A preliminary version of this paper appears in the proceedings of Eurocrypt 2016. This is the full version.

Hash-Function based PRFs: AMAC and its Multi-User Security

Mihir Bellare[1] Daniel J. Bernstein[2] Stefano Tessaro[3]

2016.02.28

Abstract

AMAC is a simple and fast candidate construction of a PRF from an MD-style hash function which applies the keyed hash function and then a cheap, un-keyed output transform such as truncation. Spurred by its use in the widely-deployed Ed25519 signature scheme, this paper investigates the provable PRF security of AMAC to deliver the following three-fold message: (1) First, we prove PRF security of AMAC. (2) Second, we show that AMAC has a quite unique and attractive feature, namely that its multi-user security is essentially as good as its single-user security and in particular superior in some settings to that of competitors. (3) Third, it is technically interesting, its security and analysis intrinsically linked to security of the compression function in the presence of leakage.

[1] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. URL: `http://cseweb.ucsd.edu/~mihir/`. Supported in part by NSF grants CNS-1526801 and CNS-1228890, ERC Project ERCC FP7/615074 and a gift from Microsoft. This work was done in part while the author was visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

[2] University of Illinois at Chicago, USA, and Technische Universiteit Eindhoven, Netherlands. URL: `https://cr.yp.to/djb.html`. Supported in part by NSF grant CNS-1314919 and NWO grant 639.073.005.

[3] Department of Computer Science, University of California Santa Barbara, Santa Barbara, California 93106, USA. URL: `http://www.cs.ucsb.edu/~tessaro/`. Supported in part by NSF grant CNS-1423566. This work was done in part while the author was visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

Contents

1 Introduction

This paper revisits a classical question, namely how can we turn a hash function into a PRF? The canonical answer is HMAC [3], which (1) first applies the keyed hash function to the message and then (2) re-applies, to the result, the hash function keyed with another key. We consider another, even simpler, candidate way, namely to change step (2) to apply a simple *un-keyed* output transform such as truncation. We call this AMAC, for augmented MAC. This paper investigates and establishes provable-security of AMAC, with good bounds, when the hash function is a classical MD-style one like SHA-512.

WHY? We were motivated to determine the security of AMAC by the following. *Usage.* AMAC with SHA-512 is used as a PRF in the Ed25519 signature scheme [7]. (AMAC under a key that is part of the signing key is applied to the hashed message to get coins for a Schnorr-like signature.) Ed25519 is widely deployed, including in SSH, Tor, OpenBSD and dozens of other places [9]. The security of AMAC for this usage was questioned in `cfrg` forum debates on Ed25519 as a proposed standard. Analysis of AMAC is important to assess security of this usage and allow informed choices. *Speed.* AMAC is faster than HMAC, particularly on short messages. See Appendix A. *Context.* Sponge-based PRFs, where truncation is the final step due to its already being so for the hash function, have been proven secure [17, 21, 1, 8, 10]. Our work can be seen as stepping back to ask if truncation works in a similar way for classical MD-style hash functions.

FINDINGS IN A NUTSHELL. Briefly, the message of this paper is the following: (1) First, we are able to prove PRF security of AMAC. (2) Second, AMAC has a quite unique and attractive feature, namely that its multi-user security is essentially as good as its single-user security and in particular superior in some settings to that of competitors. (3) Third, it is technically interesting, its security and analysis intrinsically linked to security of the compression function in the presence of leakage, so that leakage becomes of interest for reasons entirely divorced from side-channel attacks. We now step back to provide some background and discuss our approach and results.

THE BASIC CASCADE. Let $\mathsf{h}\colon \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ represent a compression function taking a c-bit chaining variable and b-bit message block to return a c-bit output. The *basic cascade* of h is the function $\mathsf{h}^*\colon \{0,1\}^c \times (\{0,1\}^b)^+ \to \{0,1\}^c$ defined by

```
Basic Cascade $\mathsf{h}^*(K, \mathbf{X})$
$Y \leftarrow K$ ; For $i = 1, \ldots, n$ do $Y \leftarrow \mathsf{h}(Y, \mathbf{X}[i])$ ; Return $Y$
```

where $\mathbf{X}$ is a vector over $\{0,1\}^b$ whose length is denoted n and whose i-th component is denoted $\mathbf{X}[i]$. This construct is the heart of MD-style hash functions [12, 22] like MD5, SHA-1, SHA-256 and SHA-512, which are obtained by setting K to a fixed, public value and then applying h^* to the padded message.

Now we want to key h^* to get PRFs. We regard h itself as a PRF on domain $\{0,1\}^b$, keyed by its c-bit chaining variable. Then h^* is the natural candidate for a PRF on the larger domain $(\{0,1\}^b)^+$. Problem is, h^* isn't secure as a PRF. This is due to the well-known *extension attack.* If I obtain $Y_1 = \mathsf{h}^*(K, X_1)$ for some $X_1 \in \{0,1\}^b$ of my choice, I can compute $Y_2 = \mathsf{h}^*(K, X_1X_2)$ for any $X_2 \in \{0,1\}^b$ of my choice *without knowing* K, via $Y_2 \leftarrow \mathsf{h}(Y_1, X_2)$. This clearly violates PRF security of h^*.

Although h^* is not a PRF, BCK2 [4] show that it is a prefix-free PRF. (A PRF as long as no input on which it is evaluated is a prefix of another. The two inputs X_1, X_1X_2 of the above attack violate this property.) When $b = 1$ and all inputs on which h^* is evaluated are of the same fixed length, the cascade h^* is the GGM construction of a PRF from a PRG [19].

To get a full-fledged PRF, NMAC applies h, under another key, to h^*. The augmented cascade $\mathsf{ACSC} = \mathsf{Out} \circ \mathsf{h}^*$ that we discuss next replaces NMAC's outer application of a keyed function with a simple un-keyed one.

Augmented cascade. The augmented cascade is parameterized by some (keyless) function $\mathsf{Out}\colon \{0,1\}^c \to \mathsf{Out.R}$ that we call the output transform, and is obtained by simply applying this function to the output of the basic cascade:

```
Augmented Cascade $(\mathsf{Out} \circ \mathsf{h}^*)(K, \mathbf{X})$
$Y \leftarrow \mathsf{h}^*(K, \mathbf{X})$ ; $Z \leftarrow \mathsf{Out}(Y)$ ; Return $Z$
```

AMAC is obtained from ACSC just as HMAC is obtained from NMAC, namely by putting the key in the input to the hash function rather than directly keying the cascade: $\mathsf{AMAC}(K, M) = \mathsf{Out}(H(K \| M))$. Just as NMAC is the technical core of HMAC, the augmented cascade is the technical core of AMAC, and our analysis will focus in it. We will be able to bridge to AMAC quite simply with the tools we develop.

The ACSC construction was suggested by cryptanalysts with the intuition that "good" choices of Out appear to allow $\mathsf{Out} \circ \mathsf{h}^*$ to evade the extension attack and thus possibly be a PRF. To understand this, first note that not all choices of Out are good. For example if Out is the identity function then the augmented cascade is the same as the basic one and the attack applies, or if Out is a constant function returning 0^r then $\mathsf{Out} \circ \mathsf{h}^*$ is obviously not a PRF over range $\{0,1\}^r$. Cryptanalysts have suggested some specific choices of Out, the most important being (1) truncation, where $\mathsf{Out}\colon \{0,1\}^c \to \{0,1\}^r$ returns, say, the first $r < c$ bits of its input, or (2) the mod function, as in Ed25519, where Out treats its input as an integer and returns the result modulo, say, a public r-bit prime number. Suppose r is sufficiently smaller than c (think $c = 512$ and $r = 256$). An adversary querying X_1 in the PRF game no longer gets back $Y_1 = \mathsf{h}^*(K, X_1)$ but rather $Z_1 = \mathsf{Out}(Y_1)$, and this does not allow the extension attack to proceed. On this basis, and for the choices of Out just named, the augmented cascade is already seeing extensive usage and is suggested for further usage and standardization.

This raises several questions. First, that $\mathsf{Out} \circ \mathsf{h}^*$ seems to evade the extension attack does not mean it is a PRF. There may be other attacks. The goal is to get a PRF, not to evade some specific attacks. Moreover we would like a proof that this goal is reached. Second, for which choices of Out does the construction work? We could try to analyze the PRF security of $\mathsf{Out} \circ \mathsf{h}^*$ in an ad hoc way for the specific choices of Out named above, but it would be more illuminating and useful to be able to establish security in a broad way, for all Out satisfying some conditions. These are the questions our work considers and resolves.

Connection to leakage. If we want to prove PRF security of $\mathsf{Out} \circ \mathsf{h}^*$, a basic question to ask is, under what assumption on the compression function h? The natural one is that h is itself a PRF, the same assumption as for the proof of NMAC [2, 16]. We observe that this is not enough. Consider an adversary who queries the one-block message X_1 to get back $Z_1 = \mathsf{Out}(Y_1)$ and then queries the two-block message X_1X_2 to get back $Z_2 = \mathsf{Out}(Y_2)$ where by definition $Y_1 = \mathsf{h}^*(K, X_1) = \mathsf{h}(K, X_1)$ and $Y_2 = \mathsf{h}^*(K, X_1X_2) = \mathsf{h}(Y_1, X_2)$. Note that Y_1 is being used as a key in applying h to X_2. But this key is not entirely unknown to the adversary because the latter knows $Z_1 = \mathsf{Out}(Y_1)$. If the application of h with key Y_1 is to provide security, it must be in the face of the fact that some information about this key, namely $\mathsf{Out}(Y_1)$, has been "leaked" to the adversary. As a PRF, h must thus be resilient to some leakage on its key, namely that represented by Out viewed as a leakage function.

Approach and qualitative results. We first discuss our results at the qualitative level and then later at the (in our view, even more interesting) quantitative level. Theorems 5.2 and 5.3

show that if h is a PRF under Out-leakage then $\mathsf{Out} \circ \mathsf{h}^*$ is indistinguishable from the result of applying Out to a random function. (The compression function h being a PRF under Out-leakage means it retains PRF security under key K even if the adversary is given $\mathsf{Out}(K)$. The formal definition is in Section 4.) This result makes no assumptions about Out beyond that implicit in the assumption on h, meaning the result is true for *all* Out, and is in the standard model. As a corollary we establish PRF security of $\mathsf{Out} \circ \mathsf{h}^*$ for a large class of output functions Out, namely those that are close to regular. (This means that the distribution of $\mathsf{Out}(Y)$ for random Y is close to the uniform distribution on the range of Out.) In summary we have succeeded in providing conditions on Out, h under which $\mathsf{Out} \circ \mathsf{h}^*$ is proven to be PRF. Our conditions are effectively both necessary and sufficient and cover cases proposed for usage and standardization.

The above is a security proof for the augmented cascade $\mathsf{Out} \circ \mathsf{h}^*$ under the assumption that the compression function h is resistant to Out leakage. To assess the validity of this assumption, we analyze the security under leakage of an ideal compression function. Theorem 7.2 shows that an ideal compression function is resistant to Out-leakage as long as no range point of Out has too few pre-images. This property is in particular true if Out is close to regular. As a result, in the ideal model, we have a validation of our Out-leakage resilience assumption. Putting this together with the above we have a proof-based validation of the augmented cascade.

Multi-user security. The standard definition of PRF security of a function family F [19] is single user (su), represented by there being a single key K such that the adversary has access to an oracle Fn that given x returns either $\mathsf{F}(K, x)$ or the result of a random function F on x. But in "real life" there are many users, each with their own key. If we look across the different entities and Internet connections active at any time, the number of users / keys is very large. The more appropriate model is thus a multi-user (mu) one, where, for a parameter u representing the number of users, there are u keys $K_1, \ldots, K_u$. Oracle Fn now takes i, x with $1 \leq i \leq u$ and returns either $\mathsf{F}(K_i, x)$ or the result of a random function F_i on x. It is in this setting that we should address security.

Multi-user security is typically neglected because it makes no *qualitative* difference: BCK2 [4], who first formalized the notion, also showed by a hybrid argument that the advantage of an adversary relative to u users is not more than u times the advantage of an adversary of comparable resources relative to a single user. Our Lemma 4.1 is a generalization of this result. But this degradation in advantage is quite significant in practice, since u is large, and raises the important question of whether one can do quantitatively better. Clearly one cannot in general, but perhaps one can for specific, special function families F. If so, these function families are preferable in practice. This perspective is reflected in recent work like [23, 26].

These special function families seem quite rare. But we show that the augmented cascade is one of them. In fact we show that mu security gives us a double benefit in this setting, one part coming from the cascade itself and the other from the security of the compression function under leakage, the end result being very good bounds for the mu security of the augmented cascade.

Theorem 5.2 establishes su security of the augmented cascade based not on the su, but on the mu security of the compression function under Out-leakage. The bound is very good, the advantage dropping only by a factor equal to the maximum length of a query. The interesting result is Theorem 5.3, establishing mu security of the augmented cascade under the same assumptions and with essentially the same bounds as Theorem 5.2 establishing its su security. In particular we do not lose a factor of the number of users u in the advantage. This is the first advance.

Now note that the assumption in both of the above-mentioned results is the mu (not su) security of the compression function under Out-leakage. Our final bound will thus depend on this. The second advance is that Theorem 7.2 shows mu security of the compression function under Out-

leakage with bounds almost as good as for su security. This represents an interesting result of independent interest, namely that, under leakage, the mu security of an ideal compression function is almost as good as its su security. This is not true in the absence of leakage. The results are summarized via Fig. 4.

QUANTITATIVE RESULTS. We obtain good quantitative bounds on the mu prf security of the augmented cascade in the ideal compression function model by combining our aforementioned results on the mu prf security under leakage of an ideal compression function with our also aforementioned reduction of the security of the cascade to the security of the compression function under leakage. We illustrate these results for the case where the compression function is of form $\mathsf{h}\colon \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ and the output transform Out simply outputs the first r bits of its c-bit input, for $r \le c$. We consider an attacker making at most q queries to a challenge oracle (that is either the augmented cascade or a random function), each query consisting of at most ℓ b-bit blocks, and $q_{\rm F}$ queries to the ideal compression function oracle. We show that such an attacker achieves distinguishing advantage at most

$$\frac{\ell^2 q^2 + \ell q q_{\rm F}}{2^c} + \frac{cr \cdot (\ell^2 q + \ell q_{\rm F})}{2^{c-r}}\ , \qquad (1)$$

where we have intentionally omitted constant factors and lower order terms. Note that this bound holds *regardless of the number of users u*. Here c is large, like $c = 512$, so the first term is small. But $c - r$ is smaller, for example $c - r = 256$ with $r = 256$. The crucial merit of the bound of Equation (1) is that the numerator in the second term does not contain quadratic terms like q^2 or $q \cdot q_{\rm F}$. In practice, $q_{\rm F}$ and q are the terms we should allow to be large, so this is significant. To illustrate, say for example $\ell = 2^{10}$ (meaning messages are about 128 KBytes if $b = 1024$) and $q_{\rm F} = 2^{100}$ and $q = 2^{90}$. The bound from Equation (1) is about 2^{-128}, which is very good. But, had the second term been of the form $\ell^2(q_{\rm F}^2 + q^2)/2^{c-r}$ then the bound would be only 2^{-36}. See Section 8 for more information.

2-TIER CASCADE. We introduce and use an extension of the basic cascade h^*. Our 2-tier cascade is associated to two function families g, h. Under key K, it applies $\mathsf{g}(K, \cdot)$ to the first message block to get a sub-key K^* and the applies $\mathsf{h}^*(K^*, \cdot)$ to the rest of the message. The corresponding augmented cascade applies Out to the result. Our results about the augmented cascade above are in fact shown for the augmented 2-tier cascade. This generalization has both conceptual and analytical value. We briefly mention two instances. (1) First, we can visualize mu security of $\mathsf{Out} \circ \mathsf{h}^*$ as pre-pending the user identity to the message and then applying the 2-tier cascade with first tier a random function. This effectively reduces mu security to su security. With this strategy we prove Theorem 5.3 as a corollary of Theorem 5.2 and avoid a direct analysis of mu security. Beyond providing a modular proof this gives some insight into why the mu security is almost as good as the su security. (2) Second, just as NMAC is the technical core and HMAC the function used (because the latter makes blackbox use of the hash function), in our case the augmented cascade is the technical core but what will be used is AMAC, defined by $\mathsf{AMAC}(K, M) = \mathsf{Out}(H(K, M))$ where H is the hash function derived from compression function $\mathsf{h}\colon \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ and K is a k-bit key. For the analysis we note (assuming $k = b$) that this is simply an augmented 2-tier cascade with the first tier being the dual of h, meaning the key and input roles are swapped. We thus directly get an analysis and proof for this case from our above-mentioned results. Obtaining HMAC from NMAC was more work [3, 2] and required assumptions about PRF security of the dual function under related keys.

DAVIES-MEYER. Above we have assessed the PRF security under Out-leakage of the compression function by modeling the latter as ideal (random). But, following CDMP [11], one might say that the compression functions underlying MD-style hash functions are not un-structured enough to be treated as random because they are built from blockciphers via the Davies-Meyer (DM) construction. To address this we analyze the mu PRF security under Out-leakage of the DM construction in the ideal-cipher model. One's first thought may be that such an analysis would follow from our analysis for a random compression function and the indifferentiability [20, 11] of DM from a random oracle, but the catch is that DM is *not* indifferentiable from a RO so a direct analysis is needed. The one we give in Section 10 shows mu security with good bounds. Similar analyses can be given for other PGV [25] compression functions.

2 Related work

Sponges. SHA-3 already internally incorporates a truncation output transform. The construction itself is a sponge. The suggested way to obtain a PRF is to simply key the hash function via its IV, so that the PRF is a keyed, truncated sponge. The security of this construct has been intensively analyzed [17, 21, 1, 8, 10] with Gaži, Pietrzak and Tessaro (GPT) [17] establishing PRF security with tight bounds. Our work can be seen as stepping back to ask whether the same truncation method would work for MD-style hash functions like SHA-512. Right now these older hash functions are much more widely deployed than SHA-3, and current standardization and deployment efforts continue to use them, making the analysis of constructions based on them important with regard to security in practice. The underlying construction in this case is the cascade, which is quite different from the sponge. The results and techniques of GPT [17] do not directly apply but were an important inspiration for our work.

