The MESH Block Ciphers

Jorge Nakahara Jr*, Vincent Rijmen**, Bart Preneel, Joos Vandewalle

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC, Belgium
{jorge.nakahara,bart.preneel,joos.vandewalle}@esat.kuleuven.ac.be
Cryptomathic, Belgium & Graz University of Technology, Austria
vincent.rijmen@cryptomathic.com

Abstract. This paper describes the MESH block ciphers, whose designs are based on the same group operations as the IDEA cipher, but with a number of novel features: flexible block sizes in steps of 32 bits (the block size of IDEA is fixed at 64 bits); larger MA-boxes; distinct key-mixing layers for odd and even rounds; and new key schedule algorithms that achieve fast avalanche and avoid the weak keys of IDEA. The software performance of MESH ciphers are estimated to be better or comparable to that of triple-DES. A number of attacks, such as truncated and impossible differentials, linear and Demirci's attack, shows that more resources are required on the MESH ciphers than for IDEA, and indicates that both ciphers seem to have a large margin of security.

Keywords: secret-key block ciphers, design and cryptanalysis, IDEA, MA-boxes.

1 Introduction

This paper presents the MESH block ciphers, whose designs are based on the same group operations on 16-bit words as the IDEA cipher [11], namely, bitwise exclusive-or, denoted $\oplus$, addition in $\mathbb{Z}_{2^{16}}$, denoted $\boxplus$, and multiplication in $\mathrm{GF}(2^{16}+1)$, denoted $\odot$, with the value 0 denoting 2^{16}. The MESH designs are built on the strength of IDEA, but include some novel features: (i) flexible block sizes in increments of 32 bits; (ii) larger MA-boxes; (iii) distinct key-mixing layers for odd and even rounds; (iv) new key schedule algorithms. This paper is organized as follows: Sect. 2 provides motivation for the new cipher designs; Sect. 3 describes MESH-64; Sect. 4 describes MESH-96, and Sect. 5 describes MESH-128. Sect. 6, 7, 8 and 9 describe attacks on the MESH ciphers. Sect. 10 discusses the software performance. Sect. 11 concludes the paper.

2 Design Rationale and Motivations

Since the publication of IDEA in [11], no extended IDEA variant has being proposed with block sizes larger than 64 bits (or word sizes larger than 16 bits).

* Sponsored by a grant from the Katholieke Universiteit Leuven, and partially by GOA project Mefisto 2000/06 of the Flemish Government.
** An abridged version was presented at WISA 2003, Jeju Island, Korea

Maybe such attempts were jeopardized due to the fact that $2^{32}+1$ is not a prime number,[1] and thus, $\mathbb{Z}^*_{2^{32}+1}$ is not a finite field [12, p. 77, Fact 2.184]. The MESH designs provide an alternative approach that does not rely on the need for larger word sizes. This motivates the design of larger MA-boxes. All MA-boxes in the MESH ciphers involve at least three interleaved layers of multiplication and addition operations in a zig-zag pattern, in comparison to two layers in IDEA. The MA-boxes of some MESH ciphers have the property that not all multiplications involve subkeys directly as an operand, but rather depend upon internal data values. These designs effectively avoid many one-round linear relations and one-round characteristics (to be discussed further). All the new MA-boxes are bijective mappings (permutations), in order to avoid non-surjective attacks [13]).

Another feature of the MESH ciphers is the key schedule algorithm. Note that in IDEA all multiplications involve a subkey as an operand. Since the modular multiplication is the main non-linear operator in the cipher, the key schedule needs to be designed to avoid weak subkeys for any choice of the user key, otherwise, all multiplications could, in principle, be manipulated (Daemen [5]). The following design principles were used in the key schedule of MESH ciphers to avoid weak keys:

- fast key avalanche: each subkey generated from the user key quickly depends, non-linearly, upon all user key words. This dependence is expressed by the exponents of a primitive polynomial (one polynomial for each MESH cipher). All key schedule algorithms interleave addition with exclusive-or operations. There is additionally a fixed bit-rotation operation, because in both $\boxplus$ and $\oplus$ the relative position of the subkey bits is preserved and otherwise, two related keys with subkeys differing only in the most significant bit could propagate this difference to several other subkeys.
- use of fixed constants to avoid patterns in subkeys. For instance, without the constants the user-defined key with all-zero words would result in all subkeys being zero (independent of the non-linear mixing or the bit rotation) for any number of rounds.

Common properties to IDEA and MESH ciphers include: (i) complete diffusion is achieved in one round; (ii) no operation is used twice in succession in any part of these ciphers; (iii) neither cipher uses explicit S-boxes, nor depend on particular properties of Boolean functions such as in Camellia [1] or AES [8].

Three designs will be described: MESH-64, MESH-96 and MESH-128, where the suffix denotes the block size.

3 The MESH-64 Block Cipher

MESH-64 is a 64-bit block cipher with a 128-bit key and 8.5 rounds (Fig. 1 and Table 1). The last 0.5 round is the output transformation (OT). The key schedule for MESH-64 is defined as follows:

[1] $2^{32} + 1 = 4294967297 = 641 \cdot 6700417$.

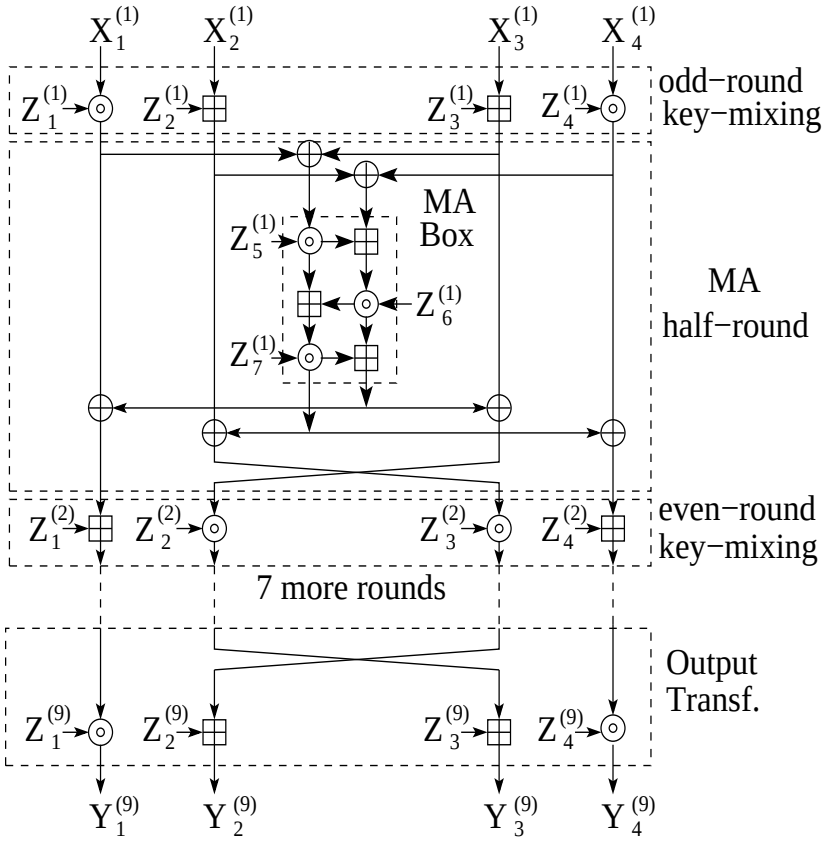

Fig. 1. Computational graph of the MESH-64 cipher.

- First, 16-bit constants c_i are defined as: $c_0 = 1$, and $c_i = 3 \cdot c_{i-1}$, $i \geq 1$ with multiplication in $\mathrm{GF}(2)[x]/p(x)$, under the primitive polynomial $p(x) = x^{16} + x^5 + x^3 + x^2 + 1$. The constant '3' is represented by the polynomial $x + 1$ in $\mathrm{GF}(2)$.
- The 128-bit user key is partitioned into eight 16-bit words K_i, $0 \leq i \leq 7$, and assigned to $Z^{(1)}_{j+1} = K_j \oplus c_j$, $0 \leq j \leq 6$, and $Z^{(2)}_1 = K_7 \oplus c_7$.
- Finally, each subsequent 16-bit subkey is defined as follows:

$$Z^{(h(i))}_{l(i)} = (((((Z^{(h(i-8))}_{l(i-8)} \boxplus Z^{(h(i-7))}_{l(i-7)}) \oplus Z^{(h(i-6))}_{l(i-6)}) \boxplus Z^{(h(i-3))}_{l(i-3)}) \oplus Z^{(h(i-2))}_{l(i-2)}) \boxplus Z^{(h(i-1))}_{l(i-1)}) \lll 7 \oplus c_i, \quad (1)$$

  for $8 \leq i \leq 59$; '$\lll$ 7' is left rotation by 7 bits; $h(i) = i \text{ div } 7 + 1$, and $l(i) = i \bmod 7 + 1$.

