# Black Box Cryptanalysis of Hash Networks Based on Multipermutations

C.P. Schnorr[1] and S. Vaudenay[2]

[1] Fachbereich Mathematik/Informatik, Universität Frankfurt, Postfach 111932, 60054 Frankfurt a.M., e-mail: schnorr@informatik.uni-frankfurt.de

[2] Dép. Math. Inf., ENS Paris, 45 Rue d'Ulm, 75230-05 Paris, e-mail: vaudenay@dmi.ens.fr

**Abstract.** Black box cryptanalysis applies to hash algorithms consisting of many small boxes, connected by a known graph structure, so that the boxes can be evaluated forward and backwards by given oracles. We study attacks that work for any choice of the black boxes, i.e. we scrutinize the given graph structure. For example we analyze the graph of the fast Fourier transform (FFT). We present optimal black box inversions of FFT–compression functions and black box constructions of collisions. This determines the minimal depth of FFT– compression networks for collision–resistant hashing. We propose the concept of *multipermutation*, which is a pair of *orthogonal latin squares*, as a new cryptographic primitive that generalizes the boxes of the FFT. Our examples of multipermutations are based on the operations circular rotation, bitwise xor, addition and multiplication.

## 1 Introduction and Surview

The security of cryptographic hash functions is a challenging problem because there are no proofs of security in terms of complexity theoretic lower bounds. We introduce *black box cryptanalysis* as a new computational model where meaningful lower bounds are possible. The goal is to prove complexity theoretic lower bounds that are of practical relevance. Here we present upper bounds which we believe are optimal. For the inversion problem we already have matching lower bounds. This shows that any fast cryptanalytic attack must use the interior structure of the boxes. One can hope that this yields hash functions that are cryptographically strong except for a weak choice of the boxes. Various known cryptographic attacks fall into our framework. These are meet-in-the-middle-attacks and other applications of the birth day paradox. The global analysis of our approach complements linear and differential cryptanalysis.

We require that the black boxes perform multipermutations for some set $E$, a condition that can easily be generalized. We call a permutation $B : E^2 \to E^2$, $B(a,b) = (B_1(a,b),\ B_2(a,b))$, a *multipermutation* if for every $a, b \in E$ the mappings $B_i(a,*), B_i(*,b)$ for $i = 1,2$ are permutations on $E$. Multipermutations are a basic cryptographic primitive for perfect generation of diffusion and confusion. They correspond to pairs of *orthogonal latin squares* . The boxes of the classical FFT are multipermutations for a finite ring $E$ having a root of unity of order $2^k$ for some k. We present multipermutations for arbitrary finite dimensional vector spaces $E = \mathbb{F}^m$ over the Galois field $\mathbb{F} = \{0,1\}$ of order 2. Examples of interest are $\mathbb{F}^8, \mathbb{F}^{16}, \mathbb{F}^{32}$. Our examples of multipermutations use the operations circular rotation, bitwise xor, addition and multiplication. They are generated from *orthomorphisms* for the group $(\mathbb{F}^m, \oplus)$ where $\oplus$ is the bitwise xor.

For example we scrutinize the graph structure of the generalized FFT. We consider FFT–networks of order $2^k$ that can have for fixed k arbitrarily many layers. We present optimal black box attacks for inverting the compression function and for the construction of collisions. This also determines the minimal number of layers (rounds) that is necessary for collision–resistant hashing via FFT–networks. Based on this analysis SCHNORR, VAUDENAY (1993) propose example hash algorithms that counter the most efficient attacks of this paper. In particular compression is confined to the final stage of the hashing. This yields a highly parallel, scalable family of collision–resistant hash functions.

We present in section 2 a family of compression functions based on FFT–networks. In section 3 we study attacks for inverting these compression functions in the case that the "black box"–multipermutations are given by oracles. Constructions of collisions are given in section 4. In section 5 we give examples of multipermutations for the case that $E$ is a linear space over the field $\mathbb{F}$.

## 2 A Family of Compression Functions

**Notation. 1.** We let $\mathbb{F}$ denote the Galois field of order 2 and let $E = \mathbb{F}^m$ be the linear space over the field $\mathbb{F}$ with dimension $m$. We call the elements of $E$ *words*.
**2.** Let $k$ be a fixed integer, $k > 0$ and let $i \in \{0, \ldots, 2^k - 1\}$. Let $i_j \in \{0,1\}$ for $j = 0, \ldots, k-1$ denote the j–th bit of $i = i_0 + 2i_1 + \ldots + 2^{k-1} i_{k-1}$. Let the integer $i(j)$ be obtained from $i$ by negating the bit $i_j$. For our purposes it is convenient to define for $j \geq k$ that $i_j = i_{j (\mathrm{mod}\, k)}$ , $i(j) = i(j \bmod k)$ .

