Permutation-Based Hashing With Stronger (Second) Preimage Resistance

Siwei Sun[1,3,6], Shun Li[1,2(✉)], Zhiyu Zhang[1], Charlotte Lefevre[4], Bart Mennink[5], Zhen Qin[1], Dengguo Feng[3,1]

[1] School of Cryptology, University of Chinese Academy of Sciences, China
siweisun.isaac@gmail.com, {lishun,zhangzhiyu}@ucas.ac.cn
[2] State Key Laboratory of Cryptography and Digital Economy Security, Shandong University
[3] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China
[4] Univ Rennes, Inria, CNRS, IRISA, Rennes, France
charlotte.lefevre@irisa.fr
[5] Maastricht University, Maastricht, The Netherlands
bart.mennink@maastrichtuniversity.nl
[6] National Engineering Research Center of Multi-dimensional Identification and Trusted Authentication Technology

Abstract. The sponge is a popular construction of hash function design. It operates with a b-bit permutation on a b-bit state, that is split into a c-bit inner part and an r-bit outer part. However, the security bounds of the sponge are most often dominated by the capacity c: if the length of the digest is n bits, the construction tightly achieves $\min\{n/2, c/2\}$-bit collision resistance, $\min\{n, c/2\}$-bit second preimage resistance, and $\min\{n, \max\{n-r, c/2\}\}$-bit preimage resistance. Here, it is noteworthy that the generic attacks matching the preimage and second preimage bounds make use of the inverse of the permutation.

We demonstrate that, by a relatively simple adjustment, significantly improved preimage and second preimage resistance can be achieved. In detail, we first present the SPONGE-DM construction, that differs from the sponge by evaluating the permutation during absorption in a Davies-Meyer mode. This construction generically achieves $\min\{n/2, c/2\}$-bit collision resistance as the sponge does, but n-bit preimage resistance and $\min\{n, c - \log_2(\alpha)\}$-bit second preimage resistance, where α is the maximum size of the first preimage in blocks. Next, we investigate how improved security can be achieved with a smaller feed-forward, and we present the SPONGE-EDM^a family of functions, indexed by a parameter $a \in \{0, \ldots, b\}$. These functions replace the permutation during absorption in the sponge by an Encrypted Davies-Meyer mode, but with only a bits of feed-forward. For $a = b$, comparable bounds as for SPONGE-DM are obtained, and these bounds gradually decrease to the original sponge bounds for decreasing values of a.

We present various instantiations of SPONGE-DM and SPONGE-EDM^a using the Keccak and Ascon permutations, and concretely demonstrate the immediate security and performance gains of these instances. For example, one can achieve up to 512-bit preimage and second preimage resistance

using the 800-bit Keccak permutation (rather than 1600-bit in SHA-3), and likewise, one can use the 1600-bit Keccak permutation to easily achieve up to 1024-bit preimage and second preimage resistance (therewith properly fitting within the recently announced Chinese call for a new generation of cryptographic algorithms). Finally, we show the benefits of using these instantiations in the context of hash-based signature schemes whose security relies solely on the (second) preimage resistance of the underlying hash functions (such as Ascon-Sign).

Keywords: SHA-3, Sponge, (Second) preimage resistance, Cryptographic permutations, Hash-based signatures

1 Introduction

The sponge construction of Bertoni et al. [11] is a popular approach for designing cryptographic hash functions and XoFs (eXtendable output Functions). The sponge construction SPONGE is designed on top of a b-bit permutation f, and operates on a b-bit state, and typically consists of an absorbing phase and a squeezing phase. In the absorbing phase, the state is split into an inner part of c bits (the capacity) and an outer part of r bits (the rate). An arbitrary-length input message msg is first injectively padded into an integral number of blocks of r bits. These blocks are absorbed one by one into the outer part of the state, interleaved with an evaluation of f on the entire state. Then, in the squeezing phase, the state is split into an inner part of c' bits and an outer part of r' bits, and a digest is formed by extracting r' bits from the state, again interleaved with an evaluation of f on the entire state, until the required amount of output bits, n, is retrieved. A visualization of SPONGE is given in Figure 1.

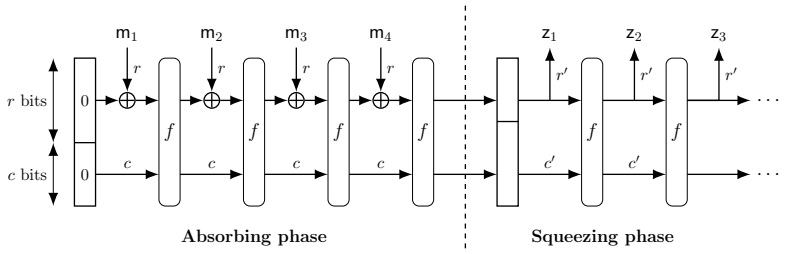


Fig. 1: The sponge construction based on a b-bit permutation f, in case the message m is first injectively padded into four blocks $\mathsf{m}_1\|\mathsf{m}_2\|\mathsf{m}_3\|\mathsf{m}_4$.

Bertoni et al. [12] proved that, if f is a random permutation, the original sponge (in case $c = c'$ and $r = r'$) is indifferentiable from a random oracle [35, 16] as long as the number of evaluations of the permutation does not exceed $2^{c/2}$. Naito and Ohta [36] proved that a similar bound holds in case $r' = r + c/2 - \log_2(c)$ (and with an initial absorption of $r + c/2$ bits). In other words, the sponge behaves

Table 1: The security bounds of SPONGE and our new constructions SPONGE-DM, SPONGE-EDM, and SPONGE-EDM^c. All parameters are expressed in bits. Constants and logarithmic factors in b or n are ignored. The bounds assume that $c' \geq c/2 + \log_2(r)$, and α denotes the maximum message length in blocks. The output size n is assumed to satisfy $n \leq b$, ensuring the existence of a preimage with high probability.

Mode	Output size	Input rate	Output rate	Security			
				Indiff.	Collision	2nd preimage	Preimage
SPONGE	n	r	r'	$\frac{c}{2}$	$\min\{\frac{n}{2}, \frac{c}{2}\}$	$\min\{n, \frac{c}{2}\}$	$\min\{n, \max\{n-r', \frac{c}{2}\}\}$
SPONGE-DM	n	r	r'	$\frac{c}{2}$	$\min\{\frac{n}{2}, \frac{c}{2}\}$	$\min\{n, c - \log_2(\alpha)\}$	n
SPONGE-EDM^b	n	r	r'	$\frac{c}{2}$	$\min\{\frac{n}{2}, \frac{c}{2}\}$	$\min\{n, c - \log_2(\alpha)\}$	n
SPONGE-EDM^a	n	r	r'	$\frac{c}{2}$	$\min\{\frac{n}{2}, \frac{c}{2}\}$	$\min\{n, c - \log_2(\alpha), \max\{a, \min\{\frac{a+c}{2}, \frac{b+c}{3}\}\}\}$	$\min\{n, \max\{n-r', a, \min\{\frac{a+c}{2}, \frac{b+c}{3}\}\}\}$
SPONGE-EDM^0	n	r	r'	$\frac{c}{2}$	$\min\{\frac{n}{2}, \frac{c}{2}\}$	$\min\{n, \frac{c}{2}\}$	$\min\{n, \max\{n-r', \frac{c}{2}\}\}$

like a random oracle up to $2^{c/2}$ queries, and this also implies generic security of the sponge against the classical collision, preimage, and second preimage attacks [2, Appendix A]. Stated differently, the sponge construction truncated to an output of n bits is collision secure up to complexity $\min\{2^{n/2}, 2^{c/2}\}$ and preimage and second preimage secure up to complexity $\min\{2^n, 2^{c/2}\}$. Lefevre and Mennink [29] derived a dedicated security bound for preimage resistance, up to complexity $\min\{2^n, \max\{2^{c/2}, 2^{n-r'}\}\}$. These bounds are also summarized in Table 1, and we refer the readers to Section 3.2 for a more technical treatment of the state-of-the-art security bounds on the sponge.

All of above-mentioned bounds on the sponge (as outlined in Table 1) are *tight*: they match the generic attacks specified in the original introduction of sponge functions [11, Section 5]. For example, for collision resistance, the attack is quite intuitive: first, there is a generic attack that does not take the internal properties of the sponge into account and that succeeds in complexity approximately $2^{n/2}$. Otherwise, one can vary the first message block, and get around $2^{c/2}$ different states after the first evaluation of f. With high probability, there exist two evaluations with the same inner part, and the difference in the outer part can be annihilated with the next message block. For preimage and second preimage resistance, the attacks are slightly more involved, and they rely on the invertibility of f: we revisit these attacks in more detail in Section 3.3, and in this section we also describe a *new* second preimage attack that particularly works if the length of the message is encoded in the padding.

1.1 Instantiating the Sponge

Before proceeding, it is worth observing that existing hash functions based on the sponge have their security parameters based on above bounds. For example, the algorithms in the SHA-3 family are sponge constructions instantiated with a 24-round iterative permutation Keccak-$p[1600, 24] : \mathbb{F}_2^{1600} \to \mathbb{F}_2^{1600}$ [38]. Denoting by Keccak$[c, n]$ the SPONGE construction instantiated with Keccak-$p[1600, 24]$ and

capacity $c = c'$ and output n, then the SHA-3 family consists of 4 hash functions

$$\begin{aligned}
\texttt{SHA-3-224}(\texttt{msg}) &= \texttt{Keccak}[448, 224](\texttt{msg} \parallel \texttt{01}) \,, \\
\texttt{SHA-3-256}(\texttt{msg}) &= \texttt{Keccak}[512, 256](\texttt{msg} \parallel \texttt{01}) \,, \\
\texttt{SHA-3-384}(\texttt{msg}) &= \texttt{Keccak}[768, 384](\texttt{msg} \parallel \texttt{01}) \,, \\
\texttt{SHA-3-512}(\texttt{msg}) &= \texttt{Keccak}[1024, 512](\texttt{msg} \parallel \texttt{01}) \,,
\end{aligned} \tag{1}$$

achieving 112, 128, 192, and 256 bits collision resistance and 224, 256, 384, and 512 bits of (second) preimage resistance, respectively. The SHA-3 family additionally consists of two XoFs

$$\begin{aligned}
\texttt{SHAKE128}(\texttt{msg}, n) &= \texttt{Keccak}[256, n](\texttt{msg} \parallel \texttt{1111}) \,, \\
\texttt{SHAKE256}(\texttt{msg}, n) &= \texttt{Keccak}[512, n](\texttt{msg} \parallel \texttt{1111}) \,,
\end{aligned} \tag{2}$$

which, provided n is large enough, achieve 128 and 256 bits of collision and (second) preimage resistance, respectively.

Likewise, the algorithms in the Ascon family are sponge constructions instantiated with a 12-round iterative permutation $\texttt{Ascon-}p[320, 12] : \mathbb{F}_2^{320} \to \mathbb{F}_2^{320}$ [45]. Let $\texttt{Ascon}[c, n]$ denote the SPONGE construction instantiated with $\texttt{Ascon-}p[320, 12]$ and capacity $c = c'$ and output n, then, according to the NIST standard [45], we have

$$\begin{aligned}
\texttt{Ascon-Hash256}(\texttt{msg}) &= \texttt{Ascon}[256, 256](\texttt{msg}) \,, \\
\texttt{Ascon-XOF128}(\texttt{msg}, n) &= \texttt{Ascon}[256, n](\texttt{msg}) \,, \\
\texttt{Ascon-CXOF128}(\texttt{msg}, n) &= \texttt{Ascon}[256, n](\texttt{msg})
\end{aligned} \tag{3}$$

(where domain separation is applied using the initial value). Also here, the capacity is selected so as to get 128 bits of collision and (second) preimage security, provided n is large enough. See also the systematization of knowledge of Lefevre and Mennink on the generic security of Ascon [31, Section 8].

1.2 The Limits on (Second) Preimage Resistance

Despite the tunability of the sponge, for the case of preimage and second preimage resistance, c *is often the limiting factor*. Indeed, consider the case of $\texttt{SHAKE128}(\texttt{msg}, n)$ of (2). Assuming $n \geq 256$, it generically achieves not only $c/2 = 128$-bit collision but also $c/2 = 128$-bit second preimage resistance (preimage resistance is up to $\max\{n - r, c/2\}$ bits, which also equals $c/2 = 128$ as long as $n \leq 1472$). Likewise, Ascon-Hash256 achieves 128-bit collision resistance as well as 128-bit second preimage resistance. (The NIST standard [45] also claims only 128-bit preimage resistance, but according to the tight bounds of Table 1 it generically achieves 192-bit preimage resistance [31, Section 8].)

Basically, this means that, if one wishes to achieve higher preimage and second preimage resistance, there is no other option than increasing the capacity and typically the permutation size. This has a negative impact on the performance. We do stress that a quest for improved preimage and second preimage resistance

is not just a theoretical exercise. For example, in March 2025, the National Institute of Commercial Cryptography Standards (NICCS) of China released the *Announcement on Launching the Next-generation Commercial Cryptographic Algorithms Program*, where it calls for hash functions with 1024-bit digest and 1024-bit security against preimage and second preimage attacks [37]. In sponge terms, this restriction implies a capacity of at least $1024 \cdot 2 = 2048$ bits, making the 1600-bit Keccak-$p[1600, 24]$ permutation a priori unsuitable. More generally, hash functions with stronger (second) preimage resistance can be applied to enhance the security of hash-based post-quantum signature schemes whose security relies solely on the security of the underlying hash function against (second) preimage attacks. This leads to the natural question of whether we can achieve λ-bit security against preimage and second preimage attacks with only about λ-bit capacity.

1.3 SPONGE-DM and the SPONGE-EDM^a Family

In this work, we investigate the possibility to improve preimage and second preimage resistance of the sponge without increasing the capacity c and permutation size b. Concretely, we propose SPONGE-DM and the SPONGE-EDM^a family for $a \in \{0, \ldots, b\}$, that in a nutshell operate by making the cryptographic operation during absorption non-invertible. The constructions are described in detail in Section 4 and depicted in Figure 4. The idea behind the constructions is loosely inspired by Foekens in her bachelor thesis (supervised by Lefevre and Mennink) [20] who, in turn inspired by the fact that (second) preimage attacks require the inverse of the permutation, investigated whether improved (second) preimage security could be achieved with a random function. However, our approach differs in two aspects. Firstly, we do *not* want to rely on a random function, and secondly, we only manipulate the primitive during absorption, not during squeezing.

The first construction, SPONGE-DM (Section 4.1), is as the sponge function described above, but where the permutations during absorption are replaced with a Davies-Meyer [34] evaluation $x \mapsto f(x) \oplus x$ for the permutation f. This means that SPONGE-DM is exactly as expensive as the sponge function in terms of evaluations of the permutation f, but in exchange it requires b bits of state to facilitate feed-forwards during absorption. We prove in Section 5 that this construction achieves the same levels of indifferentiability and collision resistance as the SPONGE. This result should not come as a surprise, as the corresponding generic attacks on SPONGE do not rely on the invertibility of the permutation. However, next we prove that SPONGE-DM achieves (second) preimage resistance *beyond the bound proven for the sponge*. In detail, we prove that it achieves preimage resistance up to complexity 2^n (i.e., independent of the capacity) and second preimage resistance up to $\min\{2^n, 2^c/\alpha\}$, where α is the length of the first preimage in blocks. The security proof relies on ideas and bad analysis separation originally presented by Lefevre and Mennink [29] on the preimage resistance of the sponge construction, but then adapted to SPONGE-DM.

Next to this construction we consider the possibility to reduce the size of the feed-forward. However, we first demonstrate (in Section 4.2) that only feed-

forwarding the inner part of the state drastically impacts the security and, in fact, diminishes above results on preimage and second preimage resistance. However, to resolve this, we present our second construction, namely SPONGE-EDM^a for $a \in \{0, \ldots, b\}$. This construction replaces the permutations during absorption with the Encrypted Davies-Meyer [15] evaluation $x \mapsto h(g(x) \oplus \lfloor x \rfloor^{(a)})$ for two permutations g, h, and where $\lfloor x \rfloor^{(a)}$ denotes the rightmost a bits of x. As we also prove in Section 5, this construction also achieves the same levels of indifferentiability and collision resistance as the SPONGE. In addition we prove that comparable preimage and second preimage bounds of SPONGE-DM are achieved by SPONGE-EDM^b, i.e., SPONGE-EDM^a for $a = b$, and that these bounds gradually degrade to the original sponge bounds for the case of $a = 0$. The security proof of SPONGE-EDM^a also starts from ideas of Lefevre and Mennink [29], but it becomes surprisingly subtle when the amount of feed-forward bits is less than c (i.e., when $a < c$). For this case, a dedicated bad event partitioning and analysis had to be performed.

These bounds on SPONGE-DM and SPONGE-EDM^a for $a \in \{0, \ldots, b\}$ are summarized in Table 1. These bounds are actually tight: as we explain in Section 5, there are matching attacks for these bounds.

1.4 Instantiation and Application

These are powerful bounds, and they can be used in two directions:

- On the one hand, one can achieve comparable security as existing schemes but with a smaller permutation. For example, one can use an 800-bit permutation in our sponge-like construction SPONGE-DM to achieve 256-bit collision resistance and 512-bit preimage and second preimage resistance (by taking $c = 576$ and $r = 224$ and assuming the maximum size of a message is at most around 2^{64} blocks). This way, the resulting design achieves better generic security than SHAKE256 with its 1600-bit permutation;
- On the other hand, one can achieve higher security with a given permutation. For example, one can take the 1600-bit permutation as in the SHA-3 family, use it in our sponge-like construction SPONGE-DM with $c = 1088$ and $r = 512$, and obtain (assuming the maximum size of a message is at most around 2^{64} blocks) 512-bit collision resistance and 1024-bit preimage and second resistance.

Note that both instantiations do not make extra permutation calls compared to the original sponge: they only make one extra exclusive-or operation and maintain a double state. The second instantiation is particularly interesting in the Chinese *Announcement on Launching the Next-generation Commercial Cryptographic Algorithms Program* [37], that calls for a cryptographic hash function with 512-bit collision resistance and 1024-bit preimage and second preimage resistance: our construction demonstrates that these bounds could be achieved using the NIST standardized Keccak-p[1600, 24] permutation [38]. We remark that this would not have been possible with the original sponge, and that with the – twice as

expensive – double sponge of Lefevre and Mennink [30] this would only be possible with an impractically low rate (i.e., $r = 64$).

We remark that SPONGE-EDM^a is generically more expensive than SPONGE-DM as it makes twice as many permutation calls during absorption. However, this scheme is mostly introduced to support constructions with round-reduced permutations during absorption. For example, taking the 24-round iterative permutation Keccak-$p[1600, 24]$ as f, one can take g to be its first half and h its second half. This way, SPONGE-EDM^c would be *equally efficient* as well as *equally generically secure* as SPONGE-DM but with only c bits of feed-forward. Even stronger, based on the state of the art on the cryptanalysis of the Keccak-$p[1600, 24]$ permutation, we expect these instantiations (with the capacity value set to $512 + 64 = 576$) to enjoy comparative concrete collision, preimage, and second preimage security levels as SHA-3-512 = Keccak$[1024, 512]$ when the length of the longest message allowed to be processed is limited to 2^{64} blocks. This means that we can achieve the same security level with smaller capacity and larger rate, which increases the efficiency of the hash function (especially for long messages). The size of the feed-forward could even be reduced with a gradual decrease in security (as shown in Table 1).

Similar instantiations can be done using the NIST standardized Ascon-$p[320, 12]$ permutation [45], i.e., right into SPONGE-DM. In this case, we expect that these instantiations (with the capacity value set to 256) achieve the same collision resistance security level as Ascon-Hash256 = Ascon$[256, 256]$, as well as 192-bit and 192-bit security against preimage and second preimage attacks, respectively, when the maximum message length is restricted to 2^{64} blocks. In short, a higher level of security could be offered by the SPONGE-DM while maintaining comparable efficiency as the original hash function. We remark that we do not suggest an instantiation of SPONGE-EDM^c using the Ascon-$p[320, 12]$ permutation, the main reason being that splitting Ascon-$p[320, 12]$ into its first and last 6 rounds is unjustifiable given state of the art on the cryptanalysis of this permutation.

The instantiation of SPONGE-DM using Ascon-$p[320, 12]$ has interesting implications on the security and efficiency of hash-based post-quantum signature schemes whose security solely rely on the (second) preimage resistance of the underlying hash functions. For example, by leveraging the instances using the Ascon permutation, we can remedy the flaws of the hash-based signature scheme Ascon-Sign [44] submitted to the NIST PQC project in achieving 192-bit security, and improve the efficiency of the 128-bit secure version of the signature algorithm in Ascon-Sign.

We elaborate on all these and similar instantiations using the Keccak-$p[1600, 24]$ permutation and Ascon-$p[320, 12]$, as well as a discussion on how these instantiations match existing cryptanalysis and on the applications of these instantiations, in Section 6.

Remark 1. Concurrently, Guo et al. [22] demonstrated the benefits of feed-forwarding the inner part to enhance preimage and second preimage security.

1.5 Outline

We start with basic notation and our security model in Section 2. We recall the sponge construction, its state-of-the-art security bounds, and its preimage and second preimage attacks in Section 3. In addition to those attacks, we present in Supplementary Material B our new long message second preimage attack on the sponge. Then, we present our new constructions, `SPONGE-DM` and the `SPONGE-EDM`^{a} family, in Section 4, and derive their security bounds in Section 5. Finally, we present example instantiations of these constructions using the permutations of `Keccak` and `Ascon` in Section 6. In this section, we also demonstrate how our results are particularly useful for hash-based signature schemes.

2 Security Model

For $n \in \mathbb{N}$, we denote by $\mathbb{F}_2^n$ the set of n-bit strings and $\mathbb{F}_2^*$ the set of arbitrarily long strings. The set of arbitrarily long strings whose length is a multiple of n is given by $(\mathbb{F}_2^n)^* = \bigcup_{i \in \mathbb{N}} \mathbb{F}_2^{i \cdot n}$. For an $x \in \mathbb{F}_2^*$, we denote by $\|x\|_{\mathbb{F}_2}$ its length in bits. For $a < \|x\|_{\mathbb{F}_2}$, the leftmost a bits of x are denoted $\lceil x \rceil^{(a)}$ and its rightmost a bits $\lfloor x \rfloor^{(a)}$. For $x \in \mathbb{N}$ such that $x < 2^n$, we denote by $\langle x \rangle_n$ the encoding of x as an n-bit string. We denote by ϵ the empty string.

We denote by $x \xleftarrow{\$} \mathcal{X}$ the uniform random selection of an element x from a finite set $\mathcal{X}$. Let $\mathrm{perm}(b)$ denote the set of all b-bit permutations. For a bijection f, we denote by $f^{\pm}$ that the user has forward and inverse access to the function.

An adversary $\mathcal{A}$ is an algorithm, that is given query access to a set of oracles $\mathcal{O}$, denoted as $\mathcal{A}[\mathcal{O}]$. It can make a certain amount of queries to the oracles, and in the end output something. In our security models, this will either be a decision bit, a collision, or a first or second preimage for the hash function. Without loss of generality, we assume that the adversary never makes a query it already knows the answer to.

Throughout, we will consider security of a XoF $\mathcal{H}$ based on a permutation $f \in \mathrm{perm}(n)$. When we consider generic security, we assume f to be a random permutation, $f \xleftarrow{\$} \mathrm{perm}(n)$, and we typically consider the quality at which $\mathcal{H}$ behaves like a random oracle $\mathcal{R}$, which is a function that outputs a random string of arbitrary length for each new input [6]. We will consider indifferentiability as well as the classical notions of collision resistance, preimage resistance, and second preimage resistance. These notions are described in Sections 2.1 and 2.2, respectively. These descriptions are adopted from those of Lefevre and Mennink [31, Section 8] on `Ascon-XOF`.

2.1 Indifferentiability

We consider indifferentiability of $\mathcal{H}$ based on random permutation $f \xleftarrow{\$} \mathrm{perm}(n)$. Intuitively, a XoF is indifferentiable [35, 16] from a random oracle $\mathcal{R}$ if there exists a simulator $\mathcal{S}$ with oracle access to $\mathcal{R}$ such that $(\mathcal{H}[f], f^{\pm})$ is hard to distinguish from $(\mathcal{R}, \mathcal{S}[\mathcal{R}]^{\pm})$.

Definition 1. *Consider a* XoF $\mathcal{H}$. *Let* $\mathcal{R}$ *be a random oracle and* $f \xleftarrow{\$} \mathrm{perm}(b)$. *Let* $\mathcal{S}[\mathcal{R}]$ *be a simulator with forward and inverse query access. The indifferentiability of* $\mathcal{H}$ *with respect to simulator* $\mathcal{S}^{\mathcal{R}}$ *against an adversary* $\mathcal{A}$ *is defined as*

$$\mathbf{Adv}^{\mathrm{indiff}}_{\mathcal{H},\mathcal{S}}(\mathcal{A}) = \left|\mathbf{Pr}\left(\mathcal{A}\left[\mathcal{H}[f], f^{\pm}\right] \to 1\right) - \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{R}, \mathcal{S}[\mathcal{R}]^{\pm}\right] \to 1\right)\right| .$$

In indifferentiability, the adversarial resources are counted in the number of accumulated permutation evaluations q that would be made in the left world.

