

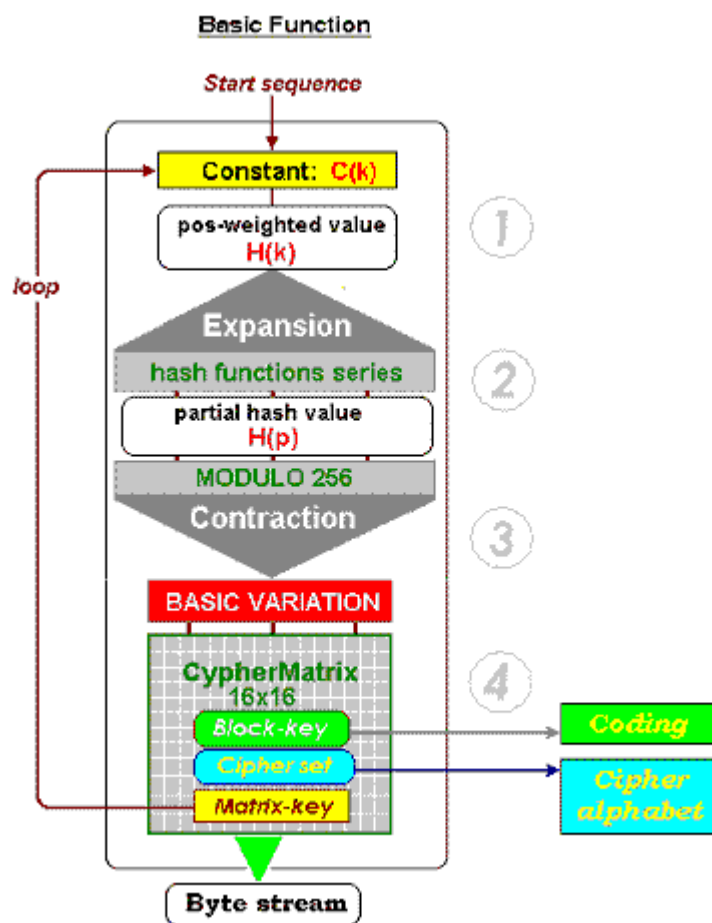
Datenverlauf im CypherMatrix Verfahren

(Ernst Erich Schnoor)

„**CypherMatrix**“ ist ein kryptographisches Werkzeug, das in fast allen Gebieten der Informationstechnik eingesetzt werden kann, sozusagen als „**Codier Maschine**“.

Es liefert alle Steuerungsparameter für kryptographische Anwendungen (z.B. Codier-Schlüssel, Block-Schlüssel, Chiffre-Alphabete, Zufallsfolgen, Hashwerte, Authentifizierungs Merkmale, Signaturen, S-Boxen usw.).

Eine ausführliche Darstellung findet sich im Artikel: [Kryptographische Basisfunktion in Byte-Technik](#). Die folgenden Erläuterungen dienen zur Ergänzung der Darstellung.



Der Datenverlauf wird durch folgende vier Rechenschritte bestimmt:

1. Positionsgewichtung der Startsequenz
2. Bestimmung der Hashfunktionsfolge
3. Verdichtung zur BASIS VARIATION
4. Permutation zur CypherMatrix (Hashwert H)

Die Ergebnisse der einzelnen Rechenschritte werden außer im Dezimalsystem auch in den Zahlensystemen zur Basis **62**, Basis **77** und Basis **78** ausgewiesen.

Die gewählten Zahlensysteme umfassen die folgenden Ziffern:

0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
 mnopqrstuvwxyz&#@àáâãäåæçèéêëìíîïðñ, f, „... †‡^%Š‹
 Basis 62 | Basis 77 | Basis 96 |
 (definiert vom Autor, nicht standardisiert)

Jede Runde (r) der Funktion beginnt mit der Eingabe einer **Startsequenz** von optimal 42 Bytes. Als Beispiel für den Datenverlauf der ersten Eingabe wählen wir die Zeichenfolge:

„**Die Schildbürger fegen den Teutoburgerwald**“ (n=42 Bytes)

44 69 65 20 53 63 68 69 6C 64 62 81 72 67 65 72 20 66 65 67 65
 6E 20 64 65 6E 20 54 65 75 74 6F 62 75 72 67 65 72 77 61 6C 64

