



# New Proofs for NMAC and HMAC: Security without Collision Resistance

Mihir Bellare

Department of Computer Science & Engineering 0404,  
University of California San Diego,  
9500 Gilman Drive, La Jolla, CA 92093-0404, USA  
mihir.bellare@gmail.com; <http://www-cse.ucsd.edu/users/mihir>

Communicated by Damgaard

Received 24 June 2006

Online publication 10 June 2014

**Abstract.** HMAC was proved in Bellare et al. (Advances in Cryptology–CRYPTO’96, Springer, Berlin, Heidelberg, 1996) to be a PRF assuming that (1) the underlying compression function is a PRF, and (2) the iterated hash function is weakly collision resistant. However, subsequent attacks showed that assumption (2) is false for MD5 and SHA-1, removing the proof-based support for HMAC in these cases. This paper proves that HMAC is a PRF under the sole assumption that the compression function is a PRF. This recovers a proof-based guarantee since no known attacks compromise the pseudorandomness of the compression function, and it also helps explain the resistance to attack that HMAC has shown even when implemented with hash functions whose (weak) collision resistance is compromised. We also show that an even weaker than PRF condition on the compression function, namely that it is a privacy-preserving MAC, suffices to establish HMAC is a secure MAC as long as the hash function meets the very weak requirement of being computationally almost universal, where again the value lies in the fact that known attacks do not invalidate the assumptions made.

**Keywords.** Message authentication, Hash functions, Pseudorandom functions, Carter–Wegman.

## 1. Introduction

HMAC [4] is a popular cryptographic-hash-function-based MAC. The basic construct is actually NMAC, of which HMAC can be viewed as a derivative.

*The Constructions Succinctly:*

$$\begin{aligned} \text{NMAC}(K_{\text{out}} \| K_{\text{in}}, M) &= H^*(K_{\text{out}}, H^*(K_{\text{in}}, M)) \\ \text{HMAC}(K_{\text{out}} \| K_{\text{in}}, M) &= H(K_{\text{out}} \| H(K_{\text{in}} \| M)). \end{aligned}$$

Here  $H$  is a cryptographic hash function, eg. MD5 [34], SHA-1 [31], or RIPEMD-160 [19]. Let  $h: \{0, 1\}^c \times \{0, 1\}^b \rightarrow \{0, 1\}^c$  denote the underlying compression function. (Here  $b = 512$  while  $c$  is 128 or 160.) Let  $h^*$  be the iterated compression function which

on input  $K \in \{0, 1\}^c$  and a message  $x = x[1] \dots x[n]$  consisting of  $b$ -bit blocks, lets  $a[0] = K$  and  $a[i] = h(a[i-1], x[i])$  for  $i = 1, \dots, n$ , and finally returns  $a[n]$ . Then  $H^*(K, M) = h^*(K, M^*)$  and  $H(M) = H^*(IV, M)$ , where  $M^*$  denotes  $M$  padded appropriately to a length that is a positive multiple of  $b$  and  $IV$  is a public  $c$ -bit initial vector that is fixed as part of the description of  $H$ . Both NMAC and HMAC use two keys, which in the case of NMAC are of length  $c$  bits each, and in the case of HMAC of length  $b$  bits each and derived from a single  $b$ -bit key. HMAC is a non-intrusive version of NMAC in the sense that it uses the cryptographic hash function only as a blackbox, making it easier to implement.

*Usage* HMAC is standardized via an IETF RFC [27], a NIST FIPS [30], and ANSI X9.71 [1], and implemented in SSL, SSH, IPsec, and TLS among other places. It is often used as a PRF (pseudorandom function [21]) rather than merely as a MAC. In particular this is the case when it is used for key-derivation, as in TLS [18] and IKE (the Internet Key Exchange protocol of IPsec) [23]. HMAC is also used as a PRF in a standard for one-time passwords [29] that is the basis for Google authenticator.

*What's Known* The results are for NMAC, but can be lifted to HMAC. It is shown in [4] that NMAC is a secure PRF if (1) the underlying compression function  $h$  is a secure PRF, and also (2) that the hash function  $H$  is *weakly collision resistant* (WCR). The latter, introduced in [4], is a relaxation of collision resistance (CR) that asks that it be computationally infeasible for an adversary, given an oracle for  $H^*(K, \cdot)$  under a hidden key  $K$ , to find a collision, meaning distinct inputs  $M_1, M_2$  such that  $H^*(K, M_1) = H^*(K, M_2)$ .

*The Problem* HMAC is usually implemented with MD5 or SHA-1. But, due to the attacks of [37, 38], these functions are *not* WCR. Thus the assumption on which the proof of [4] is based is not true. This does not reflect any actual weaknesses in the NMAC or HMAC constructs, on which no attacks are known. (Being iterated MACs, the generic birthday-based forgery attacks of [33] always break NMAC and HMAC in time  $2^{c/2}$ , but we mean no better-than-birthday attacks are known.) But it means that we have lost the proof-based guarantees. We are interested in recovering them.

*Loss of WCR* First we pause to expand on the claim above that our main hash functions are not WCR. Although WCR appears to be a weaker requirement than CR due to the hidden key, for iterated hash functions, it ends up not usually being so. The reason is that collision-finding attacks such as those on MD5 [38] and SHA-1 [37] extend to find collisions in  $H^*(IV, \cdot)$  for an arbitrary but given  $IV$ , and, any such attack, via a further extension attack, can be used to compromise WCR, meaning to find a collision in  $H^*(K, \cdot)$ , given an oracle for this function, even with  $K$  hidden. This was pointed out in [4, 25], and, for the curious, we recall the attack in Sect. 6.

*Main Result* We show (Theorem 3.3) that NMAC is a PRF under the *sole* assumption that the underlying compression function  $h$  is itself a PRF. In other words, the additional assumption that the hash function is WCR is dropped. (And, in particular, as long as  $h$  is a PRF, the conclusion is true even if  $H$  is *not* WCR, let alone CR.)

The main advantage of our result is that it is based on an assumption that is not refuted by any known attacks. (There are to date no attacks that compromise the pseudorandomness of the compression functions of MD5 or SHA-1.) Another feature of our result is that it is the first proof for NMAC that is based solely on an assumption about the compression function rather than also assuming something about the entire iterated hash function.

*Techniques* We show (Lemma 3.1) that if a compression function  $h$  is a PRF then the iterated compression function  $h^*$  is *computationally almost universal* (cAU), a computational relaxation of the standard information-theoretic notion of almost universality (AU) of [14, 36, 39]. We then conclude with Lemma 3.2 which says that the composition of a PRF and a cAU function is a PRF. (This can be viewed as a computational relaxation of the Carter–Wegman paradigm [14, 39].)

*Another Result* The fact that compression functions are underlain by block ciphers, together with the fact that no known attacks compromise the pseudorandomness of the compression functions of MD5, SHA-1, may give us some confidence that it is ok to assume these are PRFs, but it still behooves us to be cautious. What can we prove about NMAC without assuming the compression function is a PRF? We would not expect to be able to prove it is a PRF, but what about just a secure MAC? (Any PRF is a secure MAC [7, 10], so our main result implies NMAC is a secure MAC, but we are interested in seeing whether this can be proved under weaker assumptions.) We show (Theorem 4.3) that NMAC is a secure MAC if  $h$  is a *privacy-preserving MAC* (PP-MAC) [9] and  $h^*$  (equivalently,  $H^*$ ) is cAU. A PP-MAC (the definition is provided in Sect. 4) is stronger than a MAC but weaker than a PRF. This result reverts to the paradigm of [4] of making assumptions both about the compression function and its iteration, but the point is that cAU is a very weak assumption compared to WCR and PP-MAC is a weaker assumption than PRF.

*From NMAC to HMAC* The formal results (both previous and new) we have discussed so far pertain to NMAC. However, discussions (above and in the literature) tend to identify NMAC and HMAC security-wise. This is explained by an observation of [4] which says that HMAC inherits the security of NMAC as long as the compression function is a PRF when keyed via the data input. (So far when we have talked of it being a PRF, it is keyed via the chaining variable.) In our case this means that HMAC is a PRF if the compression function is a “dual-PRF,” meaning a PRF when keyed by either of its two inputs.

However, the analysis above assumes that the two keys  $K_{\text{out}}$ ,  $K_{\text{in}}$  of HMAC are chosen *independently* at random, while in truth they are equal to  $K \oplus \text{opad}$  and  $K \oplus \text{ipad}$ , respectively, where  $K$  is a random  $b$ -bit key and  $\text{opad}$ ,  $\text{ipad}$  are fixed, distinct constants. We apply the theory of PRFs under related-key attacks [8] to extend the observation of [4] to this single-key version of HMAC, showing it inherits the security of NMAC as long as the data input-keyed compression function is a PRF under an appropriate (and small) class of related-key attacks. Assuming additionally that the compression function is a PRF in the usual sense, we obtain a (in fact, the first) security proof of the single-key version of HMAC. These results are in Sect. 5.

*Related Work* If  $h^*$  were a PRF, it would imply it is cAU, but  $h^*$  is not a PRF due to the extension attack. It is, however, shown by [5] that if  $h$  is a PRF then  $h^*$  (which they call the cascade) is a “pf-PRF” (prefix-free PRF), meaning a PRF as long as no query of the adversary is a prefix of another query. It was pointed out to us by Victor Shoup after seeing an early draft of our paper that it is possible to apply this in a blackbox way to show that  $h^*$  is cAU. Lemma 3.1, however, obtains a better bound. Although this is by only a constant factor in the context of the lemma itself, Theorem 3.4 shows that it can translate into an appreciably better security guarantee in the context of NMAC itself. The proof of Lemma 3.1 exploits and extends the ideas of [5] while also adding some

new ones. For comparison, we do present the indirect proof in Sect. 7 and contrast the bounds obtained with ours.

Dodis et al. [20, Lemma 4] show that the cascade over a family of *random* functions is AU as long as the two messages whose collision probability one considers have the same length. (In this model,  $h(K, \cdot)$  is a random function for each  $K \in \{0, 1\}^c$ . That is, it is like Shannon's ideal cipher model, except the component maps are functions not permutations.) This does not imply Lemma 3.1 (showing the cascade  $h^*$  is cAU if  $h$  is a PRF), because we need to allow the two messages to have different lengths, and also because it is not clear what implication their result has for the case when  $h$  is a PRF (A PRF does *not* permit one to securely instantiate a *family* of random functions.). A second result [20, Lemma 5] in the same paper says that if  $h^*(K, M)$  is close to uniformly distributed then so is  $h^*(K, M \| X)$ . (Here  $M$  is chosen from some distribution,  $K$  is a random but known key, and  $X$  is a fixed block.) This result only assumes  $h$  is a PRF, but again we are not able to discern any implications for the problems we consider, because in our case, the last block of the input is not fixed, we are interested in the cAU property rather than randomness, and our inputs are not drawn from a distribution.

*Versions and Subsequent Work* A preliminary version of this paper appears in the proceedings of CRYPTO'06 [3]. The full version you are now reading contains a different and simpler proofs of Claim 3.8, an additional result in the form of Theorem 3.4, and proofs and other material omitted from the preliminary version due to lack of space.

Proofs in our prior version gave blackbox reductions, but results had only stated the tighter, non-blackbox reductions derived via coin-fixing. Results now explicitly state both.

Tightness of our bounds is estimated based on conjectures about the prf security of the starting compression function  $h$ . Koblitz and Menezes [26] point out that the conjectures made in prior versions of our work were over optimistic In the context of non-blackbox reductions. (Time-memory tradeoff attacks [2, 17, 22, 24] should be factored in.) Tightness estimates are now based on the blackbox versions of our reductions and indicate that our bounds are not as tight as we had thought. The gap has been filled by Pietrzak [32], who gives blackbox reduction proofs for NMAC that he shows via matching attack to be tight. In the process, he shows that an improvement to our bounds claimed by [26] was wrong.

## 2. Definitions

*Notation* We denote the concatenation of strings  $s_1, s_2$  by either  $s_1 \| s_2$  or just  $s_1 s_2$ . We denote by  $|s|$  the length of string  $s$ . Let  $b$  be a positive integer representing a block length, and let  $B = \{0, 1\}^b$ . Let  $B^+$  denote the set of all strings of length a positive multiple of  $b$  bits. Whenever we speak of blocks we mean  $b$ -bit ones. If  $M \in B^+$  then  $\|M\|_b = |M|/b$  is the number of blocks in  $M$ , and  $M[i]$  denotes its  $i$ -th  $b$ -bit block, meaning  $M = M[1] \dots M[n]$  where  $n = \|M\|_b$ . If  $M_1, M_2 \in B^+$ , then  $M_1$  is a prefix of  $M_2$ , written  $M_1 \subseteq M_2$ , if  $M_2 = M_1 \| A$  for some  $A \in B^*$ . If  $S$  is a finite set then  $s \stackrel{\$}{\leftarrow} S$  denotes the operation of selecting  $s$  uniformly at random from  $S$ . An adversary is a (possibly randomized) algorithm that may have access to one or more oracles. We let

$$A^{\mathcal{O}_1, \dots}(x_1, \dots) \Rightarrow 1 \quad \text{and} \quad y \stackrel{\$}{\leftarrow} A^{\mathcal{O}_1, \dots}(x_1, \dots)$$

denote, respectively, the event that  $A$  with the indicated oracles and inputs outputs 1, and the experiment of running  $A$  with the indicated oracles and inputs and letting  $y$  be the value returned. (This value is a random variable depending on the random choices made by  $A$  and its oracles.) Either the oracles or the inputs (or both) may be absent, and often will be.

A family of functions is a two-argument map  $f: Keys \times Dom \rightarrow Rng$  whose first argument is regarded as a key. We fix one such family  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  to model a compression function that we regard as being keyed via its  $c$ -bit chaining variable. Typical values are  $b = 512$  and  $c = 128$  or  $160$ . The *iteration* of family  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  is the family of functions  $h^*: \{0, 1\}^c \times B^+ \rightarrow \{0, 1\}^c$  defined via:

Function  $h^*(K, M) \quad // \quad K \in \{0, 1\}^c, M \in B^+$   
 $n \leftarrow \|M\|_b; a[0] \leftarrow K$   
 For  $i = 1, \dots, n$  do  $a[i] \leftarrow h(a[i-1], M[i])$   
 Return  $a[n]$

This represents the Merkle-Damgård [15,28] iteration method used in all the popular hash functions but without the “strengthening,” meaning that there is no  $|M|$ -based message padding.

*PRFs* A prf-adversary  $A$  against a family of functions  $f: Keys \times Dom \rightarrow Rng$  takes as oracle a function  $g: Dom \rightarrow Rng$  and returns a bit. The prf-advantage of  $A$  against  $f$  is the difference between the probability that it outputs 1 when its oracle is  $g = f(K, \cdot)$  for a random key  $K \stackrel{\$}{\leftarrow} Keys$ , and the probability that it outputs 1 when its oracle  $g$  is chosen at random from the set  $\text{Maps}(Dom, Rng)$  of all functions mapping  $Dom$  to  $Rng$ , succinctly written as

$$\text{Adv}_f^{\text{prf}}(A) = \Pr \left[ A^{f(K, \cdot)} \Rightarrow 1 \right] - \Pr \left[ A^{\$} \Rightarrow 1 \right]. \quad (1)$$

In both cases, the probability is over the choice of oracle and the coins of  $A$ .

*cAU and Collision Probability* Let  $F: \{0, 1\}^k \times Dom \rightarrow Rng$  be a family of functions. cAU is measured by considering an almost-universal (au) adversary  $A$  against  $F$ . It (takes no inputs and) returns a pair of messages in  $Dom$ . Its au-advantage is

$$\text{Adv}_F^{\text{au}}(A) = \Pr \left[ F(K, M_1) = F(K, M_2) \wedge M_1 \neq M_2 : (M_1, M_2) \stackrel{\$}{\leftarrow} A; K \stackrel{\$}{\leftarrow} Keys \right].$$

This represents a very weak form of collision resistance since  $A$  must produce  $M_1, M_2$  without being given any information about  $K$ . WCR [4] is a stronger notion because here  $A$  gets an oracle for  $F(K, \cdot)$  and can query this in its search for  $M_1, M_2$ .

For  $M_1, M_2 \in Dom$  it is useful to let  $\text{Coll}_F(M_1, M_2) = \Pr[F(K, M_1) = F(K, M_2)]$ , the probability being over  $K \stackrel{\$}{\leftarrow} \{0, 1\}^k$ .

*MACs* Any PRF is a MAC. (This was established for the basic notion of MAC security in [7], but holds even for the most stringent notions and with tight security reductions

[10].) Accordingly, we do not explicitly discuss MACs until Sect. 4.1 where we consider PP-MACs.