The key schedules of MESH-64 is designed to achieve fast key avalanche, due to (1) being based on the primitive polynomial $q(x) = x^8 + x^7 + x^6 + x^5 +$

$x^2 + x + 1$ in GF(2), and the interleaving of $\oplus$ and $\boxplus$ operations. For instance, $Z_4^{(2)}$ and all subsequent subkeys already depend upon all eight user key words. The dependence of (1) on $q(x)$ can be made clear by ignoring the left rotation for a while, changing the $\boxplus$ to $\oplus$, and denoting $Z_{l(i)}^{(h(i))}$ simply as $Z^{(i)}$. Then (1) becomes $Z^{(i)} = Z^{(i-8)} \oplus Z^{(i-7)} \oplus Z^{(i-6)} \oplus Z^{(i-3)} \oplus Z^{(i-2)} \oplus Z^{(i-1)} \oplus c_i$. A similar reasoning applies to the other MESH ciphers. Notice that both $\boxplus$ and $\oplus$ preserve the relative bit position of its operands. The left-rotation destroys that property, so that changes only at the most significant bit of some subkeys (in a differential related-key attack) would not propagate to other subkeys with probability one. Without the constants, the all-zero user key would result in all subkeys being zero (for any number of rounds), independent of the mixing of addition and exclusive-or. Decryption in MESH-64 uses the same framework in Fig. 1 as encryption, but with transformed round subkeys. Formally, let the r-th round encryption subkeys be denoted $(Z_1^{(r)}, \ldots, Z_7^{(r)})$, for $1 \le r \le 8$, and $(Z_1^{(9)}, \ldots, Z_4^{(9)})$, for the OT. The decryption round subkeys are:

. $((Z_1^{(9)})^{-1}, -Z_2^{(9)}, -Z_3^{(9)}, (Z_4^{(9)})^{-1}, Z_5^{(8)}, Z_6^{(8)}, Z_7^{(8)})$, for the first round.
. $(-Z_1^{(10-r)}, (Z_3^{(10-r)})^{-1}, (Z_2^{(10-r)})^{-1}, -Z_4^{(10-r)}, Z_5^{(9-r)}, Z_6^{(9-r)}, Z_7^{(9-r)})$, for the r-th even round, $r \in \{2, 4, 6, 8\}$.
. $((Z_1^{(9-r)})^{-1}, -Z_3^{(9-r)}, -Z_2^{(9-r)}, (Z_4^{(9-r)})^{-1}, Z_5^{(8-r)}, Z_6^{(8-r)}, Z_7^{(8-r)})$, for the r-th odd round, $r \in \{3, 5, 7\}$.
. $((Z_1^{(1)})^{-1}, -Z_2^{(1)}, -Z_3^{(1)}, (Z_4^{(1)})^{-1})$, for the OT.

A similar procedure applies to the decryption subkeys of the other MESH ciphers.

4 The MESH-96 Block Cipher

MESH-96 is a 96-bit block cipher, with a 192-bit key, and 10.5 rounds (Fig. 2 and Table 1). The last 0.5 round is the output transformation (OT).

The key schedule for MESH-96 is defined as follows:

- The 16-bit constants c_i are the same as defined for MESH-64.
- The 192-bit user key is partitioned into twelve 16-bit words K_i, for $0 \le i \le 11$, that are assigned to: $Z_{j+1}^{(1)} = K_j \oplus c_j$, for $0 \le j \le 8$, $Z_1^{(2)} = K_9 \oplus c_9$, $Z_2^{(2)} = K_{10} \oplus c_{10}$, and $Z_3^{(2)} = K_{11} \oplus c_{11}$.
- Finally, each subsequent 16-bit subkey is defined as follows:

$$Z_{l(i)}^{(h(i))} = (((((Z_{l(i-12)}^{(h(i-12))} \boxplus Z_{l(i-8)}^{(h(i-8))}) \oplus Z_{l(i-6)}^{(h(i-6))}) \boxplus Z_{l(i-4)}^{(h(i-4))}) \oplus Z_{l(i-2)}^{(h(i-2))}) \boxplus Z_{l(i-1)}^{(h(i-1))}) \lll 9 \oplus c_i, \quad (2)$$

for $12 \le i \le 95$, '$\lll 9$' is left rotation by 9 bits, $h(i) = i \text{ div } 9 + 1$, and $l(i) = i \bmod 9 + 1$.

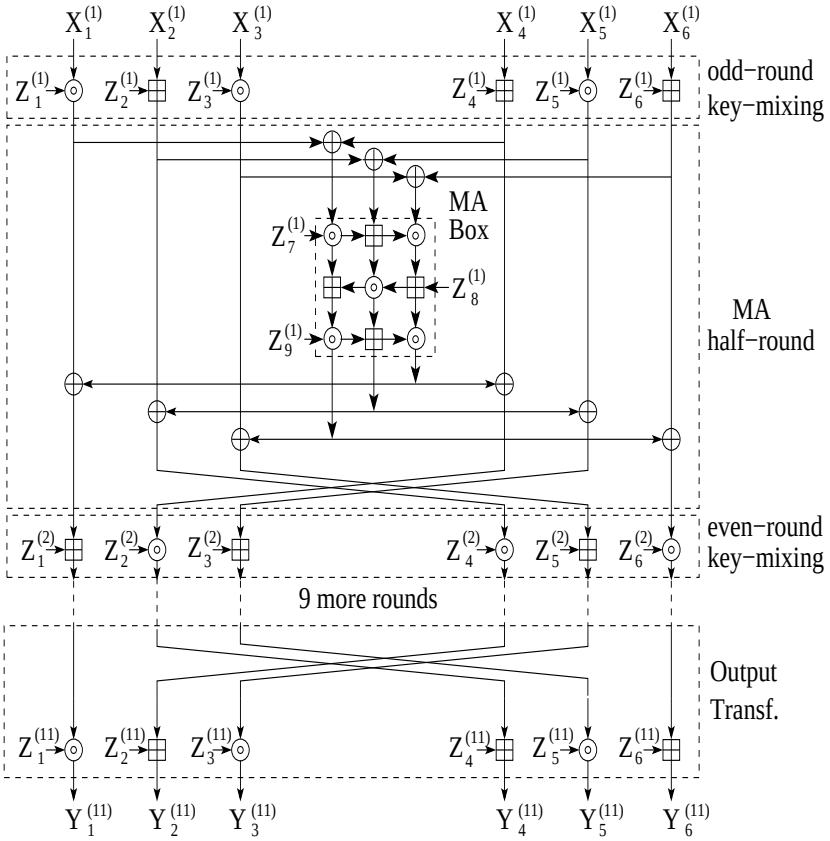

Fig. 2. Computational graph of the MESH-96 cipher.

The key schedule of MESH-96 is designed to achieve fast key avalanche due to the use of the primitive polynomial $x^{12}+x^{11}+x^{10}+x^8+x^6+x^4+1$ in GF(2), and the mixing of $\boxplus$ and $\oplus$ operations. All subkeys starting with $Z_7^{(2)}$, already depend (non-linearly) on all user key words. Decryption uses the same computational framework as in Fig. 2 for encryption, but with transformed subkeys.

5 The MESH-128 Block Cipher

MESH-128 is a 128-bit block cipher, with a 256-bit key, and 12.5 rounds (Fig. 3 and Table 1). The last 0.5 round is the output transformation (OT). The key schedule for MESH-128 is defined as follows:

- First, 16-bit constants c_i are defined as in MESH-64.
- Next, the 256-bit user key is partitioned into sixteen 16-bit words K_i, $0 \le i \le 15$, and are assigned to $Z_{j+1}^{(1)} = K_j \oplus c_j$, $0 \le j \le 11$, and $Z_{t \bmod 12+1}^{(2)} = K_t \oplus c_t$, $12 \le t \le 15$.

– Finally, each subsequent 16-bit subkey is generated as follows:

$$Z_{l(i)}^{(h(i))} = (((((Z_{l(i-16)}^{(h(i-16))} \boxplus Z_{l(i-13)}^{(h(i-13))}) \oplus Z_{l(i-12)}^{(h(i-12))}) \boxplus Z_{l(i-8)}^{(h(i-8))}) \oplus Z_{l(i-2)}^{(h(i-2))}) \boxplus Z_{l(i-1)}^{(h(i-1))}) \lll 11 \oplus c_i, \quad (3)$$

for $16 \le i \le 151$; '$\lll 11$' is left rotation by 11 bits; $h(i) = i \text{ div } 12 + 1$, and $l(i) = i \bmod 12 + 1$.

