Second Preimages for Iterated Hash Functions and Their Implications on MACs

Norbert Pramstaller, Mario Lamberger, and Vincent Rijmen

Institute for Applied Information Processing and Communications (IAIK)
Graz University of Technology, Austria
{Mario.Lamberger,Norbert.Pramstaller,Vincent.Rijmen}@iaik.tugraz.at

Abstract. In this article, we focus on second preimages for iterated hash functions. More precisely, we introduce the notion of a b-block bypass which is closely related to the notion of second preimage but specifies additional properties. We will then give two examples of iterated hash functions to which this notion applies: a double-block length hash function and a single-block length hash function. Furthermore, we look at NMAC and HMAC and show the implications of a b-block bypass regarding forgery attacks. As a result it turns out that the impact of second preimages for NMAC and HMAC heavily depends on how the second preimages are constructed.

Keywords: iterated hash functions, double block-length hash functions, block-cipher based hash functions, differential cryptanalysis, second preimage.

1 Introduction

A cryptographic hash function maps a binary string of arbitrary length to a fixed length binary string, called hash value. A cryptographic hash function H has to be secure against the following attacks:

- **Collision attack:** Find two different messages m and $m^* \neq m$ such that $H(m) = H(m^*)$
- **Preimage attack:** For a given hash value h, find a message m such that $H(m) = h$
- **Second preimage attack:** For a given message m, find a second message $m^* \neq m$ such that $H(m) = H(m^*)$

Based on the birthday paradox the expected complexity for a collision attack is about $2^{n/2}$ hash computations, where n is the size of the hash value. For a preimage attack and a second preimage attack the complexity is about 2^n hash computations. If, for a given hash function H, collisions and (second) preimages can be found with a complexity less than $2^{n/2}$ and 2^n, respectively, the hash function is considered to be broken.

Recently, a lot of progress has been made in the cryptanalysis of hash functions. Especially the breakthrough results of Wang *et al.* showing how to construct collisions for MD5 and SHA-1 [15,16], have drawn a lot of attention to

J. Pieprzyk, H. Ghodosi, and E. Dawson (Eds.): ACISP 2007, LNCS 4586, pp. 68–81, 2007.

the analysis of hash functions in the research community. To date, most of the attacks focus on collisions for iterated hash functions. Collisions are considered to be less devastating than second preimages since the adversary needs to control both messages. Kelsey and Schneier [7] have recently presented a new generic second preimage attack on iterated hash functions following the Merkle-Damgård construction (cf. [4,13]). As a result, second preimages can be found in much less than the theoretically expected 2^n hash computations for very long messages.

Besides the cryptanalysis of hash functions it is of high interest to understand the implications of these recent advances for applications employing hash functions. For instance, which implications does a collision attack on a hash function have for message authentication codes such as NMAC and HMAC? Recently, some answers to this question have been published in [3,10,14].

Being motivated by these new results, we will look at the implications of second preimages for NMAC and HMAC. We will start by introducing a new notion for iterated hash functions, namely a b-block bypass, in Section 2. This notion is closely related to the definition of a second preimage but specifies more details on how the second preimage can be constructed. To justify the newly introduced notion we discuss two hash functions for which a b-block bypass can be constructed. In Section 3, we analyze a double-block length hash function presented at FSE 2006 [6], referred to as DBLH. We will show that if this hash function scheme is instantiated with a block cipher following the FX construction [9] we can construct a 2-block, respectively 3-block, bypass. As another example, we will discuss the SMASH design strategy [11] in Section 4. We will show that the second preimage attack presented by Lamberger *et al.* in [12] satisfies the definition of a b-block bypass. In Section 5, we analyze NMAC and HMAC employing these hash functions. Finally, we present conclusions in Section 6.

2 The Notion of b-Block Bypass

In this section, we introduce a new property of iterated hash functions and show which implications it has. For the remainder of this article, we assume without loss of generality that we have message lengths that are a multiple of the block length. Furthermore, we assume that the blocks required for MD strengthening have been removed.

Definition 1. *(b-Block Bypass) Let H be an iterated hash function. We say that we can construct a* b-block bypass *for H, if for any b-block message $m = m_1, \ldots, m_b$ we can find a b-block message $m^* = m_1^*, \ldots, m_b^* \neq m$ such that for any initial value h_0 the following holds:*

$$\begin{aligned} &H(h_0; m_1, \ldots, m_i) \neq H(h_0; m_1^*, \ldots, m_i^*) \quad \text{for } i = 1, \ldots, b-1 \\ &H(h_0; m_1, \ldots, m_b) = H(h_0; m_1^*, \ldots, m_b^*) \end{aligned} \tag{1}$$

Remark 1. It follows directly from Definition 1 that the notions of b-block bypass and second preimage are closely related. To be more precise, if we can construct a b-block bypass for an iterated hash function then it is possible to

construct a second preimage m^* for any given message $m = m_1, \ldots, m_t \neq m^*$ with $t \geq b$. Furthermore, both the second preimage m^* and the message m are of equal length. Hence, a b-block bypass provides additional details such as the dependency on the chaining value.

