Hash-Konstante C

Bestimmungsfaktoren für Kollisionsfreiheit

(Ernst Erich Schnoor)

eschnoor@multi-matrix.de

Eine Ergänzung zum Artikel: **„Der Kern des CypherMatrix® Verfahrens“**

http://www.telecypher.net/CYPHKERN.HTM#Z6

Gegenstand der Hash-Berechnung ist die **Eingangs-Sequenz** von kryptographischen Verfahren mit 36 bis 64 Bytes. Das **CypherMatrix** Verfahren errechnet mit einer ersten **Einweg-Funktion** die positionsgewichtete Hash-Summe $\mathbf{H_k}$ der Eingangs-Sequenz nach folgender Formel:

$$H_k = \sum_{i=1}^{n} (a_i + 1) * (C + p_i)$$

Die Berechnung von $\mathbf{H_k}$ erfolgt unter den Bedingungen:

$1 <= (a_i + 1) <= 256$ (ASCII-Werte)

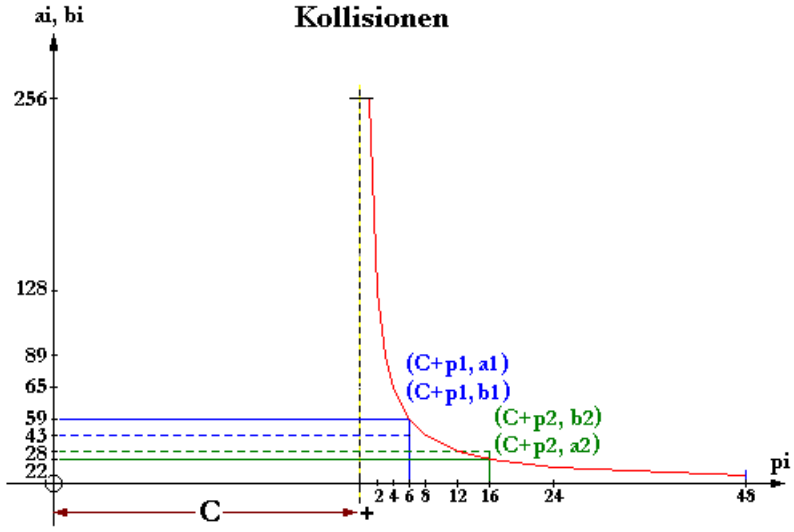
N = Länge der Eingangs-Sequenz:

$1 <= i <= N$

i, a_i, p_i = Integer

Damit für den ASCII-Wert(0) nicht der Faktor 0 entsteht, wird ($\mathbf{a_i}$) jeweils um 1 erhöht. Der zur Gewichtung der einzelnen Zeichen verwendete Positionsfaktor (C + pi) enthält die **Hash-Konstante** ($\mathbf{C_k}$). Die Konstante hat die Aufgabe, **Kollisionen** zu vermeiden.

Graphisch stellen sich die Zusammenhänge wie folgt dar:



$$a1*(C+p1)+a2*(C+p2) = b1*(C+p1)+b2*(C+p2)$$

Unter Ansatz der ASCII-Werte ist die Eingangs-Sequenz eine Zahlenfolge (**0 – 255**).. Der Hash-Wert $\mathbf{H_k}$ dagegen ist eine Summe von Produkten: **(ai + 1) * (C + pi).**

Eine Kollision entsteht, wenn trotz Austausch von Zeichen innerhalb der Zeichen-Sequenz ($\mathbf{b_i}$ für $\mathbf{a_i}$) und ($\mathbf{b_j}$ für $\mathbf{a_j}$) dieselbe Hash-Summe $\mathbf{H_k}$ resultiert:

$$\mathbf{H_k}(i) = \mathbf{H_k}(j)$$

Wird an gleicher Position ($\mathbf{p_i = p_j}$) nur ein Zeichen ausgetauscht ($\mathbf{b_i}$ für $\mathbf{a_i}$), dann muss sich rechnerisch auch das Ergebnis $\mathbf{H_k}$ ändern, da $\mathbf{b_i}$ von $\mathbf{a_i}$. verschieden ist. Eine **Kollision** ist dann natürlich nicht möglich. Um eine Kollision zu erzeugen muss beim Austausch von Zeichen die Summe der neuen Teilprodukte der Summe der bisherigen Teilprodukte entsprechen, damit die Hash-Summe $\mathbf{H_k}$ gleich bleiben kann. Bei mindestens **zwei Zeichen** ist das der Fall wie folgt:

