Increasing the Flexibility of the Herding Attack

Bart Mennink

Katholieke Universiteit Leuven, ESAT/COSIC and IBBT, Kasteelpark Arenberg 10, 3001 Leuven, Belgium
telephone +32-16-321800, fax +32-16-321969

Abstract

Chosen-target-forced-prefix (CTFP) preimage resistance is a hash function security property guaranteeing the inability of an attacker to commit to a hash function outcome h without knowing the prefix of the message to be hashed in advance. At EUROCRYPT 2006, Kelsey and Kohno described the herding attack against the Merkle-Damgård design that results in a CTFP-preimage of length about $n/3$ blocks in approximately $\sqrt{n} \cdot 2^{2n/3}$ compression function calls. Using an additional parameter ℓ, the attack can be sped-up at the cost of exponentially large preimages (the elongated herding attack). In this work, we re-investigate speed vs. message length tradeoffs for the herding attack. Using a third parameter d, we introduce the generalized *elongated multidimensional herding attack*. The parameters ℓ and d allow for full control over the efficiency of the attack versus the length of the preimages: increasing ℓ results in faster attacks with longer messages, while increasing d results in shorter messages with higher attack complexity. Using advanced methods in graph theory we analyze the complexity of the generalized attack, and we describe several variants for different values of ℓ, d. On the extreme, a CTFP-preimage of $2^{n/2}$ blocks can be found in $n \cdot 2^{n/2}$ queries. One can find a CTFP-preimage of length about $n/8$ blocks in $\sqrt[3]{n} \cdot 2^{3n/4}$ work.

Keywords: cryptography, hash functions, chosen-target-forced-prefix, generalized herding attack, hypergraphs

1. Introduction

Practical hash functions are traditionally built according to the Merkle-Damgård (MD) iterated design principle [9, 23]. Given a fixed-input-length compression function $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$, and an initial state value iv of n bits, a variable-input-length hash function hash is constructed as follows: an input message M is first injectively padded into message blocks of length m bits, and then these message blocks are compressed iteratively with the state using f. Although this design is adequate for collision and preimage resistance, second preimage resistance cannot be guaranteed [5, 10, 20]. Additionally, multicollisions can be found much faster than expected [18], and the design does not resist the length extension attack [8].

In 2006, Kelsey and Kohno [19] considered the chosen-target-forced-prefix (CTFP) preimage resistance of hash functions: an attacker commits to a digest h, and upon being challenged on a message prefix P (throughout, P is of length one message block, but the same results hold for larger prefixes P), he returns a preimage $M = P\|S$ for h, for some string S. Many hash function applications, most prominently commitment and signature schemes, rely on CTFP-preimage resistance of hash functions [24].

Ideally, if the hash function outputs strings of n bits, an attacker needs about 2^n hash function calls to obtain a CTFP-preimage. However, Kelsey and Kohno [19] introduced the *herding attack* that allows finding CTFP-preimages for the MD iterative design in approximately $\sqrt{k} \cdot 2^{(n+k)/2}$ offline (we follow the complexity analysis by Blackburn et al. [6]) and 2^{n-k} online compression function calls, and the attack results in preimages of length approximately k blocks. Here, the parameter k ensures a tradeoff between the offline and online complexity of the attack, with an optimal amount of work of $\sqrt{n} \cdot 2^{2n/3}$ calls achieved for $k = n/3$. It has been shown by Andreeva et al. [2] and Gauravaram et al. [14, 15] that many modes of operation derived from the classical MD construction suffer this herding attack or a variant of it. In [4], Andreeva and Mennink confirmed optimality of the herding attack by [19] and of most of the attacks by [2, 14, 15].

The herding attack of Kelsey and Kohno can be described as follows. Let $k \geq 0$ be any integer.

1. The attacker $\mathcal{A}$ constructs a *diamond* of k levels: he arbitrarily takes 2^k state values $h_0^{(1)}, \ldots, h_0^{(2^k)}$, and tries to find 2^{k-1} disjoint compression function collisions for these, by varying the message inputs. For the resulting 2^{k-1} state values $h_1^{(1)}, \ldots, h_1^{(2^{k-1})}$ he finds 2^{k-2} disjoint compression function collisions, etc., until he is left with one state value h_{diam} at level 0. By construction, starting from any state value $h_0^{(i)}$ he knows a path of k message blocks to h_{diam};
2. $\mathcal{A}$ computes a *commitment* $h = f(h_{\text{diam}}, M_{\text{pad}})$, where M_{pad} includes the length strengthening of the

Email address: `bmennink@esat.kuleuven.be` (Bart Mennink)

message;

3. $\mathcal{A}$ receives a *challenge* P, and computes $h_P = f(\mathsf{iv}, P)$;
4. $\mathcal{A}$ finds a message M_{link} such that $f(h_P, M_{\text{link}}) = h_0^{(i)}$ for some $i \in \{1, \ldots, 2^k\}$. He outputs a *CTFP-preimage* $P \| M_{\text{link}} \| M_{\text{diam}} \| M_{\text{pad}}$, where M_{diam} consists of the k message blocks that connect $h_0^{(i)}$ with h_{diam}.