Lemma 1. *Let H be an iterated hash function for which we can construct a b-block bypass. Then, for every message $m = m_1, \ldots, m_t$ with $t \geq b \geq 1$, we can construct at least*

$$\sum_{j=1}^{\lfloor t/b \rfloor} \binom{t - j(b-1)}{j} \tag{2}$$

distinct second preimages.

Proof. From Definition 1 it follows immediately that it doesn't matter which b consecutive blocks of the message m are taken to construct a second preimage m^* (cf. Figure 1).

If $\lfloor t/b \rfloor \geq 2$, we can apply Definition 1 not only for one b-block sub-message of m but for j sub-messages, with j ranging from $1, \ldots, \lfloor t/b \rfloor$. An illustration of this fact is also shown in Figure 1. The problem of counting all these possible second preimages of m boils down to counting the number of possibilities of putting $t - jb$ indistinguishable balls into $j + 1$ distinguishable urns. This number is known to be

$$\binom{t - j(b-1)}{j},$$

cf. [5, page 38, Eq. (5.2)]. Summing over all $j = 1, \ldots, \lfloor t/b \rfloor$ proves (2). □

Remark 2. The result of Lemma 1 seems intuitive. However, Lemma 1 does not necessarily apply to the notion of second preimage but it always holds for the notion of b-block bypass. Therefore, the notion of b-block bypass enables a better insight on the possibilities for constructing a second preimage.

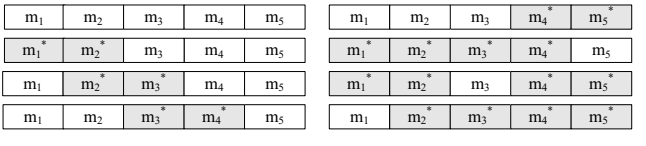

Fig. 1. For a 2-block bypass we can construct for any 5-block message $m = m_1, \ldots, m_5$ seven distinct second preimages. The shadowed rectangles show which blocks of the original message m have been modified to construct the second preimage.

3 The Double Block-Length Hash Proposal DBLH

We start this section by introducing some notation. For the concatenation of two variables, we write $a \| b$. Addition modulo 2 (XOR) is denoted by $a \oplus b$. The bit

length of variable a is denoted by $|a|$. We stick to the convention of [2] to denote a difference by $u' = u \oplus u^*$. Furthermore, we write $F_k(x)$ for the encryption of the input x with an arbitrary block cipher F under the key k. The cipher F processes blocks of n bits and the key length is denoted by $|k|$.

Shoichi Hirose proposed a double block-length hash function at FSE 2006 [6]. It is an iterated, block cipher based hash function. The compression function is defined as follows:

$$\begin{aligned} g_i &= F_{h_{i-1}\|m_i}(g_{i-1}) \oplus g_{i-1} \\ h_i &= F_{h_{i-1}\|m_i}(g_{i-1} \oplus c) \oplus g_{i-1} \oplus c, \end{aligned} \tag{3}$$

where c is an arbitrary constant ($c \neq 0$), F_k ($k = h_{i-1}\|m_i$) is an arbitrary block cipher, and $h_i\|g_i$ is the chaining value with $h_0\|g_0$ the initial value (cf. Figure 2). After t message blocks have been processed, the final hash value is the concatenation $h_t\|g_t$. As it can be seen in (3), the key length of the underlying block cipher F_k has to be greater than the block length. This is due to the fact that $|k| = |h_{i-1}| + |m_i|$, where $|h_{i-1}|$ is the block length of the cipher. In [6], Hirose proved the security of DBLH in the ideal cipher model.

3.1 Block Ciphers Following the FX Construction

The block cipher DESX [9] was proposed by Rivest to protect DES against exhaustive key search attacks. Kilian and Rogaway proved the security of the DESX construction in [8,9] against a key-search adversary. However, DESX is not an *ideal cipher*. The general form of this construction is referred to as FX [8,9], where F can be any block cipher with block length n and key length $|k|$. The FX construction is defined as follows:

$$\mathrm{FX}_{k\|k_1\|k_2}(x) = F_k(x \oplus k_1) \oplus k_2 \ , \tag{4}$$

where $|k_1| = |k_2| = n$.