### 3. Security of NMAC

Let  $h: \{0, 1\}^c \times \{0, 1\}^b \rightarrow \{0, 1\}^c$  be a family of functions that represents the compression function, here assumed to be a PRF. Let  $\text{pad}$  denote a padding function such that  $s^* = s \parallel \text{pad}(|s|) \in B^+$  for any string  $s$ . (Such padding functions are part of the description of current hash functions. Note the  $\text{pad}$  depends only on the length of  $s$ .) Then, the family **NMAC**:  $\{0, 1\}^{2c} \times D \rightarrow \{0, 1\}^c$  is defined by  $\text{NMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = h(K_{\text{out}}, h^*(K_{\text{in}}, M^*) \parallel \text{fpad})$  where  $\text{fpad} = \text{pad}(c) \in \{0, 1\}^{b-c}$  and  $h^*$  is the iterated compression function as defined in Sect. 2. The domain  $D$  is the set of all strings up to some maximum length, which is  $2^{64}$  for current hash functions.

It turns out that our security proof for NMAC does not rely on any properties of  $\text{pad}$  beyond the fact that  $M^* = M \parallel \text{pad}(|M|) \in B^+$ . (In particular, the Merkle-Damgård strengthening, namely inclusion of the message length in the padding, that is used in current hash functions and is crucial to collision resistance of the hash function, is not important to the security of NMAC.) Accordingly, we will actually prove the security of a more general construct that we call generalized NMAC. The family **GNMAC**:  $\{0, 1\}^{2c} \times B^+ \rightarrow \{0, 1\}^c$  is defined by  $\text{GNMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = h(K_{\text{out}}, h^*(K_{\text{in}}, M) \parallel \text{fpad})$  where  $\text{fpad}$  is any (fixed)  $b - c$  bit string. Note the domain is  $B^+$ , meaning inputs have to have a length that is a positive multiple of  $b$  bits. NMAC is nonetheless a special case of GNMAC via  $\text{NMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = \text{GNMAC}(K_{\text{out}} \parallel K_{\text{in}}, M^*)$  and thus the security of NMAC is implied by that of GNMAC. (Security as a PRF or a MAC, respectively, for both.)

#### 3.1. The Results

*Main Lemma* The following says that if  $h$  is a PRF then its iteration  $h^*$  is cAU.

**Lemma 3.1.** *Let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  be a family of functions and let  $n \geq 1$  be an integer. Let  $A^*$  be an au-adversary against  $h^*$  that has time complexity at most  $t$ . Assume the two messages output by  $A^*$  are at most  $n_1, n_2$  blocks long, respectively, where  $1 \leq n_1, n_2 \leq n$ . Then there exists a prf-adversary  $A$  against  $h$  such that*

$$\text{Adv}_{h^*}^{\text{au}}(A^*) \leq (n_1 + n_2 - 1) \cdot \text{Adv}_h^{\text{prf}}(A) + \frac{1}{2^c}. \tag{2}$$

*A makes at most 2 oracle queries. Its time complexity is  $t$  under a blackbox reduction and  $O(nT_h)$  under a non-blackbox reduction, where  $T_h$  is the time for one evaluation of  $h$ .*

By a blackbox reduction we mean that the proof gives an explicit procedure to construct  $A$  from  $A^*$ . In the non-blackbox case, the time complexity of  $A$  is lower, and in fact independent of that of  $A^*$ , meaning the reduction is quantitatively better, but in this case the proof establishes the existence of  $A$  without giving an explicit way to derive it from

$A^*$ . (In the latter case,  $A$  in fact depends only on  $h$  and  $n$ , not on  $A^*$ .) The proof of Lemma 3.1 is in Sect. 3.2.

One might ask whether stronger results hold. For example, assuming  $h$  is a PRF, (1) Is  $h^*$  a PRF? (2) Is  $h^*$  WCR? (Either would imply that  $h^*$  is cAU.) But the answer is NO to both questions. The function  $h^*$  is *never* a PRF due to the extension attack. On the other hand, it is easy to give an example of a PRF  $h$  such that  $h^*$  is not WCR. Also MD5 and SHA-1 are candidate counter-examples, since their compression functions appear to be PRFs, but their iterations are not WCR.

*The PRF(cAU) = PRF Lemma* The composition of families  $h: \{0, 1\}^c \times \{0, 1\}^b \rightarrow \{0, 1\}^c$  and  $F: \{0, 1\}^k \times D \rightarrow \{0, 1\}^b$  is the family  $hF: \{0, 1\}^{c+k} \times D \rightarrow \{0, 1\}^c$  defined by  $hF(K_{\text{out}} \| K_{\text{in}}, M) = h(K_{\text{out}}, F(K_{\text{in}}, M))$ . The following lemma says that if  $h$  is a PRF and  $F$  is cAU then  $hF$  is a PRF.

**Lemma 3.2.** *Let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  and  $F: \{0, 1\}^k \times D \rightarrow B$  be families of functions, and let  $hF: \{0, 1\}^{c+k} \times D \rightarrow \{0, 1\}^c$  be defined by*

$$hF(K_{\text{out}} \| K_{\text{in}}, M) = h(K_{\text{out}}, F(K_{\text{in}}, M))$$

for all  $K_{\text{out}} \in \{0, 1\}^c$ ,  $K_{\text{in}} \in \{0, 1\}^k$  and  $M \in D$ . Let  $A_{hF}$  be a prf-adversary against  $hF$  that makes at most  $q \geq 2$  oracle queries, each of length at most  $n$ , and has time complexity at most  $t$ . Then there exists a prf-adversary  $A_h$  against  $h$  and an au-adversary  $A_F$  against  $F$  such that

$$\text{Adv}_{hF}^{\text{prf}}(A_{hF}) \leq \text{Adv}_h^{\text{prf}}(A_h) + \binom{q}{2} \cdot \text{Adv}_F^{\text{au}}(A_F) \tag{3}$$

$A_h$  makes at most  $q$  oracle queries, has time complexity at most  $t$  and is obtained via a blackbox reduction. The two messages  $A_F$  outputs have length at most  $n$ . The time complexity of  $A_F$  is  $t$  under a blackbox reduction and  $O(T_F(n))$  under a non-blackbox reduction, where  $T_F(n)$  is the time to compute  $F$  on an  $n$ -bit input.

This extends the analogous PRF(AU)PRFlemma by relaxing the condition on  $F$  from AU to cAU. (The PRF(AU)PRFlemma is alluded to in [5, 12], and variants are in [12, 13].) A simple proof of Lemma 3.2, using games [11, 35], is in Sect. 3.3.

*GNMAC is a PRF* We now combine the two lemmas above to conclude that if  $h$  is a PRF then so is GNMAC.

**Theorem 3.3.** *Assume  $b \geq c$  and let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  be a family of functions and let  $\text{fpad} \in \{0, 1\}^{b-c}$  be a fixed padding string. Let GNMAC:  $\{0, 1\}^{2c} \times B^+ \rightarrow \{0, 1\}^c$  be defined by*

$$\text{GNMAC}(K_{\text{out}} \| K_{\text{in}}, M) = h(K_{\text{out}}, h^*(K_{\text{in}}, M) \| \text{fpad})$$

for all  $K_{\text{out}}, K_{\text{in}} \in \{0, 1\}^c$  and  $M \in B^+$ . Let  $A_{\text{GNMAC}}$  be a prf-adversary against GNMAC that makes at most  $q \geq 2$  oracle queries, each of at most  $m$  blocks, and has



time complexity at most  $t$ . Then there exist prf-adversaries  $A_1, A_2$  against  $h$  such that

$$\mathbf{Adv}_{\text{GNMAC}}^{\text{prf}}(A_{\text{GNMAC}}) \leq \mathbf{Adv}_h^{\text{prf}}(A_1) + \binom{q}{2} \left[ 2m \cdot \mathbf{Adv}_h^{\text{prf}}(A_2) + \frac{1}{2^c} \right]. \quad (4)$$

$A_1$  makes at most  $q$  oracle queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.  $A_2$  makes at most 2 oracle queries. The time complexity of  $A_2$  is  $t$  under a blackbox reduction and  $O(mT_h)$  under a non-blackbox reduction, where  $T_h$  is the time for one computation of  $h$ .

*Proof of Theorem 3.3.* Define  $F: \{0, 1\}^c \times B^+ \rightarrow \{0, 1\}^b$  by  $F(K_{\text{in}}, M) = h^*(K_{\text{in}}, M) \parallel \text{fpad}$ . Then  $\text{GNMAC} = hF$ . Apply Lemma 3.2 with  $k = c$ ,  $D = B^+$  and  $A_{hF} = A_{\text{GNMAC}}$  to get prf-adversary  $A_1$  and au-adversary  $A_F$  with the properties stated in the lemma. Note that  $\mathbf{Adv}_F^{\text{au}}(A_F) = \mathbf{Adv}_{h^*}^{\text{au}}(A_F)$ . This is because a pair of messages is a collision for  $h^*(K_{\text{in}}, \cdot) \parallel \text{fpad}$  iff it is a collision for  $h^*(K_{\text{in}}, \cdot)$ . Let  $n = m$  and apply Lemma 3.1 to  $A^* = A_F$  to get prf-adversary  $A_2$ .

The upper bound on the advantage of  $A_{\text{GNMAC}}$  in (4) is a function of a single assumed bound  $m$  on the block lengths of all messages queried by  $A_{\text{GNMAC}}$ . The following more general result expresses the bound in terms of assumed bounds on the block lengths of each query taken individually:

**Theorem 3.4.** Assume  $b \geq c$  and let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  be a family of functions and let  $\text{fpad} \in \{0, 1\}^{b-c}$  be a fixed padding string. Let  $\text{GNMAC}: \{0, 1\}^{2c} \times B^+ \rightarrow \{0, 1\}^c$  be defined as in Theorem 3.3. Let  $A_{\text{GNMAC}}$  be a prf-adversary against  $\text{GNMAC}$  that makes at most  $q \geq 2$  oracle queries, where the  $i$ -th query is of at most  $m_i \geq 1$  blocks for  $1 \leq i \leq q$ . Let  $n = m_1 + \dots + m_q$  and  $m = \max(m_1, \dots, m_q)$ . Assume  $A_{\text{GNMAC}}$  has time complexity at most  $t$ . Then there exist prf-adversaries  $A_1, A_2$  against  $h$  such that

$$\mathbf{Adv}_{\text{GNMAC}}^{\text{prf}}(A_{\text{GNMAC}}) \leq \mathbf{Adv}_h^{\text{prf}}(A_1) + (q-1) \cdot (n - q/2) \cdot \mathbf{Adv}_h^{\text{prf}}(A_2) + \binom{q}{2} \frac{1}{2^c}. \quad (5)$$

$A_1$  makes at most  $q$  oracle queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.  $A_2$  makes at most 2 oracle queries, has time complexity  $O(mT_h)$  and is obtained via a non-blackbox reduction, where  $T_h$  is the time for one computation of  $h$ .

Theorem 3.4 is more general than Theorem 3.3 in the sense that the former implies the (non-blackbox case of the) latter. Indeed, if we set  $m_i = m$  for all  $1 \leq i \leq q$  then we get  $n = mq$  and (5) implies (4). On the other hand, Theorem 3.4 can give significantly better security guarantees than Theorem 3.3 in cases where the messages being authenticated are of varying lengths. For example suppose  $A_{\text{GNMAC}}$  makes a single query of block length  $m$  followed by  $q-1$  queries, each a single block. Then the bound from (4) multiplies  $\mathbf{Adv}_h^{\text{prf}}(A_2)$  by  $q(q-1)m$ , whereas the bound from (5) multiplies



$\text{Adv}_h^{\text{prf}}(A_2)$  by  $(q - 1)(m + q/2 - 1)$ . These are thus different by a  $\Theta(q)$  factor if, say,  $m = \Theta(q)$ . The proof of Theorem 3.4 can be found in Sect. 3.4.

### 3.2. Proof of Lemma 3.1

*Some Definitions* In this proof it will be convenient to consider prf-adversaries that take inputs. The advantage of  $A$  against  $h$  on inputs  $x_1, \dots$  is defined as

$$\text{Adv}_h^{\text{prf}}(A(x_1, \dots)) = \Pr \left[ A^{h(K, \cdot)}(x_1, \dots) \Rightarrow 1 \right] - \Pr \left[ A^{\$}(x_1, \dots) \Rightarrow 1 \right],$$

where in the first case  $K \xleftarrow{\$} \{0, 1\}^c$  and in the second case the notation means that  $A$  is given as oracle a map chosen at random from  $\text{Maps}(\{0, 1\}^b, \{0, 1\}^c)$ .

*Overview* To start with, we ignore  $A_{hF}$  and upper bound  $\text{Coll}_F(M_1, M_2)$  as some appropriate function of the prf-advantage of a prf-adversary against  $h$  that takes  $M_1, M_2$  as input. We consider first the case that  $M_1 \subseteq M_2$  ( $M_1$  is a prefix of  $M_2$ ) and then the case that  $M_1 \not\subseteq M_2$ , building in each case a different adversaries.

*The Case  $M_1 \subseteq M_2$ .* We begin with some high-level intuition. Suppose  $M_1 \subseteq M_2$  with  $m_2 = \|M_2\|_b \geq 1 + m_1$ , where  $m_1 = \|M_1\|_b$ . The argument to upper bound  $\text{Coll}_{h^*}(M_1, M_2)$  has two parts. First, a hybrid argument is used to show that  $a[m_1] = h^*(K, M_1)$  is computationally close to random when  $K$  is drawn at random. Next, we imagine a game in which  $a[m_1]$  functions as a key to  $h$ . Let  $a[m_1 + 1] = h(a[m_1], M_2[m_1 + 1])$  and  $a[m_2] = h^*(a[m_1 + 1], M_2[m_1 + 2] \dots M_2[m_2])$ . Now, if  $a[m_2] = a[m_1]$  then we effectively have a way to recover the “key”  $a[m_1]$  given  $a[m_1 + 1]$ , amounting to a key-recovery attack on  $h(a[m_1], \cdot)$  based on one input–output example of this function. But being a PRF,  $h$  is also secure against key-recovery.

In the full proof that follows, we use the games and adversaries specified in Fig. 1. Adversaries  $A_1, A_2$  represent, respectively, the first and second parts of the argument outlined above, while  $A_3$  integrates the two.

**Claim 3.5.** *Let  $M_1, M_2 \in B^+$  with  $M_1 \subseteq M_2$  and  $1 + \|M_1\|_b \leq \|M_2\|_b$ . Suppose  $1 \leq l \leq \|M_1\|_b$ . Then*

$$\begin{aligned} \Pr \left[ A_1^{\$}(M_1, M_2, l) \Rightarrow 1 \right] &= \Pr \left[ G_1(M_1, M_2, l) \Rightarrow 1 \right] \\ \Pr \left[ A_1^{h(K, \cdot)}(M_1, M_2, l) \Rightarrow 1 \right] &= \Pr \left[ G_1(M_1, M_2, l - 1) \Rightarrow 1 \right]. \end{aligned}$$

Recall that the notation means that in the first case  $A_1$  gets as oracle  $g \xleftarrow{\$} \text{Maps}(\{0, 1\}^b, \{0, 1\}^c)$  and in the second case  $K \xleftarrow{\$} \{0, 1\}^c$ .

*Proof of Claim 3.5.*  $A_1^{\$}(M_1, M_2, l)$  sets  $a[l] = g(M_2[l])$ . If  $g$  is chosen at random then this is equivalent to the  $a[l] \xleftarrow{\$} \{0, 1\}^c$  assignment in  $G_1(M_1, M_2, l)$ . On the other hand if  $g = h(K, \cdot)$  for a random  $K$ , then  $K$  plays the role of  $a[l - 1]$  in  $G_1(M_1, M_2, l - 1)$ .