We note that keyed sponges with truncation to an r-bit output from a c-bit state can easily be distinguished from a random function with advantage roughly $q^2/2^{c-r}$ or $qq_{\mathrm{F}}/2^{c-r}$, as shown for example in [17]. The bound of Equation (1) is better, meaning the augmented cascade offers greater security. See Section 9 for more information.

Cascade. BCK2 [4] show su security of the basic cascade (for prefix-free queries) in two steps. First, they show su security of the basic cascade (for prefix-free queries) assuming not su, but mu security of the compression function. Second, they apply the trivial bound mentioned above to conclude su security of the basic cascade for prefix-free queries assuming su security of the compression function. We follow their approach to establish su security of the augmented cascade, but there are differences as well: They have no output transform while we do, they assume prefix-free queries and we do not, we have leakage and they do not. They neither target nor show mu security of the basic cascade in any form, mu security arising in their work only as an intermediate technical step and only for the compression function, not for the cascade.

Chop-MD. The chop-MD construction of CDMP [11] is the case of the augmented cascade in which the output transform is truncation. They claim this is indifferentiable from a RO when the compression function is ideal. This implies PRF security but their bound is $O(\ell^2(q+q_{\mathrm{F}})^2/2^{c-r})$ which as we have seen is significantly weaker than our bound of Equation (1). Also, they have no standard-model proofs or analysis for this construction. In contrast our results in Section 5 establish standard-model security.

NMAC and HMAC. NMAC takes keys $K_{\mathrm{in}}, K_{\mathrm{out}}$ and input $\mathbf{X}$ to return $\mathsf{h}(K_{\mathrm{out}}, \mathsf{h}^*(K_{\mathrm{in}}, \mathbf{X}) \| \mathrm{pad})$ where pad is some $(b-c)$-bit constant and $b \geq c$. Through a series of intensive analyses, the PRF security of NMAC has been established based only on the assumed PRF security of the compression function h, and with tight bounds [3, 2, 16]. Note that NMAC is not a special case of the augmented cascade because Out is not keyed but the outer application of h in NMAC is keyed.

In the model where the compression function is ideal, one can show bounds for NMAC that are somewhat better than for the augmented cascade. This is not surprising. Indeed, when attacking the augmented cascade, the adversary can learn far more information about the internal states of the hash computation. What is surprising (at least to us) is that the gap is actually quite small. See Section 9 for more information. We stress also that this is in the ideal model. In the standard model, there is no proof that NMAC has the type of good mu prf security we establish for the augmented cascade in Section 5.

AES and other MACs. Why consider new MACs? Why not just use an AES-based MAC like CMAC? The 128 bit key and block size limits security compared to $c = 512$ for SHA-512. A Schnorr signature takes the result of the PRF modulo a prime; the PRF output must have at least as many bits as the prime, and even more bits for most primes, to avoid the Bleichenbacher attack discussed in [24]. Also in that context a hash function is already being used to hash the message before signing so it is convenient to implement the PRF also with the same hash function. HMAC-SHA-512 will provide the desired security but AMAC has speed advantages, particularly on short messages, as discussed in Appendix A, and is simpler. Finally, the question is in some sense moot since AMAC is already deployed and in widespread use via Ed25519 and we need to understand its security.

Leakage. Leakage-resilience of a PRF studies the PRF security of a function h when the attacker can obtain the result of an *arbitrary* function, called the leakage function, applied to the key [15, 13]. This is motivated by side-channel attacks. We are considering a much more restricted form of leakage where there is just one, very specific leakage function, namely Out. This arises naturally, as we have seen, in the PRF security of the augmented cascade. We are not considering side-channel attacks.

3 Notation

If $\mathbf{x}$ is a vector then $|\mathbf{x}|$ denotes its length and $\mathbf{x}[i]$ denotes its i-th coordinate. (For example if $\mathbf{x} = (10, 00, 1)$ then $|\mathbf{x}| = 3$ and $\mathbf{x}[2] = 00$.) We let ε denote the empty vector, which has length 0. If $0 \leq i \leq |\mathbf{x}|$ then we let $\mathbf{x}[1..i] = (\mathbf{x}[1], \ldots, \mathbf{x}[i])$, this being ε when $i = 0$. We let S^n denote the set of all length n vectors over the set S. We let S^+ denote the set of all vectors of positive length over the set S and $S^* = S^+ \cup \{\varepsilon\}$ the set of all finite-length vectors over the set S. As special cases, $\{0,1\}^n$ and $\{0,1\}^*$ denote vectors whose entries are bits, so that we are identifying strings with binary vectors and the empty string with the empty vector.

For sets A_1, A_2 we let $[\![A_1, A_2]\!]$ denote the set of all vectors $\mathbf{X}$ of length $|\mathbf{X}| \geq 1$ such that $\mathbf{X}[1] \in A_1$ and $\mathbf{X}[i] \in A_2$ for $2 \leq i \leq |\mathbf{X}|$.

We let $x \leftarrow_{\$} X$ denote picking an element uniformly at random from a set X and assigning it to x. For infinite sets, it is assumed that a proper measure can be defined on X to make this meaningful. Algorithms may be randomized unless otherwise indicated. Running time is worst case. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots; r)$ denote running A with random coins r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \leftarrow_{\$} A(x_1, \ldots)$ be the result of picking r at random and letting $y \leftarrow A(x_1, \ldots; r)$. We let $[A(x_1, \ldots)]$ denote the set of all possible outputs of A when invoked with inputs $x_1, \ldots$.

We use the code based game playing framework of [5]. (See Fig. 1 for an example.) By $\Pr[\mathrm{G}]$ we denote the probability that game G returns true.

For an integer n we let $[1..n] = \{1, \ldots, n\}$.

```
Game $\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1}(\mathcal{A})$
$v \leftarrow 0$
$c \leftarrow_{\$} \{0,1\}$ ; $c' \leftarrow_{\$} \mathcal{A}^{\textsc{New},\textsc{Fn}}$
Return $(c = c')$

$\textsc{New}()$
$v \leftarrow v+1$ ; $F_v \leftarrow_{\$} \mathsf{F}_c$

$\textsc{Fn}(i,x)$
Return $F_i(x)$
```

```
Game $\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A})$
$v \leftarrow 0$
$c \leftarrow_{\$} \{0,1\}$ ; $c' \leftarrow_{\$} \mathcal{A}^{\textsc{New},\textsc{Fn}}$
Return $(c = c')$

$\textsc{New}()$
$v \leftarrow v+1$ ; $K_v \leftarrow_{\$} \mathsf{F}_1.\mathsf{K}$
If $(c = 1)$ then $F_v \leftarrow \mathsf{F}_1(K_v, \cdot)$ else $F_v \leftarrow_{\$} \mathsf{F}_0$
Return $\mathsf{Out}(K_v)$

$\textsc{Fn}(i,x)$
Return $F_i(x)$
```

Figure 1: **Games defining distance metric between function families $\mathsf{F}_0, \mathsf{F}_1$.** In the basic (left) case there is no leakage, while in the extended (right) case there is leakage represented by Out.

4 Function-family distance framework

We will be considering various generalizations and extensions of standard prf security. This includes measuring proximity not just to random functions but to some other family, multi-user security and leakage on the key. We also want to allow an easy later extension to a setting with ideal primitives. To enable all this in a unified way we introduce a general distance metric on function families and then derive notions of interest as special cases.

FUNCTION FAMILIES. A *function family* is a two-argument function $\mathsf{F}\colon \mathsf{F.K} \times \mathsf{F.D} \to \mathsf{F.R}$ that takes a key K in the key space $\mathsf{F.K}$ and an input x in the domain $\mathsf{F.D}$ to return an output $y \leftarrow \mathsf{F}(K,x)$ in the range $\mathsf{F.R}$. We let $f \leftarrow_{\$} \mathsf{F}$ be shorthand for $K \leftarrow_{\$} \mathsf{F.K}$; $f \leftarrow \mathsf{F}(K,\cdot)$, the operation of picking a function at random from family F.

An example of a function family that is important for us is the compression function underlying a hash function, in which case $\mathsf{F.K} = \mathsf{F.R} = \{0,1\}^c$ and $\mathsf{F.D} = \{0,1\}^b$ for integers c, b called the length of the chaining variable and the block length, respectively. Another example is a block cipher. However, families of functions do not have to be efficiently computable or have short keys. For sets D, R the *family* $\mathsf{A}\colon \mathsf{A.K} \times D \to R$ *of all functions from D to R* is defined simply as follows: let $\mathsf{A.K}$ be the set of all functions mapping D to R and let $\mathsf{A}(f,x) = f(x)$. (We can fix some representation of f as a key, for example the vector whose i-th component is the value f takes on the i-th input under some ordering of D. But this is not really necessary.) In this case $f \leftarrow_{\$} \mathsf{A}$ denotes picking at random a function mapping D to R.

Let $\mathsf{F}\colon \mathsf{F.K} \times \mathsf{F.D} \to \mathsf{F.R}$ be a function family and let $\mathsf{Out}\colon \mathsf{F.R} \to \mathsf{Out.R}$ be a function with domain the range of F and range $\mathsf{Out.R}$. Then the composition $\mathsf{Out} \circ \mathsf{F}\colon \mathsf{F.K} \times \mathsf{F.D} \to \mathsf{Out.R}$ is the function family defined by $(\mathsf{Out} \circ \mathsf{F})(K,x) = \mathsf{Out}(\mathsf{F}(K,x))$. We will use composition in some of our constructions.

BASIC DISTANCE METRIC. We define a general metric of distance between function families that will allow us to obtain other metrics of interest as special cases. Let $\mathsf{F}_0, \mathsf{F}_1$ be families of functions such that $\mathsf{F}_0.\mathsf{D} = \mathsf{F}_1.\mathsf{D}$. Consider game DIST on the left of Fig. 1 associated to $\mathsf{F}_0, \mathsf{F}_1$ and an adversary $\mathcal{A}$. Via oracle $\textsc{New}$, the adversary can create a new instance F_v drawn from F_c where c is the challenge bit. It can call this oracle multiple times, reflecting a multi-user setting. It can obtain $F_i(x)$ for any i, x of its choice with the restriction that $1 \leq i \leq v$ (instance i has been initialized) and $x \in \mathsf{F}_1.\mathsf{D}$. It wins if it guesses the challenge bit c. The advantage of adversary $\mathcal{A}$ is

$$\begin{aligned}\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F}_0,\mathsf{F}_1}(\mathcal{A}) &= 2\Pr[\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1}(\mathcal{A})] - 1 &(2)\\ &= \Pr[\,\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1}(\mathcal{A}) \,|\, c = 1\,] - (1 - \Pr[\,\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1}(\mathcal{A}) \,|\, c = 0\,]) \ . &(3)\end{aligned}$$

Equation (2) is the definition, while Equation (3) is a standard alternative formulation that can be shown equal via a conditioning argument. We often use the second in proofs.

Let F be a function family and let A be the family of all functions from $\mathsf{F.D}$ to $\mathsf{F.R}$. Let $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F}}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F},\mathsf{A}}(\mathcal{A})$. This gives a metric of multi-user prf security. The standard (single user) prf metric is obtained by restricting attention to adversaries that make exactly one New query.

Distance under leakage. We extend the framework to allow leakage on the key. Let $\mathsf{Out}\colon \mathsf{F}_1.\mathsf{K} \to \mathsf{Out.R}$ be a function with domain $\mathsf{F}_1.\mathsf{K}$ and range a set we denote $\mathsf{Out.R}$. Consider game DIST on the right of Fig. 1, now associated not only to $\mathsf{F}_0, \mathsf{F}_1$ and an adversary $\mathcal{A}$ but also to Out. Oracle New picks a key K_v for F_1 and will return as leakage the result of Out on this key. The instance F_v is either $\mathsf{F}_1(K_v, \cdot)$ or a random function from F_0. Note that the leakage is on a key for a function from F_1 regardless of the challenge bit, meaning even if $c = 0$, we leak on the key K_v drawn from $\mathsf{F}_1.\mathsf{K}$. The second oracle is as before. The advantage of adversary $\mathcal{A}$ is

$$\begin{aligned}\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A}) &= 2\Pr[\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A})] - 1 &(4)\\ &= \Pr[\,\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A}) \,|\, c = 1\,] - (1 - \Pr[\,\mathrm{DIST}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A}) \,|\, c = 0\,]) \ . &(5)\end{aligned}$$

This generalizes the basic metric because $\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F}_0,\mathsf{F}_1}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A})$ where Out is the function that returns ε on all inputs.

As a special case we get a metric of multi-user prf security under leakage. Let F be a function family and let A be the family of all functions from $\mathsf{F.D}$ to $\mathsf{F.R}$. Let $\mathsf{Out}\colon \mathsf{F.K} \to \mathsf{Out.R}$. Let $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},\mathsf{Out}}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F},\mathsf{A},\mathsf{Out}}(\mathcal{A})$.

Naive mu to su reduction. Multi-user security for PRFs was first explicitly considered in [4]. They used a hybrid argument to show that the prf advantage of an adversary $\mathcal{A}$ against u users is at most u times the prf advantage of an adversary of comparable resources against a single user. The argument extends to the case where instead of prf advantage we consider distance and where leakage is present. This is summarized in Lemma 4.1 below.

We state this lemma to emphasize that mu security is not qualitatively different from su security, at least in this setting. The question is what is the quantitative difference. The lemma represents the naive bound, which always holds. The interesting element is that for the 2-tier augmented cascade, Theorem 5.3 shows that one can do better: the mu advantage is not a factor u less than the single-user advantage, but about the same. In the proof of the lemma below we specify the adversary for the sake of making the reduction concrete but we omit the standard hybrid argument that establishes that this works.

Lemma 4.1 *Let $\mathsf{F}_0, \mathsf{F}_1$ be function families with $\mathsf{F}_0.\mathsf{D} = \mathsf{F}_1.\mathsf{D}$ and let $\mathsf{Out}\colon \mathsf{F}_1.\mathsf{K} \to \mathsf{Out.R}$ be an output transform. Let $\mathcal{A}$ be an adversary making at most u queries to its* New *oracle and at most q queries to its* Fn *oracle. The proof specifies an adversary $\mathcal{A}_1$ making one query to its* New *oracle and at most q queries to its* Fn *oracle such that*

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A}) \leq u \cdot \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{F}_0,\mathsf{F}_1,\mathsf{Out}}(\mathcal{A}_1) \ . \tag{6}$$

The running time of $\mathcal{A}_1$ is that of $\mathcal{A}$ plus the time for u computations of F_0 or F_1.

Proof of Lemma 4.1: Adversary $\mathcal{A}_1$ runs $\mathcal{A}$, simulating the latter's New, Fn oracles via subroutines $\text{New}^*, \text{Fn}^*$, as follows:

```
Adversary $\mathcal{A}_1^{\text{New},\text{Fn}}$
$v \leftarrow 0$
$g \leftarrow\!\!\$\ [1..u]$
$c' \leftarrow\!\!\$\ \mathcal{A}^{\text{New}^*,\text{Fn}^*}$
Return $c'$

$\text{New}^*()$
$v \leftarrow v+1$ ; $K_v \leftarrow\!\!\$\ \mathsf{F}_1.\mathsf{K}$
$L \leftarrow \mathsf{Out}(K_v)$
If $(v < g)$ then $F_v \leftarrow \mathsf{F}_1(K_v, \cdot)$
If $(v = g)$ then $L \leftarrow \text{New}()$
If $(v > g)$ then $F_v \leftarrow\!\!\$\ \mathsf{F}_0$
Return $L$

$\text{Fn}^*(i, x)$
If $(v = g)$ then
  $y \leftarrow \text{Fn}(1, x)$
Else $y \leftarrow F_i(x)$
Return $y$
```

We omit the standard analysis establishing Equation (6). ▮

5 The augmented cascade and its analysis

We first present a generalization of the basic cascade construction that we call the 2-tier cascade. We then present the augmented (2-tier) cascade construction and analyze its security.

2-TIER CASCADE CONSTRUCTION. Let $\mathcal{K}$ be a set. Let g, h be function families such that $\mathsf{g}\colon \mathsf{g.K} \times \mathsf{g.D} \to \mathcal{K}$ and $\mathsf{h}\colon \mathcal{K} \times \mathsf{h.D} \to \mathcal{K}$. Thus, outputs of both g and h can be used as keys for h. This is the basis of our 2-tier version of the cascade. This is a function family $\mathbf{CSC}[\mathsf{g},\mathsf{h}]\colon \mathsf{g.K} \times [\![\mathsf{g.D}, \mathsf{h.D}]\!] \to \mathcal{K}$. That is, a key is one for g. An input —as per the notation $[\![\cdot,\cdot]\!]$ defined in Section 3— is a vector $\mathbf{X}$ of length at least one whose first component is in $\mathsf{g.D}$ and the rest of whose components are in $\mathsf{h.D}$. Outputs are in $\mathcal{K}$. The function itself is defined as follows:

```
Function $\mathbf{CSC}[\mathsf{g},\mathsf{h}](K, \mathbf{X})$
$n \leftarrow |\mathbf{X}|$ ; $Y \leftarrow \mathsf{g}(K, \mathbf{X}[1])$
For $j = 2, \ldots, n$ do $Y \leftarrow \mathsf{h}(Y, \mathbf{X}[j])$
Return $Y$
```

We say that a function family G is a 2-tier cascade if $\mathsf{G} = \mathbf{CSC}[\mathsf{g},\mathsf{h}]$ for some g, h. If $\mathsf{f}\colon \mathcal{K} \times \mathsf{f.D} \to \mathcal{K}$ then its basic cascade is recovered as $\mathbf{CSC}[\mathsf{f},\mathsf{f}]\colon \mathcal{K} \times \mathsf{f.D}^+ \to \mathcal{K}$. We will also denote this function family by f^*.

Recall that even if $\mathsf{f}\colon \{0,1\}^c \times \{0,1\}^b \to \{0,1\}^c$ is a PRF, f^* is not a PRF due to the extension attack. It is shown by BCK2 [4] to be a PRF when the adversary is restricted to prefix-free queries. When $b = 1$ and the adversary is restricted to queries of some fixed length ℓ, the cascade f^* is the GGM construction of a PRF from a PRG [19]. Bernstein [6] considers a generalization of the basic cascade in which the function applied depends on the block index and proves PRF security for any fixed number ℓ of blocks.