3.2 DBLH with FX

For DBLH with underlying block cipher $\mathrm{FX}_{k\|k_1\|k_2}(x)$, we can construct the following three configurations (see Figure 2), where $m_i = l_i\|r_i$.

$$\begin{aligned} &\text{Configuration I:} \\ &k\|k_1\|k_2 = l_i\|h_{i-1}\|r_i, \text{ where } |l_i| = |k|, |h_{i-1}| = |r_i| = n \\ &\text{Configuration II:} \\ &k\|k_1\|k_2 = h_{i-1}\|l_i\|r_i, \text{ where } |h_{i-1}| = |k|, |l_i| = |r_i| = n \\ &\text{Configuration III:} \\ &k\|k_1\|k_2 = l_i\|r_i\|h_{i-1}, \text{ where } |l_i| = |k|, |r_i| = |h_{i-1}| = n \end{aligned} \tag{5}$$

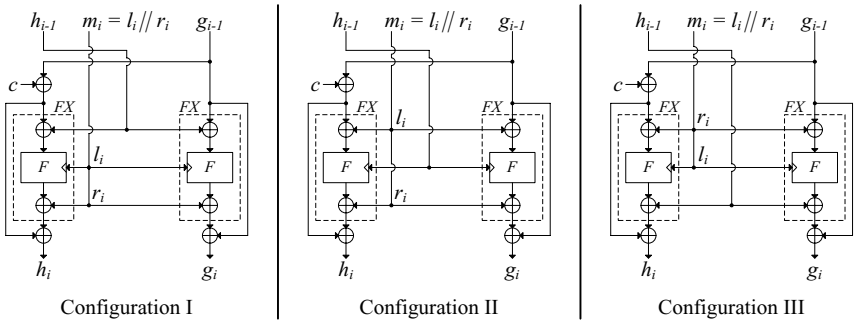

Fig. 2. Three possible configurations of DBLH with FX as underlying block cipher. The hatch denotes the key input of the block cipher F.

For each configuration, we can interchange l_i and r_i. However, without loss of generality, we take the configurations defined in (5) for the further analysis. Note that if F is a block cipher with $|k| < n$ then, for Configuration II, the chaining variable h_{i-1} needs to be truncated to match the key length $|k|$. Which bits are truncated does not have any impact on the analysis. For the remainder of this section, we assume that F is a block cipher with $|k| = n$.

For the sake of simplicity, we will write DX to denote the instantiation of DBLH with FX as underlying block cipher. If we speak of a specific configuration, we append the number of the configuration. For instance for DBLH with FX in Configuration II, we write DX-II.

3.3 Second Preimages for DX Based on a *b*-Block Bypass

We now demonstrate how to construct second preimages based on a 3-block bypass for Configuration II of DX.

Theorem 1. *For the iterated hash function DX-II we can construct a* 3*-block bypass, since for every* 3*-block message* $m = m_1, m_2, m_3$ *the following message* m^* *satisfies the conditions of Definition 1:*

$$m^* = m_1 \oplus (0\|u'), m_2 \oplus (v'\|w'), m_3 \oplus (z'\|z') \ , \tag{6}$$

where $m_i = l_i\|r_i$, $|l_i| = |r_i| = n$, u', v' *any value with* $|u'| = |v'| = n$, *and* 0 *is the* n*-bit all-zero binary string. Let* t' *be the output difference of the left* F *instance in iteration* 2*:*

$$t' = [F_{h_1}(g_1 \oplus c \oplus l_2)] \oplus [F_{h_1 \oplus u'}(g_1 \oplus u' \oplus c \oplus l_2 \oplus v')] \tag{7}$$

Then, $w' = u' \oplus t'$ *and the difference* z' *in* (6) *is defined as*

$$z' = [F_{h_1}(g_1 \oplus l_2) \oplus r_2 \oplus g_1] \oplus \\ [F_{h_1 \oplus u'}(g_1 \oplus u' \oplus l_2 \oplus v') \oplus r_2 \oplus w' \oplus g_1 \oplus u'] \,. \quad (8)$$

Furthermore, for an arbitrary message $m = m_1, \ldots, m_t$ with $t \geq 3$, we can find at least

$$\sum_{j=1}^{\lfloor t/3 \rfloor} \binom{t-2j}{j}$$

second preimages based on this 3-block bypass.

