Cryptanalysis of ESSENCE*

María Naya-Plasencia[1], Andrea Röck[2,†], Jean-Philippe Aumasson[3,‡], Yann Laigle-Chapuy[1], Gaëtan Leurent[4], Willi Meier[5,§], and Thomas Peyrin[6]

[1] INRIA project-team SECRET, France
[2] Aalto University School of Science and Technology, Finland
[3] Nagravision SA, Cheseaux, Switzerland
[4] École Normale Supérieure, Paris, France
[5] FHNW, Windisch, Switzerland
[6] Ingenico, France

Abstract. ESSENCE is a hash function submitted to the NIST Hash Competition that stands out as a hardware-friendly and highly parallelizable design. Previous analysis showed some non-randomness in the compression function which could not be extended to an attack on the hash function and ESSENCE remained unbroken. Preliminary analysis in its documentation argues that it resists standard differential cryptanalysis. This paper disproves this claim, showing that advanced techniques can be used to significantly reduce the cost of such attacks: using a manually found differential characteristic and an advanced search algorithm, we obtain collision attacks on the full ESSENCE-256 and ESSENCE-512, with respective complexities $2^{67.4}$ and $2^{134.7}$. In addition, we show how to use these attacks to forge valid (message, MAC) pairs for HMAC-ESSENCE-256 and HMAC-ESSENCE-512, essentially at the same cost as a collision.

Keywords: cryptanalysis, hash functions, SHA-3

1 Introduction

Since the results [1–4] on the two most deployed hash functions, MD5 and SHA-1, recent years have seen a surge of research on cryptographic hashing. The consequent lack of confidence in the current NIST standard SHA-2 [5], stemming from its similarity with those algorithms, motivated NIST to launch the *NIST Hash Competition*, a public competition to develop a new hash standard, which will be called SHA-3 and announced by 2012 [6, 7]. NIST received 64 submissions,

*This work is supported in part by European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II and by the French Agence Nationale de la Recherche under contract ANR-06-SETI-013-RAPIDE.

†The work was started during my PhD at INRIA project-team SECRET, France.

‡Work done while this author was with FHNW, Switzerland, and supported by the Swiss National Science Foundation under project no. 113329.

§Supported by GEBERT RÜF STIFTUNG, project no. GRS-069/07.

accepted 51 as first round candidates, and in July 2009, selected 14 second round candidates [7, 8]. That competition catches the attention not only from many academics, but also from industry—with candidates from IBM, Hitachi, Intel, Sony—and from governmental organizations.

ESSENCE [9, 10] was a first round candidate in the NIST Hash Competition that like many others has two main instances, operating on 32- and 64-bit words, respectively: ESSENCE-256 and ESSENCE-512. These functions process messages using a binary tree structure, and use a simple compression algorithm based on two nonlinear feedback shift registers (NFSR's).

This paper presents collision attacks on the full hash functions ESSENCE-256 and ESSENCE-512. At the heart of our attacks is a single differential characteristic, found manually. Our main technical achievement is an original method for searching inputs conforming to this characteristic at a reduced cost. Supplementary, we describe how to use these attacks for forging valid message/MAC pairs for HMAC-ESSENCE-256 and HMAC-ESSENCE-512 in far fewer than $2^{n/2}$ trials. These findings show that ESSENCE does not satisfy the security requirements set by NIST for the future SHA-3.

In a parallel work, Mouha et al. [11] presented results on reduced versions of ESSENCE, including a pseudo-collision attack on ESSENCE-512 reduced to 31 steps. They exploited a differential characteristic of a different type than ours, and also use different techniques to search for conforming inputs. Preprints of [11] and of the present paper were published simultaneously in June 2009, and in July ESSENCE was not selected as a second round candidate by NIST.

The rest of the paper is organized as follows: §2 briefly introduces ESSENCE; §3 describes our method for searching collisions and its complexity analysis; §4 shows how to attack the HMAC construction when instantiated with ESSENCE, and finally §5 concludes.

2 Brief description of ESSENCE

We give a brief description of the ESSENCE hash functions, which should be sufficient to understand our attacks. A complete specification can be found in [9, 10]. Henceforth statements of (non) linearity are with respect to the field $\mathrm{GF}(2) = \{0, 1\}$ and its extensions.

2.1 Structure

ESSENCE processes a message by constructing a balanced binary tree of bounded depth whose leaves correspond to calls to a compression function with message chunks as input. The size of the message chunks and the height of the tree are tunable parameters. More precisely, each leaf corresponds to a hash done by a Merkle Damgård (MD) construction [12, 13] and a unique initial value for each leaf that depends on several parameters of the hash function. Likewise, nodes correspond to a combination by a MD construction of the childrens chaning value and a unique IV.

After creation of all tree roots, one appends a *final block* to the data to be hashed. This block contains parameters of the function as well as message-dependent information, and it potentially assists to prevent near-collision attacks.

2.2 Compression Function

The compression function of ESSENCE takes as input an eight-word chaining value and an eight-word message block. Words are 32-bit for ESSENCE-256 and 64-bit for ESSENCE-512, so blocks are respectively 256- and 512-bit. Versions of ESSENCE with 224- and 384-bit digests are derived from the main instances by tweaking parameters and truncation of the final digest.

The compression function uses two NFSR's, each operating on a register of eight *words*:

- $r = (r0, \ldots, r7)$ is initialized with the chaining value, and
- $k = (k0, \ldots, k7)$ is initialized with the message block.

At each step of the compression algorithm, the mechanism in Fig. 1 is clocked using a nonlinear bitwise function F (see Fig. 2), and a linear function L that provides diffusion across word slices.

Let us consider the feedback in more details for the example of the register handling the message. The word $k7$ is combined by XOR with $F(k6, k5, k4, k3, k2, k1, k0)$ and $L(k0)$. Thus, the nonlinear function F is influenced by the seven words $k0, \ldots, k6$, any difference in $k7$ is forwarded directly and any difference δ in $k0$ is transformed into a difference $L(\delta)$. The register initialized by the chaining value employes almost the same feedback function. The only difference is that at each step we combine in addition the word $k7$ from the second register.

The documentation of ESSENCE recommends at least 24 steps, and sets 32 steps in the actual submission for extra precaution [10, §4]. The whole mechanism defines a permutation and the compression function returns as new chaining value the XOR of the r register with its initial value, as in the Davies-Meyer scheme.

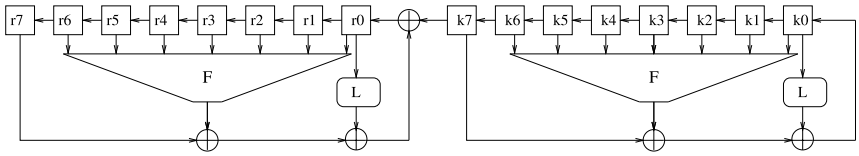

Fig. 1. Overview of the ESSENCE compression function logic.