<p><b>Game</b> <math>G_1(M_1, M_2, l) \ // \ 0 \leq l \leq \ M_1\ _b</math>  <math>m_1 \leftarrow \ M_1\ _b; m_2 \leftarrow \ M_2\ _b</math>  <math>a[l] \xleftarrow{\\$} \{0, 1\}^c</math>          For <math>i = l + 1</math> to <math>m_2</math> do  <math>a[i] \leftarrow h(a[i - 1], M_2[i])</math>          If <math>a[m_1] = a[m_2]</math> then return 1          else return 0</p> <p><b>Adversary</b> <math>A_3^g(M_1, M_2)</math>  <math>m_1 \leftarrow \ M_1\ _b; m_2 \leftarrow \ M_2\ _b</math>  <math>l \xleftarrow{\\$} \{1, \dots, m_1 + 1\}</math>          If <math>l = m_1 + 1</math> then return <math>A_2^g(M_1, M_2)</math>          Else return <math>A_1^g(M_1, M_2, l)</math></p>	<p><b>Adversary</b> <math>A_1^g(M_1, M_2, l) \ // \ 1 \leq l \leq \ M_1\ _b</math>  <math>m_1 \leftarrow \ M_1\ _b; m_2 \leftarrow \ M_2\ _b</math>  <math>a[l] \leftarrow g(M_2[l])</math>          For <math>i = l + 1</math> to <math>m_2</math> do  <math>a[i] \leftarrow h(a[i - 1], M_2[i])</math>          If <math>a[m_1] = a[m_2]</math> then return 1          else return 0</p> <p><b>Adversary</b> <math>A_2^g(M_1, M_2)</math>  <math>m_1 \leftarrow \ M_1\ _b; m_2 \leftarrow \ M_2\ _b</math>  <math>a[m_1 + 1] \leftarrow g(M_2[m_1 + 1])</math>          For <math>i = m_1 + 2</math> to <math>m_2</math> do  <math>a[i] \leftarrow h(a[i - 1], M_2[i])</math>  <math>y \xleftarrow{\\$} B \setminus \{M_2[m_1 + 1]\}</math>          If <math>h(a[m_2], y) = g(y)</math> then return 1          else return 0</p>
--	---

**Fig. 1.** Games and adversaries taking input distinct messages  $M_1, M_2$  such that  $M_1 \subseteq M_2$ . The adversaries take an oracle  $g: \{0, 1\}^b \rightarrow \{0, 1\}^c$ .

**Claim 3.6.** *Let  $M_1, M_2 \in B^+$  with  $M_1 \subseteq M_2$  and  $1 + \|M_1\|_b \leq \|M_2\|_b$ . Then*

$$\Pr \left[ A_2^g(M_1, M_2) \Rightarrow 1 \right] = 2^{-c}$$

$$\Pr \left[ A_2^{h(K, \cdot)}(M_1, M_2) \Rightarrow 1 \right] \geq \Pr [ G_1(M_1, M_2, m_1) \Rightarrow 1 ].$$

*Proof of Claim 3.6.* Suppose  $g$  is chosen at random. Since  $y \neq M_2[m_1 + 1]$ , the quantity  $g(y)$  is not defined until the last line of the code of  $A_2$ , at which point  $h(a[m_2], y)$  is fixed, and thus the probability that the two are equal is  $2^{-c}$  due to the randomness of  $g(y)$ . Now suppose  $g = h(K, \cdot)$  for a random  $K$ . Think of  $K$  as playing the role of  $a[m_1]$  in  $G_1(M_1, M_2, m_1)$ . Then  $a[m_2] = K$  in  $A_2^g(M_1, M_2)$  exactly when  $a[m_1] = a[m_2]$  in  $G_1(M_1, M_2, m_1)$ , meaning exactly when the latter game returns 1. But if  $a[m_2] = K$  then certainly  $h(a[m_2], y) = h(K, y)$ , and the latter is  $g(y)$ , so  $A_1^g(M_1, M_2)$  will return 1. (However, it could be that  $h(a[m_2], y) = h(K, y)$  even if  $a[m_2] \neq K$ , which is why we have an inequality rather than an equality in the claim.)

**Claim 3.7.** *Let  $M_1, M_2 \in B^+$  with  $M_1 \subseteq M_2$  and  $1 + \|M_1\|_b \leq \|M_2\|_b$ . Let  $m_1 = \|M_1\|_b$ . Then*

$$\text{Adv}_h^{\text{prf}}(A_3(M_1, M_2)) \geq \frac{1}{m_1 + 1} (\text{Coll}_{h^*}(M_1, M_2) - 2^{-c}).$$

*Proof of Claim 3.7.* From the description of  $A_3$ , whether  $g = \$$  or  $g = h(K, \cdot)$ ,

$$\begin{aligned} & \Pr [ A_3^g(M_1, M_2) \Rightarrow 1 ] \\ &= \frac{1}{m_1 + 1} \left( \Pr [ A_2^g(M_1, M_2) \Rightarrow 1 ] + \sum_{l=1}^{m_1} \Pr [ A_1^g(M_1, M_2, l) \Rightarrow 1 ] \right). \end{aligned}$$

Now Claims 3.6 and 3.5 imply that  $\Pr [ A_3^{h(K, \cdot)}(M_1, M_2) \Rightarrow 1 ]$  is

$$\begin{aligned} & \geq \frac{1}{m_1 + 1} \left( \Pr [ G_1(M_1, M_2, m_1) \Rightarrow 1 ] + \sum_{l=1}^{m_1} \Pr [ G_1(M_1, M_2, l - 1) \Rightarrow 1 ] \right) \\ &= \frac{1}{m_1 + 1} \cdot \sum_{l=0}^{m_1} \Pr [ G_1(M_1, M_2, l) \Rightarrow 1 ]. \end{aligned} \tag{6}$$

On the other hand, Claims 3.6 and 3.5 also imply that  $\Pr [ A_3^{\$}(M_1, M_2) \Rightarrow 1 ]$  is

$$= \frac{1}{m_1 + 1} \left( 2^{-c} + \sum_{l=1}^{m_1} \Pr [ G_1(M_1, M_2, l) \Rightarrow 1 ] \right). \tag{7}$$

Subtracting (7) from (6) and exploiting the cancellation of terms, we get

$$\text{Adv}_h^{\text{prf}}(A_3(M_1, M_2)) \geq \frac{1}{m_1 + 1} (\Pr [ G_1(M_1, M_2, 0) \Rightarrow 1 ] - 2^{-c}).$$

Now examination of Game  $G_1(M_1, M_2, 0)$  shows that in this game,  $a[m_1] = h^*(a[0], M_1)$ ,  $a[m_2] = h^*(a[0], M_2)$ , and  $a[0]$  is selected at random. Since the game returns 1 iff  $a[m_1] = a[m_2]$ , the probability that it returns 1 is exactly  $\text{Coll}_{h^*}(M_1, M_2)$ .

*The Case  $M_1 \not\subseteq M_2$ .* For  $M_1, M_2 \in B^+$  with  $\|M_1\|_b \leq \|M_2\|_b$  and  $M_1 \not\subseteq M_2$ , we let  $\text{LCP}(M_1, M_2)$  denote the length of the longest common blockwise prefix of  $M_1, M_2$ , meaning the largest integer  $p$  such that  $M_1[1] \dots M_1[p] = M_2[1] \dots M_2[p]$  but  $M_1[p+1] \neq M_2[p+1]$ . Letting  $p = \text{LCP}(M_1, M_2)$ ,  $m_1 = \|M_1\|_b$  and  $m_2 = \|M_2\|_b$  we consider the following sequence of pairs:

$$\begin{aligned} & \underbrace{(0, 0), \dots, (p, p)}_{I_1(M_1, M_2)}, (p + 1, p + 1), \underbrace{(p + 2, p + 1), \dots, (m_1, p + 1)}_{I_2(M_1, M_2)}, \\ & \underbrace{(m_1, p + 2), \dots, (m_1, m_2)}_{I_3(M_1, M_2)}. \end{aligned} \tag{8}$$

<p><b>Game</b> <math>G_2(M_1, M_2, l_1, l_2)</math> // <math>(l_1, l_2) \in I(M_1, M_2)</math></p> <p>200 <math>m_1 \leftarrow \ M_1\ _b</math>; <math>m_2 \leftarrow \ M_2\ _b</math></p> <p>210 <math>a_1[l_1] \xleftarrow{\\$} \{0, 1\}^c</math></p> <p>220 If <math>(l_1, l_2) \in I_1(M_1, M_2)</math> then <math>a_2[l_2] \leftarrow a_1[l_1]</math> else <math>a_2[l_2] \xleftarrow{\\$} \{0, 1\}^c</math></p> <p>230 For <math>i = l_1 + 1</math> to <math>m_1</math> do <math>a_1[i] \leftarrow h(a_1[i-1], M_1[i])</math></p> <p>240 For <math>i = l_2 + 1</math> to <math>m_2</math> do <math>a_2[i] \leftarrow h(a_2[i-1], M_2[i])</math></p> <p>250 If <math>a_1[m_1] = a_2[m_2]</math> then return 1 else return 0</p> <p><b>Adversary</b> <math>A_4^g(M_1, M_2, l_1, l_2)</math> // <math>(l_1, l_2) \in I^*(M_1, M_2)</math></p> <p>a00 <math>m_1 \leftarrow \ M_1\ _b</math>; <math>m_2 \leftarrow \ M_2\ _b</math>; <math>p \leftarrow \text{LCP}(M_1, M_2)</math></p> <p>a10 If <math>(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_2(M_1, M_2)</math>  then <math>a_1[l_1] \leftarrow g(M_1[l_1])</math> else <math>a_1[l_1] \xleftarrow{\\$} \{0, 1\}^c</math></p> <p>a20 If <math>(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_3(M_1, M_2)</math>  then <math>a_2[l_2] \leftarrow g(M_2[l_2])</math> else <math>a_2[l_2] \xleftarrow{\\$} \{0, 1\}^c</math></p> <p>a30 For <math>i = l_1 + 1</math> to <math>m_1</math> do <math>a_1[i] \leftarrow h(a_1[i-1], M_1[i])</math></p> <p>a40 For <math>i = l_2 + 1</math> to <math>m_2</math> do <math>a_2[i] \leftarrow h(a_2[i-1], M_2[i])</math></p> <p>a50 If <math>a_1[m_1] = a_2[m_2]</math> then return 1 else return 0</p>
---

**Fig. 2.** Games and adversaries taking input distinct messages  $M_1, M_2 \in B^+$  such that  $M_1 \not\subseteq M_2$  and  $\|M_1\|_b \leq \|M_2\|_b$ .

For  $j = 1, 2, 3$  we let  $I_j(M_1, M_2)$  be the set of pairs indicated above. We then let

$$\begin{aligned}
 I(M_1, M_2) &= I_1(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_2(M_1, M_2) \cup I_3(M_1, M_2) \\
 I^*(M_1, M_2) &= I(M_1, M_2) - \{(0, 0)\} \\
 I_1^*(M_1, M_2) &= I_1(M_1, M_2) - \{(0, 0)\}.
 \end{aligned}$$

We consider the pairs to be ordered as per (8), and for  $(l_1, l_2) \in I^*(M_1, M_2)$  we let  $\text{PrD}(l_1, l_2)$  denote the predecessor of  $(l_1, l_2)$ , meaning the pair immediately preceding  $(l_1, l_2)$  in the sequence of (8). Note that  $|I(M_1, M_2)| = m_1 + m_2 - p$ . We consider the games and adversary of Fig. 2.

**Claim 3.8.** *Let  $M_1, M_2 \in B^+$  with  $M_1 \not\subseteq M_2$ , and  $\|M_1\|_b \leq \|M_2\|_b$ . Suppose  $(l_1, l_2) \in I^*(M_1, M_2)$ . Let  $(l'_1, l'_2) = \text{PrD}(l_1, l_2)$ . Then*

$$\begin{aligned}
 \Pr \left[ A_4^{\$}(M_1, M_2, l_1, l_2) \Rightarrow 1 \right] &= \Pr \left[ G_2(M_1, M_2, l_1, l_2) \Rightarrow 1 \right] \\
 \Pr \left[ A_4^{h(K, \cdot)}(M_1, M_2, l_1, l_2) \Rightarrow 1 \right] &= \Pr \left[ G_2(M_1, M_2, l'_1, l'_2) \Rightarrow 1 \right].
 \end{aligned}$$

*Proof of Claim 3.8.* We begin by justifying the first equality, namely the one where  $g \xleftarrow{\$} \text{Maps}(\{0, 1\}^b, \{0, 1\}^c)$ . Let us compare the code of  $G_2(M_1, M_2, l_1, l_2)$  and  $A_4^{\$}(M_1, M_2, l_1, l_2)$ . Line a10 is equivalent to line 210 because  $g$  is random. Now consider line a20. If  $(l_1, l_2) \in I_1^*(M_1, M_2)$  then we have  $a_2[l_2] = a_1[l_1]$  just as per line 220. This is true because, in this case, we have  $M_1[l_1] = M_2[l_2]$  and hence  $a_2[l_2] = g(M_2[l_2]) = g(M_1[l_1]) = a_1[l_1]$ . If  $(l_1, l_2) = (p+1, p+1)$  then setting  $a_2[l_2] = g(M_2[l_2])$

```

Game  $G_3(M_1, M_2, l_1, l_2)$  //  $(l_1, l_2) \in I^*(M_1, M_2)$ 
300  $m_1 \leftarrow \|M_1\|_b$ ;  $m_2 \leftarrow \|M_2\|_b$ ;  $p \leftarrow \text{LCP}(M_1, M_2)$ ;  $K \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
310 If  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_2(M_1, M_2)$ 
    then  $a_1[l'_1] \leftarrow K$ ;  $a_1[l_1] \leftarrow h(a_1[l'_1], M_1[l_1])$  else  $a_1[l_1] \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
320 If  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_3(M_1, M_2)$ 
    then  $a_2[l'_2] \leftarrow K$ ;  $a_2[l_2] \leftarrow h(a_2[l'_2], M_2[l_2])$  else  $a_2[l_2] \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
330 For  $i = l_1 + 1$  to  $m_1$  do  $a_1[i] \leftarrow h(a_1[i-1], M_1[i])$ 
340 For  $i = l_2 + 1$  to  $m_2$  do  $a_2[i] \leftarrow h(a_2[i-1], M_2[i])$ 
350 If  $a_1[m_1] = a_2[m_2]$  then return 1 else return 0

Game  $G_4(M_1, M_2, l_1, l_2)$  //  $(l_1, l_2) \in I^*(M_1, M_2)$ 
400  $m_1 \leftarrow \|M_1\|_b$ ;  $m_2 \leftarrow \|M_2\|_b$ ;  $p \leftarrow \text{LCP}(M_1, M_2)$ ;  $K \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
410 If  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_2(M_1, M_2)$ 
    then  $a_1[l'_1] \leftarrow K$  else  $a_1[l'_1] \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
420 If  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_3(M_1, M_2)$ 
    then  $a_2[l'_2] \leftarrow K$  else  $a_2[l'_2] \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
430 For  $i = l'_1 + 1$  to  $m_1$  do  $a_1[i] \leftarrow h(a_1[i-1], M_1[i])$ 
440 For  $i = l'_2 + 1$  to  $m_2$  do  $a_2[i] \leftarrow h(a_2[i-1], M_2[i])$ 
450 If  $a_1[m_1] = a_2[m_2]$  then return 1 else return 0

Game  $G_5(M_1, M_2, l_1, l_2)$  //  $(l_1, l_2) \in I^*(M_1, M_2)$ 
500  $m_1 \leftarrow \|M_1\|_b$ ;  $m_2 \leftarrow \|M_2\|_b$ ;  $p \leftarrow \text{LCP}(M_1, M_2)$ 
510  $a_1[l'_1] \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
520 If  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\}$  then  $a_2[l'_2] \leftarrow a_1[l'_1]$  else  $a_2[l'_2] \stackrel{\$}{\leftarrow} \{0, 1\}^c$ 
530 For  $i = l'_1 + 1$  to  $m_1$  do  $a_1[i] \leftarrow h(a_1[i-1], M_1[i])$ 
540 For  $i = l'_2 + 1$  to  $m_2$  do  $a_2[i] \leftarrow h(a_2[i-1], M_2[i])$ 
550 If  $a_1[m_1] = a_2[m_2]$  then return 1 else return 0

```

**Fig. 3.** Games equivalent to  $G_2(M_1, M_2, l'_1, l'_2)$ , where  $M_1, M_2 \in B^+$  are such that  $M_1 \not\subseteq M_2$  and  $\|M_1\|_b \leq \|M_2\|_b$ , and  $(l_1, l_2) \in I^*(M_1, M_2)$ .

is equivalent to choosing  $a_2[l_2]$  at random because  $g$  is random and has not previously been invoked on  $M_2[l_2]$ . (We use here in a crucial way that the point  $M_1[p+1]$  on which  $g$  has previously been invoked is different from  $M_2[p+1]$ .) If  $(l_1, l_2) \in I_3(M_1, M_2)$  then setting  $a_2[l_2] = g(M_2[l_2])$  is equivalent to choosing  $a_2[l_2]$  at random because  $g$  is random and has not previously been invoked anywhere. (In this case,  $a_1[l_1]$  is chosen at random and not via  $g$ .) Otherwise,  $a_2[l_2]$  is chosen at random. We have shown that line a20 is equivalent to line 220. The rest of the code is the same in both cases.