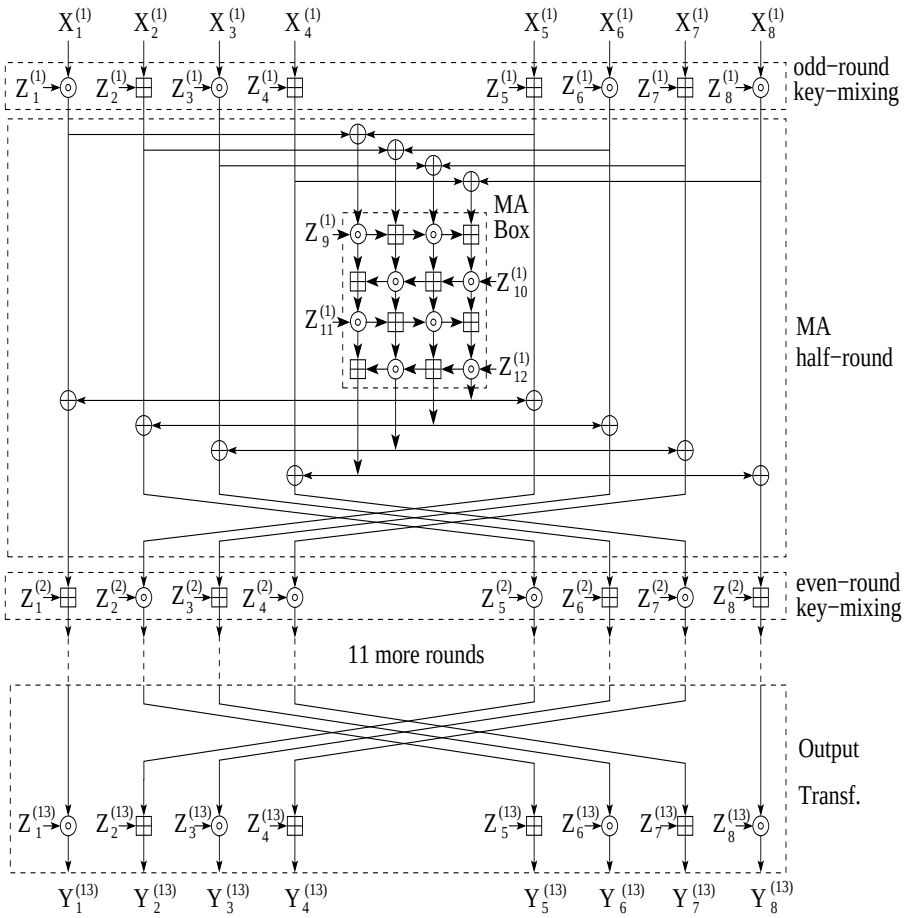

Fig. 3. Computational graph of the MESH-128 block cipher.

The key schedule of MESH-128 achieves fast key avalanche due to the use of the primitive polynomial $r(x) = x^{16} + x^{15} + x^{14} + x^8 + x^4 + x^3 + 1$, and the mixing of $\boxplus$ and $\oplus$ operations. All subkeys starting with $Z_{10}^{(2)}$, already depend upon all sixteen user key words. Decryption in MESH-128 uses the same framework as in Fig. 3, but with transformed subkeys.

Table 1 lists the main parameters for IDEA, and some MESH ciphers.

Table 1. Main parameters for IDEA and some MESH ciphers.

Cipher	Block Size	Key Size	#Rounds	#Operations			#Subkeys	
				$\boxplus$	$\oplus$	$\odot$	$\odot$	$\boxplus$
IDEA	64	128	8.5	34	48	34	34	18
MESH-64	64	128	8.5	42	48	42	42	18
MESH-96	96	192	10.5	73	90	83	53	43
MESH-128	128	256	12.5	148	144	148	100	52

6 Truncated Differential Analysis

Differential analysis of MESH ciphers followed the framework of Borst *et al.* [4]. The difference operator is bitwise exclusive-or.

6.1 Truncated Differential Attack on MESH-64

The truncated differential (4), with $A, B, C, D, E, F, G, H \in \mathbb{Z}_2^{16}$, is used in an attack on 3.5-round MESH-64.

$$\begin{aligned}
(A,0,B,0) &\overset{2^{-16}}{\rightarrow} (C,0,C,0) \overset{(0,0)\overset{1}{\rightarrow}(0,0)}{\rightarrow} (C,C,0,0)\\
(C,C,0,0) &\overset{1}{\rightarrow} (D,E,0,0) \overset{(D,E)\overset{2^{-32}}{\rightarrow}(E,D)}{\rightarrow} (0,D,0,E)\\
(0,D,0,E) &\overset{2^{-16}}{\rightarrow} (0,F,0,F) \overset{(0,0)\overset{1}{\rightarrow}(0,0)}{\rightarrow} (0,0,F,F)\\
(0,0,F,F) &\overset{1}{\rightarrow} (0,0,G,H)\,. \qquad (4)
\end{aligned}$$

In each line of (4) the leftmost arrow indicates that the 4-word difference on the left-hand side causes the difference in the middle after a key-mixing half-round, with the indicated probability on top of the arrow. Further, the middle 4-word difference causes the round output difference (on the right-hand side) across an MA-box, with the indicated probability. The truncated differential (4) has average probability 2^{-64} over all keys. A similar truncated differential, (10), is listed in the Appendix. From (4), the attack recovers subkeys $Z_1^{(1)}$, $Z_3^{(1)}$, $Z_3^{(4)}$, and $Z_4^{(4)}$ that satisfy equations $(P_1 \odot Z_1^{(1)}) \oplus (P_1^* \odot Z_1^{(1)}) = (P_3 \boxplus Z_3^{(1)}) \oplus (P_3^* \boxplus Z_3^{(1)})$, and $(C_3 \odot (Z_3^{(4)})^{-1}) \oplus (C_3^* \odot (Z_3^{(4)})^{-1}) = (C_4 \boxminus Z_4^{(4)}) \oplus (C_4^* \boxminus Z_4^{(4)})$. Estimates for the complexity of truncated differential attacks on MESH-64, using (4) are based on experimental results on mini-MESH variants, following [4], and give[2] $2^{32} \cdot 2^{31} \cdot 2 \cdot 2^{16} \cdot 2^{-3} \approx 2^{77}$ 3.5-round MESH-64 encryptions, 2^{64} chosen plaintexts, and 2^{32} 64-bit blocks (memory). Note that the new key schedule does not allow key bit overlap. Therefore, there is no reduction in complexity for a key-recovery attack. An additional differential, (4), allows to recover $Z_2^{(1)}$, $Z_4^{(1)}$, $Z_1^{(4)}$, and $Z_2^{(4)}$, with the same complexities.

[2] (#structures)×(#surviving pairs per structure)×(#equations)×(#key pairs to find per equation)× (#operations).

6.2 Truncated Differential Attack on MESH-96

A truncated differential attack on 3.5-round MESH-96 can use the following differential, where A, B, C, D, E, F, G, H, I, J, K, $L \in \mathbb{Z}_2^{16}$:

$$\begin{aligned}
(A, B, 0, C, D, 0) &\xrightarrow{2^{-32}} (E, F, 0, E, F, 0) \xrightarrow{(0,0,0)\to(0,0,0)} (E, E, F, F, 0, 0) \\
(E, E, F, F, 0, 0) &\xrightarrow{2^{-16}} (G, H, I, G, 0, 0) \xrightarrow{(0,H,I)\to(I,H,G)} (0, 0, H, 0, 0, I) \\
(0, 0, H, 0, 0, I) &\xrightarrow{2^{-16}} (0, 0, J, 0, 0, J) \xrightarrow{(0,0,0)\to(0,0,0)} (0, 0, 0, 0, J, J) \\
(0, 0, 0, 0, J, J) &\xrightarrow{1} (0, 0, 0, 0, K, L)\,, \qquad (5)
\end{aligned}$$

that has average probability 2^{-112}, and allows recovery of $Z_1^{(1)}$, $Z_2^{(1)}$, $Z_4^{(1)}$, $Z_5^{(1)}$, $Z_5^{(4)}$, and $Z_6^{(4)}$. A differential attack on MESH-96 using (5) works similarly to the attack on MESH-64, and has estimated complexity, based on experimental results following [4], as follows: $2^{32} \cdot 2^{63} \cdot 3 \cdot 2^{16} \cdot \frac{1}{13} \approx 2^{109}$ 3.5-round MESH-96 encryptions, 2^{96} chosen plaintexts, and 2^{64} 96-bit blocks of memory. There are two additional truncated differentials with the same probability, (11) and (12), listed in the Appendix, that allows to recover $Z_3^{(1)}$, $Z_6^{(1)}$, $Z_3^{(4)}$, $Z_4^{(4)}$, $Z_1^{(4)}$, and $Z_2^{(4)}$.

An attack on 4-round MESH-96 can guess subkeys $Z_7^{(4)}$, $Z_8^{(4)}$, $Z_9^{(4)}$, and apply the previous attack on 3.5 rounds, with time complexity $2^{109+48} = 2^{157}$ 4-round computations.

