Algebraic Attacks on Rasta and Dasta Using Low-Degree Equations

No Author Given

No Institute Given

Abstract. Rasta and Dasta are two fully homomorphic encryption friendly symmetric-key primitives proposed at CRYPTO 2018 and ToSC 2020, respectively. We point out that the designers of Rasta and Dasta neglected an important property of the χ operation. Combined with the special structure of Rasta and Dasta, this property directly leads to significantly improved algebraic cryptanalysis. Especially, it enables us to theoretically break 2 out of 3 instances of full Agrasta, which is the aggressive version of Rasta with the block size only slightly larger than the security level in bits. We further reveal that Dasta is more vulnerable against our attacks than Rasta for its usage of a linear layer composed of an ever-changing bit permutation and a deterministic linear transform. Based on our cryptanalysis, the security margins of Dasta and Rasta parameterized with $(n, \kappa, r) \in \{(327, 80, 4), (1877, 128, 4), (3545, 256, 5)\}$ are reduced to only 1 round, where n, κ and r denote the block size, the claimed security level and the number of rounds, respectively. These parameters are of particular interest as the corresponding ANDdepth is the lowest among those that can be implemented in reasonable time and target the same claimed security level.

Keywords: Rasta, Dasta, Agrasta, χ operation, linearization, algebraic attack

1 Introduction

Since the pioneering work [5] of Albrecht et al. on designs of ciphers friendly to secure multi-party computation (MPC), fully homomorphic encryption (FHE) and zero-knowledge proofs (ZK), an increasing number of MPC-, FHE- and ZK-friendly symmetric-key primitives have been proposed, including LowMC [5], Kreyvrium [15], FLIP [39], Rasta [23], MiMC [4], GMiMC [3], Jarvis [9], Hades [33], Poseidon [32], Vision [7], Rescue [7] and Ciminion [25]. As designing symmetric-key primitives in this domain is relatively new and not well-understood, the designers may be prone to make mistakes in their innovative proposals. Four concrete examples come from the cryptanalysis of LowMC [5], the preliminary version of FLIP [39], the initial version of MARVELlous [9] and MiMC [28].

In the case of LowMC, new higher-order differential cryptanalysis [24] and the optimized interpolation attack [22] revealed that the original parameters of LowMC were too optimistic, which directly pushed LowMC move to LowMC

v2. However, the so-called difference enumeration attack [40] in the low-data setting could still violate the security of some parameters in LowMC v2. As a countermeasure, the formula to calculate the secure number of rounds is updated and this version is called LowMC v3. However, it has been recently demonstrated in [37] that some parameters in LowMC v3 are still insecure when new algebraic techniques and the difference enumeration attack are combined. In addition, a very recent generic method [20] to solve multivariate equation systems over $GF(2)$ also shows that some parameters of LowMC v3 in the Picnic3 [36] setting are insecure.

In the case of the preliminary version of FLIP, Duval, Lallemand and Rotella revealed some weaknesses in its filter function and exploited them to devise an efficient full key recovery attack based on guess-and-determine techniques [27]. This result directly leads to a more conservative design of FLIP.

In the case of MARVELlous [9], Albrecht et al. described a clever way [2] to express the primitive as a set of low-degree equations with the introduction of intermediate variables. On the other hand, as MARVELlous works on a large field, the total number of variables in the equation system is still small even though there are intermediate variables. These directly lead to powerful Gröebner basis attacks as the Gröebner basis of such a set of polynomials can be efficiently computed in time less than that of the brute-force attack.

In the case of MiMC [4] proposed at ASIACRYPT 2016, the key-recovery attack on the full-round versions over $\mathbb{F}_{2^n}$ was presented until ASIACRYPT 2020 [28], mainly owing to a careful study of the evolution of the algebraic degree, though it is only slightly faster than the brute-force attack.

Such a trend in designing symmetric-key primitives for advanced protocols also motivates the cryptographers to generalize several cryptanalytic techniques to fields of odd characteristic [11]. As a consequence, some undesirable properties have been reported for GMiMC and Poseidon.

From the perspective of design, there are two common metrics for these primitives, i.e. the multiplicative complexity (MC) and the multiplicative depth of the circuit. In the context of Rasta [23], MC refers to the total number of AND gates and the multiplicative depth of the circuit refers to the number of rounds (called ANDdepth in Rasta [23]). The aim of Rasta is to provide a design strategy achieving d ANDdepth and d ANDs per bit at the same time. The designers proposed several parameters for the block/key size n, the ANDdepth d and the targeted security level κ. To make d as small as possible and keep its practical usage, $d \in \{4, 5, 6\}$ is recommended. Since generating the affine layers in each encryption is quite time-consuming in Rasta, Hebborn and Leander proposed Dasta [35] where the linear layer is replaced with an ever-changing bit permutation and a deterministic linear transform. Such a construction has made Dasta hundreds times faster than Rasta in the offline settings.

A feature in Rasta and Dasta is that n is much larger than κ and there is indeed no generic attack matching the claimed security level κ. To encourage more cryptanalysis, the designers of Rasta also proposed an aggressive version called Agrasta with $n = \kappa + 1$. The currently best key-recovery attack [26] on

Agrasta in the single-plaintext setting is based on a brute-force approach and only 3 rounds can be covered. Moreover, no nontrivial third-party attacks have been published for Rasta or Dasta. It should be emphasized the same key can be used to encrypt many different plaintext blocks for Rasta, Dasta and Agrasta and hence the attacks should not be limited to the single-plaintext setting. Indeed, it has been shown in [23,35] that given the capability to collect many plaintext-ciphertext pairs under the same key, the attackers still cannot break any of the three proposals.

Algebraic attacks. Algebraic attacks are potential threats to aforementioned primitives, as can be observed from the analysis of LowMC, FLIP, MARVELlous, MiMC, GMiMC and Poseidon. A crucial step to improve the efficiency of an algebraic attack is to construct a suitable equation system that can be efficiently solved with techniques like linearization, guess-and-determine, F4/F5 algorithms [29,30] (computing Gröebner basis) or XL algorithm [17]. How to construct useful equations is nontrivial and dominates the effectiveness of algebraic attacks. For methods to solve equations, the linearization technique is the simplest one, which is to treat each different monomial in the equations as an independent new variable. The drawback is hence obvious as the attacker needs to collect sufficiently many equations in order to solve it with Gaussian elimination. In addition, as the degree of the equations increases, the number of monomials will become very large and the cost of Gaussian elimination may even exceed the generic attack. For the guess-and-determine technique, its performance fully depends on the structure of the original equation system. Finding a clever guess-and-determine strategy is nontrivial. Especially, when the equation system tends to be random, the effect of such a strategy seems to be limited. For advanced algorithms like F4/F5 algorithms and the XL algorithm to solve multivariate polynomial equations, their complexity is hard to bound when the system is much over-defined. If only a portion of equations are taken into account, though the time complexity can be bounded, the resulting complexity may turn to be very high and exceeds the generic attack.

Our Contributions. We observed the feasibility to derive exploitable low-degree equations from the raw definition of the χ operation, which seems to be neglected by the designers for the high degree of the inverse of the large-scale χ operation. As a result, we could construct a system of equations of much lower degree than expected by the designers to describe the primitives equivalently. Specifically, r_0 rounds of Rasta can be represented as a system of equations of degree upper bounded by $2^{r_0-1}+1$ rather than 2^{r_0}. For Dasta, by guessing only 1-bit secret information, we even could extract a system of equations of degree upper bounded by 2^{r_0-1} from many different plaintext-ciphertext pairs for r_0 rounds, which is mainly due to the usage of a deterministic linear transform following a bit permutation in the last linear layer.

It should be emphasized that constructing low-degree equations based on high-degree equations is not new in symmetric-key cryptanalysis. The underlying idea was first utilized in the algebraic attack [18] and fast algebraic attack [16]

on several LFSR-based stream ciphers. A common notion in these attacks is the algebraic immunity of the filter function or the augmented function, which has been studied in several papers [8,31]. It should be mentioned that the resistance against these attack vectors has been taken into account in the design of FLIP [39] as it is very similar to an LFSR-based design, though the register is no longer updated by means of the LFSR, but with pseudorandom bit permutations.

However, Rasta is completely different from the LFSR-based stream cipher and it is more like a block cipher, which can explain why the designers ruled out the above attack vectors as they have not been successfully applied to block ciphers. We emphasize that this is mainly because common block ciphers always have a large number of rounds and hence the degree after a ceratin number of rounds is very high. However, this is not the case of Rasta, which has only a small number of rounds. Although our attack is based on low-degree equations, its feasibility indeed also much relies on our observation on the key feed-forward operation in Dasta and Rasta, i.e. the feature of the construction.

In a sense, our basic idea can be viewed as exploiting the algebraic immunity of the augmented function, which is the large-scale χ operation in Rasta and Dasta. As far as we know, there is no efficient method to compute the algebraic immunity of a huge S-box, which may be another reason why the designers did not take it into account. Understanding our attacks requires no knowledge of the algebraic immunity of the augmented function, though. In a nutshell, we reveal that the last nonlinear layer is ineffective to significantly increase the degree for the usage of a simple key feed-forward operation, whatever the last linear layer is.

