65 6E 20 69 73 74 20 65
69 6E 65 20 5A 69 65 72 64 65 20 66 81 72 20 6A 65 64 65 6E 20

Der ab Position **158** der CypherMatrix entnommene Block-Schlüssel umfasst:

yu5ˆÉDW"È¹´GÐ‡Á½H,(#„Ìsí^Ëº¿ö-q<ÊÅ#ÂðþØ¥¤+

79 75 35 88 C9 44 57 22 C8 B9 B4 47 D0 87 C1 BD 48 2C 28 03 84
CC 73 ED 5E CB BA BF F6 2D 71 3C CA 8F 15 C2 F0 FE D8 A5 A4 2B

XOR-Verknüpfung:
Klartext:
01000101 01101001 01101110 00100000 01010011 01110100 01100101 01101001 . . . .
Schlüssel:
01111001 01110101 00110101 10001000 11001001 01000100 01010111 00100010 . . . .
XOR:
00111100 00011100 01011011 10101000 10011010 00110000 00110010 01001011 . . . .

Hex.	3C	1C	5B	A8	9A	30	32	4B
Dez:	60	28	91	168	154	48	50	75
ASCII:	<	#	[	¨	š	0	2	K

<#[¨š02K¦ÞÕ5¤â¯¥!_\#á¥#ˆ~‘ÓÚ„I##¬#gâš›¼ÀÊ&#=

3C 1C 5B A8 9A 30 32 4B A6 DE D5 35 A4 E2 AF 9D 21 5F 5C 23
E1 A5 1D 88 7E 91 D3 DA 84 49 14 1C AC 0E 67 E2 9A 9B BC C0

Als Ergebnis entsteht ein partielles **„one-time-pad“**. Klartext und Schlüssel sind gleich lang und der Schlüssel wird auch nicht wiederholt. In jeder Runde wird ein anderer Schlüssel aus der betreffenden CypherMatrix entnommen.

4.2.2 Bit-Konversion

Das Ergebnis der XOR-Verknüpfung im Bitsystem zur Basis 8 (42x8 = 336 Bits) wird in Zeichen des Bitsystems zur Basis 7 (336 Bits = 48x7) umgewandelt. Die dezimalen Werte der umgewandelten Zeichen (Basis 7) sind Indexwerte für die Positionen der Zeichen im Chiffre-Alphabet. Die Indizes für das Chiffre-Alphabet müssen um +1 erhöht werden, da der Index „**0**“ im Array des Chiffre-Alphabets nicht erkannt wird.

Bit-Konversion:
XOR Basis 8:
00111100 00011100 01011011 10101000 10011010 00110000 00110010 01001011
Basis 7:
0011110 0000111 0001011 0111010 1000100 1101000 1100000 0110010 0100101 1

Index:	30	7	11	58	68	104	96	50	37
(+1)	31	8	12	59	69	105	97	51	38

Alphabet Basis 7:

k	T	?	q	+	•	Š	s	W

System-Alphabet zur Basis 7

```
  1  ƒ™êš¢U¦T}nñ?¡‰h9˜Q>MLié8K7Ï€èYky  32
 33  u5ˆÉDWÈ¹´GÐ‡Á½H(„Ìsí^Ëº¿ö-q<Ê¥Âð  64
 65  þØ¥¤+[:ã#{·CåS/ª‘Bx@ä3À§)0«deV!Ÿ   96
 97  Š‹Œıℓ℩Ïï¨c³&• ž¯£—l›®.’□%$ýÄ\OÔ]ûÆü  128
```

System-Alphabet (hex)

```
83 99 EA 9A A2 55 A6 54 7D 6E F1 3F A1 89 68 39
98 51 3E 4D 4C 69 E9 38 4B 37 CF 80 E8 59 6B 79
75 35 88 C9 44 57 C8 B9 B4 47 D0 87 C1 BD 48 28
84 CC 73 ED 5E CB BA BF F6 2D 71 3C CA 8F C2 F0
FE D8 A5 A4 2B 5B 3A E3 23 7B B7 43 E5 53 2F AA
91 42 78 40 E4 33 C0 A7 29 30 AB 64 65 56 21 9F
8A 8B 8C 8D A8 63 B3 26 95 A0 9E AF A3 97 6C 9B
AE 2E 92 AD 7F 25 24 FD C4 5C 4F D4 5D FB C6 FC
```