6.3 Truncated Differential Attack on MESH-128

A differential attack on 3.5-round MESH-128 can use the following differential, with average probability 2^{-128}, and with A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, $P \in \mathbb{Z}_2^{16}$.:

$$\begin{aligned}
(A, 0, 0, B, C, 0, 0, D) &\xrightarrow{2^{-32}} (E, 0, 0, F, E, 0, 0, F) \xrightarrow{(0,0,0,0)\to(0,0,0,0)} (E, E, 0, 0, 0, 0, F, F) \\
(E, E, 0, 0, 0, 0, F, F) &\xrightarrow{1} (G, H, 0, 0, 0, 0, I, J) \xrightarrow{(G,H,I,J)\to(J,I,H,G)} (0, G, H, 0, 0, I, J, 0) \\
(0, G, H, 0, 0, I, J, 0) &\xrightarrow{2^{-32}} (0, K, L, 0, 0, K, L, 0) \xrightarrow{(0,0,0,0)\to(0,0,0,0)} (0, 0, K, L, K, L, 0, 0) \\
(0, 0, K, L, K, L, 0, 0) &\xrightarrow{1} (0, 0, M, N, O, P, 0, 0)\,. \qquad (6)
\end{aligned}$$

Conservative complexity estimates, based on experimental results following [4], for a truncated differential attack on 3.5-round MESH-128 using (6) are as follows: $2^{64} \cdot 2^{63} \cdot 4 \cdot 2^{16} \cdot \frac{6}{116} \approx 2^{141}$ 3.5-round MESH-128 encryptions, 2^{128} chosen plaintexts, and 2^{64} 128-bit blocks of memory, to recover $Z_1^{(1)}$, $Z_4^{(1)}$, $Z_5^{(1)}$, $Z_8^{(1)}$, $Z_3^{(4)}$, $Z_4^{(4)}$, $Z_5^{(4)}$, $Z_6^{(4)}$. An additional truncated differential, in the Appendix, can be used to recover $Z_2^{(1)}$, $Z_3^{(1)}$, $Z_6^{(1)}$, $Z_7^{(1)}$, $Z_1^{(4)}$, $Z_2^{(4)}$, $Z_7^{(4)}$, $Z_8^{(4)}$. In total, using

both differentials, 128-bit user keys are recovered. The remaining 128 key bits can be found by exhaustive search.

An attack on 4-round MESH-128 can guess $Z_9^{(4)}$, $Z_{10}^{(4)}$, $Z_{11}^{(4)}$, $Z_{12}^{(4)}$, and apply the previous attack on 3.5 rounds, with time complexity $2^{142+64} = 2^{206}$ 4-round computations.

7 Linear Attack

Our linear analysis on MESH ciphers followed the framework of Daemen *et al.* [6]. Initially, all one-round linear relations under weak-key assumptions were exhaustively obtained for MESH-64, MESH-96, and MESH-128.

7.1 Linear Attack on MESH-64

For MESH-64, an example of a linear relation under weak-key assumption is $(0,0,0,1) \rightarrow (0,0,1,0)$, provided $Z_4^{(1)}$, $Z_6^{(1)}$, $Z_7^{(1)} \in \{0,1\}$ (an odd-numbered round). According to the key schedule, the user key words are xored to fixed constants c_i, $0 \leq i \leq 7$, and are used as the first eight subkey words. It means that the subkey restrictions $Z_4^{(1)}$, $Z_6^{(1)}$, $Z_7^{(1)} \in \{0,1\}$ can be satisfied if the most significant 15 bits of the key words match the corresponding bits of the associated constants. Multiple-round linear relations are obtained by concatenating one-round linear relations, and deriving the corresponding fraction of keys from the key space from which the relation holds. This fraction of keys can be derived from the restrictions on subkeys in the one-round linear relations. Nonetheless, the key schedule of MESH-64 does not have a simple mapping of subkey bits to user key bits such as in IDEA. The fraction of keys for the linear relations in MESH-64 was estimated from the weak-key class sizes obtained by exhaustive key search in a mini-version of MESH-64 with[3] 16-bit blocks, denoted MESH-64(16), where the key size is 32 bits. Analysis of MESH-64(16) indicated that each subkey restriction (most significant three bits equal to zero) is satisfied for a fraction of 2^{-3} or less per subkey. This observation allowed to estimate the fraction of keys (and the weak-key class size) that satisfy a linear relation for MESH-64 as 2^{-15} per subkey.

The longest linear relations obtained (starting from the first round) are as follows:

. $(0,1,0,1) \xrightarrow{1r} (0,0,1,1) \xrightarrow{1r} (1,0,1,0) \xrightarrow{1r} (1,1,0,0) \xrightarrow{1r} (0,1,0,1) \xrightarrow{1r} (0,0,1,1)$, provided $Z_4^{(1)}$, $Z_3^{(2)}$, $Z_5^{(2)}$, $Z_6^{(2)}$, $Z_1^{(3)}$, $Z_2^{(4)}$, $Z_5^{(4)}$, $Z_6^{(4)}$, $Z_4^{(5)} \in \{0,1\}$. For MESH-64(16) this relation holds for a weak-key class of size 4, which corresponds to a fraction of $4 \cdot 2^{-32} = 2^{-30}$ of its key space. This fraction is less than $2^{-3*9} = 2^{-27}$, that is to be expected if each subkey restriction held independently. For MESH-64, the weak-key class size is estimated as $2^{128-15*8} = 2^8$ for 4 rounds at most;

[3] With left rotation by 3 bits per word in the key schedule.

. $(1,0,1,0) \xrightarrow{1r} (1,1,0,0) \xrightarrow{1r} (0,1,0,1) \xrightarrow{1r} (0,0,1,1) \xrightarrow{1r} (1,0,1,0) \xrightarrow{1r} (1,1,0,0)$, provided $Z_1^{(1)}$, $Z_2^{(2)}$, $Z_5^{(2)}$, $Z_6^{(2)}$, $Z_4^{(3)}$, $Z_4^{(4)}$, $Z_5^{(4)}$, $Z_6^{(4)}$, $Z_1^{(5)} \in \{0,1\}$. For MESH-64(16) this relation holds for a weak-key class of size 5, which is a fraction of $5 \cdot 2^{-32} \approx 2^{-30}$ of its key space. This fraction is less than $2^{-3*9} = 2^{-27}$, that is to be expected if each subkey restriction held independently. For MESH-64, the weak-key class size is estimated as $2^{128-15*8} = 2^8$ for 4 rounds.

These linear relations can distinguish the first four rounds of MESH-64 from a random permutation, under weak-key assumptions, or can be used in a 0.5R attack on 4.5-round MESH-64 to recover at least one of the subkeys $Z_i^{(5)}$, $1 \leq i \leq 4$, using $N \approx 8 \cdot (2^{-1})^{-2} = 32$ known plaintexts and about $32 \cdot 2^{16} = 2^{21}$ parity computations.

7.2 Linear Attack on MESH-96

The MA-box of MESH-96 effectively avoids many one-round linear relations, as can be observed in Table 5, because it contains multiplications in which subkeys are not directly involved as operands. The approach to a linear attack on MESH-96 is similar to that on MESH-64. Exhaustive key search was applied to a mini-version MESH-96 with 4-bit words, denoted MESH-96(24), in order to estimate the weak-key class size from the fraction of the key space satisfying the weak-key restrictions. The longest linear relation (starting from the first round) uses the one-round iterative relation: $(1,1,1,1,1,1) \xrightarrow{1r} (1,1,1,1,1,1)$, repeated for 3.5 rounds, provided $Z_1^{(1)}$, $Z_3^{(1)}$, $Z_5^{(1)}$, $Z_2^{(2)}$, $Z_4^{(2)}$, $Z_6^{(2)}$, $Z_1^{(3)}$, $Z_3^{(3)}$, $Z_5^{(3)}$, $Z_2^{(4)}$, $Z_4^{(4)}$, $Z_6^{(4)} \in \{0,1\}$, for MESH-96(24) the weak-key class size is 620, which corresponds to a fraction of $2^{9.27-48} \approx 2^{-38.72}$ of its key space. This fraction is smaller than $2^{-3*12} = 2^{-36}$ that would be expected if each subkey restriction held independently. For MESH-96, the weak-key class size is estimated as at most $2^{192-15*12} \approx 2^{12}$. This linear relation can distinguish the first 3.5-round MESH-96 from a random permutation, or can be used in a key-recovery attack on 4-round MESH-96, to find $Z_7^{(4)}$, $Z_8^{(4)}$, $Z_9^{(4)}$, with $N \approx 8 \cdot (2^{-1})^{-2} = 32$ chosen plaintexts, and $32 \cdot 2^{48} = 2^{53}$ parity computations. Attacking 4.5 rounds requires guessing six subkeys of the fifth key-mixing layer, leading to a complexity of $2^{96+53} = 2^{149}$ parity computations.

7.3 Linear Attack on MESH-128

The MA-box of MESH-128, similar to that of MESH-96, also avoids, for the same reasons, many one-round relations, as can be observed in Table 6. Estimates for the weak-key class size of MESH-128 are derived similarly to that of MESH-64 and MESH-96, namely, assuming a fraction of 2^{-15} of the key space satisfies each subkey restriction. The longest linear relation (starting from the first round), uses the one-round iterative relation: $(1,1,1,1,1,1,1,1) \xrightarrow{1r} (1,1,1,1,1,1,1,1)$, repeated for 3.5 rounds, provided $Z_1^{(1)}$, $Z_3^{(1)}$, $Z_6^{(1)}$, $Z_8^{(1)}$, $Z_2^{(2)}$, $Z_4^{(2)}$, $Z_5^{(2)}$, $Z_7^{(2)}$, $Z_1^{(3)}$,

$Z_3^{(3)}$, $Z_6^{(3)}$, $Z_8^{(3)}$, $Z_2^{(4)}$, $Z_4^{(4)}$, $Z_5^{(4)}$, $Z_7^{(4)} \in \{0,1\}$, this relation implies a weak-key class of estimated size $2^{256-15*16} = 2^{16}$.

