On the Design of IDEA-128

Jorge Nakahara Jr

[1]Universidade Católica de Santos, UNISANTOS, BRAZIL

jorge_nakahara@unisantos.br

Abstract. *This paper describes five hypothetical realizations of IDEA-128, a 128-bit block cipher, using a 256-bit key, iterating 16.5 rounds, and operating on 32-bit words. These parameters are exactly double the size of the IDEA block cipher's. These IDEA-128 variants differ only in the multiplicative group structure:* $\mathbb{Z}^*_{2^{32}}$, $\mathbb{Z}^*_{2^{32}+1}$, *GF*(2^{32}), $\mathbb{Z}^*_{2^{32}-1}$, *or GF*$(2^{32}+15)$. *All of these designs have weaknesses related to the structure of these multiplicative groups, which lead to decryption failures or cryptanalytic attacks. The overall conclusion is that none of these variants constitute a secure cipher, and thus, help corroborate the design of the MESH ciphers, which operate on 16-bit words and use the same operations of IDEA, but allows text blocks larger than 64 bits, without compromising security.*

Keywords: IDEA block cipher, cryptanalysis, weak keys, algebraic groups.

1. Introduction

Ever since the publication of PES in 1990 [16], of IDEA in 1991 [16], and until 2002 [20], no variant of the IDEA cipher with block size larger than 64 bits has resisted public cryptanalysis. The closest design was Akelarre [2], but it used bitwise rotation instead of multiplication. Moreover, Akelarre was broken by Knudsen and Rijmen in [14]. Such larger-block cipher variants would be useful for instance as a buiding block for the construction of hash functions, stream ciphers and MAC algorithms [19, p.229, 340, 353]. Moreover, 128-bit block size is commonplace among modern block ciphers [1, 21], and help defeat some drawbacks inherent to 64-bit block ciphers in some modes of operation [13]: "if an n-bit block cipher is used in CBC, CFB, or OFB modes, information on the plaintext starts to leak after $2^{n/2}$ encryptions. This shows that block lengths of 128 bits are desirable in the near future.".

The original group operations on 16-bit words in IDEA are: bitwise exclusive-or, denoted $\oplus$; modular addition in $\mathbb{Z}_{2^{16}}$, denoted $\boxplus$; and multiplication in GF$(2^{16}+1)$, denoted $\odot$, where $0 \equiv 2^{16}$ (Fig. 1 with $\circledast = \odot$, but only eight rounds). An intuitive approach for the construction of larger IDEA variants would be to use operations on 32-bit words, doubling the original word size in IDEA. Extending $\oplus$ and $\boxplus$ to 32 bits is straightforward. The real problem is to find an appropriate multiplicative group operating on 32-bit words: $2^{32}+1$ is an obvious candidate but it is composite, in contrast to $2^{16}+1$, which is the last known Fermat prime [8]. This word size is also attractive for operation on popular desktop processors such as Intel's Pentium I/II/III/4 and AMD's Athlon/Duron, but slow on 8-bit and 16-bit processors. Apart from efficiency issues, this paper describes five hypothetical realizations of IDEA-128, operating on 128-bit blocks, under a 256-bit key, iterating 16 rounds plus an output transformation (OT), and with operations on 32-bit words. The OT stands for half a round. IDEA-128 has the same computational graph

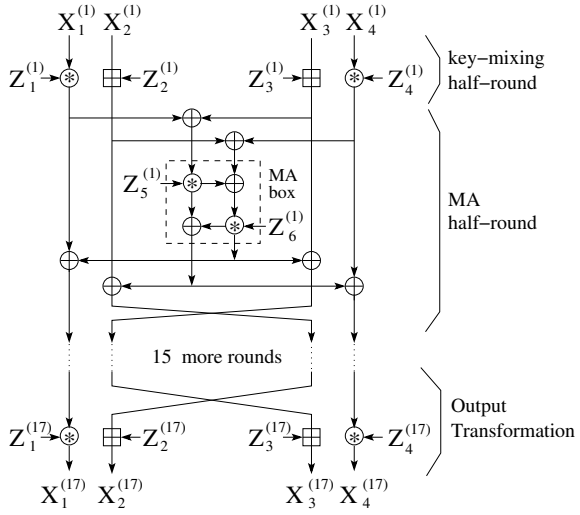

Figure 1. Computational graph of IDEA-128.

of IDEA, but more rounds and the multiplication operation in different algebraic groups. Each round in IDEA-128 is composed of a key-mixing and an MA half-round just like in IDEA. Both ciphers use the same computational graph for encryption and decryption, with just an appropriate change of subkeys for each operation. It follows that the encryption and decryption frameworks have the same cryptographic strength. Thus, without loss of generality, we will focus attention on the encryption operation.

Two hypothetical key schedule algorithms for IDEA-128 are described. The main results in each IDEA-128 variant are the existence of forbidden subkeys (for which decryption fails), and weak subkeys (allowing linear or differential attacks). The first phenomenon is absent is IDEA. Avoiding them in IDEA-128 could either cause noticeable delays during subkey generation (larger than one encryption operation), or reduce considerably the subkey space. Additionally, it could potentially lead to timing attacks [15]. The key schedule algorithm of IDEA, for example, was not designed to avoid weak keys [7], and therefore, has been the main weakness exploited by the most effective attacks on the full cipher [5, 7, 10]. These findings corroborate the design of the MESH ciphers [20], which operate on 16-bit words and use the same operations of IDEA, but allows text blocks larger than 64 bits, without compromising security.

This paper is organized as follows: Sect.2. describes two hypothetical key schedule algorithms for IDEA-128, and some of their properties. Sect. 3. describes IDEA-128 with multiplication in $\mathbb{Z}^*_{2^{32}}$. Sect. 4. discusses IDEA-128 with multiplication in $\mathbb{Z}^*_{2^{32}+1}$. Sect. 5. discusses IDEA-128 with multiplication in GF(2^{32}). Sect. 6. discusses IDEA-128 with multiplication in $\mathbb{Z}^*_{2^{32}-1}$. Sect. 7. discusses IDEA-128 with multiplication in GF($2^{32} + 15$). Sect. 8. concludes the paper.