$$a1 *(C+p1) + a2 *(C+p2) = b1 *(C+p1) + b2 *(C+p2)$$

$$a1 *(C+p1) - b1 *(C+p1) = b2 *(C+p2) - a2 *(C+p2)$$

$$(C+p1) * (a1 - b1) = (C+p2) * (b2 - a2)$$

Veränderungsquotient:

$$\frac{(C+p1)}{(C+p2)} = \frac{(b2 - a2)}{(a1 - b1)} = Q$$

$$\mathbf{(b2 – a2) = (a1 – b1) * Q}$$

Für den "Veränderungsquotienten"- hier mit **Q** bezeichnet - sind drei Fälle möglich:

Q > 1
Q = 1
Q < 1

Wenn **Q=1** dann müssen auch (**p1+C**) und (**p2+C**) gleich sein. Da der Austausch an derselben Position (p) im String geschieht, ist hier eine **Kollision ausgeschlossen**.

Ist **Q>1** oder **Q<1** dann sind auch (b2-a2) und (a1-b1) verschieden. Hier stellt sich die Frage, ob die Tatsache eine Rolle spielt, dass die ASCII-Werte **(a1, a2, b1,** und **b2) Integerwerte** sind und damit auch deren Differenzen.

In einer Minimum/Maximum Analyse und für eine Länge der Eingangs-Sequenz (**N=64** Bytes) und der Hash-Konstante **C=3968** zeigen sich die Zusammenhänge wie folgt:

min:	norm	max:	
$\frac{(C+1)}{(C+64)}$	$\frac{(C+i)}{(C+i)}$	$\frac{(C+64)}{(C+1)}$	= **<1 ..... 1 ..... >1**
$\frac{3968+1}{3968+64}$	$\frac{3968+i}{3968+i}$	$\frac{3698+64}{3968+1}$	= 0,98437 .... 1 1,01587

$$\mathbf{Q_{min} = 0,98437 \qquad Q_{max} = 1,01587}$$

Die ASCII-Werte (a1, a2, b1 und b2) sind Integerwerte und damit sind auch deren Differenzen ganzzahlig. Aus der Multiplikation (a1 - b1) * Q (mit Q = 0,98487 .... 1,01587) für den Bereich $\mathbf{p_i = 1}$ bis **64** (**N**) und $\mathbf{a_i = 1}$ bis **356** ergeben sich aber keine Ergebnisse (außer Q=1), die ganzzahlig sind und im Min-Max-Bereich liegen. Um Kollisionen zu vermeiden, muss daher der Quotient

$$\mathbf{Q = \frac{C + p1}{C + p2}}$$

größer sein als $\mathbf{Q_{min}}$ aber kleiner als $\mathbf{Q_{max}}$. Das ist der Fall bei folgendem Vergleich:

$$\mathbf{\frac{C + N}{C + 1} = \frac{N}{N - 1}}$$

Danach ergibt sich die Auflösung nach **C** wie folgt:

$$C + N = \frac{N * (C + 1)}{N - 1}$$

$$C = \frac{C*N + N}{N - 1} - N$$

$$C = \frac{C*N}{N - 1} + \frac{N}{N-1} - N$$

$$C - \frac{C*N}{N - 1} = \frac{N}{N-1} - N$$

$$C * \left(1 - \frac{N}{N -1}\right) = \frac{N}{N -1} - N$$

$$C = \frac{\left(\frac{N}{N - 1}\right) - N}{1 - \left(\frac{N}{N - 1}\right)}$$

Nach Umformung:

$$\mathbf{C_k = ((N / (N-1)-N)) / (1-(N / (N-1))}$$

$$\mathbf{C_k = N * (N - 2)}$$

$$\mathbf{N = 1/2 * sqrt(4 * (C_k + 1)) + 1}$$

Die Hash-Konstante ($\mathbf{C_k}$) muss sich im Bereich der Veränderungsquotienten $\mathbf{Q_{min}}$ bis $\mathbf{Q_{max}}$ bewegen, um Kollisionen zu vermeiden. $\mathbf{C_k}$ begründet offenbar eine Grenze zwischen **kollisionsfreien** und **kollisionsbelasteten** Hash-Werten. Die Grenze ist allerdings von der Länge **N** der Hash-Sequenz abhängig. Um die Grenze flexibel zu gestalten, kann ein individueller **UserCode** ergänzt werden. Im CypherMatrix Verfahren ist dieser Code mit (**+1**) vorgegeben:

