Hash Constant C

Determinants leading to collisionfree

(Ernst Erich Schnoor)

eschnoor@multi-matrix.de

Addendum to article: **„Core of the CypherMatrix® Method“**

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

Object of hash calculation is the **start sequence** of cryptographic operations with 36 to 64 bytes. **CypherMatrix** method computes with a first **One-way function** the position weighted hash value H_k of the start sequence by following formula:

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

Calculation of H_k is performed under these conditions:

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

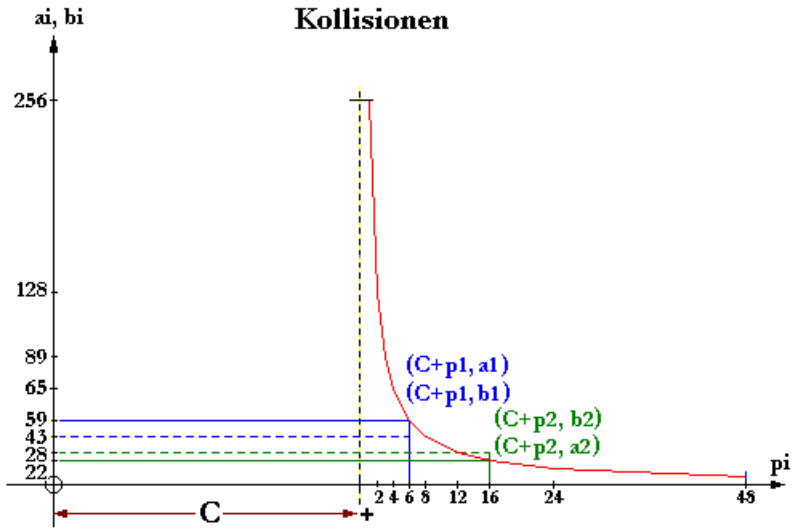
N = length of the start sequence

$1 <= i <= N$

i, a_i, p_i = integer

In order to avoid resulting to factor zero for ASCII value (0) the single values of (a_i) are incrased by (+1). The position factor (C + pi) determined to weight each single character with its position in the sequence includes the **hash constant** (C_k). The constant has the task to prevent **collisions**.

The following graph illutrates the connections:



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

The start sequence is a series of numbers (**0 – 255**) denoted by ASCII values. On the other hand the hash value H_k is a sum of multiplications **(ai + 1) * (C + pi).**

A collision occurs when despite of changing characters inside a start sequence (b_i for a_i) and (b_j for a_j) an equal hash value results.

$$H_k(i) = H_k(j)$$

If a single character only is exchanged at same position ($p_i = p_j$) then the result H_k has to change as well, because b_i is different from a_i**.** Of course, a collision is not possible at this point. To get a collision in case of exchanging characters within the sequence the sum of the new partial products have to equal the sum of the previously partial products. When exchanging atleast **two characters** the following conditions will be effective:

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

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

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

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

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

For the "**changing quotient**"- here denoted with **Q** – three cases are relevant:

Q > 1
Q = 1
Q < 1

If **Q=1** then (**C+p1**) and (**C+p2**) must be equal as well. Because exchanging of characters taking place at the same position (p) in the sequence a **collision** is **excluded**.

If **Q>1** or **Q<1** then (b2-a2) and (a1-b1) have to be different as well. The question arise: Is it important single ASCII values **(a1, a2, b1,** and **b2)** beeing **integer values** and by this their differences as well. In a **minimum / maximum** analysis of the changing quotient **Q** and a start sequence (**N = 64** bytes) and the hash constant C_k **= 3968** we get the interdependences as follows:

min: norm max:

$$\frac{(C + 1) \quad ..... \quad (C + i) \quad ..... \quad (C + 64)}{(C + 64) \quad ..... \quad (C + i) \quad ..... \quad (C + 1)} = \mathbf{<1 \ ..... \ 1 \ ..... \ >1}$$

$$\frac{3968 + 1 \quad ..... \quad 3968 + i \quad ..... \quad 3698 + 64}{3968 + 64 \quad ..... \quad 3968 + i \quad ..... \quad 3968 + 1} = 0{,}98437 \quad .... \ 1 \ .... \ 1{,}01587$$

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

The ASCII values (a1, a2, b1 and b2) are integer values and consequently their differences are integer too. But multiplikation (a1 - b1) * Q (with Q = 0,98487 .... 1,01587) in the sector $\mathbf{p_i}$ = **1** to **64** (**N**) and $\mathbf{a_i}$ = **1** to **356** leads to no results (**except Q=1**) which are integer and are inside the calculated min-max sector. To avoid collisions the changing quotient

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

has to be major than $\mathbf{Q_{min}}$ but minor than $\mathbf{Q_{max}}$. That is the case in following comparisons:

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

Dissolution to **C** develops as follows:

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

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

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

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

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

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

With transformation:

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

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

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

To avoid collisions the hash constant ($\mathbf{C_k}$) has always to result inside the sector of the

changing quotient $\mathbf{Q_{min}}$ to $\mathbf{Q_{max}}$. Obviously $\mathbf{C_k}$ establishes a border between **collisionfree** and **collision burdened** hash values. However, the border depends on length **N** of the hash sequence. In order to make the border more flexible an individual **UserCode** may be added to the formula. In CypherMatrix procedures this code is predefined with (**+1**).