2. Key Schedule Algorithms

This section describes two examples of key schedule or key setup (KS) algorithms for IDEA-128. The first of them is an extended version of IDEA's [16]:

Algorithm KS_1:

- let $K = K_1|K_2|K_3|K_4|K_5|K_6|K_7|K_8$ be the original 256-bit user key, partitioned into

Table 1. Mapping subkey bits to the 256-bit user key.

i-th round	$Z_1^{(i)}$	$Z_2^{(i)}$	$Z_3^{(i)}$	$Z_4^{(i)}$	$Z_5^{(i)}$	$Z_6^{(i)}$
1	1–32	33–64	65–96	97–128	129–160	161–192
2	193–224	225–256	50–81	82–113	114–145	146–177
3	178–209	210–241	242–17	18–49	99–130	131–162
4	163–194	195–226	227–2	3–34	35–66	67–98
5	148–179	180–211	212–243	244–19	20–51	52–83
6	84–115	116–147	197–228	229–4	5–36	37–68
7	69–100	101–132	133–164	165–196	246–21	22–53
8	54–85	86–117	118–149	150–181	182–213	214–245
9	39–70	71–102	103–134	135–166	167–198	199–230
10	231–6	7–38	88–119	120–151	152–183	184–215
11	216–247	248–23	24–55	56–87	137–168	169–200
12	201–232	233–8	9–40	41–72	73–104	105–136
13	186–217	218–249	250–25	26–57	58–89	90–121
14	122–153	154–185	235–10	11–42	43–74	75–106
15	107–138	139–170	171–202	203–234	29–60	61–92
16	93–124	125–156	157–188	189–220	221–252	253–28
OT	78–109	110–141	142–173	174–205	—	—

eight 32-bit words. These words form the first eight round subkeys: $Z_1^{(1)}, \ldots, Z_2^{(2)}$.

- rotate left the 256-bit key by 49($= \frac{3}{2} * 32 + 1$) bits (in IDEA, the 128-bit user key is rotated $\frac{3}{2} * 16 + 1 = 25$ bits to the left). Then, partition the resulting 256 bits into eight 32-bit words, forming the next eight round subkeys. Repeat this procedure until 100 subkeys for 16.5 rounds are obtained.

Let the bits of K be numbered sequentially from 1 up to 256 in left-to-right order. Thus, for example, K_1 corresponds to bits 1–32 of K. Table 1 shows the mapping of the subkey bits to the bits of K.

Properties of KS_1 include:

- The encryption and decryption subkeys can be generated on-the-fly, because each subkey depends only on 32 consecutive bits of the user key, and thus, can be computed independently. On the other hand, there is a high overlapping of bits among subkeys, allowing the reconstruction of the user key and of other subkeys if any subkey is even partially recovered (just like in IDEA).
- The subkey generation is faster than one encryption operation since only bit rotations are used. No arithmetic operations are involved.
- There is no provision to avoid patterns in the subkeys. For example, the user key with all bits equal to zero leads to all encryption subkeys being zero, too. Similarly, the user key with all bits equal to one causes all encryption subkeys to equal $2^{32} - 1 = \texttt{ffffffff}_\texttt{x}$ (in hexadecimal notation).

The second key schedule algorithm is described as follows:

Algorithm KS_2:

- define 32-bit constants c_i as follows $c_0 = 1$ and $c_i = 3 \cdot c_{i-1}$ for $i \geq 1$, with multiplication in GF(2)$[x]/q(x)$, where $q(x) = x^{32} + x^{28} + x^{25} + x^{22} + x^{18} + x^{15} +$

$x^{13} + x^{10} + x^8 + x^6 + x^5 + x^2 + 1$ is a primitive polynomial[1] over GF(2). The constant '3' is represented by the polynomial $x + 1$ in $GF(2)[x]/q(x)$.

- the 256-bit key is partitioned into eight 32-bit words K_i, for $0 \leq i \leq 7$, which constitute the first eight subkeys: $Z_i^{(1)} = K_i \oplus c_{i-1}$, $1 \leq i \leq 6$, $Z_1^{(2)} = K_7 \oplus c_6$, and $Z_2^{(2)} = K_8 \oplus c_7$.
- each subsequent 32-bit subkey is generated as follows:

$$Z_{l(i)}^{(h(i))} = \Big(\Big(\Big(\Big(\Big(\Big(Z_{l(i-8)}^{h(i-8)} \oplus c_{l(i-8)}\Big) \boxplus \Big(Z_{l(i-7)}^{h(i-7)} \oplus c_{l(i-7)}\Big)\Big) \oplus \Big(Z_{l(i-6)}^{h(i-6)} \oplus c_{l(i-6)}\Big)\Big) \boxplus \Big(Z_{l(i-5)}^{h(i-5)} \oplus c_{l(i-5)}\Big)\Big) \oplus \Big(Z_{l(i-2)}^{h(i-2)} \oplus c_{l(i-2)}\Big)\Big) \boxplus \Big(Z_{l(i-1)}^{h(i-1)} \oplus c_{l(i-1)}\Big)\Big) \lll 1\,, \quad (1)$$

for $8 \leq i \leq 99$, where '$\lll 1$' means one-bit left rotation, $h(i) = i \text{ div } 6 + 1$, and $l(i) = i \bmod 6 + 1$.

Properties of KS_2 include:

- Encryption subkey generation is on-the-fly, since subkeys can be computed sequentially according to (1). But, decryption is not on-the-fly.
- Due to the recurrence (1), based on the primitive polynomial $x^8 + x^7 + x^6 + x^5 + x^2 + x + 1$ over GF(2), there is no key overlapping as in KS_1.
- Notice that both of $\boxplus$ and $\oplus$ operations preserve the relative bit position of its operands. The one bit left-rotation destroys that property, so that changes only at the most significant bit of some subkeys (for instance, in a related-key attack [3]) would not propagate to other subkeys.
- The 32-bit constants are intended to avoid bit patterns in the subkeys. For example, without the constants, the all-zero user key would lead to all subkeys been zero.
- Each subkey after $Z_2^{(2)}$ requires three additions, eight xors, and one bit rotation according to (1). Assume that each such operation requires one machine cycle. This means that computing the remaining 92 subkeys requires $92 \cdot 8 = 736$ cycles. Assume one multiplication requires three times more than an addition or an xor. Then, one round of IDEA-128 requires four multiplications, four additions, and six xors. Thus, 16.5 rounds requires $16 \cdot (4 + 6 + 3 \cdot 4) + 2 \cdot (3 + 1) = 360$ cycles, and KS_2 is estimated to cost $\frac{736}{360} \approx 2.04$ or 104% more than one encryption.