**The compression function $\mathbf{g_{k,s}} : \mathbf{E^{2^k}} \to \mathbf{E^{2^{k-1}}}$.**

`INPUT` $e_i \in E$ for $i = 0, \ldots, 2^k - 1$

(We call $H = [e_i \mid i_0 = 0]$ the *hash input* and $M = [e_i \mid i_0 = 1]$ the *message input*)

`FOR` $j = 0, \ldots, s$ `DO`

$(e_i, e_{i(j)}) := B_{i,j}(e_i, e_{i(j)})$ for all $i$ with $(0 \leq i < 2^k$ and $i_j = 0)$ in parallel

`OUTPUT` $g_{k,s}(H, M) = [e_i \mid i_s = 0] \in E^{2^{k-1}}$

**The choice of the boxes $\mathbf{B_{i,j}}$ and the integers $\mathbf{s, k}$.** We require that the boxes $B_{i,j}$ perform multipermutations for the set $E$. We call a permutation $B : E^2 \to E^2$, $B(a,b) = (B_1(a,b),\ B_2(a,b))$, a *multipermutation* (for $E$) if for every fixed $a, b \in E$ the mappings $B_i(a,*)$, $B_i(*,b)$, for $i = 1,2$, are permutations on $E$. Thus the component mappings $B_i : E^2 \rightarrow E \quad i = 1, 2$ represent both *latin squares* ( i.e. *bipermutations* ), they act as a permutation on both inputs. A permutation $B : E^2 \rightarrow E^2$ is a multipermutation iff both component mappings $B_1, B_2$ represent latin squares, or equivalently if $(B_1,\ B_2)$ represents a pair of orthogonal latin squares.

It is important that the message inputs $e_{2i+1}$ and hash inputs $e_{2i}$ are mixed by the boxes $B_{i,0}$ of the first round $j = 0$. The hash outputs are from distinct boxes $B_{i,s}$ of the last round $j = s$. It may be of interest that $g_{k,s}$ transforms the uniform distribution on $E^{2^k}$ into the uniform distribution on $E^{2^{k-1}}$. This is because the boxes $B_{i,j}$ perform permutations on $E^2$.

We can represent the algorithm $g_{k,s}$ by a network. It consists of $s+1$ layers $j = 0, \ldots, s$. Layer $j$ has $2^{k-1} - 1$ vertices $B_{i,j}$ for $i = 0, \ldots, 2^k - 1$ with $i_j = 0$. In Figure 1 (standard numbering) vertex $B_{i,j}$ is represented by the integer $i$ and the edge $e_{i,j}$ is marked with $i$. The edges of the $g_{k,s}$-network correspond to the inputs/outputs $e_i, e_{i(j)}$ of $B_{i,j}$. Edges corresponding to hash inputs and hash outputs are in bold–face. More precisely we let $e_{i,j}, e_{i(j),j}$ denote the inputs of $B_{i,j}$ and $e_{i,j+1}, e_{i(j),j+1}$ the outputs of $B_{i,j}$, i.e. we have for $j = 0, \ldots, s$: $(e_{i,j+1}, e_{i(j),j+1}) = B_{i,j}(e_{i,j}, e_{i(j),j})$ for $(0 \leq i < 2^k$ with $i_j = 0)$.

The hash input is $H = [e_{i,0} \mid i_0 = 0]$, the hash output is $g_{k,s}(H, M) = [e_{i,s+1} \mid i_s = 0]$.

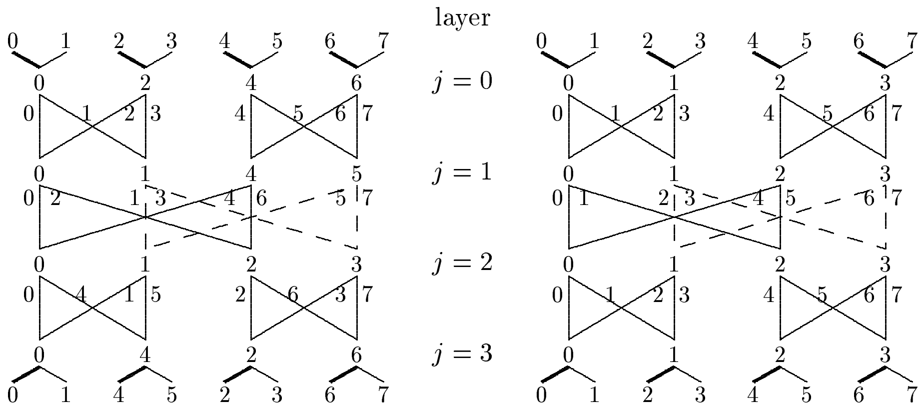


figure 1: $g_{3,3}$–network
in standard numbering

figure 2: $g_{3,3}$–network
in successive numbering

In all our examples we will use the successive numbering, shown in Figure 2, where vertices and edges of each layer are numbered from left to right in increasing order starting with $i = 0$. For the successive numbering we let $\bar{B}_{i,j}$ denote the vertex $i$ of layer $j$ and we let $\bar{e}_{2i,j+1}, \bar{e}_{2i+1,j+1}$ denote the output edges of $\bar{B}_{i,j}$.

