Information Processing Letters 112 (2012) 98–105

Contents lists available at SciVerse ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Increasing the flexibility of the herding attack

Bart Mennink *

Katholieke Universiteit Leuven, ESAT/COSIC and IBBT, Kasteelpark Arenberg 10, 3001 Leuven, Belgium

ARTICLE INFO

Article history:
Received 20 July 2011
Received in revised form 17 October 2011
Accepted 18 October 2011
Available online 20 October 2011
Communicated by D. Pointcheval

Keywords:
Cryptography
Hash functions
Chosen-target-forced-prefix
Generalized herding attack
Hypergraphs

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.

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

* Tel.: +32 16 321800; fax: +32 16 321969.
E-mail address: bmennink@esat.kuleuven.be.

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2011.10.014

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 $k \geqslant 0$ 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 message;
3. $\mathcal{A}$ receives a *challenge* P, and computes $h_P = f(\text{iv}, P)$;
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 $\ell \geqslant 0$ 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 $d \geqslant 2$. 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, Section 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 Section 2. The elongated multidimensional diamond is described and analyzed in Section 3. The generalized herding attack is given in Section 4, and several variants are given in Section 5. We conclude this work in Section 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].

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