Table 1. Differential characteristic for finding collisions on (both versions of) ESSENCE; α, β and γ are differences such that $\beta = L(\alpha)$, $\gamma = L(\beta)$ and $\alpha \vee \beta \vee \gamma = \alpha \vee \beta$. A "$\cdot$" denotes an absence of difference. Values in the column "Pr" are *heuristic approximations* of the probability to reach the *next* difference (exact probabilities significantly differ, and can be estimated empirically, cf. §3.3).

Pr.	Chaining value part									Message part								Pr.
	$r7$	$r6$	$r5$	$r4$	$r3$	$r2$	$r1$	$r0$		$k7$	$k6$	$k5$	$k4$	$k3$	$k2$	$k1$	$k0$	
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	0	α	β	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\beta\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	1	β	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	2	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	3	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	4	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	5	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	6	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	7	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
1	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	8	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	9	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$2^{-\lvert\alpha\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	10	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	β	$2^{-\lvert\alpha\vee\beta\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	11	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	β	$\cdot$	$2^{-\lvert\alpha\vee\beta\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	12	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	β	$\cdot$	$\cdot$	$2^{-\lvert\alpha\vee\beta\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	13	$\cdot$	$\cdot$	$\cdot$	α	β	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\vee\beta\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	14	$\cdot$	$\cdot$	α	β	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\vee\beta\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	15	$\cdot$	α	β	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\vee\beta\rvert}$
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	16	α	β	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\beta\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	17	β	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	18	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	19	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	20	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	21	$\cdot$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	22	$\cdot$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
$2^{-\lvert\alpha\rvert}$	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	23	$\cdot$	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$2^{-\lvert\alpha\rvert}$
1	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	24	α	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	25	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	26	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	?	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	27	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	?	?	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	28	$\cdot$	$\cdot$	$\cdot$	$\cdot$	α	?	?	?	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	29	$\cdot$	$\cdot$	$\cdot$	α	?	?	?	?	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	30	$\cdot$	$\cdot$	α	?	?	?	?	?	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	31	$\cdot$	α	?	?	?	?	?	?	1
1	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	32	α	?	?	?	?	?	?	?	1

$$
\begin{aligned}
F(a,b,c,d,e,f,g) = \; & abcdefg + abcdef + abcefg + acdefg + abceg + \\
& abdef + abdeg + abefg + acdef + acdfg + acefg + \\
& adefg + bcdfg + bdefg + cdefg + abcf + abcg + \\
& abdg + acdf + adef + adeg + adfg + bcde + \\
& bceg + bdeg + cdef + abc + abe + abf + abg + \\
& acg + adf + adg + aef + aeg + bcf + bcg + bde + \\
& bdf + beg + bfg + cde + cdf + def + deg + dfg + \\
& ad + ae + bc + bd + cd + ce + df + dg + ef + fg + \\
& a + b + c + f + 1
\end{aligned}
$$

Fig. 2. The F function of ESSENCE, which takes seven words as input and operates in a bit sliced way (that is, the i-th bit of the output word only depends on the i-th bits of the input words).

3 Collision Attacks on ESSENCE

Table 1 presents a differential characteristic for finding collisions on the compression function of ESSENCE. It is used for both ESSENCE-256 and ESSENCE-512. We found this characteristic manually, i.e., without the assistance of any automated search. Because it has no input difference in the chaining value, it can directly be used for searching colliding message blocks with respect to the same chaining value. The collision attack will then consist of

1. Finding one message block that fulfills the characteristic on the right part.
2. Trying chaining values until one conforms to the characteristic on the left part.

For the second phase of the attack, distinct pseudorandom chaining values are obtained by picking a first pseudorandom (sequence of) message block(s), and then checking differences after the insertion of the next message block. This allows us to find a collision for the full hash function.

The subsequent sections work out the details of the attack as follows:

- §3.1 explains how the characteristic works.
- §3.2 presents an efficient method for finding a message block that conforms to the characteristic.
- §3.3 discusses computation of the complexity; contrary to many similar differential attacks, an approximation solely based on Hamming weights is insufficient to obtain accurate probability estimates. Actually such heuristics *underestimate* the actual complexity of the basic attack, as we will see later.

Thereafter we use the following notations: $\vee$ for logical OR between two bits (or two words); $\wedge$ for logical AND; $\neg$ for bitwise negation; $|w|$ for the Hamming weight of word w; w_i for the i-th bit of word w, $0 \leq i < 32$ for ESSENCE-256, and $0 \leq i < 64$ for ESSENCE-512.

3.1 The Differential Characteristic

The differential characteristic in Table 1 starts with a difference in the message block, and no difference in the chaining value. To follow the characteristic, the only assumption that we make is that the function F will "absorb" certain differences (actually most of them) and "preserve" some others (at step 11). Therefore, the probability that a randomly chosen input conforms to the differential characteristic essentially depends on the Hamming weight of the word wise differences α and $\beta = L(\alpha)$. Critical steps are listed below:

- **Step 0**: α is fed back to $r0$ via an XOR and it does not enter F, unlike β. To ensure that no difference appears in the output of F, we need all the $|\beta|$ bit differences be absorbed, which is expected to occur with probability $2^{-|\beta|}$ (such heuristic estimates should not be used systematically, as discussed later).
- **Step 1**: the relation $\beta = L(\alpha)$ makes differences introduced in $r0$ vanish. This always works, but we also need that α adds no difference, that is, F needs to absorb $|\alpha|$ bit differences, thus the probability $2^{-|\alpha|}$ on both parts.
- **Steps 2 to 7**: we assume again that the $|\alpha|$ differences introduced in F are absorbed.
- **Step 8**: the two α differences cancel out in the middle of the mechanism, but α is also fed back to $k0$.
- **Step 9**: unlike as in step 1, α introduces a difference $L(\alpha) = \beta$ in $k0$, which propagates during steps 11 to 17.
- **Step 10**: to avoid the introduction of new differences, we need the output of F to have differences $L(\beta) = \gamma$, in order for the differences to vanish in the feedback operation. This is only possible if $\alpha \vee \beta \vee \gamma = \alpha \vee \beta$. As we will see later, to avoid impossibilities in the differential characteristic, we also have to add the condition $\gamma \wedge \alpha \wedge \neg\beta = 0$.
- **Steps 16 to 24**: the characteristic is the same as in steps 0 to 8.
- **Steps 25 to 32**: note that differences in the right side do not affect the value returned by the compression function after 32 steps. We thus put no condition on those particular differences.

After finding this generic characteristic, it remains to search for an α that minimizes the cost of the attack. But before that, we present a generic method for finding a message block conforming to the right part of the characteristic.