Our generalization to the 2-tier cascade has two motivations and corresponding payoffs. First, it will allow us to reduce mu security to su security in a simple, modular and tight way, the idea being that mu security of the basic cascade is su security of the 2-tier one for a certain choice of the 1st tier family. Second, it will allow us to analyze the blackbox AMAC construction in which the cascade is not keyed directly but rather the key is put in the input to the hash function.

THE AUGMENTED CASCADE. With $\mathcal{K}, \mathsf{g}, \mathsf{h}$ as above let $\mathsf{Out}\colon \mathcal{K} \to \mathsf{Out.R}$ be a function we call the output transform. The augmented (2-tier) cascade $\mathbf{ACSC}[\mathsf{g},\mathsf{h},\mathsf{Out}]\colon \mathsf{g.K} \times [\![\mathsf{g.D}, \mathsf{h.D}]\!] \to \mathsf{Out.R}$ is the composition of Out with $\mathbf{CSC}[\mathsf{g},\mathsf{h}]$, namely $\mathbf{ACSC}[\mathsf{g},\mathsf{h},\mathsf{Out}] = \mathsf{Out} \circ \mathbf{CSC}[\mathsf{g},\mathsf{h}]$, where composition was defined above. In code:

```
Function $\mathbf{ACSC}[\mathsf{g},\mathsf{h},\mathsf{Out}](K,\mathbf{X})$
$Y \leftarrow \mathbf{CSC}[\mathsf{g},\mathsf{h}](K,\mathbf{X})$ ; $Z \leftarrow \mathsf{Out}(Y)$
Return $Z$
```

We say that a function family G^+ is an augmented (2-tier) cascade if $\mathsf{G}^+ = \mathbf{ACSC}[\mathsf{g},\mathsf{h},\mathsf{Out}]$ for some $\mathsf{g},\mathsf{h},\mathsf{Out}$.

The natural goal is that an augmented cascade G^+ be a PRF. This however is clearly not true for all Out. For example Out may be a constant function, or a highly irregular one. Rather than restrict Out at this point we target a general result that would hold for any Out. Namely we aim to show that $\mathbf{ACSC}[\mathsf{g},\mathsf{h},\mathsf{Out}]$ is close under our distance metric to the result of applying Out to a random function. Next we formalize and prove this.

Single-user security of 2-tier augmented cascade. Given $\mathsf{g},\mathsf{h},\mathsf{Out}$ defining the 2-tier augmented cascade $\mathsf{Out} \circ \mathbf{CSC}[\mathsf{g},\mathsf{h}]$, we want to upper bound $\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A})$ for an adversary $\mathcal{A}$ making one New query, where A is the family of all functions with the same domain as $\mathbf{CSC}[\mathsf{g},\mathsf{h}]$. We will do this in two steps. First, in Lemma 5.1, we will consider the case that the first tier is a random function, meaning $\mathsf{g} = \mathsf{r}$ is the family of all functions with the same domain and range as g. Then, in Theorem 5.2, we will use Lemma 5.1 to analyze the general case where g is a PRF. Most interestingly we will later use these single-user results to easily obtain, in Theorem 5.3, bounds for multi-user security that are essentially as good as for single-user security. This showcases a feature of the 2-tier cascade that is rare amongst PRFs. We now proceed to the above-mentioned lemma.

Lemma 5.1 *Let $\mathcal{K},\mathcal{D}$ be non-empty sets. Let $\mathsf{h}\colon \mathcal{K}\times\mathsf{h.D}\to\mathcal{K}$ be a function family. Let r be the family of all functions with domain $\mathcal{D}$ and range $\mathcal{K}$. Let $\mathsf{Out}\colon \mathcal{K}\to\mathsf{Out.R}$ be an output transform. Let A be the family of all functions with domain $[\![\mathcal{D},\mathsf{h.D}]\!]$ and range $\mathcal{K}$. Let $\mathcal{A}$ be an adversary making exactly one query to its New oracle followed by at most q queries to its Fn oracle, the second argument of each of the queries in the latter case being a vector $\mathbf{X}\in[\![\mathcal{D},\mathsf{h.D}]\!]$ with $2\le|\mathbf{X}|\le\ell+1$. Let adversary $\mathcal{A}_{\mathsf{h}}$ be as in Fig. 2. Then*

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{r},\mathsf{h}]}(\mathcal{A}) \le \ell\cdot\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_{\mathsf{h}}) \ . \tag{7}$$

Adversary $\mathcal{A}_{\mathsf{h}}$ makes at most q queries to its New oracle and at most q queries to its Fn oracle. Its running time is that of $\mathcal{A}$ plus the time for $q\ell$ computations of h.

With the first tier being a random function, Lemma 5.1 is bounding the single-user ($\mathcal{A}$ makes one New query) distance of the augmented 2-tier cascade to the result of applying Out to a random function under our distance metric. The bound of Equation (7) is in terms of the multi-user security of h as a PRF and grows linearly with one less than the maximum number of blocks in a query.

We note that we could apply Lemma 4.1 to obtain a bound in terms of the single-user PRF security of h, but this is not productive. Instead we will go the other way, later bounding the multi-user security of the 2-tier augmented cascade in terms of the multi-user PRF security of its component functions.

The proof below follows the basic paradigm of the proof of BCK2 [4], which is itself an extension of the classic proof of GGM [19]. However there are several differences: (1) The cascade in BCK2 is single-tier and non-augmented, meaning both the r component and Out are missing (2) BCK2 assume the adversary queries are prefix-free, meaning no query is a prefix of another, an assumption we do not make (3) BCK2 bounds prf security, while we bound the distance.

Proof of Lemma 5.1: Consider the hybrid games and adversaries in Fig. 2. The following chain

```
Game $H_s$ $(0 \le s \le \ell)$
$b' \leftarrow\$ \mathcal{A}^{\text{NEW}^*,\text{FN}^*}$
Return $(b' = 1)$

$\text{NEW}^*()$
$f \leftarrow\$ \mathsf{A}$

$\text{FN}^*(i, \mathbf{X})$
$n \leftarrow |\mathbf{X}|$
If $(n \le s)$ then $Y \leftarrow f(\mathbf{X})$
Else
    $Y \leftarrow f(\mathbf{X}[1..s+1])$
    For $j = s+2, \ldots, n$ do $Y \leftarrow \mathsf{h}(Y, \mathbf{X}[j])$
$T_1[\mathbf{X}] \leftarrow Y$ ; $T_2[\mathbf{X}] \leftarrow \mathsf{Out}(T_1[\mathbf{X}])$
Return $T_2[\mathbf{X}]$
```

```
Adversary $\mathcal{A}_{\mathsf{h}}^{\text{NEW},\text{FN}}$
$g \leftarrow\$ \{1, \ldots, \ell\}$ ; $b' \leftarrow\$ \mathcal{A}_g^{\text{NEW},\text{FN}}$
Return $b'$
```

```
Adversary $\mathcal{A}_g^{\text{NEW},\text{FN}}$ $(1 \le g \le \ell)$
$v \leftarrow 0$ ; $b' \leftarrow\$ \mathcal{A}^{\text{NEW}^*,\text{FN}^*}$ ; Return $b'$

$\text{NEW}^*()$

$\text{FN}^*(i, \mathbf{X})$
$n \leftarrow |\mathbf{X}|$
If $(n \le g-1)$ then
    If (not $T_1[\mathbf{X}]$) then
        $T_1[\mathbf{X}] \leftarrow\$ \mathcal{K}$ ; $T_2[\mathbf{X}] \leftarrow \mathsf{Out}(T_1[\mathbf{X}])$
If $(n \ge g)$ then
    If (not $U[\mathbf{X}[1..g]]$) then
        $v \leftarrow v+1$ ; $U[\mathbf{X}[1..g]] \leftarrow v$
        $T_2[\mathbf{X}[1..g]] \leftarrow \text{NEW}()$
If $(n \ge g+1)$ then
    $T_1[\mathbf{X}[1..g+1]] \leftarrow \text{FN}(U[\mathbf{X}[1..g]], \mathbf{X}[g+1])$
    For $j = g+2, \ldots, n$ do
        $T_1[\mathbf{X}[1..j]] \leftarrow \mathsf{h}(T_1[\mathbf{X}[1..j-1]], \mathbf{X}[j])$
    $T_2[\mathbf{X}] \leftarrow \mathsf{Out}(T_1[\mathbf{X}])$
Return $T_2[\mathbf{X}]$
```

Figure 2: **Games and adversaries for proof of Theorem 5.1.**

of equalities establishes Equation (7) and will be justified below:

$$\ell \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_{\mathsf{h}}) = \textstyle\sum_{g=1}^{\ell} \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_g) \tag{8}$$

$$= \textstyle\sum_{g=1}^{\ell} \Pr[\mathrm{H}_{g-1}] - \Pr[\mathrm{H}_g] \tag{9}$$

$$= \Pr[\mathrm{H}_0] - \Pr[\mathrm{H}_\ell] \tag{10}$$

$$= \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{r},\mathsf{h}]}(\mathcal{A}) \tag{11}$$

Adversary $\mathcal{A}_{\mathsf{h}}$ (bottom left of Fig. 2) picks g at random in the range $1, \ldots, \ell$ and runs adversary $\mathcal{A}_g$ (right of Fig. 2) so $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_{\mathsf{h}}) = (1/\ell) \cdot \sum_{g=1}^{\ell} \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_g)$, which explains Equation (8). For the rest we begin by trying to picture what is going on.

We imagine a tree of depth $\ell+1$, meaning it has $\ell+2$ levels. The levels are numbered $0, 1, \ldots, \ell+1$, with 0 being the root. The root has $|\mathcal{D}|$ children while nodes at levels $1, \ldots, \ell$ have $|\mathsf{h}.\mathsf{D}|$ children each. A query $\mathbf{X}$ of $\mathcal{A}$ in game $\mathrm{DIST}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{r},\mathsf{h}], \mathsf{Out}}(\mathcal{A})$ specifies a path in this tree starting at the root and terminating at a node at level $n = |\mathbf{X}|$. Both the path and the final node are viewed as named by $\mathbf{X}$. To a queried node $\mathbf{X}$ we associate two labels, an internal label $T_1[\mathbf{X}] \in \mathcal{K}$ and an external label $T_2[\mathbf{X}] = \mathsf{Out}(T_1[\mathbf{X}]) \in \mathsf{Out}.\mathsf{R}$. The external label is the response to query $\mathbf{X}$. Since the first component of our 2-tier cascade is the family r of all functions from $\mathcal{D}$ to $\mathcal{K}$, we can view $\mathrm{DIST}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{r},\mathsf{h}], \mathsf{Out}}(\mathcal{A})$ as picking $T_1[\mathbf{X}[1]]$ at random from $\mathcal{K}$ and then setting $T_1[\mathbf{X}] = \mathsf{h}^*(T_1[\mathbf{X}[1]], \mathbf{X}[2..n])$ for all queries $\mathbf{X}$ of $\mathcal{A}$.

Now we consider the hybrid games $\mathrm{H}_0, \ldots, \mathrm{H}_\ell$ of Fig. 2. They simulate $\mathcal{A}$'s NEW, FN oracles via procedures $\text{NEW}^*, \text{FN}^*$, respectively. By assumption $\mathcal{A}$ makes exactly one NEW^* query, and this will have to be its first. In response H_s picks at random a function $f\colon [\![\mathcal{D},\mathcal{K}]\!] \to \mathcal{K}$. A query $\textsc{Fn}^*$ has the form $(i, \mathbf{X})$ but here i can only equal 1 and is ignored in responding. By assumption $2 \leq |\mathbf{X}| \leq \ell$. The game populates nodes at levels $2, \ldots, s$ of the tree with $T_1[\cdot]$ values that are obtained via f and thus are random elements of $\mathcal{K}$. For a node $\mathbf{X}$ at level $n \geq s+1$, the $T_1[\mathbf{X}[1..s+1]]$ value is obtained at random and then further values (if needed, meaning if $n \geq s+2$) are computed by applying the cascade h^* with key $T_1[\mathbf{X}[1..s+1]]$ to input $\mathbf{X}[s+2..n]$.

Consider game H_0, where $s = 0$. By assumption $n \geq 2$ so we will always be in the case $n \geq s+1$. In the Else statement, $Y \leftarrow f(\mathbf{X}[1])$ is initialized as a random element of $\mathcal{K}$. With this Y as the key, h^* is then applied to $\mathbf{X}[2..n]$ to get $T_1[\mathbf{X}]$. This means H_0 exactly mimics the $c = 1$ case of game $\mathrm{DIST}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{r},\mathsf{h}],\mathsf{Out}}(\mathcal{A})$, so that

$$\Pr[\mathrm{H}_0] = \Pr[\,\mathrm{DIST}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{r},\mathsf{h}]}(\mathcal{A}) \,|\, c = 1\,]\ . \tag{12}$$

At the other extreme, consider game H_ℓ, where $s = \ell$. By assumption $n \leq \ell+1$, yielding two cases. If $n \leq \ell$ we are in the $n \leq s$ case and the game, via f, the assigns $T_1[\mathbf{X}]$ a random value. If $n = \ell+1$ we are in the $n \geq s+1$ case, but the For loop does nothing so $T_1[\mathbf{X}]$ is again random. This means H_ℓ mimics the $c = 0$ case of game $\mathrm{DIST}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{r},\mathsf{h}],\mathsf{Out}}(\mathcal{A})$, except returning true exactly when the latter returns false. Thus

$$\Pr[\mathrm{H}_\ell] = 1 - \Pr[\,\mathrm{DIST}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{r},\mathsf{h}]}(\mathcal{A}) \,|\, c = 0\,]\ . \tag{13}$$

We will justify Equation (9) in a bit but we can now dispense with the rest of the chain. Equation (10) is obvious because the sum "telescopes." Equation (11) follows from Equations (12) and (13) and the formulation of dist advantage of Equation (5).

It remains to justify Equation (9), for which we consider the adversaries $\mathcal{A}_1, \ldots, \mathcal{A}_\ell$ on the right side of Fig. 2. Adversary $\mathcal{A}_g$ is playing the PRF, formally game $\mathrm{DIST}_{\mathsf{B},\mathsf{h}}$ on the left of Fig. 1 in our notation, with B the family of all functions from $\mathsf{h.D}$ to $\mathcal{K}$. It thus has oracles $\textsc{New}, \textsc{Fn}$. It will make crucial use of the assumed multi-user security of h, meaning its ability to query $\textsc{New}$ many times, keeping track in variable u of the number of instances it creates. It simulates the oracles of $\mathcal{A}$ of the same names via procedures $\textsc{New}^*, \textsc{Fn}^*$, sampling functions lazily rather than directly as in the games. Arrays T_1, T_2, U are assumed initially to be everywhere $\perp$ and get populated as the adversary assigns values to entries. A test of the form "If (not $T_1[\mathbf{X}]$) ... " returns true if $T_1[\mathbf{X}] = \perp$, meaning has not yet been initialized. In response to the (single) $\textsc{New}^*$ query of $\mathcal{A}$, adversary $\mathcal{A}_g$ does nothing. Following that, its strategy is to have the $T_1[\cdot]$ values of level g nodes populated, not explicitly, but implicitly by the keys in game $\mathrm{DIST}_{\mathsf{B},\mathsf{h}}$ created by the adversary's own $\textsc{New}$ queries, using array U to keep track of the user index associated to a node. $T_1[\cdot]$ values for nodes at levels $1, \ldots, g-1$ are random. At level $g+1$, the $T_1[\cdot]$ values are obtained via the adversary's $\textsc{Fn}$ oracle, and from then on via direct application of the cascade h^*. One crucial point is that, if $\mathcal{A}_g$ does not know the $T_1[\cdot]$ values at level g, how does it respond to a length g query $\mathbf{X}$ with the right $T_2[\cdot]$ value? This is where the leakage enters, the response being the leakage provided by the $\textsc{New}$ oracle. The result is that for every $g \in \{1, \ldots, \ell\}$ we have

$$\Pr[\,\mathrm{DIST}_{\mathsf{B},\mathsf{h}}(\mathcal{A}_g) \,|\, c = 1\,] = \Pr[\mathrm{H}_{g-1}] \tag{14}$$

$$1 - \Pr[\,\mathrm{DIST}_{\mathsf{B},\mathsf{h}}(\mathcal{A}_g) \,|\, c = 0\,] = \Pr[\mathrm{H}_g]\ , \tag{15}$$

where c is the challenge bit in game $\mathrm{DIST}_{\mathsf{B},\mathsf{h}}$. Thus

$$\begin{aligned}\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_g) &= \Pr[\,\mathrm{DIST}_{\mathsf{B},\mathsf{h}}(\mathcal{A}_g) \,|\, c = 1\,] - (1 - \Pr[\,\mathrm{DIST}_{\mathsf{B},\mathsf{h}}(\mathcal{A}_g) \,|\, c = 0\,]) \\ &= \Pr[\mathrm{H}_{g-1}] - \Pr[\mathrm{H}_g]\ . \end{aligned} \tag{16}$$

This justifies Equation (9). ▌

We now extend the above to the case where the first tier g of the 2-tier cascade is a PRF rather than a random function. We will exploit PRF security of g to reduce this to the prior case.