Proof. We show that for the 3-block messages m and m^*, where

$$\begin{aligned} m &= m_1, m_2, m_3 = (l_1 \| r_1), (l_2 \| r_2), (l_3 \| r_3) \\ m^* &= m_1 \oplus (0 \| u'), m_2 \oplus (v' \| w'), m_3 \oplus (z' \| z') = (l_1^* \| r_1^*), (l_2^* \| r_2^*), (l_3^* \| r_3^*) \\ l_1^* &= l_1 \oplus 0, \quad r_1^* = r_1 \oplus u' \\ l_2^* &= l_2 \oplus v', \quad r_2^* = r_2 \oplus w' \\ l_3^* &= l_3 \oplus z', \quad r_3^* = r_3 \oplus z' \,, \end{aligned}$$

the output difference equals zero after three iterations. After one iteration, we have

$$\begin{aligned} g_1 &= g_0 \oplus F_{h_0}(g_0 \oplus l_1) \oplus r_1 \\ g_1^* &= g_0 \oplus F_{h_0}(g_0 \oplus l_1) \oplus r_1 \oplus u' = g_1 \oplus u' \\ h_1 &= g_0 \oplus c \oplus F_{h_0}(g_0 \oplus c \oplus l_1) \oplus r_1 \\ h_1^* &= g_0 \oplus c \oplus F_{h_0}(g_0 \oplus c \oplus l_1) \oplus r_1 \oplus u' = h_1 \oplus u' \,. \end{aligned}$$

After two iterations, chaining variable h_2 is computed as follows

$$\begin{aligned} h_2 &= g_1 \oplus c \oplus F_{h_1}(g_1 \oplus c \oplus l_2) \oplus r_2 \\ h_2^* &= g_1 \oplus u' \oplus c \oplus F_{h_1 \oplus u'}(g_1 \oplus u' \oplus c \oplus l_2 \oplus v') \oplus r_2 \oplus w' \,. \end{aligned}$$

With $w' = u' \oplus t'$ and t' as defined in (7), we get

$$\begin{aligned} h_2^* &= g_1 \oplus u' \oplus c \oplus F_{h_1 \oplus u'}(g_1 \oplus u' \oplus c \oplus l_2 \oplus v') \oplus r_2 \oplus u' \\ &\quad \oplus \underbrace{F_{h_1}(g_1 \oplus c \oplus l_2) \oplus F_{h_1 \oplus u'}(g_1 \oplus u' \oplus c \oplus l_2 \oplus v')}_{t'} \\ &= g_1 \oplus u' \oplus c \oplus r_2 \oplus u' \oplus F_{h_1}(g_1 \oplus c \oplus l_2) \\ &= h_2 \,. \end{aligned}$$

The difference in chaining variable g_2 after two iterations is

$$g_2^* = g_2 \oplus z' \,,$$

where z' is defined in (8). After three iterations, we get

$$
\begin{aligned}
g_3 &= g_2 \oplus F_{h_2}(g_2 \oplus l_3) \oplus r_3 \\
g_3^* &= g_2 \oplus z' \oplus F_{h_2}(g_2 \oplus z' \oplus l_3 \oplus z') \oplus r_3 \oplus z' \\
&= g_2 \oplus F_{h_2}(g_2 \oplus l_3) \oplus r_3 \\
&= g_3 \\
h_3 &= g_2 \oplus c \oplus F_{h_2}(g_2 \oplus c \oplus l_3) \oplus r_3 \\
h_3^* &= g_2 \oplus z' \oplus c \oplus F_{h_2}(g_2 \oplus z' \oplus c \oplus l_3 \oplus z') \oplus r_3 \oplus z' \\
&= g_2 \oplus c \oplus F_{h_2}(g_2 \oplus c \oplus l_3) \oplus r_3 \\
&= h_3 \ .
\end{aligned}
$$

Therefore, after three iterations the differences in the chaining variables are $g'_3 = g_3 \oplus g_3^* = 0$ and $h'_3 = h_3 \oplus h_3^* = 0$. Since the difference of the chaining variables $g'_0 = h'_0 = 0$, we have constructed a 3-block bypass for DX-II.

The final statement of the theorem is an immediate consequence of Lemma 1 with $b = 3$. □

For Configuration I and III, we can prove similar theorems.

Theorem 2. *For the iterated hash function DX-I, we can construct a* 2*-block bypass, since for every two block message* $m = m_1, m_2$ *the following message* m^* *satisfies the conditions of Definition 1:*

$$m^* = m_1 \oplus (0\|u'), m_2 \oplus (0\|u') \ , \tag{9}$$

where $m_i = l_i\|r_i$, $|l_i| = |k|$, $|r_i| = n$, u' *any value with* $|u'| = n$, *and* 0 *is the* $|k|$*-bit all-zero binary string.*
Furthermore, for an arbitrary message $m = m_1, \ldots, m_t$ *with* $t \geq 2$, *we can find at least*