$$\mathbf{C_k = N * (N - 2) + UserCode}$$

Als Hash-Sequenz wählen wir auszugsweise Worte von Hermann Hesse (Siddhartha, Eine indische Dichtung, Montagnola 1953)

denn Ursachen erkennen so schien ihm, daß eben ist Denken, und d

N = **64** und **Hash-Konstante** = **3968** und **Code** = **0**

An der Position **41** der Hash-Sequenz wird das Zeichen **a1** = **ß** (ASCII = **225**) mit dem Zeichen **b1** = **a** (ASCII = **97**) und an der Position **10** das Zeichen **a2** = **c** (ASCII = **99**) mit dem Zeichen **b2** = **Σ** (ASCII = **228**) ausgetauscht.

denn UrsaΣhen erkennen so schien ihm, daa eben ist Denken, und d

$$\mathbf{Q_{max}} = \frac{3968 + 41}{3968 + 10} = 1{,}007793$$

a1 *(C+p1) + a2 *(C+p2) = b1 *(C+p1) + b2 *(C+p2)

225 * (3968+41) + 99 * (3968+10) = 97 * (3968+41) + 228 * (3968+10)
902025 + 393822 =/= 388873 + 906984
1295847 =/= 1295857

Durch den Austausch der Zeichen ergeben sich **ungleiche** Ergebnisse, sodass **keine Kollision** entsteht.

N = **64** und **Hash-Konstante** = **3958** und **Code** = **0**

Die Hash-Konstante wird niedriger angesetzt, als nach der Formel mit **N * (N-2)** = 3968 berechnet. Der Austausch der Zeichen wird wie oben vorgenommen.

$$\mathbf{Q_{max}} = \frac{3958 + 41}{3958 + 10} = 1{,}0078125$$

a1 *(C+p1) + a2 *(C+p2) = b1 *(C+p1) + b2 *(C+p2)

225 * (3958+41) + 99 * (3958+10) = 97 * (3958+41) + 228 * (3958+10)
899775 + 392832 = 387903 + 904704
1292607 = 1292607

Durch den Austausch der Zeichen ergeben sich **gleiche** Ergebnisse, sodass hier eine **Kollision** entstehen muss.

Die positionsgewichtete Hash-Summe $\mathbf{H_k}$ der gewählten Sequenz ergibt sich mit:

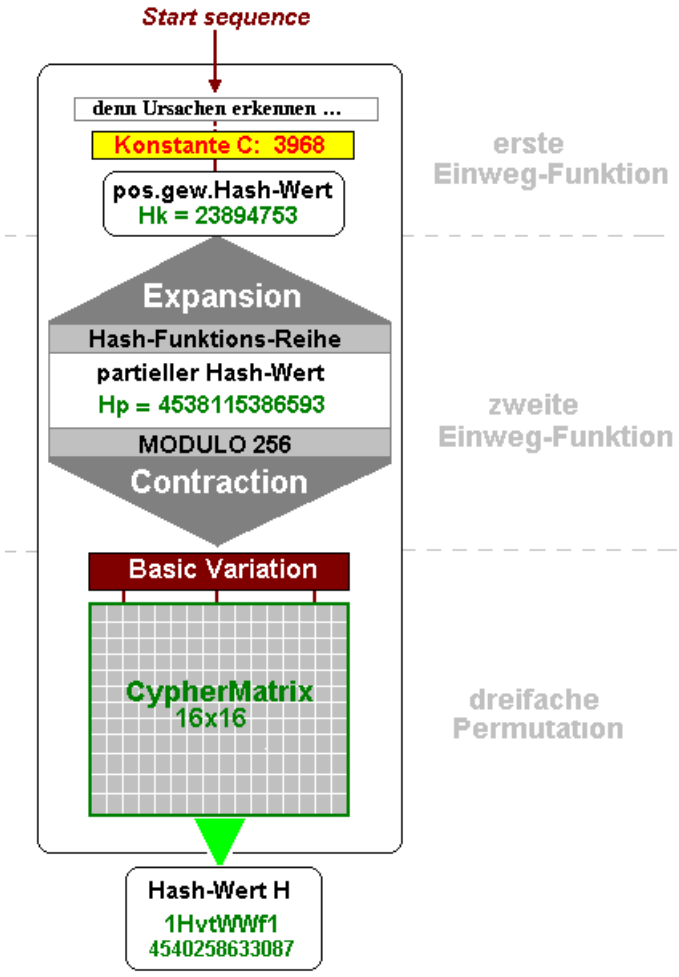
Basis 62: **1cG7F**
Basis 16: **16C9AE1**
Basis 10: **23894753**