On the complexity of Gaussian elimination. Denote the exponent of Gaussian elimination by ω. A naive implementation of Gaussian elimination leads to $\omega = 3$. Due to Strassen's divide-and-conquer algorithm [41], the upper bound of ω is updated as $\log_2 7$ and the algorithm has been practically implemented in [1]. Although there exists a more efficient algorithm [6] to perform the matrix multiplication and the upper bound can be further updated as $\omega < 2.3728596$, it is in practice useless for its hidden huge constant factor. In the preliminary analysis, the designers of Rasta [23] adopted $\omega = 2.8$ to compute the time complexity of algebraic attacks on reduced-round Agrasta and compared it with the required number of binary operations to encrypt a plaintext. The designers of Dasta [35] instead chose $\omega = 2.37$ to evaluate the resistance against algebraic attacks in order to explicitly understand the security margins of Dasta and Rasta. Therefore, in this paper, we provide the time complexity under both cases, i.e. $\omega = 2.8$ and $\omega = 2.37$. It should be emphasized that the former one is reasonable in practice.

Our results. According to the Rasta paper [23], performing r rounds of Rasta with block size n requires about $(r+1)n^2$ binary operations caused by the linear layers. In our algebraic attacks, the number of equations is always kept the same with the number of variables and it is denoted by U, even though we are able to collect more equations. When evaluating the time complexity with $\omega = 2.8$, we

adopt the formula $U^\omega / \left((r+1)n^2\right)$ as in [23]. When $\omega = 2.37$ is used, we directly compute the time complexity with the formula U^ω as in [35]. The corresponding memory complexity is obvious, i.e. U^2. Our results are summarized in Table 1.

Organization. We briefly introduce Rasta, Dasta and the trivial linearization attack in Section 2. Then, we describe how to construct exploitable low-degree equations from the raw definition of the χ operation in Section 3. The application of these low-degree equations to the cryptanalysis of Rasta and Dasta will be explained in Section 4. Before concluding the paper in Section 6, we will also discuss in Section 5 why our attacks are overlooked, the application of others techniques such as the polynomial-based method [20] and the optimized exhaustive search [14], and the experimental results.

2 Preliminaries

In this section, we briefly describe the overall structure of Rasta and Dasta. Since several instances are specified, they will be distinguished with the notations Rasta-κ-r and Dasta-κ-r, where κ and r denote the claimed security level and the total number of rounds, respectively. In addition, throughout this paper, n denotes the block size, $rank(M)$ denotes the rank of the matrix M, M^{-1} denotes the inverse of the matrix M, a_i denotes the i-th bit of the vector a, $Deg(f)$ denotes the degree of the function f. In addition, we define

$$max(p,q) = \begin{cases} p & (p \geq q) \\ q & (p < q) \end{cases}$$

2.1 Description of Rasta

Rasta is a stream cipher based design where the nonlinear layer is deterministic while the linear layer is randomly generated during the encryption phase. Specifically, its input consists of a key $K \in \mathbb{F}_2^n$, a nonce N, a counter C and a message block $m \in \mathbb{F}_2^n$. To encrypt m, Rasta first randomly generates a concrete instance with SHAKE-256 taking (N, C) as input. Then this instance is utilized to encrypt K to generate the keystream $Z \in \mathbb{F}_2^n$. Finally, $c = m \oplus Z$ is corresponding ciphertext block.

Formally, the keystream Z can be defined in the following way:

$$Z = (A_{r,N,C} \circ S \circ A_{r-1,N,C} \circ S \circ \ldots \circ A_{1,N,C} \circ S \circ A_{0,N,C}(K)) \oplus K,$$

where $A_{i,N,C}$ is an affine mapping and S is the large-scale χ operation. The corresponding illustration can be referred to Figure 1.

Nonlinear layer $y = S(x)$. Denote the input and output of the nonlinear layer by $x = (x_0, x_1, \ldots, x_{n-1}) \in \mathbb{F}_2^n$ and $y = (y_0, y_1, \ldots, y_{n-1}) \in \mathbb{F}_2^n$, respectively. In this way, $y = S(x)$ can be specified as follows:

$$y_i = x_i \oplus \overline{x_{i+1}} x_{i+2},$$

Table 1: Summary of the attacks on Rasta, Dasta and Agrasta, where R, D, M and T denote the number of attacked rounds, data complexity, memory complexity and time complexity, respectively. The number of rounds marked with $\star$ means that the corresponding time complexity exceeds the claimed security level. We recomputed the time/data complexity of the trivial linearization attacks in [35] to keep consistent with our calculations and the results only slightly differ.

Target	Methods	n	R	$\log_2$D	$\log_2$M	$\log_2$T	$\log_2 U$	ω	Reference
Agrasta-128-4	brute-force	129	3	0	25	124.2	-	-	[26]
	linearization	129	3	0	14	125.76	7	2.8	[23]
	linearization	129	**4**	35.7	90	**<u>110</u>**	45	**2.8**	this paper
Agrasta-256-5	brute-force	257	3	0	25	252.2	-	-	[26]
	linearization	257	3	0	16	253.5	8	2.8	[23]
	linearization	257	**5**	76.7	174	**<u>225.1</u>**	87	**2.8**	this paper
Rasta/Dasta-80-6		219	2	19.3	54	64	27	2.37	[35]
Rasta-80-6		219	**3**	22	64	75.9	32	2.37	this paper
Dasta-80-6	linearization	219	**3**	27	54	65	27	2.37	this paper
Rasta-80-6		219	**3**	22	64	**<u>72.1</u>**	32	**2.8**	this paper
Dasta-80-6		219	**3**	27	54	**<u>59.1</u>**	27	**2.8**	this paper
Rasta/Dasta-80-4		327	2	20.7	58	68.8	29	2.37	[35]
Rasta-80-4		327	$3^\star$	24.4	70	83	35	2.37	this paper
Dasta-80-4	linearization	327	**3**	29	58	69.8	29	2.37	this paper
Rasta-80-4		327	**3**	24.4	70	**<u>79.3</u>**	35	**2.8**	this paper
Dasta-80-4		327	**3**	29	58	**<u>62.5</u>**	29	**2.8**	this paper
Rasta/Dasta-128-6		351	3	44.6	106	125.6	53	2.37	[35]
Rasta-128-6	linearization	351	$4^\star$	47.3	116	137.5	58	2.37	this paper
Dasta-128-6		351	**4**	53	106	126.6	53	2.37	this paper
Rasta/Dasta-128-5		525	2	23	64	75.9	32	2.37	[35]
Rasta-128-5		525	**3**	27.7	78	92.5	39	2.37	this paper
Dasta-128-5	linearization	525	**3**	32	64	76.9	32	2.37	this paper
Rasta-128-5		525	**3**	27.7	78	**<u>89.2</u>**	39	**2.8**	this paper
Dasta-128-5		525	**3**	32	64	**<u>70.6</u>**	32	**2.8**	this paper
Rasta/Dasta-128-4		1877	2	28.2	78	92.5	39	2.37	[35]
Rasta-128-4		1877	**3**	34.9	96	113.8	48	2.37	this paper
Dasta-128-4	linearization	1877	**3**	39	78	93.5	39	2.37	this paper
Rasta-128-4		1877	**3**	34.9	96	**<u>111.4</u>**	48	**2.8**	this paper
Dasta-128-4		1877	**3**	39	78	**<u>87.2</u>**	39	**2.8**	this paper
Rasta/Dasta-256-6		703	4	97.6	214	253.6	107	2.37	[35]
Rasta-256-6	linearization	703	$5^\star$	101.3	226	267.9	113	2.37	this paper
Dasta-256-6		703	**5**	107	214	254.6	107	2.37	this paper
Rasta/Dasta-256-5		3545	3	68.3	160	189.7	80	2.37	[35]
Rasta-256-5		3545	**4**	73.9	176	208.6	88	2.37	this paper
Dasta-256-5	linearization	3545	**4**	80	160	190.7	80	2.37	this paper
Rasta-256-5		3545	**4**	73.9	176	**<u>221.4</u>**	88	**2.8**	this paper
Dasta-256-5		3545	**4**	80	160	**<u>200</u>**	80	**2.8**	this paper

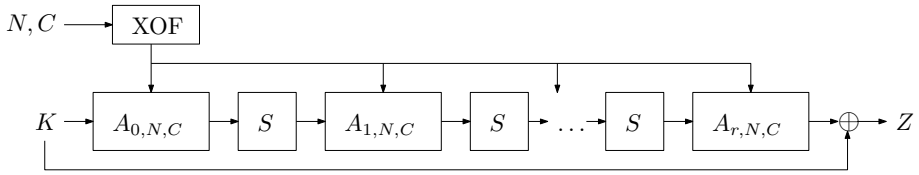

Fig. 1: Illustration of r rounds of Rasta

where $0 \leq i \leq n-1$ and the indices are considered within modulo n. For convenience, such a function $y = S(x)$ is defined as **the n-bit χ operation**. To make $y = S(x)$ bijective, n must be odd. It is also known that the degree of the inverse of the n-bit χ operation is $(n-1)/2+1$. It should be mentioned that the 5-bit χ operation is the S-box used in the Keccak round function [10].

Affine layers $u = A_{i,N,C}(v)$. Denote the input and output of the affine layers by $v \in \mathbb{F}_2^n$ and $u \in \mathbb{F}_2^n$, respectively. The affine mapping $u = A_{i,N,C}(v)$ is a binary multiplication of an $n \times n$ matrix $M_{r,N,C}$ with the n-bit input v, followed by the addition of an n-bit round constant $RC_{i,N,C}$, i.e.

$$u = M_{i,N,C} \cdot v \oplus RC_{i,N,C}.$$

A feature of Rasta is that both $M_{i,N,C}$ and $RC_{i,N,C}$ are not specified in advance. Instead, when a message block is to be encrypted, the corresponding message block counter C and a nonce N is taken as the input of SHAKE-256 and the output of SHAKE-256 will be used to fill $M_{i,N,C}$ and $RC_{i,N,C}$ such that $rank(M_{i,N,C}) = n$ $(0 \leq i \leq r)$.