This linear relation can distinguish the first 3.5 rounds of MESH-128 from a random permutation, or can be used in a 0.5R attack on 4-round MESH-128, to recover subkeys $Z_9^{(4)}$, $Z_{10}^{(4)}$, $Z_{11}^{(4)}$, $Z_{12}^{(4)}$, with about $N \approx 8 \cdot (2^{-1})^{-2} = 32$ chosen plaintexts and $32 \cdot 2^{64} = 2^{69}$ parity computations. Attacking 4.5 rounds involves guessing eight subkeys of the fifth key-mixing layer, leading to a complexity of $2^{69+128} = 2^{197}$ parity computations.

8 Impossible Differential Attacks

Impossible differential (ID) attacks on MESH ciphers follow a similar setting of Biham *et al.* [2].

8.1 Impossible Differential Attack of MESH-64

A key-recovery ID attack on 3.5-round MESH-64 uses the 2.5-round impossible differential $(a,0,a,0) \not\rightarrow (b,b,0,0)$, with $a,b \neq 0$, starting after the first key mixing until the end of the third round. Let $(X_1^1, X_2^1, X_3^1, X_4^1)$ be a plaintext block, and $(Y_1^4, Y_2^4, Y_3^4, Y_4^4)$ the corresponding ciphertext block. The attack chooses a structure of 2^{32} plaintexts with fixed X_2^1 and X_4^1, and all possible values for X_1^1 and X_3^1. There are about $2^{32} \cdot (2^{32}-1)/2 \approx 2^{63}$ plaintext pairs with xor difference $(X_1^{1'}, 0, X_3^{1'}, 0)$. Collect about 2^{31} pairs from the structure whose ciphertext difference after 3.5 rounds satisfies $Y_3^{4'} = 0$ and $Y_4^{4'} = 0$. For each such pair try all 2^{32} subkeys $(Z_1^{(1)}, Z_3^{(1)})$ and partially encrypt (X_1^1, X_3^1) in each of the two plaintexts of the pair. Collect about 2^{16} subkeys $(Z_1^{(1)}, Z_3^{(1)})$ satisfying $(X_1^1 \odot Z_1^{(1)}) \oplus (X_1^{1*} \odot Z_1^{(1)}) = (X_3^1 \boxplus Z_3^{(1)}) \oplus (X_3^{1*} \boxplus Z_3^{(1)})$. This step takes 2^{17} time, and 2^{16} memory. Next, try all 2^{32} subkeys $(Z_1^{(4)}, Z_2^{(4)})$ to partially decrypt (Y_1^4, Y_2^4) in each of the two ciphertexts of the pair. Collect about 2^{16} subkeys $(Z_1^{(4)}, Z_2^{(4)})$ such that $(Y_1^4 \boxminus Z_1^{(4)}) \oplus (Y_1^{4*} \boxminus Z_1^{(4)}) = (Y_2^4 \odot (Z_2^{(4)})^{-1}) \oplus (Y_2^{4*} \odot (Z_2^{(4)})^{-1})$. This step takes 2^{17} time and 2^{16} memory. Make a list of all 2^{32} 64-bit subkeys $(Z_1^{(1)}, Z_3^{(1)}, Z_1^{(4)}, Z_2^{(4)})$, combining the two previous steps. Those subkeys cannot be the correct values, because they lead to a pair of the impossible differential. Each pair defines a list of about 2^{32} 64-bit wrong subkey values. It is expected that after 90 structures, the number of remaining wrong subkeys is: $2^{64} \cdot (1 - \frac{2^{32}}{2^{64}})^{2^{31} \cdot 90} \approx \frac{2^{64}}{e^{45}} \approx 2^{-0.92}$. Therefore, the correct subkey can be uniquely identified. The attack requires $90 \cdot 2^{32} \approx 2^{38.5}$ chosen plaintexts. The memory complexity is 2^{61} bytes, dominated by the sieving of the correct 64-bit subkey. The time complexity is $2^{31} \cdot 90 \cdot (2^{17} + 2^{17}) \approx 2^{56}$ steps. The 2.5-round impossible differential $(0,a,0,a) \not\rightarrow (0,0,b,b)$, with $a,b \neq 0$, can be used to discover $(Z_2^{(1)}, Z_4^{(1)}, Z_3^{(4)}, Z_4^{(4)})$. The joint time complexity is about 2^{57} steps. If a step consists of a modular multiplication and there are 17 multiplications in 3.5 rounds then the latter corresponds to $2^{57} \cdot 1/17 \approx 2^{53}$ 3.5-round computations.

Data complexity amounts to $2^{39.5}$ chosen plaintexts, and 2^{61} bytes of memory. In total, the first 64 user key bits are recovered, and the remaining 64 key bits can be obtained by exhaustive search, and the final time complexity becomes 2^{64} 3.5-round computations. To attack 4-round MESH-64, the subkeys $Z_5^{(4)}$, $Z_6^{(4)}$, $Z_7^{(4)}$ are guessed, and the previous attack on 3.5 rounds is performed. The time complexity increases to $2^{64+48} = 2^{112}$ steps.

8.2 Impossible Differential Attack of MESH-96

An ID attack on MESH-96 can use the 2.5-round distinguishers $(a, 0, 0, a, 0, 0) \not\to (b, b, c, c, 0, 0)$, $(0, a, 0, 0, a, 0) \not\to (0, 0, b, b, c, c)$, and $(0, 0, a, 0, 0, a) \not\to (b, b, 0, 0, c, c)$, with $a, b, c \neq 0$, each one starting after the first key-mixing half-round, until the end of the third round. The attack discovers $(Z_i^{(1)}, Z_i^{(4)})$, $1 \leq i \leq 6$, and is analogous to the attack on MESH-64. Data complexity is about 2^{57} chosen plaintexts, 2^{93} bytes of memory, and time equivalent to $2^{73.5}$ steps. In total, the first 96 user key bits were directly recovered, and the remaining 96 key bits can be found by exhaustive search, which leads to a final time complexity of 2^{96} 3.5-round computations.

An attack on 4 rounds can guess $Z_7^{(4)}$, $Z_8^{(4)}$, $Z_9^{(4)}$, and apply the previous attack on 3.5 rounds. The time complexity becomes $2^{96+48} = 2^{144}$ 4-round computations.

8.3 Impossible Differential Attack of MESH-128

The best trade-off ID attack uses the following 2.5-round distinguishers: (a, b, 0, 0, a, b, 0, 0) $\not\to$ (c, c, d, e, d, e, 0, 0), (0, 0, a, b, 0, 0, a, b) $\not\to$ (c, c, 0, d, 0, d, e, e) with $a, b, c, d, e \neq 0$. The attack proceeds similar to the one on MESH-96, and recovers $(Z_i^{(1)}, Z_i^{(4)})$ for $1 \leq i \leq 8$. Data complexity is 2^{65} chosen plaintexts, 2^{157} bytes of memory and time equivalent to 2^{107} 3.5-round computations. In total, the first 128 user key bits are effectively recovered, and from the key schedule, $Z_i^{(4)}$, $1 \leq i \leq 8$ do not provide enough information to deduce the remaining 128 user key bits, which are then recovered by exhaustive search, leading to a final time complexity of 2^{128} 3.5-round MESH-128 computations. An attack 4 rounds can guess $(Z_9^{(4)}, Z_{10}^{(4)}, Z_{11}^{(4)}, Z_{12}^{(4)})$, and apply the attack on 3.5 rounds. Time complexity increases to $2^{128+64} = 2^{192}$ steps.

9 Demirci's Attack

This section follows the work of Demirci in [9].

9.1 Demirci's Attack on MESH-64

Demirci's attack using 1st-order integrals [10, 7] can be adapted to MESH-64 starting from the 2nd round, or any other even round. The integral operator is

exclusive-or. Consider a multiset of the form (P P P A), namely, with the first three words constant (passive) and the 4th word active. Let $C^{(i)} = (C_1^{(i)}, C_2^{(i)}, C_3^{(i)}, C_4^{(i)})$ denote the ciphertext after i rounds. After 1-round MESH-64, the output multiset has the form (? ? A $*$), that is, the first two words are garbled, the 3rd word is active and the 4th is balanced. The multiset after 1.5-round becomes (? ? A ?) but the least significant bit of $C_2^{(1.5)}$ is constant because it is a combination of only active words from the MA-box of the initial round. This property is used as a distinguisher for Demirci's attack on 2-round MESH-64:

$$\mathrm{LSB}(C_2^{(2)} \oplus C_3^{(2)} \oplus Z_6^{(3)} \odot ((C_1^{(2)} \oplus C_2^{(2)}) \odot Z_5^{(3)} \boxplus (C_3^{(2)} \oplus C_4^{(2)}))) = 0, \quad (7)$$

where LSB denotes the least significant bit function. Therefore, over a multiset, equation (7) can be used to find ($Z_5^{(3)}$, $Z_6^{(3)}$). It provides a one-bit condition, thus, this search over 32 key bits requires $32 \cdot 2^{16} = 2^{21}$ chosen plaintexts, and an effort of $2^{32} \cdot 2^{16} + 2^{31} \cdot 2^{16} + \ldots 2^1 \cdot 2^{16} \approx 2^{49}$ half-round computations or about 2^{47} 2-round computations. An attack on 2.5 rounds can guess ($Z_1^{(4)}$, $Z_2^{(4)}$, $Z_3^{(4)}$, $Z_4^{(4)}$) and apply the previous attack, at the cost of $2^{47} \cdot 2^{64} = 2^{111}$ 2.5-round MESH-64 computations.