2.2 Collision, Preimage, and Second Preimage Resistance

We consider collision, preimage, and second preimage resistance of $\mathcal{H}$ based on random permutation $f \xleftarrow{\$} \mathrm{perm}(n)$, where we take collision resistance, (everywhere) preimage resistance, and (everywhere) second preimage of Rogaway and Shrimpton [43]. Here, we require that the *minimal* output size is fixed to some value n, i.e., $\ell \geq n$. In addition, for second preimage resistance, there is an additional parameter κ that specifies the maximal length of the first preimage.

Definition 2. *Consider a* XoF $\mathcal{H}$. *Let* $f \xleftarrow{\$} \mathrm{perm}(b)$. *Let* $\kappa, n \in \mathbb{N}$.

- *The collision resistance of* $\mathcal{H}$ *against an adversary* $\mathcal{A}$ *is defined as*

$$\mathbf{Adv}^{\mathrm{col}[n]}_{\mathcal{H}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}\left[f^{\pm}\right] \to (\mathsf{msg}, \mathsf{msg}') : \begin{array}{l}\mathsf{msg} \neq \mathsf{msg}' \text{ and} \\ \mathcal{H}[f](\mathsf{msg}, n) = \mathcal{H}[f](\mathsf{msg}', n)\end{array}\right) ;$$

- *The (everywhere) preimage resistance of* $\mathcal{H}$ *against an adversary* $\mathcal{A}$ *is defined as*

$$\mathbf{Adv}^{\mathrm{pre}[n]}_{\mathcal{H}}(\mathcal{A}) = \max_{z \in \mathbb{F}_2^n} \mathbf{Pr}\left(\mathcal{A}\left[f^{\pm}\right](z) \to \mathsf{msg} : \mathcal{H}[f](\mathsf{msg}, n) = z\right) ;$$

- *The (everywhere) second preimage resistance of* $\mathcal{H}$ *against an adversary* $\mathcal{A}$ *is defined as*

$$\mathbf{Adv}^{\mathrm{sec}[\kappa, n]}_{\mathcal{H}}(\mathcal{A}) = \max_{\mathsf{msg} \in \mathbb{F}_2^{\leq \kappa}} \mathbf{Pr}\left(\mathcal{A}\left[f^{\pm}\right](\mathsf{msg}) \to \mathsf{msg}' : \begin{array}{r}\mathsf{msg} \neq \mathsf{msg}' \text{ and} \\ \mathcal{H}[f](\mathsf{msg}, n) = \\ \mathcal{H}[f](\mathsf{msg}', n)\end{array}\right) .$$

In these security notions, the adversarial resources are counted in the number of permutation evaluations q.

We remark that indifferentiability of a XoF from a random oracle implies its collision, preimage, and second preimage resistance up to the bounds that a random oracle would achieve [2, Appendix A] (see also [31, Section 8]). However, dedicated analysis may (and, in the case of our constructions, will) yield tighter bounds.

Algorithm 1: SPONGE

Input: $\mathsf{msg} \in \mathbb{F}_2^*$, $n \in \mathbb{N}$
Output: n-bit digest $\mathsf{z} \in \mathbb{F}_2^n$

$\mathsf{m} \leftarrow \mathsf{msg} \parallel \mathrm{Pad}(r, \|\mathsf{msg}\|_{\mathbb{F}_2})$
$\alpha \leftarrow \|\mathsf{m}\|_{\mathbb{F}_2}/r$
Let $\mathsf{m} = (\mathsf{m}_1, \ldots, \mathsf{m}_\alpha) \in (\mathbb{F}_2^r)^k$
$\mathsf{S} \leftarrow 0^b$
for i from 1 to α **do**
    $\mathsf{S} \leftarrow f(\mathsf{S} \oplus (\mathsf{m}_i \parallel 0^c))$
$\mathsf{z} \leftarrow \mathrm{Trunc}_{r'}(\mathsf{S})$
while $\|\mathsf{z}\|_{\mathbb{F}_2} < n$ **do**
    $\mathsf{S} \leftarrow f(\mathsf{S})$
    $\mathsf{z} \leftarrow \mathsf{z} \parallel \mathrm{Trunc}_{r'}(\mathsf{S})$
return $\mathrm{Trunc}_n(\mathsf{z})$

3 Sponge Construction and Its Security

Before diving into our constructions (in Section 4), we first recall the sponge construction in Section 3.1. We recap its generic security in Section 3.2. In Section 3.3, we recall the existing preimage and second preimage attacks. In addition to those attacks, we present a new long message second preimage attack in Supplementary Material B.

3.1 Construction

Let $b, c, r, c', r' \in \mathbb{N}$ such that $b = r + c = r' + c'$. Let $f : \mathbb{F}_2^b \to \mathbb{F}_2^b$ be a b-bit permutation. The SPONGE function takes as input a message $\mathsf{msg} \in \mathbb{F}_2^*$ and a requested output length $n \in \mathbb{N}$, and outputs a digest $\mathsf{z} \in \mathbb{F}_2^n$ of size n bits. It is defined as follows:

$$\mathtt{SPONGE}(\mathsf{msg}, n) = \mathsf{z}\,.$$

Internally, the function always first injectively pads the message msg by concatenating it with a string $\mathrm{Pad}(r, \|\mathsf{msg}\|_{\mathbb{F}_2})$, in such a way that $\mathsf{m} = \mathsf{msg} \parallel \mathrm{Pad}(r, \|\mathsf{msg}\|_{\mathbb{F}_2})$ is of length a multiple of r bits and the last block is non-zero. We typically write $\alpha = \|\mathsf{m}\|_{\mathbb{F}_2}/r$, and we denote $\beta = \lceil n/r' \rceil$ for the number of r'-bit blocks of z. The sponge construction is depicted in Figure 1 and given in Algorithm 1.

The type of padding string, $\mathrm{Pad}(r, \|\mathsf{msg}\|_{\mathbb{F}_2})$, is dependent on the construction. For example, for SHA-3 [38], the padding is defined as

$$\mathrm{Pad}(r, \mathtt{bitLen}) = 1 \parallel 0^{(r - \mathtt{bitLen} - 2) \bmod r} \parallel 1\,,$$

whereas Ascon [45] uses

$$\mathrm{Pad}(r, \mathtt{bitLen}) = 1 \parallel 0^{(r - \mathtt{bitLen} - 1) \bmod r}\,.$$

One may also encode the length of the original message into the padding, by concatenating $\langle \texttt{bitLen} \rangle_r$.

3.2 Generic Security

The sponge construction is indifferentiable up to bound

$$\mathbf{Adv}^{\text{indiff}}_{\texttt{SPONGE}}(\mathcal{A}) \leq \frac{q(q+1)}{2^c}\,,$$

as proven by Bertoni et al. [12]. Naito and Ohta proved a similar bound if the initial absorption is done at $r + c/2$ bits and squeezing is performed at $r + c/2 - \log_2(c)$ bits [36]. By implication [2, Appendix A], this bound implies collision, preimage, and second preimage resistance:

$$\begin{aligned}
\mathbf{Adv}^{\text{col}[n]}_{\texttt{SPONGE}}(\mathcal{A}) &\leq \frac{q(q+1)}{2^c} + \frac{(q-1)q}{2^{n+1}}\,,\\
\mathbf{Adv}^{\text{pre}[n]}_{\texttt{SPONGE}}(\mathcal{A}) &\leq \frac{q(q+1)}{2^c} + \frac{q}{2^n}\,,\\
\mathbf{Adv}^{\text{sec}[\kappa,\,n]}_{\texttt{SPONGE}}(\mathcal{A}) &\leq \frac{q(q+1)}{2^c} + \frac{q}{2^n}\,.
\end{aligned}$$

(Here, strictly seen, the q in the first term refers to permutation calls but the q in the second term to hash function calls. However, for the sake of the storyline, this slight abuse of notation is not important.) An improved bound on preimage resistance was derived by Lefevre and Mennink [29]:

$$\mathbf{Adv}^{\text{pre}[n]}_{\texttt{SPONGE}}(\mathcal{A}) \leq \min\left\{\frac{4\lceil n/r'\rceil q}{2^{n-r'}}, \frac{q(q+1)}{2^c}\right\} + \frac{4q}{2^n}\,.$$

These bounds are, in fact, tight. Bertoni et al. [11] already derived attacks matching exactly these bounds. The collision attack aims to get an inner collision during absorption by making forward evaluations of f, and this attack also means that the indifferentiability bound is tight. For preimage and second preimage resistance, the attacker makes forward and inverse evaluations of f to meet in the middle.

3.3 Existing Preimage and Second Preimage Attacks

Before describing the SPONGE-DM and SPONGE-EDM^a constructions, we first recall the generic preimage and second preimage attacks on the sponge construction instantiated with a permutation. These two attacks are taken from the original specification of the sponge of Bertoni et al. [11].

3.3.1 Preimage Attack For a given digest $\mathsf{z}_1 \,\|\, \cdots \,\|\, \mathsf{z}_{l-1} \,\|\, \mathsf{z}_l \in \mathbb{F}_2^{r'} \times \cdots \times \mathbb{F}_2^{r'} \times \mathbb{F}_2^{n-(l-1)r'}$, when $n \leq \max\{n - r', \frac{c}{2}\}$, i.e., $n \leq \frac{c}{2}$, we can apply the generic

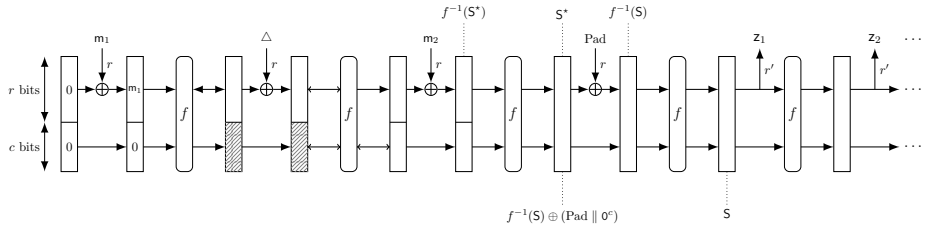


Fig. 2: Generic preimage attack on the sponge construction instantiated with a permutation.

preimage attack on the ideal hash function with n-bit output to find a preimage in time complexity $\mathcal{O}(2^n)$. When $n > \max\{n - r', \frac{c}{2}\}$, i.e., $n > \frac{c}{2}$, a more efficient attack is shown in Figure 2.

First, we try to find an intermediate $b = (r + c)$-bit state S, such that $\lceil \mathsf{S} \rfloor^{(r')} = \mathsf{z}_1$ and

$$\begin{cases} \lceil f(\mathsf{S}) \rfloor^{(r')} = \mathsf{z}_2 \,, \\ \lceil f \circ f(\mathsf{S}) \rfloor^{(r')} = \mathsf{z}_3 \,, \\ \qquad \cdots \quad \cdots \\ \lceil \underbrace{f \circ f \circ \cdots \circ f}_{l-1}(\mathsf{S}) \rfloor^{(n-(l-1)r')} = \mathsf{z}_l \,. \end{cases} \tag{4}$$

For a randomly chosen S with $\lceil \mathsf{S} \rfloor^{(r')} = \mathsf{z}_1$, the probability that (4) is fulfilled can be estimated as

$$\left(\frac{1}{2^{r'}}\right)^{l-2} \cdot \frac{1}{2^{n-(l-1)r'}} = \frac{1}{2^{n-r'}} \,.$$

Therefore, we can identify such a state S with time complexity $2^{n-r'}$. Note that when $n - r' > b - r' = c'$, the time complexity can be reduced to $2^{c'}$, since there are r' bits of $\mathsf{S} \in \mathbb{F}_2^b = \mathbb{F}_2^{r'+c'}$ are fixed, and there are only $b - r' = c'$-bit degrees of freedom. In summary, we can find a satisfied S with a time complexity $\min\{2^{n-r'}, 2^{c'}\}$. After we find S, compute

$$\mathsf{S}^\star = f^{-1}(\mathsf{S}) \oplus (\mathrm{Pad} \parallel 0^c) \,.$$

Then, we try to search for m_1 and m_2 such that

$$\lfloor f(\mathsf{m}_1 \parallel 0^c) \rfloor^{(c)} = \lfloor f^{-1}(f^{-1}(\mathsf{S}^\star) \oplus (\mathsf{m}_2 \parallel 0^c)) \rfloor^{(c)} \,. \tag{5}$$

The complexity of this procedure is about $\mathcal{O}(2^{\frac{c}{2}})$. At this point, we set

$$\Delta = \lceil f(\mathsf{m}_1 \parallel 0^c) \oplus f^{-1}(f^{-1}(\mathsf{S}^\star) \oplus (\mathsf{m}_2 \parallel 0^c)) \rceil^{(r)}.$$

Then, $\mathsf{msg} = \mathsf{m}_1 \parallel \Delta \parallel \mathsf{m}_2$ is a preimage of $\mathsf{z}_1 \parallel \cdots \parallel \mathsf{z}_{l-1} \parallel \mathsf{z}_l$. The overall complexity can be estimated as

$$\min\left\{\max\{\min\{2^{n-r'}, 2^{c'}\}, 2^{\frac{c}{2}}\}, 2^n\right\} \,.$$

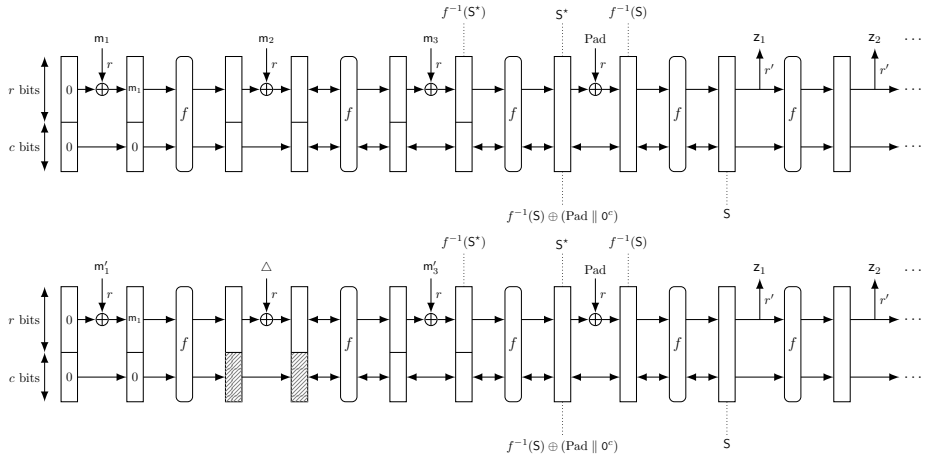


Fig. 3: Generic second preimage attack on the sponge construction instantiated with a permutation.

Remark 2. In Table 1, we listed the security bound proved by Lefevre and Mennink in [29], where it assumes $n \leq b$, under which we always have $2^{n-r'} \leq 2^{b-r'} = 2^{c'}$, and thus

$$\min\left\{\max\{\min\{2^{n-r'}, 2^{c'}\}, 2^{\frac{c}{2}}\}, 2^n\right\} = \min\{\max\{2^{n-r'}, 2^{\frac{c}{2}}\}, 2^n\}\,.$$

3.3.2 Second Preimage Attack Consider a given message $\mathsf{msg} = \mathsf{m}_1 \parallel \mathsf{m}_2 \parallel \mathsf{m}_3 \in \mathbb{F}_2^r \times \mathbb{F}_2^r \times \mathbb{F}_2^r$ and its digest

$$\mathcal{H}(\mathsf{msg}) = \mathsf{z}_1 \parallel \cdots \parallel \mathsf{z}_{l-1} \parallel \mathsf{z}_l \in \mathbb{F}_2^{r'} \times \cdots \times \mathbb{F}_2^{r'} \times \mathbb{F}_2^{n-(l-1)r'}\,.$$

When $n \leq \frac{c}{2}$, we can apply the generic second preimage attack on the ideal hash function with n-bit output to find a second preimage in time complexity $\mathcal{O}(2^n)$. When $n > \frac{c}{2}$, there exists a more efficient attack, which is shown in Figure 3.

First, based on the message msg, we can derive the intermediate state S and compute $\mathsf{S}^\star = f^{-1}(\mathsf{S}) \oplus (\mathsf{Pad} \parallel 0^c)$. Then, we search for m'_1 and m'_3 such that

$$\lfloor f(\mathsf{m}'_1 \parallel 0^c) \rfloor^{(c)} = \lfloor f^{-1}(f^{-1}(\mathsf{S}^\star) \oplus (\mathsf{m}'_3 \parallel 0^c)) \rfloor^{(c)}\,.$$

The complexity of this procedure is about $\mathcal{O}(2^{\frac{c}{2}})$. At this point, we set

$$\Delta = \lceil f(\mathsf{m}'_1 \parallel 0^c) \oplus f^{-1}(f^{-1}(\mathsf{S}^\star) \oplus (\mathsf{m}'_3 \parallel 0^c)) \rceil^{(r)}.$$

Then, $\mathsf{msg} = \mathsf{m}'_1 \parallel \Delta \parallel \mathsf{m}'_3$ is a second preimage of $\mathsf{z}_1 \parallel \cdots \parallel \mathsf{z}_{l-1} \parallel \mathsf{z}_l$.

4 SPONGE-DM and SPONGE-EDM^a for $a \in \{0, \ldots, b\}$

From Section 3.3, we can see that the generic preimage and second preimage attacks require the evaluation of the inverses of the permutations in the absorbing

phase. Therefore, we expect that replacing the permutations in the absorbing phase by random transformations may increase the security bound with respect to preimage and second preimage attacks. In fact, Foekens showed that if the permutations in the sponge are replaced by random transformations $\mathcal{F} : \mathbb{F}_2^b \to \mathbb{F}_2^b$, the security levels of the resulting construction for preimage and second preimage attacks are about n-bit and $\min\{n, c - \log_2(\alpha)\}$-bit respectively, where n is the size of the digest, c is the capacity and $\alpha \cdot r$ is the bit length of the longest message allowed to be processed [20].

Unfortunately, whereas cryptographic permutations are in abundance, random transformations are much more scarce (and the few that exist are typically dedicated designs inspired by existing permutation-to-transformation conversions [5, 1]). Yet, we can build functions $\mathcal{F} : \mathbb{F}_2^b \to \mathbb{F}_2^b$ on top of permutations over $\mathbb{F}_2^b$ that behave sufficiently well for our purpose. In detail, we introduce two functions, namely `SPONGE-DM` and `SPONGE-EDM`^{a} for $a \in \{0, \ldots, b\}$, that are entirely permutation-based and achieve improved preimage and second preimage resistance. Both construction leave the permutation $f : \mathbb{F}_2^b \to \mathbb{F}_2^b$ in the squeezing phase unchanged (see Figure 4c), but replace it in the absorption phase by a one-way function $\mathcal{F} : \mathbb{F}_2^b \to \mathbb{F}_2^b$ that differs for the two constructions.

4.1 `SPONGE-DM`

In `SPONGE-DM`, we instantiate $\mathcal{F}$ as the Davies-Meyer construction [34], or basically f with a feed-forward:

$$\mathcal{F}(x) = f(x) \oplus x\,. \tag{6}$$

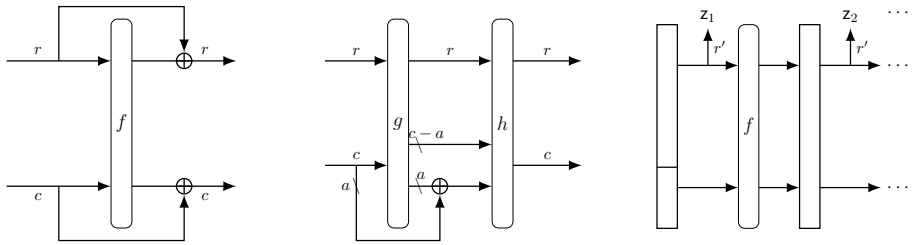
The construction is depicted in Figure 4a.

The idea to place a feed-forward around the permutation is, arguably, the simplest way of adding non-invertibility to the permutation, and as we will prove in Section 5, this, indeed suffices to achieve comparable indifferentiability and collision resistance as the sponge, but to obtain improved preimage and second preimage resistance.

4.2 Towards Reduced Feed-Forward

As a matter of fact, `SPONGE-DM` is as expensive as the original sponge in the number of permutation calls; improved security is only achieved by incurring and using extra b bits of state to perform the feed-forward. In an attempt to reduce the size of the extra state, one may consider feed-forwarding only part of the state: indeed, intuition suggests that, as the adversary has "control" over the outer part of the state, one may only need to feed-forward the inner part.

Denote by `SPONGE-DM`^{c} the construction of Section 4.1 but that only feed-forwards the inner part of the state. For this construction, we present in Supplementary Material C a generic (second) preimage attack, which in some parameter settings cannot outperform brute-force search; for example, when $r > r' = c - n$ and $n < c < 2n$.



(a) Absorption in SPONGE-DM (b) Absorption in SPONGE-EDM^a (c) Squeezing in SPONGE-DM/EDM^a

Fig. 4: The absorption and squeezing functions of SPONGE-DM and SPONGE-EDM^a for $a \in \{0, \ldots, b\}$.

Concluding, SPONGE-DM can be adjusted to work with a significantly reduced feed-forward size, and a generalized variant SPONGE-DM^a, which feed-forwards a bits of the state, may also be considered to explore different trade-offs.

4.3 SPONGE-EDM^a for $a \in \{0, \ldots, b\}$

To this end, we present SPONGE-EDM^a, a family of constructions indexed by a parameter $a \in \{0, \ldots, b\}$. For generality, we define it based on three permutations $f, g, h : \mathbb{F}_2^b \to \mathbb{F}_2^b$, where f is used for squeezing as before, but g and h for absorption (see also our remark below). In SPONGE-EDM^a, we instantiate $\mathcal{F}$ as the Encrypted Davies-Meyer construction [15], *but with only a bits of feed-forward*:

$$\mathcal{F}(x) = h\big(g(x) \oplus \lfloor x \rfloor^{(a)}\big)\,. \tag{7}$$

The construction is depicted in Figure 4b.

The design rationale behind SPONGE-EDM^a is a bit more technical than the one behind SPONGE-DM. Concretely, two major changes stand out, which we elaborate on, below.

The Number of Feed-Forward Bits. Unlike SPONGE-DM, SPONGE-EDM^a is in fact a family of constructions indexed by a, the number of bits that are fed forward. Intuitively, for $a = b$ one should get a similar level of security as SPONGE-DM, and security decreases to that of the original SPONGE for $a = 0$. We prove that this is indeed the case in Section 5. This observation shows that, while the feed-forward state in SPONGE-DM cannot just be reduced for free (as we showed in Section 4.2), it can be reduced at the cost of an extra permutation in absorption calls, and even stronger: the value a forms a true trade-off between (second) preimage security and efficiency. However, the indifferentiability and collision resistance of SPONGE-EDM^a is independent of a, because the generic attacks on the original sponge do not require the inverse of the permutation during absorption.

Choice of Underlying Permutations. Both SPONGE-DM and SPONGE-EDM^a use permutation f for squeezing, but whereas SPONGE-DM also uses f for absorption, for SPONGE-EDM^a we have opted to define absorption using permutations g and h. The rationale behind this is mostly instantiation-based: intuitively, one would not see SPONGE-EDM^a as a variant of SPONGE-DM with an extra permutation during absorption, but rather as one where (i) the permutation f is "split in two halves", as in $f = h \circ g$, and (ii) the feed-forward is inserted halfway. For example, in SPONGE-DM, one may opt to take the full 24-round Keccak-p permutation for f, whereas in SPONGE-EDM^a, permutation g consists of its first 12 rounds and h of its last 12 rounds.

That said, in our security proofs in Section 5, we will regard f, g, and h as random permutations. Even stronger, their difference is not important for the analysis and we will simply assume that $f = g = h$. We refer to [31, Remark 1] for a discussion on independence of primitives in security proofs.

5 Security Analysis

Section 5.1 introduces the preliminaries required for the security analysis. In Section 5.2 we provide bounds on the indifferentiability, collision resistance (by implication), first preimage resistance, and second preimage resistance of our constructions. Proofs of these bounds can be found in Sections 5.3 and 5.4, as well as in Supplementary Material E.

5.1 Preliminaries

For clarity of the presentation, in the proofs we abstract the padding function as $\mathrm{padF} : \mathbb{F}_2^* \to (\mathbb{F}_2^r)^*$ which outputs the complete sequence of r-bit blocks after padding. Its inverse is denoted by upadF and returns $\perp$ if the inputs does not correspond to a valid padding.

A central tool in our security analysis is the concept of multicollisions. We define below the quantity $\mathrm{mcol}(q, 2^r)$, which is the expected maximum load when q balls are thrown uniformly randomly into 2^r bins.

Definition 3. *Let $q, b, r \in \mathbb{N}^*$ such that $r \leq b$. Suppose we sample q elements uniformly with replacement from $\mathbb{F}_2^b$. Let $\mathcal{U}$ denote the resulting multiset of sampled values. Define*

$$\mathrm{mcol}_r\left(\mathcal{U}\right) = \max_{T \in \mathbb{F}_2^r} \left| \left\{ U \in \mathcal{U} \ : \ \lceil U \rceil^{(r)} = T \right\} \right| .$$

We then define

$$\mathrm{mcol}(q, 2^r) = \mathbf{Ex}\left(\mathrm{mcol}_r\left(\mathcal{U}\right)\right) .$$

For convenience, we extend this notation by setting $\mathrm{mcol}(q, 2^s) = q$ for any $s \leq 0$.

The following lemma, introduced by Choi et al. [14] and later slightly improved by Lefevre and Mennink [31], gives an upper bound on $\mathrm{mcol}(q, 2^r)$.