3. Multiplication in $\mathbb{Z}_{2^{32}}^*$

The first suggested group for IDEA128 is $(\mathbb{Z}_{2^{32}}^*, *)$, which is motivated by the multiplication operation in the MARS block cipher [6]. Formally, $\mathbb{Z}_{2^{32}}^* = \{x \in \mathbb{Z}_{2^{32}} | gcd(x, 2^{32}) = 1\}$, where gcd stands for the greatest common divisor, and $\circledast = *$ in Fig. 1. It implies that the multiplicative subkeys in the key-mixing half-round should be relatively prime to 2^{32}, otherwise decryption would not be possible. This situation is analogous to that in the NTRU cryptosystem [11]. In the MA half-round though, subkeys that are not relatively prime to 2^{32} do not lead to decryption failures, but they may result in non-bijective MA-boxes, which can be a potential weakness for other techniques, such as impossible differentials [4] and non-surjective attacks [23].

Even more serious, though, is the fact that this group operation makes IDEA-128 susceptible to a linear attack involving only the least significant bit (LSB) of some 32-bit

[1] Chosen at random.

Table 2. 1-round characteristics and linear relations under weak-subkey assumptions.

Input Diff. $\xrightarrow{1r}$ Output Diff.	Input Mask $\xrightarrow{1r}$ Output Mask	Weak Subkeys (i-th round)
$(0,0,0,\delta) \xrightarrow{1r} (\delta,\delta,\delta,0)$	$(0,0,0,1) \xrightarrow{1r} (0,0,1,0)$	$Z_4^{(i)}, Z_6^{(i)}$
$(0,0,\delta,0) \xrightarrow{1r} (\delta,0,0,0)$	$(0,0,1,0) \xrightarrow{1r} (1,0,1,1)$	$Z_5^{(i)}, Z_6^{(i)}$
$(0,0,\delta,\delta) \xrightarrow{1r} (0,\delta,\delta,0)$	$(0,0,1,1) \xrightarrow{1r} (1,0,0,1)$	$Z_4^{(i)}, Z_5^{(i)}$
$(0,\delta,0,0) \xrightarrow{1r} (\delta,\delta,0,\delta)$	$(0,1,0,0) \xrightarrow{1r} (0,0,0,1)$	$Z_6^{(i)}$
$(0,\delta,0,\delta) \xrightarrow{1r} (0,0,\delta,\delta)$	$(0,1,0,1) \xrightarrow{1r} (0,0,1,1)$	$Z_4^{(i)}$
$(0,\delta,\delta,0) \xrightarrow{1r} (0,\delta,0,\delta)$	$(0,1,1,0) \xrightarrow{1r} (1,0,1,0)$	$Z_5^{(i)}$
$(0,\delta,\delta,\delta) \xrightarrow{1r} (\delta,0,\delta,\delta)$	$(0,1,1,1) \xrightarrow{1r} (1,0,0,0)$	$Z_4^{(i)}, Z_5^{(i)}, Z_6^{(i)}$
$(\delta,0,0,0) \xrightarrow{1r} (0,\delta,0,0)$	$(1,0,0,0) \xrightarrow{1r} (0,1,1,1)$	$Z_1^{(i)}, Z_5^{(i)}, Z_6^{(i)}$
$(\delta,0,0,\delta) \xrightarrow{1r} (\delta,0,\delta,0)$	$(1,0,0,1) \xrightarrow{1r} (0,1,0,1)$	$Z_1^{(i)}, Z_4^{(i)}, Z_5^{(i)}$
$(\delta,0,\delta,0) \xrightarrow{1r} (\delta,\delta,0,0)$	$(1,0,1,0) \xrightarrow{1r} (1,1,0,0)$	$Z_1^{(i)}$
$(\delta,0,\delta,\delta) \xrightarrow{1r} (0,0,\delta,0)$	$(1,0,1,1) \xrightarrow{1r} (1,1,1,0)$	$Z_1^{(i)}, Z_4^{(i)}, Z_6^{(i)}$
$(\delta,\delta,0,0) \xrightarrow{1r} (\delta,0,0,\delta)$	$(1,1,0,0) \xrightarrow{1r} (0,1,1,0)$	$Z_1^{(i)}, Z_5^{(i)}$
$(\delta,\delta,0,\delta) \xrightarrow{1r} (0,\delta,\delta,\delta)$	$(1,1,0,1) \xrightarrow{1r} (0,1,0,0)$	$Z_1^{(i)}, Z_4^{(i)}, Z_5^{(i)}, Z_6^{(i)}$
$(\delta,\delta,\delta,0) \xrightarrow{1r} (0,0,0,\delta)$	$(1,1,1,0) \xrightarrow{1r} (1,1,0,1)$	$Z_1^{(i)}, Z_6^{(i)}$
$(\delta,\delta,\delta,\delta) \xrightarrow{1r} (\delta,\delta,\delta,\delta)$	$(1,1,1,1) \xrightarrow{1r} (1,1,1,1)$	$Z_1^{(i)}, Z_4^{(i)}$

words across all the 16.5 rounds. This attack is based on the fact that the algebraic structure of $\mathbb{Z}^*_{2^{32}}$ makes the LSB of the multiplication result depend only on the LSB of its operands. For the $\boxplus$ and $\oplus$ operations this linear relation also holds with certainty. Therefore, one can construct linear relations across the full 16.5-round IDEA-128, and actually **for any number of rounds**, with (maximum) bias 2^{-1} [17], whatever the key/subkey values, namely, independent of the key schedule algorithm.

More precisely, let $X = X_1|X_2|X_3|X_4$ and $Y = Y_1|Y_2|Y_3|Y_4$ be the input and output blocks of one round of IDEA-128. Let $(A,B,C,D) \xrightarrow{1r} (E,F,G,H)$ denote the one-round linear relation for which the four-word linear mask (A,B,C,D) applied to X causes the output mask (E,F,G,H) applied to Y. The middle column of Table 2 lists all non-trivial (with non-zero masks) one-round linear relations involving only the LSB of the words in an input/output block pair, and weak multiplicative subkey assumptions.