Here, k provides a tradeoff between the offline complexity (step 1) and the online complexity (step 4). The resulting CTFP-preimage is of length $k+3$ message blocks (and the forged suffix is of length $k+2$). Kelsey and Kohno introduce also a generalization, the *elongated herding attack*, that allows to speed up the attack at the cost of larger preimages [19]. Let $\ell \geq 0$ be an integer. In step 1, the attacker appends a sequence of 2^ℓ compression function executions to each starting state value $h_0^{(i)}$ ($i = 1, \ldots, 2^k$), and the attack succeeds in step 4 if the attacker hits any of the $2^{k+\ell}$ state values[1]. This attack results in a preimage of length slightly larger than 2^ℓ blocks.

1.1. Our Contributions

We re-investigate the possibilities in the herding attack of Kelsey and Kohno to sacrifice message length for efficiency and vice versa. As explained above, the elongated herding attack [19] allows for faster attacks at the cost of larger preimages. In this work we investigate the possibility to *reduce* the length of the preimages at a reasonable price. A naive solution is to use the original herding attack, with values k for which the offline/online equilibrium is not achieved, but as we will show, better results can be obtained.

To this end we introduce the *elongated multidimensional herding attack* as a generalization to the classical herding attack. At a high level, the generalized attack differs from the original one in the sense that the internal diamond consists of d-way collisions only, for some integer $d \geq 2$. The generalized attack employs an *elongated multidimensional diamond* with parameters k, ℓ, d. Here, k is the number of levels in the diamond, ℓ is the length of the tails attached to each end node of the diamond, and d prescribes the type of collisions within the diamond. Using advanced methods in graph theory, we compute the complexity of the attack: a preimage of about $2^\ell + k + \ell$ blocks is found in approximately $2^{\ell + k\log(d)} + \frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}} + \ell \cdot 2^{n/2+1} + 2^\ell$ offline and $2^{n-\ell-k\log(d)}$ online compression function executions.

The elongated multidimensional herding attack is a uniform attack description that covers a wide variety of herding attack variants. The original attack is naturally covered for parameters $(\ell, d) = (0, 2)$. Increasing the tail length ℓ results in faster attacks with larger messages, which corresponds to the elongated herding attack. Increasing d results in shorter messages for slightly larger costs, and we refer to it as the *multidimensional herding attack*. In an entirely different setting, namely for finding second preimages on dithered hash functions, the approach of increasing d has appeared before in Andreeva et al. [3, Sect. 6.1]. We note that our findings improve the bound of Andreeva et al. considerably.

For various values of ℓ, d, we summarize the complexity of the attack, where k is used to obtain an offline/online equilibrium. With respect to the elongated herding attack, one needs $n \cdot 2^{n/2}$ compression function calls to find a preimage of length $2^{n/2}$ blocks. Approximately $\sqrt{n} \cdot 2^{31n/48}$ calls are needed to find a preimage of size about $2^{n/16}$. Not surprisingly, the attack complexities of the elongated herding attack variants meet the security bound derived in [4]. Using the multidimensional herding attack, one can obtain messages of length about $n/8$ blocks in $\sqrt[3]{n} \cdot 2^{3n/4}$ work, much faster than when using the traditional herding attack. Several other variants of the attacks are summarized in Tables 1 and 2.

1.2. Outline

We present background information on cryptography and graph theory in Sect. 2. The elongated multidimensional diamond is described and analyzed in Sect. 3. The generalized herding attack is given in Sect. 4, and several variants are given in Sect. 5. We conclude this work in Sect. 6.

2. Preliminaries

By $\ln(x)$ we denote the natural logarithm of x with respect to base e, and by $\log(x)$ we denote the logarithm of x with respect to base 2.

2.1. Random Graphs and Hypergraphs

We briefly highlight basic definitions and results from the area of random graph theory. Although the analysis in this work centers around hypergraphs, we first discuss some results on basic graph theory. We refer readers interested in this area to [16].

2.1.1. Graphs

Let v be an integer. A (simple undirected) *graph* on v vertices is defined as $G(v) = (V, E)$. Here, V denotes a set of v vertices, and E consists of different subsets of V of size 2, the edges. A *matching* M is a subset of E with no common vertices. We call M a *perfect matching* if it covers every vertex $v \in V$. A perfect matching exists only if v is even.

Let $p \in [0, 1]$. We denote by $G(v, p)$ an Erdös-Rényi random graph [11] on v nodes with edge probability p. That is, each of the $\binom{v}{2}$ subsets of V is included in E with probability p. A central question in the area of graph

[1] With the minor difference that the attacker needs to employ an algorithm for expandable messages ([20], see Sect. 2.3) in order to set the length of the CTFP-preimage in advance.

theory is to evaluate the event that $G(v,p)$ has a perfect matching. It has been shown by Erdös and Rényi [12] that $G(v,p)$ contains a perfect matching (with high probability) if $p \geq \ln(v))/v$. Related to this question is the problem of, given a graph F on v_f vertices, determining a threshold for p for which the random graph $G(v,p)$ contains v/v_f disjoint copies of F (more formally, disjoint subgraphs of $G(v,p)$ isomorphic to F) [1, 22, 25, 27].