The data limit. To resist against algebraic attacks, it is explicitly specified in [23] that the largest number of n-bit message blocks that can be encrypted under the same key is $\sqrt{2^\kappa}/n$ for the instance parameterized with (n, κ, r).

The instances. The designers have recommended several instances that can be implemented in practical time in [23], as shown in Table 2.

In addition to the above recommended instances, the authors also proposed aggressive versions called Agrasta with $n = \kappa + 1$, as listed in Table 3. For simplicity, Agrasta parameterized with (κ, r) is denoted by Agrasta-κ-r. From the following statement by the designers, it is easy to see that the data limit remains the same for Agrasta, i.e. $\sqrt{2^\kappa}/n$. We will give a detailed explanation later.

"[23]Agrasta has a block size of 81-bit for 80-bit security having 4 rounds, 129-bit for 128-bit security having 4 rounds and 257-bits for 256-bit security having 5 rounds (in this case trivial linearization would work for 4 rounds)."

Table 2: Parameters of Rasta

κ	n	r
80	327	4
	327	5
	219	6
128	1877	4
	525	5
	351	6
256	445939	4
	3545	5
	703	6

Table 3: Parameters of Agrasta

κ	n	r
80	81	4
128	129	4
256	257	5

2.2 Description of Dasta

Dasta is in general the same with Rasta and we therefore do not distinguish the used notations. Formally, the keystream Z of Dasta is defined as follows:

$$Z = (L \circ P_{r,C} \circ S \circ L \circ P_{r-1,C} \circ S \circ \ldots \circ L \circ P_{1,C} \circ S \circ L \circ P_{0,C}(K)) \oplus K,$$

where L is a fixed $n \times n$ binary matrix while $P_{i,C}$ $(0 \leq i \leq r)$ is an ever-changing bit permutation parameterized with (i, C) and a fixed bit permutation P. The construction of Dasta is depicted in Figure 2.

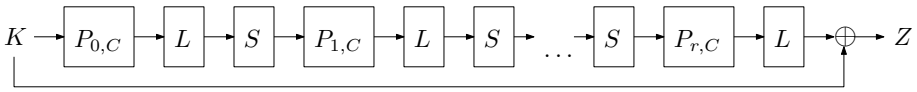

Fig. 2: Illustration of r rounds of Dasta

Our attacks are irrelevant to the details of L and $P_{i,C}$ and hence their details are omitted. The only thing we would like to emphasize is that $P_{i,C}$ is continuously changing, but it is always a bit permutation.

Differences between Rasta and Dasta. One difference is that there is no constant addition operation in Dasta. Therefore, the encryption will output failure when K is 0. Another difference is that the linear layer is composed of an ever-changing bit permutation and a deterministic linear transform. Such a way to construct linear layers will obviously significantly improve the performance of

Rasta as there is no need to use SHAKE-256 to generate a random $n \times n$ full-rank binary matrix, which is quite time-consuming. Finally, Dasta only specifies 7 instances as shown below:

$$\begin{aligned}(n, \kappa, r) \in \{&(327, 80, 4), (219, 80, 6),\\ &(1877, 128, 4), (525, 128, 5), (351, 128, 6),\\ &(3545, 256, 5), (703, 256, 6)\}.\end{aligned}$$

The parameter $(n, \kappa, r) = (445939, 256, 4)$ is not taken into account in Dasta for its huge matrix size. For this reason, the attack on Rasta with such a parameter is not included in our results, though it is trivial to derive it based on our analysis.

2.3 Trivial Linearization Attacks

Due to the special construction of Dasta and Rasta, the conventional cryptanalysis techniques such as differential attacks, higher-order differential attacks, cube attacks and integral attacks immediately become infeasible as they all require the attackers to collect a sufficiently large number of plaintext-ciphertext pairs under the same key for a fixed concrete instance. Notice that when encrypting different message blocks under the same key, both primitives behave like moving targets, i.e. different message blocks are encrypted with different concrete instances.

Consequently, the designers of Rasta [23] made a comprehensive study on a more potential threat, namely the algebraic attack. However, all the reported results derived from the linearization attack, guess-and-determine attack and Gröbner basis attack are negative. In the Dasta document [35], the designers clearly described the number of rounds that the algebraic attacks can reach, as already mentioned in Table 1. As the time complexity of the Gröbner basis attack cannot be well estimated once the equation system becomes much overdefined, it is not surprising that the resistance against the linearization attack whose time complexity can be easily computed become a main concern of the designers. Indeed, the parameters of Rasta are chosen based on the resistance against the linearization attack, though the designers estimate the complexity to solve a large-scale linear equation system in a very conservative way, i.e. $O(1)$.

Since our results are indeed based on the linearization attack, it is necessary to describe how the designers performed such an attack on Dasta and Rasta. Due to the high degree of the inverse of the χ operation, the designers only considered the nonlinear equations in terms of the key in the forward direction. Specifically, if the total number of rounds is reduced to r_0 rounds, according to the keystream $Z = (z_0, z_1, \ldots, z_{n-1})$, the attackers are able to collect the following n nonlinear equations in terms of the key $K = (k_0, k_1, \ldots, k_{n-1})$:

$$\begin{cases} F_0(k_0, k_1, \ldots, k_{n-1}) \oplus z_0 = 0 \\ F_1(k_0, k_1, \ldots, k_{n-1}) \oplus z_1 = 0 \\ \qquad\qquad \cdots \\ F_{n-1}(k_0, k_1, \ldots, k_{n-1}) \oplus z_{n-1} = 0 \end{cases} \tag{1}$$

It is trivial to deduce that $Deg(F_i) \leq 2^{r_0}$ $(0 \leq i \leq n-1)$ as the degree of the χ operation is 2. Although an attacker cannot collect many plaintext-ciphertext pairs under the same key for a fixed concrete instance in both primitives, he is able to collect many such pairs under the same key for many different instances and the number of such pairs is upper bounded by the data limit $\sqrt{2^\kappa}/n$.

A trivial linearization attack is to collect $\sum_{i=0}^{2^{r_0}} \binom{n}{i}$ such equations. Then, by renaming all the high-degree terms as new variables, the attacker indeed could construct $\sum_{i=0}^{2^{r_0}} \binom{n}{i}$ linear equations in terms of $\sum_{i=0}^{2^{r_0}} \binom{n}{i}$ variables. Solving such an equation system requires time complexity

$$\mathcal{T}(n, r_0, \omega) = \left(\sum_{i=0}^{2^{r_0}} \binom{n}{i} \right)^{\omega}.$$

The designers of Rasta also mentioned a guess-and-determine attack. Specifically, after guessing υ key bits, the attacker only needs to collect

$$\sum_{i=0}^{2^{r_0}} \binom{n-\upsilon}{i}$$

equations. Solving such an equation system would require time complexity

$$2^{\upsilon} \cdot \left(\sum_{i=0}^{2^{r_0}} \binom{n-\upsilon}{i} \right)^{\omega}.$$

It is not difficult to observe that guessing variables is not a clever choice if taking the algebra constant ω into account as

$$2^{\upsilon} \cdot \left(\sum_{i=0}^{2^{r_0}} \binom{n-\upsilon}{i} \right)^{\omega}$$

tends to increase as υ increases when n is large and 2^{r_0} is small, which is indeed the case of Rasta, Dasta and Agrasta.

The effect of the trivial linearization attack on Rasta and Dasta has been discussed in [35] with $\omega = 2.37$, as displayed in Table 1. To show that Agrasta also resists against this attack vector, we simply calculate the corresponding time complexity with $\omega \in \{2.8, 2.37\}$, as shown below:

$$\begin{aligned}
\mathcal{T}(81, 4, 2.8) &= 2^{153.72}\ , \mathcal{T}(81, 4, 2.37) = 2^{130.113} \\
\mathcal{T}(129, 4, 2.8) &= 2^{186.2}\ , \mathcal{T}(129, 4, 2.37) = 2^{157.605} \\
\mathcal{T}(257, 5, 2.8) &= 2^{379,68}\ , \mathcal{T}(257, 5, 2.37) = 2^{321.372}.
\end{aligned}$$

Even if taking the time to perform the encryption into account, the attack cannot be better than the brute force. As stated by the designers [23], there exists a trivial linearization attack on Agrasta parameterized with $(n, \kappa, r) = (257, 256, 4)$. Indeed, we have

$$\mathcal{T}(257, 4, 2.8) = 2^{232.68}\ ,$$

which means this parameter is insecure. However, it also implies that the data limit $\sqrt{2^\kappa}/n$ also works for Agrasta.

To better understand the data limit, we repeat the designers' description to determine the claimed security level. The attacker can collect at most $\sqrt{2^\kappa}/n \times n = \sqrt{2^\kappa}$ equations. In addition, there are in total

$$\sum_{i=0}^{2^r} \binom{n-\kappa}{i}$$

variables after linearization. It can be found that

$$\sum_{i=0}^{2^r} \binom{n-\kappa}{i} > 2^\kappa$$

for the parameters of Rasta displayed in Table 2. This also shows that the designers made a very conservative estimation of the complexity of Gaussian elimination, i.e. in time $O(1)$, even though that attacker are still unable to collect sufficiently many equations under the data limit.