Using Table 2 several linear relations for the full 16.5-round IDEA-128 with bias 2^{-1}, can be constructed round by round. An example is

$$(1,0,1,0) \xrightarrow{1r} (1,1,0,0) \xrightarrow{1r} (0,1,1,0) \xrightarrow{1r} (1,0,1,0)\,, \tag{2}$$

a 3-round iterative linear relation, with the least number of weak subkeys. Let the input plaintext be denoted $X^{(1)} = (X_1^{(1)}, X_2^{(1)}, X_3^{(1)}, X_4^{(1)})$, and the ciphertext output after 16.5 rounds be $X^{(17)} = (X_1^{(17)}, X_2^{(17)}, X_3^{(17)}, X_4^{(17)})$. Concatenating (2) with itself up to 16.5 rounds results in the following linear relation with bias 2^{-1}, input mask $(1,0,1,0)$, and output mask $(1,0,1,0)$, after the OT:

$$(X_1^{(1)} \oplus X_3^{(1)} \oplus X_1^{(17)} \oplus X_3^{(17)}) \cdot 1 = (Z_1^{(1)} \oplus Z_1^{(2)} \oplus Z_5^{(2)} \oplus Z_5^{(3)} \oplus Z_1^{(4)} \oplus Z_1^{(5)} \oplus Z_5^{(5)} \oplus Z_5^{(6)} \oplus Z_1^{(7)} \oplus Z_1^{(8)} \oplus Z_5^{(8)} \oplus Z_5^{(9)} \oplus Z_1^{(10)} \oplus Z_1^{(11)} \oplus Z_5^{(11)} \oplus Z_5^{(12)} \oplus Z_1^{(13)} \oplus Z_1^{(14)} \oplus Z_5^{(14)} \oplus Z_5^{(15)} \oplus Z_1^{(16)} \oplus Z_1^{(17)}) \cdot 1 \,. \quad (3)$$

Relation (3) does not require any weak subkey restrictions, and can be used to distinguish 16.5-round IDEA-128 from a random permutation with high success rate, using about $8 \cdot (2^{-1})^{-2} = 32$ known plaintexts (KP) [17]:

- collect $N \approx 32$ KP, and compute the left-hand side of (3) for all the known plaintext-ciphertext pairs;
- since the right-hand side of (3) depends on the subkey bits only, and the key is assumed to be fixed for all KP, the left-hand side of (3) might give a constant bit value for IDEA-128, while it might give a binary random value for a random permutation. There is a probability of 2^{-N} that the result is a constant bit for a random permutation.

The above strategy also results in a key-recovery attack. Just collect the one bit of information about the key on the right-hand side of (3).

Neither KS_1 nor KS_2 are appropriate key schedules because they cannot avoid even subkey values that not relatively prime to 2^{32}. But, even for a key setup that could avoid these forbidden subkeys, the linear attack above would still work for any number of rounds.

Under a chosen-plaintext setting, differential analysis of this IDEA-128 variant using differential characteristics requires the assumption that some multiplicative subkeys have the (weak) value 1. The difference operator is $\oplus$, and $\delta = \texttt{80000000}_\texttt{x}$ is the only non-zero 32-bit wordwise difference used, because δ can propagate across addition, multiplication and exclusive-or with certainty. Table 2 lists the one-round characteristics for IDEA-128, and the corresponding weak subkeys. The best differential distinguisher from Table 2, is based on the following 3-round iterative characteristic $(0, \delta, 0, \delta) \xrightarrow{1r} (0, 0, \delta, \delta) \xrightarrow{1r} (0, \delta, \delta, 0) \xrightarrow{1r} (0, \delta, 0, \delta)$, because it has restrictions on only four subkeys: $Z_4^{(i)}$, $Z_4^{(i+1)}$, $Z_5^{(i+1)}$ and $Z_5^{(i+2)}$. Concatenated with itself up to 16.5 rounds, the input difference is $(0, \delta, 0, \delta)$, and the output difference is $(0, \delta, 0, \delta)$ after the OT, with only 22 restricted subkeys. Similarly, differential characteristics can be used either to distinguish IDEA-128 from a random permutation or to recover unknown key/subkey bits.

If KS_1 were used, and there were no forbidden (multiplicative) subkeys, then this differential distinguisher could be used to identify a weak-key class ($|WKC|$) of size 2^{24} i.e. only user key bits 220–243 could be arbitrary. A weak-key class is just a subset of the key space composed of keys that lead to weak subkeys through some key schedule. Under KS_2 we have $|WKC| = 0$ after only six rounds, assuming one weak subkey happens with probability 2^{-32}. But, neither KS_1 nor KS_2 avoid the non-invertible subkeys, and thus cannot guarantee correct decryption for an arbitrary key.

4. Multiplication in $\mathbb{Z}^*_{2^{32}+1}$

The first important fact about the multiplicative group $(\mathbb{Z}^*_{2^{32}+1}, \boxdot)$, where $\mathbb{Z}^*_{2^{32}+1} = \{x \in \mathbb{Z}_{2^{32}+1} | \gcd(x, 2^{32} + 1) = 1\}$, is that $2^{32} + 1 = 641 * 6700417 = 4294967297$, unlike in

IDEA, where $2^{16}+1$ is prime. Similar to IDEA, though, multiplication in $\mathbb{Z}^*_{2^{32}+1}$ assumes $0 \equiv 2^{32}$, so that all values fit into a 32-bit word. Also, all multiplications in this IDEA-128 variant, with group operation $\circledast = \boxdot$ in Fig. 1, involve one subkey as an operand, and such operation is not invertible for any subkey $Z_i^{(j)}$, unless $\gcd(Z_i^{(j)}, 2^{32}+1) = 1$, or equivalently, $\text{gdc}(Z_i^{(j)}, 641) = \gcd(Z_i^{(j)}, 6700417) = 1$. Consequently, any hypothetical key schedule for this IDEA-128 variant has to avoid all multiplicative subkeys multiples of 641 and 6700417, otherwise decryption would not be possible. Neither KS_1 nor KS_2 have any provision to avoid these multiplicative subkeys. Therefore, neither is an appropriate key setup algorithm for this IDEA-128 variant.

There are $\phi(2^{32}+1) = \phi(641) \cdot \phi(6700417) = 4288266240$ values in the range $[1, 2^{32}]$ that are relatively prime to $2^{32}+1$, where ϕ is Euler's totient function [2]. These valid subkeys account for a fraction of $\frac{4288266240}{2^{32}} \approx 99.84\%$ of all values in the range. The number of forbidden subkey values for $\mathbb{Z}^*_{2^{32}+1}$ is only $2^{32} - \phi(2^{32}+1) = 6701055$. To invert the valid multiplicative subkeys one can use the extended Euclidean GCD algorithm [19], or Euler's theorem. The former is faster. In the latter case simply raise the element to the power $\phi(2^{32}+1) - 1$ modulo $2^{32}+1$. For instance, $2^{-1} \equiv 2^{4288266239} \equiv 2147483649 \bmod (2^{32}+1)$ is the inverse of 2 modulo $2^{32}+1$.