2.1.2. Hypergraphs

Hypergraphs differ from basic graphs essentially in how the edges are defined. A *hypergraph* on v vertices is defined as $H(v) = (V,E)$. Here, V denotes a set of v vertices, but E can contain different subsets of V of arbitrary size, called *hyperedges*. A hypergraph is called *d-uniform*, and we write $H_d(v)$, if each hyperedge is of size d. Notice that a 2-uniform hypergraph $H_2(v)$ covers the notion of a standard graph as defined before. For a hypergraph, a *perfect matching* is a set of pairwise disjoint edges whose union equals V. A perfect matching for $H_d(v)$ consists of v/d edges of size d vertices, and exists only if d divides v.

Similar to before, for $p \in [0,1]$ we denote by $H_d(v,p)$ an Erdös-Rényi random hypergraph on v nodes with edge probability p: each of the $\binom{v}{d}$ subsets of V of size d is included in E with probability p. Extending the theorem of Erdös and Rényi [12], Schmidt and Shamir [26] conjectured[2] that $H_d(v,p)$ contains a perfect matching with high probability if $p \geq \ln(v)/\binom{v-1}{d-1}$. Several attempts to prove this conjecture have been taken by Schmidt and Shamir [26], Frieze and Janson [13] and Kim [21]. The best result to date is by Johansson et al. [17], namely the following sharp threshold.

Proposition 1. *The threshold for a perfect matching in a d-uniform random hypergraph on v vertices is* $\Theta\left(\frac{\ln(v)}{v^{d-1}}\right)$ *[17, Cor. 2.6].*

In other words, $H_d(v,p)$ with high probability contains a perfect matching if $p \geq \ln(v)/v^{d-1}$. Notice that the basic result of Erdös and Rényi [12] for a graph $G(v,p) = H_2(v,p)$ directly follows from Prop. 1.

2.2. Merkle-Damgård Hash Function Design

We briefly discuss the classical Merkle-Damgård (MD) hash function construction. Let $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ be a compression function, and let $\mathsf{iv} \in \{0,1\}^n$ be an initial value. Let $\mathsf{pad} : \{0,1\}^* \to (\{0,1\}^m)^*$ be a padding function. The hash function $\mathsf{hash} : \{0,1\}^* \to \{0,1\}^n$ is defined as follows:

$$\begin{aligned} \mathsf{hash}(M) = h_z, \text{ where: } & (M_1,\ldots,M_z) = \mathsf{pad}(M), \\ & h_0 = \mathsf{iv}, \\ & h_i = f(h_{i-1}, M_i) \text{ for } i = 1,\ldots,z. \end{aligned}$$

[2]The claim by Schmidt and Shamir is somewhat different, the alternative interpretation is taken from [7].

As padding function, we consider the standard padding function with length strengthening. On input of the message M, it is first splitted into blocks of length m, the last block filled with 1 and sufficiently many 0s. Appended to this is a block $\langle(|M|+1)/m\rangle_m$ that encodes the length of the message in blocks. We note that our results also hold if a more compact encoding of the message is appended.

The complexity analysis in this work is done in terms of the amount of compression function executions. By a "(compression function) call", we denote the event of one compression function execution.

2.3. Expandable Messages

The generalized herding attack described in this work employs an algorithm introduced by Kelsey and Schneier [20], and we will briefly discuss it. On input of a state value h and a nonnegative integer k, this algorithm generates a set of 2^k messages of length varying from k to $k + 2^k - 1$ blocks, that on input of h render a 2^k-way collision. We call this set of messages a $(k, k+2^k-1)$-expandable message. The algorithm is derived from the multicollision-finding technique of Joux [18], and operates as follows. On input of state value h, one generates a multi-collision of length k in the following way: the first collision in the chain involves messages of length 2^0+1 and 1 blocks, the second collision messages of length 2^1+1 and 1, etc., until the last chain in which messages of length $2^{k-1}+1$ and 1 blocks collide. The construction of a $(k, k+2^k-1)$-expandable message requires approximately $k \cdot 2^{n/2+1} + 2^k$ calls.

3. Elongated Multidimensional Diamond

As a building block of our generalized herding attack, we revisit the elongated diamond structure of [19], and generalize it in the sense that each node in the diamond may have more than two ramifications. Formally, for three integers $k, \ell \geq 0$ and $d \geq 2$, we consider a diamond of size k levels and only consisting of d-way collisions, with a tail of length $2^\ell - 1$ state values attached to each end node. A visualization is given in Fig. 1. We now elaborate on the construction of this elongated multidimensional diamond, and its corresponding complexity.

3.1. Construction of d^k Tails

For $i = 1,\ldots,d^k$, the attacker takes an arbitrary state value $h_0^{(i)}$, and generates an arbitrary path of length $2^\ell - 1$. For each i, this phase results in the following compression function evaluations, for some $M_1^{(i)},\ldots,M_{2^\ell-1}^{(i)}$:

$$\begin{aligned} & f(h_0^{(i)}, M_1^{(i)}) = h_1^{(i)} \\ & \cdots \\ & f(h_{2^\ell-2}^{(i)}, M_{2^\ell-1}^{(i)}) = h_{2^\ell-1}^{(i)}. \end{aligned}$$

The construction of these d^k tails requires $d^k(2^\ell - 1)$ compression function executions.

