IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2025, No. 1, pp. 357–379. DOI:10.46586/tosc.v2025.i1.357-379

A More Practical Attack Against Yoroi

Runhao Wei[1,4,5], Jinliang Wang[1,4,5(✉)], Haoyang Wang[2(✉)], Muzhou Li[1,4,5], Yunling Zhang[3] and Meiqin Wang[6,1,4,5]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, China
[2] Shanghai Jiao Tong University, Shanghai, China
[3] CHECC Data Co., Ltd., Beijing, China
[4] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, China
[5] State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Qingdao, 266237, China
[6] Quan Cheng Shandong Laboratory, Jinan, China
{runhaowei,jinliangwang,muzhouli}@mail.sdu.edu.cn, haoyang.wang@sjtu.edu.cn, 15901407461@163.com, mqwang@sdu.edu.cn

Abstract. Yoroi is a family of space-hard block cipher proposed at TCHES 2021. This cipher contains two parts, a *core* part and an AES layer to prevent the blackbox adversary. At FSE 2023, Todo and Isobe proposed a code-lifting attack to recover the secret T-box in Yoroi, breaking the security claims of Yoroi. Their work shows that the AES layer is vulnerable in the whitebox model and has no contribution to the security in a hybrid of blackbox and whitebox model. Besides, their attack employs a strong hack model to modify and extract the table entries of the T-box. This hack model is suitable for the environment used by Yoroi while it is difficult to achieve in the practical application.
In this paper, we present an attack on Yoroi within a more practical scenario. Compared with the previous attack, our attack is a chosen-plaintext-ciphertext attack in the blackbox phase and assumes that the whitebox attacker has reduced capabilities, as one only needs to extract the AES key without modifying or extracting the table entries. Furthermore, we introduce a family of equivalent representations of Yoroi, using this we can recover an equivalent cipher without any leaked information of table entries. As a result, the complexities of our attack remain almost the same as that of the previous attack.

Keywords: Whitebox cryptography · Space-hard · Cryptanalysis · Yoroi

1 Introduction

Whitebox cryptography aims to protect software implementations of cryptographic algorithms under an untrusted environment, where the attackers have full access to the implementations, and can perform both static and dynamic analysis, including execution tracing, fault injection, implementation modification, and more.

In 2002, Chow et al. [CEJvO02a, CEJvO02b] proposed the first whitebox implementations of AES and DES, the main idea is to convert parts of round operations into lookup tables embedded with round keys. Each such lookup table by itself does not leak any information about the secret key. After their pioneering works, many improved whitebox implementations were proposed [BCD06, LN05, XL09, Kar10]. However, all of them were broken by a structural attack (called BGE analysis) [BGE04] and its successors [MRP12, WMGP07, MWP10, LRM+13], where the secret keys are extracted. Recently, a generic attack called *differential computation analysis* (DCA) [BHMT16] can apply key

Received: 2024-11-22 Accepted: 2025-01-23 Published: 2025-03-07

extraction attack on most existing whitebox implementations under a graybox model, which can be regarded as a limited whitebox model where the attacker requires neither knowledge about the lookup tables nor any reverse engineering effort. There are many following works under such model, just list a few here [BBIJ17, BU18, BU21].

To resist key extraction attacks, another approach involves incorporating well-designed block ciphers, such as AES, to design dedicated whitebox block ciphers [BBK14]. This approach reduces key extraction attacks to key recovery attacks of the embedded block ciphers in the standard blackbox model. For such designs, the primary security goal has shifted towards countering *code lifting*, another type of whitebox attack that aims to isolate the program code in order to copy the functionality of encryption/decryption rather than extracting the secret key. To mitigate code lifting, Bogdanov and Isobe [BI15] proposed a security notion of space hardness, which is similar to incompressibility [DLPR13] and weak whitebox security [BBK14]. It evaluates the security against code lifting by assessing the size of program a whitebox attacker must extract to maintain the cipher's functionality. Block ciphers satisfying space hardness are called *space-hard block ciphers*, and many have been proposed [BI15, FKKM16, CCD+17, KLLM20, KSHI20].

Although space-hard block ciphers can provide a sufficient level of security against code lifting, they still have drawbacks facing on continuous data leakage. This could happen when an attacker steals small amounts of data from the program over a long period, which might not be monitored by the system. Eventually, the attacker can extract the entire program. To address this problem, a space-hard block cipher Yoroi with a new property called *longevity* was introduced at TCHES 2021[KI21]. It allows the users to update lookup tables of Yoroi while keeping the functionality of the encryption/decryption function. Once a certain period has passed or the amount of data leakage reaches a limit, the tables will be updated to different ones so that the attacker must restart the process of leaking table entries from the beginning.

At ToSC 2023, Todo and Isobe [TI22] analyzed Yoroi under a new attack model called *hybrid code lifting* attack. This model is a hybrid of blackbox and whitebox model. Firstly, in the whitebox model, the attacker hacks into the encryption program, analyze and modify the implementation, and leak arbitrary data bounded by a limited size. Secondly, an collaborative attacker analyzes the algorithm in the standard blackbox model with the help of the information received from the whitebox attacker. Todo and Isobe applied this model to Yoroi. Specifically, they modified the program, and leaked 800-bit and 3008-bit information in the whitebox model for Yoroi-16 and Yoroi-32, respectively. Then, they recovered a functionally equivalent implementation based on differential cryptanalysis in the blackbox model. Last but not least, they also broke the security claim about the longevity of Yoroi.

1.1 Our Contributions

In this paper, we introduce an equivalent cipher recovery attack against Yoroi. The previous work in [TI22], which highly relies on the information obtained from the hack-model, is usually difficult to achieve in practice. Our attack only requires a whitebox ability to extract the key of the AES layer in Yoroi. The remaining processes are fully operated under the blackbox model. Compared with the previous work, we significantly weaken the capabilities of the whitebox attacker while introducing only a small amount of additional complexity. We consider our attack is more suitable in the practical environment of Yoroi and should be given priority consideration when this family of cipher is used. Detailed contributions are shown as follows.

Equivalent Representation of Yoroi. We analyze the linear layer of Yoroi and find that we can add a permutation before the linear layer and a corresponding permutation after

the linear layer to keep the function of the linear layer unchanged. This property enables us to reduce the computation complexity during our attack process by approximately $2^{7.9}$.

Break the Security Claim of Yoroi with Less Leakage. In the attack model by Todo and Isobe, they assume an attacker hacks into the encryption program who can freely read the table entries and modify the encryption program. Considering that such capabilities of an attacker would be excessively powerful and difficult to implement, we present a new attack based on the attack model of Todo et al., where the whitebox attacker does not transmit any information about the table entries, nor does she modify the original program, but only transmits the key of AES layer. The collaborative blackbox attacker can recover an equivalent algorithm faster than exhaustive search attacks. Our results on Yoroi-16 and Yoroi-32 are shown in Table 1.

Table 1: Comparison of attacks on Yoroi. CP and CPC separately denote that the data is collected in the chosen-plaintext and chosen-plaintext-ciphertext settings.