Now we justify the second equality, namely the one where  $K$  is selected at random from  $\{0, 1\}^c$  and  $g = h(K, \cdot)$ . We will consider some auxiliary games, shown in Fig. 3. We claim that

$$\Pr \left[ A_4^{h(K, \cdot)}(M_1, M_2, l_1, l_2) \Rightarrow 1 \right] = \Pr [ G_3(M_1, M_2, l_1, l_2) \Rightarrow 1 ] \quad (9)$$

$$= \Pr [ G_4(M_1, M_2, l_1, l_2) \Rightarrow 1 ] \quad (10)$$

$$= \Pr [ G_5(M_1, M_2, l_1, l_2) \Rightarrow 1 ] \quad (11)$$

$$= \Pr [ G_2(M_1, M_2, l'_1, l'_2) \Rightarrow 1 ] \quad (12)$$

We now justify the above relations.

Since  $a_1[l'_1]$  and  $a_2[l'_2]$  in  $G_3(M_1, M_2, l_1, l_2)$  both equal the key  $K$ , the output of this game is the same as that of  $A_4^{h(K, \cdot)}(M_1, M_2, l_1, l_2)$ , justifying (9).

Game  $G_4(M_1, M_2, l_1, l_2)$  was obtained by dropping the application of  $h(K, \cdot)$  in 310, 320 and beginning the loops of 330, 340 at  $l'_1, l'_2$  rather than  $l_1, l_2$ , respectively. To justify (10) we consider some cases. If  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_2(M_1, M_2)$  then  $l'_1 = l_1 - 1$ , so dropping the application of  $h(K, \cdot)$  in 310 is compensated for by beginning the loop at 430 one step earlier. On the other hand if  $(l_1, l_2) \in I_3(M_1, M_2)$  then  $l'_1 = l_1 = m_1$ , so starting the loop of 430 at  $l'_1$  effectively makes no change, and we can also replace  $l_1$  by  $l'_1$  in the “else” clause as shown in 410. Similarly if  $(l_1, l_2) \in I_1^*(M_1, M_2) \cup \{(p+1, p+1)\} \cup I_3(M_1, M_2)$  then  $l'_2 = l_2 - 1$ , so dropping the application of  $h(K, \cdot)$  in 320 is compensated for by beginning the loop at 440 one step earlier. On the other hand if  $(l_1, l_2) \in I_2(M_1, M_2)$  then  $l'_2 = l_2 = p + 1$ , so starting the loop of 440 at  $l'_2$  effectively makes no change, and we can also replace  $l_2$  by  $l'_2$  in the “else” clause as shown in 420.

Rather than picking  $K$  upfront and assigning it to  $a_1[l'_1]$ , Game  $G_5(M_1, M_2, l_1, l_2)$  picks  $a_1[l'_1]$  directly at random and then adjusts its choice of  $a_2[l'_2]$  to ensure that it equals  $a_1[l'_1]$  whenever they were both set to  $K$ . This justifies (11). Finally the test at 520 is equivalent to testing whether  $(l'_1, l'_2) \in I_1(M_1, M_2)$ , justifying (12).

We now define prf-adversary  $A_5^g(M_1, M_2)$  against  $h$  as follows. It picks  $(l_1, l_2) \xleftarrow{\$} I^*(M_1, M_2)$  and returns  $A_4^g(M_1, M_2, l_1, l_2)$ .

**Claim 3.9.** *Let  $M_1, M_2 \in B^+$  with  $M_1 \not\subseteq M_2$  and  $\|M_1\|_b \leq \|M_2\|_b$ . Let  $m = \|M_1\|_b + \|M_2\|_b - \text{LCP}(M_1, M_2) - 1$ . Then*

$$\text{Adv}_h^{\text{prf}}(A_5) \geq \frac{1}{m} \cdot (\text{Coll}_{h^*}(M_1, M_2) - 2^{-c}).$$

*Proof of Claim 3.9.* Note that  $m = |I^*(M_1, M_2)|$ . By Claim 3.8 we have the following, where the sums are over  $(l_1, l_2) \in I^*(M_1, M_2)$  and we are letting  $\text{PD}^j(l_1, l_2)$  be the  $j$ -th component of  $\text{PD}(l_1, l_2)$  for  $j = 1, 2$ :

$$\begin{aligned} \text{Adv}_h^{\text{prf}}(A_5) &= \frac{1}{m} \cdot \sum \Pr \left[ A_4^{h(K, \cdot)}(M_1, M_2, l_1, l_2) \Rightarrow 1 \right] - \frac{1}{m} \cdot \sum \Pr \left[ A_4^g(M_1, M_2, l_1, l_2) \Rightarrow 1 \right] \\ &= \frac{1}{m} \cdot \sum \Pr \left[ G_2(M_1, M_2, \text{PD}^1(l_1, l_2), \text{PD}^2(l_1, l_2)) \Rightarrow 1 \right] \\ &\quad - \frac{1}{m} \cdot \sum \Pr [ G_2(M_1, M_2, l_1, l_2) \Rightarrow 1 ] \\ &= \frac{1}{m} \cdot (\Pr [ G_2(M_1, M_2, 0, 0) \Rightarrow 1 ] - \Pr [ G_2(M_1, M_2, m_1, m_2) \Rightarrow 1 ]), \end{aligned}$$

where  $m_1 = \|M_1\|_b$  and  $m_2 = \|M_2\|_b$ . Examination of Game  $G_2(M_1, M_2, 0, 0)$  shows that in this game,  $a_1[m_1] = h^*(a_1[0], M_1)$ ,  $a_2[m_2] = h^*(a_2[0], M_2)$ , and  $a_1[0] = a_2[0]$  is selected at random. Since the game returns 1 iff  $a_1[m_1] = a_2[m_2]$ , the probability that it returns 1 is exactly  $\text{Coll}_{h^*}(M_1, M_2)$ . On the other hand, the values  $a_1[m_1]$  and  $a_2[m_2]$

in  $G_2(M_1, M_2, m_1, m_2)$  are chosen independently at random, and so the probability that they are equal, which is the probability this game returns 1, is  $2^{-c}$ .

*Putting it Together* We consider the following adversary  $A_6$  that takes input any distinct  $M_1, M_2 \in B^+$  such that  $\|M_1\|_b \leq \|M_2\|_b$ :

**Adversary**  $A_6^g(M_1, M_2)$   
 If  $M_1 \subseteq M_2$  then return  $A_3^g(M_1, M_2)$   
 Else return  $A_5^g(M_1, M_2)$

Now let  $A^*$  be any au-adversary against  $h^*$  such that the two messages output by  $A^*$  are at most  $n_1, n_2$  blocks long, respectively, where  $1 \leq n_1, n_2 \leq n$  and  $n$  is as in the Lemma statement. We assume wlog that the two messages  $M_1, M_2$  output by  $A^*$  are always distinct, in  $B^+$ , and satisfy  $\|M_1\|_b \leq \|M_2\|_b$ . Then we have

$$\begin{aligned} \mathbf{Adv}_{h^*}^{\text{au}}(A^*) &= \sum_{M_1 \subseteq M_2} \text{Coll}_{h^*}(M_1, M_2) \cdot \Pr[M_1, M_2] \\ &\quad + \sum_{M_1 \not\subseteq M_2} \text{Coll}_{h^*}(M_1, M_2) \cdot \Pr[M_1, M_2] \end{aligned}$$

where  $\Pr[M_1, M_2]$  denotes the probability that  $A^*$  outputs  $(M_1, M_2)$ . Now use Claims 3.7 and 3.9, and also the assumptions  $\|M_1\|_b \leq n_1$ ,  $\|M_2\|_b \leq n_2$  and  $n_2 \geq 1$  from the lemma statement, to get

$$\begin{aligned} \mathbf{Adv}_{h^*}^{\text{au}}(A^*) &\leq \sum_{M_1 \subseteq M_2} \left[ (n_1 + n_2 - 1) \cdot \mathbf{Adv}_h^{\text{prf}}(A_3(M_1, M_2)) + 2^{-c} \right] \cdot \Pr[M_1, M_2] \\ &\quad + \sum_{M_1 \not\subseteq M_2} \left[ (n_1 + n_2 - 1) \cdot \mathbf{Adv}_h^{\text{prf}}(A_5(M_1, M_2)) + 2^{-c} \right] \cdot \Pr[M_1, M_2] \\ &= \sum_{M_1, M_2} \left[ (n_1 + n_2 - 1) \cdot \mathbf{Adv}_h^{\text{prf}}(A_6(M_1, M_2)) + 2^{-c} \right] \cdot \Pr[M_1, M_2]. \end{aligned} \tag{13}$$

Let  $A$ , given oracle  $g$ , run  $A^*$  to get  $(M_1, M_2)$  and then return  $A_6^g(M_1, M_2)$ . Equation (2) follows from (13), establishing the blackbox claim of the lemma. The time complexity of  $A$  remains around  $t$  because, by our convention, the time complexity is that of the overlying experiment, so that of  $A^*$  includes the time to compute  $T_F$  on the messages that  $A^*$  outputs. The non-blackbox case uses a standard ‘‘coin-fixing’’ argument. Let  $M_1^*, M_2^* \in B^+$  be distinct messages such that  $\|M_1^*\|_b \leq \|M_2^*\|_b \leq n$  and

$$\mathbf{Adv}_h^{\text{prf}}(A_6(M_1, M_2)) \leq \mathbf{Adv}_h^{\text{prf}}(A_6(M_1^*, M_2^*)) \tag{14}$$

for all distinct  $M_1, M_2 \in B^+$  with  $\|M_1\|_b \leq \|M_2\|_b \leq n$ . Now let  $A$  be the adversary that has  $M_1^*, M_2^*$  hardwired in its code and, given oracle  $g$ , returns  $A_6^g(M_1^*, M_2^*)$ .



<p><b>Game <math>G_0</math></b>  <math>K_{\text{in}} \xleftarrow{\\$} \{0, 1\}^k</math>  <math>K_{\text{out}} \xleftarrow{\\$} \{0, 1\}^c</math>  <b>On query <math>M</math>:</b>                  Reply <math>h(K_{\text{out}}, F(K_{\text{in}}, M))</math></p>	<p><b>Game <math>G_1</math></b>  <math>K_{\text{in}} \xleftarrow{\\$} \{0, 1\}^k</math>  <math>f \xleftarrow{\\$} \text{Maps}(\{0, 1\}^b, \{0, 1\}^c)</math>  <b>On query <math>M</math>:</b>                  Reply <math>f(F(K_{\text{in}}, M))</math></p>	<p><b>Game <math>G_2</math></b>  <math>g \xleftarrow{\\$} \text{Maps}(D, \{0, 1\}^c)</math>  <b>On query <math>M</math>:</b>                  Reply <math>g(M)</math></p>
--	--	---

**Fig. 4.** Games  $G_0, G_1, G_2$  for the proof of Lemma 3.2.

### 3.3. Proof of Lemma 3.2

Game  $G_0$  of Fig. 4 implements an oracle for  $hF(K_{\text{out}} \| K_{\text{in}}, \cdot)$  with the keys chosen at random, while Game  $G_2$  implements an oracle for a random function. So

$$\begin{aligned}
 & \text{Adv}_{hF}^{\text{prf}}(A_{hF}) \\
 &= \Pr \left[ A_{hF}^{G_0} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G_2} \Rightarrow 1 \right] \\
 &= \left( \Pr \left[ A_{hF}^{G_0} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G_1} \Rightarrow 1 \right] \right) \\
 &\quad + \left( \Pr \left[ A_{hF}^{G_1} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G_2} \Rightarrow 1 \right] \right), \tag{15}
 \end{aligned}$$

where in the last step we simply subtracted and then added back in the value  $\Pr \left[ A_{hF}^{G_1} \Rightarrow 1 \right]$ .

Let  $A_h$ , given an oracle for a function  $f: B \rightarrow \{0, 1\}^c$ , pick  $K_{\text{in}} \xleftarrow{\$} \{0, 1\}^k$ . It then runs  $A_{hF}$ , replying to oracle query  $M$  by  $f(F(K_{\text{in}}, M))$ , and returns whatever output  $A_{hF}$  returns. A simple analysis shows that

$$\text{Adv}_h^{\text{prf}}(A_h) = \Pr \left[ A_{hF}^{G_0} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G_1} \Rightarrow 1 \right]. \tag{16}$$

The main part of the proof is to construct an au-adversary  $A'_F$  against  $F$  such that

$$\Pr \left[ A_{hF}^{G_1} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G_2} \Rightarrow 1 \right] \leq \binom{q}{2} \cdot \text{Adv}_F^{\text{au}}(A'_F). \tag{17}$$

Towards constructing  $A'_F$ , consider Games  $G_3, G_4$ , and  $G_5$  of Fig. 5. (Game  $G_3$  is defined by the code on the left of the Figure. Game  $G_4$  is the same except that the boxed code-statement is omitted.) We will assume now that a prf-adversary never repeats an oracle query. This is wlog, and is used below without explicit mention.

**Claim 3.10.** *Game  $G_4$  is equivalent to Game  $G_2$ , while Game  $G_3$  is equivalent to Game  $G_1$ .*

*Proof of Claim 3.10.* In Game  $G_4$ , the “If” statement does nothing beyond setting the bad flag, and the reply to query  $M_s$  is always the random value  $Z_s$ . Thus, Game  $G_4$  implements a random function just like Game  $G_2$ . Game  $G_3$  returns random values

<p><b>Games <math>G3, G4</math></b></p> <p><math>K_{\text{in}} \xleftarrow{\\$} \{0, 1\}^k ; s \leftarrow 0</math></p> <p><math>Z_1, \dots, Z_q \xleftarrow{\\$} \{0, 1\}^c</math></p> <p><b>On query <math>M</math>:</b></p> <p><math>s \leftarrow s + 1 ; M_s \leftarrow M</math></p> <p><math>Y_s \leftarrow F(K_{\text{in}}, M_s)</math></p> <p>If <math>(\exists r &lt; s : Y_r = Y_s)</math> then</p> <p style="padding-left: 2em;"><math>\text{bad} \leftarrow \text{true} ; \boxed{Z_s \leftarrow Z_r}</math></p> <p>Reply <math>Z_s</math></p>	<p><b>Game <math>G5</math></b></p> <p><math>s \leftarrow 0</math></p> <p><math>Z_1, \dots, Z_q \xleftarrow{\\$} \{0, 1\}^c</math></p> <p><b>On query <math>M</math>:</b></p> <p><math>s \leftarrow s + 1 ; M_s \leftarrow M</math></p> <p>Reply <math>Z_s</math></p>
---	--

**Fig. 5.** Games  $G3, G4$  are defined by the code on the left, where Game  $G3$  includes the boxed statement, while Game  $G4$  does not.

except that it also ensures that if  $F(K_{\text{in}}, M_i) = F(K_{\text{in}}, M_j)$  then the answers to queries  $M_i, M_j$  are the same. Thus, it is equivalent to Game  $G1$ .

Now we have:

$$\Pr \left[ A_{hF}^{G1} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G2} \Rightarrow 1 \right] = \Pr \left[ A_{hF}^{G3} \Rightarrow 1 \right] - \Pr \left[ A_{hF}^{G4} \Rightarrow 1 \right] \quad (18)$$

$$\leq \Pr \left[ A_{hF}^{G4} \text{ sets bad} \right]. \quad (19)$$

Above, (18) is by Claim 3.10. Since  $G3, G4$  differ only in statements that follow the setting of **bad**, (19) follows from the Fundamental Lemma of Game Playing [11].