3.2 Efficient Search for a Conforming Block

Once we have found low-weight α, $\beta = L(\alpha)$ and $\gamma = L(\beta)$ such that

$$\alpha \vee \beta \vee \gamma = \alpha \vee \beta \quad \text{and} \quad \gamma \wedge \alpha \wedge \neg\beta = 0 \ ,$$

the complexity of finding a conforming block by repeated trials is heuristically

$$2^{15|\alpha|+2|\beta|+6|\alpha\vee\beta|} \ .$$

This complexity is well above the birthday bound $2^{n/2}$ for all differences we found, let alone the fact that it underestimates the real complexity. For example, for the difference that we use to attack ESSENCE-256, the above expression yields a complexity 2^{210}, whereas a birthday attack needs only 2^{128} trials.

Strategy. To find a conforming block at a reduced cost, we use an "inside-out" strategy similar in spirit to that of the rebound attack [14], namely, we start by finding conforming values for the low-probability characteristic in the middle, then we check that they follow the simpler characteristic in both directions. What we call the *middle part* corresponds to *steps 8 to 17*, inclusive. More precisely, we

1. Find many values that conform to the middle part (i.e., steps 8 to 17);
2. Search, among those values, one that conforms to the differential characteristic in steps 0 to 8, and 17 to 24 (any such value then follows the characteristic up to step 32).

We need to find approximately $2^{14|\alpha|+|\beta|}$ messages in the first phase, in order to have a conforming one with high probability in the second phase. Below we expose our strategy for efficiently finding many values conforming to the characteristic between steps 8 and 17.

Notations. To describe the state during the middle part: in Table 2 each x_j corresponds to a 32 or 64-bit word, depending on the version used. We write $\mathcal{S}$ the set of all indices where $\alpha \vee \beta$ is nonzero, that is,

$$\mathcal{S} = \{i, 0 \leq i < 32, \alpha_i \vee \beta_i = 1\} \quad \text{for ESSENCE-256,}$$
$$\mathcal{S} = \{i, 0 \leq i < 64, \alpha_i \vee \beta_i = 1\} \quad \text{for ESSENCE-512.}$$

We write $s = |\alpha \vee \beta| = |\mathcal{S}|$ the cardinality of $\mathcal{S}$. For example, if $\alpha =$ `80000000` and $\beta =$ `00000004`, then $\alpha_{31} = \beta_2 = 1$, and so $\mathcal{S} = \{2, 31\}$ and $s = 2$. We also write ℓ for the word bit length (32 or 64, depending on the version of ESSENCE).

Table 2. Message part in steps 8-17

8	$x_0 \oplus \alpha$	x_1	x_2	x_3	x_4	x_5	x_6	x_7
9	x_1	x_2	x_3	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$
10	x_2	x_3	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$
11	x_3	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}
12	x_4	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}
13	x_5	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}
14	x_6	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}
15	x_7	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}
16	$x_8 \oplus \alpha$	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
17	$x_9 \oplus \beta$	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	$x_{16} \oplus \alpha$

Efficient Search. To search for values conforming to the middle part, we first look at an arbitrary slice i, and we count the number of possible tuples $(x_1, \ldots, x_{15})_i$ that fulfill the characteristic between steps 8 and 17. This corresponds to all tuples that satisfy the following equations:

$$
\begin{aligned}
F(x_1, x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7\ )_i &= F(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7\ )_i \\
F(x_2, x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8\ )_i &= F(x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha\ )_i \\
F(x_3, x_4,\ x_5,\ x_6,\ x_7,\ x_8,\ x_9\ )_i &= F(x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta\ )_i \oplus \gamma_i \\
F(x_4, x_5,\ x_6,\ x_7,\ x_8,\ x_9,\ x_{10})_i &= F(x_4,\ x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10})_i \\
F(x_5, x_6,\ x_7,\ x_8,\ x_9,\ x_{10}, x_{11})_i &= F(x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11})_i \\
F(x_6, x_7,\ x_8,\ x_9,\ x_{10}, x_{11}, x_{12})_i &= F(x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12})_i \\
F(x_7, x_8,\ x_9,\ x_{10}, x_{11}, x_{12}, x_{13})_i &= F(x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12}, x_{13})_i \\
F(x_8, x_9,\ x_{10}, x_{11}, x_{12}, x_{13}, x_{14})_i &= F(x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12}, x_{13}, x_{14})_i \\
F(x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15})_i &= F(x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15})_i
\end{aligned}
$$

This property is only interesting for $i \in \mathcal{S}$, since for $i \notin \mathcal{S}$ there are no differences.

For slices such that $\gamma_i = 1$, we need to have a difference in F as well, to erase γ_i. Table 3 reports the number of solutions for the x_i's depending on $(\alpha_i, \beta_i, \gamma_i)$. As we will see later (Tab. 4), in the case $(1, 0, 1)$ there is no tuple satisfying the whole differential characteristic, thus this case will not be used.

Table 3. Number of solutions for the $(x_1, \ldots, x_{15})$ depending on the input differences.

γ_i	(α_i, β_i)		
	$(0,1)$	$(1,0)$	$(1,1)$
0	96	96	96
1	128	120	176

Then, for each slice $i \in \mathcal{S}$ we fix one of these tuples and try to compute the missing bits. The number of possibilities to choose the tuples for $i \in \mathcal{S}$ is

$$N_\alpha = 96^{|\alpha \wedge \neg\beta \wedge \neg\gamma|} \times 96^{|\alpha \wedge \beta \wedge \neg\gamma|} \times 96^{|\neg\alpha \wedge \beta \wedge \neg\gamma|} \times 176^{|\alpha \wedge \beta \wedge \gamma|} \times 128^{|\neg\alpha \wedge \beta \wedge \gamma|}\ .$$

Note that to follow the characteristic, the equations below (directly derived from the ESSENCE mechanism) must hold:

$$L(\overbrace{\quad x_7 \quad}^{s \text{ bits fixed}}) = x_0 \oplus \overbrace{x_8 \oplus F(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7)}^{s \text{ bits fixed}} \tag{1}$$

$$L(\overbrace{\quad x_8 \quad}^{s \text{ bits fixed}}) = \overbrace{x_1 \oplus x_9 \oplus F(x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha)}^{s \text{ bits fixed}} \tag{2}$$

$$L(\overbrace{\quad x_9 \quad}^{s \text{ bits fixed}}) = \overbrace{x_2 \oplus x_{10} \oplus F(x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta) \oplus \gamma}^{s \text{ bits fixed}} \tag{3}$$

$$L(\overbrace{\quad x_{10} \quad}^{s \text{ bits fixed}}) = \overbrace{x_3 \oplus x_{11} \oplus F(x_4,\ x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10})}^{s \text{ bits fixed}} \tag{4}$$

$$L(\overbrace{\quad x_{11} \quad}^{s \text{ bits fixed}}) = \overbrace{x_4 \oplus x_{12} \oplus F(x_5,\ x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11})}^{s \text{ bits fixed}} \tag{5}$$