9.2 Demirci's Attack on MESH-96

For MESH-96, Demirci's attack on 2 rounds can use 1st-order multisets of the form (P P P P P A) with only the 8th input word active, and the distinguisher:

$$\begin{aligned} \mathrm{LSB}\ (&C_3^{(2)} \oplus C_5^{(2)} \oplus (Z_7^{(2)} \odot (C_1^{(2)} \oplus C_2^{(2)}) \boxplus (C_4^{(2)} \oplus C_3^{(2)})) \odot \\ &(Z_8^{(2)} \boxplus (C_1^{(2)} \oplus C_2^{(2)}) \odot Z_7^{(2)} \boxplus (C_4^{(2)} \oplus C_3^{(2)}) \odot (C_5^{(2)} \oplus C_6^{(2)}))) = 0, \end{aligned} \quad (8)$$

where $(C_1^{(i)}, C_2^{(i)}, C_3^{(i)}, C_4^{(i)}, C_5^{(i)}, C_6^{(i)})$ is the ciphertext after i rounds. The attack is analogous to that on MESH-64. Equation (9) allows to recover ($Z_7^{(2)}$, $Z_8^{(2)}$) using $32 \cdot 2^{16} = 2^{21}$ chosen plaintexts and time about 2^{49} 2-round MESH-96 computations. To attack 2.5 rounds guess ($Z_1^{(3)}$, $Z_2^{(3)}$, $Z_3^{(3)}$, $Z_4^{(3)}$, $Z_5^{(3)}$, $Z_6^{(3)}$) and apply the previous attack, leading to $2^{49} \cdot 2^{96} = 2^{145}$ 2.5-round MESH-96 computations.

9.3 Demirci's Attack on MESH-128

For MESH-128, Demirci's attack with 1st-order integrals does not apply, because the four layers in its MA-box do not allow any balanced output word, not even their LSBs, which is a necessary condition for the attack. Nonetheless, there are alternative attacks using higher-order integrals [9, 10] that can cross the 4-layer MA-box. A 2nd-order Demirci's attack on 2-round MESH-128 can use, for instance, multisets in which the first and fourth input words are jointly active, and all the other six words are passive. Such multiset requires 2^{32} chosen plaintexts. The multiset after 1.5 rounds contains only balanced words. In particular, the integral of the inputs to the MA-box of the second round all sum to zero. Some

terminology for the attack description follows: let ($C_1^{(i)}$, $C_2^{(i)}$, $C_3^{(i)}$, $C_4^{(i)}$, $C_5^{(i)}$, $C_6^{(i)}$, $C_7^{(i)}$, $C_8^{(i)}$) be the ciphertext multiset after i rounds, and $p = C_1^{(2)} \oplus C_2^{(2)}$, $q = C_3^{(2)} \oplus C_5^{(2)}$, $r = C_4^{(2)} \oplus C_6^{(2)}$, and $s = C_7^{(2)} \oplus C_8^{(2)}$. The distinguisher for the higher-order Demirci's attack is obtained by exploring the least significant bit of the leftmost two output words of the MA-box of the second round:

$$\begin{aligned}
&\text{LSB}\,(C_4^{(2)} \oplus C_7^{(2)} \oplus C_4^{(1)} \odot Z_4^{(2)} \oplus C_7^{(1)} \odot Z_7^{(2)}) = \\
&\text{LSB}\,(Z_{11}^{(2)} \odot (p \odot Z_9^{(2)} \boxplus (p \odot Z_9^{(2)} \boxplus q) \odot \\
&\qquad (r \odot (p \odot Z_9^{(2)} \boxplus q) \boxplus (r \odot (p \odot Z_9^{(2)} \boxplus q) \boxplus s) \odot Z_{10}^{(2)}))). \qquad (9)
\end{aligned}$$

Over the given plaintext multiset, the integral of LSB($C_4^{(2)} \oplus C_7^{(2)} \oplus C_4^{(1)} \odot Z_4^{(2)} \oplus C_7^{(1)} \odot Z_7^{(2)}$) is zero, because the subkeys are fixed, and the corresponding intermediate values are balanced. Therefore, (9) provides a one-bit condition to recover $Z_9^{(2)}$, $Z_{10}^{(2)}$ and $Z_{11}^{(2)}$. The data requirements are $48 \cdot 2^{32} = 2^{37.6}$ chosen plaintexts, and time equivalent to $2^{48} \cdot 2^{32} + 2^{47} \cdot 2^{32} + \ldots 2 \cdot 2^{32} = 2^{32} \cdot 2 \cdot (2^{48} - 1) \approx 2^{81}$ half-round computations, or about 2^{79} 2-round computations.

10 Performance

The software performance figures for MESH ciphers are shown in Table 2, and are estimated from the number of multiplications compared to those in IDEA. Simulations demonstrate that one $\odot$ costs about three times more than an $\boxplus$ or an $\oplus$ operation. Comparison with the performances of triple-DES [**?**] and AES [**?**], on a commom platform (Pentium III under Linux) and under similarly optimized software code, come from experiments conducted at the NESSIE Project [**?**], which comprise an independent performance evaluation of the three mentioned ciphers. Note that 8.5-round MESH-64 uses 42 multiplications (Table 1), compared to 34 in IDEA. Since both encrypt the same amount of bits, an estimate for the number of cycles per byte in MESH-64 is $56 \cdot (42/34) \approx 70$, which is about 25% slower than IDEA. Simulations show performance about 30% slower than IDEA, due to the unaccounted number of modular additions. MESH-96 with 10.5 rounds uses 83 multiplications, but encrypts 12 bytes in comparison to 8 bytes of IDEA. This implies $56 \cdot (83/34) \cdot (8/12) \approx 92$ cycles per byte, or about 64% slower than IDEA. MESH-128, with 12.5 rounds uses 148 multiplications (Table 1), but encrypts 16 bytes instead of 8 in IDEA. This implies a cost of roughly $56 \cdot (148/34) \cdot (8/16) \approx 122$ cycles per byte, an 118% overhead.

11 Conclusions

This paper described the MESH block ciphers, which are based on the same group operations of IDEA, but with a number of novel features:

. flexible block sizes in steps of 32 bits. For more than ten years since the publication of IDEA [11], no IDEA variant has been proposed with block sizes

Table 2. Performance estimates of MESH ciphers, IDEA, triple-DES and AES.

Cipher	Block Size	Key Size	#Rounds	#Cycles/Byte
AES	128	128	10	25
	128	192	12	30
	128	256	14	34
triple-DES	64	168	48	154
IDEA	64	128	8.5	56
MESH-64	64	128	8.5	70
MESH-96	96	192	10.5	92
MESH-128	128	256	12.5	122

larger than 64 bits or word sizes larger than 16 bits. Maybe such attempts were hindered by the fact that $2^{32}+1$ is not a prime number, and thus, $\mathbb{Z}^*_{2^{32}+1}$ is not a field. The MESH ciphers provide an alternative approach that do not depend on the need for larger word sizes or finite fields.

. new key schedule algorithms with fast key avalanche. Each subkey of IDEA depends (linearly) on exactly 16 user key bits, while in MESH all subkeys after the second round depend (non-linearly) on all user key words. Moreover, the new key schedules avoid (differential and linear) weak keys as existing in IDEA. Software simulations[4] of the key schedule give about 1888 cycles/key-setup for IDEA, 2054 cycles/key-setup for MESH-64, 3869 cycles/key-setup for MESH-96, and 5536 cycles/key-setup for MESH-128.

. larger MA-boxes designed to better resist differential and linear attacks.

. distinct key-mixing layers, originally designed to counter slide attacks [3], but also proved useful against Demirci's attack [9]. The design of the MESH ciphers incorporates measures to counter a number of modern cryptanalytic attacks developed along the past 12 years [2–4, 6, 9–11, 13]. The security margin of MESH ciphers, as well as of IDEA, against the described attacks seems relatively high. Table 3 in Appendix A details the attack complexity figures for IDEA and MESH ciphers, where linear attacks are restricted to a weak-key class; 'KP' means Known Plaintext and 'CP', Chosen Plaintext.

Acknowledgements

We would like to thank Alex Biryukov for the many explanations concerning the impossible differential technique. We are also grateful to the anonymous referees, whose comments helped improve the presentation of this paper.

A Attack Summary

This appendix lists the attack complexities on MESH ciphers and compares them with previous attacks on IDEA.

[4] On a Pentium III 667 MHz under Linux.

Table 3. Summary of attack complexities on IDEA and MESH ciphers.