Als Ergebnis der Bit-Konversion entsteht der Chiffretext der Datei **Garten.ctx**
(672 Zeichen)

kT?q+•ŠsW d—GÀ{Œ§&DiO®ã‹xãÌTÿã¹«¥>ˆØc„è&.È@<³šLèfÅRó~÷([GI½hO‡@~?R
E`((sMGfÉw`¢¤âúþzÈÅv9ú2[úíWn½~ýe~««�å”½YY¼ýÀþK”gO¾À“<¶æÏ¼•-”ÌI}r½«öC-
A“˜8¯='Gd'ý¿‰MD·Cm`.dš§záÓÖ(¾Sæ¢’óó+F¦¿¹gôæ:F \óž¾zBéÈÓ7<îÏfïwîOnÉ½E×Æ
÷›Ïûî¼Ð´<”ÏÅ¸�Ï³g6@5®HV N”D…ªb=GÎc·yÀSÆÃK1KO6tî¤—«†À}¼1Ép6†ÏÃáŸTàîLêI
–©yT&“&0<ýHî“ö†«œÀ-Åó¬«4¿(ñmHôní™ŠòôYŽ12÷qslqm¾jØÃôr–Gii¬sÀaÈP'—ÔËP
—ê}Ç]Ôîzµ¢�úoA‡ãV]¸m$Ë©*üêu<cK’Ì¼ù§hœÈVSÎB)ÒZÒîežú«GM+BsVL&?úŒd”«å!
Ú•Ãe¤ÈâB{û³®ãæ�ú¦eª©z+Çµ¢º¬XHjwƒ¶-Z*5k§“¶Kô5™F»áQ\2Ì¦†À†7´E™ÍOûXl-´EE
™.ÍOûXl-.GF\v1=´Ò*ÊEMðì¯÷æ>h>XECãŠ†ì2©JÃê(ìHæ‰*º´+%Á´ácZ… <ƒMNüœº`
ÓbY…kÀ¤kxc|‡|þŽ‹@1dø)§/àÀõÓ¿zAn}^™„s zÉH¤=þë€*Ø7áHáÇ¼m¼KAK¡%g’ðÚ !
7áHáÇ¼m¼KAK¡%g’ðÚ–’ðÚÇ@Têú´§ëwgP›�`džsÈ�aö¡³gc�ú.“>£ZŽ[m‰_GZÔfî
‰_GZÔf5ï6ÌàòÂM;ï□ü»7õê(þWÇîÀ□à6ÂØø
```

Chiffretext (hex):