Theorem 5.2 *Let $\mathcal{K}$ be a non-empty set. Let $\mathsf{g}\colon \mathsf{g.K} \times \mathsf{g.D} \to \mathcal{K}$ and $\mathsf{h}\colon \mathcal{K} \times \mathsf{h.D} \to \mathcal{K}$ be function families. Let $\mathsf{Out}\colon \mathcal{K} \to \mathsf{Out.R}$ be an output transform. Let A be the family of all functions with domain $[\![\mathsf{g.D}, \mathsf{h.D}]\!]$ and range $\mathcal{K}$. Let $\mathcal{A}$ be an adversary making exactly one query to its* New *oracle followed by at most q queries to its* Fn *oracle, the second argument of each of the queries in the latter case being a vector $\mathbf{X} \in [\![\mathsf{g.D}, \mathsf{h.D}]\!]$ with $2 \le |\mathbf{X}| \le \ell + 1$. The proof shows how to construct adversaries $\mathcal{A}_\mathsf{h}, \mathcal{A}_\mathsf{g}$ such that*

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A}) \le \ell \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_\mathsf{h}) + 2\,\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{g}}(\mathcal{A}_\mathsf{g}) \ . \tag{17}$$

Adversary $\mathcal{A}_\mathsf{h}$ makes at most q queries to its New *oracle and at most q queries to its* Fn *oracle. Adversary $\mathcal{A}_\mathsf{g}$ makes one query to its* New *oracle and at most q queries to its* Fn *oracle. The running time of both constructed adversaries is about that of $\mathcal{A}$ plus the time for $q\ell$ computations of h.*

Proof of Theorem 5.2: Let $\mathcal{D} = \mathsf{g.D}$. Let r be the family of all functions with domain $\mathcal{D}$ and range $\mathcal{K}$. As shorthand for the relevant games, let

$$\mathrm{G}_0 = \mathrm{DIST}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A}) \quad \text{and} \quad \mathrm{G}_1 = \mathrm{DIST}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{r},\mathsf{h}]}(\mathcal{A}) \ .$$

Then

$$\begin{aligned}\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A}) &= 2\Pr[\mathrm{G}_0] - 1 \\ &= 2\left(\Pr[\mathrm{G}_0] - \Pr[\mathrm{G}_1] + \Pr[\mathrm{G}_1]\right) - 1 \\ &= 2\left(\Pr[\mathrm{G}_0] - \Pr[\mathrm{G}_1]\right) + 2\Pr[\mathrm{G}_1] - 1 \ . \end{aligned} \tag{18}$$

Lemma 5.1 provides adversary $\mathcal{A}_\mathsf{h}$ such that

$$2\Pr[\mathrm{G}_1] - 1 \le \ell \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_\mathsf{h}) \ . \tag{19}$$

Let adversary $\mathcal{A}_\mathsf{g}$ be as follows:

```
Adversary $\mathcal{A}_\mathsf{g}^{\mathrm{New},\mathrm{Fn}}$
$c \leftarrow\!\!\$ \{0,1\}$ ; $c' \leftarrow\!\!\$ \mathcal{A}^{\mathrm{New},\mathrm{Fn}^*}$
If $(c = c')$ then return 1 else return 0

$\mathrm{Fn}^*(i, \mathbf{X})$
$Y \leftarrow \mathrm{Fn}(1, \mathbf{X}[1])$
For $j = 2, \ldots, |\mathbf{X}|$ do $Y \leftarrow \mathsf{h}(Y, \mathbf{X}[j])$
$Z \leftarrow \mathsf{Out}(Y)$ ; Return $Z$
```

Adversary $\mathcal{A}_\mathsf{g}$ responds to the single New query that $\mathcal{A}$ makes directly via its own New oracle. It responds to Fn queries of $\mathcal{A}$ via procedure Fn*. The latter applies $\mathcal{A}_\mathsf{g}$'s own Fn oracle to the first component of $\mathbf{X}$ to get a key Y, and then applies h^* with key Y to the rest of $\mathbf{X}$, finally applying the output function to get the value returned. Then we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{g}}(\mathcal{A}_\mathsf{g}) &= \Pr[\,\mathrm{DIST}_{\mathsf{r},\mathsf{g}}(\mathcal{A}_\mathsf{g}) \,|\, c = 1\,] - (1 - \Pr[\,\mathrm{DIST}_{\mathsf{r},\mathsf{g}}(\mathcal{A}_\mathsf{g}) \,|\, c = 0\,]) \\ &= \Pr[\mathrm{G}_0] - \Pr[\mathrm{G}_1] \end{aligned} \tag{20}$$

Putting together Equations (18), (19) and (20) yields Equation (17). ▌

Multi-user security of 2-tier augmented cascade. We now want to assess the multi-user security of a 2-tier augmented cascade. This means we want to bound $\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out} \circ \mathsf{A}, \mathsf{Out} \circ \mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A})$

with everything as in Theorem 5.2 above except that $\mathcal{A}$ can now make any number u of NEW queries rather than just one. We could do this easily by applying Lemma 4.1 to Theorem 5.2, resulting in a bound that is u times the bound of Equation (17). We consider Theorem 5.3 below the most interesting result of this section. It says one can do much better, and in fact the bound for the multi-user case is not much different from that for the single-user case.

Theorem 5.3 *Let $\mathcal{K}$ be a non-empty set. Let $\mathsf{g}\colon \mathsf{g.K} \times \mathsf{g.D} \to \mathcal{K}$ and $\mathsf{h}\colon \mathcal{K} \times \mathsf{h.D} \to \mathcal{K}$ be function families. Let $\mathsf{Out}\colon \mathcal{K} \to \mathsf{Out.R}$ be an output transform. Let A be the family of all functions with domain $[\![\mathsf{g.D}, \mathsf{h.D}]\!]$ and range $\mathcal{K}$. Let $\mathcal{A}$ be an adversary making at most u queries to its* NEW *oracle and at most q queries to its* FN *oracle, the second argument of each of the queries in the latter case being a vector $\mathbf{X} \in [\![\mathsf{g.D}, \mathsf{h.D}]\!]$ with $2 \le |\mathbf{X}| \le \ell + 1$. The proof shows how to construct adversaries $\mathcal{A}_{\mathsf{h}}, \mathcal{A}_{\mathsf{g}}$ such that*

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A}) \le \ell \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_{\mathsf{h}}) + 2\,\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{g}}(\mathcal{A}_{\mathsf{g}}) \ . \tag{21}$$

Adversary $\mathcal{A}_{\mathsf{h}}$ makes at most q queries to its NEW *oracle and at most q queries to its* FN *oracle. Adversary $\mathcal{A}_{\mathsf{g}}$ makes u queries to its* NEW *oracle and at most q queries to its* FN *oracle. The running time of both constructed adversaries is about that of $\mathcal{A}$ plus the time for $q\ell$ computations of* h.

A comparison of Theorems 5.2 and 5.3 shows that the bound of Equation (21) is the same as that of Equation (17). So where are we paying for u now not being one? It is reflected only in the resources of adversary $\mathcal{A}_{\mathsf{g}}$, the latter in Theorem 5.3 making u queries to its NEW oracle rather than just one in Theorem 5.2.

The proof below showcases one of the advantages of the 2-tier cascade over the basic single-tier one. Namely, by appropriate choice of instantiation of the first tier, we can reduce multi-user security to single-user security in a modular way. In this way we avoid re-entering the proofs above. Indeed, the ability to do this is one of the main reasons we introduced the 2-tier cascade.

Proof of Theorem 5.3: Let $\mathcal{D} = [1..u]$. Let $\overline{\mathsf{r}}$ be the family of all functions with domain $\mathcal{D}$ and range $\mathsf{g.K}$. Let function family $\overline{\mathsf{g}}\colon \overline{\mathsf{r}}.\mathsf{K} \times (\mathcal{D} \times \mathsf{g.D}) \to \mathcal{K}$ be defined by $\overline{\mathsf{g}}(f,(i,x)) = \mathsf{g}(f(i),x)$. Let B be the family of all functions with domain $[\![\mathcal{D} \times \mathsf{g.D}, \mathsf{h.D}]\!]$ and range $\mathcal{K}$. The main observation is as follows. Suppose $i \in \mathcal{D}$ and $\mathbf{X} \in [\![\mathsf{g.D}, \mathsf{h.D}]\!]$. Let $\mathbf{Y} \in [\![\mathcal{D} \times \mathsf{g.D}, \mathsf{h.D}]\!]$ be defined by $\mathbf{Y}[1] = (i, \mathbf{X}[1])$ and $\mathbf{Y}[j] = \mathbf{X}[j]$ for $2 \le j \le |\mathbf{X}|$. Let $f\colon \mathcal{D} \to \mathsf{g.K}$ be a key for $\overline{\mathsf{g}}$. Then $f(i) \in \mathsf{g.K}$ is a key for g, and

$$\mathbf{CSC}[\overline{\mathsf{g}},\mathsf{h}](f,\mathbf{Y}) = \mathbf{CSC}[\mathsf{g},\mathsf{h}](f(i),\mathbf{X}) \ . \tag{22}$$

Think of $f(i)$ as the key for instance i. Then Equation (22) allows us to obtain values of $\mathbf{CSC}[\mathsf{g},\mathsf{h}]$ for different instances $i \in \mathcal{D}$ via values of $\mathbf{CSC}[\overline{\mathsf{g}},\mathsf{h}]$ on a single instance with key f. This will allow us to reduce the multi-user security of $\mathbf{CSC}[\mathsf{g},\mathsf{h}]$ to the single-user security of $\mathbf{CSC}[\overline{\mathsf{g}},\mathsf{h}]$. Theorem 5.2 will allow us to measure the latter in terms of the prf security of h under leakage and the (plain) prf security of $\overline{\mathsf{g}}$. The final step will be to measure the prf security of $\overline{\mathsf{g}}$ in terms of that of g.

Proceeding to the details, let adversary $\mathcal{B}$ be as follows:

```
Adversary $\mathcal{B}^{\text{NEW},\text{FN}}$
NEW()
$b' \leftarrow_{\$} \mathcal{A}^{\text{NEW}^*,\text{FN}^*}$ ; Return $b'$
NEW*()

FN*$(i, \mathbf{X})$
$\mathbf{Y}[1] \leftarrow (i, \mathbf{X}[1])$
For $j = 2, \ldots, |\mathbf{X}|$ do $\mathbf{Y}[j] \leftarrow \mathbf{X}[j]$
$Z \leftarrow$ FN$(1, \mathbf{Y})$ ; Return $Z$
```

Then we have

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{Out}\circ\mathsf{B},\mathsf{Out}\circ\mathbf{CSC}[\overline{\mathsf{g}},\mathsf{h}]}(\mathcal{B}) \tag{23}$$

$$\leq \ell\cdot\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{h},\mathsf{Out}}(\mathcal{A}_{\mathsf{h}}) + 2\,\mathsf{Adv}^{\mathsf{prf}}_{\overline{\mathsf{g}}}(\mathcal{A}_{\overline{\mathsf{g}}}) \tag{24}$$

Adversary $\mathcal{B}$ is allowed only one New query, and begins by making it so as to initialize instance 1 in its game. It answers queries of $\mathcal{A}$ to its New oracle via procedure New*. Adversary $\mathcal{A}$ can make up to u queries to New*, but, as the absence of code for New* indicates, this procedure does nothing, meaning no action is taken when $\mathcal{A}$ makes a New* query. When $\mathcal{A}$ queries its Fn oracle, $\mathcal{B}$ answers via procedure Fn*. The query consists of an instance index i with $1 \leq i \leq u$ and a vector $\mathbf{X}$. Adversary $\mathcal{B}$ creates $\mathbf{Y}$ from $\mathbf{X}$ as described above. Namely it modifies the first component of $\mathbf{X}$ to pre-pend i, so that $\mathbf{Y}[1] \in \mathcal{D}\times\mathsf{g.D}$ is in the domain of $\overline{\mathsf{g}}$. It leaves the rest of the components unchanged, and then calls its own Fn oracle on vector $\mathbf{Y} \in [\![\mathcal{D}\times\mathsf{g.D},\mathsf{h.D}]\!]$. The instance used is 1, regardless of i, since $\mathcal{B}$ has only one instance active. The result Z of Fn is returned to $\mathcal{A}$ as the answer to its query. Equation (23) is now justified by Equation (22), thinking of $f(i)$ as the key K_i chosen in game $\mathrm{DIST}_{\mathsf{Out}\circ\mathsf{A},\mathsf{Out}\circ\mathbf{CSC}[\mathsf{g},\mathsf{h}]}(\mathcal{A})$ where f is the (single) key chosen in game $\mathrm{DIST}_{\mathsf{Out}\circ\mathsf{B},\mathsf{Out}\circ\mathbf{CSC}[\overline{\mathsf{g}},\mathsf{h}]}(\mathcal{B})$. Theorem 5.2 applied to $\overline{\mathsf{g}},\mathsf{h}$ and adversary $\mathcal{B}$ provides the adversaries $\mathcal{A}_{\mathsf{h}},\mathcal{A}_{\overline{\mathsf{g}}}$ of Equation (24).

Now consider adversary $\mathcal{A}_{\mathsf{g}}$ defined as follows:

```
Adversary $\mathcal{A}_{\mathsf{g}}^{\text{New},\text{Fn}}$             | Fn*$(j,X)$
For $i = 1,\ldots,u$ do New()                      | $(i,x) \leftarrow X$ ; $Y \leftarrow$ Fn$(i,x)$
$b' \leftarrow_{\$} \mathcal{A}_{\overline{\mathsf{g}}}^{\text{New}^*,\text{Fn}^*}$ ; Return $b'$          | Return $Y$
New*()                                             |
```

Adversary $\mathcal{A}_{\mathsf{g}}$ begins by calling its New oracle u times to initialize u instances. It then runs $\mathcal{A}_{\overline{\mathsf{g}}}$, answering the latter's oracle queries via procedures New*, Fn*. By Theorem 5.2 we know that $\mathcal{A}_{\overline{\mathsf{g}}}$ makes only one New* query. In response the procedure New* above does nothing. When $\mathcal{A}_{\overline{\mathsf{g}}}$ makes query j, X to Fn* we know that $j = 1$ and $X \in \mathcal{D}\times\mathsf{g.D}$. Procedure Fn* parses X as (i,x). It then invokes its own Fn oracle with instance i and input x and returns the result Y to $\mathcal{A}_{\overline{\mathsf{g}}}$. We have

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{g}}(\mathcal{A}_{\mathsf{g}}) = \mathsf{Adv}^{\mathsf{prf}}_{\overline{\mathsf{g}}}(\mathcal{A}_{\overline{\mathsf{g}}})\ . \tag{25}$$

Equations (24) and (25) imply Equation (21). ▌

One might ask why prove Theorem 5.3 for a 2-tier augmented cascade $\mathsf{Out}\circ\mathbf{CSC}[\mathsf{g},\mathsf{h}]$ instead of a single tier one $\mathsf{Out}\circ\mathbf{CSC}[\mathsf{h},\mathsf{h}]$. Isn't the latter the one of ultimate interest in usage? We establish a more general result in Theorem 5.3 because it allows us to analyze AMAC itself by setting g to the dual of h [2], and also for consistency with Theorem 5.2.

6 Framework for ideal-model cryptography

In Section 5 we reduced the (mu) security of the augmented cascade tightly to the assumed mu prf security of the compression function under leakage. To complete the story, we will, in Section 7, bound the mu prf security of an ideal compression function under leakage and thence obtain concrete bounds for the mu security of the augmented cascade in the same model. Additionally, we will consider the same questions when the compression function is not directly ideal but obtained via

```
Game $\mathrm{PRF}_{\mathsf{F},\mathbf{P}}(\mathcal{A})$
$v \leftarrow 0$
$c \leftarrow\$ \{0,1\}$ ; $\mathsf{P} \leftarrow\$ \mathbf{P}$ ; $c' \leftarrow\$ \mathcal{A}^{\textsc{New},\textsc{Fn},\textsc{Prim}}$
Return $(c = c')$

$\textsc{New}()$
$v \leftarrow v + 1$
If $(c = 1)$ then $F_v \leftarrow\$ \mathsf{F}^{\textsc{Prim}}$
Else $F_v \leftarrow\$ \mathsf{A}$

$\textsc{Fn}(i, x)$
Return $F_i(x)$

$\textsc{Prim}(x)$
$y \leftarrow \mathsf{P}(x)$ ; Return $y$
```

```
Game $\mathrm{PRF}_{\mathsf{F},\mathsf{Out},\mathbf{P}}(\mathcal{A})$
$v \leftarrow 0$
$c \leftarrow\$ \{0,1\}$ ; $\mathsf{P} \leftarrow\$ \mathbf{P}$ ; $c' \leftarrow\$ \mathcal{A}^{\textsc{New},\textsc{Fn},\textsc{Prim}}$
Return $(c = c')$

$\textsc{New}()$
$v \leftarrow v + 1$ ; $K_v \leftarrow\$ \mathsf{F.K}$
If $(c = 1)$ then $F_v \leftarrow \mathsf{F}^{\textsc{Prim}}(K_v, \cdot)$
Else $F_v \leftarrow\$ \mathsf{A}$
Return $\mathsf{Out}(K_v)$

$\textsc{Fn}(i, x)$
Return $F_i(x)$

$\textsc{Prim}(x)$
$y \leftarrow \mathsf{P}(x)$ ; Return $y$
```

Figure 3: **Games defining prf security of function family F in the presence of an ideal primitive $\mathbf{P}$.** In the basic (left) case there is no leakage, while in the extended (right) case there is leakage represented by Out.

the Davies-Meyer transform on an ideal blockcipher, reflecting the design in popular hash functions. If we gave separate, ad hoc definitions for all these different constructions in different ideal models for different goals, it would be a lot of definitions. Accordingly we introduce a general definition of an ideal primitive (that may be of independent interest) and give a general definition of PRF security of a function family with access to an instance of an ideal primitive, both for the basic setting and the setting with leakage. A reader interested in our results on the mu prf security of ideal primitives can jump ahead to Section 7 and refer back here as necessary.

IDEALIZED CRYPTOGRAPHY. We define an *ideal primitive* to simply be a function family $\mathbf{P}\colon \mathbf{P}.\mathsf{K} \times \mathbf{P}.\mathsf{D} \to \mathbf{P}.\mathsf{R}$. Below we will provide some examples but first let us show how to lift security notions to idealized models using this definition by considering the cases of interest to us, namely PRFs and PRFs under leakage.

An *oracle function family* F specifies for each function P in its *oracle space* $\mathsf{F.O}$ a function family $\mathsf{F}^{\mathsf{P}}\colon \mathsf{F.K} \times \mathsf{F.D} \to \mathsf{F.R}$. We say F and ideal primitive $\mathbf{P}$ are *compatible* if $\{\, \mathbf{P}(\mathsf{KK}, \cdot) : \mathsf{KK} \in \mathbf{P}.\mathsf{K} \,\} \subseteq \mathsf{F.O}$, meaning instances of $\mathbf{P}$ are legitimate oracles for F. These represent constructs whose security we want to measure in an idealized model represented by $\mathbf{P}$.