Cipher	Block Size	Key Size	Attack	#Rounds	Data	Memory	Time
IDEA	64	128	Demirci	2	23 CP	23	2^{64}
			Demirci	2.5	55 CP	55	2^{81}
(8.5 rounds)			Demirci	3	71 CP	71	2^{71}
			Demirci	3	2^{33} CP	2^{33}	2^{82}
			Demirci	3.5	103 CP	103	2^{103}
			Trunc. Diff.	3.5	2^{56} CP	2^{32}	2^{67}
			Imp. Diff.	3.5	$2^{38.5}$CP	2^{37}	2^{53}
			Demirci	4	2^{34} CP	2^{34}	2^{114}
			Imp. Diff.	4	$2^{38.5}$ CP	2^{37}	2^{70}
			Imp. Diff.	4.5	2^{64} CP	2^{32}	2^{112}
MESH-64	64	128	Demirci	2	2^{21} CP	2^{16}	2^{47}
			Demirci	2.5	2^{21} CP	2^{16}	2^{111}
(8.5 rounds)			Imp. Diff.	3.5	$2^{39.5}$ CP	2^{61}	2^{64}
			Trunc. Diff.	3.5	2^{64} CP	2^{32}	2^{78}
			Imp. Diff.	4	$2^{39.5}$ CP	2^{61}	2^{112}
			Trunc. Diff.	4	2^{64} CP	2^{32}	2^{126}
			Linear	4.5	32 KP	32	2^{21}
MESH-96	96	192	Demirci	2	2^{21} CP	2^{16}	2^{47}
			Demirci	2.5	2^{21} CP	2^{16}	2^{143}
(10.5 rounds)			Imp. Diff.	3.5	2^{56} CP	2^{93}	2^{96}
			Trunc. Diff.	3.5	2^{96} CP	2^{64}	2^{109}
			Linear	4	32 KP	32	2^{53}
			Imp. Diff.	4	2^{56} CP	2^{93}	2^{144}
			Trunc. Diff.	4	2^{96} CP	2^{64}	2^{157}
			Linear	4.5	32 KP	32	2^{149}
MESH-128	128	256	Demirci	2	$2^{37.6}$ CP	$2^{37.6}$	2^{79}
			Demirci	2.5	$2^{37.6}$ CP	$2^{37.6}$	2^{143}
(12.5 rounds)			Imp. Diff.	3.5	2^{65} CP	2^{157}	2^{128}
			Trunc. Diff.	3.5	2^{128} CP	2^{64}	2^{142}
			Linear	4	32 KP	32	2^{69}
			Imp. Diff.	4	2^{65} CP	2^{157}	2^{192}
			Trunc. Diff.	4	2^{128} CP	2^{64}	2^{206}
			Linear	4.5	32 KP	32	2^{197}

B Additional Truncated Differentials

This appendix details alternative diagrams for truncated differential attacks on MESH ciphers.

$$
\begin{aligned}
(0, A, 0, B) &\xrightarrow{2^{-16}} (0, C, 0, C) \xrightarrow{(0,0)\xrightarrow{1}(0,0)} (0, 0, C, C) \\
(0, 0, C, C) &\xrightarrow{1} (0, 0, D, E) \xrightarrow{(D,E)\xrightarrow{2^{-32}}(E,D)} (D, 0, E, 0) \\
(D, 0, E, 0) &\xrightarrow{2^{-16}} (F, 0, F, 0) \xrightarrow{(0,0)\xrightarrow{1}(0,0)} (F, F, 0, 0) \\
(F, F, 0, 0) &\xrightarrow{1} (G, H, 0, 0)
\end{aligned} \tag{10}
$$

$$
\begin{aligned}
(A, 0, B, C, 0, D) &\xrightarrow{2^{-32}} (E, 0, F, E, 0, F) \xrightarrow{(0,0,0)\xrightarrow{1}(0,0,0)} (E, E, 0, 0, F, F) \\
(E, E, 0, 0, F, F) &\xrightarrow{2^{-16}} (G, H, 0, 0, H, I) \xrightarrow{(G,0,I)\xrightarrow{2^{-48}}(I,H,G)} (0, G, 0, 0, I, 0) \\
(0, G, 0, 0, I, 0) &\xrightarrow{2^{-16}} (0, J, 0, 0, J, 0) \xrightarrow{(0,0,0)\xrightarrow{1}(0,0,0)} (0, 0, J, J, 0, 0) \\
(0, 0, J, J, 0, 0) &\xrightarrow{1} (0, 0, K, L, 0, 0)
\end{aligned} \tag{11}
$$

$$
\begin{aligned}
(0, A, B, 0, C, D) &\xrightarrow{2^{-32}} (0, E, F, 0, E, F) \xrightarrow{(0,0,0)\xrightarrow{1}(0,0,0)} (0, 0, E, E, F, F) \\
(0, 0, E, E, F, F) &\xrightarrow{2^{-16}} (0, 0, G, H, I, G) \xrightarrow{(H,I,0)\xrightarrow{2^{-48}}(G,I,H)} (H, 0, 0, I, 0, 0) \\
(H, 0, 0, I, 0, 0) &\xrightarrow{2^{-16}} (J, 0, 0, J, 0, 0) \xrightarrow{(0,0,0)\xrightarrow{1}(0,0,0)} (J, J, 0, 0, 0, 0) \\
(J, J, 0, 0, 0, 0) &\xrightarrow{1} (K, L, 0, 0, 0, 0)
\end{aligned} \tag{12}
$$

$$
\begin{aligned}
(0, A, B, 0, 0, C, D, 0) &\xrightarrow{2^{-32}} (0, E, F, 0, 0, E, F, 0) \xrightarrow{(0,0,0,0)\xrightarrow{1}(0,0,0,0)} (0, 0, E, F, E, F, 0, 0) \\
(0, 0, E, F, E, F, 0, 0) &\xrightarrow{1} (0, 0, G, H, I, J, 0, 0) \xrightarrow{(I,J,G,H)\xrightarrow{2^{-64}}(H,G,J,I)} (I, 0, 0, G, J, 0, 0, H) \\
(I, 0, 0, G, J, 0, 0, H) &\xrightarrow{2^{-32}} (K, 0, 0, L, K, 0, 0, L) \xrightarrow{(0,0,0,0)\xrightarrow{1}(0,0,0,0)} (K, K, 0, 0, 0, 0, L, L) \\
(K, K, 0, 0, 0, 0, L, L) &\xrightarrow{1} (M, N, 0, 0, 0, 0, O, P)
\end{aligned} \tag{13}
$$

C One-round linear relations for MESH ciphers

This appendix lists one-round linear relations for MESH ciphers, under weak-key assumptions.

Table 4. One-round linear relations for MESH-64, and subkey restrictions.

one-round linear relation	odd-round subkeys					even-round subkeys				
	$Z_1^{(i)}$	$Z_4^{(i)}$	$Z_5^{(i)}$	$Z_6^{(i)}$	$Z_7^{(i)}$	$Z_2^{(i)}$	$Z_3^{(i)}$	$Z_5^{(i)}$	$Z_6^{(i)}$	$Z_7^{(i)}$
$(0,0,0,1) \to (0,0,1,0)$	–	{0,1}	–	{0,1}	{0,1}	–	–	–	{0,1}	{0,1}
$(0,0,1,0) \to (1,0,0,0)$	–	–	{0,1}	–	{0,1}	–	{0,1}	{0,1}	–	{0,1}
$(0,0,1,1) \to (1,0,1,0)$	–	{0,1}	{0,1}	{0,1}	–	–	{0,1}	{0,1}	{0,1}	–
$(0,1,0,0) \to (0,0,0,1)$	–	–	–	{0,1}	{0,1}	{0,1}	–	–	{0,1}	{0,1}
$(0,1,0,1) \to (0,0,1,1)$	–	{0,1}	–	–	–	{0,1}	–	–	–	–
$(0,1,1,0) \to (1,0,0,1)$	–	–	{0,1}	{0,1}	–	{0,1}	{0,1}	{0,1}	{0,1}	–
$(0,1,1,1) \to (1,0,1,1)$	–	{0,1}	{0,1}	–	{0,1}	{0,1}	{0,1}	{0,1}	–	{0,1}
$(1,0,0,0) \to (0,1,0,0)$	{0,1}	–	{0,1}	–	{0,1}	–	–	{0,1}	–	{0,1}
$(1,0,0,1) \to (0,1,1,0)$	{0,1}	{0,1}	{0,1}	{0,1}	–	–	–	{0,1}	{0,1}	–
$(1,0,1,0) \to (1,1,0,0)$	{0,1}	–	–	–	–	–	{0,1}	–	–	–
$(1,0,1,1) \to (1,1,1,0)$	{0,1}	{0,1}	–	{0,1}	{0,1}	–	{0,1}	–	{0,1}	{0,1}
$(1,1,0,0) \to (0,1,0,1)$	{0,1}	–	{0,1}	{0,1}	–	{0,1}	–	{0,1}	{0,1}	–
$(1,1,0,1) \to (0,1,1,1)$	{0,1}	{0,1}	{0,1}	–	{0,1}	{0,1}	–	{0,1}	–	{0,1}
$(1,1,1,0) \to (1,1,0,1)$	{0,1}	–	–	{0,1}	{0,1}	{0,1}	{0,1}	–	{0,1}	{0,1}
$(1,1,1,1) \to (1,1,1,1)$	{0,1}	{0,1}	–	–	–	{0,1}	{0,1}	–	–	–