By iterating the compression function $g_{k,s}$ we can transform arbitrary binary messages into a hash value in $E^{2^{k-1}}$ that is $m2^{k-1}$ bits long. We require that a given message, consisting of $t$ bits, is padded so that its bit length becomes a multiple of $m2^{k-1}$. We recommend to append to the message a single "1" followed by a suitable number of "0" bits followed by the binary representation of $t$. So the padded message $M = M_1 M_2 \cdots M_n$ consists of $n$ blocks $M_1, \ldots, M_n \in E^{2^{k-1}}$, $n = \lceil (t + 1 + \lceil \log_2(t+1) \rceil)/m2^{k-1} \rceil$.

**The iterative hash function $\mathbf{h_{k,s}}$**

`INPUT` $M = M_1 \cdots M_n \in E^{n \cdot 2^{k-1}}$ (the padded message)

Fix an initial value $H_0 \in E^{2^{k-1}}$

$H_i := g_{k,s}(H_{i-1}, M_i)$ for $i = 1, \ldots, n$

`OUTPUT` $h_{k,s}(M) := H_n$ (the hash value of $M$)

However we advise against iterating a compression function. It yields a rather weak hash function as follows from the surprisingly efficient attacks presented in the next sections. As a consequence Schnorr and Vaudenay (1993) propose parallel FFT–hashing without iterating a compression function.

## 3 Black Box Inversion of $\mathbf{g_{k,s}}$

The problem of inverting $g_{k,s}$ is as follows: Given random, independent $H, H' \in E^{2^{k-1}}$ find $M \in E^{2^{k-1}}$ satisfying $g_{k,s}(H, M) = H'$.

The randomness of $H, H'$ can be replaced by assuming that the boxes $B_{i,j}$ for $H, H'$ are random. The latter assumption is justified since black box analysis examines the $g_{k,s}$-network without that the boxes $B_{i,j}$ have been specified.

It is necessary for collision resistance of $h_{k,s}$ that the problem of inverting $g_{k,s}$ is infeasible. We study inversion algorithms for which the multipermutations $B_{i,j}$ in the $g_{k,s}$–network are "black boxes". We call $B : E^2 \to E^2$, $B(a,b) = (B_1(a,b), B_2(a,b))$ an *oracle–multipermutation* (omp) if we are given oracles that compute, for arbitrary $a, b \in E$, the permutations $B_i(a,*)$, $B_i(*,b)$ and the inverse permutations $B_i^{-1}(a,*)$, $B_i^{-1}(*,b)$, $i = 1, 2$, on $E$.

**Lemma 1.** *Let $B : E^2 \to E^2$ be an omp, $B(a,b) = (u,v)$. Then any two words out of $a, b, u, v$ determine the other two by the given oracles.*

**Proof.** Given $a, b$ we compute $u, v$ via (the oracle for) the permutation $B$. Given $u, v$ we compute $a, b$ via $B^{-1}$. Given $a, u$ we first evaluate the inverse permutation $B_1^{-1}(a,*)$ with input $u$, this yields $b$. Then we recover $v$ from $B(a,b)$. The cases that we are given $a, v$ or $b, u$ or $b, v$ are symmetric. □

In the following we assume that the boxes $B_{i,j}$ of the $g_{k,s}$–network are omp's. We say that the vertex $B_{i,j}$ has *degree of freedom* 2, i.e. any two input/output edges of $B_{i,j}$ determine all the other edges.

**Resolving the $\mathbf{g_{k,s}}$–network.** In order to solve the equation $g_{k,s}(H, M) = H'$ for given $H, H'$ we guess some edges $e_{i,j}$ which together with $H, H'$ determine by successive application of Lemma 1 all edges of the network. A *resolution* of the $g_{k,s}$–network consists of a sequence of steps of the following types:

- *guess an edge:* Pick a position $(i,j)$ of an indeterminate edge $e_{i,j}$ and try for $e_{i,j}$ all values in $E$.
- *resolve a vertex:* Pick a position $(i,j)$ of an unresolved vertex $B_{i,j}$ and determine via Lemma 1 all its edges $e_{i,j}$, $e_{i(j),j}$, $e_{i(j),j+1}$, $e_{i(j),j+1}$ from two known edges.

The resolution of the network terminates when all the vertices are resolved and all the edges are determined. We let $C_\nu$ denote the set of *correct edge assignments* for step $\nu$. To define $C_\nu$ let $T_\nu$ denote the set of positions $(i,j)$ so that $e_{i,j}$ has either been fixed or determined in one of the first $\nu$ steps. Then $C_\nu$ consists of the assignments $(e_{i,j} \in E \mid (i,j) \in T_\nu) \in E^{\#T_\nu}$ for which all the boxes $B_{i,j}$ with $(i,j) \in T_\nu$ are correct, i.e. $(e_{i,j+1}, e_{i(j),j+1}) = B_{i,j}(e_{i,j}, e_{i(j),j})$ for all $(i,j) \in T_\nu$ .