Lemma 1. *Let $q, b, r \in \mathbb{N}^*$ such that $r \leq b$. We have*

$$\mathrm{mcol}(q, 2^r) \leq \frac{2q}{2^r} + 3\ln(2^r) + 4\,.$$

The proof of this lemma can be found in [31, Appendix A].

Remark 3. The bound in Lemma 1 also holds when the q samples in Definition 3 are drawn without replacement from $\mathbb{F}_2^b$.

Whenever possible, the bounds that include multicollisions will be presented in an abstract form, i.e., using the term $\mathrm{mcol}(q, 2^r)$.

5.2 Security Bounds

We first re-establish the indifferentiability of SPONGE-DM and SPONGE-EDM^a for $a \in \{0, \ldots, b\}$ up to $2^{c/2}$ queries.

Theorem 1. *Let $b, c, r, c', r' \in \mathbb{N}$, $a \in \{0, \ldots, b\}$, and $f \xleftarrow{\$} \mathrm{perm}(b)$, and consider $\mathcal{H}$ to be* SPONGE-DM *and $\mathcal{H}'$ to be* SPONGE-EDM^a *of Section 4. Let $\mathcal{R}$ be a random oracle. Let $q \in \mathbb{N}$ be such that $q \leq 2^{b-1}$. There exist simulators $\mathcal{S}$ and $\mathcal{S}'$ such that, for any adversary $\mathcal{A}$ making at most q accumulated queries to $f/\mathcal{S}$ or $f/\mathcal{S}'$,*

$$\mathbf{Adv}^{\mathrm{indiff}}_{\mathcal{H},\mathcal{S}}(\mathcal{A}) \leq \frac{q(3q-1)}{2^c} + \frac{2q \cdot \mathrm{mcol}(q, 2^{r'-r})}{2^{c'}} + \frac{q^2}{2^{b+1}}\,,$$
$$\mathbf{Adv}^{\mathrm{indiff}}_{\mathcal{H}',\mathcal{S}'}(\mathcal{A}) \leq \frac{2q(6q-1)}{2^c} + \frac{2q \cdot \mathrm{mcol}(2q, 2^{r'-r})}{2^{c'}} + \frac{2q^2}{2^b}\,,$$

where $\mathrm{mcol}(\cdot,\cdot)$ *is as in Definition 3.*

The proof is given in Supplementary Material E.1. By implication, this result implies collision resistance of SPONGE-DM and SPONGE-EDM^a up to $\min\{2^{c/2}, 2^{n/2}\}$ queries, as long as $c' \geq c/2 + \log_2(r)$. These bounds are tight as the generic attacks on SPONGE carry over.

Next, we derive an improved preimage resistance bound.

Theorem 2. *Let $b, c, r, c', r', n \in \mathbb{N}$, $a \in \{0, \ldots, b\}$, and $f \xleftarrow{\$} \mathrm{perm}(b)$, and consider $\mathcal{H}$ to be* SPONGE-DM *and $\mathcal{H}'$ to be* SPONGE-EDM^a *of Section 4. Let $q \in \mathbb{N}$ be such that $q \leq 2^{b-1}$, and let $\beta = \lceil n/r' \rceil$. Then, for any adversary $\mathcal{A}$ making at most q queries to f,*

$$\mathbf{Adv}^{\mathrm{pre}[n]}_{\mathcal{H}}(\mathcal{A}) \leq \frac{4q}{2^n}\,,$$
$$\mathbf{Adv}^{\mathrm{pre}[n]}_{\mathcal{H}'}(\mathcal{A}) \leq \frac{16q}{2^n} + \min\left\{ \frac{4\beta q}{2^{n-r'}}, \frac{(6(b-a)+8)q}{2^a}, \frac{24q}{2^c} + \frac{20q^2 \cdot \mathrm{mcol}(q, 2^{b-a})}{2^{c+a}} + \right.$$
$$\frac{2q \cdot \mathrm{mcol}(q, 2^c)}{2^b} + \frac{20q^3 \cdot \mathrm{mcol}(2^a, 2^a)}{2^{b+c}} +$$
$$\left. \frac{4q \cdot \mathrm{mcol}(q, 2^{c-a})}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^a) \cdot \mathrm{mcol}(q, 2^{b-a})}{2^c} \right\},$$

where $\mathrm{mcol}(\cdot,\cdot)$ *is as in Definition 3.*

The proof is given in Section 5.3 for SPONGE-DM, and in Supplementary Material E.2 for SPONGE-EDM^a. The security proof builds upon the preimage resistance analysis of the sponge by Lefevre and Mennink [29], and its refinement to the sponge based on a random function by Foekens [20].

The bound for SPONGE-DM is tight, as the generic attack takes 2^n evaluations.

For SPONGE-EDM^a, the tightness analysis is more subtle. When $a \geq c$, the bound simplifies to

$$\mathbf{Adv}^{\mathrm{pre}[n]}_{\texttt{SPONGE-EDM}^a}(\mathcal{A}) = \tilde{\mathcal{O}}\left(\frac{q}{2^n} + \min\left\{\frac{q}{2^{n-r'}}, \frac{q}{2^a}\right\}\right) .$$

The leftmost term corresponds to a generic preimage attack. Moreover, in Supplementary Material D, we present an attack on SPONGE-EDM^a with complexity $\approx \max\{2^{n-r'}, 2^a\}$ in the regime $a \geq c$. Hence, in this regime the bound is tight.

When $a < c$, the bound becomes

$$\mathbf{Adv}^{\mathrm{pre}[n]}_{\texttt{SPONGE-EDM}^a}(\mathcal{A}) = \tilde{\mathcal{O}}\left(\frac{q}{2^n} + \min\left\{\frac{q}{2^{n-r'}}, \frac{q}{2^a}, \frac{q^2}{2^{a+c}} + \frac{q^3}{2^{b+c}} + \frac{q^2}{2^{b+c-a}}\right\}\right) .$$

Therefore, security is guaranteed up to

$$\approx \min\left\{2^n, \max\left\{2^{n-r'}, 2^a, \min\left\{2^{\frac{c+a}{2}}, 2^{\frac{b+c}{3}}\right\}\right\}\right\}$$

queries. If we further assume $a \leq \frac{b+r}{3}$, the expression can be simplified to

$$\approx \min\left\{2^n, \max\left\{2^{n-r'}, 2^{\frac{c+a}{2}}\right\}\right\} ,$$

and the attack described in Supplementary Material D has complexity $\approx \max\{2^{n-r'}, 2^{\frac{c+a}{2}}\}$. Therefore, in that regime, the bound is tight. In particular, when $a = 0$, we recover a bound of the same order as SPONGE.

The proof of second preimage resistance likewise relies on the analysis of Lefevre and Mennink [29] and Foekens [20].

Theorem 3. *Let $b, c, r, c', r', n \in \mathbb{N}$, $a \in \{0, \ldots, b\}$, and $f \xleftarrow{\$} \mathrm{perm}(b)$, and consider $\mathcal{H}$ to be* SPONGE-DM *and $\mathcal{H}'$ to be* SPONGE-EDM^a *of Section 4. Let $\bar{q} = q + \alpha + \lceil n/r' \rceil - 1$ and $\bar{q}' = q + 2\alpha + \lceil n/r' \rceil - 1$ with $q \in \mathbb{N}$, α the number of r-bit blocks after padding a κ-bit message, and $\bar{q}, \bar{q}' \leq 2^{b-1}$. Then, for any adversary $\mathcal{A}$ making at most q queries to f,*

$$\mathbf{Adv}^{\mathrm{sec}[\kappa, n]}_{\mathcal{H}}(\mathcal{A}) \leq \frac{2\bar{q}}{2^b} + \frac{4\bar{q}}{2^n} + \frac{2\alpha\bar{q}}{2^c} ,$$

$$\begin{aligned}\mathbf{Adv}^{\mathrm{sec}[\kappa, n]}_{\mathcal{H}'}(\mathcal{A}) \leq{}& \frac{4\bar{q}'}{2^b} + \frac{16\bar{q}'}{2^n} + \frac{12\alpha\bar{q}'}{2^c} + \min\left\{\frac{(6(b-a)+8)\bar{q}'}{2^a} + \frac{2\alpha\bar{q}'}{2^a} + \right.\\ &\frac{2\bar{q}' \cdot \mathrm{mcol}(\alpha 2^r, 2^{b-a})}{2^a}, \frac{48\bar{q}'}{2^c} + \frac{40(\bar{q}')^2 \cdot \mathrm{mcol}(\bar{q}', 2^{b-a})}{2^{c+a}} + \\ &\left.\frac{4\bar{q}' \cdot \mathrm{mcol}(\bar{q}', 2^c)}{2^b} + \frac{40(\bar{q}')^3 \cdot \mathrm{mcol}(2^a, 2^a)}{2^{b+c}} + \frac{8\bar{q}' \cdot \mathrm{mcol}(\bar{q}', 2^{c-a})}{2^b} + \right.\end{aligned}$$

$$\frac{8\bar{q}' \cdot \mathrm{mcol}(\bar{q}', 2^a) \cdot \mathrm{mcol}(\bar{q}', 2^{b-a})}{2^c}\Bigg\},$$

where $\mathrm{mcol}(\cdot,\cdot)$ *is as in Definition 3.*

The proof is given in Section 5.4 for SPONGE-EDM, and in Supplementary Material E.3 for SPONGE-EDM^a.

The bound for SPONGE-DM is tight, as the generic attack requires 2^n work, and hitting any absorption state of the first preimage requires $2^c/\alpha$ work. Tightness holds even if the padding function includes the length of the processed message (cf., Supplementary Material B).

Once again, the tightness analysis for SPONGE-EDM^a is more subtle. When $a \geq c$, the bound simplifies to

$$\mathbf{Adv}^{\mathrm{sec}[\kappa,\, n]}_{\texttt{SPONGE-EDM}^a}(\mathcal{A}) = \tilde{\mathcal{O}}\left(\frac{q}{2^n} + \frac{\alpha q}{2^c}\right),$$

and is thus tight, as the attacks for SPONGE-DM can be applied there. In particular, increasing the feed-forward beyond $a = c$ bits provides no additional security with respect to second preimage resistance.

When $a < c$, the bound becomes

$$\mathbf{Adv}^{\mathrm{sec}[\kappa,\, n]}_{\texttt{SPONGE-EDM}^a}(\mathcal{A}) = \tilde{\mathcal{O}}\left(\frac{q}{2^n} + \frac{\alpha q}{2^c} + \min\left\{\frac{q}{2^a}, \frac{q^2}{2^{a+c}} + \frac{q^3}{2^{b+c}} + \frac{q^2}{2^{b+c-a}}\right\}\right).$$

Again, the same aforementioned attack with complexity $\approx \min\{2^n, 2^c/\alpha\}$ applies, and the minimum term can be analyzed in the same way as for preimage resistance. In particular, when $a \leq \frac{b+r}{3}$, security is guaranteed up to

$$\approx \min\left\{2^n, 2^c/\alpha, 2^{\frac{c+a}{2}}\right\}$$

queries. Finally, the attack described in Supplementary Material D applies, but the first step (finding a suitable squeezing state, as in (4)) can be skipped because the attacker is already provided by the first preimage. Hence, this attack has complexity $\approx 2^{\frac{c+a}{2}}$, showing that the bound is tight for those restricted parameter sets.

5.3 Proof of Preimage Resistance for SPONGE-DM (Theorem 2)

Let $\mathsf{z} \in \mathbb{F}_2^n$ be any image, and write $\mathsf{z}_1 \| \cdots \| \mathsf{z}_\beta$, where $\|\mathsf{z}_i\|_{\mathbb{F}_2} = r'$ for $i = 1, \ldots, \beta - 1$ and $\|\mathsf{z}_\beta\|_{\mathbb{F}_2} = n - (\beta - 1)r' \leq r'$. Consider any adversary $\mathcal{A}$ making queries to $f^{\pm}$. These queries are summarized in a transcript $\mathcal{Q}$ as tuples of the form (x, y, dir), where $y = f(x)$ and $\mathrm{dir} \in \{\mathrm{fwd}, \mathrm{inv}\}$ denotes the query direction. The adversary can make a total amount of q queries, and we denote by $\mathcal{Q}_i$ the query transcript up to (and including) tuple i. We assume that $\mathcal{A}$ never repeats any query.

We represent queries that $\mathcal{A}$ makes in a graph. This graph consists of all nodes $\mathbb{F}_2^b$. For each query (x, y, dir), the following edges are added:

– $x \oplus (\mathsf{m} \| 0^c) \xrightarrow{\mathsf{m}} x \oplus y$ for all $\mathsf{m} \in \mathbb{F}_2^r$;
– $x \to y$.

For $i = 1, \ldots, \beta$, we define

$$\boldsymbol{Z}_i := \begin{cases} \{y_i \in \mathbb{F}_2^b \mid \lceil y_i \rceil^{(r')} = \mathsf{z}_i\}\,, & \text{for } i \in \{1, \ldots, \beta - 1\}\,, \\ \{y_i \in \mathbb{F}_2^b \mid \lceil y_i \rceil^{(n-(\beta-1)r')} = \mathsf{z}_i\}\,, & \text{for } i = \beta\,. \end{cases}$$

The goal of the adversary $\mathcal{A}$ is to find a preimage for z, which is implied by the following event:

$$\mathsf{PRE}(\mathcal{Q}) : \mathcal{Q} \text{ defines a path } 0^b \xrightarrow{\mathsf{m}_1} y'_1 \cdots \xrightarrow{\mathsf{m}_{\alpha-1}} y'_{\alpha-1} \xrightarrow{\mathsf{m}_\alpha} y_1 \longrightarrow \cdots \longrightarrow y_\beta$$
$$\text{such that } y_i \in \boldsymbol{Z}_i \text{ for } i = 1, \ldots, \beta\,.$$

Indeed, once $\mathcal{A}$ sets $\mathsf{PRE}(\mathcal{Q})$ with a sequence of message blocks that correspond to a valid padding, the preimage for z can be found as $\mathsf{msg} = \mathrm{upadF}(\mathsf{m}_1 \| \cdots \| \mathsf{m}_\alpha)$.

We say that a state y is a *valid squeezing state* if $\lceil y \rceil^{(r')} = \mathsf{z}_1$, $\lceil f(y) \rceil^{(r')} = \mathsf{z}_2$, and so on, until the last block $\lceil f^{\beta-1}(y) \rceil^{(n-(\beta-1)r')} = \mathsf{z}_\beta$. Formally, we define the set of valid squeezing states:

$$\mathcal{S} = \{y \mid f^{i-1}(y) \in \boldsymbol{Z}_i \text{ for } i = 1, \ldots, \beta\}\,.$$

Clearly, in order to set $\mathsf{PRE}(\mathcal{Q})$, the adversary must *ever* make a query corresponding to an edge $y'_{\alpha-1} \xrightarrow{\mathsf{m}_\alpha} y_1$, for some α, some $y'_{\alpha-1} \in \mathbb{F}_2^b$, and some $y_1 \in \mathcal{S}$. In terms of how these edges are defined, we can observe that $\mathsf{PRE}(\mathcal{Q}) \Rightarrow \mathsf{BAD}(\mathcal{Q})$, where

$$\mathsf{BAD}(\mathcal{Q}) : (x, y, \mathrm{dir}) \in \mathcal{Q} \text{ such that } x \oplus y \in \mathcal{S}\,,$$

and thus that $\mathbf{Pr}\left(\mathsf{PRE}(\mathcal{Q})\right) \leq \mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q})\right)$.

It thus remains to bound $\mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q})\right)$. By basic probability theory

$$\begin{aligned} \mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q})\right) &= \sum_{s=0}^{2^{c'}} \mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q}) \mid |\mathcal{S}| = s\right) \cdot \mathbf{Pr}\left(|\mathcal{S}| = s\right) \\ &\leq \sum_{s=0}^{2^{c'}} \sum_{i=1}^{q} \mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q}_i) \mid |\mathcal{S}| = s \wedge \neg\mathsf{BAD}(\mathcal{Q}_{i-1})\right) \cdot \mathbf{Pr}\left(|\mathcal{S}| = s\right) \\ &\leq \sum_{s=0}^{2^{c'}} \sum_{i=1}^{q} \frac{s}{2^b - (i-1)} \cdot \mathbf{Pr}\left(|\mathcal{S}| = s\right) \\ &\leq \sum_{s=0}^{2^{c'}} \frac{2qs}{2^b} \cdot \mathbf{Pr}\left(|\mathcal{S}| = s\right) = \frac{2q}{2^b} \cdot \mathbf{Ex}\left(|\mathcal{S}|\right)\,, \end{aligned}$$

assuming that $q \leq 2^{b-1}$.

From Lefevre and Mennink [29, Eq. (17), proof of Lemma 2, p. 198], we know that $\mathbf{Ex}\left(|\mathcal{S}|\right) \leq 2\frac{2^b}{2^n}$. Plugging this bound into the above one completes the proof.

5.4 Proof of Second Preimage Resistance for SPONGE-DM (Theorem 3)

Let $\mathsf{msg} \in \mathbb{F}_2^\kappa$ be any preimage, and write $\mathsf{m}_1 \| \cdots \| \mathsf{m}_\alpha = \mathrm{padF}(\mathsf{msg})$. Let $\mathsf{z} = \texttt{SPONGE-DM}(\mathsf{msg})$, and write $\mathsf{z}_1 \| \cdots \| \mathsf{z}_\beta$, where $\|\mathsf{z}_i\|_{\mathbb{F}_2} = r'$ for $i = 1, \ldots, \beta - 1$ and $\|\mathsf{z}_\beta\|_{\mathbb{F}_2} = n - (\beta - 1)r' \leq r'$. Consider any adversary $\mathcal{A}$ making queries to $f^{\pm}$. Here, the adversary is aware of the first preimage msg. The queries corresponding to this evaluation of SPONGE-DM are given to $\mathcal{A}$ for free, prior to the experiment. These queries, along with the adversarial ones, are summarized in a transcript $\mathcal{Q}$ as tuples of the form (x, y, dir), where $y = f(x)$ and $\mathrm{dir} \in \{\mathrm{fwd}, \mathrm{inv}\}$ denotes the query direction. The adversary can make q queries, thus has access to a total of $\bar{q} = q + \alpha + \lceil n/r' \rceil - 1$ tuples in $\mathcal{Q}$. We denote by $\mathcal{Q}_i$ the query transcript up to (and including) tuple i. We assume that $\mathcal{A}$ never repeats any query.

We represent queries that $\mathcal{A}$ makes in a graph. This graph consists of all nodes $\mathbb{F}_2^b$. For each query (x, y, dir), the following edges are added:

- $x \oplus (\mathsf{m} \| 0^c) \overset{\mathsf{m}}{\Longrightarrow} x \oplus y$ for all $\mathsf{m} \in \mathbb{F}_2^r$;
- $x \to y$.

Likewise, for $i = 1, \ldots, \beta$, we define

$$\boldsymbol{Z}_i := \begin{cases} \{y_i \in \mathbb{F}_2^b \mid \lceil y_i \rceil^{(r')} = \mathsf{z}_i\}\,, & \text{for } i \in \{1, \ldots, \beta - 1\}\,, \\ \{y_i \in \mathbb{F}_2^b \mid \lceil y_i \rceil^{(n-(\beta-1)r')} = \mathsf{z}_i\}\,, & \text{for } i = \beta\,. \end{cases}$$

The goal of the adversary $\mathcal{A}$ is to find a second preimage msg, which is implied by the following event:

$$\mathsf{SEC}(\mathcal{Q}) : \mathcal{Q} \text{ defines a path } 0^b \overset{\mathsf{m}'_1}{\Longrightarrow} y'_1 \cdots \overset{\mathsf{m}'_{\alpha-1}}{\Longrightarrow} y'_{\alpha-1} \overset{\mathsf{m}'_\alpha}{\Longrightarrow} y_1 \longrightarrow \cdots \longrightarrow y_\beta$$
$$\text{such that } y_i \in \boldsymbol{Z}_i \text{ for } i = 1, \ldots, \beta \text{ and } \mathrm{padF}(\mathsf{msg}) \neq \mathsf{m}'_1 \| \cdots \| \mathsf{m}'_\alpha\,.$$

Indeed, once $\mathcal{A}$ sets $\mathsf{SEC}(\mathcal{Q})$ with a sequence of message blocks that correspond to a valid padding, the second preimage for msg can be found by unpadding $\mathsf{m}'_1 \| \cdots \| \mathsf{m}'_\alpha$.

As before, we define the set of *valid squeezing states*:

$$\mathcal{S} = \{y \mid f^{i-1}(y) \in \boldsymbol{Z}_i \text{ for } i = 1, \ldots, \beta\}\,.$$

The remainder of the proof, however, will be slightly more involved than the preimage proof, most importantly as the adversary *knows* the first preimage msg. For this message, we denote the states on the path with overlines, as in

$$0^b \overset{\mathsf{m}_1}{\Longrightarrow} \bar{y}'_1 \cdots \overset{\mathsf{m}_{\alpha-1}}{\Longrightarrow} \bar{y}'_{\alpha-1} \overset{\mathsf{m}_\alpha}{\Longrightarrow} \bar{y}_1 \longrightarrow \cdots \longrightarrow \bar{y}_\beta\,.$$

Note that, for this path, we have $\bar{y}_i \in \boldsymbol{Z}_i$ for $i = 1, \ldots, \beta$ basically by definition. In other words, the adversary *knows* $\bar{y}_1 \in \mathcal{S}$. The adversary may succeed in setting $\mathsf{SEC}(\mathcal{Q})$ the way it did for preimage resistance, $\mathsf{BAD}(\mathcal{Q})$ of Section 5.3, but it may also succeed in setting $\mathsf{SEC}(\mathcal{Q})$ if it manages to connect an absorption path from 0^b to any of the states $0^b, \bar{y}'_1, \ldots, \bar{y}'_{\alpha-1}$. However, to this end, the

adversary must *ever* make a query corresponding to an edge $y'_{i-1} \xrightarrow{\mathsf{m}'_i} \bar{y}$, for some i, some $y'_{i-1} \in \mathbb{F}_2^b$, and some $\bar{y} \in \mathbb{F}_2^b$ such that $\lfloor \bar{y} \rfloor^{(c)} \in \{0^c, \lfloor \bar{y}'_1 \rfloor^{(c)}, \ldots, \lfloor \bar{y}'_{\alpha-1} \rfloor^{(c)}\}$. (Indeed, we have to focus on the inner parts, as the adversary can correct the outer part with the next message block.)

More formally, we have the following two bad events:

$$\begin{aligned}
\mathsf{BAD}(\mathcal{Q}) &: (x, y, \mathrm{dir}) \in \mathcal{Q} \text{ such that } (x, x \oplus y) \neq (\bar{y}'_{\alpha-1} \oplus (\mathsf{m}_\alpha \| 0^c), \bar{y}_1) \\
&\qquad \text{and } x \oplus y \in \mathcal{S}\,, \\
\mathsf{CONNECT}(\mathcal{Q}) &: (x, y, \mathrm{dir}) \in \mathcal{Q} \text{ such that either } \lfloor x \oplus y \rfloor^{(c)} = 0^c\,, \\
&\qquad \text{or } \exists A < \alpha \text{ with } (x, x \oplus y) \neq (\bar{y}'_{A-1} \oplus (\mathsf{m}_A \| 0^c), \bar{y}'_A) \\
&\qquad\quad \text{and } \lfloor x \oplus y \rfloor^{(c)} = \lfloor \bar{y}'_A \rfloor^{(c)}\,.
\end{aligned}$$

Note that the bad events have become a bit more technical than the one on preimage resistance, as we have to include trivial wins coming from the hash evaluation of the first preimage. We have that $\mathsf{SEC}(\mathcal{Q}) \Rightarrow \mathsf{BAD}(\mathcal{Q}) \vee \mathsf{CONNECT}(\mathcal{Q})$, and thus that $\mathbf{Pr}\left(\mathsf{SEC}(\mathcal{Q})\right) \leq \mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q})\right) + \mathbf{Pr}\left(\mathsf{CONNECT}(\mathcal{Q})\right)$.

It thus remains to bound $\mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q})\right)$ and $\mathbf{Pr}\left(\mathsf{CONNECT}(\mathcal{Q})\right)$. For $\mathsf{BAD}(\mathcal{Q})$, the analysis of Section 5.3 carries over, but with $\mathbf{Ex}\left(|\mathcal{S}|\right) \leq 2\frac{2^b}{2^n} + 1$.

$$\mathbf{Pr}\left(\mathsf{BAD}(\mathcal{Q})\right) \leq \frac{2\bar{q}}{2^b} + \frac{4\bar{q}}{2^n}\,,$$

assuming that $\bar{q} \leq 2^{b-1}$.

For $\mathsf{CONNECT}(\mathcal{Q})$, we obtain:

$$\begin{aligned}
\mathbf{Pr}\left(\mathsf{CONNECT}(\mathcal{Q})\right) &\leq \sum_{i=1}^{\bar{q}} \mathbf{Pr}\left(\mathsf{CONNECT}(\mathcal{Q}_i) \mid \neg\mathsf{CONNECT}(\mathcal{Q}_{i-1})\right) \\
&\leq \sum_{i=1}^{\bar{q}} \frac{\alpha 2^r}{2^b - (i-1)} \leq \frac{2\alpha\bar{q}}{2^c}\,,
\end{aligned}$$

assuming that $\bar{q} \leq 2^{b-1}$.