Erweiterung der Hash-Funktion

Die vorstehende Abgrenzung allein ist für eine **sichere** Generierung kollisionsfreier Hash-Werte allerdings noch nicht ausreichend. Wie Versuchsreihen ergeben, hat der Min-Max-Bereich hin und wieder noch Löcher und lässt Ausnahmen zu. Das Verfahren wird daher um eine weitere **Einweg-Hashfunktion** erweitert.

$$H_p = \sum_{i=1}^{n} ((a_i + 1) * p_i * H_k) + (p_i + Code)$$

Die folgende Skizze zeigt die Zusammenhänge:



Jedes Zeichen der Hash-Sequenz ($\mathbf{a_i}$) wird zu einem Hash-Wert ($\mathbf{s_i}$) hochgerechnet und in einer **Hash-Funktions-Reihe** im Zahlensystem zur **Basis 77** gespeichert. Die Summe aller hochgerechneten Werte ergibt einen zweiten **partiellen Hash-Wert** ($\mathbf{H_p}$). Mit der Unterstellung, die hochgerechneten Werte valutieren im Zahlensystem zur Basis **78**

(Expansionsbasis +1) werden sie dann mit **MODULO 256** auf dezimale Werte zur **Basis Variation** zurückgeführt (Zahlen-Array mit 256 Ziffern) Dieser **Kontraktionsprozess** stellt eine zweite **Einweg-Funktion** dar. Die Funktion ist nicht umkehrbar und eine retrograde Bestimmung der Hash-Funktions-Reihe ist nicht möglich.

In der zweiten Einweg-Funktion entwickeln sich die Daten wie folgt:

denn Ursachen erkennen so schien ihm, daß eben ist Denken, und d

Char	a_i+1	Pos i	H_k	Produkt	+ Position i	s_i
U	86	6	23894753	12329692548	12329692554	4gvHàP
r	115	7	23894753	19235276165	19235276172	78ESãZ
s	116	8	23894753	22174330784	22174330792	8EzBFê
a	98	9	23894753	21075172146	21075172155	7yeeRb
c	100	10	23894753	23894753000	23894753 010	8#ujæp
h	105	11	23894753	27598439715	27598439726	AF7I1B
e	102	12	23894753	29247177672	29247177684	AzëpMW
n	111	13	23894753	34480128579	34480128592	Cuá2ä#
.	...	..	.....	.....	.....	.....
					---	---
					4538115386593	Lxhèd7J

Hash-Funktions-Reihe 391 Ziffern im Zahlensystem zur Basis 77

LximàltãoN3I1zpLTV2çR6ju3åzy9n1ZC2iG4gvHàP78ESãZ8EzBFê7yeeRb8#ujæpAF7I
1BAzëpMWCuá2ä#462ã&EDcëiáeGIsá2aGFëcXlGFëcXmIlhG7LJk0ämEIäëW0uLgåMr06r
åw3COiTXlæOtJâ0W7iGSybRnælYpOtJâ0ZQâz5c1S5gåkBRåOzåvVRWJ079lH#EiV&wävX
WY21OnYèvzwäErwWéæB5TàwyYxapn9YkhG&J14zWrêrCI8qWgctOaGCcYæLvBedêvjhj5t
êX#DrKsëwiåbáwXoDlKe7pnYnItEàëfoeVpãaSEltná9DqåPfr7qXmFnáoWnxrLtàqmL#N
38Qt1HEYEê4&ceQWåxxbVVZtLcgkcIRD1m#v4&Bqé

Die vorstehenden Zeichen sind Zahlen im Zahlensystem zur **Basis 77**. Für die folgende Verdichtung zur **GRUND VARIATION** wird unterstellt, sie seien Zahlen im Zahlensystem zur **Basis 78**.

Kontraktionsprozess

Zur Rückführung auf dezimale Größen werden jeweils drei Ziffern der Hash-Funktions-Reihe mit **MODULO 256** in dezimale Zahlen 0 bis 255 (ohne Wiederholung) umgewandelt und als GRUND-VARIATION in einem Array mit 256 Elementen gespeichert.