1. Positionsgewichtung H(k) der Startsequenz

Jedes Byte **a(i)** wird mit seiner Position in der Startsequenz **p(i)** multipliziert. Um Kollisionen zu vermeiden, wird die Position des Zeichens durch die Konstante **C(k)** in einen Bereich außerhalb der Länge (n) der Startsequenz erweitert:

$$\begin{aligned} n &= 42 \\ \text{code} &= 1 \\ C(k) &= n * (n - 2) + \text{code} \\ C(k) &= 1681 \end{aligned}$$

Mit Einbindung der Konstanten **C(k)** errechnet sich der positionsgewichtete Wert **H(k)** wie folgt:

$$\begin{aligned} H(k) &= \sum_{i=1}^n (a_i + 1) * (p_i + C(k)) \\ H(k) &= 7033932 \end{aligned}$$

Basis 16: 6B544C Basis 77: FVRx

2. Bestimmung der Hashfunktionsfolge H(p)

Eine „**Expansionsfunktion**“ erweitert die **Startsequenz** auf möglichst viele Variablen (etwa 160 bis 2400 Ziffern). Der Wert jedes Zeichens **a(i)** wird mit der Position **p(i)** und der Positionsgewichtung **H(k)** multipliziert und zur **Hashfunktionsfolge H(p)** im Zahlensystem zur **Basis 77** hochgerechnet.

$$\begin{aligned} \text{Code} &= 1 \\ r &= 1 \end{aligned}$$

$$H(p) = \sum_{i=1}^n (a_i + 1) * p_i * H(k) + (p_i + \text{code} + r)$$

$$H(p) = 642915453651$$

Basis 16: 95B0BF4AD3 Basis 77: 36e3Jx9

Die für jedes Zeichen **a(i)** errechneten Ergebnisse ergeben die **Hashfunktionsfolge** mit 248 Ziffern im Zahlensystem zur **Basis 77** als uniforme Reihe:

D&7âågWQYczHmQfQVwkH1734Yâ1h4SRâ1â5R3N2FqSyâ2gMfv#2m7RG42#âerk44BRfd3â
 Aã7t3yQ02â3êBAth4yDUd81ZJbN84&ê7gá52yNâi5VFB445hkUVi6QmijZ1éâ9rä6N2J7r
 6mlhfJ7caVrt2OLãN76EHPI27qâuoæ9FPxeã9WvFâh9OArUm8bsj6vAWy0ukAZTNWi9uBâ
 7L9&C5âLBRV&#éCCY5ldAESfhEBIHAEâB1zqpç

Im Ausschnitt zeigen sich die Berechnung der Hashfunktionsfolge wie folgt:

char	pi	(ai+1)	(ai+1)*pi	Hk	(ai+1)*pi*Hk	pi+code+r	Erg	base 77
D	69	1	69	7033932	485341308	3	485341311	D&7ââ
i	106	2	212	7033932	1491193584	4	1491193588	gWQYc
e	102	3	306	7033932	2152383192	5	2152383197	zHmQf
	33	4	132	7033932	928479024	6	928479030	QVwkH
S	84	5	420	7033932	2954251440	7	2954251447	1734Yâ
c	100	6	600	7033932	4220359200	8	4220359208	1h4SRâ
h	105	7	735	7033932	5169940020	9	5169940029	1â5R3N
i	106	8	848	7033932	5964774336	10	5964774346	2FqSyâ
l	109	9	981	7033932	6900287292	11	6900287303	2gMfv#
d	101	10	1010	7033932	7104271320	12	7104271332	2m7RG4
b	99	11	1089	7033932	7659951948	13	7659951961	2#âerk
ü	130	12	1560	7033932	10972933920	14	10972933934	44BRfd
r	115	13	1495	7033932	10515728340	15	10515728355	3âAã7t
g	104	14	1456	7033932	10241404992	16	10241405008	3yQ02â
e	102	15	1530	7033932	10761915960	17	10761915977	3êBAth
r	115	16	1840	7033932	12942434880	18	12942434898	4yDUd8
..
Summe Hp:							642915453651	36e3Jx9