Target	Code-lifting phase		Blackbox phase		Reference
	time	leaked data (bits)	time	data	
Yoroi-16	$2^{18.8}$	800	$2^{33.09}$	$2^{33.09}$CP	[TI22]
	-	128	$2^{34.22}$	$2^{33.10}$CPC	Section 5
	-	128	$2^{69.88}$	$2^{69.88}$CP	Appendix C
Yoroi-32	$2^{35.9}$	3008	$2^{65.44}$	$2^{65.44}$CP	[TI22]
	-	128	$2^{65.45}$	$2^{65.45}$CPC	Section 5
	-	128	$2^{88.76}$	$2^{88.76}$CP	Appendix C

1.2 Organizations

In Section 2, we introduce the block cipher Yoroi. Section 3 introduces the hybrid code lifting attack on Yoroi. A new set of equivalent representations is given in Section 4. Section 5 introduces our hybrid attacks on Yoroi, utilizing only the leakage of the AES key. The experiments that verify the correctness of the theories used in our attacks are presented in Section 6. Besides, all the codes used in the verified experiments are available at the following repository: https://github.com/attackYoroi/attackyoroi.

2 Space-Hard Block Cipher: Yoroi

2.1 Space-Hard Block Cipher

2.1.1 Code Lifting

The security requirements of whitebox cryptography can be classified into key extraction security and code lifting security. The code lifting considers the attacker extracts the program code to copy the functionality of encryption/decryption instead of the secret key.

Code lifting attacks are mostly defined under the whitebox model, but it can also be extended to the blackbox model. We generalize the definition as follows. In some aspects, it is also referred to as global deduction by Lars Knudsen [KR11].

Definition 1 (Function extraction)**.** A function extraction attack is an attacker's attempt to recover the original or a functionally equivalent scheme of the targeted encryption program.

In this paper, we refer to code lifting as a type of function extraction attack that involves accessing and modifying the program in the whitebox model.

2.1.2 Space Hardness

Bogdanov et al. introduced the notion of space hardness in [BIT16] to mitigate code lifting attack. It evaluates the difficulty of compressing a cipher's whitebox implementation and quantifies the security against code lifting by measuring the amount of code a whitebox attacker must extract to preserve the functionality of the cipher.

Definition 2. ((M,Z)-space hardness[BI15]). A block cipher is (M,Z)-space-hard means attackers can not correctly encrypt (decrypt) a random message with probability greater than 2^{-Z} if the size of code they get is less than M.

Even if a part of the program code is leaked, the block cipher remains secure when the space hardness is guaranteed. We call such a cipher space-hard block cipher. For example, for a ($size(T)/4$, 128)-space-hard block cipher, the attacker can not encrypt (decrypt) any plaintext (ciphertext) correctly with probability greater than 2^{-128} if the size of program code leaked is less than $size(T)/4$, where $size(T)$ denotes the program size.

Bogdanov et al. [BIT16] introduced three adversary models of space hardness. In this paper we only focus on the known-space attack, which is related to our attack.

Definition 3 (Known-Space Attack (KSA))**.** The adversary extracts M pairs of inputs and the corresponding outputs of tables $(x_i, F(x_i))$ for $i \in \{1, 2, \cdots, M\}$, where x_i cannot be chosen by the adversary.

2.2 Yoroi: Updatable Block Cipher

Yoroi is an updatable space-hard block cipher that was introduced at TCHES 2021 [KI21]. The tables in the whitebox implementation of Yoroi can be updated while preserving the same functionality of the block cipher.

2.2.1 Notations and Specification

Yoroi is an SPN cipher with n-bit block and k-bit secret key. The internal state X^r in the r-th round consists of ℓ elements of n_{in} bits $X^r = \{x_0^r, \cdots, x_{\ell-1}^r\}$, where $r \in \{1, \cdots, R\}$, $x_i \in \mathbb{F}_2^{n_{in}}$ and $n_{in} \times \ell = n$. Each element x_i is divided into the top m bits $x_i^L = msb_m(x_i)$ and the last t bits $x_i^R = lsb_t(x_i)$ such that $x_i = x_i^L || x_i^R$, where $m + t = n_{in}$ and msb_m (lsb_m) denotes the most significant (least significant) m bits of the element.

Let $P \in (\mathbb{F}_2^{n_{in}})^\ell$ be a plaintext and the corresponding ciphertext $C \in (\mathbb{F}_2^{n_{in}})^\ell$ is computed as $C = \mathcal{A} \circ \gamma^R \circ (\bigcirc_{i=1}^{R-1}(\theta \circ \sigma^i \circ \gamma^i))(P)$. $\mathcal{A}$ is a full-round AES. We call $\gamma^R \circ (\bigcirc_{i=1}^{R-1}(\theta \circ \sigma^i \circ \gamma^i))$ the Yoroi-core part, consisting of R rounds. Each round is defined as follows.

S-Layer γ^r. The S-layer $\gamma^r : (\mathbb{F}_2^{n_{in}})^\ell \to (\mathbb{F}_2^{n_{in}})^\ell$ consists of ℓ key-dependent n_{in}-bit bijective functions. S_1 and S_3 are used for the first and last rounds, respectively, while S_2 is used for the other rounds:

$$\gamma^r : (x_0, \cdots, x_{\ell-1}) \to (S_j(x_0), \cdots, S_j(x_{\ell-1})),$$

where $j = 1$ for $r = 1$, $j = 3$ for $r = R$, and $j = 2$ for the rest r.

Linear Layer θ. The linear layer θ: $(\mathbb{F}_2^{n_{in}})^\ell \to (\mathbb{F}_2^{n_{in}})^\ell$ consists of an $\ell \times \ell$ MDS matrix M_t over $\mathbb{F}_2^t$, which is applied to the part of the state. Specifically, M_t takes the least significant t bits of each n_{in}-bit element as inputs.

$$\theta : (x_0, \cdots, x_{\ell-1}) \to (x_0^L || x_0'^R, \cdots, x_{\ell-1}^L || x_{\ell-1}'^R),$$

where $(x_0'^R, \cdots, x_{\ell-1}'^R)^T = M_t \cdot (x_0^R, \cdots, x_{\ell-1}^R)^T$.

Affine Layer σ^r. In the affine layer σ^r: $(\mathbb{F}_2^{n_{in}})^\ell \to (\mathbb{F}_2^{n_{in}})^\ell$, t-bit constants $C_i (0 \leq C_i \leq 2^t - 1)$ are added to each element of the state in r-th round.

$$\sigma^r : (x_0, \cdots, x_{\ell-1}) \to (x_0 \oplus C_r, \cdots, x_{\ell-1} \oplus C_r),$$

where $C_r = r$.

AES Layer $\mathcal{A}$. Finally, a full-round AES $\mathcal{A}$ with a fixed key $K_\mathcal{A}$ is applied. $K_\mathcal{A}$ is generated independently of the keys used in S-layers.

2.2.2 Whitebox Implementation and Longevity

The whitebox implementation of Yoroi has an additional m-bit block cipher E, which is used to update the bijective functions S_1, S_2 and S_3:

$$T_1 = (E||I) \circ S_1, \qquad T_2 = (E||I) \circ S_2 \circ (E^{-1}||I), \qquad T_3 = S_1 \circ (E^{-1}||I),$$

E is applied to x^L, i.e., $(E||I)(x) = E(x^L)||x^R$. The whitebox implementation of Yoroi keeps the same functionality since the influence of E and E^{-1} can be neutralized between the bijective functions. This feature that maintaining the functionality even the tables are updated is called *longevity*.