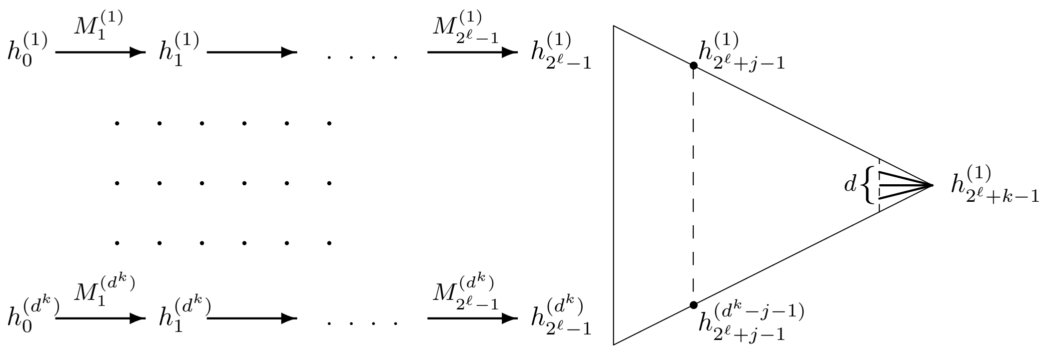

Figure 1: A (k, ℓ, d) elongated multidimensional diamond (cf. Sect. 3).

3.2. Construction of Multidimensional Diamond

Starting from the nodes $h_{2^\ell-1}^{(1)}, \ldots, h_{2^\ell-1}^{(d^k)}$, the attacker generates a d-dimensional diamond structure of k levels. The approach to build this diamond is fairly similar to the procedure described by [19]: firstly, the attacker finds d^{k-1} different d-way collisions starting from the d^k starting values, secondly he finds d^{k-2} different d-way collisions starting from the d^{k-1} intermediate state values, etc., until a single state value $h_{2^\ell+k-1}^{(1)}$ is left (see Fig. 1). The attacker finds these collisions by making L compression function executions, where the value L is determined later, from each state value for different message blocks M.

The analysis of the costs for producing a multidimensional diamond follows and generalizes the work of Blackburn et al. [6]. First, we determine the value L in order for the attacker to succeed with high probability, and then we compute the total complexity. Let $j \geq 1$ and consider the attacker mapping d^j values to d^{j-1} by ways of d-way collisions. For each of the d^j state values, the attacker makes L compression function calls. Notice that any d state values result in a d-way collision with probability $\approx L^d/(2^n)^{d-1}$. Consequently, we can model this problem as an Erdös-Rényi random hypergraph $H := H_d(d^j, L^d/(2^n)^{d-1})$, and the goal is to obtain a perfect matching in H. By Prop. 1, H contains a perfect matching with high probability if

$$\frac{L^d}{(2^n)^{d-1}} \approx \frac{\ln(d^j)}{(d^j)^{d-1}},$$

or equivalently if

$$L \approx \sqrt[d]{j \ln(d)} \cdot 2^{(n - j\log(d))\frac{d-1}{d}}.$$

In total, the attacker needs to make $d^j L \approx \sqrt[d]{j\ln(d)} \cdot 2^{n\frac{d-1}{d} + j\frac{\log(d)}{d}}$ compression function calls to reduce the set of state values from d^j to d^{j-1}. The total work to reduce from d^k to 1 values can then be computed as (cf. Appendix A)

$$\sum_{j=1}^{k} \sqrt[d]{j\ln(d)} \cdot 2^{n\frac{d-1}{d} + j\frac{\log(d)}{d}} = \Theta\left(\frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}}\right). \quad (1)$$

3.3. Total Complexity

The total amount of queries needed for constructing the elongated multidimensional diamond is about

$$d^k(2^\ell - 1) + \frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}}. \quad (2)$$

4. Generalized Herding Attack

We are ready to describe the generalized herding attack. A visualization of the attack is given in Fig. 2. Let $(k, \ell, d) \geq (0, 0, 2)$ be three integral parameters.

1. The attacker constructs an *elongated multidimensional diamond* for parameters (k, ℓ, d) (see Sect. 3). Denote its intermediate chaining values as in Fig. 1. This step requires $d^k(2^\ell - 1) + \frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}}$ compression function executions;
2. Starting from $h_{2^\ell+k-1}^{(1)}$, the attacker $\mathcal{A}$ generates a $(\log(k+2^\ell), \log(k+2^\ell)+k+2^\ell-1)$-*expandable message* (see Sect. 2.3). Denote the final state value by h_{exp}. This step requires approximately $\log(k+2^\ell) \cdot 2^{n/2+1} + k + 2^\ell$ compression function executions;
3. $\mathcal{A}$ computes a *commitment* $h = f(h_{\text{exp}}, M_{\text{pad}})$, where $M_{\text{pad}} = \langle \log(k+2^\ell)+k+2^\ell+1 \rangle_m$. He commits to this hash value h. This step requires 1 compression function execution;
4. $\mathcal{A}$ receives a *challenge* P, and computes $h_P = f(\mathsf{iv}, P)$. This step requires 1 compression function execution;
5. $\mathcal{A}$ finds a message M_{link} such that $f(h_P, M_{\text{link}}) = h_j^{(i)}$ for some $j \in \{0, \ldots, 2^\ell + k - 1\}$, $i \in \{1, \ldots, d^k\}$. He outputs a *CTFP-preimage* $P \| M_{\text{link}} \| M_{\text{diam}} \| M_{\text{exp}} \| M_{\text{pad}}$, where M_{diam} is the message of length $2^\ell + k - 1 - j$ blocks that connects $h_j^{(i)}$ with $h_{2^\ell+k-1}^{(1)}$, and M_{exp} is the message of length $\log(k+2^\ell) + j$ blocks coming from the expandable message phase. As there are

$$2^\ell d^k + d^{k-1} + d^{k-2} + \cdots + 1 = 2^\ell d^k + \frac{d^k - 1}{d - 1}$$