6 Instantiations, Comparisons, and Applications

In this section, we present various instantiations of SPONGE-DM and SPONGE-EDM^a using the Keccak [38] and Ascon [45] permutations, in Section 6.1 and Section 6.2, respectively. We demonstrate that these instances outperform their counterparts in SHA-3 and Ascon with comparable security levels for long messages. In addition, we demonstrate that the instantiations using the Keccak permutation perfectly fit the Chinese *Announcement on Launching the Next-generation Commercial Cryptographic Algorithms Program* [37]. Finally, we demonstrate the security and performance benefits of using Ascon-based instantiations of our schemes in

favor of the original Ascon scheme as the underlying hash functions of the post-quantum signature scheme Ascon-Sign submitted to the NIST post-quantum cryptography competition of additional digital signature schemes [44].

In all instantiations, we adopt the padding as discussed in Section 3. All instantiations also have $c = c'$ and $r = r'$. Finally, for both of below sections, on Keccak and Ascon, respectively, the choice of the functions (f, g, h) will be fixed at the start, and the actual instantiations differ in the choice of the capacity c and the output value n. To avoid ambiguity, we will thus consistently append $[c, n]$ to the function names.

6.1 Instantiations Using Keccak Permutation

Let f be the Keccak-p[1600, 24] permutation, g its first 12 rounds, and h its last 12 rounds. We define the following functions:

$$\texttt{Keccak-DM}[c, n] = \texttt{SPONGE-DM}[c, n]\,,$$
$$\texttt{Keccak-EDM}^a[c, n] = \texttt{SPONGE-EDM}^a[c, n]\,.$$

The best known collision and (second-)preimage attacks on variants of the hash functions in SHA-3 penetrate at most 6 rounds [24, 41, 17, 25, 23, 42, 48, 49]. The community has strong confidence in the security margin provided by the 12-round Keccak-p permutation. In fact, the 12-round Keccak-p permutation is employed in TurboSHAKE and KangarooTwelve [10, 13], which are being standardized by IETF [46]. Some researchers even suggest that 10-round Keccak-p is sufficient [3]. Therefore, we believe that using 12-round Keccak-p permutation ensures the security of the Keccak-EDM^c instances listed in Table 2, provided that the maximum admissible message length is restricted to $r \cdot 2^{64}$ bits, as required to meet the bounds on second preimage resistance. Table 2, particularly, demonstrates that Keccak-DM$[c, n]$ and Keccak-EDM$^c[c, n]$ are generically exactly as secure as SHA-3 with respect to collision, second preimage, and preimage resistance, while offering a higher rate and thus being more efficient. These gains come at the use of a larger state, and a thinner, yet sufficient, margin for SPONGE-EDM^c.

For certain lengths of short messages, the efficiency (in the number of permutation round evaluations) of Keccak-DM$[c, n]$ and Keccak-EDM$^c[c, n]$ are almost twice that of SHA-3. For example, for messages with lengths greater than 1084 bits but less than 1276 bits, SHA-3-256 requires two calls to the permutation f, whereas Keccak-DM$[320, 256]$ and Keccak-EDM$^c[320, 256]$ (split over g and h) require only one. This ratio stabilizes for longer messages. For example, in Figure 5, we demonstrate the ratio of the number of internal permutation calls made by SHA-3-n and that of Keccak-EDM$^c[n + 64, n]$. We see that, as the message length increases, Keccak-EDM$^c[320, 256]$ achieves greater efficiency by requiring only about $\frac{1088}{1280} = 85\%$ of the permutation calls needed by SHA-3-256, and Keccak-EDM$^c[576, 512]$ even requires approximately $\frac{576}{1024} \approx 56\%$ of the permutation calls needed by SHA-3-512.

Table 2: Comparison of the efficiency and generic security of instances of SHA-3 with instances of Keccak-DM$[c, n]$ and Keccak-EDM$^c[c, n]$. The state includes the permutation size plus the feed-forward size. All parameters are expressed in bits.

Algorithm	State size	Output size	r	c	Security		
					Coll.	2nd pre.	Pre.
SHA-3-224	1600	224	1152	448	112	224	224
Keccak-DM$[288, 224]$	3200	224	1312	288	112	224	224
Keccak-EDM$^c[288, 224]$	1888	224	1312	288	112	224	224
SHA-3-256	1600	256	1088	512	128	256	256
Keccak-DM$[320, 256]$	3200	256	1280	320	128	256	256
Keccak-EDM$^c[320, 256]$	1920	256	1280	320	128	256	256
SHA-3-384	1600	384	832	768	192	384	384
Keccak-DM$[448, 384]$	3200	384	1152	448	192	384	384
Keccak-EDM$^c[448, 384]$	2048	384	1152	448	192	384	384
SHA-3-512	1600	512	576	1024	256	512	512
Keccak-DM$[576, 512]$	3200	512	1024	576	256	512	512
Keccak-EDM$^c[576, 512]$	2176	512	1024	576	256	512	512
Keccak-DM$[832, 768]$	3200	768	768	832	384	768	768
Keccak-EDM$^c[832, 768]$	2432	768	768	832	384	768	768
Keccak-DM$[1088, 1024]$	3200	1024	512	1088	512	1024	1024
Keccak-EDM$^c[1088, 1024]$	2688	1024	512	1088	512	1024	1024

Table 3: Performance comparison of Keccak-EDM^c and SHA-3. Measurements are in cycles per byte (cpb).

Algorithms	Size of input (bytes)				
	64	256	1024	8192	16384
SHA-3-224	19.88	12.85	12.73	12.00	12.00
Keccak-EDM$^c[288, 224]$	20.78	13.06	12.08	11.44	11.45
Ratio	0.96	0.98	1.05	1.05	1.05
SHA-3-256	19.69	14.91	12.75	12.43	12.37
Keccak-EDM$^c[320, 256]$	20.16	13.12	12.07	11.66	11.60
Ratio	0.98	1.14	1.05	1.07	1.07
SHA-3-384	19.59	16.21	14.41	14.27	14.30
Keccak-EDM$^c[448, 384]$	20.38	13.77	12.94	12.23	12.20
Ratio	0.96	1.18	1.11	1.17	1.17
SHA-3-512	19.94	22.71	18.66	17.97	17.99
Keccak-EDM$^c[576, 512]$	20.41	16.66	13.89	13.09	13.03
Ratio	0.98	1.36	1.34	1.37	1.38

Meeting the Requirements of Chinese Call. We note that Keccak-EDM$^c[1088, 1024]$ fully meets the requirements of the Chinese call for a new generation of cryptographic algorithms expecting hash functions with 1024-bit digests and 1024-bit preimage and second preimage resistance. Concretely, this configuration ensures 512-bit collision resistance, 1024-bit preimage resistance, and second preimage resistance of $1088 - \log_2(\alpha)$, where α is the maximum length of the first preimage in blocks (which we suggest to bound to 2^{64} blocks)). In terms of efficiency,

the proposed scheme introduces no additional permutation calls compared to the standard sponge construction, resulting in negligible computational overhead. Furthermore, permutations with sizes smaller than 1600 bits may also be considered to achieve a more favorable trade-off among memory, security, and efficiency. For instance, embedding an ideal 1280-bit or 1536-bit permutation into SPONGE-EDM^c yields a total state size of 2368 or 2624 bits, respectively, which is more compact than the 2688-bit state of Keccak-EDM$^c[1088, 1024]$. However, while the absorbing rate would be reduced to 192 or 448 bits, dedicated cryptanalysis of such instantiations would become significantly more challenging, as the degree of freedom afforded by the rate part is considerably diminished.

Smaller Permutations. The comparison of Table 2 only focuses on instantiations with the Keccak-$p[1600, 24]$ permutation. Adding the size of the feed-forward, this results in a total implementation state of 3200 bits for Keccak-DM$[c, n]$ and $1600 + a$ bits for Keccak-EDM$^a[c, n]$. However, one can alternatively instantiate our constructions with a smaller permutation, e.g., Keccak-$p[800, 22]$. This instantiation would still allow to achieve even 256-bit indifferentiability and collision security and 512-bit preimage and second preimage resistance, unlike SHA-3-512. Taking the 400-bit Keccak-$p[400, 20]$ even allows to hash with 128-bit collision resistance and 256-bit preimage and second preimage resistance.

Hardware Implementation. Keccak is a hardware-friendly design, and SHA-3 serves as the baseline for comparison. Accordingly, we instantiate both constructions using the 24-round Keccak permutation. When SPONGE-EDM^c employs two permutations whose total number of rounds equals that of the permutation used in SPONGE-DM, SPONGE-EDM^c incurs lower cost due to the use of non-full feed-forward, assuming a similar implementation strategy. In other cases, the optimal choice depends strongly on the specific implementation strategy and the underlying design constraints.

Software Implementation. We implemented the Keccak-$p[1600, 24]$ permutation and the corresponding Keccak-EDM^c function based on the standalone XKCP codebase [9]. To accurately measure the computational overhead, we executed each function 10^6 times on a modern x86-64 platform and recorded the minimum cycle count to eliminate timing noise. The base permutation requires 796 cycles per call, whereas Keccak-EDM^c (with 12-round g and 12-round h) requires 832 cycles, representing an overhead of approximately 5%. This moderate increase confirms that, on modern x86-64 processors with abundant registers and efficient bitwise operations, the additional feed-forward XOR and state management operations introduce only a slight performance penalty relative to the permutation evaluation itself. Despite this overhead, the efficiency gains of our constructions remain significant due to the reduced number of permutation calls achieved by the increased rate r. For example, when processing long messages, Keccak-EDM$^c[320, 256]$ processes 1280-bit rate blocks compared to the 1088-bit blocks of SHA-3-256, yielding approximately 15% fewer primitive evaluations. Even after accounting for the

Table 4: Comparison of the efficiency and generic security of instances of SHA-3-512 with instances of Keccak-EDM^a[576, 512] for various values of a. The state includes the permutation size plus the feed-forward size. All parameters are expressed in bits.

Algorithm	State size	Output size	r	c	Security		
					Coll.	2nd pre.	Pre.
SHA-3-512	1600	512	576	1024	256	512	512
Keccak-EDM^{576}[576, 512]	2176	512	1024	576	256	512	512
Keccak-EDM^{448}[576, 512]	2048	512	1024	576	256	512	512
Keccak-EDM^{256}[576, 512]	1856	512	1024	576	256	416	416
Keccak-EDM^{0}[576, 512]	1600	512	1024	576	256	288	288

5% per-call overhead, the net throughput improvement remains approximately 1.07× (see Table 3).

We experimentally evaluated the performance of our implementations across various message lengths, with results detailed in Table 3. For short messages (e.g., 64 bytes), Keccak-EDM^c exhibits a slight performance degradation compared to standard SHA-3 (ratios 0.96–0.98), attributable to the additional XOR operations and state management inherent in Keccak-EDM^c. However, as message length increases, the throughput improvement stabilizes and aligns with our theoretical analysis: for 512-bit security, Keccak-EDM^c[576, 512] achieves approximately a 1.38× throughput improvement over SHA-3-512 at 16 KB message size, while for 256-bit security, the throughput improvement stabilizes at around 1.07×. These empirical results confirm that the efficiency gains from the increased rate r outweigh the modest 5% per-call overhead observed in our cycle-accurate measurements. We also implemented the 800-bit variant based on Keccak-p[800, 22]. Our measurements indicate that the performance improvement of the 800-bit instantiation over the 1600-bit version can be largely attributed to the reduced number of rounds (22 rounds versus 24 rounds), rather than to the smaller state size. This suggests that the computational cost is dominated by the round transformations rather than by the state width. We note that on resource-constrained embedded platforms (e.g., 8-bit or 32-bit microcontrollers), the relative overhead of state maintenance and XOR operations may be higher than the approximately 5% observed on x86-64 platforms.

Influence of a. In the instantiations above, we focus on Keccak-EDM^c[c, n], i.e., the instantiation of SPONGE-EDM^a for $a = c$ (although setting $a = 2n - c$ with $c = n + \log_2(\alpha)$ can results in a modest reduction in state size—corresponding to a reduction of $2\log_2(\alpha)$ in the feed-forward size—security and efficiency remains equivalent). The reason is that this instantiation achieves the same generic security as the instantiation of SPONGE-DM, and security degrades for degrading a. Yet, SPONGE-EDM^a also becomes more efficient for degrading a, in that it needs less state. To see the effect of a, Table 4 gives a summary of the security of Keccak-EDM^a[576, 512] for different values of a.

Table 5: Comparison of the efficiency and generic security of Ascon-HASH with Ascon-DM and Ascon-DM-128. The state includes the permutation size plus the feed-forward size. All parameters are expressed in bits.

Algorithm	State size	Output size	r	c	Security		
					Coll.	2nd pre.	Pre.
Ascon-HASH	320	256	64	256	128	128	192
Ascon-DM	640	256	64	256	128	192	256
Ascon-DM-128	640	128	128	192	64	128	128

6.2 Instantiations Using Ascon Permutation

Let f be the Ascon-$p[320, 12]$, we define the following functions:

$$\texttt{Ascon-DM} = \texttt{SPONGE-DM}[256, 256]\,,$$
$$\texttt{Ascon-DM-128} = \texttt{SPONGE-DM}[192, 128]\,.$$

A digest length of 128 bits is widely regarded as inadequate for general-purpose hashing. Consequently, Ascon-DM-128 is proposed exclusively for hash-based signature constructions that do not require collision resistance and instead prioritize (second) preimage resistance as the principal security objective.

The best known collision and (second-)preimage attacks on Ascon-HASH and Ascon-XOF penetrate at most 6 rounds [42, 32, 21, 4, 40, 18, 33]. We remark that, unlike for the case of the Keccak permutation in Section 6.1, we did not suggest an instantiation using SPONGE-EDM^a. This choice is mainly supported by the fact that, given these existing cryptanalyses, splitting f into its first and last 6 rounds is unjustifiable. However, for Ascon-DM and Ascon-DM-128, the instantiations seem sound, and it is fair to compare the schemes with Ascon-HASH, which we do in Table 5. Also here, we can conclude that Ascon-DM is generically more secure than Ascon-HASH, while Ascon-DM-128 achieves competitive security as well.

Application to Hash-Based Signature Schemes. We next show how our instantiations can be used to improve the security and efficiency of hash-based signature schemes. For example, there have been several approaches to build post-quantum signature schemes by instantiating the SPHINCS$^+$ framework [8] with the Ascon family of hash functions. Such signature schemes turn out to be more efficient and cost effective than those algorithms in the FIPS 205 standard [39]. For example, the SLH-DSA-SHAKE-128s signature scheme achieves 128-bit security, but so does SPHINCS$^+$ instantiated with Ascon-HASH or with SHAKE, and they achieve this with only a state of 320 or 1600 bits and with a reported efficiency speedup of 16% [19]. We refer to Table 6 for a more detailed performance comparison of Ascon-Sign and SPHINCS$^+$-SHAKE.

However, the security of the SPHINCS$^+$ framework is upper bounded by the security of its underlying hash functions against second preimage attacks. This means that (cf. Section 3.2) if it is instantiated with a sponge function with a capacity of c bits, the hash-based signature scheme cannot be claimed to achieve

Table 6: Comparison of Ascon-Sign and SPHINCS$^+$-SHAKE (from [44] and [26]) in the numbers of megacycles required for key generation, signing, and verification.

Version	Ascon-Sign			SPHINCS$^+$-SHAKE		
	Keygen	Sign	Verify	KeyGen	Sign	Verify
128s-simple	315.84	2413.17	2.43	616.48	4682.57	4.76
128s-robust	554.68	4225.83	5.52	1195.41	8995.48	9.23
128f-simple	5.94	115.38	6.97	9.65	239.79	12.91
128f-robust	10.16	198.14	12.47	18.73	460.76	28.15
192s-simple	599.39	5458.91	4.70	898.36	8091.42	6.47
192s-robust	1046.16	9916.98	10.28	1753.65	15306.01	13.51
192f-simple	10.94	243.02	13.06	14.22	386.86	19.88
192f-robust	18.83	419.87	23.01	27.46	734.07	39.30

Table 7: Performance comparison (measured in cycles/byte) of Ascon-HASH and Ascon-DM-128.

Algorithm	Input size (bytes)				
	64	80	96	104	128
Ascon-HASH	22.5	23.72	21.75	20.64	18.7
Ascon-DM-128	11.8	11.82	11.29	10.47	9.7

beyond $c/2$ bits of security. More concretely, we can consider Ascon-Sign [44], a hash-based signature scheme submitted to the NIST PQC project for additional digital signature schemes. Ascon-Sign is an instantiation of the SPHINCS$^+$ framework with the Ascon family of hash functions. However, combining above observations, it is impossible to achieve 192-bit security as the resistance of Ascon-HASH against second preimage attacks is well below that bound. As a matter of fact, Ascon-Sign did not advance to the second round of competition, and in the NIST official comments for the first-round signatures, Saarinen presented a concrete forgery attack on Ascon-Sign's 192-bit security, which is in essence a second preimage attack on Ascon-HASH with time $\mathcal{O}(2^{128})$. Retrospectively, this could have been easily resolved by instantiating SPHINCS$^+$ with Ascon-DM, because this one achieves 192 bits of second preimage resistance (cf. Table 5).

Improving the efficiency of Ascon-Sign *with 128-bit security.* At the same time, it is possible to improve the performance of Ascon-Sign by using Ascon-DM. For example, by taking Ascon-DM with a $(128 + 64)$-bit capacity, we already achieve 128-bit second preimage resistance, and one does not need to take a 256-bit capacity as in Ascon-HASH. Of course, these results only hold if the length of the longest message is at most 2^{64} blocks, but this is not a problem in SPHINCS$^+$. Indeed, the most computation extensive part comes from the hash function invocations for computing the non-leaf nodes of the underlying XMSS trees. For Ascon-Sign-128, the function $\mathbf{H}(\mathsf{PK.seed}, \mathsf{ADRS}, \mathfrak{n}_{\text{Left}} \parallel \mathfrak{n}_{\text{Right}})$ used to compute the non-leaf nodes of the XMSS trees is defined as

$$\text{Trunc}_{128}(\texttt{Ascon-HASH}(\mathsf{PK.seed} \parallel \mathsf{ADRS} \parallel \mathfrak{n}_{\text{Left}} \parallel \mathfrak{n}_{\text{Right}}))\,,$$

where PK.seed is a 16-byte public seed, ADRS is a 32-byte structure indicating the relative position of the computed node, and $\mathfrak{n}_{\text{Left}}$ and $\mathfrak{n}_{\text{Right}}$ are two 16-byte nodes of an XMSS tree. Therefore the size of the input to Ascon-HASH is 640 bits. Taking the padding scheme into account, it takes $\frac{640+64}{64} = 11$ evaluations of Ascon-p[12] to compute one non-leaf XMSS tree node. If we use Ascon-DM-128 instead, we can shrink the capacity to $(128 + 64 = 192)$-bit, resulting in a $(320 - 192 = 128)$-bit rate. Consequently, we only need $\lceil \frac{640+128}{128} \rceil = 6$ evaluations of the Ascon-p[12] permutation. To further demonstrate the advantage of using Ascon-DM-128 in this scenario, we perform more experiments measuring the software speed of Ascon-HASH and Ascon-DM-128 for several input lengths used in Ascon-Sign and the results are listed in Table 7.

7 Conclusion and Open Questions

In this work, we presented the SPONGE-DM and SPONGE-EDM^a constructions and derived provable security bounds in the ideal permutation model. The proposed constructions achieve comparable or higher security with smaller permutations, giving performance and cost improvements over standard hash functions targeting 192-, 256-, and 512-bit security. These advantages are particularly valuable in applications like hash-based signature schemes, where enhanced second-preimage resistance directly bolsters security.

While these results advance the state of the art, several open questions remain for further investigation. First, we presented additional variants, namely SPONGE-DM^c and SPONGE-DM^a, whose theoretical security proofs are more intricate yet warrant further exploration. Second, it would be of significant interest to assess the security of SPONGE-DM, SPONGE-EDM^a, SPONGE-DM^c, and SPONGE-DM^a when instantiated with specific permutations, such as Keccak or Ascon, under concrete adversarial strategies including differential and linear cryptanalysis, and algebraic attacks, as such analysis is essential to provide concrete security assurances in practice.

Acknowledgements. This work is supported by the National Cryptologic Science Fund of China (2025NCSF01012), the National Natural Science Foundation of China (62532013, 62502494), and the Fundamental Research Funds for the Central Universities and the Scientific Research Innovation Capability Support Project for Young Faculty. Shun Li is supported by by State Key Laboratory of Cryptography and Digital Economy Security, Shandong University (No. KFYB2503). Charlotte Lefevre is supported by the Netherlands Organisation for Scientific Research (NWO) under grant OCENW.KLEIN.435 and by the France 2030 program under grant agreement No. ANR-22-PECY-0010. Bart Mennink is supported by the Netherlands Organisation for Scientific Research (NWO) under grant VI.Vidi.203.099.

References

1. Anand, R., Banik, S., Caforio, A., Ishikawa, T., Isobe, T., Liu, F., Minematsu, K., Rahman, M., Sakamoto, K.: Gleeok: A Family of Low-Latency PRFs and its Applications to Authenticated Encryption. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2024**(2), 545–587 (2024). https://doi.org/10.46586/TCHES.V2024.I2.545-587
2. Andreeva, E., Mennink, B., Preneel, B.: Security Reductions of the Second Round SHA-3 Candidates. In: Burmester, M., Tsudik, G., Magliveras, S.S., Ilic, I. (eds.) Information Security - 13th International Conference, ISC 2010, Boca Raton, FL, USA, October 25-28, 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6531, pp. 39–53. Springer (2010). https://doi.org/10.1007/978-3-642-18178-8_5
3. Aumasson, J.P.: Too Much Crypto. Cryptology ePrint Archive, Paper 2019/1492 (2019), https://eprint.iacr.org/2019/1492
4. Baek, S., Kim, G., Kim, J.: Preimage attacks on reduced-round Ascon-Xof. Des. Codes Cryptogr. **92**(8), 2197–2217 (2024). https://doi.org/10.1007/S10623-024-01383-0
5. Banik, S., Isobe, T., Liu, F., Minematsu, K., Sakamoto, K.: Orthros: A Low-Latency PRF. IACR Trans. Symmetric Cryptol. **2021**(1), 37–77 (2021). https://doi.org/10.46586/TOSC.V2021.I1.37-77
6. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993. pp. 62–73. ACM (1993). https://doi.org/10.1145/168588.168596
7. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4004, pp. 409–426. Springer (2006). https://doi.org/10.1007/11761679_25
8. Bernstein, D.J., Hülsing, A., Kölbl, S., Niederhagen, R., Rijneveld, J., Schwabe, P.: The SPHINCS$^+$ Signature Framework. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019. pp. 2129–2146. ACM (2019). https://doi.org/10.1145/3319535.3363229
9. Bertoni, G., Daemen, J., Hoffert, S., Peeters, M., Assche, G.V., Keer, R.V.: The eXtended Keccak Code Package (XKCP). GitHub Repository, https://github.com/XKCP/XKCP
10. Bertoni, G., Daemen, J., Hoffert, S., Peeters, M., Assche, G.V., Keer, R.V., Viguier, B.: TurboSHAKE. Cryptology ePrint Archive, Paper 2023/342 (2023), https://eprint.iacr.org/2023/342
11. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Sponge Functions. Ecrypt Hash Workshop (2007)
12. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: On the Indifferentiability of the Sponge Construction. In: Smart, N.P. (ed.) Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings. Lecture Notes in Computer Science, vol. 4965, pp. 181–197. Springer (2008). https://doi.org/10.1007/978-3-540-78967-3_11

13. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V., Keer, R.V., Viguier, B.: KangarooTwelve: Fast hashing based on Keccak-p. In: Preneel, B., Vercauteren, F. (eds.) Applied Cryptography and Network Security - 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10892, pp. 400–418. Springer (2018). https://doi.org/10.1007/978-3-319-93387-0_21
14. Choi, W., Lee, B., Lee, J.: Indifferentiability of Truncated Random Permutations. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11921, pp. 175–195. Springer (2019). https://doi.org/10.1007/978-3-030-34578-5_7
15. Cogliati, B., Seurin, Y.: EWCDM: an efficient, beyond-birthday secure, nonce-misuse resistant MAC. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I. Lecture Notes in Computer Science, vol. 9814, pp. 121–149. Springer (2016). https://doi.org/10.1007/978-3-662-53018-4_5
16. Coron, J., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård Revisited: How to Construct a Hash Function. In: Shoup, V. (ed.) Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3621, pp. 430–448. Springer (2005). https://doi.org/10.1007/11535218_26
17. Dinur, I.: Cryptanalytic Applications of the Polynomial Method for Solving Multivariate Equation Systems over GF(2). In: Canteaut, A., Standaert, F. (eds.) Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12696, pp. 374–403. Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_14
18. Dong, X., Zhao, B., Qin, L., Hou, Q., Zhang, S., Wang, X.: Generic MitM Attack Frameworks on Sponge Constructions. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14923, pp. 3–37. Springer (2024). https://doi.org/10.1007/978-3-031-68385-5_1
19. Faria, H., Valença, J.M.: Post-Quantum Authentication with Lightweight Cryptographic Primitives. Cryptology ePrint Archive, Paper 2021/1298 (2021), https://eprint.iacr.org/2021/1298
20. Foekens, R.: Security of the Sponge Construction with a Random Transformation. Bachelor's Thesis (2023)
21. Fu, Q., Luo, Y., Yang, Q., Song, L.: Preimage and collision attacks on reduced Ascon using algebraic strategies. Cybersecur. **8**(1), 34 (2025). https://doi.org/10.1186/S42400-024-00340-7
22. Guo, C., Hu, K., Fan, Y., Fu, Y., Wang, M.: Adding Feeding Forward Back to the Sponge Construction. Cryptology ePrint Archive, Paper 2025/1006 (2025), https://eprint.iacr.org/2025/1006
23. Guo, J., Liu, G., Song, L., Tu, Y.: Exploring SAT for Cryptanalysis: (Quantum) Collision Attacks Against 6-Round SHA-3. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December