Unlike $\mathbb{Z}^*_{2^{32}}$, the multiplication in $\mathbb{Z}^*_{2^{32}+1}$ has a wrap-around effect [16] such as in $\text{GF}(2^{16}+1)$, which means that the LSB of the multiplication result does not always depend exclusively on the LSBs of its operands. Nonetheless, this property does not avoid linear attacks altogether. Linear relations with bias 2^{-1} exploiting the LSB of round input and output words, across $\boxdot$, still exist for subkeys 2^{32} and 1. From Table 2, with $\boxdot$ as the multiplication operator, it is possible to build several linear relations, with bias 2^{-1} (maximum), across multiple rounds, under these weak multiplicative subkeys and with high bias, similar to [7]. These relations use the 3-round iterative relation described in Sect. 3.: $(1,0,1,0) \xrightarrow{1r} (1,1,0,0) \xrightarrow{1r} (0,1,1,0) \xrightarrow{1r} (1,0,1,0)$. The bias is 2^{-1} as long as the subkeys $Z_1^{(i)}, Z_1^{(i+1)}, Z_5^{(i+1)}, Z_5^{(i+2)}$ have value 2^{32} or 1, in rounds i to $i+2$. Concatenated with itself up to 16.5 rounds results in a linear relation with input mask $(1,0,1,0)$, output mask $(1,0,1,0)$ after the OT, under 22 weak subkeys. Therefore, this IDEA-128 variant with $\boxdot$ can be distinguished from a random permutation, with high success rate, using about $8 \cdot (2^{-1})^{-2} = 32$ KP [17], under some key schedule algorithm that avoids the forbidden subkeys. For both KS_1 and KS_2, though, these 22 subkey restrictions lead to $|WKC| = 0$, across 16.5 rounds. Anyway, neither KS_1 nor KS_2 are appropriate key schedule algorithms because neither can avoid the forbidden subkeys.

The differential analysis of this IDEA-128 variant is analogous to that of Sect. 3.: the differential characteristic covers 16.5 rounds and assumes 22 weak subkeys. The subkey values 1 and 2^{32} are weak because they turn the multiplication into the identity mapping, namely they make the other operand became a fixed point. Under KS_1, we expect $|WKC| = 2^{24}$ after 16.5 rounds. Under KS_2 we expect $|WKC| = 0$ after only six rounds.

[2] Curiously, 641 and 6700417 form a pair of RSA primes in an unbalanced RSA variant [24], namely one prime number is much smaller than the other: 641 is about nine bits long, and 6700417 is about 22 bits long. If the multiplication operation $X \boxdot Z \bmod (2^{32}+1)$ in IDEA-128 were substituted for an exponentiation, $X^Z \bmod (2^{32}+1)$, where $\gcd(Z, \phi(2^{32}+1)) = 1$, then, encryption and decryption of this operation could be accomplished as in RSA. For the latter, given Z, one can easily find the decryption exponent (or inverse subkey) d of Z such that $X^{Z \cdot d} \bmod (2^{32}+1) \equiv X$.

5. Multiplication in GF(2^{32})

Another IDEA-128 variant can use the finite field GF(2^{32}) = GF(2)[x]/$p(x)$, where $p(x)$ is a primitive polynomial of degree 32 over GF(2), for instance, $p(x) = x^{32} + x^{27} + x^{25} + x^{22} + x^{20} + x^{18} + x^{16} + x^{10} + x^9 + x^5 + x^4 + x^2 + 1$. The computational graph for this IDEA-128 variant uses $\circledast = \diamond$ as the group operation in Fig. 1.

Multiplication in GF(2^{32}) can be accomplished efficiently like the xtime operation in the AES [1], with polynomial representation in GF(2)[x]/$p(x)$. Multiplicative inverse subkeys can also be efficiently computed using Euclid's extended GCD algorithm for polynomials [18]. Most multiplicative subkeys are invertible, except for the zero subkey which is the only forbidden value (in the key-mixing half-round). This subkey exception is less serious than in Sect. 4., because any potential key schedule algorithm for IDEA-128 using GF(2)[x]/$p(x)$ would have to avoid only one multiplicative subkey value. Neither KS_1 nor KS_2, though, have any provisions to avoid this multiplicative subkey. This situation is analogous to a key schedule problem in the block cipher SHARK [22]. Nonetheless, this exception alone may prohibit the use of this IDEA-128 variant as a primitive in the hash function construction [19], particularly if in the latter arbitrary values can be input independently as subkeys without appropriate processing.

Disregarding the forbidden subkeys for a moment, a linear attack on IDEA-128 using $\diamond$, similar to the attack described in Sect. 3., would be effective only for the (weak) subkey value 1. In order to estimate the $|WKC|$ for KS_2, one can assume that KS_2 behaves as a pseudorandom number generator. Under this hypothesis, one can estimate that each 32-bit multiplicative subkey assume value 1 with probability about $\frac{1}{2^{32}-1}$, since the value 0 is forbidden. From Table 2, the 3-round iterative relation $(0,1,1,0) \xrightarrow{1r} (1,0,1,0) \xrightarrow{1r} (1,1,0,0) \xrightarrow{1r} (0,1,1,0)$ has only four subkey restrictions: $Z_5^{(i)}$, $Z_1^{(i+1)}$, $Z_1^{(i+2)}$, and $Z_5^{(i+2)}$, from rounds i to $i+2$. A 5-round distinguisher based on this relation would require only six weak subkeys. Assuming independence of each weak subkey, the estimated weak-key class size for a linear attack on 6-round IDEA-128, with this linear distinguisher is $2^{256} \cdot (\frac{1}{2^{32}-1})^6 \approx 2^{256-192} = 2^{64}$. For a 6-round distinguisher, the estimated $|WKC|$ is $2^{256} \cdot (\frac{1}{2^{32}-1})^8 \approx 1$. Under KS_1 we have $|WKC| = 2^3$ after 13 rounds, and $|WKC| = 0$ after 14 rounds.

The differential characteristics under weak-subkey assumptions for this IDEA-128 variant follow the same reasoning as in Sect. 4.

6. Multiplication in $\mathbb{Z}^*_{2^{32}-1}$

The group $(\mathbb{Z}^*_{2^{32}-1}, \otimes)$, where $\mathbb{Z}^*_{2^{32}-1} = \{x \in \mathbb{Z}_{2^{32}-1} | gcd(x, 2^{32}-1) = 1\}$ has already been used in the MMB block cipher [7, chap. 6]. In IDEA-128 this multiplicative group operation will be denoted with $\circledast = \otimes$ in Fig. 1. Notice that $2^{32} - 1 = 3 * 5 * 17 * 257 * 65537$ is the product of all known Fermat primes [8]. According to [7], $A \otimes B \bmod (2^{32}-1) = A * B \bmod 2^{32} + \lfloor \frac{A*B}{2^{32}} \rfloor$, where B is relatively prime to $2^{32}-1$.