Die Ergebnisse der Rechenschritte bilden die Basis für die Bestimmung folgender Steuerungsparameter:

	dezimal	Basis 77
Hashkonstante C(k):	1681	L@
Positionsgewichteter Wert (H _k):	7033932	FVRx
Hashfunktionsfolge (H _p):	642915453651	36e3Jx9
Gesamtwert (H _p +H _k):	642922487583	36elp9ã

Variante	$(H_k \text{ MOD } 11) + 1$	=	5	Beginn der Kontraktion
Alpha	$((H_k + H_p) \text{ MOD } 255) + 1$	=	49	Offset Chiffre-Alphabet
Beta	$(H_k \text{ MOD } 169) + 1$	=	153	Offset Block-Schlüssel
Gamma	$((H_p + \text{code}) \text{ MOD } 196) + 1$	=	193	Offset Matrix-Schlüssel
Delta	$((H_k + H_p) \text{ MOD } 155) + \text{code}$	=	104	dynamische Bitfolgen
Theta	$(H_k \text{ MOD } 32) + 1$	=	13	dynamischer Faktor

3. Verdichtung zur BASIS VARIATION

Zur Rückführung auf dezimale Größen wird die **Hashfunktionsfolge** mit einer **Kontraktionsfunktion** verdichtet. Dazu wird für die Hashfunktionsfolge das Zahlensystem zur **Basis 78** (Expansionsbasis +1) unterstellt. Jeweils drei Ziffern der Folge werden durch **MODULO 256** in dezimale Zahlen 0 bis 255 (ohne Wiederholung) zurückgerechnet. Theta wird abgezogen und das Ergebnis in einem Array von 16x16 Elementen **BASIS VARIATION** gespeichert.

Die Rückrechnung beginnt beim Parameter **Variante = 5**:

.... ägWQYczHmQfQVwkH1.....

Ziffern Basis 78	dezimal	MODULO 256	Element -Theta	
....
ägW	429188	132	13	119
gWQ	258050	2	13	245
WQY	196750	142	13	129
QYc	160874	106	13	93
Ycz	209881	217	13	204
czH	235967	191	13	178
zHm	372498	18	13	5
HmQ	107198	190	13	177
mQf	294101	213	13	200
QfQ	161408	128	13	115
fQV	251503	111	13	98
QVw	160660	148	13	135
Vwk	193174	150	13	137
wkH	356477	125	13	112
kH1	281191	103	13	90
....

Die Kontraktion führt zu folgendem Array **BASIS-VARIATION** (256 Elemente):

119 245 129 093 204 178 005 177 200 115 098 135 137 112 090 076
220 061 153 165 080 004 213 051 166 222 110 075 016 044 006 160
067 058 065 099 136 240 235 141 094 248 130 244 203 251 034 241
154 131 109 223 242 095 045 231 143 103 070 120 194 068 060 253

001 083 091 092 020 053 161 207 221 208 077 236 036 195 162 138
 046 084 088 247 050 214 064 132 116 174 224 081 003 139 052 078
 171 096 062 007 159 218 038 111 056 097 072 079 205 155 066 180
 015 002 113 140 029 145 167 168 073 124 018 087 134 014 054 069
 147 179 144 063 082 215 123 142 071 100 254 114 040 199 008 035
 042 012 085 196 225 255 216 013 183 049 226 243 169 149 101 209
 106 102 133 086 230 028 146 074 057 148 089 227 210 228 117 163
 234 181 055 150 229 219 151 104 152 232 009 118 246 197 156 187
 027 059 233 105 107 172 108 125 237 157 249 121 158 182 190 010
 201 024 239 048 252 126 184 122 211 238 202 127 250 164 176 128
 000 011 017 170 019 173 047 175 198 039 043 021 186 217 022 185
 188 189 023 025 191 206 192 193 026 030 041 212 031 032 033 037

Die BASIS VARIATION ist Grundlage für die weiteren Ableitungen zur CypherMatrix.

4. Permutation zur CypherMatrix (Hashwert H)

Die dezimalen Werte der **BASIS VARIATION** (16x16) werden direkt auf die 256 Indexwerte des erweiterten ASCII-Zeichensatzes bezogen. Sie ergeben die erste **CypherMatrix (CM1)** mit 16x16 Elementen.