3 Low-Degree Equations Hidden in the χ Operation

Both the designers of Rasta and Dasta expect that the degree of the equations that the attacker can collect is upper bounded by 2^{r_0} when the number of rounds is reduced to r_0. The main reason is that the inverse of the χ operation is too costly and they directly gave up in this direction. In the following, we demonstrate that there exist exploitable low-degree equations if relating the input and output of the χ operation in a more clever way.

Low-degree exploitable equations. Denote the input and output of the χ operation by $(x_0, x_1, \ldots, x_{n-1})$ and $(y_0, y_1, \ldots, y_{n-1})$, respectively. Consider two consecutive output bits (y_i, y_{i+1}), as shown below:

$$\begin{aligned} y_i &= x_i \oplus \overline{x_{i+1}} x_{i+2}, \\ y_{i+1} &= x_{i+1} \oplus \overline{x_{i+2}} x_{i+3}. \end{aligned}$$

It can be derived that

$$y_{i+1}(y_i \oplus x_i) = 0. \tag{2}$$

Proof. This can be easily proved. As $y_i \oplus x_i = \overline{x_{i+1}} x_{i+2}$, we have

$$y_{i+1}(y_i \oplus x_i) = y_{i+1}\overline{x_{i+1}} x_{i+2} = (x_{i+1} \oplus \overline{x_{i+2}} x_{i+3})\overline{x_{i+1}} x_{i+2} = 0.$$

This completes the proof of Equation 2.

Another very similar useful low-degree equation has been discussed in [34] to mount preimage attacks on reduced-round Keccak, as shown below:

$$y_i \oplus x_i = (y_{i+1} \oplus 1)x_{i+2}. \tag{3}$$

Indeed, Equation 2 can also be derived from Equation 3 if both sides of Equation 3 are multiplied by y_{i+1}.

In addition, we further observed an exploitable cubic boolean equation from our experiments on the small-scale χ operation (e.g. $n \in \{7, 9\}$) with sagemath, as shown in Equation 4. How to perform the experiments will be explained in Section 5.

$$y_{i+3}(y_{i+2}y_{i+1} \oplus y_{i+2} \oplus y_i \oplus x_i) = 0. \tag{4}$$

Proof. From the definition of the χ operation, we have

$$\begin{aligned} y_{i+2}y_{i+1} \oplus y_{i+2} \oplus y_i \oplus x_i &= y_{i+2}\overline{y_{i+1}} \oplus \overline{x_{i+1}}x_{i+2} \\ &= (x_{i+2} \oplus \overline{x_{i+3}}x_{i+4})(\overline{x_{i+1}} \oplus \overline{x_{i+2}}x_{i+3}) \oplus \overline{x_{i+1}}x_{i+2} \\ &= x_{i+2}\overline{x_{i+1}} \oplus \overline{x_{i+1}}x_{i+4}\overline{x_{i+3}} \oplus \overline{x_{i+1}}x_{i+2} \\ &= \overline{x_{i+1}}x_{i+4}\overline{x_{i+3}}. \end{aligned}$$

Hence,

$$y_{i+3}(y_{i+2}y_{i+1} \oplus y_{i+2} \oplus y_i \oplus x_i) = (x_{i+3} \oplus \overline{x_{i+4}}x_{i+5})\overline{x_{i+1}}x_{i+4}\overline{x_{i+3}} = 0.$$

This completes the proof.

$$\begin{cases} y_1y_0 \oplus y_1x_0 = 0 \\ y_1x_2 \oplus y_0 \oplus x_0 \oplus x_2 = 0 \\ y_1(y_0y_{n-1} \oplus y_0 \oplus y_{n-2} \oplus x_{n-2}) = 0 \\ y_2y_1 \oplus y_2x_1 = 0 \\ y_2x_3 \oplus y_2 \oplus x_2 \oplus x_3 = 0 \\ y_2(y_1y_0 \oplus y_1 \oplus y_{n-1} \oplus x_{n-1}) = 0 \\ \quad \cdots \\ y_{i+1}y_i \oplus y_{i+1}x_i = 0 \\ y_{i+1}x_{i+2} \oplus y_i \oplus x_i \oplus x_{i+2} = 0 \\ y_{i+1}(y_iy_{i-1} \oplus y_i \oplus y_{i-2} \oplus x_{i-2}) = 0 \\ \quad \cdots \\ y_{n-1}y_{n-2} \oplus y_{n-1}x_{n-2} = 0 \\ y_{n-1}x_0 \oplus y_{n-2} \oplus x_{n-2} \oplus x_0 = 0 \\ y_{n-1}(y_{n-2}y_{n-3} \oplus y_{n-2} \oplus y_{n-4} \oplus x_{n-4}) = 0 \\ y_0y_{n-1} \oplus y_0x_{n-1} = 0 \\ y_0x_1 \oplus y_0 \oplus x_0 \oplus x_1 = 0 \\ y_0(y_{n-1}y_{n-2} \oplus y_{n-1} \oplus y_{n-3} \oplus x_{n-3}) = 0 \end{cases} \tag{5}$$

The total number of exploitable equations of degree upper bounded by 3. If treating $y_{i+1}x_{i+2}$, $y_{i+1}x_i$, $y_{i+1}y_i$, $y_{i+3}y_i$, $y_{i+3}x_i$ and $y_{i+3}y_{i+2}y_{i+1}$ as new variables, we can say that Equation 2, Equation 3 and Equation 4 are linearly independent. Taking all the input bits into account, we obtain the equation system (5).

It is not difficult to observe that these $3n$ equations are linearly independent if the high-degree terms are treated as new variables. This is because each equation contains one high-degree term that never appears in other equations.

What benefits can be brought by such a system of equations? Imagine the case when the degree of the boolean expressions of the input $x = (x_0, \ldots, x_{n-1})$ and output $y = (y_0, \ldots, y_{n-1})$ of the χ operation in terms of the key bits are upper bounded by $\mathcal{D}_x$ and $\mathcal{D}_y$, respectively. If only the raw definition of the χ operation is taken into account, i.e. the equations are constructed based on

$$y_i = x_i \oplus \overline{x_{i+1}} x_{i+2},$$

the degree of the collected equations will be upper bounded by

$$max(2\mathcal{D}_x, \mathcal{D}_y).$$

However, the equation system (5) can also be utilized to describe the relations between x and y. Moreover, the degree of the equations in the equation system (5) is upper bounded by

$$max(\mathcal{D}_x + \mathcal{D}_y, 3\mathcal{D}_y).$$

If we can know $\mathcal{D}_y = 1$ and $\mathcal{D}_x \geq 2$, there will be

$$max(\mathcal{D}_x + \mathcal{D}_y, 3\mathcal{D}_y) = \mathcal{D}_x + 1 < 2\mathcal{D}_x = max(2\mathcal{D}_x, \mathcal{D}_y).$$

In other words, we could construct equations of much lower degree based on the equation system (5). As the degree of the equations is reduced, the number of all possible monomials in the equations will be reduced to $\sum_{i=0}^{\mathcal{D}_x+1} \binom{n}{i}$ from $\sum_{i=0}^{2\mathcal{D}_x} \binom{n}{i}$, which will be extremely useful to improve the trivial linearization attack where the equations are derived only based on $y_i = x_i \oplus \overline{x_{i+1}} x_{i+2}$.

3.1 A General Approach to Search for Exploitable Equations

The above 3 equations are found manually or by performing experiments on the small-scale χ operation, which are sufficient to devise the attacks in this paper. However, it is still possible to miss similar equations and more equations can be utilized to reduce the data complexity. Therefore, we are motivated to find a more general approach to search for such useful equations. For this purpose, we introduce the notion of **exploitable equation**.

Definition 1. *An exploitable equation is defined as an equation where the input bits of the χ operation are only allowed to form linear terms or quadratic terms with the output bits.*

Now we discuss our general idea to identify more exploitable equations. Consider the vectorial Boolean function $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Suppose $(x_0, x_1, \ldots, x_{n-1})$ be the input of S and $(y_0, y_1, \ldots, y_{n-1})$ be the corresponding output. We aim to find equations involving input variables and output variables of the function S. Suppose our target is to bound the degree of input variables as 1 and degree of the output variables as ℓ. In other words, for any input $(x_0, x_1, \ldots, x_{n-1})$ and output $(y_0, y_1, \ldots, y_{n-1})$, the following relation should hold:

$$a^0 \oplus \sum_{0 \le i < n} b_i^1 x_i \oplus \sum_{0 \le i, j_1 < n} b_{i,j_1}^2 x_i y_{j_1} \oplus \sum_{0 \le j_1 < n} c_{j_1}^1 y_{j_1} \oplus$$
$$\sum_{0 \le j_1 < j_2 < n} {c^2}_{j_1,j_2} y_{j_1} y_{j_2} \oplus \cdots \oplus \sum_{0 \le j_1 < j_2 < \cdots < j_\ell < n} {c^\ell}_{j_1,j_2,\ldots,j_\ell} y_{j_1} y_{j_2} \ldots y_{j_\ell} = 0, \quad (6)$$

where $a^0, b_i^1, b_{i,j_1}^2, c_{j_1}^1, \ldots, c_{j_1,j_2,\ldots,j_\ell}^\ell$ denote coefficients and are in $\mathbb{F}_2$.