$$L(\overbrace{\quad x_{12} \quad}^{s \text{ bits fixed}}) = \overbrace{x_5 \oplus x_{13} \oplus F(x_6,\ x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12})}^{s \text{ bits fixed}} \tag{6}$$

$$L(\overbrace{\quad x_{13} \quad}^{s \text{ bits fixed}}) = \overbrace{x_6 \oplus x_{14} \oplus F(x_7,\ x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12}, x_{13})}^{s \text{ bits fixed}} \tag{7}$$

$$L(\overbrace{\quad x_{14} \quad}^{s \text{ bits fixed}}) = \overbrace{x_7 \oplus x_{15} \oplus F(x_8 \oplus \alpha,\ x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12}, x_{13}, x_{14})}^{s \text{ bits fixed}} \tag{8}$$

$$L(\overbrace{\quad x_{15} \quad}^{s \text{ bits fixed}}) = x_{16} \oplus \overbrace{x_8 \oplus F(x_9 \oplus \beta,\ x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15})}^{s \text{ bits fixed}} \tag{9}$$

The bits fixed in $x_1, \ldots, x_{15}$ are those in slices $i \in \mathcal{S}$. Consider new intermediate variables R_8,R_9,...,R_{14} corresponding to the value of the right hand sides of Eq. (2)-(8). Each of these equations corresponds to a *linear* system

$$L(x_j) = R_j \ ,$$

for j in $\{8, \ldots, 14\}$. These are systems of ℓ equations between bits, wherein $2s$ variables are fixed and $2(\ell - s)$ variables are free. Due to the linearity, we can rewrite them as

$$L(x_{j,\overline{\mathcal{S}}}) \oplus R_{j,\overline{\mathcal{S}}} = L(x_{j,\mathcal{S}}) \oplus R_{j,\mathcal{S}} \ , \tag{10}$$

where $\overline{\mathcal{S}}$ is the complement of the set $\mathcal{S}$ and $x_{j,\mathcal{T}}$ is the vector $(x_{j,i})$ with values 0 for i not in $\mathcal{T} \in \{\mathcal{S}, \overline{\mathcal{S}}\}$, thus $x_{j,\mathcal{S}} = x_j \wedge (\alpha \vee \beta)$ and $x_{j,\overline{\mathcal{S}}} = x_j \wedge \neg(\alpha \vee \beta)$. The position of the free variables depends only on $\mathcal{S}$. We can therefore perform a Gaussian elimination once for all on the left hand side of Equation (10).

We have more equations than free variables, so if the system is of maximal rank, we obtain $2s - \ell$ equations which must be satisfied by the fixed variables in order for solutions to exist. For our seven linear systems we have in total $7(2s - \ell)$ equations. Thus for any choice of $(x_1, \ldots, x_{15})$ fixed at $i \in \mathcal{S}$ we have a probability $2^{-7(2s-\ell)}$ of finding a valid solution for all the 7 systems.

Once we know that our choice corresponds to a solution, we can compute efficiently the remaining bits of $x_j, R_j, j \in \{8, \ldots, 14\}$ by the other $7(2\ell - 2s)$ equations of the Gaussian elimination. To find a solution $x_0, \ldots, x_{16}$ which satisfies the middle part, one thus proceeds as follows:

1. Fix the s bits in $x_1, \ldots, x_{15}$ to one of the N_α admissible values;
2. Try to solve the linear systems

$$L(x_j) = R_j \ ,$$

   for j in $\{8, \ldots, 14\}$. If there is no solution, go back to step 1. Once a solution to the seven systems is found, we have all bits of x_j, R_j fixed for j in $\{8, \ldots, 14\}$. In $x_1, \ldots, x_7, x_{15}$ we only have the s bit fixed from the previous step, and in x_0, x_{16} no bits at all are fixed.
3. Freely choose the value of the $(\ell - s)$ remaining bits in x_7 since modifying those bits does not affect the previous steps.
4. We have now to consider the system

$$R_8 = x_1 \oplus x_9 \oplus F(x_2,\ x_3,\ x_4, x_5,\ x_6,\ x_7,\ x_8) \tag{11}$$
$$R_9 = x_2 \oplus x_{10} \oplus F(x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8,\ x_9) \tag{12}$$
$$R_{10} = x_3 \oplus x_{11} \oplus F(x_4,\ x_5,\ x_6,\ x_7,\ x_8,\ x_9,\ x_{10}) \tag{13}$$
$$R_{11} = x_4 \oplus x_{12} \oplus F(x_5,\ x_6,\ x_7,\ x_8,\ x_9,\ x_{10}, x_{11}) \tag{14}$$
$$R_{12} = x_5 \oplus x_{13} \oplus F(x_6,\ x_7,\ x_8,\ x_9,\ x_{10}, x_{11}, x_{12}) \tag{15}$$
$$R_{13} = x_6 \oplus x_{14} \oplus F(x_7,\ x_8,\ x_9,\ x_{10}, x_{11}, x_{12}, x_{13}) \tag{16}$$
$$R_{14} = x_7 \oplus x_{15} \oplus F(x_8,\ x_9,\ x_{10}, x_{11}, x_{12}, x_{13}, x_{14}) \tag{17}$$

   where the R_j's are fixed. In these equations, we can skip α, β and γ since we chose admissible values for $(x_1, \ldots, x_{15})$. This system is almost in triangular form : Eq. (17) fixes x_{15}; Eq. (16) fixes x_6; Eq. (15) fixes x_5; Eq. (14) fixes x_4; Eq. (13) fixes x_3; Eq. (12) fixes x_2; Eq. (11) fixes x_1. Finally, Eq. (1) fixes x_0 and Eq. (9) fixes x_{16}.

Each valid solution in step 2 gives us $2^{\ell - s}$ results, by exploiting the extra degrees of freedom in step 3. We obtain in total about $N_\alpha \cdot 2^{7(\ell - 2s)} \cdot 2^{\ell - s} \cdot 2^{-1}$ possible pairs that satisfy the characteristic from step 8 to 17. The factor 2^{-1} comes from the fact that we counted each possible pair twice.

We can improve this general method in two ways.

- First, we can do better than trying $2^{7(2s-\ell)}$ tuples to find a solution in step 2. This is based on a Gaussian elimination on the $2s - \ell$ equations allowing us to explore the set of all candidate tuples by a depth-first search procedure. For the sake of simplicity, we will explain the details later on the example of ESSENCE-256 in §3.4. Without this method we would need $2^{7(2s-\ell)}$ trials to find one solution, which would increase the complexity a lot.

- Secondly, we can improve the choice of $(x_1, \ldots, x_{15})_i, i \in \mathcal{S}$. This time we only consider a tuple $(x_1, \ldots, x_{15})_i$ admissible if it can be extended to a whole characteristic from 0 to 24. This reduces the values in Table 3 to the following ones. We can see that the case $(\alpha_i, \beta_i, \gamma_i) = (1, 0, 1)$ leads to an impossibility.