We associate to $\mathsf{F}, \mathbf{P}$ and adversary $\mathcal{A}$ the game PRF in the left of Fig. 3. In this game, A is the family of all functions with domain $\mathsf{F.D}$ and range $\mathsf{F.R}$. The game begins by picking an instance $\mathsf{P}\colon \mathbf{P}.\mathsf{D} \to \mathbf{P}.\mathsf{R}$ of $\mathbf{P}$ at random. The function P is provided as oracle to F and to $\mathcal{A}$ via procedure $\textsc{Prim}$. The game is in the multi-user setting, and when $c = 1$ it selects a new instance F_v at random from the function family F^{P}. Otherwise it selects F_v to be a random function from $\mathsf{F.D}$ to $\mathsf{F.R}$. As usual a query i, x to $\textsc{Fn}$ must satisfy $1 \le i \le v$ and $x \in \mathsf{F.D}$. A query to $\textsc{Prim}$ must be in the set $\mathbf{P}.\mathsf{D}$. We let $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},\mathbf{P}}(\mathcal{A}) = 2\Pr[\mathrm{PRF}_{\mathsf{F},\mathbf{P}}(\mathcal{A})] - 1$ be the advantage of $\mathcal{A}$.

We now extend this to allow leakage on the key. Let $\mathsf{Out}\colon \mathsf{F.K} \to \mathsf{Out.R}$ be a function with domain $\mathsf{F.K}$ and range $\mathsf{Out.R}$. Game PRF on the right of Fig. 3 is now associated not only to $\mathsf{F}, \mathbf{P}$ and an adversary $\mathcal{A}$ but also to Out. The advantage of $\mathcal{A}$ is $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},\mathsf{Out},\mathbf{P}}(\mathcal{A}) = 2\Pr[\mathrm{PRF}_{\mathsf{F},\mathsf{Out},\mathbf{P}}(\mathcal{A})] - 1$.

Capturing particular ideal models. The above framework allows us to capture the random oracle model, ideal cipher model and many others as different choices of the ideal primitive $\mathbf{P}$. Not all of these are relevant to our paper but we discuss them to illustrate how the framework captures known settings.

Let $\mathcal{Y}$ be a non-empty set. Let $\mathbf{P}.\mathsf{K}$ be the set of all functions $\mathsf{P}\colon \{0,1\}^* \to \mathcal{Y}$. (Each function is represented in some canonical way, in this case for example as a vector over $\mathcal{Y}$ of infinite length.) Let $\mathbf{P}\colon \mathbf{P}.\mathsf{K} \times \{0,1\}^* \to \mathcal{Y}$ be defined by $\mathbf{P}(\mathsf{P}, x) = \mathsf{P}(x)$. Then $\mathsf{P} \leftarrow\$ \mathbf{P}$ is a random oracle with domain $\{0,1\}^*$ and range $\mathcal{Y}$. In this case, an oracle function family compatible with $\mathbf{P}$ is simply a function family in the random oracle model, and its prf security in the random oracle model is measured by $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},\mathbf{P}}(\mathcal{A})$.

Similarly let $\mathbf{P}.\mathsf{K}$ be the set of all functions $\mathsf{P}\colon \{0,1\}^* \times \mathbb{N} \to \{0,1\}^*$ with the property that $|\mathsf{P}(x,l)| = l$ for all $(x,l) \in \{0,1\}^* \times \mathbb{N}$. Let $\mathbf{P}\colon \mathbf{P}.\mathsf{K} \times (\{0,1\}^* \times \mathbb{N}) \to \{0,1\}^*$ be defined by $\mathbf{P}(\mathsf{P},(x,l)) = \mathsf{P}(x,l)$. Then $\mathsf{P} \leftarrow\$ \mathbf{P}$ is a variable output length random oracle with domain $\{0,1\}^*$ and range $\{0,1\}^*$.

Let $\mathcal{D}$ be a non-empty set. To capture the single random permutation model, let $\mathbf{P}.\mathsf{K}$ be the set of all permutations $\pi\colon \mathcal{D} \to \mathcal{D}$. Let $\mathbf{P}.\mathsf{D} = \mathcal{D} \times \{+,-\}$. Let $\mathbf{P}.\mathsf{R} = \mathcal{D}$. Define $\mathbf{P}(\pi,(x,+)) = \pi(x)$ and $\mathbf{P}(\pi,(y,-)) = \pi^{-1}(y)$ for all $\pi \in \mathbf{P}.\mathsf{K}$ and all $x, y \in \mathcal{D}$. An oracle for an instance $\mathsf{P} = \mathbf{P}(\pi,\cdot)$ of $\mathbf{P}$ thus allows evaluation of both π and π^{-1} on inputs of the caller's choice.

Finally we show how to capture the ideal cipher model. If $\mathcal{K}, \mathcal{D}$ are non-empty sets, a function family $E\colon \mathcal{K} \times \mathcal{D} \to \mathcal{D}$ is a blockcipher if $E(K,\cdot)$ is a permutation on $\mathcal{D}$ for every $K \in \mathcal{K}$, in which case $E^{-1}\colon \mathcal{K} \times \mathcal{D} \to \mathcal{D}$ denotes the blockcipher in which $E^{-1}(K,\cdot)$ is the inverse of the permutation $E(K,\cdot)$ for all $K \in \mathcal{K}$. Let $\mathbf{P}.\mathsf{K}$ be the set of all block ciphers $E\colon \mathcal{K} \times \mathcal{D} \to \mathcal{D}$. Let $\mathbf{P}.\mathsf{D} = \mathcal{K} \times \mathcal{D} \times \{+,-\}$. Let $\mathbf{P}.\mathsf{R} = \mathcal{D}$. Define $\mathbf{P}(E,(K,X,+)) = E(K,X)$ and $\mathbf{P}(E,(K,Y,-)) = E^{-1}(K,Y)$ for all $E \in \mathbf{P}.\mathsf{K}$ and all $X, Y \in \mathcal{D}$. An oracle for an instance $\mathsf{P} = \mathbf{P}(E,\cdot)$ of $\mathbf{P}$ thus allows evaluation of both E and E^{-1} on inputs of the caller's choice.

7 Security of the compression function under leakage

In Section 5 we reduced the (multi-user) security of the augmented cascade tightly to the assumed multi-user prf security of the compression function under leakage. To complete the story, we now study (bound) the multi-user prf security of the compression function under leakage. This will be done assuming the compression function is ideal. Combining these results with those of Section 5 we will get concrete bounds for the security of the augmented cascade for use in applications, discussed in Section 8.

In the (leak-free) multi-user setting, it is well known that prf security of a compression function decreases linearly in the number of users. We will show that this is an extreme case, and as the amount of leakage increases, the multi-user prf security degrades far more gracefully in the number of users (Theorem 7.2). This (perhaps counterintuitive) phenomenon will turn out to be essential to obtain good bounds on augmented cascades. We begin below with an informal overview of the bounds and why this phenomenon occurs.

Overview of bounds. The setting of an ideal compression function mapping $\mathcal{K} \times \mathcal{X} \to \mathcal{D}$ is formally captured, in the framework of Section 6, by the ideal primitive $\mathbf{F}\colon \mathbf{F}.\mathsf{K} \times (\mathcal{K} \times \mathcal{X}) \to \mathcal{K}$ defined as follows. Let $\mathbf{F}.\mathsf{K}$ be the set of all functions mapping $\mathcal{K} \times \mathcal{X} \to \mathcal{K}$ and let $\mathbf{F}(\mathsf{f},(K,X)) = \mathsf{f}(K,X)$. Now, the construction we are interested in is the simplest possible, namely the compression function itself. Formally, again as per Section 6, this means we consider the oracle function family CF whose oracle space $\mathsf{CF.O}$ consists of all functions $\mathsf{f}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{K}$, and with $\mathsf{CF}^{\mathsf{f}} = \mathsf{f}$.

For this overview we let $\mathcal{K} = \{0,1\}^c$. We contrast the prf security of an ideal compression

	$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathbf{F}}(\mathcal{B})$	$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}(\mathcal{B})$
su	$\frac{q_{\mathrm{F}}}{2^c}$	$\frac{q_{\mathrm{F}}}{2^{c-r}}$
mu, trivial	$\frac{u(q+q_{\mathrm{F}})}{2^c}$	$\frac{u(q+q_{\mathrm{F}})}{2^{c-r}}$
mu, dedicated	$\frac{u^2+2uq_{\mathrm{F}}}{2^{c+1}}$	$\frac{u^2+2uq_{\mathrm{F}}+1}{2^c}+\frac{3crq_{\mathrm{F}}}{2^{c-r}}$

Figure 4: **Upper bounds on prf advantage of an adversary $\mathcal{B}$ attacking an ideal compression function mapping $\{0,1\}^c \times \mathcal{X}$ to $\{0,1\}^c$. Left:** Basic case, without leakage. **Right:** With leakage Out being the truncation function that returns the first $r \le c$ bits of its output. **First row:** Single user security, q_{F} is the number of queries to the ideal compression function. **Second row:** Multi-user security as obtained trivially by applying Lemma 4.1 to the su bound, u is the number of users. **Third row:** Multi-user security as obtained by a dedicated analysis, with the bound in the leakage case being from Theorem 7.2.

function along two dimensions: (1) Number of users, meaning su or mu, and (2) basic (no leakage) or with leakage. The bounds are summarized in Fig. 4 and discussed below. When we say the (i,j) table entry we mean the row i, column j entry of the table of Fig. 4.

First consider the basic (no leakage) case. We want to upper bound $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathbf{F}}(\mathcal{B})$ for an adversary $\mathcal{B}$ making q_{F} queries to the ideal compression function (oracle PRIM) and q queries to oracle FN. In the su setting (one NEW query) it is easy to see that the bound is the $(1,1)$ table entry. This is because a fairly standard argument bounds the advantage by the probability that $\mathcal{B}$ makes a PRIM query containing the actual secret key K used to answer FN queries. We refer to issuing such a query as *guessing the secret key* K. Note that this probability is actually independent of the number q of FN queries and q does not figure in the bound. Now move to the mu setting, and let $\mathcal{B}$ make u queries to its NEW oracle. Entry (2,1) of the table is the trivial bound obtained via Lemma 4.1 applied with F_1 being our ideal compression function and F_0 a family of all functions, but one has to be careful in applying the lemma. The subtle point is that adversary $\mathcal{A}_1$ built in Lemma 4.1 runs $\mathcal{B}$ but makes an additional q queries to PRIM to compute the function F_1, so its advantage is the $(1,1)$ table entry with q_{F} replaced by $q_{\mathrm{F}}+q$. This term gets multiplied by u according to Equation (6), resulting in our $(2,1)$ table entry. A closer look shows one can do a tad better: the bound of the $(1,1)$ table entry extends with the caveat that a collisions between two different keys also allows the adversary to distinguish. In other words, the advantage is now bounded by the probability that $\mathcal{B}$ guesses *any* of the u keys $K_1,\ldots,K_u$, or that any two of these keys collide. This yields the $(3,1)$ entry of the table. Either way, the (well known) salient point here is that the advantage in the mu case is effectively u times the one in the su case.

We show that the growth of the advantage as a function of the number of users becomes far more favorable when the adversary obtains some leakage about the secret key under some function Out. For concreteness we take the leakage function to be truncation to r bits, meaning $\mathsf{Out} = \mathsf{TRUNC}_r$ is the function that returns the first $r \le c$ bits of its input. (Theorem 7.2 will consider a general Out.) Now we seek to bound $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}(\mathcal{B})$. Now, given only $\mathsf{TRUNC}_r(K)$ for a secret key K, then there are only 2^{c-r} candidate secret keys consistent with this leakage, thus increasing the probability that the adversary can guess the secret key. Consequently, the leakage-free bound from of the (1,1) entry generalizes to the bound of the (1,2) entry. Moving to multiple users, the (2,2) entry represents the naive bound obtained by applying Lemma 4.1. It is perhaps natural to expect that this is best possible as in the no-leakage case. We however observe that this is overly pessimistic. To this end, we exploit the following simple fact: *Every* PRIM *query* (K, X) *made by* $\mathcal{B}$ *to the ideal compression function can only help in guessing a key* K_i *such that* $\mathsf{Out}(K) = \mathsf{Out}(K_i)$. In particular, every PRIM query (K, X) has only roughly $m \cdot 2^{-(c-r)}$ chance of guessing one of the u keys, where m is the number of generated keys K_i such that $\mathsf{Out}(K_i) = K$. A standard balls-into-bins arguments (Lemma 7.1) can be used to infer that except with small probability (e.g., 2^{-c}), we always have $m \le 2u/2^r + 3cr$ for any K. Combining these two facts yields our bound, which is the (3,2) entry of the table. Theorem 7.2 gives a more general result and the full proof. Note that if $r = 0$, i.e., nothing is leaked, this is close to the bound of the (1,3) entry and the bound does grow linearly with the number of users, but as r grows, the $3crq_{\mathrm{F}} \cdot 2^{-(c-r)}$ term becomes the leading one, and does *not* grow with u. We now proceed to the detailed proof of the (3,2) entry.

COMBINATORIAL PRELIMINARIES. Our statements below will depend on an appropriate multi-collision probability of the output function $\mathsf{Out}\colon \mathsf{Out.D} \to \mathsf{Out.R}$. In particular, for any $X_1, \ldots, X_u \in \mathsf{Out.R}$, we first define

$$\mu(X_1, \ldots, X_u) = \max_{Y \in \mathsf{Out.R}} |\{\, i \,:\, X_i = Y \,\}| \ ,$$

i.e., the number of occurrences of the most frequent value amongst $X_1, \ldots, X_u$. In particular, this is an integer between 1 and u, and $\mu(X_1, \ldots, X_u) = 1$ if all elements are distinct, whereas $\mu(X_1, \ldots, X_u) = u$ if they are all equal. (Note when $u = 1$ the function has value 1.) Then, the m-collision probability of Out for u users is defined as

$$\mathsf{P}^{\mathsf{coll}}_{\mathsf{Out}}(u, m) = \Pr_{K_1,\ldots,K_u \twoheadleftarrow\$\, \mathsf{Out.D}}\left[\, \mu(\mathsf{Out}(K_1), \ldots, \mathsf{Out}(K_u)) \ge m \,\right] \ . \tag{26}$$

We provide a bound on $\mathsf{P}^{\mathsf{coll}}_{\mathsf{Out}}(u, m)$ for the case where $\mathsf{Out}(K)$, for a random K, is close enough to uniform. (We stress that a combinatorial restriction on Out is necessary for this probability to be small – it would be one if Out is the contant function, for example.) To this end, denote

$$\delta(\mathsf{Out}) = \mathbf{SD}(\mathsf{Out}(\mathrm{K}), \mathrm{R}) = \frac{1}{2} \sum_{y \in \mathsf{Out.R}} \left| \Pr\left[\, \mathsf{Out}(\mathrm{K}) = y \,\right] - \frac{1}{|\mathsf{Out.R}|} \right| \ , \tag{27}$$

i.e., the statistical distance between $\mathsf{Out}(\mathrm{K})$, where K is uniform on $\mathsf{Out.D}$, and a random variable R uniform on $\mathsf{Out.R}$.

We will use the following lemma, which we prove using standard balls-into-bins techniques.

Lemma 7.1 (Multi-collision probability) *Let* $\mathsf{Out} : \mathsf{Out.D} \to \mathsf{Out.R}$*,* $u \ge 1$*, and* $\lambda \ge 0$*. Then, for any* $m \le u$ *such that*

$$m \ge \frac{2u}{|\mathsf{Out.R}|} + \lambda \ln |\mathsf{Out.R}| \ , \tag{28}$$

we have

$$\mathsf{P}^{\mathsf{coll}}_{\mathsf{Out}}(u, m) \le u \cdot \delta(\mathsf{Out}) + \exp(-\lambda/3) \ .$$

We stress that the factor 2 in Equation (28) can be omitted (one can use an additive Chernoff bound when u is sufficiently large in the proof given below, rather than a multiplicative one) at the cost of a less compact statement. As this factor will not be crucial in the following, we keep this simpler variant.

Proof of Lemma 7.1: To start with, note that the probability of having at least m values colliding increases only by at most $u \cdot \delta(\mathsf{Out})$ if in the definition of $\mathsf{P}^{\mathsf{coll}}$, we replace the u outputs of Out under random inputs $K_1, \ldots, K_u$ by uniform elements of $\mathsf{Out.R}$. Thus, we are going to consider the uniform case only, at the cost of the additive factor $u \cdot \delta(\mathsf{Out})$. Throughout this proof, denote $R = |\mathsf{Out.R}|$ for notational convenience.

To this end, let $\mathrm{R}_1, \ldots, \mathrm{R}_u$ be each uniform over $\mathsf{Out.R}$, and let us fix $Y \in \mathsf{Out.R}$. Let $\mathrm{T}_i \in \{0,1\}$ be 1 if and only if $\mathrm{R}_i = Y$, and 0 otherwise. We are interested in bounding $\mathrm{T} = \sum_{i=1}^{u} \mathrm{T}_i$, i.e., showing that the probability it is at least m is at most $\frac{1}{R} \cdot \exp(-\lambda/3)$. Note that $\mu = \mathbf{E}[\,\mathrm{T}\,] = \frac{u}{R}$, and recall that by the Chernoff bound,

$$\Pr[\,\mathrm{T} \geq (1+\epsilon) \cdot \mu\,] \leq \exp\left(-\frac{\epsilon^2}{2+\epsilon}\mu\right) .$$

Note that $m \geq 2\mu + \lambda \ln R$, and thus $\Pr[\,\mathrm{T} \geq m\,] \leq \Pr[\,\mathrm{T} \geq 2\mu + \lambda \ln R\,]$. To bound the latter, we consider two cases here: First, assume that $u \geq \lambda R \ln R$. Then,

$$\Pr[\,\mathrm{T} \geq 2\mu + \lambda \ln R\,] \leq \Pr[\,\mathrm{T} \geq 2\mu\,] \leq \exp(-\mu/3) \leq \frac{1}{R}\exp(-\lambda/3) .$$

Second, assume instead that $u \leq \lambda R \ln R$. Then, let $\epsilon = \lambda R \ln R / u$. We have in particular,

$$\frac{\epsilon^2}{2+\epsilon}\mu = \frac{\lambda^2 R (\ln R)^2}{2u + \lambda R \ln R} \geq \frac{1}{3}\lambda \ln R .$$

Therefore, again by the above Chernoff bound,

$$\Pr[\,\mathrm{T} \geq 2\mu + \lambda \ln R\,] \leq \Pr[\,\mathrm{T} \geq \mu(1+\epsilon)\,] \leq \frac{1}{R}\exp(-\lambda/3) .$$

So far, we have computed the probability for a specific and arbitrary $Y \in \mathsf{Out.R}$. The final bound follows by the union bound over all R elements of $\mathsf{Out.R}$. ▌

For the analysis below, we also need to use a lower bound the number of potential preimages of a given output. To this end, given Out: $\mathsf{Out.D} \rightarrow \mathsf{Out.R}$, we define

$$\rho(\mathsf{Out}) = \min_{y \in \mathsf{Out.R}} \left|\mathsf{Out}^{-1}(y)\right| .$$

Security of ideal compression functions. The following theorem establishes the multi-user security under key-leakage of a random compression function. We stress that the bound here does *not* depend on the number of queries the adversary $\mathcal{B}$ makes to oracle Fn. Also, the parameter m can be set arbitrarily in the theorem statement for better flexibility, even though our applications below will mostly use the parameters from Lemma 7.1.