$$\sum_{j=1}^{\lfloor t/2 \rfloor} \binom{t-j}{j}$$

second preimages based on this 2*-block bypass.*

Theorem 3. *For the iterated hash function DX-III, we can construct a* 3*-block bypass, since for every* 3*-block message* $m = m_1, m_2, m_3$ *the following message* m^* *satisfies the conditions of Definition 1:*

$$m^* = m_1 \oplus (u'\|v'), m_2 \oplus (0\|z'), m_3 \oplus (0\|(w' \oplus z')) \ , \tag{10}$$

where $m_i = l_i\|r_i$, $|l_i| = |k|$, $|r_i| = n$, u', v' *any value with* $|u'| = |k|$ *and* $|v'| = n$, *and* 0 *is the* $|k|$*-bit all-zero binary string. Once the values* u', v' *have been chosen for the given input message block* m_1, *the differences* w' *and* z' *can be computed:*

$$
\begin{aligned}
w' &= [g_0 \oplus c \oplus F_{l_1}(g_0 \oplus c \oplus r_1) \oplus h_0] \\
&\quad \oplus [g_0 \oplus c \oplus F_{l_1 \oplus v'}(g_0 \oplus c \oplus r_1 \oplus u') \oplus h_0] \ , \\
z' &= [g_0 \oplus F_{l_1}(g_0 \oplus r_1) \oplus h_0] \\
&\quad \oplus [g_0 \oplus F_{l_1 \oplus v'}(g_0 \oplus r_1 \oplus u') \oplus h_0]
\end{aligned}
$$

Furthermore, for an arbitrary message $m = m_1, \ldots, m_t$ *with* $t \geq 3$*, we can find at least*

$$\sum_{j=1}^{\lfloor t/3 \rfloor} \binom{t-2j}{j}$$

second preimages based on this 3*-block bypass.*

The proof of Theorem 2 and Theorem 3 works along the same lines as the proof of Theorem 1 and is given in Appendix A and B.

4 The Hash Function Design Strategy SMASH

In [11], Knudsen presented a new design strategy for iterated hash functions. For a message $m = m_1, m_2, \ldots, m_t$ consisting of t blocks of length n, the hash output h_{t+1} gets computed via

$$h_0 = f(iv) + iv \tag{11}$$

$$h_i = f(h_{i-1} + m_i) + h_{i-1} + \theta m_i \quad \text{for } i = 1, \ldots, t \tag{12}$$

$$h_{t+1} = f(h_t) + h_t, \tag{13}$$

where f denotes a bijective, non-linear n-bit mapping. Note that "+" and multiplication by θ is defined as an operation in the finite field $GF(2^n)$ with the only restriction that $\theta \notin \{0, 1\}$. In [11], also two instantiations of SMASH have been proposed, namely SMASH-256 and SMASH-512 which produce a 256-bit, respectively 512-bit output.

Let us for now consider a slightly reduced variant of SMASH-n by omitting the final step (13) in the definition of SMASH-n. The main result of [12] is a method to effectively construct preimages for this reduced variant. Their method makes use of the following simple observation (which was already pointed out in [11]): Let h_i and h_i^* be two intermediate hash values and let m_i be an arbitrary n-bit message block. Then, if we set $m_i^* = m_i + h_{i-1} + h_{i-1}^*$ we have

$$h_i + h_i^* = (1 + \theta)(h_{i-1} + h_{i-1}^*).$$

This can be used to derive an equation of the form:

$$h_t = a + b \sum_{j=1}^{t} \delta_j (1 + \theta)^{t-j}, \tag{14}$$

where $1 \leq t \leq n$, a, b are values depending on the used initial value and compression function f, and the $\delta_i \in \{0, 1\}$ are unknowns on which the respective blocks of the preimage $m^* = m_1^*, \ldots, m_n^*$ will depend. Equation (14) can be interpreted as an inhomogenous system of n linear equations in t variables over $GF(2)$.

For the solvability of this system we have to look on the element θ. If $(1 + \theta)$ is not contained in a proper subfield of $GF(2^n)$, then the elements $(1 + \theta)^i$ are

linearly independent for $0 \leq i \leq n-1$. For SMASH-256 and SMASH-512, it is easy to show that this condition is satisfied. In most applications we will have $n = 2^\ell$. Then, a randomly selected θ fulfills this requirement with probability $1 - 2^{-n/2}$.

Thus, if we set $t = n$ in equation (14) we are guaranteed a unique solution $\delta_1, \ldots, \delta_n$ from which an n-block preimage m^* can be constructed. For a more detailed description of the method we refer to [12].

Remark 3. To clarify the multiple use of n we recapitulate: SMASH-n operates on n-bit blocks but since we need exactly n variables to derive a unique solution for the system (14) the method of [12] also produces a preimage consisting of n blocks.