5-9, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13793, pp. 645–674. Springer (2022). https://doi.org/10.1007/978-3-031-22969-5_22
24. Guo, J., Liu, M., Song, L.: Linear Structures: Applications to Cryptanalysis of Round-Reduced Keccak. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10031, pp. 249–274 (2016). https://doi.org/10.1007/978-3-662-53887-6_9
25. He, L., Lin, X., Yu, H.: Improved Preimage Attacks on 4-Round Keccak-224/256. IACR Trans. Symmetric Cryptol. **2021**(1), 217–238 (2021). https://doi.org/10.46586/TOSC.V2021.I1.217-238
26. Hülsing, A., Bernstein, D.J., Dobraunig, C., Eichlseder, M., Fluhrer, S., Gazdag, S.L., Kampanakis, P., Kölbl, S., Lange, T., Lauridsen, M.M., Mendel, F., Niederhagen, R., Rechberger, C., Rijneveld, J., Schwabe, P., Aumasson, J.P., Westerbaan, B., Beullens, W.: SPHINCS$^+$: Submission to the NIST post-quantum project, v.3 (2020), https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
27. Joux, A.: Multicollisions in iterated hash functions. Application to cascaded constructions. In: Annual International Cryptology Conference. pp. 306–316. Springer (2004)
28. Kelsey, J., Schneier, B.: Second preimages on n-bit hash functions for much less than 2^n work. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 474–490. Springer (2005)
29. Lefevre, C., Mennink, B.: Tight Preimage Resistance of the Sponge Construction. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13510, pp. 185–204. Springer (2022). https://doi.org/10.1007/978-3-031-15985-5_7
30. Lefevre, C., Mennink, B.: Permutation-Based Hashing Beyond the Birthday Bound. IACR Trans. Symmetric Cryptol. **2024**(1), 71–113 (2024). https://doi.org/10.46586/TOSC.V2024.I1.71-113
31. Lefevre, C., Mennink, B.: SoK: Security of the Ascon modes. IACR Trans. Symmetric Cryptol. **2025**(1), 138–210 (2025). https://doi.org/10.46586/TOSC.V2025.I1.138-210
32. Li, H., He, L., Chen, S., Guo, J., Qiu, W.: Automatic Preimage Attack Framework on Ascon Using a Linearize-and-Guess Approach. IACR Trans. Symmetric Cryptol. **2023**(3), 74–100 (2023). https://doi.org/10.46586/TOSC.V2023.I3.74-100
33. Liu, G., Li, S., Li, H., Qiu, W., Sun, S.: SAT-based space partitioning and applications to ascon-hash256. Cryptology ePrint Archive, Paper 2025/1542 (2025), https://eprint.iacr.org/2025/1542
34. Matyas, S.M.: Generating strong one-way functions with cryptographic algorithm. IBM Technical Disclosure Bulletin **27**, 5658–5659 (1985)
35. Maurer, U.M., Renner, R., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In: Naor, M. (ed.) Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings. Lecture Notes in Computer Science, vol. 2951, pp. 21–39. Springer (2004). https://doi.org/10.1007/978-3-540-24638-1_2

36. Naito, Y., Ohta, K.: Improved Indifferentiable Security Analysis of PHOTON. In: Abdalla, M., Prisco, R.D. (eds.) Security and Cryptography for Networks - 9th International Conference, SCN 2014, Amalfi, Italy, September 3-5, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8642, pp. 340–357. Springer (2014). https://doi.org/10.1007/978-3-319-10879-7_20
37. NICCS: Announcement on Launching the Next-generation Commercial Cryptographic Algorithms Program (2025), https://www.niccs.org.cn/niccs/Notice/pc/content/content_1937427069632253952.html
38. NIST: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions . FIPS PUB 202 (2015), https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.202.pdf
39. NIST: Stateless Hash-Based Digital Signature Standard. Federal Information Processing Standards Publication, FIPS 205 (2024), https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf
40. Niu, Z., Hu, K., Sun, S., Zhang, Z., Wang, M.: Speeding Up Preimage and Key-Recovery Attacks with Highly Biased Differential-Linear Approximations. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14923, pp. 73–104. Springer (2024). https://doi.org/10.1007/978-3-031-68385-5_3
41. Qiao, K., Song, L., Liu, M., Guo, J.: New Collision Attacks on Round-Reduced Keccak. In: Coron, J., Nielsen, J.B. (eds.) Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III. Lecture Notes in Computer Science, vol. 10212, pp. 216–243 (2017). https://doi.org/10.1007/978-3-319-56617-7_8
42. Qin, L., Hua, J., Dong, X., Yan, H., Wang, X.: Meet-in-the-Middle Preimage Attacks on Sponge-Based Hashing. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14007, pp. 158–188. Springer (2023). https://doi.org/10.1007/978-3-031-30634-1_6
43. Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In: Roy, B.K., Meier, W. (eds.) Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers. Lecture Notes in Computer Science, vol. 3017, pp. 371–388. Springer (2004). https://doi.org/10.1007/978-3-540-25937-4_24
44. Srivastava, V., Gupta, N., Jati, A., Baksi, A., Breier, J., Chattopadhyay, A., Debnath, S.K., Hou, X.: Ascon-Sign Submission to the NIST Post-quantum Project (2023), https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures
45. Sönmez Turan, M., McKay, K.A., Chang, D., Kang, J., Kelsey, J.: Ascon-Based Lightweight Cryptography Standards for Constrained Devices. NIST SP 800-232 (August 2025), https://doi.org/10.6028/NIST.SP.800-232
46. Viguier, B., Wong, D., Assche, G.V., Dang, Q., Daemen, J.: KangarooTwelve and TurboSHAKE. Internet-Draft (2024), https://www.ietf.org/archive/id/draft-irtf-cfrg-kangarootwelve-12.html
47. Wu, H.: The hash function JH. Submission to NIST (round 3) **6** (2011)

48. Zhang, Z., Hou, C., Liu, M.: Probabilistic Linearization: Internal Differential Collisions in up to 6 Rounds of SHA-3. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14923, pp. 241–272. Springer (2024). https://doi.org/10.1007/978-3-031-68385-5_8
49. Zhang, Z., Hou, C., Liu, M.: Preimage Attacks on up to 5 Rounds of SHA-3 Using Internal Differentials. In: Fehr, S., Fouque, P. (eds.) Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4-8, 2025, Proceedings, Part I. Lecture Notes in Computer Science, vol. 15601, pp. 333–363. Springer (2025). https://doi.org/10.1007/978-3-031-91107-1_12

Supplementary Material

A Comparison between SHA-3-n and Keccak-EDM$^c[n+64, n]$

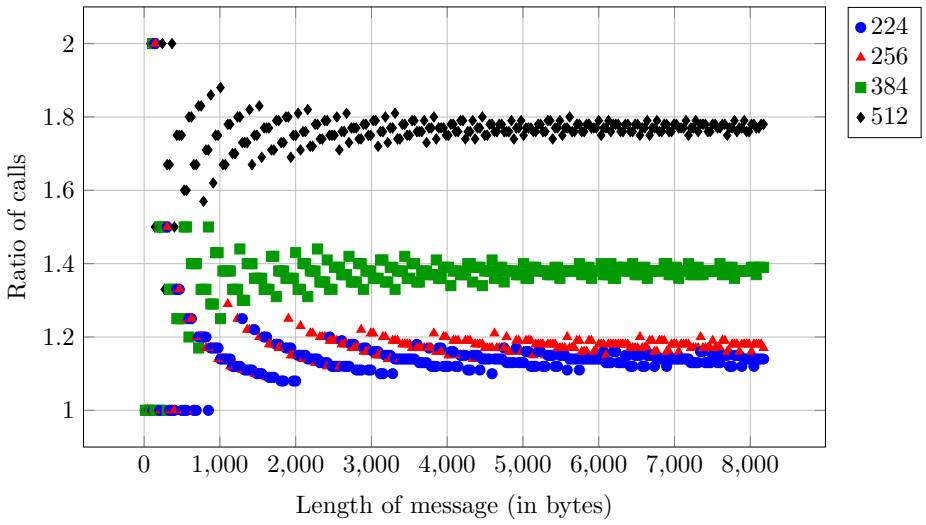


Fig. 5: Ratio of primitive calls between the SHA-3-n and Keccak-EDM$^c[n+64, n]$.

B Long Message Second Preimage Attack

In addition to the attacks of Section 3.3, we present a *new* long message second preimage attack on the sponge construction, which works even if the length of the processed message is encoded in the padding. We remark that this long-message attack is not very relevant for the SHA-3 and Ascon families, since only $\frac{c}{2}$-bit security is claimed with respect to second preimage attacks. However, the long message second preimage attack is very important in our case, since we intend to overcome the $\frac{c}{2}$-bit bound with the SPONGE-DM and SPONGE-EDM^a constructions in Section 4.

The procedure of this attack depends on the padding scheme of the hash function. Unlike Merkle-Damgård hash functions, for many sponge-based hash functions, the last blocks of two padded messages with different lengths can be the same (e.g., the padding scheme of SHA-3). In the following, we describe the long message second preimage attack under the assumption that the last blocks of two padded messages with different lengths are different, which makes the attack more complicated. For example, Wu [47] proposed a sponge-like hash function that employs message length padding. For padding schemes of SHA-3, the attack can be carried out more straightforwardly.

The attack starts with the construction of a set of expandable messages [28] based on the multi-collision technique [27]. Let $\mathfrak{a} \in \mathbb{F}_2^{r \cdot \ell}$ be a bit string of ℓ r-bit blocks, we use $x \xrightarrow{\mathfrak{a}} y$ to denote that after absorbing $\mathfrak{a}$ by the sponge with the starting state $x \in \mathbb{F}_2^b$, we get the output state $y \in \mathbb{F}_2^b$. For example, $0^b \xrightarrow{\mathsf{m}_1 \| \mathsf{m}_2} y$ is visualized in Figure 6.

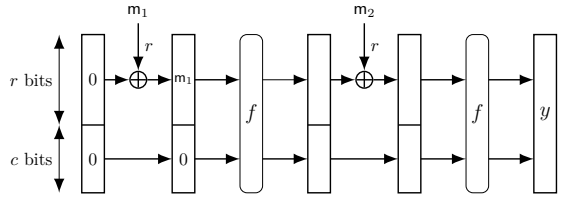


Fig. 6: A visualization of $0^b \xrightarrow{\mathsf{m}_1 \| \mathsf{m}_2} y$.

Then, the entire attack operates as follows, and it is visualized in Figure 7. We first construct two sequences $[\mathfrak{a}_1^{(0)}, \ldots, \mathfrak{a}_t^{(0)}]$ and $[\mathfrak{a}_1^{(1)}, \ldots, \mathfrak{a}_t^{(1)}]$ with $\mathfrak{a}_i^{(0)} \in \mathbb{F}_2^{2r}$, $\mathfrak{a}_i^{(1)} = 0^{r \cdot 2^{i-1}} \| \mathfrak{s}_i$, $\mathfrak{s}_i \in \mathbb{F}_2^{2r}$ $(1 \le i \le t)$ such that the following condition

$$\begin{cases} 0^b \xrightarrow{\mathfrak{a}_1^{(0)}} x_1, x_1 \xrightarrow{\mathfrak{a}_2^{(0)}} x_2, \cdots, x_{t-1} \xrightarrow{\mathfrak{a}_t^{(0)}} x_t\,, \\ 0^b \xrightarrow{\mathfrak{a}_1^{(1)}} x_1, x_1 \xrightarrow{\mathfrak{a}_2^{(1)}} x_2, \cdots, x_{t-1} \xrightarrow{\mathfrak{a}_t^{(1)}} x_t \end{cases} \tag{8}$$

is fulfilled. With these two sequences, we can construct a set

$$\mathbb{M} = \{\mathfrak{a}_1^{(b_1)} \| \mathfrak{a}_2^{(b_2)} \| \cdots \| \mathfrak{a}_t^{(b_t)} : (b_t, b_{t-1}, \cdots, b_1) \in \mathbb{F}_2^t\}$$

of 2^t messages called an expandable message set. It is easy to check that the length of the message $\mathfrak{a}_1^{(b_1)} \| \mathfrak{a}_2^{(b_2)} \| \cdots \| \mathfrak{a}_t^{(b_t)}$ is $((b_1 \cdot 2^0 + 2)r + (b_2 \cdot 2^1 + 2)r + \cdots + (b_t \cdot 2^{t-1} + 2)r)$-bit, and the lengths of the messages in $\mathbb{M}$ range from $2t$ to $2^t + 2t - 1$ r-bit blocks. Moreover, for any $\mathsf{msg} \in \mathbb{M}$, $0^b \xrightarrow{\mathsf{msg}} x_t$. Such an expandable message set can be produced with time $\mathcal{O}(t \cdot 2^{\frac{c}{2}})$, as follows: Given x_i, we can find in time $\mathcal{O}(2^{\frac{c}{2}})$ two r-bit blocks $\mathfrak{u} \in \mathbb{F}_2^r$ and $\dot{\mathfrak{u}} \in \mathbb{F}_2^r$ such that the capacity parts of y and $\dot{y}$ collide or $\lfloor y \rfloor^{(c)} = \lfloor \dot{y} \rfloor^{(c)}$, where $x_i \xrightarrow{\mathfrak{u}} y$ and $x_i \xrightarrow{0^{r \cdot (2^i - 1)} \| \dot{\mathfrak{u}}} \dot{y}$. At this point, if we set

$$\begin{cases} \mathfrak{a}_{i+1}^{(0)} = \mathfrak{u} \| \lceil \dot{y} \rceil^{(r)}, \\ \mathfrak{a}_{i+1}^{(1)} = 0^{r \cdot (2^i - 1)} \| \dot{\mathfrak{u}} \| \lceil y \rceil^{(r)}, \end{cases} \tag{9}$$

then $x_i \xrightarrow{\mathfrak{a}_{i+1}^{(0)}} x_{i+1}$ and $x_i \xrightarrow{\mathfrak{a}_{i+1}^{(1)}} x_{i+1}$ for some $x_{i+1} \in \mathbb{F}_2^b$. We can repeat this process t times to construct $\mathbb{M}$.

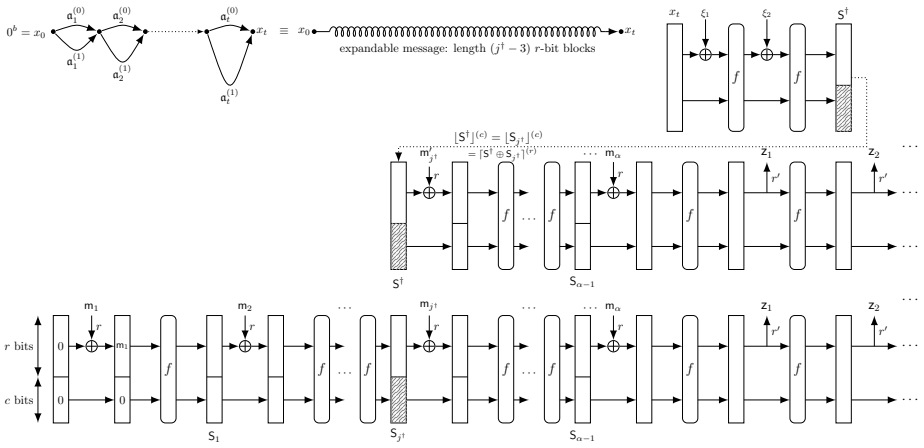


Fig. 7: The long-message second preimage attack

Let $\mathsf{msg} = \mathsf{m}_1 \parallel \cdots \parallel \mathsf{m}_\alpha \in \mathbb{F}_2^{r\cdot\alpha}$ be a message and after padding it becomes $\mathsf{m}_1 \parallel \cdots \parallel \mathsf{m}_\alpha \parallel \mathsf{m}_{\alpha+1} \in \mathbb{F}_2^{r\cdot(\alpha+1)}$. Assume that α is much greater than $2t$ and

$$\begin{cases} 0^b \xrightarrow{\mathsf{m}_1} \mathsf{S}_1, \ldots, \mathsf{S}_{2t+1} \xrightarrow{\mathsf{m}_{2t+2}} \mathsf{S}_{2t+2}\,, \\ \mathsf{S}_{2t+2} \xrightarrow{\mathsf{m}_{2t+3}} \mathsf{S}_{2t+3}, \ldots, \mathsf{S}_\alpha \xrightarrow{\mathsf{m}_{\alpha+1}} \mathsf{S}_{\alpha+1}\,. \end{cases}$$

Then, we try to identify $(\xi_1, \xi_2) \in \mathbb{F}_2^r \times \mathbb{F}_2^r$ such that $x_t \xrightarrow{\xi_1\|\xi_2} \mathsf{S}^\dagger$ and

$$\lfloor \mathsf{S}^\dagger \rfloor^{(c)} \in \{\lfloor \mathsf{S}_{2t+2} \rfloor^{(c)}, \cdots, \lfloor \mathsf{S}_{\alpha-1} \rfloor^{(c)}\}\,.$$

It takes about $\mathcal{O}(\frac{1}{\alpha}\cdot 2^c)$ to find $(\xi_1, \xi_2) \in \mathbb{F}_2^r \times \mathbb{F}_2^r$ such that $\lfloor \mathsf{S}^\dagger \rfloor^{(c)} = \lfloor \mathsf{S}_{j^\dagger} \rfloor^{(c)}$ for some $j^\dagger \in \{2t+2, \ldots, \alpha-1\}$. Then, we select a message $\mathfrak{M}$ from $\mathbb{M}$ whose length is $(j^\dagger - 3)\cdot r$-bit. The second preimage corresponding to the original message msg is

$$\mathfrak{M} \parallel \xi_1 \parallel \xi_2 \parallel \lceil \mathsf{S}^\dagger \oplus \mathsf{S}_{j^\dagger} \rceil^{(r)} \parallel \mathsf{m}_{j^\dagger+1} \parallel \cdots \parallel \mathsf{m}_{\alpha-1} \parallel \mathsf{m}_\alpha \in \mathbb{F}_2^{r\cdot\alpha}\,.$$

Note that, given the long message msg with α r-bit blocks, we need to determine t such that the expendable message set $\mathbb{M}$ contains long enough messages to cover about $\alpha - 4$ blocks. On the other hand, we should not waste time producing an expendable message set containing a message that is too long to be useful. For this reason, we choose $t = \mathcal{O}(\log_2 \alpha)$ fulfilling the following inequality

$$2^{t-1} + 2(t-1) - 1 < \alpha - 3 \leq 2^t + 2t - 1\,.$$

Consequently, the time complexity of the attack can be estimated as

$$\mathcal{O}\left(\max\{\alpha, 2^{c-\log_2\alpha}, \log_2\alpha \cdot 2^{\frac{c}{2}}\}\right)\,.$$

Typically, $\mathcal{O}(2^{c-\log_2(\alpha)})$ dominates.

Remark 4. Unlike other generic attacks introduced in this section, in the long message second preimage attack, we do not need to compute the inverse of the permutation f employed in the sponge function. Therefore, the attack remains valid even if we replace the permutations in the sponge function with one-way functions.

C Generic (Second) Preimage Attack on SPONGE-DM^c

We assume $c \geq n$ to ensure that the collision security achieve its optimal bound. Consider the SPONGE-DM^c construction given in Figure 8, where $\mathcal{F}$ is instantiated with the SPONGE-DM^c compression function, while only feed-forwarding the inner part.

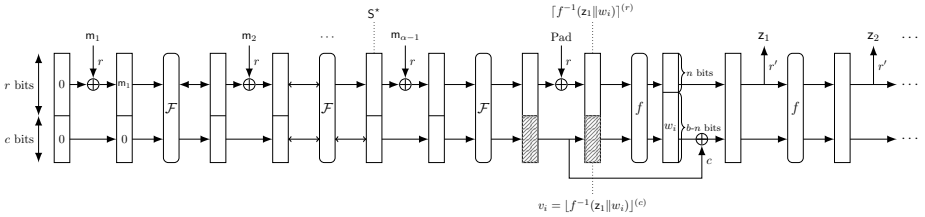


Fig. 8: Generic preimage attack on the SPONGE-DM^c construction.

When $r' \geq n$, the digest consists only z_1, where $\|\mathsf{z}_1\|_{\mathbb{F}_2} = n$. In this case, we randomly select 2^t strings $w_i \in \mathbb{F}_2^{b-n}$ and concatenate each with z_1 to form a set of b-bit candidate states:

$$\mathsf{S}_i = \mathsf{z}_1 \parallel w_i, \quad i \in \{1, 2, \cdots, 2^t\}\,.$$

For each candidate state, we compute $v_i = \lfloor f^{-1}(\mathsf{S}_i) \rfloor^{(c)}$. Next, we search for a message m such that

$$\lfloor \mathcal{F}\big(\mathsf{S}^\star \oplus (\mathsf{m}_{\alpha-1} \parallel 0^c)\big) \rfloor^{(c)} = v_j\,.$$

The expected number of trials required to find such a state m is approximately 2^{c-t}. If m is found, then $\mathsf{m}_1 \parallel \mathsf{m}_2 \parallel \cdots \parallel \mathsf{m}_{\alpha-1} \parallel \lceil \mathcal{F}\big(\mathsf{S}^\star \oplus \big(\mathsf{m}_{\alpha-1} \parallel 0^c\big)\big) \bigoplus \big(f^{-1}(\mathsf{S}_j)\big) \rceil^{(r)}$ can produce the digest z_1, regardless of the padding rule. The time complexity is $2^t + 2^{c-t}$ in case $r \geq n$ and $2^{n-r+t} + 2^{c-t}$ in case $r \leq n$, which is minimized to

$$\begin{cases} 2^{\frac{c}{2}} & \text{if } \min\{r, r'\} \geq n\,, \\ 2^{\frac{c+n-r}{2}} & \text{if } r' \geq n \geq r\,. \end{cases}$$

or

$$2^{\frac{c+n-\min\{n,r\}}{2}}\,. \tag{10}$$

When $r' < n$, the process above could be repeated approximately $2^{n-r'}$ times to expect a full n-bit digest matching. Thus, the time complexity scales accordingly:

$$\begin{cases} 2^{n-r'+\frac{c}{2}} & \text{if } \min\{n,r\} \geq r', \\ 2^{n-r'+\frac{c+r'-r}{2}} & \text{if } n \geq r' \geq r\,. \end{cases}$$

or

$$2^{\frac{c+r'-\min\{r',r\}}{2}+n-r'}. \tag{11}$$

However, if we set the repetition in the first phrase, the complexity is $2^{t+n-r'}+2^{c-t}$ in case $r \geq r'$ and $2^{t+n-r}+2^{c-t}$ in case $r \leq r'$. Hence, the total complexity of this alternative attack is given by

$$2^{\frac{c+n-\min\{r',r\}}{2}}, \tag{12}$$

which improves upon Equation (11) since $r' < n$. Furthermore, Equations (10) and (12) can be unified into the following form:

$$2^{\frac{c+n-\min\{n,r',r\}}{2}}. \tag{13}$$

Analogously to the generalization from $\texttt{SPONGE-EDM}^c$ to $\texttt{SPONGE-EDM}^a$, we could extend $\texttt{SPONGE-DM}^c$ to $\texttt{SPONGE-DM}^a$ and establish the security bound of (second) preimage resistance for $\texttt{SPONGE-DM}^a$ using the aforementioned approach, which yields

$$2^{\frac{c+n-\min\{n,r',b-a\}}{2}} \tag{14}$$

when $a \geq c$. This leads us to the following conjecture:

Conjecture 1. For $\texttt{SPONGE-DM}^a$ to achieve the optimal security bound for both collision and (second) preimage resistance, the parameters a and r' must satisfy the following conditions:

i. $a \geq 2n - c$,
ii. $\min\{n, r', b-a\} \leq c - n$.

D Generic Preimage Attack on $\texttt{SPONGE-EDM}^a$

We adapt the generic preimage attack on the standard sponge construction accordingly. Referring to Figure 9, the attack proceeds as follows.

Firstly, we aim to find a intermediate b-bit states S such that: $\lceil \mathsf{S} \rceil^{(r')} = \mathsf{z}_1$ and

$$\begin{cases} \lceil f(\mathsf{S}) \rceil^{(r')} = \mathsf{z}_2\,, \\ \lceil f \circ f(\mathsf{S}) \rceil^{(r')} = \mathsf{z}_3\,, \\ \qquad \cdots \quad \cdots \\ \lceil \underbrace{f \circ f \circ \cdots \circ f}_{l-1}(\mathsf{S}) \rceil^{(n-(l-1)r')} = \mathsf{z}_l\,. \end{cases} \tag{15}$$

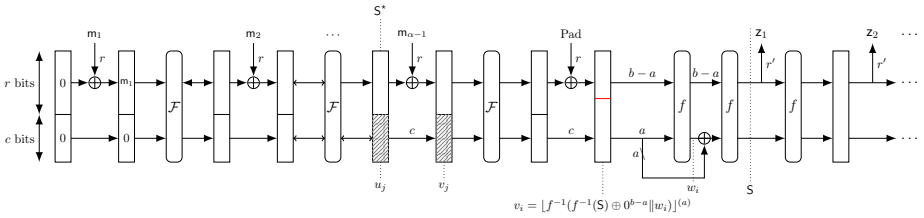


Fig. 9: Generic preimage attack on the SPONGE-EDM^a construction.