Theorem 7.2 *Let* Out: $\mathcal{K} \rightarrow \mathsf{Out.R}$. *Then, for all* $m \geq 1$*, and all adversaries* $\mathcal{B}$ *making* u *queries to* New*, and* q_{F} *queries to* Prim*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}(\mathcal{B}) \leq \frac{u^2}{2\,|\mathcal{K}|} + \mathsf{P}^{\mathsf{coll}}_{\mathsf{Out}}(u,m) + \frac{(m-1) \cdot q_{\mathrm{F}}}{\rho(\mathsf{Out})} .$$

The statement could be rendered useless whenever $\rho(\mathsf{Out}) = 1$ because a single point has a single pre-image. We note here that Theorem 7.2 can easily be generalized to use a "soft" version of $\rho(\mathsf{Out})$ guaranteeing that the number of preimages of a point is bounded from below by $\rho(\mathsf{Out})$, except with some small probability ϵ, at the cost of an extra additive term $u \cdot \epsilon$. This more general version will not be necessary for our applications. We also note that it is unclear how to use the *average* number of preimages of $\mathsf{Out}(\mathrm{K})$ in our proof.

```
Game $G_0$, $\boxed{G_1}$
$v \leftarrow 0$
$c' \leftarrow\!\!\$\ \mathcal{B}^{\text{New},\text{Fn},\text{Prim}}$
Return $(c' = 1)$

New()
$v \leftarrow v + 1$ ; $K_v \leftarrow\!\!\$\ \mathcal{K}$
Return $\mathsf{Out}(K_v)$

Prim$(k, x)$
if $T_F[k, x] = \bot$ then
   $T_F[k, x] \leftarrow\!\!\$\ \mathcal{K}$
   If $\exists j : k = K_j$ and $T_{\text{Fn}}[j, x] \neq \bot$ then
      $\mathsf{bad}_1 \leftarrow \mathsf{true}$
      $\boxed{T_F[k, x] \leftarrow T_{\text{Fn}}[j, x]}$
Return $T_F[k, x]$

Fn$(i, x)$
If $T_{\text{Fn}}[i, x] = \bot$ then
   $T_{\text{Fn}}[i, x] \leftarrow\!\!\$\ \mathcal{K}$
   If $T_F[K_i, x] \neq \bot$ then
      $\mathsf{bad}_1 \leftarrow \mathsf{true}$
      $\boxed{T_{\text{Fn}}[i, x] \leftarrow T_F[K_i, x]}$
   else if $\exists j \neq i$: $K_j = K_i$
            and $T_{\text{Fn}}[j, x] \neq \bot$ then
      $\mathsf{bad}_2 \leftarrow \mathsf{true}$
      $\boxed{T_{\text{Fn}}[i, x] \leftarrow T_{\text{Fn}}[j, x]}$
Return $T_{\text{Fn}}[i, x]$
```

Figure 5: **Games G_0 and G_1 in the proof of Theorem 7.2.** The boxed assignment statements are only executed in Game G_1, but not in Game G_0.

Proof of Theorem 7.2: The first step of the proof involves two games, G_0 and G_1, given in Fig. 5. Game G_1 is semantically equivalent to $\text{PRF}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}$ with challenge bit $c = 1$, except that we have modified the concrete syntax of the oracles. In particular, the randomly sampled function $\mathsf{f} \leftarrow\!\!\$\ \mathbf{F}$ is now implemented via lazy sampling, and the table entry $T_F[k, x]$ contains the value of $\mathsf{f}(k, x)$ if it has been queried. Otherwise, T_F is $\bot$ on all entries which have not been set. Also, the game keeps another table T_{Fn} such that $T_{\text{Fn}}[i, x]$ contains the value returned upon a query $\text{Fn}(i, x)$. Note that the game enforces that any point in time, if $T_{\text{Fn}}[i, x]$ and $T_F[K_i, x]$ are both set (i.e., they are not equal $\bot$), then we also have $T_{\text{Fn}}[i, x] = T_F[K_i, x]$ and that, moreover, if $K_i = K_j$, then $T_{\text{Fn}}[i, x] = T_{\text{Fn}}[j, x]$ whenever both are not $\bot$. Finally, whenever any of these entries is set for the first time, then it is set to a fresh random value from $\mathcal{K}$. This guarantees that the combined behavior of the Fn and the Prim oracles are the same as in $\text{PRF}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}$ for the case $c = 1$. Thus,

$$\Pr[G_1] = \Pr[\text{PRF}_{\mathsf{CF},\mathsf{Out},\mathbf{F}} \mid c = 1] .$$

It is easier to see that in game G_0, in contrast, the Prim and Fn oracles always return random values, and thus, since we are checking whether c' equals 1, rather than c, we get $\Pr[G_0] = 1 - \Pr[\text{PRF}_{\mathsf{CF},\mathsf{Out},\mathbf{F}} \mid c = 0]$, and consequently,

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}(\mathcal{B}) = \Pr[G_1] - \Pr[G_0] .$$

Both games G_0 and G_1 also include two flags bad_1 and bad_2, initially false, which can be set to true when specific events occur. In particular, bad_1 is set whenever one of the following two events happens: Either $\mathcal{B}$ queries $\text{Fn}(i, x)$ after querying $\text{Prim}(K_i, x)$, or $\mathcal{B}$ queries $\text{Prim}(K_i, x)$ after querying $\text{Fn}(i, x)$. Moreover, bad_2 is set whenever $\mathcal{B}$ queries $\text{Fn}(i, x)$ after $\text{Fn}(j, x)$, $K_i = K_j$, and $\text{Prim}(K_i, x) = \text{Prim}(K_j, x)$ was not queried earlier. (Note that if the latter condition is not true, then bad_1 has been set already.) It is immediate to see that G_0 and G_1 are identical until

```
Game H_0
$v \leftarrow 0$
$c' \leftarrow_{\$} \mathcal{B}^{\text{New},\text{Fn},\text{Prim}}$
Return $(\exists j, x\colon T_{\text{F}}[K_j, x] \neq \bot)$

Game H_1
$v \leftarrow 0$
$c' \leftarrow_{\$} \mathcal{B}^{\text{New},\text{Fn},\text{Prim}}$
for $i = 0$ to $v - 1$ do
    $K'_i \leftarrow_{\$} \{\, k' : \mathsf{Out}(k') = Y_i \,\}$
Return $(\exists j, x\colon T_{\text{F}}[K'_j, x] \neq \bot)$

New()
$v \leftarrow v + 1$ ; $K_v \leftarrow_{\$} \mathcal{K}$ ; $Y_v \leftarrow \mathsf{Out}(K_v)$
Return $Y_v$

Prim(k, x)
if $T_{\text{F}}[k, x] = \bot$ then $T_{\text{F}}[k, x] \leftarrow_{\$} \mathcal{K}$
Return $T_{\text{F}}[k, x]$

Fn(i, x)
If $T_{\text{Fn}}[i, x] = \bot$ then $T_{\text{Fn}}[i, x] \leftarrow_{\$} \mathcal{K}$
Return $T_{\text{Fn}}[i, x]$
```

Figure 6: **Games H_0 and H_1 in the proof of Theorem 7.2.** Both games share the same New, Prim, and Fn oracles, the only difference being the additional re-sampling of the secret keys K'_i in the main procedure of H_1.

$\mathsf{bad}_1 \vee \mathsf{bad}_2$ is set. Therefore, by the fundamental lemma of game playing [5],

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}(\mathcal{B}) = \Pr[\,G_1\,] - \Pr[\,G_0\,] \leq \Pr[\,G_0 \text{ sets } \mathsf{bad}_1\,] + \Pr[\,G_0 \text{ sets } \mathsf{bad}_2\,]\,. \tag{29}$$

We immediately note that in order for bad_2 to be set in G_0, we *must* have $K_i = K_j$ for distinct $i \neq j$, i.e., two keys must collide. Since we know that at most u calls are made to New, a simple Birthday bound yields

$$\Pr[\,G_0 \text{ sets } \mathsf{bad}_2\,] \leq \frac{u^2}{2 \cdot |\mathcal{K}|}\,. \tag{30}$$

The rest of the proof thus deals with the more difficult problem of bounding $\Pr[\,G_0 \text{ sets } \mathsf{bad}_1\,]$. To simplify this task, we first introduce a new game, called H_0 (cf. Fig. 6), which behaves as G_0, except that it only checks at the end of the game whether the bad event triggering bad_1 has occurred during the interaction, in which case the game outputs true. Note that we are relaxing this check a bit further compared with G_0, allowing it to succeed as long as a query to Prim of form (K_j, x) for some j and some x was made, even if $\text{Fn}(j, x)$ was never queried before. Therefore,

$$\Pr[\,G_0 \text{ sets } \mathsf{bad}_1\,] \leq \Pr[\,H_0\,]\,. \tag{31}$$

Note that in H_0, the replies to all oracle calls made by $\mathcal{B}$ do not depend on the keys $K_1, K_2, \ldots$ anymore, *except* for the leaked values $\mathsf{Out}(K_1), \mathsf{Out}(K_2), \ldots$ returned by calls to New. We introduce a new and final game H_1 which modifies H_0 by pushing the sampling of the actual key values as far as possible in the game: That is, we first only gives values to $\mathcal{B}$ with the correct leakage *distribution*, and in the final phase of H_1, when computing the game output, we sample keys that are consistent with this leakage. In other words, in the final check we replace the keys $K_1, K_2, \ldots$ with *freshly* sampled key $K'_1, K'_2, \ldots$, which are uniform, under the condition that $\mathsf{Out}(K_i) = \mathsf{Out}(K'_i) = Y_i$.

It is not hard to see that $\Pr[\,H_0\,] = \Pr[\,H_1\,]$. This follows from two observations: First, for every i, the joint distribution of $(K_i, Y_i = \mathsf{Out}(K_i))$ is identical to that of $(K'_i, Y_i = \mathsf{Out}(K_i))$, since given Y_i, both K_i and K'_i are uniformly distributed over the set of pre-images of Y_i. Second, the behavior of both H_0 and H_1, before the final check to decide their outputs, only depends on values

$Y_i = \mathsf{Out}(K_i)$, and *not* on the K_i's. The actual keys K_i are only used for the final check, and since the probability distributions of K_i and K'_i conditioned on $\mathsf{Out}(Y_i)$ are identical, then so are the probabilities of outputting true in games H_0 and H_1.

Thus, combining Equation (29), Equation (30), and Equation (31), we have

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{Out},\mathbf{F}}(\mathcal{B}) \leq \frac{u^2}{2 \cdot |\mathcal{K}|} + \Pr[\mathrm{H}_1] . \tag{32}$$

We are left with computing an upper bound on $\Pr[\mathrm{H}_1]$. For this purpose, denote by $\mathcal{S}$ the set of pairs (k, x) on which $T_{\mathrm{F}}[k, x] \neq \perp$ after $\mathcal{B}$ outputs its bit c' in H_1. Also, let $\mathcal{Y}$ be the multi-set $\{Y_0, Y_1, \ldots, Y_{u-1}\}$ of values output by NEW to $\mathcal{B}$, and denote $\overline{\mathcal{Y}}$ the resulting set obtained by removing repetitions. Note that $|\mathcal{S}| \leq q_{\mathrm{F}}$ and $|\overline{\mathcal{Y}}| \leq |\mathcal{Y}| \leq u$, and the first inequality may be strict, since some elements can be repeated due to collisions $\mathsf{Out}(K_i) = \mathsf{Out}(K_j)$.

Asume that now $\mathcal{S}$ and $\mathcal{Y}$ are given and fixed. We proceed to compute the probability that H_1 outputs true conditioned on the event that $\mathcal{S}$ and $\mathcal{Y}$ have been generated. For notational help, for every $y \in \overline{\mathcal{Y}}$, also denote

$$\mathcal{S}_y = \{ (k, x) \in \mathcal{S} \, : \, \mathsf{Out}(k) = y \} ,$$

and let $q_y = |\mathcal{S}_y|$. Also, let n_y be the number of occurrence of $y \in \overline{\mathcal{Y}}$ in $\mathcal{Y}$. Note that except with probability $\mathsf{P}^{\mathsf{coll}}(u, m)$, we have $n_y \leq m - 1$ for all $y \in \overline{\mathcal{Y}}$, and thus

$$\begin{aligned} \Pr[\mathrm{H}_1] &\leq \Pr\left[\exists y \in \overline{\mathcal{Y}} \, : \, n_y \geq m\right] + \Pr[\mathrm{H}_1 \,|\, \forall y \in \overline{\mathcal{Y}} \, : \, n_y < m] \\ &= \mathsf{P}^{\mathsf{coll}}_{\mathsf{Out}}(u, m) + \Pr[\mathrm{H}_1 \,|\, \forall y \in \overline{\mathcal{Y}} \, : \, n_y < m] . \end{aligned} \tag{33}$$

Therefore, let us assume we are given $\mathcal{S}$ and $\mathcal{Y}$ sich that $n_y \leq m - 1$ for all $y \in \overline{\mathcal{Y}}$. Denote by $\Pr[\mathrm{H}_1 \,|\, \mathcal{S}, \mathcal{Y}]$ the probability that H_1 outputs true conditioned on the fact that this $\mathcal{S}$ and $\mathcal{Y}$ has been generated. Using the fact that the keys $K'_0, K'_1, \ldots K'_{u-1}$ are sampled independently of $\mathcal{S}$, we compute

$$\begin{aligned} \Pr[\mathrm{H}_1 \,|\, \mathcal{S}, \mathcal{Y}] = \Pr\left[\exists j, x : (K'_j, x) \in \mathcal{S}\right] &\leq \sum_{y \in \mathcal{Y}} \frac{q_y \cdot n_y}{\left|\mathsf{Out}^{-1}(y)\right|} \\ &\leq (m-1) \cdot \sum_{y \in \mathcal{Y}} \frac{q_y}{\left|\mathsf{Out}^{-1}(y)\right|} \leq \frac{m-1}{\rho(\mathsf{Out})} \sum_{y \in \mathcal{Y}} q_y \leq \frac{(m-1) q_{\mathrm{F}}}{\rho(\mathsf{Out})} . \end{aligned}$$

Since the bound holds for all such $\mathcal{S}$ and $\mathcal{Y}$, we also have

$$\Pr[\mathrm{H}_1 \,|\, \forall y \in \overline{\mathcal{Y}} \, : \, n_y < m] \leq \frac{(m-1) q_{\mathrm{F}}}{\rho(\mathsf{Out})} . \tag{34}$$

The final bound follows by combining Equation (32), Equation (33), and Equation (34). ▌

8 Quantitative bounds for augmented cascades and AMAC

We consider two instantiations of augmented cascades, one using bit truncation, the other using modular reduction. We give concrete bounds on the mu prf security of these constructions in the ideal compression function model, combining results from above. This will give us good guidelines for a comparison with existing constructions – such as NMAC and sponges – in Section 9.

BIT TRUNCATION. Let $\mathcal{K} = \{0,1\}^c$, and $\mathsf{Out} = \mathsf{TRUNC}_r : \{0,1\}^c \to \{0,1\}^r$, for $r \leq c$, outputs the first r bits of its inputs, i.e., $\mathsf{TRUNC}_r(X) = X[1..r]$. Note that $\delta(\mathsf{TRUNC}_r) = 0$, since omitting

$c - r$ bits does not affect uniformity, and $\rho(\mathsf{TRUNC}_r) = 2^{c-r}$, since every r-bit strings has 2^{c-r} preimages. Then, combining Lemma 7.1 with Theorem 7.2, using $m = 2u/2^r + 3cr$, we obtain the following corollary, denoting with $\mathbf{F}_c$ the ideal compression function for $\mathcal{K} = \{0,1\}^c$. (We do not specify $\mathcal{X}$ further, as it does not influence the statement.)

Corollary 8.1 *For any $c \leq r$, and all adversaries $\mathcal{B}$ making u queries to* New *and q_{F} queries to* Prim*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{TRUNC}_r,\mathbf{F}_c}(\mathcal{B}) \leq \frac{u^2}{2^{c+1}} + \frac{2u \cdot q_{\mathrm{F}}}{2^c} + \frac{3cr \cdot q_{\mathrm{F}}}{2^{c-r}} + \exp(-c) \ . \ \blacksquare$$

We can then use this result to obtain our bounds for the augmented cascade $\mathbf{ACSC}[\mathsf{CF},\mathsf{CF},\mathsf{TRUNC}_r]$ when using an ideal compression function $\{0,1\}^c \times \mathcal{X} \to \{0,1\}^c$.