```
6B 54 3F 71 2B 95 8A 73 57 A0 64 97 47 C0 7B 8C A7 26 44 69 4F AE
E3 8B 78 E3 CC 54 7F E3 B9 AB A5 3E 88 D8 63 84 E8 26 2E C8 40 3C
B3 9A 4C 3F E8 66 C5 52 F3 7E F7 28 5B 47 49 BD 68 4F 87 40 7E 52
C9 60 28 28 73 4D 47 66 C9 77 60 A2 A4 E2 FA FE 7A C8 C5 76 39 FA
32 5B FA ED 57 6E BD 7E FD 65 7E AB 9D E5 94 BD 59 59 BC FD C0 FE
4B 94 67 4F BE C0 93 3C B6 E6 CF BC D7 95 2D 94 CC 49 7D 72 BD AB
F6 43 AD 41 91 98 38 AF 3D 27 47 64 27 FD BF 89 4D 44 B7 43 6D 60
2E 64 9A A7 7A E1 D3 D6 28 BE 53 E6 A2 92 F3 F3 2B 46 A6 BF B9 67
F4 E6 3A 46 A0 5C F3 9E BE 7A 42 E9 C8 D3 37 3C EE CF 66 EF 77 EE
4F 6E C9 BD 45 D7 C6 F7 9B CF FB EE BC D0 B4 3C 94 CF C5 B8 9D CF
B3 67 36 40 35 AE 48 56 A0 4E 94 44 85 AA 62 3D 47 CE 63 B7 79 C0
53 C6 C3 4B 31 4B 0D 0A 4F 36 74 EE A4 97 FE AB 86 C0 7D BC 31 90
70 36 86 CF C3 E1 9F 54 E0 EE 4C EA 49 FD 96 A9 79 54 26 93 26 30
3C FD 48 EE 93 F6 86 AB 9C C0 2D C5 F3 AC AB 34 BF 28 F1 6D 48 F4
6E ED 99 8A F2 F4 59 8E 31 32 F7 71 73 6C 71 6D BE 6A D8 C3 F4 72
96 47 69 69 AC 73 C0 61 C8 50 27 97 D4 CB 50 97 EA 7D C7 5D D4 EE
7A B5 A2 8F FA 6F 41 87 E3 56 5D B8 6D 24 CB A9 2A FC EA 75 3C 63
4B 92 CC BC F9 A7 68 9C C8 56 53 CE 42 29 D2 5A D2 EE 65 9E FA AB
```
```

```
47 4D 2B 42 73 56 4C 26 3F FA 8C 64 94 AB E5 21 DA 95 C3 65 A4 C8
E2 42 7B FB B3 AE E3 E6 8D FA A6 65 AA A9 7A 2B C7 B5 A2 BA AC 58
48 6A 77 83 B6 2D 5A 2A 35 6B A7 93 B6 4B F4 35 99 46 BB E1 51 5C
32 CC A6 86 C0 86 37 9E B4 45 99 CD 4F FB 58 6C 2D 2E 47 46 5C 76
31 3D B4 D2 2A CA 45 4D F0 EC AF F7 E6 3E 0D 0A 68 3E 58 45 43 E3
8A 86 EC 32 A9 4A C3 EA 28 8D 48 E6 89 2A BA B4 2B 25 C1 B4 E1 63
5A 85 A0 3C 83 4D 4E FC 9C BA 60 D3 62 59 85 6B C0 A4 6B 78 63 7C
87 7C FE 8E 8B 40 31 64 F8 29 A7 2F E0 C0 F5 D3 BF 7A 41 6E 7D 5E
99 84 73 A0 7A C9 48 A4 3D FE EB 80 2A D8 21 37 E1 48 E1 C7 BC 6D
96 BC 4B 41 4B A1 25 67 92 F0 DA C7 40 54 EA FA B4 A7 EB 77 67 50
9B 81 60 64 9E 73 C8 90 61 F6 A1 B3 67 63 90 FA 2E 93 3E AD A3 5A
8E 5B 6D EE 89 5F 47 5A D4 66 35 EF 36 CC E0 F2 C2 4D 3B EF 7F FC
BB 37 F5 EA 28 FE 57 C7 EE C0 7F E0 36 C2 D8 F8
```

5 Entschlüsselung

Für die Entschlüsselung erzeugt der Generator einen inhaltsgleichen Ablauf, wie bei der Verschlüsselung. Die Entschlüsselung wird im Codierbereich abgearbeitet, nur in der umgekehrten Reihenfolge:

1. Analyse des Chiffretextes
   Chiffretext → Chiffre-Alphabet (0...127) → 7 bit Indexwerte
2. Bit-Konversion
   7 bit Indexwerte (0 ...127) → 8-bit XOR-Verknüpfung
3. XOR-Verknüpfung
   8-bit XOR-Verknüpfung → Block-Schlüssel → Klartext-Block

Aus Blöcken von **48** Zeichen Chiffretext sucht das Verfahren im identisch erzeugten Chiffre-Alphabet die dezimalen Index-Werte der einzelnen Zeichen und verbindet deren binäre Zahlen zu einer Bitfolge von 336 Bits. Diese Bitfolge wird wiederum in **42** 8-bit Block-Schlüssel XOR-verknüpft. Als Ergebnis erscheint der ursprüngliche Klartext.

Als Beispiel die Entschlüsselung des Chiffretextes **Garten.ctx:**
Das Programm liest als ersten Chiffretext-Block folgende 48 Zeichen ein:

Alphabet Basis 7:
kT?q+•ŠsW d—GÀ{Œ§&DiO®ã‹xãÌTÿã¹«¥>ˆØc„è&.È@<³šLèfÅRó~÷([GI½hO‡@~?