Table 4. Number of solutions for the $(x_1, \ldots, x_{15})$ which can be extended to satisfy the whole characteristic.

γ_i	(α_i, β_i)		
	$(0,1)$	$(1,0)$	$(1,1)$
0	96	2	4
1	128	0	2

Finally, for each slice $i \in \mathcal{S}$ we fix one of these tuples and try to compute the missing bits. The number of possibilities to choose the tuples for $i \in \mathcal{S}$ is

$$\tilde{N}_\alpha = 2^{|\alpha \wedge \neg\beta \wedge \neg\gamma|} \times 4^{|\alpha \wedge \beta \wedge \neg\gamma|} \times 96^{|\neg\alpha \wedge \beta \wedge \neg\gamma|} \times 2^{|\alpha \wedge \beta \wedge \gamma|} \times 128^{|\neg\alpha \wedge \beta \wedge \gamma|} \ .$$

This method increases the probability of passing the rest of the characteristic as we will see in §3.3. The choice of rounds $8 - 17$ for the middle part was done to get the lowest possible value for $\tilde{N}_\alpha$, see Appendix A.

The subsequent sections discuss the complexity of performing the search of the rest of the characteristic, and give concrete complexity estimates for each instance of ESSENCE.

3.3 Finding Accurate Probabilities

Relying only on the Hamming weight to approximate the probability of the differential characteristic gives unacceptably inaccurate approximations. Indeed, for a given word slice, probabilities of differences to be absorbed at each step are not independent, and neglecting this leads to estimates far from actual values. For example, a single bit difference is absorbed during seven steps with probability $2^{-8.4}$, which is significantly lower than the heuristic estimate 2^{-7} based on the one bit difference. However, for the characteristic considered, the dependency between word slices seems negligible. We thus give complexities with respect to empirical estimates, computed independently for each word slice. That is, we compute the probability of the differential characteristic as 32 (or 64) independent differential characteristics, i.e., one for each slice.

We could estimate the real probability of our characteristic for any given difference α. We found that having $\alpha_i = 1$, $\beta_i = 0$ and $\gamma_i = 1$ leads to an impossibility (the differential cannot be satisfied for that α). This is why we need the condition

$$\gamma \wedge \alpha \wedge \neg\beta = 0 \ .$$

When considering the middle part, we also computed the real probability of verifying the sliced characteristic once this part of the characteristic is satisfied. The complexities given in the next section were computed with respect to those empirical estimates, not with the heuristic values based only on the Hamming weight.

Reusing our notation α, β, γ, we give in Tab. 5(a) the probabilities for *a given slice* i to follow the complete characteristic on 32 steps (the impossible cases—of probability zero—are not included), depending on $(\alpha_i, \beta_i, \gamma_i) \in \{0,1\}^3$. To compute the probability for a given difference $(\alpha_i, \beta_i, \gamma_i)$ we count the number of possible bit sequences following the whole differential characteristic. The probability that a random input follows the characteristic is the product of those probabilities, with each raised to a power that equals the number of slices corresponding to this case. For the α's used in our attacks and the exact values of the probabilities, we obtain probabilities $2^{-240.6}$ and $2^{-478.9}$, respectively for ESSENCE-256 and ESSENCE-512.

Taking into account our basic technique in §3.2 for solving the middle part at a reduced cost, we obtain the probabilities in Tab. 5(b). If we consider only those tuples $(x_1, \ldots, x_{15})$ where there is at least one possibility of verifying the whole characteristic we get the values in Tab. 5(c). In both cases, we count for every difference $(\alpha_i, \beta_i, \gamma_i)$ the number of extensions of the valid tuples $(x_1, \ldots, x_{15})$ satisfying the whole characteristic and compare it to the number of arbitrary extensions.

Given those numbers, we find that the probability that a value conforming to the middle part follows the rest of the characteristic is $2^{-87.1}$ for ESSENCE-256 and $2^{-158.7}$ for ESSENCE-512 with the basic method and respectively $2^{-62.2}$ and $2^{-116.1}$ for the improved one.

Table 5. Probability of passing the differential characteristic depending on the input differences.

	$(\alpha_i, \beta_i, \gamma_i)$	$(0,0,0)$	$(0,1,0)$	$(0,1,1)$	$(1,0,0)$	$(1,1,0)$	$(1,1,1)$
(a)	complete characteristic	1	$2^{-9.5}$	$2^{-9.1}$	$2^{-24.4}$	2^{-23}	2^{-26}
(b)	basic method	1	$2^{-1.1}$	$2^{-1.1}$	2^{-16}	$2^{-14.6}$	$2^{-18.5}$
(c)	improved method	1	$2^{-1.1}$	$2^{-1.1}$	$2^{-10.4}$	2^{-10}	2^{-12}

There are at least two ways to compute the total number of message pairs that is going to satisfy the whole characteristic. As we will obtain nearly the same result with both of them, we can verify the soundness of the probabilities after solving the middle characteristic. First, some additional notations are required: we let $\rho_0, \ldots, \rho_{\ell-1}$ denote the probabilities for each slice in $0, \ldots, \ell-1$ of conforming to the differential, i.e., each ρ_i lies in $\{1, 2^{-9.5}, 2^{-9.1}, 2^{-24.4}, 2^{-23}, 2^{-26}\}$; and we let $\tau_0, \ldots, \tau_{\ell-1}$ be the conditional probabilities for each slice to follow the differential characteristic, assuming that the middle part is satisfied. Now, the two equivalent ways to express the number of conforming messages are:

1. The probability of the whole characteristic is $\prod_{i=0}^{\ell-1} \rho_i$, hence the number of pairs of conforming messages is

$$2^{8\ell} \cdot \prod_{i=0}^{\ell-1} \rho_i \ ,$$

   where 8ℓ is the digest bit length.
2. The probability of the characteristic once the middle part is satisfied is $\prod_{i=0}^{\ell-1} \tau_i$; calling N the number of pairs conforming to the middle part, the number of conforming message pairs is then

$$N \cdot \prod_{i=0}^{\ell-1} \tau_i \ .$$

In both cases we find each possible message pair twice. We verified that these two ways of computing the total number yield similar values (up to rounding approximations), which shows that the probabilities of verifying the whole characteristic and the characteristic after solving the middle part correspond to each other.

For example for our α: for ESSENCE-256 we find $2^{256} \cdot 2^{-240.6} = 2^{15.4}$ message pairs considering the whole characteristic. With our improved version we have $\tilde{N_\alpha} = 2^{106.5}$; a probability of 2^{-42} of solving the seven linear systems; 2^{13} times more solutions for free and thus $N = 2^{77.5}$. Using the probability $2^{-62.1}$ of passing the rest of the characteristic we again get $2^{15.4}$ message pairs.