Because of (13) this preimage attack cannot be augmented to the full variant of SMASH-n. However, we can use this result to construct an n-block bypass for an arbitrary message $m = m_1, m_2, \ldots, m_n$. Let h_n denote the chaining value computed after n applications of (12) starting from our initial message m. Then, the technique described above leads to a message m^* such that $h_n^* = h_n$ and therefore $h_{n+1}^* = h_{n+1}$. The method shown in [12] guarantees that the constructed second preimage m^* differs from m at least in the first message block. Since this can be carried out independent of the choice of h_0 we arrive at the following theorem:

Theorem 4. *For almost all instantiations of SMASH-n, we can construct an n-block bypass. Especially, we can construct a* 256*-block, respectively* 512*-block bypass for the hash functions SMASH-256, respectively SMASH-512.*

5 Implications of a *b*-Block Bypass for NMAC and HMAC

In this section, we will look at the implications if one of the hash functions described in Section 3 and Section 4 is employed in applications such as message authentication codes. In particular, we will focus on NMAC and HMAC [1]:

$$\mathrm{NMAC}_{k_1,k_2}(m) = H(k_1, H(k_2, m)) \tag{15}$$

$$\mathrm{HMAC}_k(m) = H(H(iv, k \oplus opad), H(H(iv, k \oplus ipad), m)) \ , \tag{16}$$

where $H(cv, m)$ denotes the application of the iterated hash function H with chaining value cv (iv or k_i) and t-block message $m = m_1, \ldots, m_t$. For HMAC, two appropriate padding methods *ipad* and *opad* for the secret key k are required (see [1] for further details). Both constructions are depicted in Figure 3.

For NMAC the initial value and for HMAC the chaining value of the iterated hash function H processing the message m are not known to an adversary unless he/she knows the secret key k. Therefore, a second preimage attack on an iterated

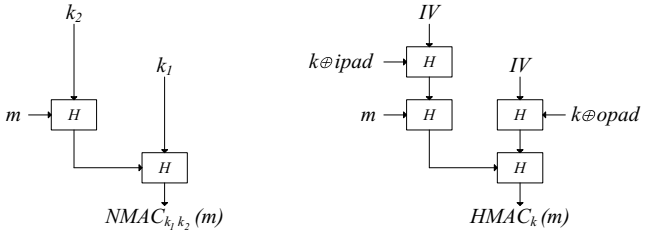

Fig. 3. The NMAC (left) and HMAC (right) construction based on an iterative hash function H

hash function for which the attacker needs to know certain chaining values will not lead to an immediate forgery. On the other side, if the second preimage attack is independent of the initial chaining value, an adversary will always succeed in forging an authenticated message: for any given valid message-MAC pair $\{m, MAC_k(m)\}$ he/she can construct a second valid message-MAC pair by just replacing m with the second preimage m^*. If we look at the hash functions described in Section 3 and Section 4, we can now conclude the following:

Fact 1. *The second preimages based on the* 3*-block bypass for DX-II and DX-III, as well as the n-block bypass for SMASH-n cannot directly be exploited to mount a forgery attack on NMAC and HMAC. This is an immediate consequence of the fact that certain chaining values need to be known by the adversary for constructing the second preimage.*

Fact 2. *For the DX-I construction we see that the second preimage based on the* 2*-block bypass can be constructed in a pure differential way, i.e. it is independent of the chaining values. Therefore, both NMAC and HMAC with DX-I as underlying hash function are vulnerable to forgery attacks.*

From these facts we observe that even if we can construct second preimages for both hash functions in all configurations, the implications for the security of hash-based MACs depend heavily on how the second preimage is constructed.

6 Conclusion

In this article, we have introduced the notion of b-block bypass for iterated hash functions, which is closely related to the notion of second preimage. A b-block bypass is more accurate in the sense that the structure of second preimages based on a b-block bypass is more clear. We presented two entirely different hash functions for which we can construct a b-block bypass. Even if we can construct second preimages deterministically for both hash functions, we have shown that if we look at NMAC/HMAC the implications are different. It turned out that

for NMAC/HMAC it is important how the second preimage is constructed: the DX construction in Configuration I implies immediate forgery, whereby DX in Configuration II and III as well as the SMASH construction do not lead to a forgery attack on NMAC/HMAC. We can derive from our results that a weak hash function does not necessarily imply a weak application employing this hash function. Therefore, it makes sense to not only define properties for the hash function but to specify additional properties concerning the application in which a hash function is employed.

Acknowledgements

The authors wish to thank Florian Mendel, Christian Rechberger, and the anonymous referees for useful comments and discussions.