The intention behind this design is to ensure the security even if partial table entries are leaked. By updating T_1, T_2 and T_3, the information leaked to attackers will be changed, and thus, massive leakage is accepted.

2.2.3 Parameters and Claimed Security

Yoroi has two versions: Yoroi-16 and Yoroi-32 adopt key-dependent 16- and 32-bit tables (bijective functions), respectively. Both of them are 128-bit block ciphers.

- Yoroi-16: $n = 128$, $\ell = 8$, $n_{in} = 16$, $m = 12$, $t = 4$, $R = 8$, $S_j(x) : \{0,1\}^{16} \to \{0,1\}^{16}$,
- Yoroi-32: $n = 128$, $\ell = 4$, $n_{in} = 32$, $m = 28$, $t = 4$, $R = 16$, $S_j(x) : \{0,1\}^{32} \to \{0,1\}^{32}$.

The following are the designer's security claims, where the space hardness is mainly focused on KSA model.

- Resist all attacks in the blackbox model.
- Secure against key extraction in the whitebox model.
- $((3 \times 2^{n_{in}})/4, 128)$-space hardness against KSA.
- $((3 \times 2^{n_{in}})/64, 128)$-space hardness against KSA every table update.

For example, Yoroi-16 ensures that an attacker can not correctly encrypt (decrypt) a random message with probability higher than 2^{-128} given less than 3×2^{14} table entries. For longevity, the attack is still impossible even if 3×2^{10} table entries are leaked for every updated table.

3 Hybrid Code Lifting and Application to Yoroi

In this section, we introduce the concept of hybrid code lifting and present the corresponding attacks on Yoroi by Todo and Isobe.

3.1 Hybrid Code Lifting

The blackbox model assumes that the attacker can access the inputs and the outputs of a cipher. In contrast, the whitebox model assumes that the attacker has full control over the execution environment of a cipher, allowing him to access and modify the implementation. In practice, these two models can be combined to analyze a whitebox block cipher.

When the concept of space hardness was first introduced, it did not assume a blackbox attacker receiving the leakage. However, Todo and Isobe argue that considering a blackbox attacker is necessary to fulfill the intent of the secure leakage-resilient system and emphasize the necessity of hybrid code lifting in their paper [TI22]:

> "If we do not need to consider a blackbox attacker after the leakage, the use case of the space-hard block cipher is limited."

They propose a hybrid model of blackbox and whitebox for code lifting attack [TI22], named *hybrid code lifting*, which breaks the security claim of Yoroi.

Definition 4 (Hybrid code lifting)**.** The attack model consists of two phases: the first phase being the code lifting phase and the second phase being the blackbox phase. In the first phase, the whitebox attacker hacks into the encryption program, and can read and modify the whole table entries. The amount of leakage and the time complexity of this phase are bounded. In the second phase, the blackbox attacker receives the leakage in the first phase and then analyzes the encryption by making queries in the blackbox model. Similarly, the data and time complexities are bounded.

3.2 Hybrid Code Lifting on Yoroi by Todo and Isobe

At FSE 2023, Todo and Isobe [TI22] proposed function extraction attacks on Yoroi under the hybrid code lifting model. They first extract table entries in the whitebox model, and then use the additional information to conduct function extraction in the blackbox model.

They introduced a canonical representation of Yoroi, which applies additional m-bit permutations to the three tables T_1, T_2 and T_3. The representation is shown in Figure 1. Specifically, the tables used in each round are updated as follows:

$$\begin{aligned}\tilde{T}_1 =&(E_1||I) \circ T_1,\\ \tilde{T}_r =&(E_r||I) \circ T_2 \circ (D_{r-1}||I), \text{ for } r \in \{2,3,...,R-1\},\\ \tilde{T}_R =&T_3 \circ (D_{R-1}||I).\end{aligned}$$

Note that $D_r = E_r^{-1}$, thus the canonical representation maintains its functionality no matter what E_r is selected.

They exploited the freedom of E_r to construct tables $\tilde{T}_r$ for $r \in \{1, \cdots, R-1\}$ that satisfy Property 1, and proved that such tables are uniquely determined by S_1, S_2 and S_3. We omit the searching procedure and proof here and refer readers to their original paper for details.

Property 1. When $lsb_t(\tilde{T}_r(y_1)) = lsb_t(\tilde{T}_r(y_2)) = 0, msb_m(\tilde{T}_r(y_1)) < msb_m(\tilde{T}_r(y_2))$ holds for all $y_1 < y_2$.

The attack consists of two phases. We firstly clarify some notations based on canonical representation.

- $\rho \in (\mathbb{F}_2^m)^{2^t}$: 2^t-dimensional vector whose elements take a value over $\mathbb{F}_2^m$.
- $\mathbb{A}_i := \{x \in \mathbb{F}_2^{n_{in}} | lsb_t(\tilde{T}_r(x)) = i\}$. We use $\mathbb{A}_{\rho_i}$ when ρ has not been recovered yet.
- $\eta \in (\mathbb{F}_2^t)^{2^m}$: 2^m-dimensional vector whose elements take a value over $\mathbb{F}_2^t$.

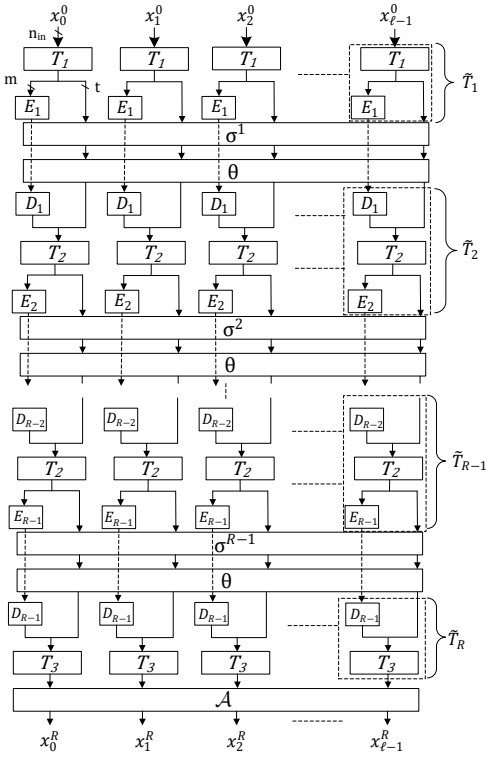

Figure 1: Canonical representation of Yoroi

- $\mathbb{B}_j := \{x \in \mathbb{F}_2^{n_{in}} | msb_m(\tilde{T}_r(x)) = j\}$. We use $\mathbb{B}_{\eta_j}$ when η has not been recovered yet.
- $\mathrm{x}_{j,i} \in \mathbb{F}_2^{n_{in}}$: input of $\tilde{T}_r(x)$ such that $\tilde{T}_r(x_{j,i}) = j||i$.

Since $\tilde{T}_r$ is a permutation, there are 2^t distinct $\mathbb{A}_i$ of size 2^m, and similarly 2^m different $\mathbb{B}_j$ of size 2^t.