The $\otimes$ operation is used in different ways in MMB and IDEA-128. In MMB, fixed 32-bit (invertible) constants are used as one of the operands to $\otimes$. In IDEA-128, a variable 32-bit subkey is used as one of the operands, and thus have to be relatively prime to $2^{32}-1$ in order for the multiplication to be invertible. This situation is similar to that with $\mathbb{Z}^*_{2^{32}+1}$: there are $\phi(2^{32}-1) = \phi(3) * \phi(5) * \phi(17) * \phi(257) * \phi(65537) = 2^{31}$ values relatively prime

to $2^{32} - 1$ in the range $[1, 2^{32} - 1]$, where ϕ is Euler's totient function. These elements account for a fraction of $\frac{2^{31}}{2^{32}} = \frac{1}{2}$ of all values in the range. To find the multiplicative inverse of the encryption subkeys, one can use Euler's theorem as in Sect. 4., or Euclid's algorithm [19]. One forbidden subkey for the multiplication in $\mathbb{Z}^*_{2^{32}-1}$ is zero, because it does not allow decryption when used in the key-mixing half-round, a phenomenon also observed in Sect. 3.. Moreover, $(2^{32} - 1) \otimes A = 2^{32} - 1$, where $A \in \mathbb{Z}_{2^{32}-1}$, namely, the multiplicative subkey $2^{32} - 1$ is the same as multiplication by 1 (a fixed point).

The same linear relations of Sect. 3. also apply to IDEA-128 with $\otimes$. The weak subkey values $\{1, 2^{32} - 1\}$ are not forbidden, and have bias 2^{-1} across $\otimes$ with masks involving only the LSB of the input and output words. Under KS_1, these weak subkeys are contradictory, because $1 = \texttt{00000001}_\texttt{x}$ and $2^{32} - 1 = \texttt{FFFFFFFF}_\texttt{x}$, namely, the first subkey requires the 31 MSBs to be zero, but the second weak subkey requires the same 31 MSBs to be one. Thus, under KS_1, but using only one weak subkey value, the expected $|WKC|$ are the same as in Sect. 4.. Using both weak subkey values, $|WKC| = 0$. For KS_2, the same reasoning as in Sect. 4. apply.

Differential cryptanalysis using characteristics in Table 2 applies similarly to this IDEA-128 variant but with weak subkey values $\{1, 2^{32} - 1\}$. The attack complexitities, and considerations about KS_1 and KS_2 in Sect. 3. hold accordingly.

7. Multiplication in GF(2^{32} + 15)

The finite field GF(2^{32} + 15) has already been used[3] in the Peanut98 block cipher [25]. In this IDEA-128 variant, the multiplication operation will be denoted with $\circledast = \boxtimes$ in Fig. 1. Notice that $2^{32} + 15 = 4294967311$ is a prime number, but in order to fit all values into 32 bits, a further reduction modulo 2^{32} is needed, so that $a \boxtimes b = (a * b \bmod (2^{32}+15)) \bmod 2^{32}$, $\forall a, b \in \text{GF}(2^{32} + 15)$. Even though $2^{32} + 15$ is prime, there can only be 2^{32} multiplicative subkeys, from 0 up to $2^{32} - 1$. Consequently, the subkey values $2^{32} + i$, for $0 \leq i \leq 14$, have to be discarded. Moreover, the subkeys which have these larger-than-32-bit values as multiplicative inverses (for decryption) represent a problem for the key schedule algorithm. For instance, $(2^{32})^{-1} \bmod (2^{32} + 15) \equiv 572662308$, so the inverse of subkey 572662308 is larger than 32 bits. Particularly, $(2^{32} + 14)^{-1} = 4294967310 = 2^{32} + 14$. Thus, even if the fifteen subkeys $2^{32}+i$, for $0 \leq i \leq 14$, were discarded for encryption, they could still be required for decryption, and the key schedule would need to make provisions for subkeys larger than 32 bits. These problems arise in the context of IDEA-128, because in Peanut98 operations involving GF($2^{32} + 15$) are not limited to a multiplication between a subkey and a data words.

The subkey 0 is a forbidden subkey since it is non-invertible. Another problem is related to the double modular reduction. Suppose two inputs a, b and a multiplicative subkey $Z_i^{(j)}$ such that $a \boxtimes Z_i^{(j)} = b \boxtimes Z_i^{(j)} = c$. An example is $Z_i^{(j)} = 11$, $a = 3123612591$ which results in $a \boxtimes Z_i^{(j)} = 13$ (one modular reduction), and $b = 1171354721$ which leads to $b \boxtimes Z_i^{(j)} = 2^{32} + 13 \equiv 13$ (two modular reductions). This means that all multiplicative subkeys have equivalent subkeys, depending on the other data to multiplication. Moreover, depending on the key schedule algorithm, there may be equivalent keys, namely, different user keys leading to the same encryption transformation. This situation can happen either

[3]The Peanut97 cipher uses the prime number $2^{32} - 5$, while the DFC cipher [9] uses the prime number $2^{64} + 13$.

in the key-mixing or in the MA half-round. It has not been determined yet if KS_1 or KS_2 can generate equivalent keys for this IDEA-128 variant.

On the other way around suppose, for example, two subkeys $Z_i = 3123612591$, $Z_j = 1171354721$, and a 32-bit data word $a = 11$. Then, $a \boxtimes Z_i = a \boxtimes Z_j = 13$, namely, an operation in the key-mixing half-round. Thus, this particular decryption can be accomplished in two different ways (equivalent decryption subkeys).

Disregarding the double-modular-reduction problem for a moment, both linear and differential attacks on this IDEA-128 variant would still depend on the key schedule algorithm to guarantee the existence of weak multiplicative subkeys. Assuming (weak) multiplicative subkeys with value 1, the same linear relations and characteristics of Sect. 4. apply to this IDEA-128 variant.

8. Conclusions

This paper described five hypothetical realizations of 128-bit block variants of the IDEA cipher [16], and two examples of key schedule algorithms. The parameters for IDEA-128 are exactly double the size of IDEA's. Ciphers with 128-bit blocks are useful for instance as a buiding block for the construction of other cryptographic primitives such as hash functions, stream ciphers and MACs [19, p. 229,340,353]. Moreover, 128-bit block size is commonplace among modern block ciphers [1, 21], and help defeat some drawbacks inherent to 64-bit block ciphers [13].