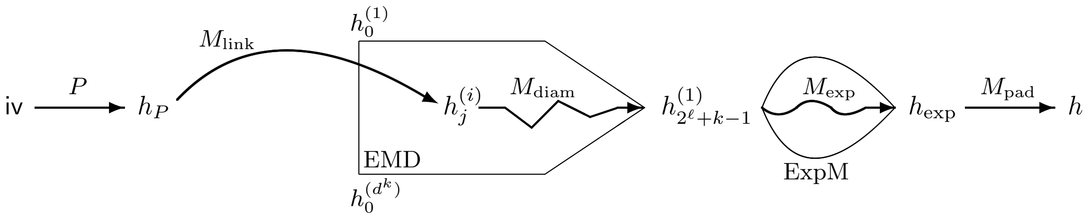

Figure 2: A sketch of the generalized herding attack of Sect. 4. The EMD block denotes a (k, ℓ, d) elongated multidimensional diamond (Fig. 1). The ExpM block denotes a $(\log(k + 2^\ell), \log(k + 2^\ell) + k + 2^\ell - 1)$-expandable message (Sect. 2.3).

state values $h_j^{(i)}$ for which a hit yields success of the attack (and with high probability these are almost all different), this phase requires approximately $2^n/(2^\ell d^k + \frac{d^k-1}{d-1})$ calls.

4.1. Total Complexity

The described generalized herding attack consists of an offline phase (part 1 and 2) and an online phase (part 3, 4 and 5). In its offline phase, the attack requires about

$$d^k(2^\ell - 1) + \frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}} + \log(k + 2^\ell) \cdot 2^{n/2+1} + k + 2^\ell + 1$$

calls. The online phase requires approximately

$$1 + \frac{2^n}{2^\ell d^k + \frac{d^k-1}{d-1}}$$

calls. The resulting preimage is of length $k + 2^\ell + \log(k + 2^\ell) + 2$ blocks, 1 block of which corresponds to the prefix.

5. Concrete Attack Variants

The general herding attack of Sect. 4 can be tweaked via three parameters $(k, \ell, d) \geq (0, 0, 2)$. The original herding attack of Kelsey and Kohno is covered for minimal values of ℓ, d, and for arbitrary k [3]. The attack requires about $\sqrt{k} \cdot 2^{(n+k)/2}$ offline and 2^{n-k} online computations. An equilibrium is achieved at $k = n/3$, for which the attack requires about $\sqrt{n} \cdot 2^{2n/3}$ calls and the preimage is of length about $n/3 + \log(n) + 2$ blocks.

The parameters ℓ, d can be used to influence the relation between the efficiency of the protocol and the length of the CTFP-preimage: increasing ℓ results in faster attacks with longer messages, while increasing d results in shorter messages but with less efficient attacks. We elaborate on these two scenarios.

[3]With the negligible difference that expandable messages are not used, and the attacker succeeds in phase 5 of the attack only if he hits $h_0(i)$ for some $i \in \{1, \dots, 2^k\}$.

5.1. Higher Efficiency with Longer Messages

We consider the affect of parameter ℓ on the efficiency of the attack. Therefore, we set $d = 2$. This attack corresponds to the elongated herding attack of [19]. It requires approximately $2^{k+\ell} - 2^k + \sqrt{k} \cdot 2^{(n+k)/2} + \log(k+2^\ell) \cdot 2^{n/2+1} + k + 2^\ell$ offline and $2^{n-k-\ell}$ online compression function calls. The resulting preimage is of length $k + 2^\ell + \log(k + 2^\ell) + 2$ blocks. To exemplify the strength of this attack, we only consider equilibria: we use ℓ to adjust the aimed length of the preimage, and k is set so as to make the dominating factors in both complexities approximately equal: $k = \frac{n-2\ell}{3}$.

For $n = 512$, the graph in Fig. 3 shows the total complexity and the length of the message as functions of ℓ. At the one extreme we have the classical herding attack for $\ell = 0$, $k = n/3$. At the other extreme, for $\ell = n/2$ and $k = 0$ the algorithm requires approximately $n \cdot 2^{n/2}$ work and the message is of length slightly more than $2^{n/2}$ blocks. Other variants are summarized in Table 1. These attacks meet the security bound derived by Andreeva and Mennink for the MD mode of operation [4].