3.2.1 The Code Lifting Phase

In this phase, the whitebox attacker extracts information of the program and leaks it to the blackbox attacker in the second phase.

The whitebox attacker first extracts the AES key, which is assumed to be an easy task since the designer of Yoroi employs AES only for blackbox security. Then, the attacker modifies the tables in the program to generate $\tilde{T}_r$ for $r \in \{1, \cdots, R\}$ of the canonical representation. Finally, the attacker leaks all elements in $\mathbb{B}_0$ of $\tilde{T}_r$ for $r \in \{1, 2, \cdots, R-1\}$.

In summary, the leakage size is $128 + (R-1) \times 2^t \times n_{in}$ bits, which are 1920 and 7808 bits for Yoroi-16 and Yoroi-32, respectively. These leakage amounts are significantly smaller than the safety bounds claimed by the designers of Yoroi.

3.2.2 The Blackbox Phase

The blackbox attacker receives the leakage from the whitebox attacker, and launches differential attack in this phase.

Step 1. In this step, $\mathbb{A}_i$ is recovered. We first show a truncated differential of the Yoroi-core part in Figure 2, which is constructed by the fact that the linear layer is only applied

to the last t bits of the output of $\tilde{T}_r$. The attacker uses this differential distinguisher to

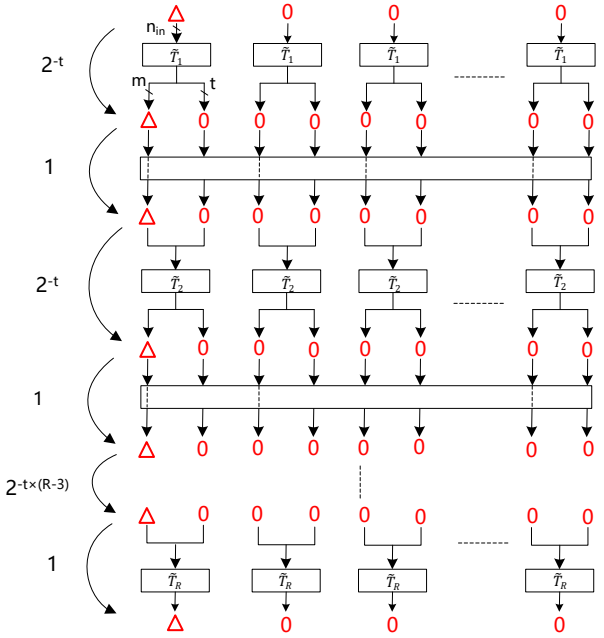

Figure 2: Truncated differential trail of Yoroi-core part. Δ represent non-zero difference.

categorize the input space of $\tilde{T}_1$ based on whether the lower t bits of the output from $\tilde{T}_1$ are the same. The procedure is as follows.

1. Prepare a structure of $2^{n_{in}}$ plaintexts by taking all values in the first element and constant for the others.

2. Store all the corresponding ciphetexts in a hash table indexed by the $lsb_{n_{in}(l-1)}$ bits. Collect the corresponding plaintext pair (P_1, P_2) of each ciphertext collision.

The probability of the truncated differential is $2^{-t(R-1)}$, thus each collected pair (P_1, P_2) very likely conform to the truncated differential, which means $lsb_t(\tilde{T}_1(P_1[0]) \oplus \tilde{T}_1(P_2[0])) = 0$, i.e., $P_1[0]$ and $P_2[0]$ belong to the same subset $\mathbb{A}$.

Continuously repeat the above procedure, merging collections where the intersection of the collected sets is non-empty. Each procedure requires $2^{n_{in}}$ times, data, and memory complexities. Todo et al. utilized graph connectivity [ER59] for analysis, concluding that the procedure has to be repeated $2^{15.52}$ and $2^{32.45}$ times for Yoroi-16 and Yoroi-32, respectively, to divide $\mathbb{F}_2^{n_{in}}$ into $\mathbb{A}_{\rho_i}$ with a success probability of 50%.

After that, they determined the specific values of ρ_i using the information of $\mathbb{B}_0$ that was leaked by the whitebox attacker. Specifically, the attacker checks whether the subset $\mathbb{A}_{\rho_i}$ includes $x_{0,j}$, where $x_{0,j} \in \mathbb{B}_0$. If so, they have $\rho_i = j$.

Step 2. In this step, $\mathbb{B}_{\eta_j}$ is recovered. We do not need to know η as it can be complemented from $\mathbb{A}_0$. We use another truncated differential shown in Figure 3, where the differences $\{\zeta_1, \cdots, \zeta_\ell\}$ are uniquely determined by β according to the MDS matrix M_t, where β is chosen by the attacker. Based on the knowledge of $\mathbb{A}_i$ from step 1 and $\mathbb{B}_0$ from the leakage, we are able to prepare a plaintext pair (P_1, P_2) that conforms to the first round differential in Figure 3 with probability 2^{-m}. The attack procedure is as follows.

1. Compute $(\zeta_1, \zeta_2, ..., \zeta_\ell)$ from a non-zero $\beta \in \mathbb{F}_2^t$.

2. Construct a set of 2^{2m} plaintexts. Each plaintext $P_1 = (P_1[1], P_1[2], \cdots, P_1[\ell])$ has the following form: $P_1[1]$ iterates through the elements in $\mathbb{A}_{i_1}$, where i_1 denotes a random element over $\mathbb{F}_2^t$. $P_1[2]$ iterates through the elements in $\mathbb{A}_0$. For the rest $P_1[l]$, the attacker randomly chooses the element x_{0,i_l} from $\mathbb{B}_0$. Obtain the corresponding ciphertexts by using the encryption oracle.

3. Construct another set of 2^{2m} plaintexts. Each plaintext $P_2 = (P_2[1], P_2[2], \cdots, P_2[\ell])$ has the following form: $P_2[1]$ iterates through the elements in $\mathbb{A}_{i_1 \oplus \zeta_1}$, $P_2[2]$ iterates through the elements in $\mathbb{A}_{\zeta_2}$. For the rest $P_2[l]$, the attacker chooses the element $x_{0,i_l \oplus \zeta_l}$ from $\mathbb{B}_0$ such that $\tilde{T}_1(x_{0,i_l}) \oplus \tilde{T}_1(P_2[l]) = (0||\zeta_l)$. Obtain the corresponding ciphertexts by using the encryption oracle.

4. Collect pairs $(P_1[2], P_2[2])$ satisfying $lsb_{n_{in}(\ell-1)}(C_1 \oplus C_2) = 0$, where C_1 and C_2 denote the ciphertexts of P_1 and P_2, respectively.

Continuously iterate the above procedure, with the time, data, and memory complexity required for each iteration being 2^{2m+1}. In the second branch, there are $(2^t - 1) \times 2^m \approx 2^{n_{in}}$ different pairs $(P_1[2], P_2[2])$ satisfying $msb_m(\tilde{T}_1(P_1[2])) = msb_m(\tilde{T}_1(P_2[2]))$ that need to be detected. To reduce the probability of each pair not being detected to $2^{-n_{in}}$, Todo and Isobe estimate that 12 and 23 iterations are required for Yoroi-16 and Yoroi-32, respectively. Finally, in Yoroi-16, the total complexity is $2^{32.49}$, and in Yoroi-32, the total complexity is $2^{64.43}$.