Our aim is to identify these coefficients and they are treated as unknown variables. Thus, there are $t = n + n^2 + \sum_{i=0}^{\ell} \binom{n}{i}$ many unknown variables. If $\ell \ll n$, we have $t < 2^n$. Our procedure is to first fix some small odd number n. Next, we generate $t' > t$ many random input $(x_0, x_1, \ldots, x_{n-1})$ output $(y_0, y_1, \ldots, y_{n-1})$ pairs and put these values in Equation 6. Thus we have t' linear equations over $GF(2)$. Each solution of the linear equation system gives a possible option of an exploitable equation. We then generate few more random input-output pairs and check the validity of the expression. If the expression is still valid, we can assume the expression to be valid for any input-output pair. From this expression, we try to estimate the expression for any odd number n. Thus our approach is based on interpolation-guess technique.

From Equation (5), we know there are $3n$ linearly independent exploitable equations. First, we construct a set $\mathcal{S} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{3n}\}$ where each $\mathcal{F}_i$ denotes a different equation among the $3n$ equations. Let $\mathcal{M}$ be the union of monomials of the polynomials of $\mathcal{S}$. Now if a new equation $\mathcal{F}'$ is generated using our interpolation-guess technique and contains at least one monomial outside $\mathcal{M}$, we include $\mathcal{F}'$ in $\mathcal{S}$ and update $\mathcal{M}$. We continue this process for each possible expression using our interpolation-guess technique.

In our interpolation-guess idea[1], we take $\ell = 5$ and $n = 11$. Hence, we are searching for $t = 11 + 11^2 + \sum_{i=0}^{5} \binom{11}{i} = 1156$ many binary variables. From the results, we found the following two simple polynomials:

$$\mathcal{F}_1^i = y_{i+5}\big(x_i \oplus x_{i+2} \oplus y_i \oplus y_{i+1}y_{i+2} \oplus y_{i+1}\overline{y_{i+3}}y_{i+4}\big), \quad (7)$$
$$\mathcal{F}_2^i = y_{i+7}\big(x_i \oplus y_i \oplus \overline{y_{i+1}}y_{i+2} \oplus \overline{y_{i+1}}(y_{i+4} \oplus \overline{y_{i+5}}y_{i+6})\overline{y_{i+3}}\big). \quad (8)$$

Let $f_i = y_i \oplus x_i \oplus \overline{y_{i+1}}x_{i+2}$ for $0 \le i \le n-1$. Then we have $\mathcal{F}_1^i = y_{i+5}(f_i \oplus y_{i+1}f_{i+2} \oplus y_{i+1}\overline{y_{i+3}}f_{i+4})$, $\mathcal{F}_2^i = y_{i+7}(f_i \oplus \overline{y_{i+1}}f_{i+2} \oplus \overline{y_{i+1}}\ \overline{y_{i+3}}f_{i+4} \oplus \overline{y_{i+1}}\ \overline{y_{i+3}}\ \overline{y_{i+5}}f_{i+6})$. From 3, we know $f_i = 0$. Thus we have $2n$ extra relations:

$$y_{i+5}\big(x_i \oplus x_{i+2} \oplus y_i \oplus y_{i+1}y_{i+2} \oplus y_{i+1}\overline{y_{i+3}}y_{i+4}\big) = 0, \quad (9)$$

[1] Obviously, all the $3n$ equations in the equation system (5) can also be detected with this technique if it starts from an empty set $\mathcal{S}$.

$$y_{i+7}\big(x_i \oplus y_i \oplus \overline{y_{i+1}}y_{i+2} \oplus \overline{y_{i+1}}(y_{i+4} \oplus \overline{y_{i+5}}y_{i+6})\overline{y_{i+3}}\big) = 0 \tag{10}$$

Consider the ideal $\mathcal{I} = \langle f_0, \ldots, f_{n-1}\rangle$. It is further found that

$$\begin{aligned}
&\mathcal{F}_1^i \in \mathcal{I}, \mathcal{F}_2^i \in \mathcal{I},\\
&y_{i+1}(y_i \oplus x_i) = y_{i+1}(y_i \oplus x_i \oplus \overline{y_{i+1}}x_{i+2}) \in \mathcal{I},\\
&y_{i+3}(y_{i+2}y_{i+1} \oplus y_{i+2} \oplus y_i \oplus x_i)\\
&= y_{i+3}(y_i \oplus x_i \oplus \overline{y_{i+1}}x_{i+2}) \oplus y_{i+3}\overline{y_{i+1}}(y_{i+2} \oplus x_{i+2} \oplus \overline{y_{i+3}}x_{i+4}) \in \mathcal{I}
\end{aligned}$$

Thus, all the $5n$ useful relations are in $\mathcal{I}$.

Remark. Apart from $5n$ such relations, we obtained many other useful relations for $n = 11, \ell = 5$. However, these expressions are too complicated. Hence, we do not try to generalize them. One interesting observation is that these relations are also in $\mathcal{I}$. We emphasize that our algorithm is very similar to the algorithm proposed by Fischer and Meier at FSE 2007, which is used to compute the algebraic immunity of S-boxes and augmented functions [31]. However, as most similar algorithms [19,12] to search for quadratic boolean functions of a certain S-box based on Gaussian elimination, the algorithm [31] soon becomes impractical for a huge S-box, i.e. the large-scale χ operation. This is because as the size of the S-box increases, the number of possible monomials, i.e. the number of to-be-determined coefficients, will become very large, which will result in high costs of Gaussian elimination. The feasibility of our algorithm contributes to our critical observation that some forms of exploitable equations holding for the small-scale χ operation (with small n) might also apply to the large-scale χ operation (with large n). This directly allows to first search for exploitable equations for the small-scale χ operation, and then to check whether they also hold for the large-scale one.

4 Algebraic Cryptanalysis of Rasta and Dasta

Notice that there exists a key feed-forward phase just before computing the final keystream Z in Rasta and Dasta. This special construction together with the above low-degree exploitable equations will lead to significantly improved linearization attacks.

For simplicity, denote the state after $A_{i,N,C}$ by α^i and the state before $A_{i,N,C}$ by β^i. In this way, the state transitions in Rasta can be described as follows:

$$K = \beta^0 \xrightarrow{A_{0,N,C}} \alpha^0 \xrightarrow{S} \beta^1 \xrightarrow{A_{1,N,C}} \alpha^1 \xrightarrow{S} \ldots \xrightarrow{A_{r-1,N,C}} \alpha^{r-1} \xrightarrow{S} \beta^r \xrightarrow{A_{r,N,C}} \alpha^r$$

For Dasta, similarly, denote the state after $P_{i,C}$ by ρ^i, the state after L by π^i and the state before $P_{i,C}$ by λ^i. In this way, the state transitions in Dasta can be expressed as follows:

$$\lambda^0 \xrightarrow{P_{0,C}} \rho^0 \xrightarrow{L} \pi^0 \xrightarrow{S} \lambda^1 \xrightarrow{P_{1,C}} \rho^1 \xrightarrow{L} \pi^1 \xrightarrow{S} \ldots \xrightarrow{L} \pi^{r-1} \xrightarrow{S} \lambda^r \xrightarrow{P_{r,C}} \rho^r \xrightarrow{L} \pi^r,$$

where $K = \lambda^0$.

4.1 Constructing Low-degree Equations for Rasta

First of all, we discuss the attacks on r_0 rounds of Rasta. In the forward direction, α^{r_0-1} can be written as boolean expressions in terms of the key. Denote the expression of $\alpha_i^{r_0-1}$ $(0 \leq i \leq n-1)$ in terms of $K = (k_0, k_1, \ldots, k_{n-1})$ by $g_i(k_0, k_1, \ldots, k_{n-1})$, i.e.

$$\alpha_i^{r_0-1} = g_i(k_0, k_1, \ldots, k_{n-1}).$$

As the degree of the χ operation is 2, we have

$$Deg(g_i) \leq 2^{r_0-1}. \tag{11}$$

According to the plaintext-ciphertext pair (m, c), the corresponding keystream Z can be computed with $Z = m \oplus c$. Since

$$\begin{aligned} \alpha^{r_0} &= Z \oplus K, \\ \alpha^{r_0} &= M_{r_0,N,C} \cdot \beta^{r_0} \oplus RC_{r_0,N,C}, \end{aligned}$$

we have

$$\beta^{r_0} = M_{r_0,N,C}^{-1} \cdot (m \oplus c \oplus K \oplus RC_{r_0,N,C}).$$

In other words, in the backward direction, β^{r_0} can be written as linear expressions in terms of K. For simplicity, denote the corresponding linear expression of $\beta_i^{r_0}$ $(0 \leq i \leq n-1)$ by $h_i(k_0, k_1, \ldots, k_{n-1})$, i.e.

$$\beta_i^{r_0} = h_i(k_0, k_1, \ldots, k_{n-1}).$$

Hence, we have

$$Deg(h_i) = 1. \tag{12}$$

Notice that

$$\beta^{r_0} = S(\alpha^{r_0-1}).$$

Hence, according to Equation 2, Equation 3, Equation 4, Equation 9 and Equation 10, the following low-degree equations can be derived:

$$\begin{aligned} h_{i+1} \cdot h_i \oplus h_{i+1} \cdot g_i &= 0, \\ h_i \oplus g_i \oplus h_{i+1} \cdot g_{i+2} \oplus g_{i+2} &= 0, \\ h_{i+3}(h_{i+2}h_{i+1} \oplus h_{i+2} \oplus h_i \oplus g_i) &= 0, \\ h_{i+5}\big(g_i \oplus g_{i+2} \oplus h_i \oplus h_{i+1}h_{i+2} \oplus h_{i+1}\overline{h_{i+3}}h_{i+4}\big) &= 0, \\ h_{i+7}\big(g_i \oplus h_i \oplus \overline{h_{i+1}}h_{i+2} \oplus \overline{h_{i+1}}(h_{i+4} \oplus \overline{h_{i+5}}h_{i+6})\overline{h_{i+3}}\big) &= 0. \end{aligned}$$

where the indices are considered within modulo n. Based on Equation 11 and Equation 12, it can be found that the degree of the above 5 equations is upper bounded by

$$\mathcal{D} = max(Deg(g_i) + Deg(h_i), 5Deg(h_i)) = max(2^{r_0-1} + 1, 5).$$

When $r_0 \geq 3$, which is the case in our attacks[2], we have

$$\mathcal{D} = 2^{r_0-1} + 1. \tag{13}$$

As h_i is linearly independent from each other and g_i can also be viewed as linearly independent from each other once all high-degree monomials are renamed with new variables, we can then construct $5n$ linearly independent equations in terms of the key K for each pair (m, c). Different from the designers' analysis, the degree of our $5n$ equations is upper bounded by $2^{r_0-1} + 1$ rather than 2^{r_0}. This is a great reduction in the number of all possible monomials, i.e. reduced from $\sum_{i=0}^{2^{r_0}} \binom{n}{i}$ to $\sum_{i=0}^{2^{r_0-1}+1} \binom{n}{i}$. Obviously, such a reduction contributes to our clever way to utilize the low-degree equations discussed in Section 3.