The *average complexity* of step $\nu$, notation $ac_\nu$, is defined to be the expected size of $C_\nu$, where the probability space is the set $E^{\#T_\nu}$ of all assignments $(e_{i,j} \mid (i,j) \in T_\nu)$ with uniform probability distribution.

The *average complexity* of the resolution is $\max_\nu ac_\nu$, the maximal average step complexity over all steps. This is the average time for the resolution, assuming that the time to resolve all boxes in the network is one time unit.

There is a simple calculus for computing $ac_\nu$. Initially we have $ac_0 = 1$. If we guess in step $\nu$ a new edge we have $ac_\nu = ac_{\nu-1} 2^m$. If we resolve in step $\nu$ a box $B_{i,j}$, with $\ell \geq 2$ of its edges known, we have $ac_\nu = ac_{\nu-1}\ 2^{m(2-\ell)}$ . This holds because the guessed edges are mutually independent and since this independence is preserved during the process of resolution. If we are given $\ell \geq 3$ edges of the box then we call the box *overdetermined.* The exceptional case that we are given $\ell = 4$ edges of an unresolved box $B_{i,j}$ does not occur in our examples. Thus if step $\nu$ resolves an overdetermined box $B_{i,j}$ we have $ac_\nu = ac_{\nu-1}\ 2^{-m}$ .

If we resolve the $g_{k,s}$-network, with some edges initially fixed, the calculus for $ac_\nu$ remains valid if the fixed edges in $E$ are chosen uniformly at random. Thus for inverting $g_{k,s}$ we assume that the given $(H,H')$ is random in $E^{2^k}$.

**Inverting $\mathbf{g_{k,s}}$.** In order to solve for given, random $H, H' \in E^{2^{k-1}}$ the equation $g_{k,s}(H,M) = H'$ we evaluate the network for $g_{k,s}$ with the edges for $H, H'$ correctly representing the given $H, H'$. We require that the sets $T_\nu$ for $\nu = 1, 2, \ldots$ contain the set $T_0 = \{(2i,0), (2i, s+1) \mid i = 0, \ldots, 2^{k-1}-1\}$ of positions of the edges $e_{2i,0}$ for $H$ and $e_{2i,s+1}$ for $H'$. We also require that the assignments in $C_\nu \subset E^{\#T_\nu}$ correctly represent $H, H'$. In particular $C_0$ consists of the unique assignment in $E^{\#T_0}$ that represents $H, H'$.

**Inverting $\mathbf{g_{4,4}}$ in average time $\mathbf{2^{5m}}$.** Consider the $g_{4,4}$–network without input/output edges. The vertices $B_{i,0}, B_{i,4}$ of layers $j = 0, 4$ have degree of freedom 1 since we are given $H, H'$. The other vertices have degree of freedom 2.

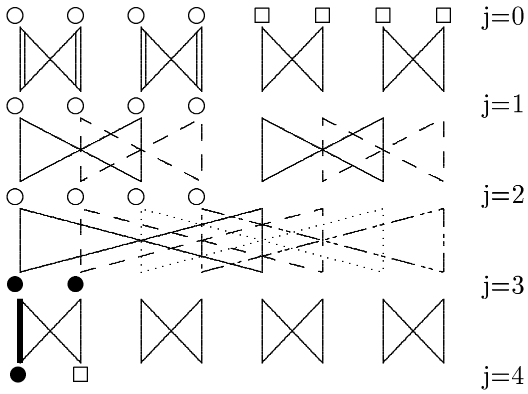

figure 3: $g_{4,4}$–network

To invert $g_{4,4}$ we guess the four double edges. With these edges we can resolve the 12 ○–boxes. Next we guess the bold–faced edge $e_{0,4}$ and we resolve three ●–and one □–boxes at the bottom. The □–box is overdetermined which reduces the average step complexity $ac_\nu$ from $2^{5m}$ to $2^{4m}$. In the same way we successively resolve the other three butterflies with vertices $B_{i,3}, B_{i,4}$. Each time we guess one of its edges we resolve all four boxes and we reduce the average step complexity from $2^{5m}$ to $2^{4m}$ via an overdetermined box. After resolving all boxes of layers $j = 4$ and $j = 3$ we can resolve the remaining boxes of layers $j = 2, 1, 0$. The □–boxes $B_{i,0}$ $i = 4, 5, 6, 7$ are overdetermined. Thus the average number of solutions $M$ is 1. □

**Inverting $\mathbf{g_{4,3}}$ in maximal time $\mathbf{2^{4m}}$.** The above resolution shows that $g_{4,3}$ can be resolved in maximal time $2^{4m}$. In the $g_{4,3}$–network the boxes of layer $j = 3$ have degree of freedom 1 since $H'$ is given. Thus after resolving the ○–boxes we can resolve all boxes of layer $j = 3$ and then all the other boxes. □

We define the *inversion complexity* $I(k, s)$ as the minimal complexity of black box inversions for $g_{k,s}$. We let $LI(k, s) = \log_{2^m} I(k, s)$ denote its logarithm to base $2^m$ . So far we have shown that $LI(4, 3) \leq 4$, $LI(4, 4) \leq 5$, $LI(4, 5) \leq 6$. In general we have the following upper bounds. Matching lower bounds exist for $L(k, k-1)$ and will appear in a subsequent paper.