Index:	31	8	12	59	69	105	97	51	38
(-1)	30	7	11	58	68	104	96	50	37

Basis 7:
0011110 0000111 0001011 0111010 1000100 1101000 1100000 0110010 0100101 1 . . .
Bit-Konversion nach Basis 8:
00111100 00011100 01011011 10101000 10011010 00110000 00110010 01001011 . . . .
01111001 01110101 00110101 10001000 11001001 01000100 01010111 00100010 . . . .
XOR-Verknüpfung mit Schlüssel:
01000101 01101001 01101110 00100000 01010011 01110100 01100101 01101001 . . . .
Index Basis 8:

	69	105	110	32	83	116	101	105
Zeichen:	E	i	n		S	t	e	i

Als Klartext ergibt sich: **Ein Stei**ngarten ist eine Zierde für jeden□ . . . . . .

6 Kryptanalyse

Die Kryptanalyse umfasst bekanntlich alle Versuche, aus dem Chiffretext irgendwelche Rückschlüsse auf den Klartext zu gewinnen. Zu den Auffälligkeiten der Sprache zählen Wiederholungsmuster und Wortkombinationen, Häufigkeitsstrukturen und Bigramme [#5]. Diese Sachverhalte setzen voraus, dass Klartext und Chiffretext zueinander im Verhältnis 1:1 stehen. Zu den bekanntesten Angriffen gehören Strukturanalyse, "known plaintext attack", "chosen plaintext attack" und "brute force attack", eventuell noch "differenzielle" und "lineare" Kryptoanalyse [#6]. Mit diesen Angriffen sollen aus dem Chiffretext statistisch erfassbare Regelmäßigkeiten herausgefiltert werden, die mölicherweise einen Weg zum Klartext aufzeigen. Eine Analyse anhand dieser Merkmale, setzt allerdings voraus, dass Klartext und Chiffretext Strukturen enthalten, die sich auch vergleichen lassen. Daher muss ein einheitliches **Ordnungssystem** bestehen, das sowohl im Klartext als auch im Chiffretext wirksam ist.

6.1 Aktuelle Verfahren

Bei fast allen herkömmlichen Verfahren haben Klartext und Chiffretext die gleiche Länge: **"Längenkongruenz"** [#5]. Daraus folgt, dass für jedes Klartextzeichen auch ein bestimmtes Chiffrezeichen vorhanden sein muss. Beide Bereiche - Eingaben und Ausgaben - arbeiten im selben System-Alphabet (ASCII-Zeichensatz), und damit auch im gleichen Bitsystem zur **Basis 8.** Mithin besteht auch ein einheitliches Ordnungssystem.
Ein Systemwechsel findet nicht statt. Eigenheiten im Klartext müssen sich in irgendeiner Form auch im Chiffretext wiederfinden lassen.