3.4 Collisions for ESSENCE-256

For ESSENCE-256, we could perform an exhaustive search over all 2^{32} possible differences α and found as optimal value $\alpha =$ `80102040`, for which $|\alpha| = 4$, $|\beta| = 18$, and $|\alpha \vee \beta| = s = 19$. Heuristic estimates based on Hamming weights suggest that we need about $2^{14\times4+18} = 2^{74}$ messages that conform to the middle part to find at least one conforming to the differential characteristic on the right side. However, the empirical complexity is (cf. §3.3) approximately $2^{87.1}$ for the basic method and $2^{62.2}$ for the improved one.

In the following, we focus on the improved version.

Solving the Right Side. For that α, we have in total

$$\tilde{N_\alpha} = 96^6 \times 2^1 \times 128^9 \times 2^3 \approx 2^{106.5}$$

possibilities to set the bits in $\mathcal{S}$. We have a probability $2^{7(32-2\times19)} = 2^{-42}$ of finding a solution to the seven systems defined by Eq. (3) to (9). Following our assumption in §3.2, we get about $2^{64.5}$ solutions. For each solution, we obtain 2^{13} additional solutions by varying the bits $(x_7)_j$ for j not in $\mathcal{S}$, yielding in total up to $2^{77.5}$ solutions.

For each message pair found, we must check that it satisfies the rest of the characteristic. As found in §3.3, we need about $2^{62.2}$ values conforming to the middle part to find one value following the rest of the characteristic. Below we detail the cost of finding those messages.

We look for solutions of systems (2) to (8). The linear systems $L(x_j) = R_j$ consist each of 32 equations and 26 free variables and has full rank. We have thus 6 linear equations which the fixed bits must fulfill to guarantee that there exists a solution of the linear system.

We can choose those equations such that:

- choosing the values of the bit slices $0, 1, 2, 3, 5, 6, 9, 10, 12, 16, 18$ fixes the parity of the first equation. This does not change with the choice of the remaining slices;
- if we choose in addition the values of the bit slices $7, 11, 13$, we fix the second equation;
- if we choose in addition the values of the bit slices $4, 17$, we fix the third equation;
- if we choose in addition the values of the bit slices $14, 15$, we fix the fourth equation;
- finally, choosing the value for the last slice, the eighth one, fixes the parity of the remaining two equations.

This allows us to explore the set of candidate tuples efficiently by a depth-first search. Moreover, we can precompute the parity corresponding to the last three equations for any 3-tuple of choices for slices $8, 14$, and 15. That way, we do not even need to test the different tuples, but only to enumerate the ones giving us a valid solution. The cost to find a solution is therefore very low.

Using the degree of freedom coming from step 3 of the solving procedure, our implementation is able to generate solutions for the middle part systems and to test the rest of the characteristic at a rate of approximately 650 cycles per candidate on an Intel Core 2 processor, against about 1600 cycles for hashing 256 bits[7].

Solving the Left Side. Once a conforming pair of message blocks is found, we just need to try approximately $2^{67.4}$ distinct random chaining values to find a collision (for comparison, the heuristic estimate based on Hamming weights is $2^{14\times 4} = 2^{56}$). This value limits our attack. Since there is no α with a hamming weight of 3, which verifies the characteristic on the right side, we cannot improve this value. Note that our attack can be carried out with negligible memory (the $2^{62.2}$ messages that satisfy the middle part don't have to be stored: we test repeatedly each candidate message, and discard it if it does not conform to the full characteristic).

If we only search for a semi-free-start collision we can reduce the complexity of the left side to $2^{33.7}$, which makes again the right side the limiting part. We apply the same techniques as for the right side to compute IV pairs that passes

[7] See eBASH: `http://bench.cr.yp.to/ebash.html`.

from step 1 to step 8. We have to compute about $2^{33.7}$ IV pairs to find one satisfying the whole characteristic. This method was applied in Appendix C.

3.5 Collisions for ESSENCE-512

For ESSENCE-512, the best difference is $\alpha = \texttt{8408400000480082}$, giving $|\alpha| = 8$, $|\beta| = 35$, and $|\alpha \vee \beta| = s = 39$. We tested all 64-bit differences with a Hamming weight of up to 10. Since the weight of α is the limiting property on the left side and thus for the whole attack, we are sure to have found the best value for the whole attack. For our α, the matrix of the linear system has again full rank, thus we can directly apply the same techniques as for ESSENCE-256. As discussed in §3.3, with the basic method we need about $2^{158.7}$ solutions of the middle part to find one solution for the right side of the characteristic (against 2^{147} with heuristic estimates based on Hamming weights). With the improved method we only need $2^{116.1}$ solutions. In the following, we consider only the improved method.

Solving the Right Side. For our α we have

$$\tilde{N_\alpha} = 96^{14} \times 2^4 \times 4^3 \times 128^{17} \times 2^1 \times \approx 2^{222.2}$$

possibilities for the tuples at the indices $i \in \mathcal{S}$ and a probability of about 2^{-98} to find a solution for all the systems of Eq. (2)-(8). Thus, we expect about $2^{124.2}$ solutions. Using the free bits, we get for each solution $2^{64-39} = 2^{25}$ additional solutions. In total, there are thus about $2^{149.2}$ solutions, which is high enough for finding one conforming to the full characteristic (trying $2^{116.1}$ is sufficient).

Solving the Left Side. Now, we have a pair of messages that verify the differential characteristic. The probability for a random chaining value of verifying the differential characteristic is approximately $2^{-134.7}$. Again, this value is the limiting part of our attack.

4 Attacking HMAC-ESSENCE

HMAC [15] is a widely used construction for building message authentication codes out of hash functions. Proposed in 1996 by Bellare, Canetti, and Krawczyk, HMAC has been standardized by NIST in 2002 [16] and requirements for SHA-3 include compatibility with HMAC.

The results in §3, can directly be turned into a distinguisher for ESSENCE-256 and ESSENCE-512 when used in keyed mode, be it with an unknown prefix message, or within HMAC. More precisely, we use the property that we can precompute a conforming message block once, and then *separately* seek a conforming chaining value. We just make the standard assumption that we can query an oracle (non-adaptively) with messages, and that this returns the digests produced by the keyed ESSENCE with this message as input, for a randomly preselected key.

A distinguisher then works as follows:

1. Find a pair of blocks (x, y) that conforms to the message part differential.
2. Repeat until a collision is found:
3. Pick a unique prefix m.
4. Query for oracle with $m\|x$ and $m\|y$.

Ideally 2^{128} trials are expected before a collision for ESSENCE-256, but here we'll make only $2^{67.4}$ trials in average, after a precomputation of complexity $2^{62.2}$. For ESSENCE-512, we have a complexity $2^{134.7}$ instead of 2^{256} ideally.