Table 5. One-round linear relations for MESH-96, and subkey restrictions.

one-round linear relation	odd-round subkeys					even-round subkeys				
	$Z_1^{(i)}$	$Z_3^{(i)}$	$Z_5^{(i)}$	$Z_7^{(i)}$	$Z_9^{(i)}$	$Z_2^{(i)}$	$Z_4^{(i)}$	$Z_6^{(i)}$	$Z_7^{(i)}$	$Z_9^{(i)}$
$(1,0,0,1,0,0) \to (1,1,0,0,0,0)$	{0,1}	–	–	–	–	–	{0,1}	–	–	–
$(0,1,0,0,1,0) \to (0,0,1,1,0,0)$	–	–	{0,1}	–	–	{0,1}	–	–	–	–
$(0,0,1,0,0,1) \to (0,0,0,0,1,1)$	–	{0,1}	–	–	–	–	–	{0,1}	–	–
$(1,1,0,1,1,0) \to (1,1,1,1,0,0)$	{0,1}	–	{0,1}	–	–	{0,1}	{0,1}	–	–	–
$(1,0,1,1,0,1) \to (1,1,0,0,1,1)$	{0,1}	{0,1}	–	–	–	–	{0,1}	{0,1}	–	–
$(0,1,1,0,1,1) \to (0,0,1,1,1,1)$	–	{0,1}	{0,1}	–	–	{0,1}	–	{0,1}	–	–
$(1,1,1,1,1,1) \to (1,1,1,1,1,1)$	{0,1}	{0,1}	{0,1}	–	–	{0,1}	{0,1}	{0,1}	–	–
$(1,0,0,1,1,0) \to (0,0,1,0,0,0)$	{0,1}	–	{0,1}	{0,1}	{0,1}	–	{0,1}	–	{0,1}	{0,1}
$(0,0,0,0,1,0) \to (1,1,1,0,0,0)$	–	–	{0,1}	{0,1}	{0,1}	–	–	–	{0,1}	{0,1}
$(1,1,0,1,0,0) \to (0,0,0,1,0,0)$	{0,1}	–	–	{0,1}	{0,1}	{0,1}	{0,1}	–	{0,1}	{0,1}
$(0,1,0,0,0,0) \to (1,1,0,1,0,0)$	–	–	–	{0,1}	{0,1}	{0,1}	–	–	{0,1}	{0,1}
$(1,1,1,1,0,1) \to (0,0,0,1,1,1)$	{0,1}	{0,1}	–	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}
$(1,0,1,1,1,1) \to (0,0,1,0,1,1)$	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	–	{0,1}	{0,1}	{0,1}	{0,1}
$(0,1,1,0,0,1) \to (1,1,0,1,1,1)$	–	{0,1}	–	{0,1}	{0,1}	{0,1}	–	{0,1}	{0,1}	{0,1}
$(0,0,1,0,1,1) \to (1,1,1,0,1,1)$	–	{0,1}	{0,1}	{0,1}	{0,1}	–	–	{0,1}	{0,1}	{0,1}

Table 6. One-round linear relations for MESH-128, and subkey restrictions.

one-round linear relation	odd-round subkeys				even-round subkeys			
	$Z_1^{(i)}$	$Z_3^{(i)}$	$Z_6^{(i)}$	$Z_8^{(i)}$	$Z_2^{(i)}$	$Z_4^{(i)}$	$Z_5^{(i)}$	$Z_7^{(i)}$
$(1,0,0,0,1,0,0,0) \rightarrow (1,1,0,0,0,0,0,0)$	{0,1}	–	–	–	–	–	{0,1}	–
$(0,1,0,0,0,1,0,0) \rightarrow (0,0,1,0,1,0,0,0)$	–	–	{0,1}	–	{0,1}	–	–	–
$(0,0,1,0,0,0,1,0) \rightarrow (0,0,0,1,0,1,0,0)$	–	{0,1}	–	–	–	–	–	{0,1}
$(0,0,0,1,0,0,0,1) \rightarrow (0,0,0,0,0,0,1,1)$	–	–	–	{0,1}	–	{0,1}	–	–
$(1,1,0,0,1,1,0,0) \rightarrow (1,1,1,0,1,0,0,0)$	{0,1}	–	{0,1}	–	{0,1}	–	{0,1}	–
$(1,0,1,0,1,0,1,0) \rightarrow (1,1,0,1,0,1,0,0)$	{0,1}	{0,1}	–	–	–	–	{0,1}	{0,1}
$(1,0,0,1,1,0,0,1) \rightarrow (1,1,0,0,0,0,1,1)$	{0,1}	–	–	{0,1}	–	{0,1}	{0,1}	–
$(0,1,1,0,0,1,1,0) \rightarrow (0,0,1,1,1,1,0,0)$	–	{0,1}	{0,1}	–	{0,1}	–	–	{0,1}
$(0,1,0,1,0,1,0,1) \rightarrow (0,0,1,0,1,0,1,1)$	–	–	{0,1}	{0,1}	{0,1}	{0,1}	–	–
$(0,0,1,1,0,0,1,1) \rightarrow (0,0,0,1,0,1,1,1)$	–	{0,1}	–	{0,1}	–	{0,1}	–	{0,1}
$(1,1,1,0,1,1,1,0) \rightarrow (1,1,1,1,1,1,0,0)$	{0,1}	{0,1}	{0,1}	–	{0,1}	–	{0,1}	{0,1}
$(1,1,0,1,1,1,0,1) \rightarrow (1,1,1,0,1,0,1,1)$	{0,1}	–	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	–
$(1,0,1,1,1,0,1,1) \rightarrow (1,1,0,1,0,1,1,1)$	{0,1}	{0,1}	–	{0,1}	–	{0,1}	{0,1}	{0,1}
$(0,1,1,1,0,1,1,1) \rightarrow (0,0,1,1,1,1,1,1)$	–	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	–	{0,1}
$(1,1,1,1,1,1,1,1) \rightarrow (1,1,1,1,1,1,1,1)$	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}

References

1. Aoki,K., Ichikawa,T., Kanda,M., Matsui,M., Moriai,S., Nakajima,J., Tokita,T.: Camellia: a 128-bit Block Cipher Suitable for Multiple Platforms, 1st NESSIE Workshop, Heverlee, Belgium, Nov. 2000.
2. Biham, E., Biryukov, A., Shamir, A., Miss-in-the-Middle Attacks on IDEA, Khufu and Khafre, In: Knudsen, L.R. (ed.): 6th Fast Software Encryption Workshop, LNCS, Vol. 1636. Springer-Verlag (1999), 124–138.
3. Biryukov,A., Wagner,D.: Slide Attacks, In: Knudsen,L.R. (ed): 6th Fast Software Encryption Workshop, LNCS, Vol. 1636. Springer-Verlag (1999), 245–259.
4. Borst,J., Knudsen,L.R., Rijmen,V.: Two Attacks on Reduced IDEA, In: Fumy, W. (ed.): Advances in Cryptology, Eurocrypt'97, LNCS, Vol. 1233. Springer-Verlag (1997), 1–13.
5. Daemen, J.: Cipher and Hash Function Design – Strategies based on Linear and Differential Cryptanalysis," PhD Dissertation, Dept. Elektrotechniek, Katholieke Universiteit Leuven, Belgium, Mar. 1995.
6. Daemen, J., Govaerts, R., Vandewalle, J.: Weak Keys for IDEA, In: Stinson, D.R. (ed.): Advances in Cryptology, Crypto'93, LNCS, Vol. 773. Springer-Verlag (1994), 224–231.
7. Daemen, J., Knudsen,L.R., Rijmen, V.: The Block Cipher SQUARE, In: Biham,E. (ed.): 4th Fast Software Encryption Workshop, LNCS, Vol. 1267. Springer-Verlag (1997), 149–165.
8. Daemen,J., Rijmen, V.: The Design of Rijndael – AES – The Advanced Encryption Standard, Springer-Verlag, 2002.
9. Demirci,H.: Square-like Attacks on Reduced Rounds of IDEA, In: Nyberg,K., Heys,H. (eds.): 9th Selected Areas in Cryptography Workshop, SAC'02, LNCS, Vol. 2595. Springer-Verlag (2002), 147–159.

10. Knudsen,L.R., Wagner,D.: Integral Cryptanalysis, In: Daemen, J., Rijmen,V. (eds): 9th Fast Software Encryption Workshop, LNCS, Vol. 2365. Springer-Verlag (2002), 112–127.
11. Lai,X., Massey, J.L., Murphy,S.: Markov Ciphers and Differential Cryptanalysis, In: Davies, D.W. (ed.): Advances in Cryptology, Eurocrypt'91, LNCS, Vol. 547. Springer-Verlag (1991), 17–38.
12. Menezes, A.J., van Oorschot, P.C., Vanstone, S., Handbook of Applied Cryptography, CRC Press, 1997.
13. Rijmen, V., Preneel, B., De Win, E.: On Weaknesses of Non-Surjective Round Functions, In: Design, Codes and Cryptography, Vol. 12, number 3, Nov. 1997, 253–266.