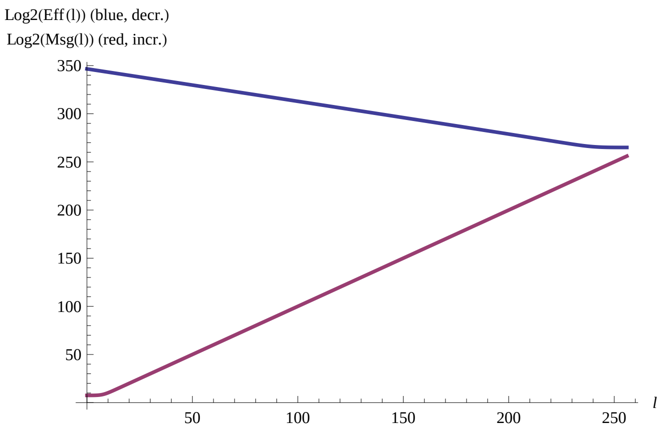

Figure 3: The efficiency and the size of the message for the elongated herding attack, as functions of ℓ. The functions are plotted in $\log_2$ scale. The blue (decreasing) line denotes the amount of work, and the red (increasing) line the length of the message in blocks. For various values of ℓ, the details are given in Table 1.

5.2. Shorter Messages with Lower Efficiency

In order to carry out the herding attack for messages considerably shorter than the one from [19], we consider the value d and set $\ell = 0$. We call this attack the multidimensional herding attack. This attack requires approximately $\frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d}+k\frac{\log(d)}{d}} + \log(k) \cdot 2^{n/2+1} + k$ offline and $2^{n-k\log(d)}$ online compression function calls, and results in a preimage of length $k + \log(k+1) + 3$ blocks. Again, we consider equilibria to analyze this attack: d is used to adjust the aimed length of the preimage, and k is set so as to make the dominating factors in both complexities equal: $k = \frac{n}{(d+1)\log(d)}$.

For $n = 512$, the graph in Fig. 4 shows the total complexity and the length of the message as functions of d. At the extreme we have the classical herding attack for $d = 2$, $k = n/3$. For increasing d, the complexity increases but the message length decreases. For instance, $d = 4$ results in an attack in about $2^{4n/5}$ work and with messages of length $n/14 + \log(n)$ blocks. For larger values of m, the attack approaches the generic attack. Other variants are summarized in Table 2.

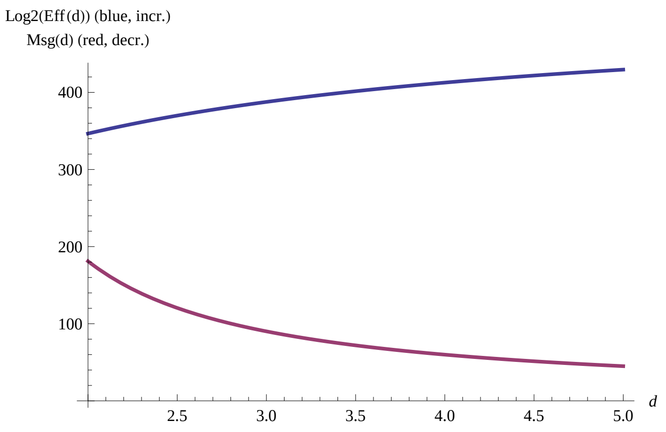

Figure 4: The efficiency and the size of the message for the multidimensional herding attack, as functions of d. The efficiency is plotted in $\log_2$ scale while the message size is plotted in linear scale. The blue (increasing) line denotes the amount of work, and the red (decreasing) line the length of the message in blocks. The case $d > 5$ is non-interesting due to the minor differences. For various values of d, the details are given in Table 2.

6. Conclusions

At EUROCRYPT 2006, Kelsey and Kohno described a herding attack for the MD iterated hash function design that requires approximately $\sqrt{n} \cdot 2^{2n/3}$ compression function calls, as well as an elongated herding attack that using a parameter ℓ allows for increasing efficiency at the cost of exponentially large messages. In this paper, we re-analyzed and generalized these findings to allow for a bidirectional tradeoff between the efficiency of the attack and the length of its preimages. Together with an additional parameters d, the attack allows for a full tradeoff between the efficiency and the length of the messages.

In the attack with shorter messages, the multidimensional herding attack, the diamond consists of d-way collisions only. Consequently, the construction of such diamond of k levels results in d^k state values that eventually render a collision. An approach to improve the flexibility may be to allow for more variation within the diamond: for instance, the first part of the diamond layers consists of d_1-way collisions, while the second part consists of d_2-way collisions. We leave it as a further research question to analyze whether this generalized diamond renders interesting complexity results.

Acknowledgments

This work has been funded in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II, and in part by the Research Council K.U.Leuven: GOA TENSE. The author is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

[1] Noga Alon and Raphael Yuster. Threshold functions for H-factors. *Probability and Computing*, 2:137–144, 1993.

[2] Elena Andreeva, Charles Bouillaguet, Orr Dunkelman, and John Kelsey. Herding, second preimage and trojan message attacks beyond Merkle-Damgård. In *Selected Areas in Cryptography 2009*, volume 5867 of *Lecture Notes in Computer Science*, pages 393–414, Berlin, 2009. Springer-Verlag.

[3] Elena Andreeva, Charles Bouillaguet, Pierre-Alain Fouque, Jonathan Hoch, John Kelsey, Adi Shamir, and Sébastien Zimmer. New second preimage attacks on hash functions. *Submitted to Journal of Cryptology*, 2010. `http://www.di.ens.fr/~bouillaguet/pub/JOC2010.pdf`.

[4] Elena Andreeva and Bart Mennink. Provable chosen-target-forced-midfix preimage resistance. In *Selected Areas in Cryptography 2011*, Lecture Notes in Computer Science, 2011. To appear.