Theorem 8.2 (mu prf security for r-bit truncation) *For any $r \leq n$, and all adversaries $\mathcal{A}$ making q queries to* Fn *consisting of vectors from $\mathcal{X}^*$ of length at most ℓ, q_{F} queries to* Prim*, and $u \leq q$ queries to* New*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathbf{ACSC}[\mathsf{CF},\mathsf{CF},\mathsf{TRUNC}_r],\mathbf{F}_c}(\mathcal{A}) \leq \frac{5\ell^2 q^2 + 3\ell q q_{\mathrm{F}}}{2^c} + \frac{3cr\ell \cdot (q\ell + q_{\mathrm{F}})}{2^{c-r}} + \ell \exp(-c) \ \blacksquare$$

Proof of Theorem 8.2: Let A_{c-r} be the set of all functions with domain $\mathcal{X}^*$ and $c-r$-bit outputs, and let A_c be set of all functions with domain $\mathcal{X}^*$ and c-bit outputs. First note that by Theorem 5.3,

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{TRUNC}_r \circ \mathsf{A}_c,\mathsf{TRUNC}_r \circ \mathbf{CSC}[\mathsf{CF},\mathsf{CF}],\mathbf{F}_c}(\mathcal{A})$$
$$\leq \ell \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{TRUNC}_r,\mathbf{F}_c}(\mathcal{A}_{\mathsf{h}}) + 2\,\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathbf{F}_c}(\mathcal{A}_{\mathsf{g}}) \ . \quad (35)$$

Note that $\mathsf{TRUNC}_r \circ \mathsf{A}_c$ and A_{c-r} have the same distribution, and thus

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathbf{ACSC}[\mathsf{CF},\mathsf{CF},\mathsf{TRUNC}_r],\mathbf{F}_c}(\mathcal{A}) = \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{TRUNC}_r \circ \mathsf{A}_c,\mathsf{TRUNC}_r \circ \mathbf{CSC}[\mathsf{CF},\mathsf{CF}],\mathbf{F}_c}(\mathcal{A}) \ . \quad (36)$$

To upper bound the two advantages in Equation (35), recall that adversary $\mathcal{A}_{\mathsf{h}}$ makes at most q queries to its New oracle and at most q queries to its Fn oracle. Adversary $\mathcal{A}_{\mathsf{g}}$ makes u queries to its New oracle and at most q queries to its Fn oracle. The running time of both constructed adversaries is about that of $\mathcal{A}$ plus the time for $q\ell$ computations of CF, which in particular means that both adversaries make $q_{\mathrm{F}} + q \cdot \ell$ queries to Prim. Therefore, by Corollary 8.1,

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{TRUNC}_r,\mathbf{F}_c}(\mathcal{A}_{\mathsf{h}}) \leq \frac{q^2}{2^{c+1}} + \frac{2q \cdot (q\ell + q_{\mathrm{F}})}{2^c} + \frac{3cr \cdot (q\ell + q_{\mathrm{F}})}{2^{c-r}} + \exp(-c) \ , \quad (37)$$

as well as

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{TRUNC}_r,\mathbf{F}_c}(\mathcal{A}_{\mathsf{g}}) \leq \frac{u^2}{2^{c+1}} + \frac{u \cdot (q\ell + q_{\mathrm{F}})}{2^c} \ . \quad (38)$$

The theorem statement then follow by combining Equation (35), Equation (36), and Equation (37). $\blacksquare$

Modular reduction. Our second example becomes particularly important for the application to the Ed25519 signature scheme.

Here, we let $\mathcal{K} = \mathbb{Z}_N$, and consider the output function $\mathsf{Out} = \mathsf{MOD}_M : \mathbb{Z}_N \to \mathbb{Z}_M$ for $M \leq N$ is such that $\mathsf{MOD}_M(X) = X \mod M$. (Note that as a special case, we think of $\mathcal{K} = \{0,1\}^c$ here as $\mathbb{Z}_{2^c}$.) We need the following two properties of MOD_M.

Lemma 8.3 *For all $M \leq N$: (1) $\rho(\mathsf{MOD}_M) \geq \frac{N}{M} - 1$, (2) $\delta(\mathsf{MOD}_M) \leq M/N$.*

Proof of Of Lemma 8.3: For every $a \in \mathbb{Z}_M$, we start by counting the number of $x \in \mathbb{Z}_N$ with $x \mod M = a$. In particular, the $\lfloor N/M \rfloor$ integers $a_i = M \cdot i + a$ for $i \in [0 \ldots \lfloor N/M \rfloor - 1]$ are all those with this property if $a \geq N \mod M$. Otherwise, if $a < N \mod M$, also $a_{\lfloor N/M \rfloor} = M \cdot \lfloor N/M \rfloor + a$ additionally has this property. Thus

$$\rho(\mathsf{MOD}_M) \geq \left\lfloor \frac{N}{M} \right\rfloor \geq \frac{N}{M} - 1 \; .$$

Now, for the statistical distance, $\delta(\mathsf{MOD}_M)$, note that by the above, for all $a \geq N \mod M$, and U_N uniformly distributed on $\mathbb{Z}_N$,

$$\Pr\left[\, \mathsf{MOD}_M(\mathrm{U}_N) = a \,\right] = \frac{1}{N} \left\lfloor \frac{N}{M} \right\rfloor \leq \frac{1}{M} \; ,$$

whereas for all $a < N \mod M$ (if they exist),

$$\Pr\left[\, \mathsf{MOD}_M(\mathrm{U}_N) = a \,\right] = \frac{1}{N} \left\lceil \frac{N}{M} \right\rceil \geq \frac{1}{M} \; .$$

Thus,

$$\begin{aligned}\delta(\mathsf{MOD}_M) &= \sum_{a : a < N \mod M} \left(\Pr\left[\, \mathsf{MOD}_M(\mathrm{U}_N) = a \,\right] - \frac{1}{M} \right) \\ &\leq M \cdot \left(\frac{1}{N} \left\lceil \frac{N}{M} \right\rceil - \frac{1}{M} \right) \leq M \cdot \left(\frac{1}{N} \frac{N}{M} + \frac{1}{N} - \frac{1}{M} \right) \leq \frac{M}{N} \; .\end{aligned}$$

▮

Then, combining Lemma 7.1 and Lemma 8.3 with Theorem 7.2, using $m = 2u/M + 3 \ln N \ln M$, we obtain the following corollary, denoting with $\mathbf{F}_N$ the ideal compression function with $\mathcal{K} = Z_N$. (As above, we do not specify $\mathcal{X}$ further, as it does not influence the statement.)

Corollary 8.4 *For any $M \leq N/2$, and all adversaries $\mathcal{B}$ making u queries to* New *and q_{F} queries to* Prim*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{MOD}_M,\mathbf{F}_N}(\mathcal{B}) \leq \frac{u^2}{2N} + \frac{uM}{N} + \frac{4u \cdot q_{\mathrm{F}}}{N} + \frac{6M \ln N \ln M \cdot q_{\mathrm{F}}}{N} + \frac{1}{N} \; . \; ▮$$

This can once again be used to obtain the final analysis of the augmented cascade using modular reduction. The proof is similar to that of Theorem 8.2.

Theorem 8.5 (mu prf security for modular reduction) *For any $M \leq N/2$, and all adversaries $\mathcal{A}$ making q queries to* Fn *consisting of vectors from $\mathcal{X}^*$ of length at most ℓ, q_{F} queries to* Prim*, and $u \leq q$ queries to* New*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathbf{ACSC}[\mathsf{CF},\mathsf{CF},\mathsf{MOD}_M],\mathbf{F}_N}(\mathcal{A}) \leq \frac{5\ell^2 q^2 + 3\ell q q_{\mathrm{F}}}{N} + \frac{7M \ln N \ln M (\ell^2 q + \ell q_{\mathrm{F}})}{N} + \frac{\ell}{N} \; . \; ▮$$

Proof of Theorem 8.5: Let A_M be the set of all functions with domain $\mathcal{X}^*$ and outputs in $\mathbb{Z}_M$, and let A_N be set of all functions with domain $\mathcal{X}^*$ and outputs in $\mathbb{Z}_N$. First note that by

Theorem 5.3,

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{MOD}_M \circ \mathsf{A}_N, \mathsf{MOD}_M \circ \mathbf{CSC}[\mathsf{CF},\mathsf{CF}], \mathbf{F}_N}(\mathcal{A}) \\ \leq \ell \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{MOD}_M,\mathbf{F}_N}(\mathcal{A}_{\mathsf{h}}) + 2\,\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathbf{F}_N}(\mathcal{A}_{\mathsf{g}}) \ . \quad (39)$$

Note that $\mathsf{MOD}_M \circ \mathsf{A}_N$ and A_M do *not* have the same distribution. Still, by the triangle inequality,

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathbf{ACSC}[\mathsf{CF},\mathsf{CF},\mathsf{MOD}_M],\mathbf{F}_N}(\mathcal{A}) \\ \leq \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{A}_M,\mathsf{MOD}_M \circ \mathsf{A}_N,\mathbf{F}_N}(\mathcal{A}) + \mathsf{Adv}^{\mathsf{dist}}_{\mathsf{MOD}_M \circ \mathsf{A}_N,\mathsf{MOD}_M \circ \mathbf{CSC}[\mathsf{CF},\mathsf{CF}],\mathbf{F}_N}(\mathcal{A}) \ , \quad (40)$$

and Lemma 8.3 directly yields

$$\mathsf{Adv}^{\mathsf{dist}}_{\mathsf{A},\mathsf{MOD}_M \circ \mathsf{A}_N,\mathbf{F}_N}(\mathcal{A}) \leq q \cdot \delta(\mathsf{MOD}_M) \leq q \cdot \frac{M}{N} \ . \quad (41)$$

To upper bound the two advantages in Equation (39), recall that adversary $\mathcal{A}_{\mathsf{h}}$ makes at most q queries to its New oracle and at most q queries to its Fn oracle. Adversary $\mathcal{A}_{\mathsf{g}}$ makes u queries to its New oracle and at most q queries to its Fn oracle. The running time of both constructed adversaries is about that of $\mathcal{A}$ plus the time for $q\ell$ computations of CF, which in particular means that both adversaries make $q_{\mathrm{F}} + q \cdot \ell$ queries to Prim. Therefore, by Corollary 8.4,

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{MOD}_M,\mathbf{F}_N}(\mathcal{A}_{\mathsf{h}}) \\ \leq \frac{q^2}{2N} + \frac{qM}{N} + \frac{4q \cdot (q_{\mathrm{F}} + q \cdot \ell)}{N} + \frac{6Mc \ln M \cdot (q_{\mathrm{F}} + q \cdot \ell)}{N} + \exp(-c) \ . \quad (42)$$

as well as

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{CF},\mathsf{MOD}_N,\mathbf{F}_N}(\mathcal{A}_{\mathsf{g}}) \leq \frac{u^2}{2N} + \frac{u \cdot (q\ell + q_{\mathrm{F}})}{N} \ . \quad (43)$$

The theorem statement then follow by combining all of the above equations. ▌

Bounds for AMAC. The above bounds are for augmented cascades, but they can easily be adapted to AMAC, at the cost of adding an extra additive term, which we now discuss. Recall that $\mathsf{AMAC}(K, M) = \mathsf{Out}(H(K \| M))$, where the iterated hash function H is derived from a compression function h. We only consider here the special case where the key K is completely handled by the first compression function call of H (and is exactly a random element of $\mathcal{X}$), and the message is processed from the second call onwards. In other words, AMAC is the 2-tier cascade with the first tier being the dual of h, meaning the key and input roles are swapped. In particular, we can use Theorem 5.3, which would give us a modified version of the above bounds with an additional additive term, accounting $2\,\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{g}}(\mathcal{A}_{\mathsf{g}})$ for $\mathcal{A}_{\mathsf{g}}$ as given in the reduction. This can easily be upper bounded (using the dedicated mu bound from Fig. 4) as

$$2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{g}}(\mathcal{A}_{\mathsf{g}}) \leq \frac{u^2 + u(q_{\mathrm{F}} + q\ell)}{|\mathcal{X}|} \leq \frac{q^2 + q(q_{\mathrm{F}} + q\ell)}{|\mathcal{X}|} \ .$$

9 Comparisons

We compare the quantitative bounds obtained for augmented cascades above with those from NMAC and sponges. In particular, we show that the security of augmented cascades is comparable to that of NMAC when its output is truncated, and superior to that of keyed sponges.

Comparison with NMAC. We start with concrete bounds for the ideal compression function security of NMAC, for both cases where the output is processed by TRUNC_r or MOD_m. In particular, for a compression function $\mathsf{f} : \mathcal{K} \times \mathcal{X} \to \mathcal{K}$, we consider the construction NMAC such that

$$\mathsf{NMAC}[\mathsf{f}]((K^{\mathsf{in}}, K^{\mathsf{out}}), \mathbf{X}) = \mathsf{f}(K^{\mathsf{out}}, \mathbf{CSC}[\mathsf{f},\mathsf{f}](K^{\mathsf{in}}, \mathbf{X}) \parallel \mathrm{pad}) \ .$$

We are interested in the concrete security of NMAC when f is replaced by CF using the ideal compression function $\mathbf{F}$ as in Section 7. To the best of our knowledge, the following concrete bounds for NMAC have not appeared anywhere, but the statement is somewhat folklore in the single-user case. (For some parameter regime, similar bounds can be inferred from [14]) The proof follows by a fairly standard argument, as sketched below.

Theorem 9.1 (NMAC with r-bit truncation) *For any $r \leq n$, and all adversaries $\mathcal{A}$ making q queries to* Fn *consisting of vectors from $\mathcal{X}^*$ of length at most ℓ, q_{F} queries to* Prim*, and u queries to* New*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{TRUNC}_r \circ \mathsf{NMAC}[\mathsf{CF}], \mathbf{F}_c}(\mathcal{A}) \leq \frac{u^2}{2^{c+1}} + \frac{2\ell^2 q^2}{2^c} + \frac{2q\ell q_{\mathrm{F}}}{2^c} + \frac{2uq_{\mathrm{F}}}{2^c} \ . \ ▮$$

Theorem 9.2 (NMAC with modular truncation) *For any $M \leq N$, and all adversaries $\mathcal{A}$ making q queries to* Fn *consisting of vectors from $\mathcal{X}^*$ of length at most ℓ, q_{F} queries to* Prim*, and u queries to* New*,*

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{MOD}_M \circ \mathsf{NMAC}[\mathsf{CF}], \mathbf{F}_N}(\mathcal{A}) \leq q \cdot \frac{M}{N} + \frac{u^2}{2N} + \frac{2\ell^2 q^2}{N} + \frac{2q\ell q_{\mathrm{F}}}{N} + \frac{2uq_{\mathrm{F}}}{2^c} \ . \ ▮$$

Proof sketch. Let us look at Theorem 9.1 for the case $r = n$ first.

Each of the at most q queries $(u_i, \mathbf{X}_i)$ to Fn makes internally up to $\ell + 1$ queries to $\mathbf{F}_c$. A pair $(u_i, \mathbf{X}_i)$ and $(u_j, \mathbf{X}_j)$ of such Fn queries may be forced to invoke $\mathbf{F}_c$ on some common inputs trivially, in particular when these two queries are for (1) the same user, *and* (2) the corresponding messages share a common prefix (this includes the case that $\mathbf{X}_i$ is itself a prefix of $\mathbf{X}_j$, or vice versa).

A sufficient condition for the outputs of these Fn queries to be random looking is that every internal query (K, X) to $\mathbf{F}_c$ is *unique*, i.e., no other Fn or Prim query results in invoking $\mathbf{F}_c$ on the same input (K, X). The only exception are those unavoidable collisions as above, i.e., the same query (K, X) to $\mathbf{F}_c$ is made when processing another query which needs to evaluate some common queries. It is not hard to show that the probability that this condition is not true is at most

$$\frac{u^2}{2^{c+1}} + \frac{2\ell^2 q^2}{2^c} + \frac{2q\ell q_{\mathrm{F}}}{2^c} + \frac{2uq_{\mathrm{F}}}{2^c} \ ,$$

which is also an upper bound on the advantage. This also implies the general case for $r \leq n$, as truncation does not increase the adversary's advantage. As for modular reduction, one needs to additionally take into account the additional bias of the outputs, which results in an additive $q \cdot \delta(\mathsf{MOD}_M) = q \cdot \frac{M}{N}$ term.

As a side remark, we note that the bound is somewhat generous and not tight, yet improving upon these bounds remains an open problem. A recent paper by Gaži, Pietrzak, and Tessaro [18] improves this bound by considering a modification of NMAC using block whitening, but it is open to verify whether their bound holds for NMAC, too. Still, we note that the main issue with respect to tightness is the growth of the bound with respect to the message length. For short message length, the above bound is essentially tight.

NMAC VS AUGMENTED CASCADES. We note that the bounds of Theorem 8.2 and Theorem 9.1 are similar in many respects, showing that the concrete security for both constructions is comparable. The main difference between the two constructions is the presence of additional terms with denominator 2^{c-r} in Theorem 8.2 which are not present in Theorem 9.1. These have order $q\ell^2/2^{c-r}$ and $q_F\ell/2^{c-r}$ (ignoring small multiplicative factors), respectively. The crucial point here is that *the dependencies on q and q_F are linear.* For example, in the typical case that $c = 2r$ (e.g., $r = 256$), for small ℓ (which is common in many applications) these terms become large roughly when q and q_F approach $2^{c/2}$. In this regime, NMAC is similarly insecure.

SPONGES VS AUGMENTED CASCADES. In contrast to NMAC, the situation is inverted with respect to keyed sponges, which also use truncation as a means to prevent extension attacks and achieve prf security. Indeed, GPT [17] show that when keyed sponges use a c-bit state and output r bits of this state as their output, then there exist attacks with advantage $q^2/2^{c-r}$ and $qq_F/2^{c-r}$, even in the single-user case. For $c = 2r = 512$, for example, keyed sponges can be distinguished with $q = 2^{128}$ queries to FN. In contrast, no terms of such order appear in Theorem 8.2, thus showing that the security of augmented cascades is superior to that of sponges. Needless to say this is not a problem for practical instantiations (like those based on SHA-3), where $c - r$ is generally at least 512, but it shows theoretical gains of augmented cascades over sponges when setting equal parameters.

10 Security of the Davies-Meyer construction

One might object that practical compression functions are not un-structured enough to be treated as random because they are built from blockciphers via the Davies-Meyer construction. Accordingly, we study the mu PRF security under leakage of the Davies-Meyer construction with an ideal blockcipher and show that bounds of the quality we have seen for a random compression function continue to hold. Of course, one could extend this and prove similar bounds for other compression functions, like the PGV [25] ones.