The main problem is constructing a 128-bit block cipher variant of IDEA is to find an appropriate multiplicative group on 32-bit words to replace $GF(2^{16} + 1)$ used in IDEA. This paper analysed five algebraic group candidates. But, it does not mean that there are no other alternatives. It is left as an open problem whether there are any other suitable algebraic groups on 32-bit words that would allow a secure IDEA-128 cipher. Moreover, even though operations on 32-bit words seem attractive for modern desktop processors such as Intel's Pentium and AMD's Athlon, the number of multiplication operations greatly increases in these IDEA-128 variants (Table 4), making them even slower than IDEA.

Two key schedule algorithms for IDEA-128 were suggested: one of them, KS_1, is an extension of the original key schedule of IDEA. The other, KS_2, is based on the key setup of the MESH ciphers [20]. KS_2 is more complex, but does not have the key overlapping property of KS_1, due to the recurrence relation (1). The aim is to avoid bit patterns in the key to propagate to the subkeys, a behavior that may lead to weak subkeys. It may seem that KS_1 and KS_2 were too poorly designed to avoid weak and forbidden subkeys in IDEA-128, but it must be understood that some attacks work independent of the key scheduke algorithm; and further, it is not the responsibility of a key setup algorithm to repair weaknesses in the encryption and decryption schemes. Nonetheless, it is important to design key setup algorithms that simultaneously: (i) avoid weak, equivalent and forbidden subkeys; (ii) key agile, i.e. significantly faster than one encryption operation; (iii) are immune to manipulation by an adversary (for example, in some hash function constructions [19, p. 340], or in related-key attacks [12]).

The first problem with all the five groups is related to multiplicative subkeys for which decryption could not be accomplished because of non-invertible subkeys. These forbidden subkeys are unacceptable. Their number depends on the algebraic group. One

Table 3. Estimated weak-key class size and number of rounds for DC and LC attacks.

Group		$\mathbb{Z}^*_{2^{32}}$	$\mathbb{Z}^*_{2^{32}+1}$	GF(2^{32})	$\mathbb{Z}^*_{2^{32}-1}$	GF($2^{32}+15$)
Weak Subkeys		$\{1\}$	$\{1, 2^{32}\}$	$\{1\}$	$\{1, 2^{32}-1\}$	$\{1\}$
LC	KS_1	$(2^{256}; \infty)$	(0; 14)	(0;14)	(0;14) ‡	(0;14)
	KS_2	$(2^{256}; \infty)$	(0; 6)	(1;6)	(0;6) ‡	(0;6)
DC	KS_1	$(2^{24}; 16.5)$	$(2^{24}; 16.5)$	$(2^{24}; 16.5)$	$(2^{24}; 16.5)$ ‡	$(2^{24}; 16.5)$
	KS_2	(0; 7)	(0;6)	(0;6)	(0;6) ‡	(0;6)

‡: WKC for one weak subkey only, not both at once.

solution would be for the key schedule to avoid these subkeys, but this procedure could reduce the subkey space considerably or imply long delays (performance loss) during subkey generation, such as in the MARS block cipher [6]. Further, it is left as an open problem whether these exception handlings would be exploitable by timing attacks [15]. Ignoring the decryption problem for a moment, linear and differential attacks were also considered for the IDEA-128 variants. The linear attacks described in Sect. 3., with multiplication in $(\mathbb{Z}^*_{2^{32}}, *)$, can either distinguish IDEA-128 from a random permutation, or recover information on the subkeys. The attacks apply not only to 16.5-round IDEA-128, but to any number of rounds. The attack complexity is only 32 KP, and equivalent encryption effort. The corresponding differential attack could identify a weak-key class of 2^{24} keys for 16.5 rounds, under KS_1. Under KS_2, weak-key classes are expected for no more than seven rounds.

For the IDEA-128 variant using multiplication in $(\mathbb{Z}^*_{2^{32}+1}, \boxdot)$, any key schedule algorithm would need to avoid multiplicative subkeys that are multiples of 641 and 6700417, in order to allow proper decryption. Under this assumption, less effective linear and differential attacks apply, compared to $(\mathbb{Z}^*_{2^{32}}, *)$ (see Table. 3).

The IDEA-128 variant using multiplication in GF(2^{32}) has at least one weak subkey, 1, and one forbidden subkey, 0 (when used in the key-mixing half-round). These exceptions compare favorably with the 2^{31} forbidden subkeys, and 2^{32} weak subkeys in the IDEA-128 in Sect. 3.. Nonetheless, the weak subkey value still allows linear and differential attacks under weak-subkey assumptions, depending on the key schedule algorithm (Table. 3).

The linear and differential distinguishers for IDEA-128, with multiplication in $(\mathbb{Z}^*_{2^{32}-1}, \otimes)$, were the same as those in Sect 3.. The weak (multiplicative) subkey values are $\{1, 2^{32}-1\}$ for the $\otimes$ operation. The $|WKC|$ and the considerations about KS_1 and KS_2 are summarized in Table. 3.

The linear and differential attacks of Sect. 7., with multiplication in (GF($2^{32}+15$), $\boxtimes$) work similarly to those in Sect 6.. The weak subkey value is 1, and the $|WKC|$ and the considerations about KS_1 and KS_2 are in Table. 3. The results in this Table assume that the presence of forbidden subkeys, or other algebraic problems in the cipher will not affect the effectiveness of attacks.

The overall conclusion concerning the five realizations of IDEA-128 is that none of them constitute a sound cipher design. Additionally, they help corroborate the rationale for the MESH designs [20], that used the same group operations on 16-bit words as the IDEA cipher, but achieved a relatively high level of security without the need to double the word size to 32 bits, and without forbidden nor equivalent subkeys.

Table 4. Comparing parameters of IDEA, MESH-64, and IDEA-128.