We can also mount an *existential forgery* attack by making one additional adaptive query:

1. Run the distinguisher above to obtain blocks m, x, y such that $m\|x$ and $m\|y$ collide by HMAC-ESSENCE.
2. Pick an arbitrary block m'.
3. Query the oracle for the MAC of $m\|x\|m'$, obtain a value z.
4. Return z as forgery of $m\|y\|m'$.

The complexity of this attack is essentially the same as that of the simple distinguisher.

5 Conclusion

We presented collision attacks on ESSENCE-256 and ESSENCE-512 of respective complexities $2^{67.4}$ and $2^{134.7}$. More precisely, these values are upper bounds on the cost of running our attacks, in terms of compression-equivalent units. Implementations of our attacks need only negligible memory, and in particular avoid expensive memory accesses. We combine several methods to achieve our goal: separate treatment of message and chaining value, exact estimation of the probabilities, computation of the low probability part, efficient solution finding for linear systems and reduction of the search space by considering the whole characteristic. The attacks were experimentally verified on reduced versions of ESSENCE, and also apply to the versions of ESSENCE with 224- and 384-bit digests. An example of a practical free-start-collision on 29 out of 32 rounds can be found in Appendix C.

Attacks to the HMAC are usually much harder than collision attacks, as we can see at the examples of MD4 [17], MD5 [17, 18] or SHA-1 [19]. However, we could show direct applications of our collision attack to the HMAC construction instantiated with ESSENCE, giving a distinguisher and an existential forgery attack with same complexity as the collision attacks.

Our results reveal significant weaknesses in the version of ESSENCE submitted to NIST.

Acknowledgments We would like to thank for their help: Anne Canteaut, Stéphane Jacob, Nicky Mouha, Gautham Sekar, and Fabien Viger.

References

1. Wang, X., Yu, H.: How to break MD5 and other hash functions. In Cramer, R., ed.: EUROCRYPT. Volume 3494 of Lecture Notes in Computer Science., Springer (2005) 19–35
2. Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In Shoup, V., ed.: CRYPTO. Volume 3621 of Lecture Notes in Computer Science., Springer (2005) 17–36
3. Cannière, C.D., Rechberger, C.: Finding SHA-1 characteristics: General results and applications. In Lai, X., Chen, K., eds.: ASIACRYPT. Volume 4284 of Lecture Notes in Computer Science., Springer (2006) 1–20
4. Stevens, M., Lenstra, A.K., de Weger, B.: Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities. In Naor, M., ed.: EUROCRYPT. Volume 4515 of Lecture Notes in Computer Science., Springer (2007) 1–22
5. NIST: FIPS 180-2 – secure hash standard (2002)
6. NIST: Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (sha-3) family. In: Federal Register. (Nov. 2007) Vol. 72, No. 212.
7. NIST: Cryptographic hash algorithm competition. http://csrc.nist.gov/groups/ST/hash/sha-3/index.html
8. ECRYPT II: The sha-3 zoo. http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo
9. Martin, J.W.: ESSENCE: A candidate hashing algorithm for the NIST competition. Submission to NIST (2008)
10. Martin, J.W.: ESSENCE: A family of cryptographic hashing algorithms. Submission to NIST (2008)
11. Mouha, N., Sekar, G., Aumasson, J.P., Peyrin, T., Thomsen, S.S., Turan, M.S., Preneel, B.: Cryptanalysis of the ESSENCE family of hash functions. In: Information Security and Cryptology - Inscrypt 2009. LNCS, Springer (2009) to appear.
12. Merkle, R.C.: One way hash functions and DES. In: Advances in Cryptology - CRYPTO'89. Volume 435 of Lecture Notes in Computer Science., Springer (1989) 428–446
13. Damgård, I.: A Design Principle for Hash Functions. In Brassard, G., ed.: Advances in Cryptology - CRYPTO'89. Volume 435 of Lecture Notes in Computer Science., Springer (1989) 416–427
14. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl. In Dunkelman, O., ed.: FSE. Volume 5665 of Lecture Notes in Computer Science., Springer (2009) 260–276
15. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In Koblitz, N., ed.: CRYPTO. Volume 1109 of Lecture Notes in Computer Science., Springer (1996) 1–15
16. NIST: FIPS 198 – the keyed-hash message authentication code (HMAC) (2002)
17. Wang, L., Ohta, K., Kunihiro, N.: New key-recovery attacks on HMAC/NMAC-MD4 and NMAC-MD5. In Smart, N.P., ed.: EUROCRYPT. Volume 4965 of Lecture Notes in Computer Science., Springer (2008) 237–253
18. Wang, X., Yu, H., Wang, W., Zhang, H., Zhan, T.: Cryptanalysis on HMAC/NMAC-MD5 and MD5-MAC. In Joux, A., ed.: EUROCRYPT. Volume 5479 of Lecture Notes in Computer Science., Springer (2009) 121–133
19. Rechberger, C., Rijmen, V.: New results on NMAC/HMAC when instantiated with popular hash functions. Journal of Universal Computer Science **14**(3) (2008) 347–376

A Choice of the Position of the Middle Part

We can see in Table 6 that the choice of rounds 8 to 17 minimizes the number of solutions for the middle part $(x_1, \ldots, x_{15})$. Since the total number of solutions is always the same, a smaller number of solutions in the middle part means that we have a higher probability of passing the rest of the characteristic.

Table 6. Number of solutions for the $(x_1, \ldots, x_{15})$ which can be extended to satisfy the whole characteristic, depending on the input differences and on the rounds.

Rounds	$(\alpha_i, \beta_i, \gamma_i)$				
	$(1,0,0)$	$(0,1,0)$	$(1,1,0)$	$(0,1,1)$	$(1,1,1)$
0-9	12	3968	8	4960	4
1-10	12	1984	8	2480	4
2-11	8	3072	8	3840	4
3-12	8	2160	8	2640	4
4-13	4	1152	4	1408	4
5-14	4	576	4	704	4
6-15	4	288	8	352	4
7-16	4	192	8	224	4
8-17	2	96	4	128	2
9-18	4	96	8	128	4
10-19	4	96	12	128	4
11-20	4	176	12	208	4
12-21	4	352	12	384	4
13-22	4	512	12	640	4
14-23	4	1024	16	1280	4

B Probabilities of the Right Side of the Characteristic

Table 7 compares heuristic probability estimates based on Hamming weights with actual, empirically verified, probabilities. The empirically verified values are taken from the improved method, considering only values $(x_1, \ldots, x_{15})_i$, $i \in \mathcal{S}$, which are able to satisfy the rest of the characteristic. Therefore, the overall probability is higher than the heuristic approximation.