Recall that given a block cipher $E\colon \mathcal{K} \times \mathcal{D} \to \mathcal{D}$, the Davies-Meyer (DM) construction outputs $E_X(H) \oplus H$ for chaining value $H \in \mathcal{D}$ and message block $X \in \mathcal{K}$. Formally, we can model the (keyed) Davies-Meyer construction as an oracle function family DM where $\mathsf{DM.K} = \mathcal{D}$, $\mathsf{DM.D} = \mathcal{K}$, and $\mathsf{DM.R} = \mathcal{D}$. Its oracle space consist of all oracles allowing both forward and backward access to a block cipher, which is the same as that of the ideal-cipher primitive (which we denote as **IC**) defined above.

Theorem 10.1 (prf security of Davies-Meyer) *Let* $\mathsf{Out}\colon \mathcal{D} \to \mathsf{Out.R}$. *Then, for all $m \geq 2$, and all adversaries $\mathcal{B}$ making u queries to* NEW, *q_{FN} queries to* FN, *and q_F queries to* PRIM,

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}}(\mathcal{B}) \leq \frac{u(u + 2q_{\mathrm{FN}})}{2\,|\mathcal{D}|} + \mathsf{P}^{\mathsf{coll}}_{\mathsf{Out}}(u, m) + \frac{(m-1) \cdot q_F}{\rho(\mathsf{Out})} \ .$$

Proof of Theorem 10.1: The proof is similar to that of Theorem 7.2, and we assume that the reader is familiar with its proof, as this will allow for somewhat more compact explanations here. The first step of the proof involves two games, G_0 and G_1, given in Fig. 7.

Game G_1 (where the boxed statements are executed) is semantically equivalent to $\mathrm{PRF}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}}$ with challenge bit $c = 1$, except that we have modified the concrete syntax of the oracles. In particular, the underlying ideal cipher **IC** is implemented via lazy sampling, and the table entries $T_{\mathrm{PRIM}}[K, X, +]$ and $T_{\mathrm{PRIM}}[K, Y, -]$ contain the values $E(K, X)$ and $E^{-1}(K, Y)$, respectively, for the sampled block cipher E, if it has been queried via a PRIM query. (Note that the game always set

```
Game $G_0$,$\boxed{G_1}$
$v \leftarrow 0$
$c' \leftarrow\!\!\$\ \mathcal{B}^{\text{New},\text{Fn},\text{F}}$
Return $(c' = 1)$

Prim$(x, k, +)$
if $T_{\text{Prim}}[x, k, +] = \bot$ then
    $T_{\text{Prim}}[x, k, +] \leftarrow\!\!\$\ \mathcal{D} \setminus T_{\text{Prim}}[x, \cdot, +].\mathsf{R}$
    If $\exists j : k = K_j$ and $T_{\text{Fn}}[j, x] \neq \bot$ then
        $\mathsf{bad}_1 \leftarrow \mathsf{true}$
        $\boxed{T_{\text{Prim}}[x, k, +] \leftarrow T_{\text{F}}[j, x]}$
    $T_{\text{Prim}}[x, T_{\text{Prim}}[x, k, +], -] \leftarrow k$
Return $T_{\text{Prim}}[x, k, +]$

Prim$(x, z, -)$
if $T_{\text{Prim}}[x, z, -] = \bot$ then
    $T_{\text{Prim}}[x, z, -] \leftarrow\!\!\$\ \mathcal{D} \setminus T_{\text{Prim}}[x, \cdot, -].\mathsf{R}$
    If $\exists j : T_{\text{Fn}}[j, x] = z$ then
        $\mathsf{bad}_1 \leftarrow \mathsf{true}$
        $\boxed{T_{\text{Prim}}[x, k, +] \leftarrow K_j}$
    $T_{\text{Prim}}[x, T_{\text{Prim}}[x, z, -], +] \leftarrow z$
Return $T_{\text{Prim}}[k, z, -]$
```

```
New()
$v \leftarrow v + 1$ ; $K_v \leftarrow\!\!\$\ \mathcal{K}$
Return $\mathsf{Out}(K_v)$

Fn$(i, x)$
If $T_{\text{Fn}}[i, x] = \bot$ then
    $T_{\text{Fn}}[i, x] \leftarrow\!\!\$\ \mathcal{D}$
    If $T_{\text{Prim}}[x, K_i, +] \neq \bot$ then
        $\mathsf{bad}_1 \leftarrow \mathsf{true}$
        $\boxed{T_{\text{Fn}}[i, x] \leftarrow T_{\text{Prim}}[x, K_i, +]}$
    else if $\exists j \neq i$: $K_j = K_i$ and $T_{\text{Fn}}[j, x] \neq \bot$ then
        $\mathsf{bad}_2 \leftarrow \mathsf{true}$
        $\boxed{T_{\text{Fn}}[i, x] \leftarrow T_{\text{Fn}}[j, x]}$
    else if $\exists j \neq i$: $T_{\text{Fn}}[j, x] = T_{\text{Fn}}[i, x]$ then
        $\mathsf{bad}_3 \leftarrow \mathsf{true}$
        $\boxed{T_{\text{Fn}}[i, x] \leftarrow\!\!\$\ \mathcal{D} \setminus T_{\text{Fn}}[\cdot, x].\mathsf{R}}$
Return $T_{\text{Fn}}[i, x] \oplus K_i$
```

Figure 7: **Games G_0 and G_1 in the proof of Theorem 10.1.** The boxed statements are only executed in Game G_1, but not in Game G_0. Here, $T_{\text{Prim}}[x, \cdot, +].\mathsf{R}$ is the set of all values z such that there exists a k with $T_{\text{Fn}}[x, k, +] = z$. The notations $T_{\text{Prim}}[x, \cdot, -].\mathsf{R}$ and $T_{\text{Fn}}[\cdot, x].\mathsf{R}$ are defined analogously.

$T_{\text{Prim}}[K, X, +]$ and $T_{\text{Prim}}[K, Y, -]$ jointly in a consistent way.) Otherwise, T_{Prim} is $\bot$ on all entries which have not been set so far.

Also, the game keeps another table T_{Fn} such that $T_{\text{Fn}}[i, x] \oplus K_i$ is the value returned upon a query Fn(i, x). Note that the game enforces that any point in time, if $T_{\text{Fn}}[i, x]$ and $T_{\text{Prim}}[x, K_i, +]$ are both set (i.e., they are not equal $\bot$), then we also have $T_{\text{Fn}}[i, x] = T_{\text{Prim}}[x, K_i, +]$ and that, moreover, if $K_i = K_j$, then $T_{\text{Fn}}[i, x] = T_{\text{Fn}}[j, x]$ whenever both are not $\bot$. Finally, whenever any of these entries is set for the first time, then it is set to a fresh random value from $\mathcal{D}$ constrained on not violating the permutation constraint. This guarantees that the combined behavior of the Fn and the Prim oracles are the same as in $\text{PRF}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}}$ for the case $c = 1$. Thus,

$$\Pr[G_1] = \Pr[\text{PRF}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}} \mid c = 1] \,.$$

It is easier to see that in game G_0, in contrast, the Fn oracles always return random values (the fact that K_i is xored to the $T_{\text{Fn}}[i, x]$ does not modify the distribution), and Prim behaves like an independent ideal cipher. Thus, since we are checking whether c' equals 1, rather than c, we have

$$\Pr[G_0] = 1 - \Pr[\mathrm{PRF}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}} \mid c = 0] .$$

Consequently,

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}}(\mathcal{B}) = \Pr[\mathrm{G}_1] - \Pr[\mathrm{G}_0] .$$

Both games G_0 and G_1 also include three flags bad_1, bad_2, and bad_3, initially false, which can be set to true when specific events occur. It is immediate to see that G_0 and G_1 are identical until $\mathsf{bad}_1 \vee \mathsf{bad}_2 \vee \mathsf{bad}_3$ is true. Therefore, by the fundamental lemma of game playing [5],

$$\begin{aligned}\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{DM},\mathsf{Out},\mathbf{IC}}(\mathcal{B}) &= \Pr[\mathrm{G}_1] - \Pr[\mathrm{G}_0] \\ &\le \Pr[\mathrm{G}_0 \text{ sets } \mathsf{bad}_1] + \Pr[\mathrm{G}_0 \text{ sets } \mathsf{bad}_2] + \Pr[\mathrm{G}_0 \text{ sets } \mathsf{bad}_3] .\end{aligned} \tag{44}$$

As in the proof of Theorem 7.2,

$$\Pr[\mathrm{G}_0 \text{ sets } \mathsf{bad}_2] \le \frac{u^2}{2 \cdot |\mathcal{D}|} . \tag{45}$$

As for bad_3, note that for this to happens, one of the random values generated when answering an $\textsc{Fn}$ query for (i, x) must hit a previously generated value for (j, x), for which there are at most u candidates. Thus, by the union bound,

$$\Pr[\mathrm{G}_0 \text{ sets } \mathsf{bad}_3] \le \frac{q_{\textsc{Fn}} \cdot u}{|\mathcal{D}|} . \tag{46}$$

We are left with the problem of upper bounding $\Pr[\mathrm{G}_0 \text{ sets } \mathsf{bad}_1]$. We note however that this part of the proof can be carried out exactly as in the proof of Theorem 7.2, and results in the identical bound. It is thus omitted. ▌

Acknowledgments

We thank the Eurocrypt 2016 reviewers for their comments.

References

[1] E. Andreeva, J. Daemen, B. Mennink, and G. V. Assche. Security of keyed sponge constructions using a modular proof approach. In G. Leander, editor, *FSE 2015*, volume 9054 of *LNCS*, pages 364–384. Springer, Heidelberg, Mar. 2015. 3, 7

[2] M. Bellare. New proofs for NMAC and HMAC: Security without collision-resistance. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 602–619. Springer, Heidelberg, Aug. 2006. 4, 6, 7, 17

[3] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In N. Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 1–15. Springer, Heidelberg, Aug. 1996. 3, 6, 7

[4] M. Bellare, R. Canetti, and H. Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *37th FOCS*, pages 514–523. IEEE Computer Society Press, Oct. 1996. 3, 5, 7, 10, 11, 12

[5] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006. 8, 24, 32

[6] D. J. Bernstein. Extending the Salsa20 nonce. In *Symmetric key encryption workshop (SKEW)*, February 2011. URL: https://cr.yp.to/papers.html#xsalsa. 11

[7] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang. High-speed high-security signatures. In B. Preneel and T. Takagi, editors, *CHES 2011*, volume 6917 of *LNCS*, pages 124–142. Springer, Heidelberg, Sept. / Oct. 2011. 3, 34

[8] G. Bertoni, J. Daemen, M. Peeters, and G. Assche. On the security of the keyed sponge construction. In *Symmetric key encryption workshop (SKEW)*, February 2011. 3, 7

[9] N. Brown. Things that use Ed25519. `http://ianix.com/pub/ed25519-deployment.html`. 3

[10] D. Chang, M. Dworkin, S. Hong, J. Kelsey, and M. Nandi. A keyed sponge construction with pseudorandomness in the standard model. In *The Third SHA-3 Candidate Conference (March 2012)*, 2012. 3, 7

[11] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård revisited: How to construct a hash function. In V. Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 430–448. Springer, Heidelberg, Aug. 2005. 6, 7

[12] I. Damgård. A design principle for hash functions. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 416–427. Springer, Heidelberg, Aug. 1990. 3

[13] Y. Dodis and K. Pietrzak. Leakage-resilient pseudorandom functions and side-channel attacks on Feistel networks. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 21–40. Springer, Heidelberg, Aug. 2010. 8

[14] Y. Dodis, T. Ristenpart, J. P. Steinberger, and S. Tessaro. To hash or not to hash again? (In)differentiability results for h^2 and HMAC. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 348–366. Springer, Heidelberg, Aug. 2012. 29

[15] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *49th FOCS*, pages 293–302. IEEE Computer Society Press, Oct. 2008. 8

[16] P. Gaži, K. Pietrzak, and M. Rybár. The exact PRF-security of NMAC and HMAC. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 113–130. Springer, Heidelberg, Aug. 2014. 4, 7

[17] P. Gazi, K. Pietrzak, and S. Tessaro. The exact PRF security of truncation: Tight bounds for keyed sponges and truncated CBC. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 368–387. Springer, Heidelberg, Aug. 2015. 3, 7, 30

[18] P. Gazi, K. Pietrzak, and S. Tessaro. Generic security of NMAC and HMAC with input whitening. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 85–109. Springer, Heidelberg, Nov. / Dec. 2015. 29

[19] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, Oct. 1986. 3, 5, 11, 12

[20] U. M. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 21–39. Springer, Heidelberg, Feb. 2004. 7

[21] B. Mennink, R. Reyhanitabar, and D. Vizár. Security of full-state keyed sponge and duplex: Applications to authenticated encryption. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 465–489. Springer, Heidelberg, Nov. / Dec. 2015. 3, 7

[22] R. C. Merkle. One way hash functions and DES. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 428–446. Springer, Heidelberg, Aug. 1990. 3

[23] N. Mouha and A. Luykx. Multi-key security: The Even-Mansour construction revisited. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 209–223. Springer, Heidelberg, Aug. 2015. 5

[24] E. D. Mulder, M. Hutter, M. E. Marson, and P. Pearson. Using Bleichenbacher's solution to the hidden number problem to attack nonce leaks in 384-bit ECDSA. In G. Bertoni and J.-S. Coron, editors, *CHES 2013*, volume 8086 of *LNCS*, pages 435–452. Springer, Heidelberg, Aug. 2013. 8

[25] B. Preneel, R. Govaerts, and J. Vandewalle. Hash functions based on block ciphers: A synthetic approach. In D. R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 368–378. Springer, Heidelberg, Aug. 1994. 7, 30

[26] S. Tessaro. Optimally secure block ciphers from ideal primitives. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 437–462. Springer, Heidelberg, Nov. / Dec. 2015. 5

A Comparing speed of different hash-based MACs

This appendix quantifies the speed advantage of AMAC over HMAC for short messages.

Context. We note at the outset that applications concerned purely with hashing speed should use neither AMAC nor HMAC: non-hash-based MACs are faster. However, non-hash-based MACs cost extra code size, while hash-based MACs have the advantage of reusing hash implementations. Standards use HMAC rather than NMAC for a similar reason: an HMAC implementation can treat the entire hash function as a black box.

SHA-512 speed for long messages. For concreteness we focus on the common Intel Haswell line of CPUs, and we focus on SHA-512 as the underlying hash function. Despite its very high target security level, SHA-512 is the fastest standard hash function on Haswell, running at 8 cycles/byte. For comparison, SHA3-256 uses nearly 9 cycles/byte, SHA-256 uses more than 11 cycles/byte, and SHA3-512 uses more than 16 cycles/byte.

The basic reason that SHA-512 outperforms SHA-256 here is that SHA-512 compression handles 128 message bytes, while SHA-256 compression handles only 64 message bytes. SHA-512 compression has only 25% more operations than SHA-256 compression; each operation inside SHA-512 compression handles 64 bits rather than 32 bits, and most of these 64-bit operations run as quickly as 32-bit operations on these CPUs.

SHA-512 speed for short messages. An "8 cycles/byte" statement for SHA-512 is obtained from observing that, e.g., hashing a 2048-byte message takes $8 \cdot 1024$ cycles more than hashing a 1024-byte message. This underestimates the cost of hashing short messages. Specifically, an n-byte message actually produces $\lceil (n+17)/128 \rceil$ compression-function calls. Benchmarks show a constant per-hash-call overhead of approximately 256 cycles, for a total cost of approximately $256 + 1024\lceil (n+17)/128 \rceil$ cycles.

Speed of SHA-512-based MACs. Both AMAC and HMAC take the same 8 cycles/byte for long messages. We now extrapolate from the observed speed of SHA-512 to predict the AMAC overhead and the HMAC overhead for short messages.

The definition of AMAC includes a computation of Out: e.g., the specific Out in [7] maps a 512-bit integer h to $h \bmod \ell$ for a standard prime $\ell \approx 2^{252}$. We have checked that existing software for this (`sc25519_barrett` in `ed25519/amd64-51`) takes only 100 cycles. More to the point, if HMAC-SHA-512 were used in [7] then one would also want to reduce its 512-bit output modulo ℓ. In other words, the Out overhead exists anyway; AMAC takes advantage of this by integrating Out into the MAC definition. From now on we ignore the cost of Out.

The most obvious advantage of AMAC over HMAC is that it hashes fewer bytes of data. If the key has 32 bytes and the message has n bytes then AMAC hashes $n+32$ bytes, taking approximately

$256 + 1024\lceil(n + 49)/128\rceil$ cycles; e.g., approximately 1280 cycles for $n \leq 79$, and approximately 9472 cycles for $n = 1024$. For HMAC there are two hashing layers:

- The first hashing layer expands the key to a 128-byte block and hashes this block together with the message. If the first compression call is precomputed then this hashes n bytes, taking approximately $256+1024\lceil(n+17)/128\rceil$ cycles. If the hash function is instead called as a black box then this instead hashes $n + 128$ bytes, taking approximately $1280 + 1024\lceil(n + 17)/128\rceil$ cycles.

- The second hashing layer expands the key to another 128-byte block and hashes this block together with the 64-byte output of the first hashing layer. With precomputation this hashes 64 bytes, taking approximately 1280 cycles. Without precomputation this hashes 192 bytes, taking approximately 2304 cycles.

Overall HMAC takes approximately $2560 + 1024\lceil(n + 17)/128\rceil$ cycles if 128 bytes of compression outputs are precomputed, and $4608 + 1024\lceil(n + 17)/128\rceil$ cycles if not; e.g., for $n \leq 111$, approximately 3584 cycles with precomputation, or 5632 cycles without; for $n = 1024$, approximately 11776 cycles with precomputation, or 13824 cycles without.

In other words: The time for AMAC is approximately the time for $0.25 + \lceil(n + 49)/128\rceil$ compression-function calls. The time for HMAC is approximately the time for $2.5 + \lceil(n + 17)/128\rceil$ compression-function calls with precomputation, or $4.5 + \lceil(n + 17)/128\rceil$ compression-function calls without.