Identifying such state S has a time complexity of $2^{n-r'}$. Next, for such state S, we compute $f^{-1}(\mathsf{S})$ and attempt to find an a-bit value w_i such that:

$$\lfloor f^{-1}(\lceil f^{-1}(\mathsf{S})\rceil^{(b-a)} \| w_i)\rfloor^{(a)} \oplus w_i = \lfloor f^{-1}(\mathsf{S})\rfloor^{(a)} .$$

The complexity of this procedure is at most $O(2^a)$. We assume that the composition of f and the a-bit feed-forward is a permutation, ensuring that a suitable w_i can always be found via exhaustive search. This allows us to generate a valid v_i at a cost of 2^a. Next, we randomly select 2^x valid values Pad and apply the same procedure to perform one additional inverse round of $\mathcal{F}$. This results in the generation of 2^x valid values v_j, which are stored in a hash table to facilitate efficient lookup. The time complexity of this step is therefore $2^x \cdot 2^a = 2^{x+a}$.

Subsequently, we randomly choose 2^{c-x} message values m, compute their corresponding values u_j, and check for collisions with the previously stored v_j values in the hash table. Any collision found in this step directly yields a preimage for the target state S.

The total time complexity of the attack is $2^{n-r'} + 2^a + 2^{x+a} + 2^{c-x}$. We further distinguish on the value of a:

- when $a \geq c$, it follows that $2^{x+a} \geq 2^a \geq 2^{c-x}$. Therefore, the optimal total complexity is given by $2^{n-r'} + 2^a$, which is attained when $x = 0$.
- when $a < c$, the third and fourth terms are balanced at $x = \frac{c-a}{2}$, resulting in an optimal total complexity of $2^{n-r'} + 2^{\frac{c+a}{2}+1}$.

Hence, the optimal total complexity is given by

$$\approx 2^{n-r'} + \max\{2^a, 2^{\frac{a+c}{2}}\} \approx \max\{2^{n-r'}, 2^a, 2^{\frac{a+c}{2}}\} .$$

E Deferred Security Proofs

E.1 Proof of Indifferentiability (Theorem 1)

We first prove SPONGE-DM, and at the end explain how the proof generalizes to SPONGE-EDM^a. We have to demonstrate that there exists a simulator $\mathcal{S}$ such that

$$\mathbf{Adv}^{\text{indiff}}_{\mathcal{H},\mathcal{S}}(\mathcal{A}) = \left|\mathbf{Pr}\left(\mathcal{A}\left[\mathcal{H}[f], f^{\pm}\right] \to 1\right) - \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{R}, \mathcal{S}[\mathcal{R}]^{\pm}\right] \to 1\right)\right| \qquad (16)$$

is small for any possible adversary $\mathcal{A}$ making q accumulated queries. Here, $\left(\mathcal{H}[f], f^{\pm}\right)$ is called the real world and $\left(\mathcal{R}, \mathcal{S}[\mathcal{R}]^{\pm}\right)$ the ideal world.

The proof closely follows the approach taken by Naito and Ohta [36]. We first describe our simulator in Section E.1.1. Then, we describe an intermediate world in Section E.1.2, which allows to decompose the advantage into two distances. Each of these distances is then upper bounded in Sections E.1.3 and E.1.4, respectively, leading directly to the conclusion of Theorem 1.

E.1.1 Simulator The simulator $\mathcal{S}$ aims to simulate the permutation f in both directions while ensuring consistency with $\mathcal{R}$. It maintains an initially empty database $\mathcal{T}$, storing tuples of the form (x, y, dir), where $y = \mathcal{S}(x)$ and $\mathrm{dir} \in \{\mathrm{fwd}, \mathrm{inv}\}$ denotes the query direction. For ease of reasoning, we represent these tuples in a graph. This graph consists of all nodes $\mathbb{F}_2^b$. For each tuple (x, y, dir) that gets added to $\mathcal{T}$, the following edges are added:

- $x \oplus (\mathsf{m} \| 0^c) \overset{\mathsf{m}}{\Longrightarrow} x \oplus y$ for all $\mathsf{m} \in \mathbb{F}_2^r$;
- $x \to y$.

Here, the double-lined arrow corresponds to a block absorption (potentially a zero-block) whereas the single-lined arrow does not absorb a block. For convenience of notation, we denote $\mathsf{S}_1 \overset{\mathsf{m}_1}{\Longrightarrow} \mathsf{S}_2 \overset{\mathsf{m}_2}{\Longrightarrow} \mathsf{S}_3$ as $\mathsf{S}_1 \overset{\mathsf{m}_1 \| \mathsf{m}_2}{\Longrightarrow} \mathsf{S}_3$, and we write $\mathsf{S} \overset{\epsilon}{\Rightarrow} \mathsf{S}$.

Clearly, the simulator should intuitively make sure that, whenever there is a path

$$0^b \overset{\mathrm{padF}(\mathsf{msg})}{\Longrightarrow} y_1 \longrightarrow y_2 \longrightarrow \cdots,$$

for some $\mathsf{msg} \in \mathbb{F}_2^*$, any subsequent single-lined evaluations starting from y_1 should adhere to the random oracle output on input of msg:

$$\lceil y_1 \rceil^{(r')} \| \lceil y_2 \rceil^{(r')} \| \cdots = \mathcal{R}(\mathsf{msg}).$$

The simulator is described in Algorithm 2.

Next, let $\mathcal{Q}$ be the permutation query history of the adversary. It contains the tuples (x, y, dir) queried by the adversary. In the ideal world, we have $\mathcal{Q} = \mathcal{T}$, but looking ahead, in the intermediate world, $\mathcal{T}$ may contain additional information that is not present in $\mathcal{Q}$. Given $\mathcal{X} \in \{\mathcal{T}, \mathcal{Q}\}$, let $\mathtt{Rooted}(\mathcal{X})$ denote the set of $y \in \mathbb{F}_2^b$ such that there exists a path that can be reconstructed from $\mathcal{X}$ of the form $0^b \overset{\mathrm{padF}(\mathsf{msg})}{\Longrightarrow} y_1 \to \cdots \to y_\beta$, for some $\mathsf{msg} \in \mathbb{F}_2^*$ and $\beta \geq 1$ with $y_\beta = y$.

E.1.2 Intermediate World To facilitate the analysis, we introduce an intermediate world $(\mathcal{H}[\mathcal{S}[\mathcal{R}]], \mathcal{S}[\mathcal{R}]^{\pm})$. This world implements the real world, except that the random permutation f is replaced by the simulator $\mathcal{S}$, which internally uses a random oracle $\mathcal{R}$ hidden from the adversary. By the triangle inequality, we have

$$(16) \leq \left| \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{H}[f], f^{\pm}\right] \to 1\right) - \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{H}[\mathcal{S}[\mathcal{R}]], \mathcal{S}[\mathcal{R}]^{\pm}\right] \to 1\right) \right| \tag{17}$$

Algorithm 2: Simulator

Function $\mathcal{S}(x)$**:**
  if $\exists (x, y, \mathrm{dir}) \in \mathcal{T}$ **then**
    return y
  $y^{(c')} \xleftarrow{\$} \mathbb{F}_2^{c'}$
  if $\exists \mathsf{u} \in \mathbb{F}_2^r \setminus \{0^r\}, \mathsf{m} \in (\mathbb{F}_2^r)^* \cup \{\epsilon\}$ *with* $0^b \overset{\mathsf{m}}{\Longrightarrow} x \oplus (\mathsf{u} \parallel 0^c)$ *and* $\mathrm{upadF}(\mathsf{m} \parallel \mathsf{u}) \neq \perp$ **then** `// last absorb call`
    $y^{(r')} \leftarrow \mathcal{R}(\mathrm{upadF}(\mathsf{m} \parallel \mathsf{u}), r') \oplus \lceil x \rceil^{(r')}$
  else if $\exists \mathsf{msg} \in \mathbb{F}_2^*, \beta \geq 1$ *with* $0^b \xRightarrow{\mathrm{padF(msg)}} y_1 \to \cdots \to y_\beta$ *and* $y_\beta = x$ **then** `// squeezing`
    $y^{(r')} \leftarrow \lfloor \mathcal{R}(\mathsf{msg}, (\beta + 1) \cdot r') \rfloor^{(r')}$
  else
    $y^{(r')} \xleftarrow{\$} \mathbb{F}_2^{r'}$
  $\mathcal{T} \leftarrow \mathcal{T} \cup \left\{ \left(x, y^{(r')} \parallel y^{(c')}, \mathrm{fwd} \right) \right\}$
  return $y^{(r')} \parallel y^{(c')}$

Function $\mathcal{S}^{-1}(y)$**:**
  if $\exists (x, y, \mathrm{dir}) \in \mathcal{T}$ **then**
    return x
  $x \xleftarrow{\$} \mathbb{F}_2^b$
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{(x, y, \mathrm{inv})\}$
  return x

$$+ \left| \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{H}[\mathcal{S}[\mathcal{R}]], \mathcal{S}[\mathcal{R}]^{\pm} \right] \to 1 \right) - \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{R}, \mathcal{S}[\mathcal{R}]^{\pm} \right] \to 1 \right) \right| . \tag{18}$$

Distance (17) is upper bounded in Section E.1.3, and distance (18) in Section E.1.4.

E.1.3 Distance Between Real and Intermediate World The only distinction between these worlds is the primitive oracle's source of randomness: in the ideal world this is a random permutation, whereas in the intermediate world the simulator implements a two-sided random function.[7] As long as no full-state collision occurs, the simulator is indistinguishable from a random permutation. Therefore, by the fundamental lemma of game playing [7],

$$(17) \leq \frac{q^2}{2^{b+1}} . \tag{19}$$

[7] A two-sided random function F is similar to a random permutation. It maintains a table of pairs (X, Y), meaning that $F(X) = Y$ and $F^{-1}(Y) = X$. When queried on a new input, it returns a uniformly random answer from $\mathbb{F}_2^b$ and stores the pair in the table.

E.1.4 Distance Between Intermediate and Ideal World For $i = 1, \ldots, q$, let $\mathcal{T}_i$ and $\mathcal{Q}_i$ denote the state of respectively $\mathcal{T}$ and $\mathcal{Q}$ after the first i queries.

Bad Events. We define a family of bad events, indexed by a query index $i = 1, \ldots, q$, and defined over $\mathcal{T}$ and $\mathcal{Q}$.

$$\begin{aligned}
\mathsf{COLL}_i : &\exists (x, y, \mathrm{dir}) \in \mathcal{T}_i \setminus \mathcal{T}_{i-1}, (x', y', \mathrm{dir}') \in \mathcal{T}_{i-1} \text{ such that} \\
&\quad \text{either } \mathrm{dir} = \mathrm{fwd} \text{ and} \\
&\qquad \left\{ \lfloor y \rfloor^{(c)}, \lfloor y \oplus x \rfloor^{(c)} \right\} \cap \left\{ \lfloor y' \rfloor^{(c)}, \lfloor y' \oplus x' \rfloor^{(c)}, \lfloor x' \rfloor^{(c)}, 0^c \right\} \neq \emptyset \,, \\
&\quad \text{or } \mathrm{dir} = \mathrm{inv} \text{ and } \lfloor x \rfloor^{(c)} \in \left\{ \lfloor y' \rfloor^{(c)}, \lfloor y' \oplus x' \rfloor^{(c)}, \lfloor x' \rfloor^{(c)}, 0^c \right\} , \\
\mathsf{GUESS}_i : &\exists (x, y, \mathrm{dir}) \in \mathcal{Q}_i \setminus \mathcal{Q}_{i-1} \text{ such that } y \in \mathtt{Rooted}(\mathcal{T}_i) \setminus \mathtt{Rooted}(\mathcal{Q}_i) \,, \\
\mathsf{BAD}_i : &\mathsf{COLL}_i \vee \mathsf{GUESS}_i \,.
\end{aligned}$$

Moreover, for $\mathsf{EVT} \in \{\mathsf{COLL}, \mathsf{GUESS}, \mathsf{BAD}\}$, let $\mathsf{EVT} = \bigvee_{i=1}^{q} \mathsf{EVT}_i$.

The bad event COLL ensures that the simulator responds consistently with respect to the random oracle by forbidding inner collisions and paths connecting. GUESS is an event that can be set only in the intermediate world. It happens when the adversary queries a state in the middle of a path without having queried the earlier states first, causing the simulators in the intermediate and ideal worlds to behave differently.

Distance Between Intermediate and Ideal World as Long as no Bad. Assuming that BAD does not occur, there are no inner collisions between the intermediate states generated by $\mathcal{H}[\mathcal{S}[\mathcal{R}]]$. Consequently, for any $y \in \mathtt{Rooted}(\mathcal{T})$, there exists a unique path of the form $0^b \xrightarrow{\mathrm{padF}(\mathsf{msg})} y_1 \to \cdots \to y_\beta$, for some $\mathsf{msg} \in \mathbb{F}_2^*$ and $\beta \geq 1$ with $y_\beta = y$, where the queries were made from left to right and no intermediate state hits 0^b. Together with the sponge-compliant padding, this ensures that one primitive call produces at most one rooted note. This ensures that the simulator remains consistent with the random oracle, expressed formally as:

$$\lceil y \rceil^{(r')} = \lfloor \mathcal{R}(\mathsf{msg}, \beta \cdot r') \rfloor^{(r')} \,.$$

Since GUESS does not occur, the additional $\mathcal{H}$-originated queries that $\mathcal{S}$ receives in the intermediate world do not influence its responses to the adversary's queries. Therefore,

$$\mathbf{Pr}\left(\mathcal{A}\left[\mathcal{H}[\mathcal{S}[\mathcal{R}]], \mathcal{S}[\mathcal{R}]^{\pm}\right] \to 1 \mid \neg\mathsf{BAD}\right) = \mathbf{Pr}\left(\mathcal{A}\left[\mathcal{R}, \mathcal{S}[\mathcal{R}]^{\pm}\right] \to 1 \mid \neg\mathsf{BAD}\right) . \quad (20)$$

Probability of the Bad Events. We bound the probability of BAD in the following lemma.

Lemma 2. *In both the intermediate and ideal worlds, we have*

$$\mathbf{Pr}\left(\mathsf{BAD}\right) \leq \frac{q(3q-1)}{2^c} + \frac{2q \cdot \mathrm{mcol}(q, 2^{r'-r})}{2^{c'}} \,. \quad (21)$$

Proof. By basic probability theory, we have

$$\mathbf{Pr}\left(\mathsf{BAD}\right) \leq \sum_{i=1}^{q} \mathbf{Pr}\left(\mathsf{COLL}_i \mid \neg\mathsf{BAD}_{i-1}\right) + \mathbf{Pr}\left(\mathsf{GUESS}_i \mid \neg\mathsf{BAD}_{i-1}\right) , \tag{22}$$

where BAD_0 denotes an event always set. COLL covers at most 6 inner collision scenarios, as well as two inner collisions with the initial value 0^b. Therefore,

$$\mathbf{Pr}\left(\mathsf{COLL}_i \mid \neg\mathsf{BAD}_{i-1}\right) \leq \frac{6(i-1)+2}{2^c} . \tag{23}$$

For the event GUESS, the adversary must be able to guess some $y \in \mathtt{Rooted}(\mathcal{T}_i) \setminus \mathtt{Rooted}(\mathcal{Q}_i)$. Each of these values are uniformly random, and the only potential information available to the adversary consists of the outer r' bits of these nodes.

In the case where $r' < r$, the adversary has access to and can control r' bits. It can add arbitrary values to the middle $r - r'$ bits, but has no access to them. Therefore, in this setting the success probability of the conditioned GUESS_i is upper bounded by $\frac{(i-1)}{2^{c'}}$.

On the other hand, if $r' \geq r$, the adversary has access and can control the outer r bits, has access to the middle $r' - r$ bits, and has to guess the c' remaining ones. The situation therefore boils down to a bins-and-balls game. Define the random variable U as follows:

$$U = \max_{v \in \mathbb{F}_2^{r'-r}} \left| \{(x, y, \mathrm{fwd}) \in \mathcal{T} \mid \lceil \lfloor y \rfloor^{(c)} \rceil^{(r'-r)} = v\} \right| + \max_{v \in \mathbb{F}_2^{r'-r}} \left| \{(x, y, \mathrm{fwd}) \in \mathcal{T} \mid \lceil \lfloor y \oplus x \rfloor^{(c)} \rceil^{(r'-r)} = v\} \right| .$$

Intuitively, U upper bounds the size of the largest multicollision on the middle $r' - r$ bits among the intermediate states produced during the squeezing phase. Given $u \in \{1, \ldots, q\}$, the probability that a permutation query sets GUESS, conditioned on $U = u$, is upper bounded by $\frac{u}{2^{c'}}$. Therefore,

$$\sum_{i=1}^{q} \mathbf{Pr}\left(\mathsf{GUESS}_i \mid \neg\mathsf{BAD}_{i-1}\right) \leq \frac{q \cdot \mathbf{Ex}\left(U\right)}{2^{c'}} \leq \frac{2q \cdot \mathrm{mcol}(q, 2^{r'-r})}{2^{c'}} ,$$

for any choice of r and r'. By combining those equations along with (23) into (22), we obtain (21), hence the lemma. □

Combining (20) with (21), and using the fundamental lemma of game playing [7], we obtain

$$(18) \leq \frac{q(3q-1)}{2^c} + \frac{2q \cdot \mathrm{mcol}(q, 2^{r'-r})}{2^{c'}} . \tag{24}$$

Finally, we plug (24) and (19) into (16), which completes the proof.

E.1.5 Extension to SPONGE-EDM^a The proof for SPONGE-EDM^a is almost identical. To account for the fact that only a bits are used in the feed-forward, the double edges $y \overset{\mathsf{m}}{\Longrightarrow} y'$ are replaced by the following family of edges. For each query $(x, y, \mathrm{dir}) \in \mathcal{Q}$ and every $\mathsf{m} \in \mathbb{F}_2^r$, we add the edge

$$x \oplus (\mathsf{m} \| 0^c) \overset{\mathsf{m}}{\underset{a}{\Longrightarrow}} \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} .$$

The simulator is modified as follows. Upon a forward query with input x, if there exists a path

$$0^b \overset{\mathsf{m}_1}{\underset{a}{\Longrightarrow}} \cdot \longrightarrow \cdot \overset{\mathsf{m}_2}{\underset{a}{\Longrightarrow}} \cdot \longrightarrow \cdots \overset{\mathsf{m}_\alpha}{\underset{a}{\Longrightarrow}} \cdot \longrightarrow x \oplus (\mathsf{u} \| 0^c)$$

with $\mathsf{u} \neq 0^r$, then the simulator samples a random y, and proactively sets the value $\mathcal{S}\left(\lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)}\right)$ so as to remain consistent with the random oracle. The queries on squeezing paths remain unchanged. In the bad events COLL_i, the terms $\lfloor y \oplus x \rfloor^{(c)}$ are replaced with $\lfloor \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} \rfloor^{(c)}$, which does not affect the bounding. The query complexity of the simulator is at worst doubled, resulting in a corresponding loss in the bounds.

E.2 Proof of Preimage Resistance for SPONGE-EDM^a (Theorem 2)

The proof for SPONGE-EDM^a gets closer to the one of Lefevre and Mennink [29], but is significantly more intricate. Indeed, they did a case distinction depending on the direction of specific queries, which we will follow, but (i) we will have one more bad event and (ii) one of the bad events will get significantly more technical. As in the extension to the proof of indifferentiability (Section E.1.5), the double edges $y \overset{\mathsf{m}}{\Longrightarrow} y'$ are replaced by the following family of edges. For each query $(x, y, \mathrm{dir}) \in \mathcal{Q}$ and every $\mathsf{m} \in \mathbb{F}_2^r$, we add the edge

$$x \oplus (\mathsf{m} \| 0^c) \overset{\mathsf{m}}{\underset{a}{\Longrightarrow}} \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} .$$

$\mathsf{PRE}(\mathcal{Q})$ is now defined as

$\mathsf{PRE}(\mathcal{Q})$: $\mathcal{Q}$ defines a path

$$0^b \overset{\mathsf{m}_1}{\underset{a}{\Longrightarrow}} y'_1 \longrightarrow y''_1 \cdots \overset{\mathsf{m}_{\alpha-1}}{\underset{a}{\Longrightarrow}} y'_{\alpha-1} \longrightarrow y''_{\alpha-1} \overset{\mathsf{m}_\alpha}{\underset{a}{\Longrightarrow}} y'_\alpha \longrightarrow y_1 \longrightarrow \cdots \longrightarrow y_\beta$$

such that $y_i \in \boldsymbol{Z}_i$ for $i = 1, \ldots, \beta$.

In order to set $\mathsf{PRE}(\mathcal{Q})$, the adversary must ever make the queries corresponding to the edges $y''_{\alpha-1} \overset{\mathsf{m}_\alpha}{\underset{a}{\Longrightarrow}} y'_\alpha \longrightarrow y_1$ for some $y'_\alpha \in f^{-1}(\mathcal{S})$. We split $\mathsf{PRE}(\mathcal{Q})$ depending on the direction of the queries $y''_{\alpha-1} \overset{\mathsf{m}_\alpha}{\underset{a}{\Longrightarrow}} y'_\alpha$ and $y'_\alpha \longrightarrow y_1$:

$\mathsf{PRE\text{-}FWD}(\mathcal{Q})$: $\mathsf{PRE}(\mathcal{Q})$ with the restriction that *at least* one of the two queries triggering the edges $y''_{\alpha-1} \overset{\mathsf{m}_\alpha}{\underset{a}{\Longrightarrow}} y'_\alpha$ and $y'_\alpha \longrightarrow y_1$ is made in the

forward direction,

$\mathsf{PRE\text{-}INV}(\mathcal{Q})$: $\mathsf{PRE}(\mathcal{Q})$ with the restriction that the queries triggering the edges $y''_{\alpha-1} \xRightarrow[a]{\mathsf{m}_\alpha} y'_\alpha$ and $y'_\alpha \longrightarrow y_1$ are *both* made in the inverse direction.

Clearly, we have $\mathsf{PRE}(\mathcal{Q}) \iff \mathsf{PRE\text{-}FWD}(\mathcal{Q}) \vee \mathsf{PRE\text{-}INV}(\mathcal{Q})$. We define the set $\mathcal{S}_{\mathrm{fwd}}$ and the multiset $\mathcal{S}_{\mathrm{inv}}$ as follows:

$$\begin{aligned}\mathcal{S}_{\mathrm{fwd}} &= f^{-1}(\mathcal{S})\,,\\ \mathcal{S}_{\mathrm{inv}} &= \left\{Z, f(Z), \ldots, f^{\beta-1}(Z) \mid Z \in \mathcal{S}\right\}\,.\end{aligned}$$

The event $\mathsf{PRE\text{-}FWD}(\mathcal{Q})$ can be upper bounded in the same way as before. Stated differently, we define

$$\mathsf{BAD\text{-}FWD}(\mathcal{Q}) : \exists (x, y, \mathrm{fwd}) \in \mathcal{Q} \text{ such that } \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} \in \mathcal{S}_{\mathrm{fwd}} \text{ or } x \in \mathcal{S}_{\mathrm{fwd}}\,,$$

and we have

$$\mathsf{PRE\text{-}FWD}(\mathcal{Q}) \implies \mathsf{BAD\text{-}FWD}(\mathcal{Q})\,.$$

However, to upper bound $\mathsf{PRE\text{-}INV}(\mathcal{Q})$, we follow a different strategy. In order to set $\mathsf{PRE\text{-}INV}(\mathcal{Q})$, the adversary must necessarily satisfy each of the following three events:

- it must ever obtain a value y_1, which it can only do by ever guessing any value in $\mathcal{S}_{\mathrm{inv}}$. This will be captured by bad event $\mathsf{BAD\text{-}INV}(\mathcal{Q})$ below;
- it actually makes the inverse query $y''_{\alpha-1} \xRightarrow[a]{\mathsf{m}_\alpha} y'_\alpha$. This will be captured by bad event $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ below;
- it constructs the path $0^b \underset{a}{\Rightarrow} \cdots \longrightarrow y''_{\alpha-1}$. This will be captured by bad event $\mathsf{INNER}(\mathcal{Q})$ below.