The work in this paper has been supported in part by the Austrian Science Fund (FWF), project P18138 and by the European Commission through the IST Programme under contract IST2002507 932 ECRYPT. The information in this paper is provided as is, and no guarantee or warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

References

1. Bellare, M., Canetti, R., Krawczyk, H.: Keying Hash Functions for Message Authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1–15. Springer, Heidelberg (1996)
2. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology 4(1), 3–72 (1991)
3. Contini, S., Yin, Y.L.: Forgery and Partial Key-Recovery Attacks on HMAC and NMAC Using Hash Collisions. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 37–53. Springer, Heidelberg (2006)
4. Damgård, I.: A Design Principle for Hash Functions. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer, Heidelberg (1990)
5. Feller, W.: An introduction to probability theory and its application, 3rd edn. vol. I. John Wiley & Sons, New York (1968)
6. Hirose, S.: Some Plausible Constructions of Double-Block-Length Hash Functions. In: Robshaw, M. (ed.) FSE 2006. LNCS, vol. 4047, pp. 210–225. Springer, Heidelberg (2006)
7. Kelsey, J., Schneier, B.: Second Preimages on n-bit Hash Functions for Much less than 2^n Work. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 474–490. Springer, Heidelberg (2005)
8. Kilian, J., Rogaway, P.: How to Protect DES Against Exhaustive Key Search. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 252–267. Springer, Heidelberg (1996)

9. Kilian, J., Rogaway, P.: How to Protect DES Against Exhaustive Key Search (an Analysis of DESX). J. Cryptology 14(1), 17–35 (2001)
10. Kim, J., Biryukov, A., Preneel, B., Hong, S.: On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0 and SHA-1 (Extended Abstract). In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 242–256. Springer, Heidelberg (2006)
11. Knudsen, L.R.: SMASH - A Cryptographic Hash Function. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 228–242. Springer, Heidelberg (2005)
12. Lamberger, M., Pramstaller, N., Rechberger, C., Rijmen, V.: Second Preimages for SMASH. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 101–111. Springer, Heidelberg (2006)
13. Merkle, R.C.: One Way Hash Functions and DES. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg (1989)
14. Rechberger, C., Rijmen, V.: On Authentication with HMAC and Non-Random Properties. In: Financial Cryptography and Data Security, 11th International Conference, FC, Lowlands, Scarborough, Trinidad/Tobago, February 12–15, 2007 (to appear in LNCS)
15. Wang, X., Yao, A., Yao, F.: New Collision Search for SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, Springer, Heidelberg (2005)
16. Wang, X., Yu, H.: How to Break MD5 and Other Hash Functions. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)

A Proof of Theorem 2

Proof. Assume, we have the following 2-block messages m, m^*, where:

$$\begin{aligned} m &= m_1, m_2 = (l_1 \| r_1), (l_2 \| r_2) \\ m^* &= m_1^*, m_2^* = m_1 \oplus (0 \| u'), m_2 \oplus (0 \| u') = (l_1^* \| r_1^*), (l_2^* \| r_2^*) \\ l_1^* &= l_1 \oplus 0 = l_1, \quad r_1^* = r_1 \oplus u' \\ l_2^* &= l_2 \oplus 0 = l_2, \quad r_2^* = r_2 \oplus u' \end{aligned}$$

After one iteration, we have

$$\begin{aligned} g_1 &= g_0 \oplus F_{l_1}(g_0 \oplus h_0) \oplus r_1 \\ g_1^* &= g_0 \oplus F_{l_1}(g_0 \oplus h_0) \oplus r_1 \oplus u' = g_1 \oplus u' \text{ , and} \\ h_1 &= g_0 \oplus c \oplus F_{l_1}(g_0 \oplus c \oplus h_0) \oplus r_1 \\ h_1^* &= g_0 \oplus c \oplus F_{l_1}(g_0 \oplus c \oplus h_0) \oplus r_1 \oplus u' = h_1 \oplus u' \text{ .} \end{aligned}$$

The outputs after two iterations are

$$\begin{aligned} g_2 &= g_1 \oplus F_{l_2}(g_1 \oplus h_1) \oplus r_2 \\ g_2^* &= g_1 \oplus u' \oplus F_{l_2}(g_1 \oplus u' \oplus h_1 \oplus u') \oplus r_2 \oplus u' \\ &= g_1 \oplus F_{l_2}(g_1 \oplus h_1) \oplus r_2 = g_2 \text{ , and} \end{aligned}$$

$$
\begin{aligned}
h_2 &= g_1 \oplus c \oplus F_{l_2}(g_1 \oplus c \oplus h_1) \oplus r_2 \\
h_2^* &= g_1 \oplus u' \oplus c \oplus F_{l_2}(g_1 \oplus u' \oplus c \oplus h_1 \oplus u') \oplus r_2 \oplus u' \\
&= g_1 \oplus c \oplus F_{l_2}(g_1 \oplus c \oplus h_1) \oplus r_2 = h_2 \ .
\end{aligned}
$$