*LximàltãoN3I1zpLTV2çR6ju3åzy9n1ZC2iG*4gvHàP78ESãZ8EzBFê7yeeRb8#ujæpAF7I
1BAzëpMWCuá2ä#*462ã&EDcëiáeGIsá2aGFëcXlGFëcXmIlhG7LJk0ämEI.....*

Basis 78	Dezimal	Modulo 256
4gv	27669	021
gvH	259991	151
vHà	348179	019
HàP	108523	235
àP7	397417	105
P78	152654	078
78E	43226	218

Die mit MODULO 256 auf dezimale Zahlen umgewandelten Zahlen Basis 77 speichert das Verfahren im Array BASIC-VARIATION

160 119 206 006 099 077 152 201 083 141 084 143 009 168 161 147
096 173 134 135 016 221 106 005 223 211 245 122 118 248 000 148
162 021 151 019 235 105 078 218 128 132 075 194 074 163 089 029
073 093 202 204 072 236 255 250 003 138 121 222 076 240 039 065
176 217 033 040 181 139 015 208 224 164 097 130 254 107 205 010
123 230 120 126 002 203 212 213 108 219 244 200 146 242 127 124
079 144 129 030 207 229 085 166 237 031 209 231 086 246 109 069
038 214 125 215 012 145 131 054 133 022 136 026 028 241 184 037
112 041 149 137 001 154 011 004 034 071 094 013 172 232 167 113
225 182 227 053 042 247 110 199 210 216 174 007 226 081 191 228
087 220 140 170 150 055 043 249 111 196 251 153 233 098 114 142
082 032 234 171 169 175 238 024 044 060 066 023 049 025 155 080
156 045 088 090 115 068 177 063 100 239 116 178 091 243 046 252
165 253 070 117 047 157 158 159 179 180 092 027 101 183 095 102
187 017 008 061 185 014 186 048 103 104 018 188 189 056 190 020
057 067 035 058 198 192 036 193 195 197 050 051 052 059 062 064

In drei Schleifen (**Permutationen**) generiert das Verfahren aus den Ziffern des Arrays die **CypherMatrix** als finales Ergebnis der Hash-Funktion.

CypherMatrix

1	5C 33 76 F0 7F 25 57 2D 08 06 EB 8B 55 04 6F EF	16
17	12 8F 4A 6B 6D 71 52 FD 23 87 48 CB 83 C7 2C B4	32
33	32 7A 4C F2 B8 E4 9C 11 CE 13 B5 E5 0B F9 64 68	48
49	54 C2 FE F6 A7 8E A5 43 86 CC 02 91 6E 18 B3 C5	64
65	F5 DE 92 F1 BF 50 BB 77 97 28 CF 9A 2B 3F 67 8D	80
81	4B 82 56 E8 72 FC 39 AD CA 7E 0C F7 EE 9F C3 D3	96
97	79 C8 1C 51 9B 66 A0 15 21 1E 01 37 B1 30 53 84	112
113	61 E7 AC 62 2E 14 60 5D 78 D7 2A AF 9E C1 DF 8A	128
129	F4 1A E2 19 5F 40 A2 D9 81 89 96 44 BA C9 80 A4	144
145	D1 0D E9 F3 BE 93 49 E6 7D 35 A9 9D 24 05 03 DB	160
161	88 07 31 B7 3E 94 B0 90 95 AA 73 0E 98 DA E0 1F	176
177	5E 99 5B 38 A1 1D 7B D6 E3 AB 2F C0 6A FA 6C 16	192
193	AE 17 65 3B 00 41 4F 29 8C 5A B9 4D 4E D0 ED 47	208
209	FB B2 BD A8 59 0A 26 B6 EA 75 C6 DD FF D5 85 D8	224
225	42 1B 34 F8 27 7C 70 DC 58 3D 63 69 0F A6 22 C4	240
241	74 BC 09 A3 CD 45 E1 20 46 3A 10 EC D4 36 D2 3C	256

Für die CypherMatrix errechnet das Verfahren den abschließenden Hash-Wert **H** mit:

Basis 62: **1HvtWWf1**
Basis 16: **4211C80C57F**
Basis 10: **4540258633087**

Zurück zum Artikel: **„Der Kern des CypherMatrix® Verfahrens“**

http://www.telecypher.net/CYPHKERN.HTM#Z6

München, den 13.02.2007