Bad events $\mathsf{BAD\text{-}INV}(\mathcal{Q})$ and $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ are defined as follows:

$$\begin{aligned}\mathsf{BAD\text{-}INV}(\mathcal{Q}) &: \exists (x, y, \mathrm{fwd}) \in \mathcal{Q} \text{ such that } x \in \mathcal{S}_{\mathrm{inv}} \text{ or}\\ &\qquad \exists (x, y, \mathrm{inv}) \in \mathcal{Q} \text{ such that } y \in \mathcal{S}_{\mathrm{inv}}\,,\\ \mathsf{BAD\text{-}FF}(\mathcal{Q}) &: \exists (x, y, \mathrm{inv}) \in \mathcal{Q} \text{ such that } \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} \in \mathcal{S}_{\mathrm{fwd}}\,.\end{aligned}$$

Defining $\mathsf{INNER}(\mathcal{Q})$ is a bit more involved, in particular as the value $y''_{\alpha-1}$ is not fixed yet and depends on the attack. Define the set $\mathcal{N}(\mathcal{Q})$ as follows:

$$\mathcal{N}(\mathcal{Q}) = \Big\{ Y \in \mathbb{F}_2^b \mid \mathcal{Q} \text{ defines a path } Y \xRightarrow[a]{\mathsf{m}} Y' \longrightarrow Y'' \text{ where both queries were made in the inverse direction} \Big\}\,.$$

The third event as outlined above is implied by the event that there exists a path from 0^b to $\mathcal{N}(\mathcal{Q})$ (noting that $y''_{\alpha-1} \in \mathcal{N}(\mathcal{Q})$ by assumption of the current case)

or that the inverted feed-forward directly connects to 0^b. Therefore, we define $\mathsf{INNER}(\mathcal{Q})$ as follows:[8]

$$\begin{aligned}\mathsf{INNER}(\mathcal{Q}) : &\exists (x, y, \mathrm{fwd}) \in \mathcal{Q}, z \in \mathcal{N}(\mathcal{Q}) \text{ such that } y = z\,, &(25)\\ &\text{or } \exists (x, y, \mathrm{fwd}), (x', y', \mathrm{inv}) \in \mathcal{Q} \text{ such that } y' \in \mathcal{N}(\mathcal{Q}) \text{ and}\\ &\lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} = x'\,, &(26)\\ &\text{or } \exists (x, y, \mathrm{inv}) \in \mathcal{Q} \text{ such that } \lfloor x \rfloor^{(c)} = 0^c\,. &(27)\end{aligned}$$

We have, as for $\mathsf{PRE\text{-}INV}(\mathcal{Q})$ all three events must be set,

$$\mathsf{PRE\text{-}INV}(\mathcal{Q}) \implies \mathsf{BAD\text{-}INV}(\mathcal{Q}) \wedge \mathsf{BAD\text{-}FF}(\mathcal{Q}) \wedge \mathsf{INNER}(\mathcal{Q})\,.$$

Therefore,

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{PRE}(\mathcal{Q})\right) \leq\ &\mathbf{Pr}\left(\mathsf{BAD\text{-}FWD}(\mathcal{Q})\right) +\\ &\min\left\{\mathbf{Pr}\left(\mathsf{BAD\text{-}INV}(\mathcal{Q})\right), \mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q})\right), \mathbf{Pr}\left(\mathsf{INNER}(\mathcal{Q})\right)\right\}\,. \quad (28)\end{aligned}$$

Compared to the proof of Lefevre and Mennink, we extend the event "inner collisions" to a more advanced event that takes into account the feedforward, and add an event of the form "the adversary inverts the feedforward".

The probability of $\mathsf{BAD\text{-}FWD}(\mathcal{Q})$ can be upper bounded in the same way as in the case of `SPONGE-DM`, and the bad event $\mathsf{BAD\text{-}INV}(\mathcal{Q})$ is exactly the same as defined by Lefevre and Mennink [29]. Therefore,

$$\mathbf{Pr}\left(\mathsf{BAD\text{-}FWD}(\mathcal{Q})\right) \leq \frac{8q}{2^n}\,, \quad (29)$$

$$\mathbf{Pr}\left(\mathsf{BAD\text{-}INV}(\mathcal{Q})\right) \leq \frac{4\beta q}{2^{n-r'}}\,. \quad (30)$$

It remains to evaluate the probabilities of $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ and $\mathsf{INNER}(\mathcal{Q})$.

We start with $\mathsf{BAD\text{-}FF}(\mathcal{Q})$. Define the random variable $U_{\mathcal{S}}$ as follows:

$$U_{\mathcal{S}} = \max_{t \in \mathbb{F}_2^{b-a}} \left| \left\{ S \in \mathcal{S}_{\mathrm{fwd}} : \lceil S \rceil^{(b-a)} = t \right\} \right|\,.$$

$U_{\mathcal{S}}$ upper bounds the maximum multicollision size over the outer $b-a$ bits, over all elements in $\mathcal{S}_{\mathrm{fwd}}$. Conditioned on $U_{\mathcal{S}} = u$, the ith query sets $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ with a probability of at most $\frac{u \cdot 2^{b-a}}{2^b - (i-1)}$. Therefore, by basic probability theory,

$$\mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q})\right) \leq \sum_{u=0}^{2^b} \sum_{i=1}^{q} \mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q}_i) \mid U_{\mathcal{S}} = u \wedge \neg \mathsf{BAD\text{-}FF}(\mathcal{Q}_{i-1})\right) \cdot \mathbf{Pr}\left(U_{\mathcal{S}} = u\right)$$

[8] Here, we remark that the direct connection to 0^b is already simplified in this bad event because, when $a > c$, the remaining bad events $\mathsf{BAD\text{-}INV}(\mathcal{Q})$ and $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ suffice to obtain a tight bound.

$$\leq \sum_{u=0}^{2^b} \sum_{i=1}^{q} \frac{u \cdot 2^{b-a}}{2^b - (i-1)} \cdot \mathbf{Pr}\left(U_{\mathcal{S}} = u\right)$$
$$\leq \sum_{u=0}^{2^b} \frac{2qu}{2^a} \cdot \mathbf{Pr}\left(U_{\mathcal{S}} = u\right) = \frac{2q}{2^a} \cdot \mathbf{Ex}\left(U_{\mathcal{S}}\right) ,$$

assuming that $q \leq 2^{b-1}$. Let $s \leq 2^{c'}$, we can use Lemma 1 and obtain

$$\mathbf{Ex}\left(U_{\mathcal{S}} \mid |\mathcal{S}| = s\right) \leq \frac{2s}{2^{b-a}} + 3(b-a) + 4 ,$$

so that

$$\begin{aligned}\mathbf{Ex}\left(U_{\mathcal{S}}\right) &\leq \frac{2 \cdot \mathbf{Ex}\left(|\mathcal{S}|\right)}{2^{b-a}} + 3(b-a) + 4 \\ &\leq \frac{4 \cdot 2^a}{2^n} + 3(b-a) + 4 ,\end{aligned}$$

where the last inequality uses that $\mathbf{Ex}\left(|\mathcal{S}|\right) \leq 2\frac{2^b}{2^n}$ [29, Eq. (17), proof of Lemma 2, p. 198]. Finally, we get

$$\mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q})\right) \leq \frac{8q}{2^n} + \frac{(6(b-a)+8)q}{2^a} . \tag{31}$$

We now proceed to upper bound the probability to set $\mathsf{INNER}(\mathcal{Q})$. Looking ahead, the bounds are valid for any $a \in \{0, \ldots, b\}$, but meaningful only when $a < c$; otherwise, it is better to bound the minimum term using $\mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q})\right)$ or $\mathbf{Pr}\left(\mathsf{BAD\text{-}INV}(\mathcal{Q})\right)$.

$\mathsf{INNER}(\mathcal{Q})$ is split into three cases, i.e., (25), (26), and (27). We first start with case (27), which concerns the probability that an inverse query hits the initial value. One query sets this event with a probability of at most

$$\frac{2^r}{2^b - (i-1)} \leq \frac{2}{2^c} ,$$

using that $q \leq 2^{b-1}$. Now, for the cases (25) and (26), we divide the analysis into five subcases:

- $\mathsf{AUX1}(\mathcal{Q})$: cases (25) and (26) of $\mathsf{INNER}(\mathcal{Q})$ with the restriction that the query triggering this event is the forward query (x, y, fwd);
- $\mathsf{AUX2}(\mathcal{Q})$: case (26) of $\mathsf{INNER}(\mathcal{Q})$ with the restriction that the query triggering this event is the inverse query $(x', y', \mathrm{inv}) \in \mathcal{Q}_i \setminus \mathcal{Q}_{i-1}$ where $y' \in \mathcal{N}(\mathcal{Q}_{i-1})$ for some $i \in \{1, \ldots, q\}$;
- $\mathsf{AUX3}(\mathcal{Q})$: cases (25) and (26) of $\mathsf{INNER}(\mathcal{Q}_i)$ with the restriction that the query triggering this event is some query $(x'', y'', \mathrm{inv}) \in \mathcal{Q}_i \setminus \mathcal{Q}_{i-1}$ such that
  - $\lfloor x'' \rfloor^{(c)} \neq \lfloor z \rfloor^{(c)}$ and $z \in \mathcal{N}(\mathcal{Q}_i) \setminus \mathcal{N}(\mathcal{Q}_{i-1})$ for case (25);
  - $x'' \neq x'$ and $y' \in \mathcal{N}(\mathcal{Q}_i) \setminus \mathcal{N}(\mathcal{Q}_{i-1})$ for case (26);

  for some $i \in \{1, \ldots, q\}$;

– $\mathsf{AUX4}(\mathcal{Q})$: case (25) of $\mathsf{INNER}(\mathcal{Q})$ with the restriction that the query triggering this event is an inverse query $(x', y', \mathrm{inv}) \in \mathcal{Q}_i \setminus \mathcal{Q}_{i-1}$, where $\lfloor x' \rfloor^{(c)} = \lfloor z \rfloor^{(c)}$ and $z \in \mathcal{N}(\mathcal{Q}_i) \setminus \mathcal{N}(\mathcal{Q}_{i-1})$ for some $i \in \{1, \ldots, q\}$;
– $\mathsf{AUX5}(\mathcal{Q})$: case (26) of $\mathsf{INNER}(\mathcal{Q})$ with the restriction that the query triggering this event is the inverse query $(x', y', \mathrm{inv}) \in \mathcal{Q}_i \setminus \mathcal{Q}_{i-1}$, where $y' \in \mathcal{N}(\mathcal{Q}_i) \setminus \mathcal{N}(\mathcal{Q}_{i-1})$, for some $i \in \{1, \ldots, q\}$.

$\mathsf{AUX1}(\mathcal{Q})$ means that a forward query connects with some y that was already known to belong to $\mathcal{N}(\mathcal{Q})$ or $f^{-1}(\mathcal{N}(\mathcal{Q}))$. Similarly, $\mathsf{AUX2}(\mathcal{Q})$ pinpoints the event that a query that was already tied to $\mathcal{N}(\mathcal{Q})$ connects in the inverse direction to a forward query. $\mathsf{AUX3}(\mathcal{Q})$ denotes the event that a forward-inverse collision occurs, but the resulting double edge is added to $\mathcal{N}(\mathcal{Q})$ only by a later query. $\mathsf{AUX4}(\mathcal{Q})$ means that the triggering query is inverse, connects to a forward query, and the resulting double edge $z \underset{a}{\Rightarrow} z'$ connects to an inverse query, yielding to the addition of z to $\mathcal{N}(\mathcal{Q})$. Finally, $\mathsf{AUX5}(\mathcal{Q})$ refers to the event where an inverse query connects to a forward query through its answer x', while at the same time triggering y' to be added to $\mathcal{N}(\mathcal{Q})$.

By incorporating the case (27) into the analysis, we have

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{INNER}(\mathcal{Q})\right) \leq \frac{2q}{2^c} + \sum_{\nu}\sum_{i} \mathbf{Pr}\left(\mathsf{AUX1}(\mathcal{Q}_i) \mid \neg\mathsf{INNER}(\mathcal{Q}_{i-1}) \wedge |\mathcal{N}(\mathcal{Q}_{i-1})| = \nu\right) \cdot \\ \mathbf{Pr}\left(|\mathcal{N}(\mathcal{Q}_{i-1})| = \nu\right) + \\ \sum_{\nu} \mathbf{Pr}\left(\mathsf{AUX2}(\mathcal{Q}) \vee \mathsf{AUX3}(\mathcal{Q}) \mid \neg\mathsf{AUX1}(\mathcal{Q}) \wedge \neg\mathsf{AUX4}(\mathcal{Q}) \wedge \neg\mathsf{AUX5}(\mathcal{Q}) \wedge |\mathcal{N}(\mathcal{Q})| = \nu\right) \cdot \\ \mathbf{Pr}\left(|\mathcal{N}(\mathcal{Q})| = \nu\right) + \\ \sum_{i} \mathbf{Pr}\left(\mathsf{AUX4}(\mathcal{Q}_i) \wedge \neg\mathsf{INNER}(\mathcal{Q}_{i-1})\right) + \mathbf{Pr}\left(\mathsf{AUX5}(\mathcal{Q}_i) \wedge \neg\mathsf{INNER}(\mathcal{Q}_{i-1})\right) . \quad (32)\end{aligned}$$

For the conditioned $\mathsf{AUX1}(\mathcal{Q}_i)$, because of the condition $|\mathcal{N}(\mathcal{Q}_{i-1})| = \nu$, any forward query can hit at most ν values, thus one query sets this event with a probability of at most

$$\frac{2\nu}{2^b - (i-1)} \leq \frac{4\nu}{2^b} ,$$

using that $q \leq 2^{b-1}$. Therefore,

$$\sum_{i} \mathbf{Pr}\left(\mathsf{AUX1}(\mathcal{Q}_i) \mid \neg\mathsf{INNER}(\mathcal{Q}_{i-1}) \wedge |\mathcal{N}(\mathcal{Q}_{i-1})| = \nu\right) \leq \frac{4q\nu}{2^b} . \quad (33)$$

For the conditioned $\mathsf{AUX2}(\mathcal{Q}) \vee \mathsf{AUX3}(\mathcal{Q})$, one query sets one of the targeted forward-inverse collisions with a probability of at most

$$\frac{3q}{2^b - (i-1)} \leq \frac{6q}{2^b} ,$$

using that $q \leq 2^{b-1}$. Moreover, due to the condition $|\mathcal{N}(\mathcal{Q})| = \nu$, the number of inverse queries that can set this event with non-zero probability is of at most ν. Therefore,

$$\mathbf{Pr}\left(\mathsf{AUX2}(\mathcal{Q}) \vee \mathsf{AUX3}(\mathcal{Q}) \mid \neg\mathsf{AUX1}(\mathcal{Q}) \wedge \neg\mathsf{AUX4}(\mathcal{Q}) \wedge \neg\mathsf{AUX5}(\mathcal{Q}) \wedge |\mathcal{N}(\mathcal{Q})| = \nu\right) \leq \frac{6q\nu}{2^b}\,. \quad (34)$$

For $\mathsf{AUX4}(\mathcal{Q}_i)$, there are two possibilities:

- $\mathsf{AUX4a}(\mathcal{Q}_i)$: $\mathsf{AUX4}(\mathcal{Q}_i)$ where the triggering query is of the form (x', y', inv) such that $\lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} = x'$;
- $\mathsf{AUX4b}(\mathcal{Q}_i)$: $\mathsf{AUX4}(\mathcal{Q}_i)$ where there exists a prior query (x'', y'', inv) such that $\lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} = x''$.

We define the random variables , $U_{\lceil \mathrm{fwd} \rceil^{(c-a)}}$, $U_{\lfloor \mathrm{fwd} \rfloor^{(a)}}$, and $U_{\lceil \mathrm{inv} \rceil^{(b-a)}}$ as follows:

$$U_{\lceil \mathrm{fwd} \rceil^{(c-a)}} = \begin{cases} \max\limits_{t \in \mathbb{F}_2^{c-a}} \left| \left\{ (x, y, \mathrm{fwd}) \in \mathcal{Q} : \lceil \lfloor y \rfloor^{(c)} \rceil^{(c-a)} = t \right\} \right| & \text{if } a \leq c\,, \\ q & \text{if } a > c\,, \end{cases}$$

$$U_{\lfloor \mathrm{fwd} \rfloor^{(a)}} = \max_{t \in \mathbb{F}_2^a} \left| \left\{ (x, y, \mathrm{fwd}) \in \mathcal{Q} : \lfloor y \rfloor^{(a)} = t \right\} \right|\,,$$

$$U_{\lceil \mathrm{inv} \rceil^{(b-a)}} = \max_{t \in \mathbb{F}_2^{b-a}} \left| \left\{ (x, y, \mathrm{inv}) \in \mathcal{Q} : \lceil x \rceil^{(b-a)} = t \right\} \right|\,.$$

We have

$$\mathbf{Ex}\left(U_{\lceil \mathrm{fwd} \rceil^{(c-a)}}\right) \leq \mathrm{mcol}(q, 2^{c-a})\,, \qquad \mathbf{Ex}\left(U_{\lfloor \mathrm{fwd} \rfloor^{(a)}}\right) \leq \mathrm{mcol}(q, 2^a)\,,$$
$$\mathbf{Ex}\left(U_{\lceil \mathrm{inv} \rceil^{(b-a)}}\right) \leq \mathrm{mcol}(q, 2^{b-a})\,.$$

For $\mathsf{AUX4a}(\mathcal{Q}_i)$ to occur with query (x', y', inv), it is necessary that

$$\lceil x' \rceil^{(b-a)} = \lceil y' \rceil^{(b-a)} \quad \text{and} \quad \lfloor x' \rfloor^{(c)} = \lfloor y \rfloor^{(c)}\,,$$

for some prior query (x, y, fwd). Let us first assume that $a < c$. The probability of this event can be decomposed into two:

- $\lceil x' \rceil^{(b-a)} = \lceil y' \rceil^{(b-a)}$, which holds with a probability of at most $\frac{2}{2^{b-a}}$;
- $\lfloor x' \rfloor^{(c)} = \lfloor y \rfloor^{(c)}$, conditioned on $\lceil \lfloor x' \rfloor^{(c)} \rceil^{(c-a)} = \lceil \lfloor y' \rfloor^{(c)} \rceil^{(c-a)} = \lceil \lfloor y \rfloor^{(c)} \rceil^{(c-a)}$. Conditioning further on $U_{\lceil \mathrm{fwd} \rceil^{(c-a)}} = u_{\mathrm{fwd}}$, there are at most u_{fwd} states y that share this specific value in their middle $c - a$ bits. For each such state, the probability of satisfying the equality is at most $\frac{2}{2^a}$. Therefore, the probability of this case is bounded by $\frac{2u_{\mathrm{fwd}}}{2^a}$.

When $a \geq c$, the same reasoning applies, except that the condition $\lceil \lfloor y' \rfloor^{(c)} \rceil^{(c-a)} = \lceil \lfloor y \rfloor^{(c)} \rceil^{(c-a)}$ in the second item is dropped, and u_{fwd} is fixed to q, so the resulting probabilities remain the same. Therefore,

$$\sum_i \mathbf{Pr}\left(\mathsf{AUX4a}(\mathcal{Q}_i) \wedge \neg\mathsf{INNER}(\mathcal{Q}_{i-1})\right) \leq \frac{4q}{2^b} \mathbf{Ex}\left(U_{\lceil \mathrm{fwd} \rceil^{(c-a)}}\right)\,.$$

To set $\mathsf{AUX4b}(\mathcal{Q}_i)$ with the query (x', y', inv), we first consider the regime $a < c$. In that case we require some $\gamma \geq 1$ such that

– there are γ different (x'', y'', inv) such that $\lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} = x''$;
– $\lfloor y \rfloor^{(c)} = \lfloor x' \rfloor^{(c)}$, conditioned on the fact that there are γ different (x'', y'', inv) such that $\lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} = x''$. This implies that there are at most γ possible values for $\lfloor y \rfloor^{(a)}$. Conditioning further on $U_{\lfloor \mathrm{fwd} \rfloor^{(a)}} = u_{\mathrm{fwd}}$, we know that at most u_{fwd} queries have their inner a bits equal to a given value. Hence, this case holds with a probability of at most $\frac{2\gamma u_{\mathrm{fwd}}}{2^{c-a}}$.

By summing over all possible $\gamma \geq 1$, the term γ above is replaced with the expected number of queries (x'', y'', inv) such that $\lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} = x''$. We condition on $U_{\lceil \mathrm{inv} \rceil^{(b-a)}} = u_{\mathrm{inv}}$, and obtain the upper bound $\frac{2u_{\mathrm{inv}}}{2^a}$. Thus,

$$\sum_i \mathbf{Pr}\left(\mathsf{AUX4b}(\mathcal{Q}_i) \wedge \neg \mathsf{INNER}(\mathcal{Q}_{i-1})\right) \leq \frac{4q \cdot \mathbf{Ex}\left(U_{\lfloor \mathrm{fwd} \rfloor^{(a)}}\right) \cdot \mathbf{Ex}\left(U_{\lceil \mathrm{inv} \rceil^{(b-a)}}\right)}{2^c}. \quad (35)$$

When $a \geq c$, we only consider the event that there exists a prior query (x'', y'', inv) such that $\lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} = x''$. This gives the bound $\frac{2q\mathbf{Ex}\left(U_{\lceil \mathrm{inv} \rceil^{(b-a)}}\right)}{2^a}$, and in that regime (35) is a valid (though loose) upper bound.

Therefore,

$$\begin{aligned}
&\sum_i \mathbf{Pr}\left(\mathsf{AUX4}(\mathcal{Q}_i) \wedge \neg \mathsf{INNER}(\mathcal{Q}_{i-1})\right) \\
&\quad \leq \frac{4q \cdot \mathbf{Ex}\left(U_{\lceil \mathrm{fwd} \rceil^{(c-a)}}\right)}{2^b} + \frac{4q \cdot \mathbf{Ex}\left(U_{\lfloor \mathrm{fwd} \rfloor^{(a)}}\right) \cdot \mathbf{Ex}\left(U_{\lceil \mathrm{inv} \rceil^{(b-a)}}\right)}{2^c} \\
&\quad \leq \frac{4q \cdot \mathrm{mcol}(q, 2^{c-a})}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^a) \cdot \mathrm{mcol}(q, 2^{b-a})}{2^c}. \quad (36)
\end{aligned}$$

Similarly, $\mathsf{AUX5}(\mathcal{Q}_i)$ can be decomposed into two:

– Either the triggering query (x', y', inv) satisfies $\lfloor x' \rfloor^{(c)} = \lfloor y' \rfloor^{(c)}$, triggering the edges $x' \longrightarrow y' \underset{a}{\overset{\mathsf{m}}{\Longrightarrow}} \lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)}$ with $\mathsf{m} = \lceil x' \rceil^{(r)} \oplus \lceil y' \rceil^{(r)}$. Summing over all queries, this event happens with a probability of at most $\frac{2q}{2^c}$;
– Or there exists a prior query (x'', y'', inv) with
  • $\lfloor y' \rfloor^{(c)} = \lfloor x'' \rfloor^{(c)}$, and
  • $x' = \lceil y'' \rceil^{(b-a)} \parallel \lfloor x'' \oplus y'' \rfloor^{(a)}$

  which triggers the edges $x \underset{a}{\Longrightarrow} x' \longrightarrow y' = x'' \oplus \mathsf{m} \| 0^c \underset{a}{\overset{\mathsf{m}}{\Longrightarrow}} \lceil y'' \rceil^{(b-a)} \parallel \lfloor x'' \oplus y'' \rfloor^{(a)} = x' \longrightarrow y'$. Similarly to the previous cases, let $U_{\lfloor \mathrm{inv} \rfloor^{(c)}}$ be the size of the largest multicollision over c bits, over all inverse queries. Conditioning on $U_{\lfloor \mathrm{inv} \rfloor^{(c)}} = u$, there are at most u candidate queries (x'', y'', inv), for each of them, the answer x' is equal to $\lceil y'' \rceil^{(b-a)} \parallel \lfloor x'' \oplus y'' \rfloor^{(a)}$ with a probability of at most $\frac{2u}{2^b}$.

Therefore,

$$\sum_i \mathbf{Pr}\left(\mathsf{AUX5}(\mathcal{Q}_i) \wedge \neg\mathsf{INNER}(\mathcal{Q}_{i-1})\right) \leq \frac{2q}{2^c} + \frac{2q \cdot \mathrm{mcol}(q, 2^c)}{2^b}\,.$$

Combining this equation, along with (33), (34), and (36), into (32) gives

$$\mathbf{Pr}\left(\mathsf{INNER}(\mathcal{Q})\right) \leq \frac{4q}{2^c} + \frac{10q \cdot \mathbf{Ex}\left(|\mathcal{N}(\mathcal{Q})|\right)}{2^b} + \frac{2q \cdot \mathrm{mcol}(q, 2^c)}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^{c-a})}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^a) \cdot \mathrm{mcol}(q, 2^{b-a})}{2^c}\,.$$

Plugging this equation, along with (29), (30), and (31), into (28) gives

$$\mathbf{Pr}\left(\mathsf{PRE}(\mathcal{Q})\right) \leq \frac{16q}{2^n} + \min\left\{\frac{4\beta q}{2^{n-r'}}, \frac{(6(b-a)+8)q}{2^a}, \frac{4q}{2^c} + \frac{10q \cdot \mathbf{Ex}\left(|\mathcal{N}(\mathcal{Q})|\right)}{2^b} + \frac{2q \cdot \mathrm{mcol}(q, 2^c)}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^{c-a})}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^a) \cdot \mathrm{mcol}(q, 2^{b-a})}{2^c}\right\}. \quad (37)$$

It now remains to upper bound the expected value of the size of $\mathcal{N}(\mathcal{Q})$, which we do in the following lemma.