Cipher	IDEA	MESH-64	IDEA-128	IDEA-128	IDEA-128	IDEA-128	IDEA-128
Group	$GF(2^{16}+1)$	$GF(2^{16}+1)$	$\mathbb{Z}^*_{2^{32}}$	$\mathbb{Z}^*_{2^{32}+1}$	$GF(2^{32})$	$\mathbb{Z}^*_{2^{32}-1}$	$GF(2^{32}+15)$
Notation Mult.	$\odot$	$\odot$	$*$	$\boxdot$	$\diamond$	$\otimes$	$\boxtimes$
Word Size (bits)	16	16	32	32	32	32	32
Structure	Finite Field	Finite Field	Group	Group	Finite Field	Group	Finite Field
Group Origin	Fermat prime	Fermat prime	MARS	IDEA	SHARK	MMB	Peanut98
# Invertible Subkeys	2^{16}	2^{16}	$\phi(2^{32})$	$\phi(2^{32}+1)$	$2^{32}-1$	$\phi(2^{32}-1)$	$\phi(2^{32}+15)$
Block Size (bits)	64	64	128	128	128	128	128
Key Size (bits)	128	128	256	256	256	256	256
# Rounds	8.5	8.5	16.5	16.5	16.5	16.5	16.5
# Mult. Oper. (†)	34	42	66	66	66	66	66
# $\boxplus$ Oper. (†)	34	42	66	66	66	66	66
# $\oplus$ Oper. (†)	48	48	96	96	96	96	96
Reference	[16]	[20]	this paper				

†: only encryption and decryption, i.e. not including the key schedule.

Table 4 compares IDEA and MESH-64 with the five IDEA-128 variants.

Acknowledgements

Many thanks to Jasper Scholten for his patience explaining the algebraic structure of $\mathbb{Z}^*_{2^{32}+1}$, which motivated this research. The many useful comments from anonymous referees, past and present, are also very much appreciated.

References

[1] AES: The Advanced Encryption Standard Development Process, 1997, *http://csrc.nist.gov/encryption/aes/*.

[2] Álvarez,G., de la Guia,D., Montoya,F., Peinado,A.: Akelarre: a new Block Cipher Algorithm, 3rd Selected Areas in Cryptography (SAC) Workshop, 1996, 1–14.

[3] Biham,E.: New Types of Cryptanalytic Attacks using Related Keys, Adv. in Cryptology, Eurocrypt'93, T. Helleseth, Ed., Springer-Verlag, LNCS 765, 1994, 398–409.

[4] Biham,E., Biryukov,A., Shamir,A.: Cryptanalysis of Skipjack Reduced to 31 Rounds using Impossible Differentials, Technion, CS Dept., 1998, Tech Report CS0947 revised.

[5] Biryukov,A., Nakahara Jr,J., Preneel,B., Vandewalle,J.: New Weak-Key Classes of IDEA, ICICS 2002, R. Deng, S. Qing, F. Bao, J. Zhou,Eds., Springer-Verlag, LNCS 2513, Dec, 2002, 315–326.

[6] Burwick,C., Coppersmith,D., D'Avignon,E., Genario,R., Halevi,S., Jutla,C., Matyas Jr,S.M., O'Connor,L., Peyravian,M., Safford,D., Zunic,N.: MARS – a Candidate Cipher for AES, 1st AES Conference, California, USA, Jun, 1998, *http://csrc.nist.gov/encryption/aes/*.

[7] Daemen,J.: Cipher and Hash Function Design – Strategies based on Linear and Differential Cryptanalysis, COSIC group, Dept. Elektrotechniek, Katholieke Universiteit Leuven, Belgium, Mar. 1995.

[8] Fermat Primes website, *http://www.prothsearch.net/fermat.html*

[9] Gilbert,H., Girault,M., Hoogvorst,P., Noilhan,F., Pornin,T., Poupard,G., Stern,J., Vaudenay,S.: Decorrelated Fast Cipher: an AES candidate, 1st AES Conference, California, USA, 1998, Aug, *http://csrc.nist.gov/encryption/aes/*

[10] Hawkes,P.M.: Asymptotic Bounds on Differential Probabilities and an Analysis of the Block Cipher IDEA, The University of Queensland, St. Lucia, Australia, Dec, 1998.

[11] Howgrave-Graham,N., Nguyen,P., Pointcheval,D., Proos,J.A., Silverman,J.H., Singer,A., Whyte,W.: The Impact of Decryption Failures on the Security of NTRU Encryption, Adv. in Cryptology, Crypto'2003, D. Boneh, Ed., Springer-Verlag, LNCS 2729, 226-246.

[12] Kelsey,J., Schneier.B., Wagner,D.: Related-Key Cryptanalysis of 3-Way, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA, ICICS 1997, Springer-Verlag, Nov.1997, 233–246.

[13] Knudsen,L.R.: Block Ciphers – A Survey, State of the Art in Applied Cryptography, B. Preneel, V. Rijmen, Eds., Springer-Verlag, LNCS 1528, 1998, 18–48.

[14] Knudsen,L.R., Rijmen,V.: Ciphertext-Only Attack on Akelarre, Cryptologia, vol.XXIV, n.2, Apr, 2000, 135–147.

[15] Kocher,P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, Adv. in Cryptology, Crypto'96, N. Koblitz,Ed., Springer-Verlag, LNCS 1109, 1996, 104–113.

[16] Lai,X.: On the Design and Security of Block Ciphers, ETH Series in Information Processing, J.L. Massey,Ed., vol. 1, 1995, Hartung-Gorre Verlag, Konstanz.

[17] Matsui,M.: Linear Cryptanalysis Method for DES Cipher, Adv. in Cryptology, Eurocrypt'93, T. Helleseth, Ed., Springer-Verlag, LNCS 765, 1994, 386–397.

[18] McEliece, R.J.,: Finite Fields for Computer Scientists and Engineers, Kluwer Academic Publishers, 1987.

[19] Menezes,A.J., van Oorschot,P.C., Vanstone,S.: Handbook of Applied Cryptography, CRC Press, 1997.

[20] Nakahara Jr,J., Rijmen,V., Preneel,B., Vandewalle,J.: The MESH Block Ciphers, The 4th International Workshop on Info. Security Applications, WISA 2003, K. Chae, M. Yung, Eds., Springer-Verlag, LNCS 2908, 2003, 458–473.

[21] NESSIE: New European Schemes for Signatures, Integrity and Encryption, Jan, 2000, http://cryptonessie.org

[22] Rijmen,V., Daemen,J., Preneel,B., Bosselaers,A., De Win,E.: The Cipher SHARK, 3rd Fast Software Encryption Workshop, D. Gollmann,Ed., Springer-Verlag, LNCS 1039, 1996, 99–112.

[23] Rijmen,V., Preneel,B., De Win,E.: On Weaknesses of Non-Surjective Round Functions, Design, Codes and Cryptography, vol. 12, no. 3, Nov, 1997, 253–266.

[24] Shamir,A.: RSA for Paranoids, RSA Laboratories CryptoBytes (1):3, 1995, 1–4.

[25] Vaudenay,S.: Provable Security for Block Ciphers by Decorrelation, STACS´98, Paris, France, LNCS 1373, Springer-Verlag, 1998, 249–275.