At this point, the attacker has obtained $\mathbb{A}_i$ for all $i \in \mathbb{F}_2^t$ and $\mathbb{B}_{\eta_j}$ for all $\eta_j \in \mathbb{F}_2^m$. Since $\tilde{T}_1$ satisfies Property 1, η_j can be recovered, and thus $\tilde{T}_1$ is recovered. The attacker can bypass the first round, and use the same procedure to recover $\tilde{T}_2, \cdots, \tilde{T}_{R-1}$. Furthermore, if the attacker has already recovered $\tilde{T}_1, \cdots, \tilde{T}_{R-1}$, recovering $\tilde{T}_R$ is straightforward.

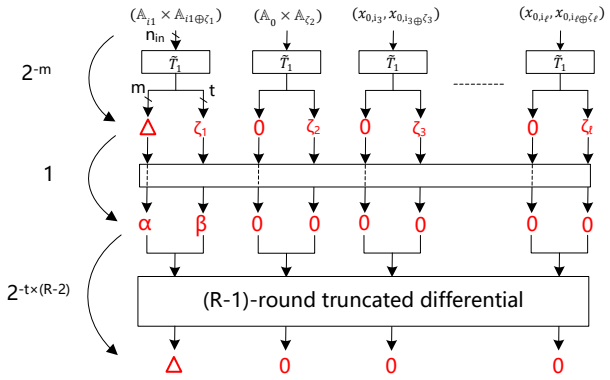

Figure 3: The truncated differential trail of Yoroi-core part used to recover $\mathbb{B}_{\eta_j}$.

4 Equivalent Representations of Yoroi

In this section, we introduce a set of equivalent representations of Yoroi.

Definition 5 (Equivalent mappings for a matrix)**.** We define that two mappings $F_I, F_O : (\mathbb{F}_2^n)^p \mapsto (\mathbb{F}_2^n)^p$ are equivalent for a $p \times p$ matrix M over $\mathbb{F}_2^n$, if

$$M \cdot \mathbf{x} = F_O \circ M \cdot F_I(\mathbf{x}), \forall \mathbf{x} \in (\mathbb{F}_2^n)^p.$$

Lemma 1. *For any affine mapping $A^I : (\mathbb{F}_2^n)^p \mapsto (\mathbb{F}_2^n)^p$, where $A^I(\mathbf{x}) = a \cdot \mathbf{x} + \tilde{\mathbf{b}}$, $\tilde{\mathbf{b}} = (b, b, \cdots, b)$ for $b \in \mathbb{F}_2^n$, we can always find an equivalent affine mapping $A^O : (\mathbb{F}_2^n)^p \mapsto (\mathbb{F}_2^n)^p$, where $A^O(\mathbf{x}) = c \cdot \mathbf{x} + \tilde{\mathbf{d}}$, $\tilde{\mathbf{d}} = (d, d, \cdots, d)$ for $d \in \mathbb{F}_2^n$, for a $p \times p$ cyclic matrix M over $\mathbb{F}_2^n$.*

Proof. Since matrix M is over a finite field, it satisfies commutativity with scalars. Therefore, A^I and A^O being equivalent for M means

$$\begin{aligned} M \cdot \mathbf{x} &= A^O \circ M \cdot A^I(\mathbf{x}) \\ &= A^O(Ma \cdot \mathbf{x} + M \cdot \tilde{\mathbf{b}}) \\ &= A^O(aM \cdot \mathbf{x} + M \cdot \tilde{\mathbf{b}}), \end{aligned}$$

we can easily derive that $A^O(\mathbf{x}) = a^{-1}(\mathbf{x} + M \cdot \tilde{\mathbf{b}}) = a^{-1}\mathbf{x} + a^{-1}M \cdot \tilde{\mathbf{b}}$, thus obtain $c = a^{-1}$ and $\tilde{\mathbf{d}} = a^{-1}M \cdot \mathbf{b}$. Since the matrix M is a cyclic matrix, each element $\tilde{\mathbf{d}}[i] = a^{-1}b\sum_{j=0}^{p-1} M_{i,j}$ is a constant value, which means $d = a^{-1}b\sum_{j=0}^{p-1} M_{i,j}$. □

Recall that the linear layer θ of Yoroi applies an $\ell \times \ell$ MDS matrix to a part of the internal state in each round: $(x_0'^R, \cdots, x_{\ell-1}'^R)^T = M_t \cdot (x_0^R, \cdots, x_{\ell-1}^R)^T$, we can add some specific affine mappings before and after the matrix multiplication, while the output remains the same as before according to Lemma 1.

Lemma 2. *For the t-bit constant addition in the affine layer and the multiplication of the matrix M_t in the linear layer of Yoroi, we can define a set of affine mappings $\{A_i^I, A_i^O\} : (\mathbb{F}_2^t)^\ell \mapsto (\mathbb{F}_2^t)^\ell$, such that $M_t \cdot (\mathbf{c} + \mathbf{x}) = A^O \circ M_t \cdot (\mathbf{c} + A^I(\mathbf{x}))$, and $A_i^I = L_i^I || L_i^I \cdots || L_i^I$ and $A_i^O = L_i^O || L_i^O \cdots || L_i^O$, where L_i^I and L_i^O are affine mappings : $\mathbb{F}_2^t \mapsto \mathbb{F}_2^t$.*

Proof. Since the affine layer adds the same t-bit constant in the lsb_t bits of each element of the state, that is, $\mathbf{c} = \{c, c, \cdots, c\} \in (\mathbb{F}_2^t)^\ell$. It can be regarded as a part of the A_i^I such that $A_i^I = a\mathbf{x} + \tilde{\mathbf{b}'}$ where $\tilde{\mathbf{b}'} = \tilde{\mathbf{b}} + \mathbf{c}$. The rest proof is similar to Lemma 1. □

Based on Lemma 2, we propose an equivalent representation of Yoroi below, and the representation is shown in Figure 4.

Theorem 1. *We can construct an equivalent representation of Yoroi by updating the T-box in each round as follows:*

$$\begin{cases} \hat{T}_1 = (E_1 || L_1^I) \circ T_1, & r = 1, \\ \hat{T}_r = (E_r || L_r^I) \circ T_2 \circ (E_{r-1}^{-1} || L_{r-1}^O), & 2 \le r \le R-1, \\ \hat{T}_n = T_3 \circ (E_{n-1}^{-1} || L_{n-1}^O), & r = R, \end{cases}$$

where E_i is an m-bit arbitrary permutation, L_i^I and L_i^O are chosen according to Lemma 2.

Proof. When two boxes $\hat{T}_i$ and $\hat{T}_{i+1}$ meet in the encryption process, the application of the layer $(E_i || L_i^I)$ after $\hat{T}_i$ can be canceled out by applying $(E_i^{-1} || L_i^O)$ before $\hat{T}_{i+1}$. This can be easily derived for the left part. For the right part, L_i^I and L_i^O are determined by Lemma 2, thus the applications of L_i^I and L_i^O will also be canceled out. Therefore, the new representation will keep the input of each T-box inside the $\hat{T}$-box same as that of the T-box in the original Yoroi, thus it has the same functionality as Yoroi.