C Practical Semi-Free-Start Collision on 29 out of the 32 rounds

Once we have a message passing the right side, we can apply the same techniques that we used to compute from step 8 to step 17, to get a semi-free start collision. We compute IV pairs that passes from step 1 to 8 on the left side and test if they pass the rest of the characteristic. With our α for ESSENCE-256 we have to test about $2^{33.7}$ IV pairs. Together with a message pair we found passing 29 out of the 32 rounds we got the semi-free-start collision presented in Table 8.

Table 7. Comparison between the *heuristic approximations* of the probability to reach the *next* difference and the real probabilities, empirically estimated.

	Message part								Heuristic approximation	Empirical estimate of improved method
	$k7$	$k6$	$k5$	$k4$	$k3$	$k2$	$k1$	$k0$		
0	α	β	·	·	·	·	·	·	2^{-18}	$2^{-21.3}$
1	β	·	·	·	·	·	·	α	2^{-4}	2^{-4}
2	·	·	·	·	·	·	α	·	2^{-4}	$2^{-1.7}$
3	·	·	·	·	·	α	·	·	2^{-4}	2^{-4}
4	·	·	·	·	α	·	·	·	2^{-4}	2^{-4}
5	·	·	·	α	·	·	·	·	2^{-4}	1
6	·	·	α	·	·	·	·	·	2^{-4}	2^{-4}
7	·	α	·	·	·	·	·	·	2^{-4}	1
8	α	·	·	·	·	·	·	·	—	—
9	·	·	·	·	·	·	·	α	—	—
10	·	·	·	·	·	·	α	β	—	—
11	·	·	·	·	·	α	β	·	—	—
12	·	·	·	·	α	β	·	·	—	—
13	·	·	·	α	β	·	·	·	—	—
14	·	·	α	β	·	·	·	·	—	—
15	·	α	β	·	·	·	·	·	—	—
16	α	β	·	·	·	·	·	·	—	—
17	β	·	·	·	·	·	·	α	2^{-4}	1
18	·	·	·	·	·	·	α	·	2^{-4}	2^{-3}
19	·	·	·	·	·	α	·	·	2^{-4}	2^{-1}
20	·	·	·	·	α	·	·	·	2^{-4}	2^{-4}
21	·	·	·	α	·	·	·	·	2^{-4}	2^{-3}
22	·	·	α	·	·	·	·	·	2^{-4}	2^{-8}
23	·	α	·	·	·	·	·	·	2^{-4}	2^{-4}
24	α	·	·	·	·	·	·	·	1	1
25	·	·	·	·	·	·	·	α	1	1
26	·	·	·	·	·	·	α	?	1	1
27	·	·	·	·	·	α	?	?	1	1
28	·	·	·	·	α	?	?	?	1	1
29	·	·	·	α	?	?	?	?	1	1
30	·	·	α	?	?	?	?	?	1	1
31	·	α	?	?	?	?	?	?	1	1
32	α	?	?	?	?	?	?	?	1	1
								Total	2^{-74}	2^{-62}

Table 8. Example of semi-free-start collision on 29 of the 32 rounds of the differential characteristic, for $\alpha =$ 80102040 and $\beta =$ 537874EB.

	Initial values for r									Initial values for k								
	$r7$	$r6$	$r5$	$r4$	$r3$	$r2$	$r1$	$r0$		$k7$	$k6$	$k5$	$k4$	$k3$	$k2$	$k1$	$k0$	
	B0741769	BA2BA1A1	349A4DC8	54204D82	292006B1	80096194	D23020E1	9098A7EA		4CD35806	4759FB6D	3ED267E5	17641536	BE1F35ED	688B0C3C	DF126549	5FAE0827	
round	differences								round	differences								round
0	0	0	0	0	0	0	0	0	0	80102040	537874EB	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	80102040	1	537874EB	0	0	0	0	0	0	80102040	1
2	0	0	0	0	0	0	80102040	0	2	0	0	0	0	0	0	80102040	0	2
3	0	0	0	0	0	80102040	0	0	3	0	0	0	0	0	80102040	0	0	3
4	0	0	0	0	80102040	0	0	0	4	0	0	0	0	80102040	0	0	0	4
5	0	0	0	80102040	0	0	0	0	5	0	0	0	80102040	0	0	0	0	5
6	0	0	80102040	0	0	0	0	0	6	0	0	80102040	0	0	0	0	0	6
7	0	80102040	0	0	0	0	0	0	7	0	80102040	0	0	0	0	0	0	7
8	80102040	0	0	0	0	0	0	0	8	80102040	0	0	0	0	0	0	0	8
9	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	80102040	9
10	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	80102040	537874EB	10
11	0	0	0	0	0	0	0	0	11	0	0	0	0	0	80102040	537874EB	0	11
12	0	0	0	0	0	0	0	0	12	0	0	0	0	80102040	537874EB	0	0	12
13	0	0	0	0	0	0	0	0	13	0	0	0	80102040	537874EB	0	0	0	13
14	0	0	0	0	0	0	0	0	14	0	0	80102040	537874EB	0	0	0	0	14
15	0	0	0	0	0	0	0	0	15	0	80102040	537874EB	0	0	0	0	0	15
16	0	0	0	0	0	0	0	0	16	80102040	537874EB	0	0	0	0	0	0	16
17	0	0	0	0	0	0	0	80102040	17	537874EB	0	0	0	0	0	0	80102040	17
18	0	0	0	0	0	0	80102040	0	18	0	0	0	0	0	0	80102040	0	18
19	0	0	0	0	0	80102040	0	0	19	0	0	0	0	0	80102040	0	0	19
20	0	0	0	0	80102040	0	0	0	20	0	0	0	0	80102040	0	0	0	20
21	0	0	0	80102040	0	0	0	0	21	0	0	0	80102040	0	0	0	0	21
22	0	0	80102040	0	0	0	0	0	22	0	0	80102040	0	0	0	0	80000040	22
23	0	80102040	0	0	0	0	0	0	23	0	80102040	0	0	0	0	80000040	38C32419	23
24	80102040	0	0	0	0	0	0	0	24	80102040	0	0	0	0	80000040	38C32419	3B50EAEF	24
25	0	0	0	0	0	0	0	0	25	0	0	0	0	80000040	38C32419	3B50EAEF	E9F738F8	25
26	0	0	0	0	0	0	0	0	26	0	0	0	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	26
27	0	0	0	0	0	0	0	0	27	0	0	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	27
28	0	0	0	0	0	0	0	0	28	0	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	28
29	0	0	0	0	0	0	0	0	29	80000040	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	29
30	0	0	0	0	0	0	0	80000040	30	38C32419	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	A7EF91F9	30
31	0	0	0	0	0	0	80000040	102040	31	3B50EAEF	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	A7EF91F9	21E1C70	31
32	0	0	0	0	0	80000040	102040	3336DACE	32	E9F738F8	D59E6BC4	519ECD90	8199374F	1B9B997C	A7EF91F9	21E1C70	1B715D5F	32