Hence, $g_2' = g_2 \oplus g_2^* = 0$ and $h_2' = h_2 \oplus h_2^* = 0$. Since the difference of the chaining variables $g_0' = h_0' = 0$, we have constructed a 2-block bypass for DX-I. The final statement of the theorem is an immediate consequence of Lemma 1 with $b = 2$. □

B Proof of Theorem 3

As for the proof of Theorem 1 and Theorem 2, we show that for the 3-block messages m and m^*, where $m = m_1, m_2, m_3 = (l_1 \| r_1), (l_2 \| r_2), (l_3 \| r_3)$ and

$$
\begin{aligned}
m^* = m_1 \oplus (u' \| v'), m_2 \oplus (0 \| z'), m_3 \oplus (0 \| (w' \oplus z')) &= (l_1^* \| r_1^*), (l_2^* \| r_2^*), (l_3^* \| r_3^*) \\
l_1^* = l_1 \oplus u', \quad r_1^* &= r_1 \oplus v' \\
l_2^* = l_2 \oplus 0, \quad r_2^* &= r_2 \oplus z' \\
l_3^* = l_3 \oplus 0, \quad r_3^* &= r_3 \oplus (w' \oplus z') \ ,
\end{aligned}
$$

the output difference equals zero after three iterations, *i.e.* $g_3' = h_3' = 0$. After the first iteration, we have

$$
\begin{aligned}
g_1 &= g_0 \oplus F_{l_1}(g_0 \oplus r_1) \oplus h_0 \\
g_1^* &= g_1 \oplus z', \text{where} \\
z' &= [g_0 \oplus F_{l_1}(g_0 \oplus r_1) \oplus h_0] \\
&\quad \oplus [g_0 \oplus F_{l_1 \oplus v'}(g_0 \oplus r_1 \oplus u') \oplus h_0] \ , \text{and} \\
h_1 &= g_0 \oplus c \oplus F_{l_1}(g_0 \oplus c \oplus r_1) \oplus h_0 \\
h_1^* &= h_1 \oplus w', \text{where} \\
w' &= [g_0 \oplus c \oplus F_{l_1}(g_0 \oplus c \oplus r_1) \oplus h_0] \\
&\quad \oplus [g_0 \oplus c \oplus F_{l_1 \oplus v'}(g_0 \oplus c \oplus r_1 \oplus u') \oplus h_0] \ .
\end{aligned}
$$

The difference of the chaining variables after two iterations is

$$
\begin{aligned}
g_2 &= g_1 \oplus F_{l_2}(g_1 \oplus r_2) \oplus h_1 \\
g_2^* &= g_1 \oplus z' \oplus F_{l_2}(g_1 \oplus z' \oplus r_2 \oplus z') \oplus h_1 \oplus w' \\
&= g_2 \oplus (w' \oplus z') \ , \text{and} \\
h_2 &= g_1 \oplus c \oplus F_{l_2}(g_1 \oplus c \oplus r_2) \oplus h_1 \\
h_2^* &= g_1 \oplus z' \oplus c \oplus F_{l_2}(g_1 \oplus z' \oplus c \oplus r_2 \oplus z') \oplus h_1 \oplus w' \\
&= h_2 \oplus (w' \oplus z') \ .
\end{aligned}
$$

The output difference after three iterations is computed as follows. For the sake of clearness, we write $y' = w' \oplus z'$:

$$
\begin{aligned}
g_3 &= g_2 \oplus F_{l_3}(g_2 \oplus r_3) \oplus h_2 \\
g_3^* &= g_2 \oplus y' \oplus F_{l_3}(g_3 \oplus y' \oplus r_3 \oplus y') \oplus h_2 \oplus y' \\
&= g_3 \text{ , and} \\
h_3 &= g_2 \oplus c \oplus F_{l_3}(g_2 \oplus c \oplus r_3) \oplus h_2 \\
h_3^* &= g_2 \oplus y' \oplus c \oplus F_{l_3}(g_2 \oplus y' \oplus c \oplus r_3 \oplus y') \oplus h_2 \oplus y' \\
&= h_3
\end{aligned}
$$

Hence, $g'_3 = g_3 \oplus g_3^* = 0$ and $h'_3 = h_3 \oplus h_3^* = 0$. Since the difference of the chaining variables $g'_0 = h'_0 = 0$, we have constructed a 3-block bypass for DX-III.
The final statement of the theorem is an immediate consequence of Lemma 1 with $b = 3$. □