We define au-adversary  $A'_F$  against  $F$ , as follows: It runs  $A_{hF}^{G5}$ , then picks at random  $i, j$  subject to  $1 \leq i < j \leq q$ , and finally outputs the messages  $M_i, M_j$ . In other words, it runs  $A_{hF}$ , replying to the oracle queries of the latter with random values, and then outputs a random pair of messages that  $A_{hF}$  queries to its oracle. (In order for  $M_i, M_j$  to always be defined, we assume  $A_{hF}$  always makes *exactly*  $q$  oracle queries rather than at most  $q$  where by “always” we mean no matter how its oracle queries are replied to. This is wlog.) We claim that

$$\Pr \left[ A_{hF}^{G4} \text{ sets bad} \right] \leq \binom{q}{2} \cdot \text{Adv}_F^{\text{au}}(A'_F). \quad (20)$$

Combining (19) and (20) yields (17). We now justify (20). Intuitively, it is true because  $i, j$  are chosen at random after the execution of  $A_{hF}$  is complete, so  $A_{hF}$  has no information about them. A rigorous proof, however, needs a bit more work. Consider the experiment defining the au-advantage of  $A'_F$ . (Namely, we run  $A_{hF}^{G5}$ , pick  $i, j$  at random subject to  $1 \leq i < j \leq q$ , and then pick  $K_{\text{in}} \xleftarrow{\$} \{0, 1\}^k$ .) In this experiment, consider the following events defined for  $1 \leq \alpha < \beta \leq q$ :

$$C_{\alpha, \beta} : F(K_{\text{in}}, M_\alpha) = F(K_{\text{in}}, M_\beta)$$

$$C : \bigvee_{1 \leq \alpha < \beta \leq q} C_{\alpha, \beta}.$$

Notice that the events “ $C_{\alpha,\beta} \wedge (i, j) = (\alpha, \beta)$ ” ( $1 \leq \alpha < \beta \leq q$ ) are disjoint. (Even though the events  $C_{\alpha,\beta}$  for  $1 \leq \alpha < \beta \leq q$  are not.) Thus:

$$\begin{aligned} \mathbf{Adv}_F^{\text{au}}(A'_F) &= \Pr \left[ \bigvee_{1 \leq \alpha < \beta \leq q} (C_{\alpha,\beta} \wedge (i, j) = (\alpha, \beta)) \right] \\ &= \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} \wedge (i, j) = (\alpha, \beta) ]. \end{aligned}$$

Since  $i, j$  are chosen at random after the execution of  $A_{hF}^{G5}$  is complete, the events “ $(i, j) = (\alpha, \beta)$ ” and  $C_{\alpha,\beta}$  are independent. Thus the above equals

$$\begin{aligned} \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} ] \cdot \Pr [ (i, j) = (\alpha, \beta) ] &= \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} ] \cdot \frac{1}{\binom{q}{2}} \\ &= \frac{1}{\binom{q}{2}} \cdot \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} ] \\ &\geq \frac{1}{\binom{q}{2}} \cdot \Pr [ C ]. \end{aligned}$$

The proof of (20) is concluded by noting that  $\Pr [ C ]$  equals  $\Pr [ G4 \text{ sets bad} ]$ .

To obtain the blackbox claim of the lemma, we set  $A_F = A'_F$ . Combining (15), (16) and (17) we then get (3). We neglect the small overhead in time complexity for  $A_F$  compared to  $A_{hF}$ . The coin-fixing argument for the non-blackbox case is as follows. We note that

$$\mathbf{Adv}_F^{\text{au}}(A'_F) = \mathbf{E}_{M_1, M_2} [ \mathbf{Coll}_F(M_1, M_2) ]$$

where the expectation is over  $(M_1, M_2) \stackrel{\$}{\leftarrow} A'_F$ . Thus there must exist  $M_1, M_2 \in B^+$  such that  $\mathbf{Coll}_F(M_1, M_2) \geq \mathbf{Adv}_F^{\text{au}}(A'_F)$ , and these messages are distinct because  $A_{hF}$  never repeats an oracle query. Let  $A_F$  be the au-adversary that has  $M_1, M_2$  hardwired as part of its code and, when run, simply outputs these messages and halts. Then

$$\mathbf{Adv}_F^{\text{au}}(A'_F) \leq \mathbf{Adv}_F^{\text{au}}(A_F). \tag{21}$$

Furthermore the time complexity of  $A_F$  is  $O(mT_h)$ . (By our convention, the time complexity is that of the overlying experiment, so includes the time to compute  $T_F$  on the messages that  $A_F$  outputs.) Combining (15), (16), (17) and (21) we get (3).

### 3.4. Proof of Theorem 3.4

Define  $F: \{0, 1\}^c \times B^+ \rightarrow \{0, 1\}^b$  by  $F(K_{\text{in}}, M) = h^*(K_{\text{in}}, M) \parallel \text{fpad}$ . Note  $\mathbf{GNMAC} = hF$ . Consider the proof of Lemma 3.2 with  $A_{\mathbf{GNMAC}}$  playing the role of  $A_{hF}$ . We assume as usual that  $A_{hF}$  does not repeat an oracle query and makes exactly  $q$  oracle queries. Let  $A_1$  be the adversary  $A_h$  from that proof. Referring also to the games in that proof,

we have

$$\mathbf{Adv}_{\text{GNMAC}}^{\text{prf}}(A_{hF}) \leq \mathbf{Adv}_h^{\text{prf}}(A_1) + \Pr \left[ A_{hF}^{G4} \text{ sets bad} \right]. \tag{22}$$

The non-blackbox case of Lemma 3.1 tells us that there is an adversary  $A_2$ , with resource bounds as in the theorem statement, such that, for all distinct  $M, M' \in B^+$  satisfying  $\|M\|_b, \|M'\|_b \leq m$ , we have

$$\text{Coll}_F(M, M') \leq (\|M\|_b + \|M'\|_b - 1) \cdot \mathbf{Adv}_h^{\text{prf}}(A_2) + \frac{1}{2^c}. \tag{23}$$

(We obtain  $A_2$  by letting  $M, M'$  maximize the above collision probability over the indicated domain and then applying Lemma 3.1 to the adversary  $A_{M, M'}^*$  that has  $M, M'$  hardwired in its code and just outputs them.) Let  $M_1, \dots, M_q$  denote the queries made by  $A_{hF}$  in its execution with  $G4$ . Regarding these as random variables over the coins of this execution, and also taking the probability over the choice of  $K_{\text{in}}$ , we have

$$\Pr \left[ A_{hF}^{G4} \text{ sets bad} \right] = \Pr \left[ \exists i < j : F(K_{\text{in}}, M_i) = F(K_{\text{in}}, M_j) \right] \tag{24}$$

$$\begin{aligned} &\leq \sum_{i < j} \Pr \left[ F(K_{\text{in}}, M_i) = F(K_{\text{in}}, M_j) \right] \\ &\leq \sum_{i < j} (n_i + n_j - 1) \cdot \mathbf{Adv}_h^{\text{prf}}(A_2) + \frac{1}{2^c} \tag{25} \\ &= \mathbf{Adv}_h^{\text{prf}}(A_2) \cdot \sum_{i < j} (n_i + n_j - 1) + \binom{q}{2} \frac{1}{2^c} \\ &= \mathbf{Adv}_h^{\text{prf}}(A_2) \cdot (q - 1) \cdot (n - q/2) + \binom{q}{2} \frac{1}{2^c}. \end{aligned}$$

Above, (24) is by the definition of game  $G4$ . We obtained (25) by applying (23). To do this, one can first write each term of the sum as itself a sum, over all distinct messages  $M, M'$ , of  $\text{Coll}_F(M, M')$ , weighted by the probability that  $M_i = M$  and  $M_j = M'$ , the latter taken only over the execution of  $A_{hF}$  with  $G4$ . Note we use our convention that resource bounds assumed on an adversary (in this case, the block lengths of its queries) hold not only in the game defining security but across all possible answers to queries, and hence in particular in its execution with  $G4$ .

### 3.5. Tightness of Bound

We can get concrete estimates of the security that Theorem 3.3 guarantees for GNMAC by making conjectures about the prf security of  $h$  and then applying the theorem. As an example, one such conjecture is that the best attack against  $h$  as a PRF is exhaustive key search. (Birthday attacks do not apply since  $h$  is not a family of permutations.) This means that  $\mathbf{Adv}_h^{\text{prf}}(A) \leq \bar{t} \cdot 2^{-c}$  for any prf-adversary  $A$  of time complexity  $t$  making  $q \leq \bar{t}$  queries, where  $\bar{t} = t/T_h$ . Let us refer to this as the EKS (exhaustive key search) conjecture on  $h$ .

Now let  $A_{\text{GNMAC}}$  be a prf-adversary against GNMAC who has time complexity  $t$  and makes at most  $q \approx \bar{t}$  queries, each of at most  $m$  blocks. The blackbox case of Theorem 3.3 says that the prf-advantage of  $A_{\text{GNMAC}}$  is at most  $O(mq^2\bar{t}/2^c) \approx O(mq^3/2^c)$ , which hits 1 when  $q \approx 2^{c/3}/m^{1/3}$ , meaning the bound justifies GNMAC up to roughly  $2^{c/3}/m^{1/3}$  queries. This falls short of the  $2^{c/2}/m^{1/2}$  queries of the best known attack on GNMAC, namely the birthday one of [33], indicating the bound is not tight.

If we use the non-blackbox case of Theorem 3.3 while continuing to make the EKS conjecture on  $h$ , the bound on the prf-advantage of  $A_{\text{GNMAC}}$  is  $O(m^2q^2T_h/2^c)$ , which hits 1 when  $q \approx 2^{c/2}/m$ , which is almost tight. This, indeed, was the estimate made in the preliminary version of this paper [3]. However, De, Trevisan and Tulsiani [17] use time-memory tradeoffs in the style of [2,22,24] to show that there exist better than exhaustive key search attacks on the prf security of any function family, and thus in particular on  $h$ . These results are non-constructive, and hence can be ignored if we are using blackbox reductions, but in the non-blackbox case one must take them into account in making conjectures on the prf security of  $h$ . This was pointed out by Koblitz and Menezes [26]. The EKS conjecture on  $h$  is thus not an appropriate starting point in the context of non-blackbox reductions. It is not clear to us what is.

In summary, while the non-blackbox case of Theorem 3.3 provides a tighter reduction, the gain is offset by being forced to conjecture poorer security of the starting  $h$ . The gap has been filled by Pietrzak [32], who gives blackbox reduction proofs of prf security for NMAC that he shows via matching attack to be tight. In the process he shows that an improvement to the bounds of Theorem 3.3 claimed by [26] was incorrect.

#### 4. MAC-Security of NMAC Under Weaker Assumptions

Since any PRF is a secure MAC [7,10], Theorem 3.3 implies that NMAC is a secure MAC if the compression function is a PRF. Here we show that NMAC is a secure MAC under a weaker than PRF assumption on the compression function —namely that it is a privacy-preserving MAC— coupled with the assumption that the hash function is cAU. This is of interest given the still numerous usages of HMAC as a MAC (rather than as a PRF). This result can be viewed as attempting to formalize the intuition given in [4, Remark 4.4].

##### 4.1. Privacy-Preserving MACs

*MAC Forgery* Recall that the mac-advantage of mac-adversary  $A$  against a family of functions  $f: Keys \times Dom \rightarrow Rng$  is

$$\text{Adv}_f^{\text{mac}}(A) = \Pr \left[ A^{f(K,\cdot), \text{VF}_f(K,\cdot,\cdot)} \text{ forges} : K \xleftarrow{\$} Keys \right].$$

The verification oracle  $\text{VF}_f(K, \cdot, \cdot)$  associated to  $f$  takes input  $M, T$ , returning 1 if  $f(K, M) = T$  and 0 otherwise. Queries to the first oracle are called mac queries, and ones to the second are called verification queries.  $A$  is said to forge if it makes a

verification query  $M, T$  the response to which is 1 but  $M$  was not previously a mac query. Note we allow multiple verification queries [10].

*Privacy-Preserving MACs* The privacy notion for MACs that we use adapts the notion of left-or-right indistinguishability of encryption [6] to functions that are deterministic, and was first introduced by [9] who called it indistinguishability under distinct chosen-plaintext attack. An oracle query of an ind-adversary  $A$  against family  $f: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^L$  is a pair of  $l$ -bit strings. The reply is provided by one or the other of the following games:

<p style="text-align: center;"><b>Game Left</b></p> <p style="text-align: center;"><math>K \xleftarrow{\\$} \{0, 1\}^c</math></p> <p><b>On query</b> <math>(x_0, x_1)</math>:</p> <p style="padding-left: 20px;">Reply <math>f(K, x_0)</math></p>	<p style="text-align: center;"><b>Game Right</b></p> <p style="text-align: center;"><math>K \xleftarrow{\\$} \{0, 1\}^c</math></p> <p><b>On query</b> <math>(x_0, x_1)</math>:</p> <p style="padding-left: 20px;">Reply <math>f(K, x_1)</math></p>
---	--

Each game has an initialization step in which it picks a key; it then uses this key in computing replies to all the queries made by  $A$ . The ind-advantage of  $A$  is

$$\text{Adv}_f^{\text{ind}}(A) = \Pr \left[ A^{\text{Right}} \Rightarrow 1 \right] - \Pr \left[ A^{\text{Left}} \Rightarrow 1 \right].$$

However, unlike for encryption, the oracles here are deterministic. So  $A$  can easily win (meaning, obtain a high advantage), for example by making a pair of queries of the form  $(x, z), (y, z)$ , where  $x, y, z$  are distinct, and then returning 1 iff the replies returned are the same. (We expect that  $h(K, x) \neq h(K, y)$  with high probability over  $K$  for functions  $h$  of interest, for example compression functions.) We fix this by simply outlawing such behavior. To be precise, let us say that  $A$  is *legitimate* if for any sequence  $(x_0^1, x_1^1), \dots, (x_0^q, x_1^q)$  of oracle queries that it makes,  $x_0^1, \dots, x_0^q$  are all distinct  $l$ -bit strings, and also  $x_1^1, \dots, x_1^q$  are all distinct  $l$ -bit strings. (As a test, notice that the adversary who queried  $(x, z), (y, z)$  was not legitimate.) It is to be understood henceforth that an ind-adversary means a legitimate one. When we say that  $f$  is privacy-preserving, we mean that the ind-advantage of any (legitimate) practical ind-adversary is low.

Privacy-preservation is not, by itself, a demanding property. For example, it is achieved by a constant family such as the one defined by  $f(K, x) = 0^L$  for all  $K, x$ . We will, however, want the property for families that are also secure MACs.

*PP-MAC < PRF* We claim that a privacy-preserving MAC (PP-MAC) is strictly weaker than a PRF, in the sense that any PRF is (a secure MAC [7, 10] and) privacy-preserving, but not vice-versa. This means that when (below) we assume that a compression function  $h$  is a PP-MAC, we are indeed assuming less of it than that it is a PRF. Let us now provide some details about the claims made above. First, the following is the formal statement corresponding to the claim that any PRF is privacy-preserving:

**Proposition 4.1.** *Let  $f: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^L$  be a family of functions, and  $A_{\text{ind}}$  an ind-adversary against it that makes at most  $q$  oracle queries and has time complexity at most  $t$ . Then there is a prf-adversary  $A_{\text{prf}}$  against  $f$  such that  $\text{Adv}_f^{\text{ind}}(A_{\text{ind}}) \leq 2 \cdot \text{Adv}_f^{\text{prf}}(A_{\text{prf}})$ .  $A_{\text{prf}}$  makes at most  $q$  oracle queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.*

The proof is a simple exercise and is omitted. Next we explain why a PP-MAC need not be a PRF. The reason (or one reason) is that if the output of a family of functions has some structure, for example, always ending in a 0 bit, it would disqualify the family as a PRF but need not preclude its being a PP-MAC. To make this more precise, let  $f: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^L$  be a PP-MAC. Define  $g: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^{L+1}$  by  $g(K, x) = f(K, x) \| 0$  for all  $K \in \{0, 1\}^k$  and  $x \in \{0, 1\}^l$ . Then  $g$  is also a PP-MAC, but is clearly not a PRF.

#### 4.2. Results

The following implies that if  $h$  is a PP-MAC and  $F$  is cAU then their composition  $hF$  is a secure MAC.

**Lemma 4.2.** *Let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  and  $F: \{0, 1\}^k \times D \rightarrow B$  be families of functions, and let  $hF: \{0, 1\}^{c+k} \times D \rightarrow \{0, 1\}^c$  be defined by*

$$hF(K_{\text{out}} \| K_{\text{in}}, M) = h(K_{\text{out}}, F(K_{\text{in}}, M))$$

for all  $K_{\text{out}} \in \{0, 1\}^c$ ,  $K_{\text{in}} \in \{0, 1\}^k$  and  $M \in D$ . Let  $A_{hF}$  be a mac-adversary against  $hF$  that makes at most  $q_{\text{mac}}$  mac queries and at most  $q_{\text{vf}}$  verification queries, with the messages in each of these queries being of length at most  $n$ . Suppose  $A_{hF}$  has time complexity at most  $t$ . Let  $q = q_{\text{mac}} + q_{\text{vf}}$  and assume  $2 \leq q < 2^b$ . Then there exists a mac-adversary  $A_1$  against  $h$ , an ind-adversary  $A_2$  against  $h$ , and an au-adversary  $A_F$  against  $F$  such that

$$\mathbf{Adv}_{hF}^{\text{mac}}(A_{hF}) \leq \mathbf{Adv}_h^{\text{mac}}(A_1) + \mathbf{Adv}_h^{\text{ind}}(A_2) + \binom{q}{2} \cdot \mathbf{Adv}_F^{\text{au}}(A_F). \quad (26)$$

$A_1$  makes at most  $q_{\text{mac}}$  mac queries and at most  $q_{\text{vf}}$  verification queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.  $A_2$  makes at most  $q$  oracle queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.  $A_F$  outputs messages of length at most  $n$  and makes 2 oracle queries. The time complexity of  $A_F$  is  $t$  under a blackbox reduction and  $O(T_F(n))$  under a non-blackbox reduction, where  $T_F(n)$  is the time to compute  $F$  on an  $n$ -bit input.