[5] Elena Andreeva, Gregory Neven, Bart Preneel, and Thomas Shrimpton. Seven-property-preserving iterated hashing: ROX. In *Advances in Cryptology - ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 130–146, Berlin, 2007. Springer-Verlag.

[6] Simon Blackburn, Douglas Stinson, and Jalaj Upadhyay. On the complexity of the herding attack and some related attacks on hash functions. *Des. Codes Cryptography*, 2011. To appear.

[7] Colin Cooper, Alan Frieze, Michael Molloy, and Bruce Reed. Perfect matchings in random r-regular, s-uniform hypergraphs. *Combin. Probab. Comput*, 5:1–14, 1996.

[8] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård revisited: How to construct a hash function. In *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448, Berlin, 2005. Springer-Verlag.

[9] Ivan Damgård. A design principle for hash functions. In *Advances in Cryptology - CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427, Berlin, 1990. Springer-Verlag.

[10] Richard Dean. *Formal Aspects of Mobile Code Security.* PhD thesis, Princeton University, Princeton, 1999.

[11] Paul Erdös and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.*, 5:17–61, 1960.

[12] Paul Erdös and Alfréd Rényi. On the existence of a factor of degree one of a connected random graph. *Acta Mathematica Hungarica*, 17(3):359–368, 1966.

Table 1: The elongated herding attack (Sect. 5.1) and various variants. Here, ℓ is chosen in order to adjust the message length, and k is set so as to obtain an equal amount of work in both phases of the protocol. Negligible terms are removed for clarity.

	(ℓ, d)	k	Offline Work ($\approx$)	Online Work ($\approx$)	Message Length (blocks)
Elongated Herding [19]	$(\ell, 2)$	k	$2^{k+\ell} - 2^k + \sqrt{k} \cdot 2^{(n+k)/2} +$ $\log(k + 2^\ell) \cdot 2^{n/2+1} + k + 2^\ell$	$2^{n-k-\ell}$	$\log(k + 2^\ell) + k + 2^\ell + 2$
$\lvert\text{msg}\rvert \approx 2^{n/2}$	$(n/2, 2)$	0	$n \cdot 2^{n/2}$	$2^{n/2}$	$2^{n/2} + n/2 + 2$
$\lvert\text{msg}\rvert \approx 2^{n/4}$	$(n/4, 2)$	$n/6$	$\sqrt{n} \cdot 2^{7n/12}$	$2^{7n/12}$	$2^{n/4} + 5n/12 + 3$
$\lvert\text{msg}\rvert \approx 2^{n/16}$	$(n/16, 2)$	$7n/24$	$\sqrt{n} \cdot 2^{31n/48}$	$2^{31n/48}$	$2^{n/16} + 17n/48 + 3$
$\lvert\text{msg}\rvert \approx 2^{n/32}$	$(n/32, 2)$	$15n/48$	$\sqrt{n} \cdot 2^{63n/96}$	$2^{63n/96}$	$2^{n/32} + 33n/96 + 3$
Herding [19]	$(0, 2)$	$n/3$	$\sqrt{n} \cdot 2^{2n/3}$	$2^{2n/3}$	$n/3 + \log(n) + 2$

Table 2: The multidimensional herding attack (Sect. 5.2) and various variants. Here, d is chosen in order to adjust the message length, and k is set so as to obtain an equal amount of work in both phases of the protocol. Negligible terms are removed for clarity.

	(ℓ, d)	k	Offline Work ($\approx$)	Online Work ($\approx$)	Message Length (blocks)
Multidimensional Herding	$(0, d)$	k	$\frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}} +$ $\log(k) \cdot 2^{n/2+1} + k$	$2^{n-k\log(d)}$	$k + \log(k+1) + 3$
Herding ($d = 2$) [19]	$(0, 2)$	$n/3$	$\sqrt{n} \cdot 2^{2n/3}$	$2^{2n/3}$	$n/3 + \log(n) + 2$
$d = 3$	$(0, 3)$	$0.158n$	$\sqrt[3]{n} \cdot 2^{3n/4}$	$2^{3n/4}$	$0.126n + \log(n)$
$d = 4$	$(0, 4)$	$n/10$	$\sqrt[4]{n} \cdot 2^{4n/5}$	$2^{4n/5}$	$n/14 + \log(n)$
$d = 5$	$(0, 5)$	$0.072n$	$\sqrt[5]{n} \cdot 2^{5n/6}$	$2^{5n/6}$	$0.048n + \log(n)$

[13] Alan Frieze and Svante Janson. Perfect matchings in random s-uniform hypergraphs. *Random Struct. Algorithms*, 7:41–57, 1997.

[14] Praveen Gauravaram and John Kelsey. Linear-XOR and additive checksums don't protect Damgård-Merkle hashes from generic attacks. In *CT-RSA 2008*, volume 4964 of *Lecture Notes in Computer Science*, pages 36–51, Berlin, 2008. Springer-Verlag.

[15] Praveen Gauravaram, John Kelsey, Lars Knudsen, and Søren Thomsen. On hash functions using checksums. *International Journal of Information Security*, 9(2):137–151, 2010.

[16] Svante Janson, Tomasz Łuczak, and Andrzej Ruciński. *Random Graphs*. Wiley, 2000.