□

We introduce the following notations in our attacks in the first round for better explanation[1]:

[1]The other rounds can use similar notations when the previous rounds are broken, so that the added layer before the T-box can be removed.

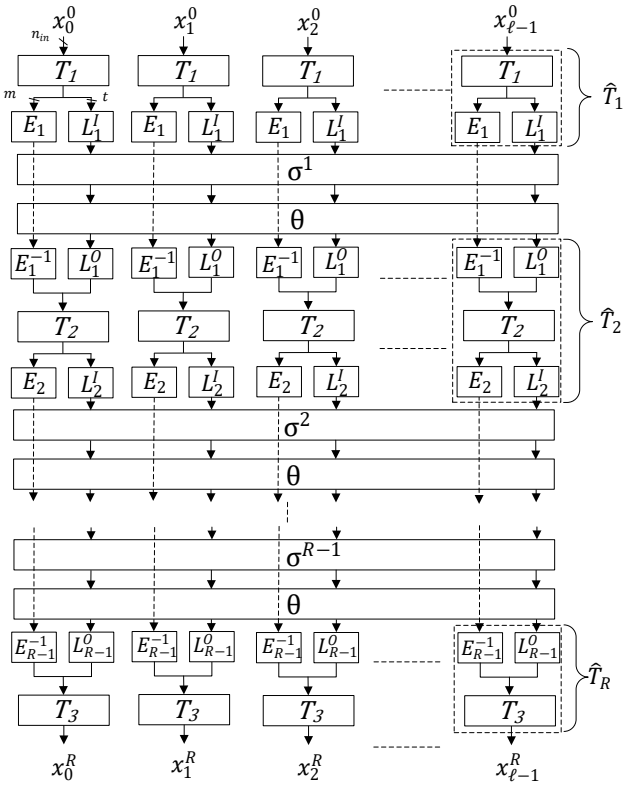

Figure 4: An equivalent representation of Yoroi.

- $\mathcal{L} := \{L(x) = ax + b \text{ for } x \in \mathbb{F}_2^t \mid a, b \in \mathbb{F}_2^t \text{ and } a \neq 0\}$. The size of the set is $2^t(2^t - 1)$.
- $F_{\mathbb{A}} : \mathbb{F}_2^{n_{in}} \mapsto \mathbb{F}_2^t$. $F_{\mathbb{A}}(x) = \rho_i$, when $x \in \mathbb{A}_{\rho_i}$.
- $\hat{F}_{\mathbb{A}} : \mathbb{F}_2^{n_{in}} \mapsto \mathbb{F}_2^t$. $\hat{F}_{\mathbb{A}}(x) = L_1^I \circ F_{\mathbb{A}}(x) = L_1^I(\rho_i)$, when $x \in \mathbb{A}_{\rho_i}$.
- $F_{\mathbb{B}} : \mathbb{F}_2^{n_{in}} \mapsto \mathbb{F}_2^m$. $F_{\mathbb{B}}(x) = \eta_j$, when $x \in \mathbb{B}_{\eta_j}$.
- $\hat{F}_{\mathbb{B}} : \mathbb{F}_2^{n_{in}} \mapsto \mathbb{F}_2^m$. $\hat{F}_{\mathbb{B}}(x) = E_1(\eta_j)$, when $x \in \mathbb{B}_{\eta_j}$.

Recall the previous attack against Yoroi in Section 3.2. When $\mathbb{F}_2^{n_{in}}$ is divided into the subsets $\mathbb{A}_{\rho_i}$and $\mathbb{B}_{\eta_j}$ separately, the attacker has to determine the values of ρ_i and η_j based on the knowledge of the leakage. Since the canonical representation is unique, the attacker needs to recover the unique $F_{\mathbb{A}}(x)$ and $F_{\mathbb{B}}(x)$. However, it is not the case for our attack as there are many equivalent representations of Yoroi.

Corollary 1. *For the equivalent representation of Yoroi, there are* $2^t(2^t - 1)$ *choices of* $\hat{F}_{\mathbb{A}}(x)$.

Proof. Note that $\hat{F}_{\mathbb{A}}(x) = L_r^I \circ F_{\mathbb{A}}(x)$. Since there is only one correct function $F_{\mathbb{A}}(x)$ and $L_r^I \in \mathcal{L}$, we can derive $2^t(2^t - 1)$ choices of $\hat{F}_{\mathbb{A}}(x)$. □

Corollary 2. *For the equivalent representation of Yoroi, the* $\hat{F}_{\mathbb{B}}(x)$ *has* $2^m!$ *choices.*

Proof. Since E_r is a random m-bit permutation, and note that the function $msb_m(T_r(x))$ is unique and $\hat{F}_{\mathbb{B}}(x) = E_r \circ msb_m(T_r(x))$, we have $2^m!$ choices of $\hat{F}_{\mathbb{B}}(x)$. □

Once a correct $\hat{F}_{\mathbb{A}}(x)$ and $\hat{F}_{\mathbb{B}}(x)$ are recovered, we can recover the corresponding $\hat{T}$-box. Our attack in the following section is a function extraction attack to recover a set of $\hat{T}_i$-boxes $\{\hat{T}_1, \cdots, \hat{T}_R\}$.

5 Attacks Against Yoroi

In this section, we provide a hybrid code lifting attack on Yoroi, which requires less leakage from the white-box attacker who has limited capabilities.

5.1 The Code Lifting Phase

In this phase, the whitebox attacker only has to extract the AES key and transmit it to the blackbox attacker. The leakage size is only 128 bits, and the time complexity can be omitted.

As comparison, the pioneering hybrid code lifting attack [TI22] proposed by Todo and Isobe assumes that the whitebox attacker can freely read the table entries of Yoroi and perform arbitrary computations on it. Besides, the leakage size of their attacks is 800 and 3008 bits for Yoroi-16 and Yoroi-32, respectively.

5.2 The Blackbox Phase

The remaining process of the attacks is conducted under the blackbox model, targeting Yoroi without the last AES layer. The goal is to recover one of the equivalent representations of Yoroi.

5.2.1 Process of Recovering $\hat{F}_{\mathbb{A}}$

Step 1: Divide the Subspace. The first step is to divide the input space of T_1 into 2^t subsets $\{\mathbb{A}_{\rho_0}, \cdots, \mathbb{A}_{\rho_{2^t-1}}\}$ by the same way as explained in Step 1 in Section 3.2.2. However, without the leaked information of $\mathbb{B}_0$, the value of ρ_i can not be recovered as before.

As demonstrated in Corollary 1, there are $2^t(2^t-1)$ choices of $\hat{F}_{\mathbb{A}}(x)$, which means the sequence of indexes $(\rho_0, \cdots, \rho_{2^t-1})$ has $2^t(2^t-1)$ choices. Each of them can be obtained by applying an affine mapping, which is also a permutation, to another one. Our strategy is first assigning a random value to the index sequence, say $\rho_i = i$, and then exhausting the t-bit permutation π such that the sequence turns to $(\pi(\rho_0), \cdots, \pi(\rho_{2^t-1}))$ after the permutation. The goal is to find one permutation such that the resulting sequence is one of the $2^t(2^t-1)$ candidates.