The proof is in Sect. 4.3. As a corollary we have the following, which says that if  $h$  is a PP-MAC and  $h^*$  is cAU then GNMAC is a secure MAC.

**Theorem 4.3.** *Assume  $b \geq c$  and let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  be a family of functions and let  $\text{fpad} \in \{0, 1\}^{b-c}$  be a fixed padding string. Let GNMAC:  $\{0, 1\}^{2c} \times B^+ \rightarrow \{0, 1\}^c$  be defined by*

$$\text{GNMAC}(K_{\text{out}} \| K_{\text{in}}, M) = h(K_{\text{out}}, h^*(K_{\text{in}}, M) \| \text{fpad})$$

for all  $K_{\text{out}}, K_{\text{in}} \in \{0, 1\}^c$  and  $M \in B^+$ . Let  $A_{\text{GNMAC}}$  be a mac-adversary against GNMAC that makes at most  $q_{\text{mac}}$  mac queries and at most  $q_{\text{vf}}$  verification queries, with



the messages in each of these queries being of at most  $m$  blocks. Suppose  $A_{\text{GNMAC}}$  has time complexity at most  $t$ . Let  $q = q_{\text{mac}} + q_{\text{vf}}$  and assume  $2 \leq q < 2^b$ . Then there exists a mac-adversary  $A_1$  against  $h$ , an ind-adversary  $A_2$  against  $h$ , and an au-adversary  $A^*$  against  $h^*$  such that

$$\text{Adv}_{\text{GNMAC}}^{\text{mac}}(A_{\text{GNMAC}}) \leq \text{Adv}_h^{\text{mac}}(A_1) + \text{Adv}_h^{\text{ind}}(A_2) + \binom{q}{2} \cdot \text{Adv}_{h^*}^{\text{au}}(A^*). \quad (27)$$

$A_1$  makes at most  $q_{\text{mac}}$  mac queries and at most  $q_{\text{vf}}$  verification queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.  $A_2$  makes at most  $q$  oracle queries, has time complexity at most  $t$  and is obtained via a blackbox reduction.  $A^*$  outputs messages of at most  $m$  blocks and makes 2 oracle queries. The time complexity of  $A^*$  is  $t$  under a blackbox reduction and  $O(mT_h)$  under a non-blackbox reduction, where  $T_h$  is the time for one computation of  $h$ .

We remark that Lemma 4.2 can be extended to show that  $hF$  is not only a MAC but itself privacy-preserving. (This assumes  $h$  is privacy-preserving and  $F$  is cAU. We do not prove this here.) This implies that GNMAC is privacy-preserving as long as  $h$  is privacy-preserving and  $h^*$  is cAU. This is potentially useful because it may be possible to show that a PP-MAC is sufficient to ensure security in some applications where HMAC is currently assumed to be a PRF.

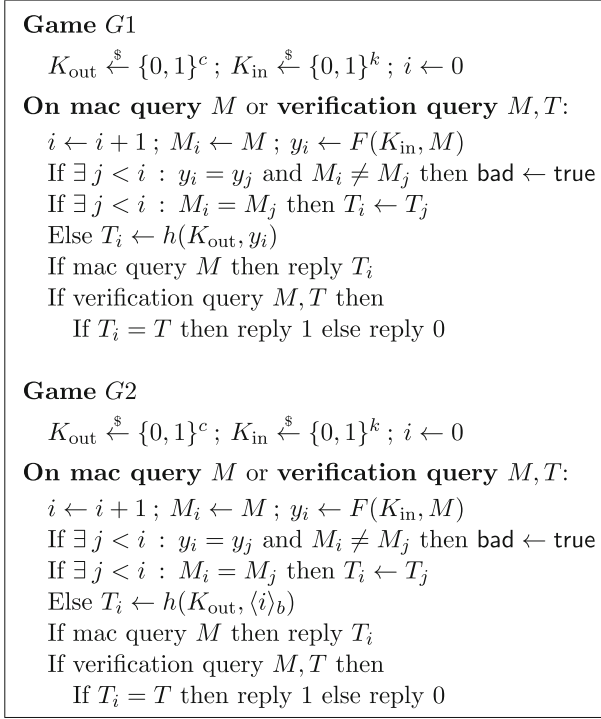
### 4.3. Proof of Lemma 4.2

A mac-adversary against  $h$  gets a mac oracle  $h(K_{\text{out}}, \cdot)$  and corresponding verification oracle  $\text{VF}_h(K_{\text{out}}, \cdot, \cdot)$ . By itself picking key  $K_{\text{in}}$  and invoking these oracles, it can easily simulate the mac oracle  $h(K_{\text{out}}, F(K_{\text{in}}, \cdot))$  and verification oracle  $\text{VF}_h(K_{\text{out}}, F(K_{\text{in}}, \cdot), \cdot)$  required by a mac-adversary against  $hF$ . This leads to the following natural construction of  $A_1$ :

**Adversary**  $A_1^{h(K_{\text{out}}, \cdot), \text{VF}_h(K_{\text{out}}, \cdot, \cdot)}$   
 $K_{\text{in}} \xleftarrow{\$} \{0, 1\}^k ; i \leftarrow 0$   
 Run  $A_{hF}$ , replying to its oracle queries as follows:  
**On mac query  $M$  or verification query  $M, T$ :**  
 $i \leftarrow i + 1 ; M_i \leftarrow M ; y_i \leftarrow F(K_{\text{in}}, M)$   
 If mac query  $M$  then reply  $h(K_{\text{out}}, y_i)$  to  $A_{hF}$   
 If verification query  $M, T$  then reply  $\text{VF}_h(K_{\text{out}}, y_i, T)$  to  $A_{hF}$

Consider the experiment defining the mac-advantage of  $A_1$ . Namely, choose  $K_{\text{out}} \xleftarrow{\$} \{0, 1\}^c$  and run  $A_1$  with oracles  $h(K_{\text{out}}, \cdot)$  and  $\text{VF}_h(K_{\text{out}}, \cdot, \cdot)$ . Let Coll (for ‘‘collision’’) be the event that there exist  $j, l$  such that  $y_j = y_l$  but  $M_j \neq M_l$ . Then

$$\begin{aligned} \text{Adv}_h^{\text{mac}}(A_1) &= \Pr [ A_1 \text{ forges } ] \\ &\geq \Pr [ A_{hF} \text{ forges } \wedge \overline{\text{Coll}} ] \\ &\geq \Pr [ A_{hF} \text{ forges } ] - \Pr [ \text{Coll} ] \\ &= \text{Adv}_{hF}^{\text{mac}}(A_{hF}) - \Pr [ \text{Coll} ]. \end{aligned} \quad (28)$$



**Fig. 6.** Games for the proof of Lemma 4.2.

The rest of the proof is devoted to upper bounding  $\Pr[\text{Coll}]$ . Consider the games  $G1, G2$  of Fig. 6, where we denote by  $\langle i \rangle_b$  the representation of integer  $i$  as a  $b$ -bit string. (The assumption  $q < 2^b$  made in the lemma statement means that we can always represent  $i$  this way in  $G2$ .) These games differ only in the manner in which tag  $T_i$  is computed. In  $G1$ , it is equal to  $hF(K_{\text{out}} \| K_{\text{in}}, M_i)$ . In  $G2$ , however, it is the result of applying  $h(K_{\text{out}}, \cdot)$  to the current value of the counter  $i$ , and as such does not depend on  $K_{\text{in}}$ . Now note that

$$\begin{aligned}
 \Pr[\text{Coll}] &= \Pr \left[ A_{hF}^{G1} \text{ sets bad} \right] \\
 &= \left( \Pr \left[ A_{hF}^{G1} \text{ sets bad} \right] - \Pr \left[ A_{hF}^{G2} \text{ sets bad} \right] \right) + \Pr \left[ A_{hF}^{G2} \text{ sets bad} \right].
 \end{aligned} \tag{29}$$

Now consider the adversaries  $A_2, A'_F$  described in Fig. 7. We claim that

$$\Pr \left[ A_{hF}^{G1} \text{ sets bad} \right] - \Pr \left[ A_{hF}^{G2} \text{ sets bad} \right] \leq \text{Adv}_h^{\text{ind}}(A_2) \tag{30}$$

$$\Pr \left[ A_{hF}^{G2} \text{ sets bad} \right] \leq \binom{q}{2} \cdot \text{Adv}_F^{\text{au}}(A'_F). \tag{31}$$

**Adversary  $A_2^{g(\cdot, \cdot)}$**   
 $K_{\text{in}} \xleftarrow{\$} \{0, 1\}^k$ ;  $i \leftarrow 0$   
 Run  $A_{hF}$ , replying to its oracle queries as follows:  
**On mac query  $M$  or verification query  $M, T$ :**  
 $i \leftarrow i + 1$ ;  $M_i \leftarrow M$ ;  $y_i \leftarrow F(K_{\text{in}}, M)$   
 If  $\exists j < i$  :  $y_i = y_j$  and  $M_i \neq M_j$  then return 1  
 If  $\exists j < i$  :  $M_i = M_j$  then  $T_i \leftarrow T_j$   
 Else  $T_i \leftarrow g(\langle i \rangle_b, y_i)$   
 If mac query  $M$  then reply  $T_i$  to  $A_{hF}$   
 If verification query  $M, T$  then  
     If  $T_i = T$  then reply 1 to  $A_{hF}$  else reply 0 to  $A_{hF}$   
 Return 0

**Adversary  $A'_F$**   
 $K_{\text{out}} \xleftarrow{\$} \{0, 1\}^c$ ;  $i \leftarrow 0$   
 Run  $A_{hF}$ , replying to its oracle queries as follows:  
**On mac query  $M$  or verification query  $M, T$ :**  
 $i \leftarrow i + 1$ ;  $M_i \leftarrow M$   
 If  $\exists j < i$  :  $M_i = M_j$  then  $T_i \leftarrow T_j$   
 Else  $T_i \leftarrow h(K_{\text{out}}, \langle i \rangle_b)$   
 If mac query  $M$  then reply  $T_i$  to  $A_{hF}$   
 If verification query  $M, T$  then  
     If  $T_i = T$  then reply 1 to  $A_{hF}$  else reply 0 to  $A_{hF}$   
 Pick  $i, j$  at random subject to  $1 \leq i < j \leq q$   
 Return  $M_i, M_j$

**Fig. 7.** Ind-adversary  $A_2$  against  $h$ , taking an oracle  $g$  that on input a pair of  $b$ -bit strings returns a  $c$ -bit string, and au-adversary  $A'_F$  against  $F$ , that outputs a pair of strings in  $D$ .

To obtain the blackbox claim of the lemma, we set  $A_F = A'_F$ . Clearly

$$\mathbf{Adv}_F^{\text{au}}(A'_F) \leq \mathbf{Adv}_F^{\text{au}}(A_F). \quad (32)$$

In the non-blackbox case, coin-fixing can be used just as in the proof of Lemma 3.2 to derive from  $A'_F$  an au-adversary  $A_F$  that has time complexity  $O(T_F(n))$  (and also outputs messages of  $n$  bits) such that (32) holds. Combining (32), (31), (30), (29) and (28) yields (26) and completes the proof of Lemma 4.2. It remains to prove (30), (31). We begin with the first of these.

Recall that an ind-adversary against  $h$  is given an oracle that takes as input a pair of  $b$ -bit strings  $x_0, x_1$ . We are denoting this oracle by  $g$ . Now it is easy to see that

$$\begin{aligned} \Pr \left[ A_2^{\text{Right}} \Rightarrow 1 \right] &= \Pr \left[ A_{hF}^{G1} \text{ sets bad} \right] \\ \Pr \left[ A_2^{\text{Left}} \Rightarrow 1 \right] &= \Pr \left[ A_{hF}^{G2} \text{ sets bad} \right], \end{aligned}$$

which implies (30). However, there is one important thing we still need to verify, namely that  $A_2$  is legitimate. So consider the sequence  $(x_1, y_1), (x_2, y_2), \dots$  of oracle queries it

makes. The left halves  $x_1, x_2, \dots$  are values of the counter  $i$  in different loop iterations and are thus strictly increasing (although not necessarily successive) and in particular different. On the other hand, the right half values  $y_1, y_2, \dots$  are distinct because as soon as  $y_i = y_j$  for some  $j < i$ , adversary  $A_2$  halts (and returns 1), never making an oracle query whose right half is  $y_i$ .

Next we turn to  $A'_F$ . In order for the messages  $M_i, M_j$  it returns to always be defined, we assume wlog that  $A_{hF}$  always makes exactly, rather than at most,  $q_{\text{mac}}$  mac queries and exactly, rather than at most,  $q_{\text{vf}}$  verification queries. Intuitively, (31) is true because  $i, j$  are chosen at random after the execution of  $A_{hF}$  is complete, so  $A_{hF}$  has no information about them. This can be made rigorous just as in the proof of Lemma 3.2, and the details follow. Consider the experiment defining the au-advantage of  $A'_F$ . In this experiment, consider the following events defined for  $1 \leq \alpha < \beta \leq q$ :

$$C_{\alpha,\beta} : F(K_{\text{in}}, M_\alpha) = F(K_{\text{in}}, M_\beta) \wedge M_\alpha \neq M_\beta$$

$$C : \bigvee_{1 \leq \alpha < \beta \leq q} C_{\alpha,\beta}.$$

Notice that the events “ $C_{\alpha,\beta} \wedge (i, j) = (\alpha, \beta)$ ” ( $1 \leq \alpha < \beta \leq q$ ) are disjoint. (Even though the events  $C_{\alpha,\beta}$  for  $1 \leq \alpha < \beta \leq q$  are not.) Thus:

$$\begin{aligned} \text{Adv}_F^{\text{au}}(A'_F) &= \Pr \left[ \bigvee_{1 \leq \alpha < \beta \leq q} (C_{\alpha,\beta} \wedge (i, j) = (\alpha, \beta)) \right] \\ &= \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} \wedge (i, j) = (\alpha, \beta) ]. \end{aligned}$$

Since  $i, j$  are chosen at random after the execution of  $A_{hF}$  is complete, the events “ $(i, j) = (\alpha, \beta)$ ” and  $C_{\alpha,\beta}$  are independent. Thus the above equals

$$\begin{aligned} \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} ] \cdot \Pr [ (i, j) = (\alpha, \beta) ] &= \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} ] \cdot \frac{1}{\binom{q}{2}} \\ &= \frac{1}{\binom{q}{2}} \cdot \sum_{1 \leq \alpha < \beta \leq q} \Pr [ C_{\alpha,\beta} ] \\ &\geq \frac{1}{\binom{q}{2}} \cdot \Pr [ C ]. \end{aligned}$$

The proof of (31) is concluded by noting that  $\Pr [ C ]$  equals  $\Pr [ G2 \text{ sets bad} ]$ .

## 5. Security of HMAC

In this section, we show how our security results about NMAC lift to corresponding ones about HMAC. We begin by recalling the observation of [4] as to how this works for HMAC with two independent keys, and then discuss how to extend this to the single-keyed version of HMAC.

*The Constructs* Let  $h: \{0, 1\}^c \times \{0, 1\}^b \rightarrow \{0, 1\}^c$  as usual denote the compression function. Let  $\text{pad}$  be the padding function as described in Sect. 3, so that  $s^* = s \parallel \text{pad}(|s|) \in B^+$  for any string  $s$ . Recall that the cryptographic hash function  $H$  associated to  $h$  is defined by  $H(M) = h^*(\text{IV}, M^*)$ , where  $\text{IV}$  is a  $c$ -bit initial vector that is fixed as part of the description of  $H$  and  $M$  is a string of any length up to some maximum length that is related to  $\text{pad}$ . (This maximum length is  $2^{64}$  for current hash functions.) Then  $\text{HMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = H(K_{\text{out}} \parallel H(K_{\text{in}} \parallel M))$ , where  $K_{\text{out}}, K_{\text{in}} \in \{0, 1\}^b$ . If we write this out in terms of  $h^*$  alone we get

$$\text{HMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = h^*(\text{IV}, K_{\text{out}} \parallel h^*(\text{IV}, K_{\text{in}} \parallel M \parallel \text{pad}(b + |M|)) \parallel \text{pad}(b + c)).$$

As with NMAC, the details of the padding conventions are not important to the security of HMAC as a PRF, and we will consider the more general construct  $\text{GHMAC}: \{0, 1\}^{2b} \times B^+ \rightarrow \{0, 1\}^c$  defined by

$$\text{GHMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = h^*(\text{IV}, K_{\text{out}} \parallel h^*(\text{IV}, K_{\text{in}} \parallel M) \parallel \text{fpad}) \quad (33)$$

for all  $K_{\text{out}}, K_{\text{in}} \in \{0, 1\}^b$  and all  $M \in B^+$ . Here  $\text{IV} \in \{0, 1\}^c$  and  $\text{fpad} \in \{0, 1\}^{b-c}$  are fixed strings. HMAC is a special case, via  $\text{HMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) = \text{GHMAC}(M \parallel \text{pad}(b + |M|))$  with  $\text{fpad} = \text{pad}(b + c)$ , and thus security properties of GHMAC (as a PRF or MAC) are inherited by HMAC, allowing us to focus on the former.