**Theorem 2.** (1) $LI(1, 0) = 0$, (2) $LI(k, k-1) \leq 2^{k-2}$ *for* $k \geq 2$, (3) $LI(k, k-1+t) \leq 2^{k-2} + 2^{t-1}$ *for* $0 \leq t \leq k-1$ *and* $k \geq 3$.

**Proof. 1.** The $g_{1,0}$–network consists of a single multipermutation with two given edges.
**2.** The $g_{k,k-1}$–network consists of $k$ layers $j = 0, \dots, k-1$. The boxes of layers 0 and $k-1$ have degree (of freedom) 1 since $H, H'$ is given. We guess the first half of the message words consisting of the edges $e_{2i,0}$ for $i = 0, \dots, 2^{k-2}-1$. Then we resolve, from the top to the bottom, the first half of the layers $j = 0, \dots, k-2$

consisting (in successive enumeration) of the boxes $\bar{B}_{i,j}$ for $i = 0, \ldots, 2^{k-1} - 1$. Since the boxes $\bar{B}_{i,k-1}$ have degree 1 we can now resolve layer $k-1$ and then the remaining second half of layers in the order $j = k-2, \ldots, 0$.

**3.** The $g_{k,k-1+t}$–network has $k+t$ layers $j = 0, \ldots, k-1+t$. We first resolve, as under (2), the first half of layers $j = 0, 1, \ldots, k-2$. This reduces, for the boxes in layer $k-1$, the degree to 1. This part is equivalent to inverting $g_{k,k-1}$ and has complexity $I(k, k-1)$. Now the boxes of layers $j = k-1$ and $j = k-1+t$ have degree 1. Thus resolving layers $j = k-1, \ldots, k-1+t$ can be done by performing iteratively $2^{k-t-1}$ many $g_{t+1,t}$–inversions. The latter task has complexity $I(t+1, t)$. Finally we resolve the second half of the layers $j = k-2, \ldots, 0$. We see that $I(k, k-1+t) \leq I(k, k-1) \cdot I(t+1, t)$ which proves the claim by inequality (2). $\square$

## 4 Black Box Collisions for $\mathbf{g_{k,s}}$

The collison problem for $g_{k,s}$ is as follows: Given random, non independent $H, H' \in E^{2^{k-1}}$ find $M, M' \in E^{2^{k-1}}$ satisfying $g_{k,s}(H, M) = g_{k,s}(H', M')$.

A solution of this problem with $H = h_{k,s}(\bar{M})$ and $H' = h_{k,s}(\bar{M}')$ yields a collision $h_{k,s}(\bar{M}M) = h_{k,s}(\bar{M}', M')$ of the hash function $h_{k,s}$.

The collision construction below is for the case $H = H'$. It can easily be extended to the case of random, not independent $H, H'$. We first construct collisions for $g_{4,4}$ in average time $2^{3m}$, then we consider arbitrary $g_{k,s}$. We assume that the time to resolve all boxes in the network is 1.

**The method for producing $\mathbf{g_{k,s}}$–collisions.** Let $\mathtt{OUT} = \{2i \mid i = 0, 1, \ldots, 2^{k-1} - 1\}$ be the set of indices $i$ of hash outputs $\bar{e}_{i,s}$ of $g_{k,s}$ . In order to construct a collision we choose a suitable subset $S \subset \mathtt{OUT}$ , we pick random values $\bar{e}_{i,s} \in E$ for $i \in S$ , and we solve for the given $H$, $(\bar{e}_{i,s} \mid i \in S)$ the equation

$$g_{k,s}(H, M) = (\bar{e}_{i,s} \mid e \in \mathtt{OUT}) \tag{1}$$

with indeterminates $M$ and $\bar{e}_{i,s}$ for $i \in \mathtt{OUT} - S$ . We solve equation (1) by evaluating the $g_{k,s}$–network, as described in section 3, with the edges corresponding to $H$ and the edges $(\bar{e}_{i,s} \mid i \in S)$ already determined. This way we generate $\#E^{|\mathrm{OUT}-S|/2}$ random solutions of (1). By the birthday paradox, applied to the $\bar{e}_{i,s}$ with $i \in \mathrm{OUT} - S$ , this yields a $g_{k,s}$–collision with $H = H'$. The average time for producing a collision is $\#E^{|\mathrm{OUT}-S|/2}$ times the average time to solve equation (1).

**Construction of collisions for $\mathbf{g_{4,4}}$ in average time $\mathbf{2^{3m}}$.** Consider in figure 4 the network for $g_{4,4}$ without input/output edges. Its hash outputs are $\bar{e}_{2i,5}$ for $i = 0, 1, \ldots, 7$.