Lemma 3. *We have*

$$\mathbf{Ex}\left(|\mathcal{N}(\mathcal{Q})|\right) \leq 2^{r+1} + \frac{2q \cdot \mathrm{mcol}(q, 2^{b-a})}{2^{a-r}} + \frac{2q^2 \cdot \mathrm{mcol}(2^a, 2^a)}{2^c}\,. \quad (38)$$

Proof. We have

$$|\mathcal{N}(\mathcal{Q})| = 2^r \times \left|\left\{Y \in \mathbb{F}_2^b \mid \mathcal{Q} \text{ defines a path } Y \underset{a}{\Longrightarrow} Y' \longrightarrow Y'' \text{ where both queries were made in the inverse direction}\right\}\right|.$$

Denote the set on the right-hand side of this equation by $\mathcal{N}^*(\mathcal{Q})$ (we will focus on the size of this set). Stated equivalently, $\mathcal{N}^*(\mathcal{Q})$ counts the number of distinct paths

$$Y \underset{a}{\Longrightarrow} Y' \longrightarrow Y'$$

with both queries made in the inverse direction. We partition $\mathcal{N}^*(\mathcal{Q})$ into three subsets, $\mathcal{N}_1^*(\mathcal{Q})$, $\mathcal{N}_2^*(\mathcal{Q})$, and $\mathcal{N}_3^*(\mathcal{Q})$, according to which query causes the path to be added to $\mathcal{N}^*(\mathcal{Q})$. The set $\mathcal{N}_1^*(\mathcal{Q})$ consists of paths where a single inverse query already yields the path, namely when the query is (x, y, inv) with $\lceil y \rceil^{(b-a)} \parallel \lfloor y \oplus x \rfloor^{(a)} = x$. The set $\mathcal{N}_2^*(\mathcal{Q})$ corresponds to the case where the triggering query is $Y \underset{a}{\Longrightarrow} Y'$, and $\mathcal{N}_3^*(\mathcal{Q})$ to the case where the triggering query is $Y' \longrightarrow Y''$. Hence,

$$|\mathcal{N}(\mathcal{Q})| = 2^r \left(|\mathcal{N}_1^*(\mathcal{Q})| + |\mathcal{N}_2^*(\mathcal{Q})| + |\mathcal{N}_3^*(\mathcal{Q})|\right). \quad (39)$$

We start by upper bounding the expected size of $\mathcal{N}_1^*(\mathcal{Q})$. One query can only increment the count by one, and must be of the form (x, y, inv) with $y = \lceil x \rceil^{(b-a)} \parallel 0^a$. Define by I_i the Bernoulli variable equal to 1 if and only if the ith query is of this form. We have

$$\mathbf{Pr}\left(\mathsf{I}_i = 1\right) \leq \frac{2^a}{2^b - (i-1)} \leq \frac{2}{2^{b-a}}\,,$$

using that $q \leq 2^{b-1}$. Moreover, due to the restriction $\lfloor y \rfloor^{(a)} = 0^a$, there are at most 2^{b-a} indexes with $\mathsf{I}_i \neq 0$. Therefore,

$$\mathbf{Ex}\left(|\mathcal{N}_1^*(\mathcal{Q})|\right) = \mathbf{Ex}\left(\sum_i \mathsf{I}_i\right) = \sum_i \mathbf{Pr}\left(\mathsf{I}_i = 1\right) \leq 2^{b-a} \times \frac{2}{2^{b-a}} = 2\,. \tag{40}$$

For $\mathcal{N}_2^*(\mathcal{Q})$, one single inverse query can also increase the count by at most one. To analyze this, we reuse the random variable

$$U_{\lceil \mathrm{inv} \rceil^{(b-a)}} = \max_{t \in \mathbb{F}_2^{b-a}} \left| \left\{ (x, y, \mathrm{inv}) \in \mathcal{Q} : \lceil x \rceil^{(b-a)} = t \right\} \right|\,,$$

where we have

$$\mathbf{Ex}\left(U_{\lceil \mathrm{inv} \rceil^{(b-a)}}\right) \leq \mathrm{mcol}(q, 2^{b-a}).$$

Let $\mathsf{J}_{i|u}$ the Bernoulli variable equal to 1 if and only if the query number i is of the form (x, y, inv) and there exists a prior query (x', y', inv) with $\lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} = x'$, conditioned on $U_{\lceil \mathrm{inv} \rceil^{(b-a)}} = u$ for $u \in \{1, \ldots, q\}$. Then,

$$\mathbf{Pr}\left(\mathsf{J}_{i|u} = 1\right) \leq \frac{2^{b-a} \cdot u}{2^b - (i-1)} \leq \frac{2u}{2^a}\,,$$

using that $q \leq 2^{b-1}$. Therefore,

$$\mathbf{Ex}\left(|\mathcal{N}_2^*(\mathcal{Q})| \mid U_{\lceil \mathrm{inv} \rceil^{(b-a)}} = u\right) = \mathbf{Ex}\left(\sum_i \mathsf{J}_{i|u}\right) = \sum_i \mathbf{Pr}\left(\mathsf{J}_{i|u} = 1\right) \leq \frac{2uq}{2^a}\,,$$

so that

$$\mathbf{Ex}\left(|\mathcal{N}_2^*(\mathcal{Q})|\right) \leq \frac{2 \cdot \mathbf{Ex}\left(U_{\lceil \mathrm{inv} \rceil^{(b-a)}}\right) q}{2^a} \leq \frac{2q \cdot \mathrm{mcol}(q, 2^{b-a})}{2^a}\,. \tag{41}$$

For $\mathcal{N}_3^*(\mathcal{Q})$, the situation is more complex, since there may exist multiple edges

$$Y_1 \underset{a}{\Longrightarrow} Y' \text{ and } Y_2 \underset{a}{\Longrightarrow} Y' \text{ with } Y_1 \neq Y_2\,,$$

so a single query $Y' \longrightarrow Y''$ can increase the count by more than one. To address this, we partition $\mathcal{N}_3^*(\mathcal{Q})$ into 2^{b-a} sets, denoted $\mathcal{N}_3^*(\mathcal{Q})[\gamma]$ for $\gamma \in \{1, \ldots, 2^{b-a}\}$.

Each set $\mathcal{N}_3^*(\mathcal{Q})[\gamma]$ consists of elements in $\mathcal{N}_3^*(\mathcal{Q})$ where the edge $Y \underset{a}{\Longrightarrow} Y'$ satisfies $\lceil Y' \rceil^{(b-a)} = \gamma$. Let q_γ denote the number of inverse queries whose inputs have their outer $b-a$ bits equal to γ. Then, we define Ξ_γ to be the maximum multicollision size of $\lfloor x \oplus y \rfloor^{(a)}$, over all (x, y, inv) with $\lceil y \rceil^{(b-a)} = \gamma$. Note that given the restriction $\lceil y \rceil^{(b-a)} = \gamma$, there are at most $\min\{q_\gamma, 2^a\} \le 2^a$ candidates (x, y, inv), so that

$$\mathbf{Ex}\left(\Xi_\gamma\right) \le \mathrm{mcol}(2^a, 2^a)\,. \tag{42}$$

We then have

$$\begin{aligned}\mathbf{Ex}\left(\mathcal{N}_3^*(\mathcal{Q})\right) &= \sum_{\gamma=1}^{2^{b-a}} \mathbf{Ex}\left(\mathcal{N}_3^*(\mathcal{Q})[\gamma]\right) \\ &= \sum_{\gamma=1}^{2^{b-a}} \sum_{\xi_\gamma=1}^{2^a} \mathbf{Ex}\left(\mathcal{N}_3^*(\mathcal{Q})[\gamma] \mid \Xi_\gamma = \xi_\gamma\right) \cdot \mathbf{Pr}\left(\Xi_\gamma = \xi_\gamma\right)\,. \end{aligned} \tag{43}$$

Now, conditioned on $\Xi_\gamma = \xi_\gamma$, to contribute to the set $\mathcal{N}_3^*(\mathcal{Q})[\gamma]$, each inverse query can connect to at most q_γ inverse queries and each connection increments the count of this set to at most ξ_γ. Hence, using that $q \le 2^{b-1}$,

$$\mathbf{Ex}\left(\mathcal{N}_3^*(\mathcal{Q})[\gamma] \mid \Xi_\gamma = \xi_\gamma\right) \le \sum_i \frac{q_\gamma \xi_\gamma}{2^b - (i-1)} \le \frac{2q q_\gamma \xi_\gamma}{2^b}\,.$$

Finally, plugging this into (43) and using (42), we obtain

$$\begin{aligned}\mathbf{Ex}\left(\mathcal{N}_3^*(\mathcal{Q})\right) &\le \sum_{\gamma=1}^{2^{b-a}} \frac{2q q_\gamma \cdot \mathbf{Ex}\left(\Xi_\gamma\right)}{2^b} \\ &\le \sum_{\gamma=1}^{2^{b-a}} \frac{2q q_\gamma \cdot \mathrm{mcol}(2^a, 2^a)}{2^b} \le \frac{2q^2 \cdot \mathrm{mcol}(2^a, 2^a)}{2^b}\,. \end{aligned} \tag{44}$$

Combining (40), (41), and (44) into (39), this concludes the proof. □

Combining (38) into (37), we obtain

$$\mathbf{Pr}\left(\mathsf{PRE}(\mathcal{Q})\right) \le \frac{16q}{2^n} + \min\left\{ \frac{4\beta q}{2^{n-r'}}, \frac{(6(b-a)+8)q}{2^a}, \frac{24q}{2^c} + \frac{20q^2 \cdot \mathrm{mcol}(q, 2^{b-a})}{2^{c+a}} + \right.$$
$$\left. \frac{2q \cdot \mathrm{mcol}(q, 2^c)}{2^b} + \frac{20q^3 \cdot \mathrm{mcol}(2^a, 2^a)}{2^{b+c}} + \frac{4q \cdot \mathrm{mcol}(q, 2^{c-a})}{2^b} + \frac{4q \cdot \mathrm{mcol}(q, 2^a) \cdot \mathrm{mcol}(q, 2^{b-a})}{2^c} \right\},$$

hence the theorem.

E.3 Proof of Second Preimage Resistance of SPONGE-EDM^a (Theorem 3)

The extension to SPONGE-EDM^a is, as in the proof of preimage resistance, technically more involved. As in the extension to the proof of indifferentiability (Section E.1.5), the double edges $y \overset{\mathsf{m}}{\Longrightarrow} y'$ are replaced by the following family of edges. For each query $(x, y, \mathrm{dir}) \in \mathcal{Q}$ and every $\mathsf{m} \in \mathbb{F}_2^r$, we add the edge

$$x \oplus (\mathsf{m} \| 0^c) \overset{\mathsf{m}}{\underset{a}{\Longrightarrow}} \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} .$$

The first preimage consists of $2\alpha + \lceil n/r' \rceil - 1$ permutation calls that $\mathcal{A}$ gets for free. These result in the following states:

$$0^b \overset{\mathsf{m}_1}{\underset{a}{\Longrightarrow}} \bar{y}'_1 \longrightarrow \bar{y}''_1 \cdots \overset{\mathsf{m}_{\alpha-1}}{\underset{a}{\Longrightarrow}} \bar{y}'_{\alpha-1} \longrightarrow \bar{y}''_{\alpha-1} \overset{\mathsf{m}_\alpha}{\underset{a}{\Longrightarrow}} \bar{y}'_\alpha \longrightarrow \bar{y}_1 \longrightarrow \cdots \underset{a}{\Longrightarrow} \bar{y}_\beta .$$

The event $\mathsf{SEC}(\mathcal{Q})$ would now be defined as:

$\mathsf{SEC}(\mathcal{Q})$: $\mathcal{Q}$ defines a path

$$0^b \overset{\mathsf{m}'_1}{\underset{a}{\Longrightarrow}} y'_1 \longrightarrow y''_1 \cdots \overset{\mathsf{m}'_{\alpha'-1}}{\underset{a}{\Longrightarrow}} y'_{\alpha'-1} \longrightarrow y''_{\alpha'-1} \overset{\mathsf{m}'_{\alpha'}}{\underset{a}{\Longrightarrow}} y'_{\alpha'} \longrightarrow y_1 \longrightarrow \cdots \underset{a}{\Longrightarrow} y_\beta$$
$$\text{such that } y_i \in \boldsymbol{Z}_i \text{ for } i = 1, \ldots, \beta \text{ and } \mathrm{padF}(\mathsf{msg}) \neq \mathsf{m}'_1 \| \cdots \| \mathsf{m}'_{\alpha'} .$$

We again define the set $\mathcal{N}(\mathcal{Q})$ as follows:

$$\mathcal{N}(\mathcal{Q}) = \Big\{ Y \in \mathbb{F}_2^b \mid \mathcal{Q} \text{ defines a path } Y \overset{\mathsf{m}}{\underset{a}{\Longrightarrow}} Y' \longrightarrow Y'' \text{ where both queries were made in the inverse direction} \Big\} .$$

Moreover, define the set $\bar{\mathcal{N}}(\mathcal{Q})$ as follows:

$$\bar{\mathcal{N}}(\mathcal{Q}) = \mathcal{N}(\mathcal{Q}) \cup \Big\{ X \oplus (\mathsf{m} \| 0^c) \in \mathbb{F}_2^b \mid \mathsf{m} \in \mathbb{F}_2^r \text{ and } \exists (X, Y, \mathrm{inv}) \in \mathcal{Q} \text{ where } \lceil Y \rceil^{(b-a)} \parallel \lfloor X \oplus Y \rfloor^{(a)} = \bar{y}'_A \text{ for some } A \leq \alpha \Big\} .$$

Thus, $\bar{\mathcal{N}}(\mathcal{Q})$ is the union of the set $\mathcal{N}(\mathcal{Q})$ and the elements Y such that there exists an edge $Y \overset{\mathsf{m}}{\underset{a}{\Longrightarrow}} Y'$ for some $Y' = \bar{y}'_A$, with the query made in the inverse direction. Note that, compared to the proof for first preimage resistance, we omit a bad event similar to $\mathsf{BAD\text{-}INV}(\mathcal{Q})$, since looking ahead, $\mathsf{CONNECT}(\mathcal{Q})$ does not include such an event, and adding it in $\mathsf{BAD}(\mathcal{Q})$ would not make the bound tighter. The bad event $\mathsf{BAD}(\mathcal{Q})$ is split as follows:

$$\begin{aligned}
\mathsf{BAD\text{-}FWD}(\mathcal{Q}) : \exists (x, y, \mathrm{fwd}) \in \mathcal{Q} \text{ such that either } & x \in f^{-1}(\mathcal{S}) \setminus \{\bar{y}'_\alpha\} \\
\text{or } & (x, \lceil y \rceil^{(b-a)} \parallel \lfloor y \oplus x \rfloor^{(a)}) \neq (\bar{y}''_{\alpha-1} \oplus (\mathsf{m}_\alpha \| 0^c), \bar{y}'_\alpha) \\
\text{and } & \lceil y \rceil^{(b-a)} \parallel \lfloor y \oplus x \rfloor^{(a)} \in f^{-1}(\mathcal{S}) , \\
\mathsf{BAD\text{-}INNER}(\mathcal{Q}) : \exists (x, y, \mathrm{fwd}) \in \mathcal{Q}, z \in \bar{\mathcal{N}}(\mathcal{Q}) & \text{ such that } y = z
\end{aligned}$$

$$
\begin{aligned}
&\text{or } \exists (x, y, \mathrm{fwd}), (x', y', \mathrm{inv}) \in \mathcal{Q}, \text{ such that } y' \in \bar{\mathcal{N}}(\mathcal{Q}) \text{ and} \\
&\lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} = x' \\
&\text{or } \exists (x, y, \mathrm{inv}) \in \mathcal{Q} \text{ such that } \lfloor x \rfloor^{(c)} = 0^c \,, \\
\mathsf{BAD\text{-}FF}(\mathcal{Q}) &: \exists (x, y, \mathrm{inv}) \in \mathcal{Q} \text{ such that } \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} \in f^{-1}(\mathcal{S}) \,, \\
\mathsf{BAD}(\mathcal{Q}) &: \mathsf{BAD\text{-}FWD}(\mathcal{Q}) \vee (\mathsf{BAD\text{-}INNER}(\mathcal{Q}) \wedge \mathsf{BAD\text{-}FF}(\mathcal{Q})) \,.
\end{aligned}
$$

Because the adversary makes no redundant queries and the first preimage is derived solely from forward queries, trivial wins via inverse queries do not appear, in particular the events $\mathsf{BAD\text{-}INNER}(\mathcal{Q})$ and $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ remain untouched.

Bad event $\mathsf{CONNECT}(\mathcal{Q})$ considers an attack connecting an absorption path from 0^b to any of the states $0^b, \bar{y}''_1, \ldots, \bar{y}''_{\alpha-1}$ on their inner part or any of the states $f^{-1}(0^b), \bar{y}'_1, \ldots, \bar{y}'_{\alpha-1}$. It is split into three cases:

- The connecting query is made in the forward direction. This is captured by the bad event $\mathsf{CON\text{-}FWD}(\mathcal{Q})$ below;
- The connecting query is $(y, \bar{y}''_A \oplus (\mathsf{m} \| 0^c), \mathrm{inv})$, and it connects with some double edge $y' \overset{\mathsf{m}}{\underset{a}{\Longrightarrow}} y$ that was triggered by a query in the forward direction. This is captured by the bad event $\mathsf{CON\text{-}INV}(\mathcal{Q})$ below;
- Otherwise, then necessarily the adversary must construct a path $0^b \overset{\mathsf{m}'_1}{\underset{a}{\Longrightarrow}} y'_1 \longrightarrow y''_1 \cdots \overset{\mathsf{m}'_A}{\underset{a}{\Longrightarrow}} y'_A$ in the forward direction that connects to the set $\bar{\mathcal{N}}(\mathcal{Q})$. This is captured by the aforementioned bad event $\mathsf{BAD\text{-}INNER}(\mathcal{Q})$.

The bad event $\mathsf{CONNECT}(\mathcal{Q})$ is split as follows:

$$
\begin{aligned}
\mathsf{CON\text{-}FWD}(\mathcal{Q}) &: \exists (x, y, \mathrm{fwd}) \in \mathcal{Q} \text{ such that either } \lceil y \rceil^{(b-a)} \| \lfloor y \oplus x \rfloor^{(a)} = f^{-1}(0^b) \,, \\
&\text{or } \exists A \leq \alpha \text{ with } (x, \lceil y \rceil^{(b-a)} \parallel \lfloor y \oplus x \rfloor^{(b)}) \neq (\bar{y}''_{A-1} \oplus (\mathsf{m}_A \| 0^c), \bar{y}'_A) \\
&\qquad\qquad \text{and } \lceil y \rceil^{(b-a)} \parallel \lfloor y \oplus x \rfloor^{(a)} = \bar{y}'_A \,, \\
&\text{or } \lfloor y \rfloor^{(c)} = 0^c \,, \\
&\text{or } \exists A < \alpha \text{ with } (x, y) \neq (\bar{y}'_A, \bar{y}''_A) \text{ and } \lfloor y \rfloor^{(c)} = \lfloor \bar{y}''_A \rfloor^{(c)} \,, \\
\mathsf{CON\text{-}INV}(\mathcal{Q}) &: \exists (x, y, \mathrm{inv}), (x', y', \mathrm{fwd}) \in \mathcal{Q} \text{ with } \lfloor y \rfloor^{(c)} \in \{0^c, \lfloor \bar{y}''_1 \rfloor^{(c)}, \ldots, \lfloor \bar{y}''_{\alpha-1} \rfloor^{(c)}\} \\
&\text{such that } x = \lceil y' \rceil^{(b-a)} \parallel \lfloor y' \oplus x' \rfloor^{(a)} \,, \\
\mathsf{CON\text{-}FF}(\mathcal{Q}) &: \exists (x, y, \mathrm{inv}) \in \mathcal{Q}, A \leq \alpha \text{ such that } \lceil y \rceil^{(b-a)} \parallel \lfloor x \oplus y \rfloor^{(a)} = \bar{y}'_A \,, \\
&\text{or } \exists (x, y, \mathrm{inv}), (x', y', \mathrm{inv}) \in \mathcal{Q} \text{ with } \lfloor y \rfloor^{(c)} \in \{0^c, \lfloor \bar{y}''_1 \rfloor^{(c)}, \ldots, \lfloor \bar{y}''_{\alpha-1} \rfloor^{(c)}\} \,, \\
&\qquad \text{and } x = \lceil y' \rceil^{(b-a)} \parallel \lfloor x' \oplus y' \rfloor^{(a)} \,, \\
&\text{or } \exists (x, y, \mathrm{inv}) \in \mathcal{Q} \text{ such that } \lfloor x \rfloor^{(c)} = 0^c \,, \\
\mathsf{CONNECT}(\mathcal{Q}) &: \mathsf{CON\text{-}FWD}(\mathcal{Q}) \vee \mathsf{CON\text{-}INV}(\mathcal{Q}) \vee (\mathsf{BAD\text{-}INNER}(\mathcal{Q}) \wedge \mathsf{CON\text{-}FF}(\mathcal{Q})) \,.
\end{aligned}
$$

Similarly to the analysis of first preimage resistance (Section E.2), the bound of $\mathsf{CON\text{-}FF}(\mathcal{Q})$ will be tighter than that of $\mathsf{INNER}(\mathcal{Q})$ when $a > c$. We have

$$
\mathsf{SEC}(\mathcal{Q}) \implies \mathsf{BAD}(\mathcal{Q}) \vee \mathsf{CONNECT}(\mathcal{Q}) \,,
$$

and thus that

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{SEC}(\mathcal{Q})\right) \leq\ & \mathbf{Pr}\left(\mathsf{BAD\text{-}FWD}(\mathcal{Q})\right) + \mathbf{Pr}\left(\mathsf{CON\text{-}FWD}(\mathcal{Q})\right) + \\ & \mathbf{Pr}\left(\mathsf{CON\text{-}INV}(\mathcal{Q})\right) + \min\left\{\mathbf{Pr}\left(\mathsf{BAD\text{-}INNER}(\mathcal{Q})\right),\right. \\ & \left.\mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q})\right) + \mathbf{Pr}\left(\mathsf{CON\text{-}FF}(\mathcal{Q})\right)\right\}.\end{aligned}$$

For $\mathsf{BAD\text{-}FWD}(\mathcal{Q})$, the analysis of Section 5.3 carries over, but with $\mathbf{Ex}\left(|\mathcal{S}|\right) \leq 2\frac{2^b}{2^n} + 1$.

$$\mathbf{Pr}\left(\mathsf{BAD\text{-}FWD}(\mathcal{Q})\right) \leq \frac{4\bar{q}'}{2^b} + \frac{8\bar{q}'}{2^n},$$

For $\mathsf{CON\text{-}FWD}(\mathcal{Q})$, the analysis of Section 5.4 applies, so that

$$\mathbf{Pr}\left(\mathsf{CON\text{-}FWD}(\mathcal{Q})\right) \leq \frac{4\alpha\bar{q}'}{2^c}.$$

For $\mathsf{CON\text{-}INV}(\mathcal{Q})$, we can use that there are at most $2^r \cdot \alpha$ inverse queries (x, y, inv) with $\lfloor y \rfloor^{(c)} = \lfloor \bar{y}''_A \rfloor^{(c)}$, and obtain

$$\mathbf{Pr}\left(\mathsf{CON\text{-}INV}(\mathcal{Q})\right) \leq \frac{4\alpha\bar{q}'}{2^c}.$$

For $\mathsf{BAD\text{-}INNER}(\mathcal{Q})$, note that the intermediate states $Y' = \bar{y}'_\alpha$ and $f^{-1}(0^b)$ are randomly generated. Hence, the analysis made in Section E.2 applies, yielding among others the same upper bound on the expected size of $\bar{\mathcal{N}}(\mathcal{Q})$, with a factor 2 loss. Therefore,

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{INNER}(\mathcal{Q})\right) \leq\ & \frac{48\bar{q}'}{2^c} + \frac{40(\bar{q}')^2 \cdot \mathrm{mcol}(\bar{q}', 2^{b-a})}{2^{c+a}} + \frac{4\bar{q}' \cdot \mathrm{mcol}(\bar{q}', 2^c)}{2^b} + \\ & \frac{40(\bar{q}')^3 \cdot \mathrm{mcol}(2^a, 2^a)}{2^{b+c}} + \frac{8\bar{q}' \cdot \mathrm{mcol}(\bar{q}', 2^{c-a})}{2^b} + \frac{8\bar{q}' \cdot \mathrm{mcol}(\bar{q}', 2^a) \cdot \mathrm{mcol}(\bar{q}', 2^{b-a})}{2^c}.\end{aligned} \tag{45}$$

The bounding of $\mathsf{BAD\text{-}FF}(\mathcal{Q})$ also remains unchanged, so that

$$\mathbf{Pr}\left(\mathsf{BAD\text{-}FF}(\mathcal{Q})\right) \leq \frac{8\bar{q}'}{2^n} + \frac{(6(b-a)+8)\bar{q}'}{2^a}.$$

Finally, for $\mathsf{CON\text{-}FF}(\mathcal{Q})$, the first case means that a bits of the output of an inverse query connect to one of the α states, hence it occurs with a probability of at most $\frac{2\alpha\bar{q}'}{2^a}$. Regarding the second case, we again use that there are at most $2^r \cdot \alpha$ inverse queries (x, y, inv) with $\lfloor y \rfloor^{(c)} = \lfloor \bar{y}''_A \rfloor^{(c)}$. We can apply a multicollision analysis similar to the approach used throughout Section E.2. We obtain

$$\mathbf{Pr}\left(\mathsf{CON\text{-}FF}(\mathcal{Q})\right) \leq \frac{2\alpha\bar{q}'}{2^a} + \frac{4\alpha\bar{q}'}{2^c} + \frac{2\bar{q}' \cdot \mathrm{mcol}(\alpha 2^r, 2^{b-a})}{2^a}.$$

Hence, we obtain

$$\mathbf{Pr}\left(\mathsf{SEC}(\mathcal{Q})\right) \leq \frac{4\bar{q}'}{2^b} + \frac{16\bar{q}'}{2^n} + \frac{12\alpha\bar{q}'}{2^c} + \\ \min\left\{(45), \frac{(6(b-a)+8)\bar{q}'}{2^a} + \frac{2\alpha\bar{q}'}{2^a} + \frac{2\bar{q}' \cdot \mathrm{mcol}(\alpha 2^r, 2^{b-a})}{2^a}\right\}.$$