Linearization attacks on reduced-round Rasta. The attacks are now quite straightforward. Specifically, the attacker collects sufficiently many plaintext-ciphertext pairs. For each pair, $5n$ equations in terms of K can be constructed and the degree of these equations is upper bounded by $\mathcal{D}$ (Equation 13). To solve this equation system, the linearization technique is applied. As a result, the time complexity T_0 and data complexity D_0 of our attacks on r_0 rounds of Rasta can be formalized as follows, where U denotes the maximal number of possible monomials.

$$U = \sum_{i=0}^{2^{r_0-1}+1} \binom{n}{i}, T_0 = U^{\omega}, D_0 = U/(3n).$$

As the maximal number of message blocks that can be encrypted under the same key is $\sqrt{2^{\kappa}}/n$, we need to ensure

$$D_0 = \left(\sum_{i=0}^{2^{r_0-1}+1} \binom{n}{i} \right) / (5n) < \sqrt{2^{\kappa}}/n \rightarrow \left(\sum_{i=0}^{2^{r_0-1}+1} \binom{n}{i} \right) < 5\sqrt{2^{\kappa}}. \tag{14}$$

In addition, as mentioned before, when the time complexity is evaluated with the algebra constant $\omega = 2.8$, the final time complexity will be computed with Equation 15, i.e. the time to encrypt a plaintext requires about $(r_0+1)n^2$ binary operations for r_0 rounds of Rasta.

$$T_0' = \left(\sum_{i=0}^{2^{r_0-1}+1} \binom{n}{i} \right)^{2.8} / \left((r_0+1)n^2 \right) \tag{15}$$

When the time complexity is evaluated with $\omega = 2.37$ as in [35], the time complexity will be directly computed with

$$T_0 = \left(\sum_{i=0}^{2^{r_0-1}+1} \binom{n}{i} \right)^{2.37}. \tag{16}$$

[2] For $r_0 = 2$, we then only use Equation 2, Equation 3 and Equation 4.

To violate the claimed security levels, it is essential to require

$$T_0' < 2^\kappa \tag{17}$$

when $\omega = 2.8$ or

$$T_0 < 2^\kappa \tag{18}$$

when $\omega = 2.37$.

Based on the formulas Equation 15, Equation 17 and Equation 14, we directly break 2 out of 3 instances of Agrasta. In addition, the trivial linearization attacks on Rasta taking the parameters

$$(n, \kappa, r) \in \{(327, 80, 4), (1877, 128, 4), (3545, 256, 5)\}$$

are significantly improved, which directly reduces the security margins of these instances to only 1 round.

If evaluating the complexity with Equation 16 and Equation 14 as in [23], under the constraint Equation 18, almost all linearization attacks described in [23] are improved by one round. All the results are summarized in Table 1.

Remark. For the high-degree nonlinear function, the designers should make a careful investigation of whether low-degree equations exist. For Rasta, the degree of the inverse of the χ operation is very high. However, this does not mean that we cannot derive useful low-degree equations if considering the relations between the input bits and output bits in a more careful way, which is obviously neglected by the designers. Especially, when the design has an additional structure, the neglected useful equations will become potential threats to the security.

4.2 Constructing Low-degree Equations for Dasta

The above results can be trivially applied to Dasta. However, we further observe that the last linear layer of Dasta is constructed in the way to apply a bit permutation followed by a fixed linear transform. In the following, we describe how to exploit this feature to further obtain nonlinear equations of lower degree.

Based on similar analysis, when the target is r_0 rounds of Dasta, from the forward direction, π^{r_0-1} can be written as expressions in terms of K and the degree of these equations is upper bounded by 2^{r_0-1}. In the backward direction, both λ^{r_0} and ρ^{r_0} can be written as linear expressions in terms of K.

Firstly, focus on the expressions of ρ^{r_0}. It can be derived that

$$\rho^{r_0} = L^{-1} \cdot (m \oplus c \oplus K) = L^{-1} \cdot (m \oplus c) \oplus L^{-1} \cdot K.$$

Let

$$\sigma = L^{-1} \cdot K. \tag{19}$$

It can be found that the expressions of σ_i $(0 \leq i \leq n-1)$ remain invariant due to the usage of a fixed linear transform L. As

$$\rho^{r_0} = L^{-1} \cdot (m \oplus c) \oplus \sigma,$$

under different (m, c), the expressions of ρ^{r_0} only vary in the constant parts. As λ^{r_0} is just a bit permutation on ρ^{r_0}, we have that the set of expressions of λ^{r_0} also only vary in the constant parts that only depend on (m, c).

In other words, if guessing one bit of σ, we can always find a bit of λ^{r_0} that can be uniquely determined based on this guess. More specifically, since the bit permutation may change when different message blocks are encrypted, a fixed guessed bit of σ will always lead to a computable bit of λ^{r_0} whose bit position is not fixed. How to exploit this fact to improve the attacks on Dasta is detailed as follows.

Linearization attacks on reduced-round Dasta. Denote the expression of $\lambda_i^{r_0}$ by $h'_i(k_0, k_1, \ldots, k_{n-1})$ and the expression of $\pi_i^{r_0-1}$ by $g'_i(k_0, k_1, \ldots, k_{n-1})$ $(0 \leq i \leq n-1)$. Similarly, we have

$$Deg(h'_i) = 1, Deg(g'_i) \leq 2^{r_0-1}.$$

Based on the above analysis, guessing a fixed bit of σ will lead to a determined bit of λ^{r_0}, though its position is not fixed and is indeed a moving position. However, we can always find a bit λ^{r_0} that can be determined. Since

$$\lambda^{r_0} = S(\pi^{r_0-1}),$$

according to Equation 3, we can deduce that

$$h'_i \oplus g'_i = (h'_{i+1} \oplus 1)g'_{i+2}. \tag{20}$$

Based on Equation 4, we have

$$h'_{i+1}(h'_i h'_{i-1} \oplus h'_i \oplus h'_{i-2} \oplus g'_{i-2}) = 0. \tag{21}$$

In addition, based on Equation 9 and Equation 10, we further have

$$h'_{i+1}\left(g'_{i-4} \oplus g'_{i-2} \oplus h'_{i-4} \oplus h'_{i-3}h'_{i-2} \oplus h'_{i-3}\overline{h'_{i-1}}h'_i\right) = 0. \tag{22}$$

$$h'_{i+1}\left(g'_{i-6} \oplus h'_{i-6} \oplus \overline{h'_{i-5}}h'_{i-4} \oplus \overline{h'_{i-5}}(h'_{i-2} \oplus \overline{h'_{i-1}}h'_i)\overline{h'_{i-3}}\right) = 0. \tag{23}$$

Therefore, if the value of the expression h'_{i+1} is known, an equation of degree upper bounded by 2^{r_0-1} can be constructed based on Equation 20, further reducing the degree by 1. If $h'_{i+1} = 1$, three more equations of degree upper bounded by 2^{r_0-1} can be derived from Equation 21, Equation 22 and Equation 23 given that $r_0 \geq 3$.

As mentioned several times, once a fixed bit of σ is guessed, there always exists a bit of λ^{r_0} that can be uniquely determined. In other words, we can always find an expression h'_{i+1} whose value can be uniquely calculated based on the

guessed bit. However, different from the attacks on Rasta, the number of useful equations of degree upper bounded by 2^{r_0-1} is 4 for each plaintext-ciphertext pair. Among the 4 equations, one can always be constructed, while whether the remaining three equations can be constructed will depend on the collected plaintext-ciphertext pair. Therefore, to make our results more convincing, we only use the probability-1 equation derived from Equation 20. Therefore, the data complexity of our attack on Dasta is just an upper bound.