*The Dual Family* To state the results, it is useful to define  $\bar{h}: \{0, 1\}^b \times \{0, 1\}^c \rightarrow \{0, 1\}^c$ , the dual of family  $h$ , by  $\bar{h}(x, y) = h(y, x)$ . The assumption that  $h$  is a PRF when keyed by its data input is formally captured by the assumption that  $\bar{h}$  is a PRF.

### 5.1. Security of GHMAC

Let  $K'_{\text{out}} = h(\text{IV}, K_{\text{out}})$  and  $K'_{\text{in}} = h(\text{IV}, K_{\text{in}})$ . The observation of [4] is that

$$\begin{aligned} \text{GHMAC}(K_{\text{out}} \parallel K_{\text{in}}, M) &= h(K'_{\text{out}}, h^*(K'_{\text{in}}, M) \parallel \text{fpad}) \\ &= \text{GNMAC}(K'_{\text{out}} \parallel K'_{\text{in}}, M). \end{aligned} \quad (34)$$

This effectively reduces the security of GHMAC to GNMAC. Namely, if  $\bar{h}$  is a PRF and  $K_{\text{out}}, K_{\text{in}}$  are chosen at random, then  $K'_{\text{out}}, K'_{\text{in}}$  will be computationally close to random. Now (34) implies that if GNMAC is a PRF then so is GHMAC. The formal statement follows.

**Lemma 5.1.** *Assume  $b \geq c$  and let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  be a family of functions. Let  $\text{fpad} \in \{0, 1\}^{b-c}$  be a fixed padding string and  $\text{IV} \in \{0, 1\}^c$  a fixed initial vector. Let  $\text{GHMAC}: \{0, 1\}^{2b} \times B^+ \rightarrow \{0, 1\}^c$  be defined by (33) above. Let  $A$  be a prf-adversary against GHMAC that has time complexity at most  $t$ . Then there exists a prf-adversary  $A_{\bar{h}}$  against  $\bar{h}$  such that*

$$\text{Adv}_{\text{GHMAC}}^{\text{prf}}(A) \leq 2 \cdot \text{Adv}_{\bar{h}}^{\text{prf}}(A_{\bar{h}}) + \text{Adv}_{\text{GNMAC}}^{\text{prf}}(A).$$

<p><b>Game <math>G_0</math></b>  <b>On query <math>M</math>:</b>  <math>Z \xleftarrow{\\$} \{0, 1\}^c</math>                  Reply <math>Z</math></p>	<p><b>Game <math>G_1</math></b>  <math>K'_{\text{out}}, K'_{\text{in}} \xleftarrow{\\$} \{0, 1\}^c</math>  <b>On query <math>M</math>:</b>                  Reply <math>\text{GNMAC}(K'_{\text{out}} \  K'_{\text{in}}, M)</math></p>
<p><b>Game <math>G_2</math></b>  <math>K_{\text{in}} \xleftarrow{\\$} \{0, 1\}^b</math>  <math>K'_{\text{out}} \xleftarrow{\\$} \{0, 1\}^c</math>; <math>K'_{\text{in}} \leftarrow h(\text{IV}, K_{\text{in}})</math>  <b>On query <math>M</math>:</b>                  Reply <math>\text{GNMAC}(K'_{\text{out}} \  K'_{\text{in}}, M)</math></p>	<p><b>Game <math>G_3</math></b>  <math>K_{\text{in}}, K_{\text{out}} \xleftarrow{\\$} \{0, 1\}^b</math>  <math>K'_{\text{out}} \leftarrow h(\text{IV}, K_{\text{out}})</math>; <math>K'_{\text{in}} \leftarrow h(\text{IV}, K_{\text{in}})</math>  <b>On query <math>M</math>:</b>                  Reply <math>\text{GNMAC}(K'_{\text{out}} \  K'_{\text{in}}, M)</math></p>

**Fig. 8.** Games  $G_0, G_1, G_2, G_3$  for the proof of Lemma 5.1.

Furthermore,  $A_{\bar{h}}$  makes only 1 oracle query, this being IV, has time complexity at most  $t$  and is obtained via a blackbox reduction.

*Proof of Lemma 5.1.* Assume wlog that  $A$  never repeats an oracle query. Consider the games in Fig. 8, and let  $p_i = \Pr[A^{G_i} \Rightarrow 1]$  for  $0 \leq i \leq 3$ . Then

$$\text{Adv}_{\text{GHMAC}}^{\text{prf}}(A) = p_3 - p_0 = (p_3 - p_1) + (p_1 - p_0).$$

Clearly  $p_1 - p_0 = \text{Adv}_{\text{GNMAC}}^{\text{prf}}(A)$ . To complete the proof we construct  $A_{\bar{h}}$  so that

$$\text{Adv}_{\bar{h}}^{\text{prf}}(A_{\bar{h}}) = \frac{p_3 + p_2}{2} - \frac{p_2 + p_1}{2} = \frac{p_3 - p_1}{2}. \tag{35}$$

Given an oracle  $g: \{0, 1\}^c \rightarrow \{0, 1\}^c$ , adversary  $A_{\bar{h}}$  picks a bit  $c$  at random. Then it picks keys via

If  $c = 1$  then  $K'_{\text{out}} \leftarrow g(\text{IV})$ ;  $K_{\text{in}} \xleftarrow{\$} \{0, 1\}^b$ ;  $K'_{\text{in}} \leftarrow h(\text{IV}, K_{\text{in}})$

Else  $K'_{\text{out}} \xleftarrow{\$} \{0, 1\}^c$ ;  $K'_{\text{in}} \leftarrow g(\text{IV})$ .

Finally it runs  $A$  with oracle  $\text{GNMAC}(K'_{\text{out}} \| K'_{\text{in}}, \cdot)$ , and returns whatever  $A$  returns.

Combining this with Theorem 3.3 or Theorem 3.4 yields the result that GHMAC is a PRF assuming  $h, \bar{h}$  are both PRFs. Note that the PRF assumption on  $\bar{h}$  is mild because  $A_{\bar{h}}$  makes only one oracle query.

### 5.2. Single-Keyed HMAC

HMAC, GHMAC as described and analyzed above use two keys that are assumed to be chosen independently at random. However, HMAC is in fact usually implemented with these keys derived from a single  $b$ -bit key. Here we provide the first security proofs for this single-key version of HMAC.

Specifically, let  $\text{opad}, \text{ipad} \in \{0, 1\}^b$  be distinct, fixed and known constants. (Their particular values can be found in [4, 27] and are not important here.) Then the single-key

version of HMAC is defined by

$$\text{HMAC-1}(K, M) = \text{HMAC}(K \oplus \text{opad} \| K \oplus \text{ipad}, M).$$

As before, we look at this as a special case of a more general construct, namely GHMAC-1:  $\{0, 1\}^b \times B^+ \rightarrow \{0, 1\}^c$ , defined by

$$\text{GHMAC-1}(K, M) = \text{GHMAC}(K \oplus \text{opad} \| K \oplus \text{ipad}, M) \tag{36}$$

for all  $K \in \{0, 1\}^b$  and all  $M \in B^+$ . We now focus on GHMAC-1. We will show that GHMAC-1 inherits the security of GNMAC as long as  $\bar{h}$  is a PRF against an appropriate class of related key attacks. In such an attack, the adversary can obtain input–output examples of  $h$  under keys related to the target key. Let us recall the formal definitions following [8].

A related-key attack on a family of functions  $\bar{h}: \{0, 1\}^b \times \{0, 1\}^c \rightarrow \{0, 1\}^c$  is parameterized by a set  $\Phi \subseteq \text{Maps}(\{0, 1\}^b, \{0, 1\}^b)$  of *key-derivation* functions. We define the function  $\text{RK}: \Phi \times \{0, 1\}^b \rightarrow \{0, 1\}^b$  by  $\text{RK}(\phi, K) = \phi(K)$  for all  $\phi \in \Phi$  and  $K \in \{0, 1\}^b$ . A rka-adversary  $A_{\bar{h}}$  may make an oracle query of the form  $\phi, x$  where  $\phi \in \Phi$  and  $x \in \{0, 1\}^c$ . Its rka-advantage is defined by

$$\text{Adv}_{\bar{h}, \Phi}^{\text{rka}}(A_{\bar{h}}) = \Pr \left[ A_{\bar{h}}^{\bar{h}(\text{RK}(\cdot, K), \cdot)} \Rightarrow 1 \right] - \Pr \left[ A_{\bar{h}}^{G(\text{RK}(\cdot, K), \cdot)} \Rightarrow 1 \right].$$

In the first case,  $K$  is chosen at random from  $\{0, 1\}^b$  and the reply to query  $\phi, x$  of  $A_{\bar{h}}$  is  $\bar{h}(\phi(K), x)$ . In the second case,  $G \xleftarrow{\$} \text{Maps}(\{0, 1\}^b \times \{0, 1\}^c, \{0, 1\}^c)$  and  $K \xleftarrow{\$} \{0, 1\}^b$ , and the reply to query  $\phi, x$  of  $A_{\bar{h}}$  is  $G(\phi(K), x)$ . For any string  $s \in \{0, 1\}^b$  let  $\Delta_s: \{0, 1\}^b \rightarrow \{0, 1\}^b$  be defined by  $\Delta_s(K) = K \oplus s$ .

**Lemma 5.2.** *Assume  $b \geq c$  and let  $B = \{0, 1\}^b$ . Let  $h: \{0, 1\}^c \times B \rightarrow \{0, 1\}^c$  be a family of functions. Let  $\text{fpad} \in \{0, 1\}^{b-c}$  be a fixed padding string,  $\text{IV} \in \{0, 1\}^c$  a fixed initial vector, and  $\text{opad}, \text{ipad} \in \{0, 1\}^b$  fixed, distinct strings. Let GHMAC-1:  $\{0, 1\}^b \times B^+ \rightarrow \{0, 1\}^c$  be defined by (36) above. Let  $\Phi = \{\Delta_{\text{opad}}, \Delta_{\text{ipad}}\}$ . Let  $A$  be a prf-adversary against GHMAC-1 that has time complexity at most  $t$ . Then there exists a rka-adversary  $A_{\bar{h}}$  against  $\bar{h}$  such that*

$$\text{Adv}_{\text{GHMAC-1}}^{\text{prf}}(A) \leq \text{Adv}_{\bar{h}, \Phi}^{\text{rka}}(A_{\bar{h}}) + \text{Adv}_{\text{GNMAC}}^{\text{prf}}(A).$$

Furthermore,  $A_{\bar{h}}$  makes 2 oracle queries, these being  $\Delta_{\text{opad}}, \text{IV}$  and  $\Delta_{\text{ipad}}, \text{IV}$ , has time complexity at most  $t$  and is obtained via a blackbox reduction.

*Proof Lemma 5.2.* Adversary  $A_{\bar{h}}$  queries its oracle with  $\Delta_{\text{opad}}, \text{IV}$  and lets  $K'_{\text{out}}$  denote the value returned. It also queries its oracle with  $\Delta_{\text{ipad}}, \text{IV}$  and lets  $K'_{\text{in}}$  denote the value returned. It then runs  $A$ , answering the latter’s oracle queries via  $\text{GNMAC}(K'_{\text{out}} \| K'_{\text{in}}, \cdot)$ , and returns whatever  $A$  returns.



Combining this with Theorem 3.3 or Theorem 3.4 yields the result that GHMAC-1 is a PRF assuming  $h$  is a PRF and  $\bar{h}$  is a PRF under  $\Phi$ -restricted related-key attacks, where  $\Phi$  is as in Lemma 5.2. We remark that  $\Phi$  is a small set of simple functions, which is important because it is shown in [8] that if  $\Phi$  is too rich then no family can be a PRF under  $\Phi$ -restricted related-key attacks. Furthermore, the assumption on  $\bar{h}$  is rendered milder by the fact that  $A_{\bar{h}}$  makes only two oracle queries, in both of which the message is the same, namely is the initial vector.

### 5.3. Hashing of Keys

Single-key HMAC as defined and analyzed above is the core construct of the RFC 2104 standard [27]. The standard allows the option that longer keys are hashed and padded before use with HMAC. Some standards [30] build this step into the description. This step can be justified under additional assumptions, including (but not limited to) that the hash function is close to regular.

### 5.4. Lifting the Results of Sect. 4

The procedure above to lift the NMAC results of Sect. 3 to HMAC applies also to lift the results of Sect. 4 to HMAC. Specifically, if  $h$  is a PP-MAC,  $h^*$  is AU and  $\bar{h}$  is a PRF then GHMAC is a (privacy-preserving) MAC. Also if  $h$  is a PP-MAC,  $h^*$  is AU and  $\bar{h}$  is a PRF under  $\Phi$ -restricted related-key attacks, with  $\Phi$  as in Lemma 5.2, then GHMAC-1 is a (privacy-preserving) MAC. Note that the assumption on  $\bar{h}$  continues to be that it is a PRF or PRF against  $\Phi$ -restricted related-key attacks. (Namely, this has not been reduced to its being a PP-MAC.) This assumption is, however, mild in this context since (as indicated by Lemmas 5.2 and 5.1) it need only hold with respect to adversaries that make very few queries and these of a very specific type.

### 5.5. Remarks

Let  $h: \{0, 1\}^{128} \times \{0, 1\}^{512} \rightarrow \{0, 1\}^{128}$  denote the compression function of MD5 [34]. An attack by den Boer and Bosselaers [16] finds values  $x_0, x_1, K$  such that  $h(x_0, K) = h(x_1, K)$  but  $x_0 \neq x_1$ . In a personal communication, Rijmen has said that it seems possible to extend this to an attack that finds such  $x_0, x_1$  even when  $K$  is unknown. If so, this might translate into the following attack showing  $h$  is not a PRF when keyed by its data input. (That is,  $\bar{h}$  is not a PRF.) Given an oracle  $g: \{0, 1\}^{128} \rightarrow \{0, 1\}^{128}$ , the attacker would find  $x_0, x_1$  and obtain  $y_0 = g(x_0)$  and  $y_1 = g(x_1)$  from the oracle. It would return 1 if  $y_0 = y_1$  and 0 otherwise. How does this impact the above, where we are assuming  $\bar{h}$  is a PRF? Interestingly, the actual PRF assumptions we need on  $\bar{h}$  are so weak that even such an attack does not break them. In Lemma 5.1, we need  $\bar{h}$  to be a PRF only against adversaries that make just one oracle query. (Because  $A_{\bar{h}}$  makes only one query.) But the attack above makes two queries. On the other hand, in Lemma 5.2, we need  $\bar{h}$  to be a related-key PRF only against adversaries that make two related-key queries in both of which the 128-bit message for  $\bar{h}$  is the same, this value being the initial vector used by the hash function. Furthermore, the related-key functions must be  $\Delta_{\text{opad}}, \Delta_{\text{ipad}}$ . The above-mentioned attack, however, uses two different messages  $x_0, x_1$

and calls the oracle under the original key rather than the related keys. In summary, the attack does not violate the assumptions made in either of the lemmas.

## 6. Attacking Weak Collision Resistance

Recall that  $H$  represents the cryptographic hash function (eg, MD5, SHA-1) while  $H^*$  is the extended hash function, which is the hash function with the initial vector made explicit as an (additional) first input. Let us use the term *general collision-finding attack* to refer to an attack that finds collisions in  $H^*(IV, \cdot)$  for an arbitrary but given IV. As we discussed in Sect. 1, it was noted in [4,25] that any general collision-finding attack can be used to compromise the weak collision resistance (WCR) of  $H$ . (And since the known collision-finding attacks on MD5 and SHA-1 [37,38] do extend to general ones, the WCR of these functions is no more than their CR.) Here we recall the argument that shows this. It is a simple extension attack, and works as follows.