In the following, we show that the search space of the t-bit permutation can be reduced to $(2^t-2)!$ in our attack.

Definition 6. Two t-bit permutations π_1 and π_2 are affine equivalent if there exists an affine mapping $L \in \mathcal{L}$ satisfying $\pi_1 = L \circ \pi_2$. Those permutations that are mutually affine equivalent form an affine equivalence class.

The size of each affine equivalence class is $2^t(2^t-1)$ due to the size of $\mathcal{L}$. Hence, there are $2^t!/(2^t(2^t-1)) = (2^t-2)!$ affine equivalence classes for t-bit permutations. Particularly, we observe that there is a representative for each affine equivalence class. This observation is crucial as it allows us to reduce the complexity of the attack by focusing on a smaller set of representatives rather than considering all possibilities within each class.

Observation 1. *Each permutation π such that $\pi(0) = 0$ and $\pi(1) = 1$ is a representative of an affine equivalence class.*

The proof is simple. Assume that two different such permutations π_1 and π_2 belong to the same affine equivalence class, then there is an affine mapping $L(x) = ax + b$ such that $\pi_1 = L \circ \pi_2$. From

$$\pi_1(0) = a\pi_2(0) + b = 0,$$
$$\pi_1(1) = a\pi_2(1) + b = 1,$$

we obtain $a = 1$ and $b = 0$, which means π_1 and π_2 are identical, contradicting the assumption.

Complexity. The complexity of this step matches that of the step described in Section 3.2.2.

Step 2: Recover the Correct Permutation. In this step, one of the correct permutations is recovered. Note that we have assigned $\rho_i = i$ in the first step. We use the truncated differential trail in Figure 5, where β is a non-zero difference. We first make queries expecting some ciphertext pairs satisfying the differential trail from bottom to up, then identify such pairs using so-called *extended plaintexts*, finally recover the permutation. The details are as follows.

1. Prepare a ciphertext structure of size $2^s (s \leq n_{in})$ that only activates the most significant n_{in} bits, out of which 2^{2s-1} pairs can be constructed. Ask for the corresponding plaintexts. We refer to these texts as *original plaintexts/ciphertexts*.

2. For each original plaintext, construct its set of extended plaintexts: change the most significant n_{in} bits, ensuring the least significant t bits of the output of the first $\hat{T}$-box unchanged by checking the subsets of $\mathbb{A}_{\rho_i}$. The size of the extended set is 2^m.

3. For all the $2^m \times 2^s$ plaintexts, ask for their ciphertexts. We call it a *correct pair* if it meets the conditions: its two ciphertexts collide in the least significant $((\ell - 1) \times n_{in})$ bits and at least one of the plaintexts is an extended one.

If a ciphertext pair conforms to the differential trail, the plaintext pairs that are built from the two corresponding extended plaintext sets will satisfy the output difference of the first round in the differential trail. Thus, these plaintext pairs will satisfy the output of the distinguisher with probability $2^{-t\times(R-2)}$, while the probability is $2^{-(\ell-1)\times n_{in}}$ otherwise.

4. For each correct pair (P_1, P_2), apply a t-bit permutation π to $F_{\mathbb{A}}$ and check whether the following truncated difference is satisfied:

$$M_t \cdot (\pi(F_{\mathbb{A}}(P_1[1]) \oplus F_{\mathbb{A}}(P_2[1])), \cdots, \pi(F_{\mathbb{A}}(P_1[\ell]) \oplus F_{\mathbb{A}}(P_2[\ell])))^T = (\Delta, 0, \cdots, 0)^T. \quad (1)$$

Since a correct pair must satisfy

$$M_t \cdot (lsb_t(T_1(P_1[1]) \oplus T_1(P_2[1])), \cdots, lsb_t(T_1(P_1[\ell]) \oplus T_1(P_2[\ell])))^T = (\Delta, 0, \cdots, 0)^T$$

and the affine mapping in the correct choice of $\hat{F}_{\mathbb{A}}(x)$ ensures that the difference in the last $(\ell - 1) \times n_{in}$ bits remains 0, the permutation π that satisfies the condition is very likely to be the correct permutation. If the condition is not met, we discard this candidate of permutation.

Using several correct pairs, the correct permutation will be recovered, and $\hat{F}_{\mathbb{A}}$ is accordingly determined.

Complexity. This step is an additional process compared to the attack by Todo and Isobe described in Section 3.2. The complexity is analyzed later in Section 5.3.

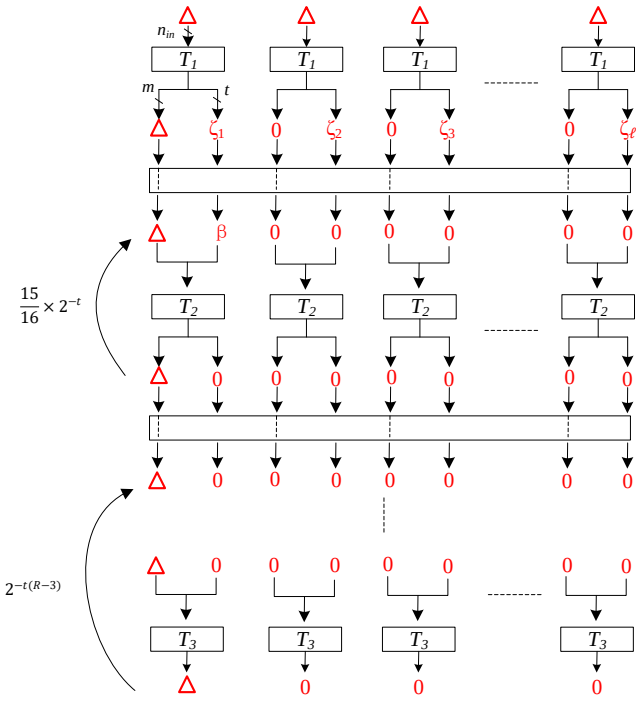

Figure 5: The truncated differential of Yoroi-core part used to recover $\hat{F}_{\mathbb{A}}$.

5.2.2 Process of Recovering $\hat{F}_{\mathbb{B}}$

According to Corollary 2, dividing the input space of the T-box into 2^m subsets $\mathbb{B}_{\eta_j}$ is enough to recover the equivalent representation of Yoroi. Recall the procedure of step 2 in Section 3.2.2, using the leakage of $\mathbb{B}_0$, the attacker constructs input pairs of the last $(\ell - 2)$ T-boxes to make sure that the output of the first round only activates in the most significant n_{in} bits, as illustrated in Figure 3.

In the previous steps, we have recovered $\hat{F}_{\mathbb{A}}$ and obtained some correct pairs. We can use them to construct plaintext pairs achieving the same result. The details are as follows.

1. For each correct pair (P_1, P_2), we first compute $\zeta_1 = \hat{F}_{\mathbb{A}}(P_1[1]) \oplus \hat{F}_{\mathbb{A}}(P_2[1])$ and $\zeta_2 = \hat{F}_{\mathbb{A}}(P_1[2]) \oplus \hat{F}_{\mathbb{A}}(P_2[2])$.