77	F5	81	5D	CC	B2	05	B1	C8	73	62	87	89	70	5A	4C
DC	3D	99	A5	50	04	D5	33	A6	DE	6E	4B	10	2C	06	A0
43	3A	41	63	88	F0	EB	8D	5E	F8	82	F4	CB	FB	22	F1
9A	83	6D	DF	F2	5F	2D	E7	8F	67	46	78	C2	44	3C	FD
01	53	5B	5C	14	35	A1	CF	DD	D0	4D	EC	24	C3	A2	8A
2E	54	58	F7	32	D6	40	84	74	AE	E0	51	03	8B	34	4E
AB	60	3E	07	9F	DA	26	6F	38	61	48	4F	CD	9B	42	B4
0F	02	71	8C	1D	91	A7	A8	49	7C	12	57	86	0E	36	45
93	B3	90	3F	52	D7	7B	8E	47	64	FE	72	28	C7	08	23
2A	0C	55	C4	E1	FF	D8	0D	B7	31	E2	F3	A9	95	65	D1
6A	66	85	56	E6	1C	92	4A	39	94	59	E3	D2	E4	75	A3
EA	B5	37	96	E5	DB	97	68	98	E8	09	76	F6	C5	9C	BB
1B	3B	E9	69	6B	AC	6C	7D	ED	9D	F9	79	9E	B6	BE	0A
C9	18	EF	30	FC	7E	B8	7A	D3	EE	CA	7F	FA	A4	B0	80
00	0B	11	AA	13	AD	2F	AF	C6	27	2B	15	BA	D9	16	B9
BC	BD	17	19	BF	CE	C0	C1	1A	1E	29	D4	1F	20	21	25

Aus dieser ersten **CypherMatrix (CM1)** können durch dreifache Permutationen der Elemente drei Varianten (A, B und C) gestaltet werden.

Im Pseudo-Code geschieht folgendes:

a) Einfacher Ersatz des Index: i (**Variante A**)

```
CM1Set$ = ""           Elemente der ersten CypherMatrix (CM1)
CM3Set$ = ""           Elemente der dritten CypherMatrix (CM3)

FOR s = 1 TO 3           drei Schleifen
  FOR i = 1 TO 16       (Permutation)
    FOR j = 1 TO 16
      a = i - j
      IF a <= 0 THEN a = 16 + a
      SELECT CASE s
        CASE 1
          CM1Set$ = CM1Set$ + Matrix$(1,i,j)  CypherString
          Matrix$(2,a,j) = Matrix$(1,i,j)
        CASE 2
          Matrix$(3,a,j) = Matrix$(2,i,j)
        CASE 3
          CM3Set$ = CM3Set$ + Matrix$(3,i,j)  CypherString
      END SELECT
    NEXT j
  NEXT i
NEXT s
```

b) Versetzter Austausch der Indizes: i,j (**Variante B**)

```
a = i - j
IF a <= 0 THEN a = 16 + a
SELECT CASE s
  CASE 1
    CM1Set$ = CM1Set$ + Matrix$(1,i,j)  CypherString
    Matrix$(2,a,j) = Matrix$(1,i,j)
  CASE 2
    Matrix$(3,i,a) = Matrix$(2,i,j)
  CASE 3
    CM3Set$ = CM3Set$ + Matrix$(3,i,j)  CypherString
END SELECT
```

c) Dynamische Generierung der Indizes: i,j (**Variante C**)

Alle Indexwerte (16x16) werden in einem gesonderten Index-Array **Index\$(16,16)** neu generiert (Anwendung der CypherMatrix Funktion):

```
a = i - j
IF a <= 0 THEN a = 16 + a
SELECT CASE s
  CASE 1
    CM1Set$ = CM1Set$ + Matrix$(1,i,j)  CypherString
    Matrix$(2,a,j) = Matrix$(1,i,j)
  CASE 2
    x = VAL(Index$(i,j))
    Matrix$(3,i,x) = Matrix$(2,i,j)
  CASE 3
    CM3Set$ = CM3Set$ + Matrix$(3,i,j)  CypherString
END SELECT
```