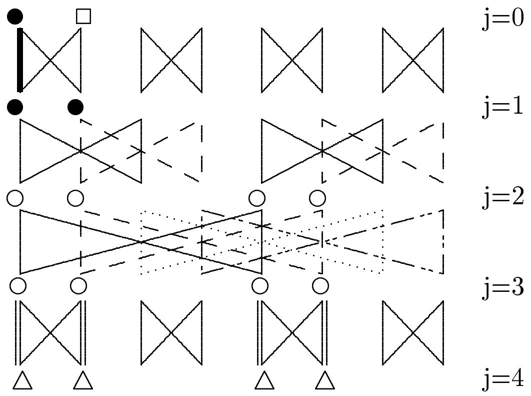


figure 4: $g_{4,4}$–network

We pick random elements $(\bar{e}_{i,5} \mid i = 0, 2, 8, 10) \in E^4$ , and we solve for these values and the given $H$ the equation

$$g_{4,4}(H, M) = (\bar{e}_{2i,5} \mid i = 0, 1, \dots, 7) \tag{2}$$

with indeterminates $H$, $\bar{e}_{i,5}$ for $i = 4, 6, 12, 14$. We generate $2^{2m}$ random solutions of equation (2). By the birthday paradox this yields distinct messages $M, M'$ with $g_{4,4}(H, M) = g_{4,4}(H, M')$. The average time for collision construction is $2^{2m}$ times the average time to solve equation (2).

**Solving equation (2) in average time $\mathbf{2^m}$.** The boxes $\bar{B}_{i,0}$ $i = 0, \dots, 7$ have degree of freedom 1 since $H$ is given. The $\triangle$–boxes $\bar{B}_{i,4}$ $i = 0, 1, 4, 5$ have degree of freedom 1 since $\bar{e}_{i,5}$ $i = 0, 2, 8, 10$ are known.

We next choose random values for the double edges $\bar{e}_{i,4}$ $i = 0, 3, 8, 11$ and we keep these values fixed during the resolution of the $g_{4,4}$–network.

**Resolution of the $\mathbf{g_{4,4}}$–network for the given $H$; $\mathbf{\bar{e}_{i,5}}$ $\mathbf{i = 0, 2, 8, 10}$; $\mathbf{\bar{e}_{i,4}}$ $\mathbf{i = 0, 3, 8, 11}$.** Resolve the $\triangle$–boxes $\bar{B}_{i,4}$ $i = 0, 1, 4, 5$ , the $\circ$–boxes $\bar{B}_{i,3}$ $i = 0, 1, 4, 5$ and the $\circ$–boxes $\bar{B}_{i,2}$ $i = 0, 1, 4, 5$. Now we know one edge of each of the boxes $\bar{B}_{i,1}$ $i = 0, 1, \dots, 7$.

Next we guess the bold–faced edge $\bar{e}_{0,1}$ and we resolve the $\bullet$–boxes $\bar{B}_{0,0}, \bar{B}_{0,1}, \bar{B}_{1,1}$. Then we resolve the overdetermined $\square$–box $\bar{B}_{1,0}$. This reduces the average step complexity $ac_\nu$ from $2^m$ to 1.

Similar to $\bar{e}_{0,1}$ we can guess successively the edges $\bar{e}_{4,1}, \bar{e}_{8,1}, \bar{e}_{12,1}$ , and we reduce each time the average step complexity $ac_\nu$ from $2^m$ to 1 by resolving the overdetermined boxes $\bar{B}_{3,0}, \bar{B}_{5,0}, \bar{B}_{7,0}$.

Finally we resolve all other boxes. None of these boxes is overdetermined.

This resolution of the $g_{4,4}$–network has average complexity $\max_\nu ac_\nu = 2^m$, and it finds on the average $ac_n = 1$ solutions of equation (2). The probability space is the set $E^{16}$ of all $(H;\ \bar{e}_{i,5}\ \ i = 0, 2, 8, 10;\ \bar{e}_{i,4}\ \ i = 0, 3, 8, 11)$ with uniform probability distribution. Thus we can find a solution of equation (2) in average time $2^m$ (times the costs to resolve all boxes of the network). $\square$

With the same method we can construct collisions for $g_{4,3}$ and $g_{4,5}$ in average time $2^{2m}$ and $2^{4m}$. Thus, for $g_{4,s}$ to be collision–resistant with $m = 16$ we must have $s \geq 5$.

Let the *collision complexity* $C(k, s)$ be defined as the minimal complexity of black box collisions for $g_{k,s}$. Let $LC(k, s) = \log_{2^m} C(k, s)$ denote its logarithm. We have the following inequalities which we believe to be tight.