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

As an example we choose the hash sequence with words from Hermann Hesse (Siddhartha, Eine indische Dichtung, Montagnola 1953):

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

N = **64** and **hash constant** = **3968** and **code** = **0**

At position **41** of a hash sequence the sign **a1** = **ß** (ASCII = **225**) is exchanged with the character **b1** = **a** (ASCII = **97**) and at position **10** the character **a2** = **c** (ASCII = **99**) is exchanged with the sign **b2** = **Σ** (ASCII = **228**):

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

Exchanging the signs leads to **unequal** results, hence, there occurs **no collision**.

N = **64** and **hash constant** = **3958** and **code** = **0**

The hash constant is fixed minor than calculated by formula **N * (N-2)** = 3968.
Exchanging of characters are the same as above.

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

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**

Exchanging the signs lead to **equal** results. Consequently here arises a **collision**.

The position weighted hash value $\mathbf{H_k}$ of the chosen sequence results in:

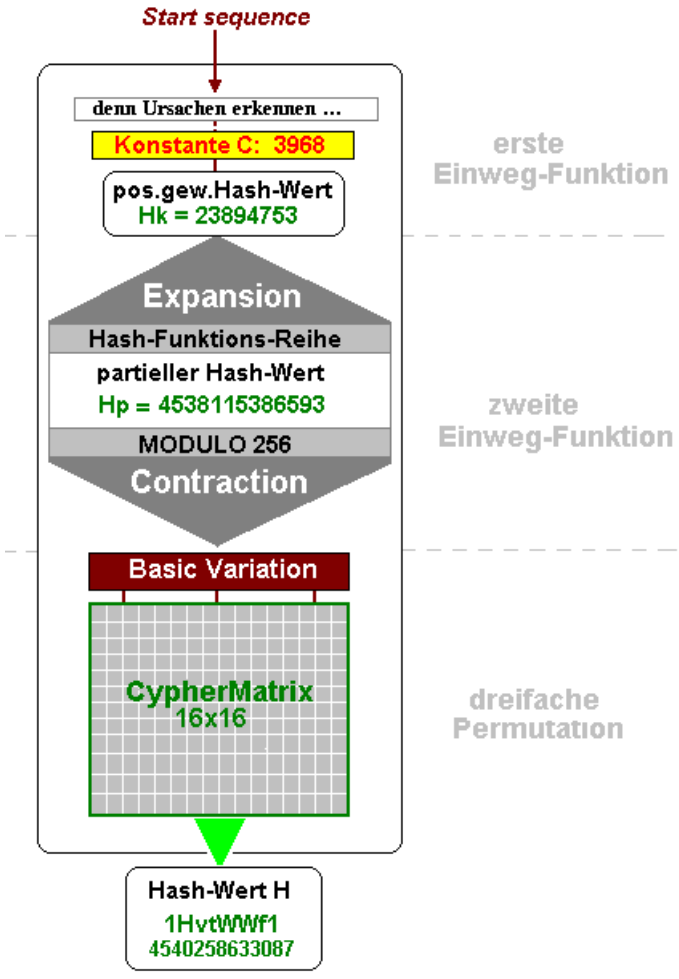
base 62: **1cG7F**
base 16: **16C9AE1**
base 10: **23894753**

Extension of the hash function

But even the preceding demarcation single-handed does not guarantee a secure hash value. How test series realize the min-max-area on and off has some leaks and tolerates exclusions. Therefore the procedure is extended by another **One-way function.**

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

Following scheme demonstrates the interdependences:



Each character of the hash sequence is extended to a hash value ($\mathbf{s_i}$) and aggregated to a **Hash function series** in number system on **base 77**. The sum of all extended values result in a second **partial hash value** ($\mathbf{H_p}$). Next with assuming the digits beeing numbers in number system on **base 78** (expansion base +1) the values are converted back by **MODULO 256** to decimal numbers and are stored in an array **Basic Variation**. This contraction process constitutes a second **One-way function.** The function is irreversible

and a retrospective destination of the hash function series is not possible.

Data in the second one-way function promote as follows:

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

char	a_i+1	pos i	H_k	product	+ 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 functions series 391 digits in number system on base 77

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

The foregoing characters are numbers in number system on **base 77**. The following contraction to **BASIC VARIATION** assumes the numbers are part of number system on **base 78.**

Contraction process

In order to reduce the base 78 digits to decimal values each three numbers of the hash function series are converted by **MODULO 256** to decimal numbers 0 to 255 (without repetition) and are stored in an array with 256 elements: BASIC VARIATION.

*LximàltãoN3I1zpLTV2çR6ju3åzy9n1ZC2iG4gvHàP78ESãZ8EzBFê7yeeRb8#ujæpAF7I
1BAzëpMWCuá2ä#462ã&EDcëiáeGIsá2aGFëcXlGFëcXmIlhG7LJk0ämEI.....*

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

The MODULO 256 reduced digits are stored in the 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 three rounds (**permutations**) the procedure generates the elements of the array to the **CypherMatrix** as a final result of the hash function..

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

The procedure calculates the **CypherMatrix** as a complete hash value **H** with the following results:

base 62: **1HvtWWf1**
base 16: **4211C80C57F**
base 10: **4540258633097**

Back to article: **„Core of the CypherMatrix® Method“**

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

Munich, in February 2007