The attacks now become quite straightforward. Specifically, denote the data complexity and time complexity by D_1 and T_1, respectively. As we only aim at equations of degree upper bounded by 2^{r_0-1}, the maximal number of possible monomials is

$$U = \sum_{i=0}^{2^{r_0-1}} \binom{n}{i}.$$

Since only 1 equation is useful for a pair (m, c), we have

$$D_1 = \sum_{i=0}^{2^{r_0-1}} \binom{n}{i}.$$

As we need to guess a bit of σ, the time complexity is computed as follows:

$$T_1 = 2 \times \left(\sum_{i=0}^{2^{r_0-1}} \binom{n}{i} \right)^{\omega}.$$

Again, when $\omega = 2.8$, the time complexity is refined as

$$T_1' = 2 \times \left(\sum_{i=0}^{2^{r_0-1}} \binom{n}{i} \right)^{2.8} / \left((r_0+1)n^2 \right).$$

The time complexity should not exceed the claimed security level. The data complexity cannot exceed the data limit. Under the two constraints, we can significantly improve the linearization attacks on reduced-round Dasta, as shown in Table 1. It is not surprising to find that the attacks become more powerful as the degree decreases.

Countermeasures. A countermeasure to keep Dasta as secure as Rasta is to swap the bit permutation and linear transform in the last linear layer. In addition, the bit permutation should always be different when different message blocks are encrypted under the same key, which is indeed the strategy used in the first linear layer of Dasta. In this case, under different (m, c), the attacker needs to guess different bits in order to collect one equation of degree upper bounded by 2^{r_0-1}, which is obviously more time-consuming than the attacks based on equations of degree upper bounded by $2^{r_0-1} + 1$.

5 Discussions

The presented attack is surprisingly simple and can be treated as a generic attack on Rasta-like constructions. It should be emphasized that such a simple generic attack has remained undiscovered since the publication of Rasta [23] at CRYPTO 2018 and that designing and analyzing symmetric-key primitives for advanced protocols is an active field in recent years. Especially, Equation 3 has been frequently exploited to mount preimage attacks on reduced-round Keccak [10] since the linear structure of Keccak was proposed at ASIACRYPT 2016 [34], though it is always interpreted in another way due to the sponge construction. Specifically, as the 5-bit χ operation is adopted in Keccak, Equation 3 is always interpreted as follows in the context of preimage attacks:

Observation 1 *[34] When l $(1 < l < 5)$ consecutive output bits of the 5-bit S-box are known, there exist $l-1$ linear equations* ***only in terms of the input bits*** *holding with probability 1.*

The reason to construct equations only in terms of the input bits is that some output bits of the 5-bit S-box are unknown to adversaries and the degree of their expressions in terms of the message bits is very high. Therefore, equations like

$$\begin{aligned} y_{i+1}(y_i \oplus x_i) &= 0, \\ y_i \oplus x_i \oplus (y_{i+1} \oplus 1)x_{i+2} &= 0, \\ y_{i+3}(y_{i+2}y_{i+1} \oplus y_{i+2} \oplus y_i \oplus x_i) &= 0 \end{aligned}$$

are not friendly to attacks when only y_i is known to adversaries. Otherwise, the involved equations will contain more unknown variables (e.g. y_{i+1}) or the degree of the constructed equations in terms of the message bits will increase, both of which will have negative influences on the preimage attacks.

Based on the above fact, it is imaginable why the presented attack in this paper is overlooked. Specifically, due to the key feed-forward operation in Rasta, none of the output bits of the last χ operation is known, even though it is very easy to observe that these output bits are linear in the key bits in the backward direction. Hence, the above widely-used observation does not apply anymore as it requires known output bits of the χ operation and guessing output bits is too costly for Rasta.

Our simple attacks also demonstrate that the designers should make a thorough study on the new components in their innovative proposals, e.g. the large-scale χ operation in Rasta and Dasta. Indeed, finding a set of quadratic boolean equations satisfying a given S-box in terms of the input and output bits is well-known since the algebraic attack on AES [19], though our attacks require some special equations where the input bits are only allowed to form linear terms or quadratic terms with the output bits. We could only imagine that the large-scale χ operation is too large to handle, thus making the exploitable low-degree equations neglected.

However, dealing with a small-scale χ operation is sufficient and such equations can be easily observed. Indeed, there is an interface[3] in sagemath to compute the reduced Gröebner basis of the quadratic polynomials satisfying a given S-box, i.e. `sbox.polynomials(groebner=True)`. This function first computes a set of polynomials of degree upper bounded by 2 satisfying a given S-box with the method in [12] and then computes the reduced Gröebner basis for the obtained polynomials. We tested the 7-bit and 9-bit χ operations and observed Equation 4. We argue that this is not a general method and we may miss some exploitable equations. We recommend to use the dedicated approach discussed in Section 3 to search for more exploitable equations, which is also based on the idea to detect equations in the small-scale χ operation and then to further verify them for the large-scale χ operation.

Indeed, Equation 2, Equation 3 and Equation 4 are sufficient to mount attacks on full Agrasta, Rasta and Dasta. With the general approach to search for more complicated exploitable equations, the data complexity can be reduced as more equations can be constructed based on a plaintext-ciphertext pair. However, the final time complexity and memory complexity of the linearization attack will remain the same. Moreover, it seems that the number of exploitable equations of degree upper bounded by a certain value is still small and the data complexity cannot be significantly reduced.

5.1 On the Polynomial Method [20]

Recently, based on the polynomial method [13,21,38], an improved generic method to solve multivariate equation systems over $GF(2)$ is proposed [20]. The conclusion is that the time complexity and memory complexity of solving systems of equations in terms of $\mathcal{N}$ variables are $\mathcal{N}^2 \cdot 2^{(1-1/2.7D)\mathcal{N}}$ bit operations and $\mathcal{N}^2 \cdot 2^{(1-1/1.35D)\mathcal{N}}$ bits, respectively, where D represents the upper bound of the degree of the equations. A disadvantage of such a generic method is that it cannot benefit from an overdefined system of equations.

When such a method is applied to Agrasta-128-4 and Agrasta-256-5, based on our way to construct low-degree equations, the memory complexity of the corresponding attacks is $129 \times 129 \times 2^{118.4} \approx 2^{132.4}$ and $257 \times 257 \times 2^{245.8} \approx 2^{261.8}$ bits, respectively. Thus, it is not better than the generic attack and requires much more memory than ours. In addition, as mentioned in [20], an optimized exhaustive search algorithm [14] for solving polynomial systems of degree D over $GF(2)$ requires $2D\log_2\mathcal{N} \cdot 2^{\mathcal{N}}$ bit operations. In other words, based on our way to construct low-degree equations, the optimized exhaustive search for Agrasta-128-4 and Agrasta-256-5 requires at least 2^{136} and 2^{265} bit operations, respectively. For the technique in [20], without guessing key bits, it requires $2^{137.7}$ and 2^{267} bit operations, respectively. Guessing key bits will increase the time complexity and hence the technique in [20] will not be faster than the optimized exhaustive search. If counting the number of bit operations for our linearization

[3] `https://doc.sagemath.org/html/en/reference/cryptography/sage/crypto/sbox.html`

attacks on Agrasta-128-4 and Agrasta-256-5, we need about $2^{45\times 2.8} = 2^{126}$ and $2^{87\times 2.8} = 2^{243.6}$ bit operations, respectively, which are still significantly below that of the optimized exhaustive search.

For attacks on reduced-round Rasta and Dasta based on the proposed polynomial method [20] or the optimized exhaustive search [14], even with our method to construct low-degree equations, the corresponding memory complexity and time complexity will be much higher than the claimed security level because n is much larger than κ. This shows the advantage of the linearization attacks which can greatly benefit from an over-defined system of equations.

5.2 Experimental Verification

The main concern of the linearization attack is whether the constructed equations are indeed linearly independent. To address it, we performed some experiments[4] on the small-state Rasta with small n for $r_0 \in \{2, 3\}$. Notice that the number of possible monomials increases very fast as the number of rounds increases. Consequently, the experiments are performed on 2 and 3 rounds of Rasta for efficiency. We are aware that the linearization attacks on such instances may not be competitive to the pure brute-force attack. However, we emphasize that the experiments are mainly used to check whether the constructed equations with our method are indeed linearly independent.

For the experiments on 2-round attack, only Equation 2, Equation 3 and Equation 4 will be considered, while Equation 9 and Equation 10 will be included in the 3-round attack. This is because the degree of Equation 9 and Equation 10 is upper bounded by 4 and 5, respectively.

The aim of our experiments is to compute the number of linearly independent equations after gaussian elimination, which is denoted by EQA, i.e. the rank of the coefficient matrix. If it is almost the same with the total number of equations before gaussian elimination, which is denoted by EQB, our assumption on the linear independence between the equations is reasonable. We performed 100 random tests for each small instance, it was found that

$$0 \leq \text{EQB} - \text{EQA} \leq 3, \tag{24}$$

which indicates that our assumption is reasonable. The experimental results are displayed in Table 4.

6 Conclusion

While fully inverting the large-scale χ operation will make the linearization attack worse for its high degree, by carefully studying the relations between its input bits and output bits, we find that there exist some hidden low-degree equations where the input bits are only allowed to form linear terms or quadratic

[4] The source code can be found at `https://anonymous.4open.science/r/AlgebraicAttackOnRasta-21D1`.

Table 4: Experimental results on small-state versions, where $\#(= i)$ represents the number of tests when $\mathrm{EQB} - \mathrm{EQA} = i$ among the 100 tests.

r_0	n	EQB	EQB − EQA			
			$\#(=0)$	$\#(=1)$	$\#(=2)$	$\#(=3)$
2	21	1561	29	54	17	0
2	23	2047	38	55	7	0
2	25	2625	32	51	17	0
2	27	3303	25	63	12	0
2	29	4089	27	56	16	1
3	9	381	25	67	7	1
3	11	1023	27	61	12	0
3	13	2379	25	56	19	0

terms with the output bits. Combined with the key feed-forward operation in Dasta and Rasta, these hidden equations can be utilized to significantly improve the linearization attacks on reduced-round Rasta and Dasta. Especially, the improvement directly allows us to theoretically break 2 out of 3 instances of Agrasta. Based on our analysis, some recommended parameters of Dasta and Rasta seem to be aggressive for their small security margins. Our cryptanalysis also implies that the last nonlinear layer in Rasta and Dasta cannot effectively increase the degree in a fast way as expected by the designers.