**Theorem 3.** *$LC(k, k-1+t) \leq 2^{k-3} + 2^{t-1}$ for $0 \leq t \leq k-2$ and $k \geq 3$.*

**Proof.** To construct a collision for $g_{k,k-1+t}$ we fix at random half of the hash outputs $e_{2i,k+t}$ and we repeatedly resolve the $g_{k,k-1+t}$–network after fixing at random half of the dead outputs $e_{2i+1,k+t}$. Applying the birth day paradox to the $2^{k-2}$ many free hash outputs yields a collision after resolving the network on the average $2^{k-3}$ times.

It remains to show that the complexity for resolving the $g_{k,k-1+t}$– network, after fixing $2^{k-1}$ output edges, is at most $I(t+1, t)$. For this we choose the positions $2i$ of the fixed hash outputs $e_{2i,k+t}$ and those of the dead outputs $e_{2i+1,k+t}$ such that half of the layers $j = k+t, \ldots, t+1$ can be evaluated. After this evaluation the boxes of layer $t$ have all degree 1. The remaining resolution of layers $j = t, \ldots, 0$ can be done by performing iteratively $2^{k-t+1}$ many $g_{t+1,t}$– inversions. This shows $C(k, k-1+t) \leq 2^{m2^{k-3}} I(t+1, t)$ which proves the claim by Theorem 2. $\square$

## 5 Examples of Multipermutations and Orthomorphisms

For $E = \mathbb{F}^m$ we introduce multipermutations on $E^2$ that are based on the operations $\oplus$ (bitwise xor), $\wedge$ (bitwise and), $+$ (addition modulo $2^m$), $\cdot$ (multiplication modulo $2^m$), $*$ (multiplication in $\mathbb{Z}^*_{2^m+1}$) and the circular rotation $\mathbf{R}$ on $E$ to the right. For a particular proposal of multipermutations in hash algorithms see Schnorr, Vaudenay (1993).

We present multipermutations on $E^2$ of the particular form $(a, b) \mapsto (a * b, a * f(b))$, where $f$ is a permutation on $E$ and $(E, *)$ is a group. Obviously, this mapping is a permutation on (a,b) iff both $f$ and $b^{-1} * f(b)$ are permutations on $E$. For an arbitrary group $(G, *)$ (whether abelian or not) a permutation $f$ on $G$ is called an *orthomorphism* if $f$ and $f(a) * a^{-1}$ are both permutations on $G$. The term orthomorphism has been introduced by Johnson, Dulmage and Mendelsohn. Orthomorphisms have been studied under the name *complete mappings* by Hall and Paige. In fact $f$ is an orthomorphism iff $f^{-1}$ is a complete mapping. Hall and Paige show that a finite group $G$ of even order admits an orthomorphism only if its Sylow 2–subgroup is cyclic. This necessary condition is sufficient for solvable groups of even order. An interesting consequence of the necessary condition is that the group $(E, +)$, where $+$ is the addition modulo $2^m$, does not admit an orthomorphism. The group $(E, +)$ coincides with its Sylow 2–subgroup and is cyclic of order $2^m$ with generator 1.

One possibility of constructing orthomorphisms for $(E,*)$ consists of producing a homomorphism of $E$ that is an orthomorphism. In case of $E=\mathbb{F}^m$ and the group $(E,\oplus)$ one constructs a linear mapping $L$ such that $det(L)$ and $det(L-I)$ are both non-zero. We study for $L$ the circular rotation on $E$ by $\ell$ positions to the right, which we denote $\mathbf{R}^\ell : E \to E$ .

**Theorem 4.** *The mapping* $L_c : E^2 \to E^2$ , $(a,b) \mapsto (a \oplus b,\ a \oplus (b \wedge c) \oplus \mathbf{R}^\ell(b))$ *is a multipermutation for* $c \in E,\ \ell \in \mathbb{Z}$ *if and only if the iterates of* $\mathbf{R}^\ell$ *on* $c$ *take for each bit position both values* $0, 1$.

**Remarks. 1.** We see that if $L_c$ is a multipermutation then $c \notin \{0^m, 1^m\}$ since otherwise the bits of $c$ are constant and so are the bits of $\mathbf{R}^{\ell \cdot n}(c)$ for all $n$.
**2.** If $gcd(\ell, m) = 1$ , i.e. if $\ell$ is odd, then $L_c$ is a multipermutation if and only if $c \notin \{0^m, 1^m\}$. This is because, for odd $\ell$, the iterates of $\mathbf{R}^\ell$ carry every bit of $c$ to all positions.
**3.** Theorem 4 remains valid if $\mathbf{R}^\ell$ is replaced by any permutation of bit positions on $E$.

**Proof.** $L_c$ is a multipermutation if and only if both mappings

$$b \ \mapsto \ (b \wedge c) \oplus \mathbf{R}^\ell(b) \qquad b \ \mapsto \ b \oplus (b \wedge c) \oplus \mathbf{R}^\ell(b)$$

are permutations of $E$, i.e. $b \ \mapsto \ (b \wedge c) \oplus \mathbf{R}^\ell(b)$ is an orthomorphism fo the group $(E, \oplus)$. Now the claim follows from Lemma 5 with $d = c$ and $d = \bar{c}$ , the bitwise negation of $c$. For the second mapping we use that $b \oplus (b \wedge c) = b \wedge \bar{c}$ . □

**Lemma 5.** *For* $d \in E$ *the linear mapping* $f_d : b \ \mapsto \ (b \wedge d) \oplus \mathbf{R}^\ell(b)$ *is a permutation of* $E$ *if and only if the iterates* $\mathbf{R}^{\ell \cdot n}(d)$ *take for each bit position the* $0$*–bit for some* $n$.