To compromise WCR of  $H$ , an attacker given an oracle for  $H^*(K, \cdot)$  under a hidden key  $K$  must output distinct  $M_1, M_2$  such that  $H^*(K, M_1) = H^*(K, M_2)$ . Our attacker picks some string  $x$  and calls its oracle to obtain  $IV = H^*(K, x)$ . Then it runs the given general collision-finding attack on input IV to obtain a collision  $X_1, X_2$  for  $H^*(IV, \cdot)$ . (That is,  $X_1, X_2$  are distinct strings such that  $H^*(IV, X_1) = H^*(IV, X_2)$ .) Now let  $M_1 = x \parallel \text{pad}(|x|) \parallel X_1 \parallel \text{pad}(|X_1|)$  and  $M_2 = x \parallel \text{pad}(|x|) \parallel X_2 \parallel \text{pad}(|X_2|)$ . (Here  $\text{pad}(n)$  is a padding string that when appended to a string of length  $n$  results in a string whose length is a positive multiple of  $b$  bits where  $b$  is the block length of the underlying compression function. The function  $\text{pad}$  is part of the description of the cryptographic hash function.) Then it follows that  $H^*(K, M_1) = H^*(K, M_2)$ .

We clarify that these attacks on the WCR of the hash function do not break HMAC. What these attacks show is that the WCR assumption made for the security proof of [4] is not true for MD5 and SHA-1. This means we lose the proof-based guarantee of [4], but it does not imply any weakness in the construct. Our results show that WCR of the iterated hash function is not necessary for the security of HMAC: pseudorandomness of the compression function suffices. This helps explain why no attacks have emerged on HMAC even when it is implemented with hash functions that are not WCR.

## 7. The Reduction-From-pf-PRF Proof

We sketch how one can obtain the result that  $h$  a PRF implies  $h^*$  is cAU using the result of [5] that says that  $h$  a PRF implies  $h^*$  is a pf-PRF (a PRF as long as no query of the adversary is a prefix of another query). We then compare this with the direct proof given in Sect. 3.2.

*The Result of [5]* A prf-adversary is said to be prefix-free if no query it makes is a prefix of another. The result of [5] is that if  $D$  is a prefix-free prf-adversary against  $h^*$  that makes at most  $q$  queries, each of at most  $m$  blocks, then there is a prf-adversary  $A$  against  $h$  such that

$$\text{Adv}_{h^*}^{\text{prf}}(D) \leq qm \cdot \text{Adv}_h^{\text{prf}}(A) \quad (37)$$

and  $A$  has about the same time complexity as  $D$ . (We remark that there is a typo in the statement of Theorem 3.1 of the proceedings version of [5] in this regard: the factor  $q$  is missing from the bound. This is, however, corrected in the on-line version of the paper.)

*The Reduction-From-pf-PRF* The result obtained via the reduction-from-pf-PRF proof will be slightly worse than the one of Lemma 3.1. Namely we claim that, under the same conditions as in that lemma, (2) is replaced by

$$\mathbf{Adv}_{h^*}^{\text{au}}(A^*) \leq 2 \cdot [\max(n_1, n_2) + 1] \cdot \mathbf{Adv}_h^{\text{prf}}(A) + \frac{1}{2^c}, \tag{38}$$

and, in the non-blackbox case, the time complexity of  $A$  increases from  $(n_1 + n_2)$  computations of  $h$  to  $2 \max(n_1, n_2)$  computations of  $h$ . For some intuition about the proof, imagine  $h^*$  were a PRF. (It is not.) Then  $\text{Coll}_{h^*}(M_1, M_2)$  would be about the same as the probability that  $f(M_1) = f(M_2)$  for a random function  $f$ , because otherwise the prf-adversary who queried its oracle with  $M_1, M_2$  and accepted iff the replies were the same would be successful. With  $h^*$  in fact only a pf-PRF, the difficulty is the case that  $M_1 \subseteq M_2$ , which renders the adversary just described not prefix-free. There is a simple (and well-known) observation —we will call it the extension trick— to get around this. Namely, assuming wlog  $M_1 \neq M_2$  and  $\|M_1\|_b \leq \|M_2\|_b$ , let  $x \in B$  be a block different from  $M_2[\|M_1\|_b + 1]$ , and let  $M'_1 = M_1 \|x$  and  $M'_2 = M_2 \|x$ . Then  $\text{Coll}_{h^*}(M_1, M_2) \leq \text{Coll}_{h^*}(M'_1, M'_2)$  but  $M_1$  is not a prefix of  $M_2$ . This leads to the prefix-free prf-adversary against  $h^*$  below:

**Adversary  $D^f$**

$(M_1, M_2) \xleftarrow{\$} A^*$

If  $M_1 \subseteq M_2$  then  $x \xleftarrow{\$} B \setminus \{M_2[\|M_1\|_b + 1]\}$ ;  $M'_1 \leftarrow M_1 \|x$ ;  $M'_2 \leftarrow M_2 \|x$

Else  $M'_1 \leftarrow M_1$ ;  $M'_2 \leftarrow M_2$

If  $f(M'_1) = f(M'_2)$  then return 1 else return 0

Here  $f: B^+ \rightarrow \{0, 1\}^c$  and we assume wlog that  $M_1, M_2 \in B^+$  are distinct messages with  $\|M_1\|_b \leq \|M_2\|_b$ . Now the result of [5] gives us a prf-adversary  $A$  against  $h$  such that (37) holds. Thus:

$$\begin{aligned} \mathbf{Adv}_{h^*}^{\text{au}}(A^*) - 2^{-c} &\leq \mathbf{Adv}_{h^*}^{\text{prf}}(D) \\ &\leq 2 \cdot [\max(n_1, n_2) + 1] \cdot \mathbf{Adv}_h^{\text{prf}}(A). \end{aligned}$$

Re-arranging terms yields (38).

The time complexity of  $A$  as per [5] is (essentially) the time complexity  $t$  of  $A^*$ , rather than being a small quantity independent of  $t$  as in Lemma 3.1. It is not clear whether or not the coin-fixing argument of our proof of can be applied to  $A$  to reduce this time complexity. (One would have to enter into the details of the proof of [5] to check.) However, instead, we can first modify  $A^*$  to an adversary that has embedded in its code a pair  $M_1, M_2 \in B^+$  of distinct messages that maximize  $\text{Coll}_{h^*}(M_1, M_2)$ . It just outputs these messages and halts. We then apply the argument above to this modified  $A^*$ , and now  $A$  will have time complexity of  $2 \max(n_1, n_2)$  computations of  $h$  plus minor overhead.

*Comparisons* For the case that  $M_1 \subseteq M_2$ , our direct proof (meaning the one of Sect. 3.2) uses a different (and novel) ideas as opposed to the extension trick, which leads to a factor of only  $n_1 + 1$  in the bound (Claim 3.7) in this case, as opposed to the  $2[\max(n_1, n_2) + 1]$  factor obtained via the reduction-from-pf-PRF proof. The difference can be significant in the case that  $M_2$  is long and  $M_1$  is short. In the case  $M_1 \not\subseteq M_2$  our direct proof relies heavily on ideas of [5], but avoids the intermediate reduction to the multi-oracle model they use and exploits the non-adaptive nature of the setting to improve the factor in the bound from  $2 \max(n_1, n_2)$  to  $n_1 + n_2 - 1$ . These improvements (overall, a constant factor) may not seem significant in this context. But they become significant when translated to NMAC, as evidenced by Theorem 3.4. As we saw, the latter gives appreciably better bounds than Theorem 3.3 in the case that authenticated messages have varying lengths. However, our direct proof (meaning, Lemma 3.1) is crucial to obtaining Theorem 3.4. The reduction-from-pf-PRF proof will only yield (a result that is slightly worse than) Theorem 3.3.

The reduction-from-pf-PRF proof is certainly simpler than our direct proof if one is willing to take the result of [5] as given. However, especially for a construct that is as widely standardized as HMAC, we think it is useful to have from-scratch proofs that are as easily verifiable as possible. If the measure of complexity is that of a from-scratch (i.e., self-contained) proof, we contend that our direct one (although not trivial) is simpler than that of [5]. (In particular because we do not use the multi-oracle model.) We remark that if a reader's primary interest is the simplest possible self-contained proof regardless of the quality of the bound, the way to get it is to use our direct proof for the case  $M_1 \not\subseteq M_2$  and then the extension trick (as opposed to our direct proof) for the case  $M_1 \subseteq M_2$ .

### Acknowledgements

Thanks to Ran Canetti, Hugo Krawczyk, Mridul Nandi, Vincent Rijmen, Phillip Rogaway, Victor Shoup, Paul Van Oorschot, the Crypto 2006 PC and the referees of the Journal of Cryptology for comments and references. Author supported in part by NSF Grants ANR-0129617, CCR-0208842 and CNS-0524765 and by an IBM Faculty Partnership Development Award.

### References

- [1] American National Standards Institution. ANSI X9.71, *Keyed hash message authentication code* (ABA, Washington, D.C., 2000)
- [2] E. Barkan, E. Biham, A. Shamir, Rigorous bounds on cryptanalytic time/memory tradeoffs, in C. Dwork ed., *Advances in Cryptology—CRYPTO'06*. LNCS, vol. 4117 (Springer, Berlin, Heidelberg, 2006)
- [3] M. Bellare, New proofs for NMAC and HMAC: security without collision-resistance. Preliminary version of this paper, in C. Dwork ed., *Advances in Cryptology—CRYPTO'06*. LNCS, vol. 4117 (Springer, Berlin, Heidelberg, 2006)
- [4] M. Bellare, R. Canetti, H. Krawczyk, Keying hash functions for message authentication, in N. Koblitz ed., *Advances in Cryptology—CRYPTO'96*. LNCS, vol. 1109 (Springer, Berlin, Heidelberg, 1996)

- [5] M. Bellare, R. Canetti, H. Krawczyk, Pseudorandom functions revisited: the cascade construction and its concrete security. <http://www-cse.ucsd.edu/users/mihir>. (Preliminary version in Proceedings of the 37th Symposium on Foundations of Computer Science (IEEE, 1996)
- [6] M. Bellare, A. Desai, E. Jorjipii, P. Rogaway, A concrete security treatment of symmetric encryption, in *Proceedings of the 38th Symposium on Foundations of Computer Science* (IEEE, 1997)
- [7] M. Bellare, J. Kilian, P. Rogaway, The security of the cipher block chaining message authentication code. *J. Comput. Syst. Sci.* 61(3), 362–399 (2000).
- [8] M. Bellare, T. Kohno, A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications, in E. Bihani ed., *Advances in Cryptology–EUROCRYPT’03*. LNCS, vol. 2656 (Springer, Berlin, Heidelberg, 2003)
- [9] M. Bellare, C. Namprempre, T. Kohno, Authenticated encryption in SSH: provably fixing the SSH binary packet protocol. *ACM Trans. Inf. Syst. Secur.* (TISSEC) 7(2), 206–241 (2004).
- [10] M. Bellare, O. Goldreich, A. Mityagin, The power of verification queries in message authentication and authenticated encryption. *Cryptology ePrint Archive*. Report 2004/309 (2004)
- [11] M. Bellare, P. Rogaway, The security of triple encryption and a framework for code-based game-playing proofs, in S. Vaudenay ed., *Advances in Cryptology–EUROCRYPT’06*. LNCS, vol. 4004 (Springer, Berlin, Heidelberg, 2006)
- [12] J. Black, S. Halevi, H. Krawczyk, T. Krovetz, P. Rogaway, UMAC: fast and secure message authentication, in M. Wiener ed., *Advances in Cryptology–CRYPTO’99*. LNCS, vol. 1666 (Springer, Berlin, Heidelberg, 1999)
- [13] J. Black, P. Rogaway, CBC MACs for arbitrary-length messages: The three-key constructions, in M. Bellare ed., *Advances in Cryptology–CRYPTO’00*. LNCS, vol. 1880 (Springer, Berlin, Heidelberg, 2000)
- [14] L. Carter, M. Wegman, Universal classes of hash functions. *J. Comput. Syst. Sci.* 18(2), 143–154 (1979).
- [15] I. Damgard, A design principle for hash functions, in G. Brassard ed., *Advances in Cryptology–CRYPTO’89*. LNCS, vol. 435, (Springer, New York, 1989)
- [16] B. den Boer, A. Bosselaers, Collisions for the compression function of MD5, in T. Helleseht ed., *Advances in Cryptology–EUROCRYPT’93*. LNCS, vol. 765 (Springer, Berlin Heidelberg, 1993)
- [17] A. De, L. Trevisan, M. Tulsiani, Time space tradeoffs for attacks against one-way functions and PRGs, in T. Rabin ed., *Advances in Cryptology–CRYPTO’10*. LNCS, vol. 6223 (Springer, Berlin, Heidelberg, 2010)
- [18] T. Dierks, C. Allen, The TLS protocol. Internet RFC 2246 (1999)
- [19] H. Dobbertin, A. Bosselaers, B. Preneel. RIPEMD-160: A strengthened version of RIPEMD, in D. Gollmann ed., *Fast Software Encryption’96*. LNCS, vol. 1039 (Springer, Berlin, Heidelberg, 1996)
- [20] Y. Dodis, R. Gennaro, J. Hastad, H. Krawczyk, T. Rabin, Randomness extraction and key derivation using the CBC, Cascade, and HMAC modes, in M. Franklin ed., *Advances in Cryptology–CRYPTO’04*. LNCS, vol. 3152 (Springer, Berlin, Heidelberg, 2004)
- [21] O. Goldreich, S. Goldwasser, S. Micali, How to construct random functions. *J. ACM* 33(4), 210–217 (1986).
- [22] A. Fiat, M. Naor, Rigorous time/space tradeoffs for inverting functions. *SIAM J. Comput.* 29(3), 790–803 (1999).
- [23] D. Harkins, D. Carrel, The internet key exchange (IKE). Internet RFC 2409 (1998)
- [24] M. Hellman, A cryptanalytic time-memory trade-off. *IEEE Trans. Inf. Theory* 26(4), 401–406 (1980).
- [25] S. Hirose, A note on the strength of weak collision resistance. *IEICE Trans. Fundam.* E87-A(5), 1092–1097 (2004)
- [26] N. Kobitz, A. Menezes, Another look at HMAC. *Cryptology ePrint Archive*. Report 2012/074 (2012)
- [27] H. Krawczyk, M. Bellare, R. Canetti. HMAC: Keyed-hashing for message authentication. Internet RFC 2104 (1997)
- [28] R. Merkle, One-way hash functions and DES, in G. Brassard ed., *Advances in Cryptology–CRYPTO’89*. LNCS, vol. 435 (Springer, New York, 1989). (Based on an unpublished paper from 1979 and the author’s Ph.D. Thesis, Stanford, 1979)
- [29] D. M’Raihi, M. Bellare, F. Hoornaert, D. Naccache, O. Ranen, HOTP: An HMAC-based one time password algorithm. Internet RFC 4226, December 2005
- [30] National Institute of Standards and Technology, The keyed-hash message authentication code (HMAC). FIPS PUB 198–1, July 2008

- [31] National Institute of Standards and Technology, Secure hash standard. FIPS PUB 180–2, August 2000
- [32] K. Pietrzak, A closer look at HMAC. Cryptology ePrint Archive. Report 2013/212 (2013)
- [33] B. Preneel, P. van Oorschot, On the security of iterated message authentication codes. *IEEE Trans. Inf. Theory* 45(1), 188–199 (1999). (Preliminary version, entitled "MD-x MAC and building fast MACs from hash functions," in CRYPTO 95.)
- [34] R. Rivest, The MD5 message-digest algorithm. Internet RFC 1321, April 1992
- [35] V. Shoup, Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive. Report 2004/332 (2004)
- [36] D. Stinson, Universal hashing and authentication codes. *Des. Codes Cryptogr.* 4, 369–380 (1994).
- [37] X. Wang, Y. L. Yin, H. Yu, Finding collisions in the full SHA-1, in V. Shoup ed., *Advances in Cryptology–CRYPTO'05*. LNCS, vol. 3621 (Springer, Berlin, Heidelberg, 2005)
- [38] X. Wang, H. Yu, How to break MD5 and other hash functions, in R. Cramer ed., *Advances in Cryptology–EU-ROCRYPT'05*. LNCS, vol. 3494 (Springer, Berlin, Heidelberg, 2005)
- [39] M. Wegman, L. Carter, New hash functions and their use in authentication and set equality. *J. Comput. Syst. Sci.* 22(3), 265–279 (1981)