References

1. M. Albrecht and G. Bard. *The M4RI Library*. The M4RI Team, 2021. `http://m4ri.sagemath.org`.
2. M. R. Albrecht, C. Cid, L. Grassi, D. Khovratovich, R. Lüftenegger, C. Rechberger, and M. Schofnegger. Algebraic Cryptanalysis of STARK-Friendly Designs: Application to MARVELlous and MiMC. In S. D. Galbraith and S. Moriai, editors, *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 III*, volume 11923 of *Lecture Notes in Computer Science*, pages 371–397. Springer, 2019.
3. M. R. Albrecht, L. Grassi, L. Perrin, S. Ramacher, C. Rechberger, D. Rotaru, A. Roy, and M. Schofnegger. Feistel Structures for MPC, and More. In K. Sako, S. A. Schneider, and P. Y. A. Ryan, editors, *Computer Security - ESORICS 2019 - 24th European Symposium on Research in Computer Security, Luxembourg, September 23-27, 2019, Proceedings, Part II*, volume 11736 of *Lecture Notes in Computer Science*, pages 151–171. Springer, 2019.
4. M. R. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen. MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity. In J. H. Cheon and T. Takagi, editors, *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*, volume 10031 of *Lecture Notes in Computer Science*, pages 191–219, 2016.

5. M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 430–454. Springer, 2015.
6. J. Alman and V. V. Williams. A Refined Laser Method and Faster Matrix Multiplication. In D. Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 522–539. SIAM, 2021.
7. A. Aly, T. Ashur, E. Ben-Sasson, S. Dhooghe, and A. Szepieniec. Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. *IACR Trans. Symmetric Cryptol.*, 2020(3):1–45, 2020.
8. F. Armknecht, C. Carlet, P. Gaborit, S. Künzli, W. Meier, and O. Ruatta. Efficient Computation of Algebraic Immunity for Algebraic and Fast Algebraic Attacks. In S. Vaudenay, editor, *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*, volume 4004 of *Lecture Notes in Computer Science*, pages 147–164. Springer, 2006.
9. T. Ashur and S. Dhooghe. MARVELlous: a STARK-Friendly Family of Cryptographic Primitives. Cryptology ePrint Archive, Report 2018/1098, 2018. `https://eprint.iacr.org/2018/1098`.
10. G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche. Keccak. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 313–314. Springer, 2013.
11. T. Beyne, A. Canteaut, I. Dinur, M. Eichlseder, G. Leander, G. Leurent, M. Naya-Plasencia, L. Perrin, Y. Sasaki, Y. Todo, and F. Wiemer. Out of Oddity - New Cryptanalytic Techniques Against Symmetric Primitives Optimized for Integrity Proof Systems. In D. Micciancio and T. Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 299–328. Springer, 2020.
12. A. Biryukov and C. D. Cannière. Block Ciphers and Systems of Quadratic Equations. In T. Johansson, editor, *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, volume 2887 of *Lecture Notes in Computer Science*, pages 274–289. Springer, 2003.
13. A. Björklund, P. Kaski, and R. Williams. Solving Systems of Polynomial Equations over GF(2) by a Parity-Counting Self-Reduction. In C. Baier, I. Chatzigiannakis, P. Flocchini, and S. Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 26:1–26:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.
14. C. Bouillaguet, H. Chen, C. Cheng, T. Chou, R. Niederhagen, A. Shamir, and B. Yang. Fast Exhaustive Search for Polynomial Systems in F_2. In S. Mangard and F. Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 203–218. Springer, 2010.

15. A. Canteaut, S. Carpov, C. Fontaine, T. Lepoint, M. Naya-Plasencia, P. Paillier, and R. Sirdey. Stream Ciphers: A Practical Solution for Efficient Homomorphic-Ciphertext Compression. In T. Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, volume 9783 of *Lecture Notes in Computer Science*, pages 313–333. Springer, 2016.
16. N. T. Courtois. Fast Algebraic Attacks on Stream Ciphers with Linear Feedback. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 176–194. Springer, 2003.
17. N. T. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer, 2000.
18. N. T. Courtois and W. Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer, 2003.
19. N. T. Courtois and J. Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In Y. Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer, 2002.
20. I. Dinur. Cryptanalytic Applications of the Polynomial Method for Solving Multivariate Equation Systems over GF(2). Cryptology ePrint Archive, Report 2021/578, 2021. `https://eprint.iacr.org/2021/578`.
21. I. Dinur. Improved Algorithms for Solving Polynomial Systems over GF(2) by Multiple Parity-Counting. In D. Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2550–2564. SIAM, 2021.
22. I. Dinur, Y. Liu, W. Meier, and Q. Wang. Optimized Interpolation Attacks on LowMC. In T. Iwata and J. H. Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 535–560. Springer, 2015.
23. C. Dobraunig, M. Eichlseder, L. Grassi, V. Lallemand, G. Leander, E. List, F. Mendel, and C. Rechberger. Rasta: A Cipher with Low ANDdepth and Few ANDs per Bit. In H. Shacham and A. Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 662–692. Springer, 2018.
24. C. Dobraunig, M. Eichlseder, and F. Mendel. Higher-Order Cryptanalysis of LowMC. In S. Kwon and A. Yun, editors, *Information Security and Cryptology - ICISC 2015 - 18th International Conference, Seoul, South Korea, November 25-27,*

2015, Revised Selected Papers, volume 9558 of *Lecture Notes in Computer Science*, pages 87–101. Springer, 2015.
25. C. Dobraunig, L. Grassi, A. Guinet, and D. Kuijsters. Ciminion: Symmetric Encryption Based on Toffoli-Gates over Large Finite Fields. Cryptology ePrint Archive, Report 2021/267, 2021. `https://eprint.iacr.org/2021/267`.
26. C. Dobraunig, F. Moazami, C. Rechberger, and H. Soleimany. Framework for faster key search using related-key higher-order differential properties: applications to Agrasta. *IET Inf. Secur.*, 14(2):202–209, 2020.
27. S. Duval, V. Lallemand, and Y. Rotella. Cryptanalysis of the FLIP Family of Stream Ciphers. In M. Robshaw and J. Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 457–475. Springer, 2016.
28. M. Eichlseder, L. Grassi, R. Lüftenegger, M. Øygarden, C. Rechberger, M. Schofnegger, and Q. Wang. An Algebraic Attack on Ciphers with Low-Degree Round Functions: Application to Full MiMC. In S. Moriai and H. Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 477–506. Springer, 2020.
29. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1-3):61–88, June 1999.
30. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero F5. In *International Symposium on Symbolic and Algebraic Computation Symposium - ISSAC 2002*, pages 75–83, Villeneuve d'Ascq, France, July 2002. ACM. Colloque avec actes et comité de lecture. internationale.
31. S. Fischer and W. Meier. Algebraic Immunity of S-Boxes and Augmented Functions. In A. Biryukov, editor, *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, volume 4593 of *Lecture Notes in Computer Science*, pages 366–381. Springer, 2007.
32. L. Grassi, D. Kales, D. Khovratovich, A. Roy, C. Rechberger, and M. Schofnegger. Starkad and Poseidon: New Hash Functions for Zero Knowledge Proof Systems. *IACR Cryptol. ePrint Arch.*, 2019:458, 2019.
33. L. Grassi, R. Lüftenegger, C. Rechberger, D. Rotaru, and M. Schofnegger. On a Generalization of Substitution-Permutation Networks: The HADES Design Strategy. In A. Canteaut and Y. Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 674–704. Springer, 2020.
34. J. Guo, M. Liu, and L. Song. Linear Structures: Applications to Cryptanalysis of Round-Reduced Keccak. In J. H. Cheon and T. Takagi, editors, *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*, volume 10031 of *Lecture Notes in Computer Science*, pages 249–274, 2016.
35. P. Hebborn and G. Leander. Dasta - Alternative Linear Layer for Rasta. *IACR Trans. Symmetric Cryptol.*, 2020(3):46–86, 2020.
36. D. Kales and G. Zaverucha. Improving the Performance of the Picnic Signature Scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(4):154–188, 2020.

37. F. Liu, T. Isobe, and W. Meier. Cryptanalysis of Full LowMC and LowMC-M with Algebraic Techniques. Cryptology ePrint Archive, Report 2020/1034, 2020. https://eprint.iacr.org/2020/1034.
38. D. Lokshtanov, R. Paturi, S. Tamaki, R. R. Williams, and H. Yu. Beating Brute Force for Systems of Polynomial Equations over Finite Fields. In P. N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2190–2202. SIAM, 2017.
39. P. Méaux, A. Journault, F. Standaert, and C. Carlet. Towards Stream Ciphers for Efficient FHE with Low-Noise Ciphertexts. In M. Fischlin and J. Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 311–343. Springer, 2016.
40. C. Rechberger, H. Soleimany, and T. Tiessen. Cryptanalysis of Low-Data Instances of Full LowMCv2. *IACR Trans. Symmetric Cryptol.*, 2018(3):163–181, 2018.
41. V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.