Let $d = (d_0, \ldots, d_{m-1}) \in E = \{0,1\}^m$ and for $i \notin \{0, \ldots, m-1\}$ let $d_i = d_{i (\bmod m)}$. The claim means that $f_d$ is a permutation if and only if for every $i$ there is some $n$ with $d_{i+\ell n} = 0$.

**Proof.** Since $f_d$ is linear over $\mathbb{F}$ $f_d$ is a permutation if and only if $f_d(b) = 0$ implies $b = 0$. Now we show both directions of the claim

"$\Rightarrow$" (by contradiction) Assuming $f_d(b) = 0,\ b \neq 0$ we show that there is a position $i$ with $d_{i+\ell n} = 1$ for all $n$. We see from $f_d(b) = 0$ that $b_{i+\ell n} = b_{i+(\ell+1)n} \wedge d_{i+(\ell+1)n}$ for all $n$ .
If $b \neq 0$ and $b_i = 1$ these equalities imply $b_{i+\ell} = d_{i+\ell} = 1$ and we see by recursion that $b_{i+\ell n} = d_{i+\ell n} = 1$ for all $n$.

"$\Leftarrow$" Suppose that for some position $i$ we have $d_{i+\ell n} = 1$ for all $n$. Define $b = (b_0, \ldots, b_{m-1})$ by setting $b_j = 1$ iff $\exists i, n : j = i + \ell n$. We see that $f_d(b) = 0,\ b \neq 0$. □

**Further multipermutations** can be constructed by composition. The composition of a multipermutation $P : E^2 \to E^2$ with arbitrary permutations $\sigma_1, \sigma_2 : E \to E$ yields new multipermutations $P(\sigma_1(a), \sigma_2(b))$, $(\sigma_1 P_1(a,b),\ \sigma_2 P_2(a,b))$ . The inverse of a multipermutation is again a multipermutation.

For the permutations $\sigma_1, \sigma_2$ we can use the multiplication modulo $2^m$ with an odd integer in $E = \{0, \ldots, 2^m - 1\}$. Instead of the multiplication modulo $2^m$ we can as well use the binary operation $*$ on $E = \{0, \ldots, 2^m - 1\}$ defined as $a*b := (a'b' \bmod 2^m+1) \bmod 2^m$ where $a' := [\ 2^m$ if $a = 0$ and $a$ otherwise $]$. Lai and Massey (1990) use the operation $*$ in the case $m = 16$. If $2^m+1$ is prime, e.g. for $m = 8, 16$, the operation $*$ is invertible. Then $(E, *)$ is a cyclic group of order $2^m$ with neutral element 1. The group $(E, *)$ is isomorphic to $\mathbb{Z}^*_{2^m+1}$, the multiplicative group of residues modulo $2^m + 1$. We have an isomorphism $\varphi : (E, *) \to \mathbb{Z}^*_{2^m+1}$ , $a \mapsto a'$. In particular we have the

**Lemma 6.** *If $2^m+1$ is prime then every $c \in E$ defines a permutation $a \mapsto a*c$ on $E$.*

## References


**Baritaud, T., Gilbert, H., and Girault, M.:** F.F.T. Hashing is not Collision-free. Proceedings of EUROCRYPT'92, Springer LNCS 658 (1993), pp. 35–44.

**Hall, M. and Paige, L.J.:** Complete mappings of finite groups. Pac. J. Math., 5 (1955), pp. 541–549.

**Johnson, D.M., Dulmage, A.L., and Mendelsohn, N.S.:** Orthomorphisms of groups and orthogonal latin squares. I. Can. J. Math. 13, (1961), pp. 356–372.

**Lai, X. and Massey, J.L.:** A proposal of a new block encryption standard. Advances in Cryptology. Proceedings of EUROCRYPT'90. Springer LNCS 473, (1991), pp. 389–404.

**Schnorr, C.P.:** FFT–Hash II, efficient cryptographic hashing. Proceedings of EUROCRYPT'92. Springer LNCS 658 (1992), pp. 45–54.

**Schnorr, C.P. and Vaudenay, S.:** Parallel FFT–Hashing. Proceedings of Cambridge Security Workshop, Cambridge, December 9–11, 1993. to appear in Springer LNCS, Ed.: R. Anderson.

**Vaudenay, S.:** FFT–Hash II is not yet Collision–Free. Proceedings of CRYPTO'93, Springer LNCS 740 (1993), pp. 587–593.



This article was processed using the LaTeX macro package with LLNCS style