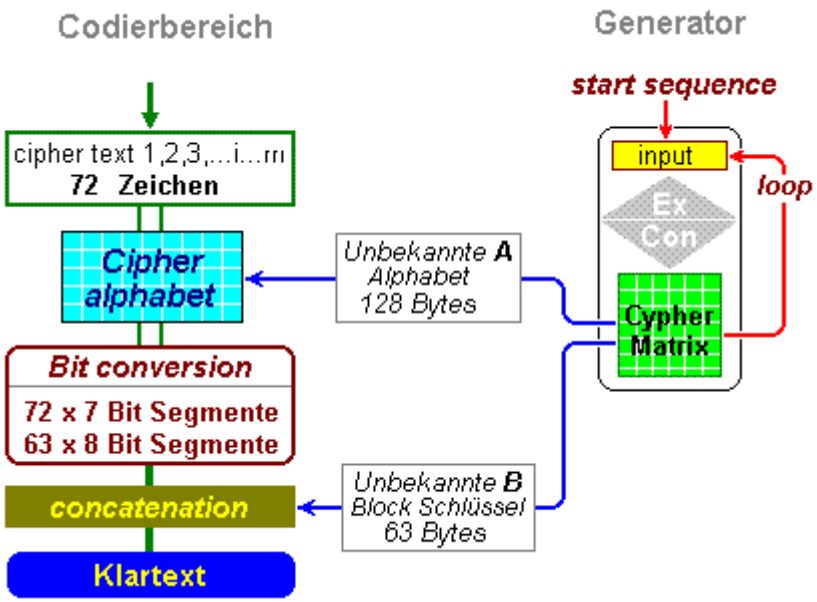
6.2 „CypherMatrix“ Verfahren

Diese Bedingungen sind im CypherMatrix Verfahren nicht erfüllt. Es findet ein Wechsel im Bitsystem statt: Umwandlung von Klartext-Zeichen im Bitsystem zur Basis 8 zu Chiffretext-Zeichen im Bitsystem zur Basis 7 mit der Folge, dass beide Bereiche sich nicht mehr vergleichen lassen. Es fehlt die Längenkongruenz. Durch die Umwandlung der Zeichen vom Bitsystem zur Basis 8 in Zeichen zur Basis 7 entfällt auf jedes Klartextzeichen ein Chiffrezeichen, das um den Faktor **1,143** (8/7) länger ist als das verursachende Klartextzeichen. Außerdem enthält das eigenständige Chiffretext-Alphabet nur 128 Zeichen, während das Klartext-Alphabet 256 Zeichen umfasst. Es fehlt ein einheitliches Ordnungssystem für beide Bereiche und damit auch die Basis für alle oben genannten Angriffszenarien. Sie sind **wirkungslos** und wir können sie vergessen. Um dies zu testen, können Sie sich einige mit CypherMatrix Programmen verschlüsselte Nachrichten als ZIP-Datei auf Ihren Rechner holen und versuchen, die Chiffre mit heute gängigen Angriffen zu brechen [#7].

6.3 Sicherheit des Verfahrens

Immerhin bleibt noch die Möglichkeit einer „brut force“ Attacke. 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. Er könnte also nur eine Iteration aller Möglichkeiten versuchen. Bei einem Angriff auf die Startsequenz mit 42 Bytes Länge ergibt sich eine Entropie von **336** und eine exponentielle Komplexität von **O(2^336) =** 1.4E+101. Ein Angreifer kann dann versuchen vom Chiffretext ausgehend, retrograd durch Iteration die einzelnen Stufen der Entschlüsselung 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).

Es gibt somit viele Paare **Chiffre-Alphabet** / **Block-Schlüssel**, die nach einem versuchten "brute force" Angriff irgendwelche lesbaren Texte liefern, von denen man aber nicht weiß, welcher der Richtige ist: Angriff mit "brute force" . Somit kann auch „brute force“ keinen Erfolg haben.

7 Zusammenfassung

Die vorstehenden Erläuterungen lassen einige alternative Grundsätze erkennen:

1. Ein einheitliches Ordnungssystem (Struktur der Bitfolgen und System-Alphabete) ist bisher nicht definiert,
2. Die Darstellung der Kryptographie beschränkt sich auf Verschlüsselungen im Bitsystem zur Basis 8 (Eingaben und Ausgaben) und
3. Fast alle Angriffszenarien setzen für Klartext und Chiffretext ein vergleichbares einheitliches Ordnungsystem voraus (Bitsystem zur Basis 8 und System-Alphabet mit 256 Zeichen).

Weitere Einzelheiten zum **CypherMatrix** Verfahren unter: www.telecypher.net/ [#8]
Wer sich intensiver mit dem Verfahren beschäftigen möchte, kann einzelne Programme mit oder ohne Source-Code beim Autor per e-mail anfordern und unter der CMLizenz damit arbeiten (eschnoor@multi-matrix.de).

München, im Januar 2013

Hinweise

[#1] Paradigmenwechsel in der Kryptographie, www.telecypher.net/ParadigmaDe.pdf
[#2] Schneier, Bruce, Angewandte Kryptographie (dt. Ausgabe), Bonn … 1996, S.229
[#3] Basisfunktion.pdf, www.telecypher.net/Basisfunktion.pdf
[#4] Esslinger, Bernhard, Uni Siegen, http://www.cryptool.org/de/
[#5] Strukturvergleich Klartext und Geheimtext, www.telecypher.net/Equilang.pdf
[#6] Bauer, Friedrich L., Entzifferte Geheimnisse – Methoden und Maximen der Kryptologie. Berlin Heidelberg New York, 1995, S. 186 ff.,
[#7] Download, www.telecypher.net/ZUSENDEN.HTM
[#8] Der Kern des CypherMatrix Verfahrens, www.telecypher.net/CYPHKERN.HTM
[#9] Schmeh, Klaus, Safer Net, Kryptografie im Internet und Intranet, Heidelberg 1998, S.61,