[17] Anders Johansson, Jeff Kahn, and van Vu. Factors in random graphs. *Random Struct. Algorithms*, 33(1):1–28, 2008. `http://onlinelibrary.wiley.com/doi/10.1002/rsa.20224/pdf`.

[18] Antoine Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 306–316, Berlin, 2004. Springer-Verlag.

[19] John Kelsey and Tadayoshi Kohno. Herding hash functions and the Nostradamus attack. In *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 183–200, Berlin, 2006. Springer-Verlag.

[20] John Kelsey and Bruce Schneier. Second preimages on n-bit hash functions for much less than 2^n work. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 474–490, Berlin, 2005. Springer-Verlag.

[21] Jeong Han Kim. Perfect matchings in random uniform hypergraphs. *Random Struct. Algorithms*, 23:111–132, 2003.

[22] Michael Krivelevich. Triangle factors in random graphs. *Combin. Probab. Comput*, 6:337–347, 1997.

[23] Ralph Merkle. One way hash functions and DES. In *Advances in Cryptology - CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446, Berlin, 1990. Springer-Verlag.

[24] Gregory Neven, Nigel Smart, and Bogdan Warinschi. Hash function requirements for Schnorr signatures. *Journal of Mathematical Cryptology*, 3(1):69–87, 2009.

[25] Andrzej Ruciński. Matching and covering the vertices of a random graph by copies of a given graph. *Discrete Mathematics*, 105(1-3):185–197, 1992.

[26] Jeanette Schmidt and Eli Shamir. A threshold for perfect matchings in random d-pure hypergraphs. *Discrete Mathematics*, 45(2-3):287–295, 1983.

[27] Yun-Zhi Yan, Han-Xing Wang, Jun Wang, and Xiang-Feng Yin. H(n)-factors in random graphs. *Statistics & Probability Letters*, 78(11):1255–1258, 2008.

Appendix A. Proof of Equation (1)

We will provide an upper and lower bound for

$$W := \sum_{j=1}^{k} \sqrt[d]{j \ln(d)} \cdot 2^{n\frac{d-1}{d} + j\frac{\log(d)}{d}}.$$

In order to bound this quantity, we first derive the following two results for integral $d \geq 2$:

$$\sqrt{\ln(2)} \leq \sqrt[d]{\ln(d)} \leq 2, \tag{A.1}$$

$$\frac{d}{\ln(d)} \leq \frac{d^{1/d}}{d^{1/d} - 1} \leq \frac{2d}{\ln(d)}. \tag{A.2}$$

Note that (A.1) indeed holds: the upper bound is clear, and for the lower bound we particularly have $\sqrt[d]{\ln(d)} \geq 1 \geq \sqrt{\ln(2)}$ for $d \geq 3$. Equation (A.2) directly follows if we show that

$$\ln(x) \leq x - 1 \leq \ln(x)/2 \tag{A.3}$$

holds for any $e^{-1/e} \leq x \leq 1$. This is true if we put $x = d^{-1/d}$ (which is indeed $\geq e^{-1/e}$ for all $d \geq 2$). As $\ln(x)$ is a concave function, it lies below any of its tangents. In other words, we obtain $\ln(a) \leq \ln(b) + (a - b)\left[\frac{d}{dy}\ln(y)\right]_b$ for any a, b. The lower bound of (A.3) is now obtained by putting $(a, b) = (x, 1)$. For the upper bound of (A.3), putting $(a, b) = (1, x)$ gives $\ln(x) \geq (x - 1)/x$, which is $\geq 2(x - 1)$ for $1/2 \leq e^{-1/e} \leq x \leq 1$. We next derive upper and lower bounds for W.

Appendix A.1. Upper Bound for W

$$
\begin{aligned}
W &\leq 2 \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d}} \cdot \sum_{j=1}^{k} d^{j/d}, \text{ by eqn. (A.1)} \\
&= 2 \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d}} \cdot \frac{d^{1/d}\left(d^{k/d} - 1\right)}{d^{1/d} - 1} \\
&= 2 \cdot \frac{d^{1/d}}{d^{1/d} - 1} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}} \\
&= O\left(\frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}}\right), \text{ by eqn. (A.2).}
\end{aligned}
$$

Appendix A.2. Lower Bound for W

$$
\begin{aligned}
W &\geq \sqrt{\ln(2)} \cdot \sqrt[d]{k/2} \cdot 2^{n\frac{d-1}{d}} \cdot \sum_{j=k/2}^{k} d^{j/d}, \text{ by eqn. (A.1)} \\
&= \sqrt{\ln(2)} \cdot \sqrt[d]{k/2} \cdot 2^{n\frac{d-1}{d}} \cdot \frac{d^{k/(2d)}\left(d^{(k+2)/(2d)} - 1\right)}{d^{1/d} - 1} \\
&\geq \frac{\sqrt{\ln(2)}}{2\sqrt[d]{2}} \cdot \frac{d^{1/d}}{d^{1/d} - 1} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}} \\
&= \Omega\left(\frac{d}{\ln(d)} \cdot \sqrt[d]{k} \cdot 2^{n\frac{d-1}{d} + k\frac{\log(d)}{d}}\right), \text{ by eqn. (A.2).}
\end{aligned}
$$