2. Prepare a set $\{P\}$ with 2^{2m} plaintexts from P_1 such that the first element satisfies $\hat{F}_{\mathbb{A}}(P[1]) = i_1$ and the second element satisfies $\hat{F}_{\mathbb{A}}(P[2]) = i_2$ for fixed i_1 and i_2, while the last $(\ell - 2)$ elements keep the same with P_1. Similarly, construct another set $\{P'\}$ from P_2 such that the first element satisfies $\hat{F}_{\mathbb{A}}(P'[1]) = i_1 \oplus \zeta_1$ and the second element satisfies $\hat{F}_{\mathbb{A}}(P'[2]) = i_2 \oplus \zeta_2$ for the same i_1 and i_2, while the other elements keep the same with P_2.

3. Collect pairs $(P[2], P'[2])$ satisfying $lsb_{n_{in}(\ell-1)}(C \oplus C') = 0$.

The above procedure checks all pairs $(P[2], P'[2])$ for a fixed ζ_2. We should try all $\zeta_2 (\neq 0)$. However, ζ_2 is fixed for each correct pair. We cannot freely choose its value. As a solution, we can repeat the above procedure for the same correct pair by constructing the plaintext sets on other elements. For Yoroi-16, the set of differences $\{\zeta_2, ..., \zeta_\ell\}$ for a correct pair can form a base of the finite field $\mathbb{F}_2^4$, while for Yoroi-32, two such distinct sets

are needed to form the base, which can be verified in Appendix B. Base on Observation 2, for Yoroi-16, only 1 correct pair is required, while for Yoroi-32 2 correct pairs are required.

In order to divide the input space of the T-box into 2^m subsets $\mathbb{B}_{\eta_j}$, we need to repeat the procedure (steps 1-3) 12 and 23 times for Yoroi-16 and Yoroi-32, respectively, as explained in Section 3.2.2. To do so, we can traverse the values of i_1 and i_2, and we can also swap one element of the correct pair, i.e. $P_1[i]$ and $P_2[i]$, to generate another pair that has the same output difference at the first round.

Complexity. The complexity of this step is the same as that of recovering $\mathbb{B}_{\eta_j}$ in Section 3.2.2.

We have recovered all the $\hat{F}_{\mathbb{A}}$ and $\hat{F}_{\mathbb{B}}$, thus we can recover $\hat{T}_1$. We can remove the first round of the equivalent representation of Yoroi, and the tables in the remaining rounds can be recovered similarly with less complexity.

Observation 2. *Consider several isolated points $e_i, i \in \mathbb{F}_2^t$ and we can connect the point e_i with $e_{i\oplus\delta}$ if we obtain the difference δ. Then we can connect all the 2^t points after knowing t differences such that the t differences are a set of bases of finite field $\mathbb{F}_2^t$.*

5.3 Complexity and Success Rate

In this section, we analyze the extra complexity required in our attack compared to [TI22] and the success rate. The extra complexity comes from the process of finding correct pairs and recovering correct permutation in step 2 in Section 5.2.1.

Extra Data Complexity. The probability that one correct pair can be detected is $Pr = 1 - (1 - 2^{-t(R-2)})^{2^{2m}}$. Due to the fact that $2^{t(R-2)} = 2^{2m}$ for both Yoroi-16 and Yoroi-32, we have $Pr \approx 1 - e^{-1}$. We expect $2^{2s-1} \times 2^{-t\times(R-2)} \times \frac{15}{16}$ correct pairs for 2^s ciphertexts, and the number of correct pairs that can be detected is $2^{2s-1} \times 2^{-t\times(R-2)} \times \frac{15}{16} \times Pr$. Thus, in order to have N_c correct pairs, the required number of original ciphertexts can be determined by the equation

$$2^{2s-1} \times 2^{-t\times(R-2)} \times \frac{15}{16} \times Pr = N_c,$$

and the data complexity is

$$2^s \times 2^m = \sqrt{\frac{N_c \times 2^{t(R-2)} \times 32}{Pr \times 15}} \times 2^m.$$

Extra Time Complexity. We measure a single operation on the candidate permutations, specifically checking whether each permutation satisfies Equation (1), as one T-box operation. This is a reasonable measurement as each permutation is only 4 bits, which is smaller than the T-box. Considering the fact that each correct pair will exclude a large number of permutation candidates, the process for the first correct pair should dominate the time complexity. Our experiments revealed that for Yoroi-16, a single correct choice would typically recommend approximately $2^{13.15}$ candidate permutations on average. For Yoroi-32, this number increases to approximately $2^{24.74}$. We refer to Appendix A for the analysis of how many candidate permutations a correct pair can recommend.

Since there are $(2^t - 2)!$ candidate permutations, the time complexity for recovering permutation is $(2^t - 2)! \times \ell$ T-box operations. Since Yoroi has $(\ell \times R)$ T-box operations, the corresponding time complexity is equivalent to $\frac{(2^t-2)!}{R}$ Yoroi encryptions.

Therefore, the extra time complexity is

$$\left(\sqrt{\frac{N_c \times 2^{t(R-2)} \times 32}{Pr \times 15}} \times 2^m + \frac{(2^t - 2)!}{R}\right) \text{ Yoroi encryptions.}$$

Success Rate Analysis. Assume that each correct pair will recommend R_{p_w} wrong permutations and the total number of wrong permutations is N_{p_w}. On average, the probability that a correct pair recommends a wrong permutation is R_{p_w}/N_{p_w}. If we set a counter for each permutation, incrementing it by 1 each time a permutation is recommended, the value of a counter follows a binomial distribution. When N_c correct pairs are used, and all the counters corresponding to the incorrect permutations are less than N_c, the correct permutation can be successfully recovered. The success rate is given by

$$Ps = (1 - p_B(N_c, \frac{R_{p_w}}{N_{p_w}}, N_c))^{N_{p_w}}, \tag{2}$$

where $p_B(n, p, k)$ denotes the probability that a random variable is equal to k when it follows a binomial distribution $B(n, p)$.

5.4 Application to Yoroi

5.4.1 Application to Yoroi-16

Our experiments showed that $R_{p_w} \approx 2^{13.15}$ for Yoroi-16. According to the Equation (2), two correct pairs can increase the success rate to over 90%. Considering the correlation among candidate permutations, we set $N_c = 4$ to ensure the correct permutation can be recovered. Consequently, the extra data complexity is $\sqrt{\frac{4\times 2^{24}\times 32}{(1-e^{-1})\times 15}} \times 2^{12} = 2^{25.88}$. The extra time complexity is $2^{25.88} + (2^4 - 2)!/8 = 2^{33.35}$.

In summary, for the whole attack, the data complexity is $2^{33.09} + 2^{25.88} = 2^{33.10}$, and the time complexity is $2^{33.09} + 2^{33.35} = 2^{34.22}$.

5.4.2 Application to Yoroi-32

Our experiments showed that that $R_{p_w} \approx 2^{24.74}$ for Yoroi-32. According to the Equation (2), four correct pairs can increase the success rate to over 90%. Considering the correlation among candidate permutations, we set $N_c = 8$ to ensure the correct permutation can be recovered. Consequently, the extra data complexity is $\sqrt{\frac{8\times 2^{56}\times 32}{(1-e^{-1})\times 15}} \times 2^{28} = 2^{58.38}$. The extra time complexity is $2^{58.38} + (2^4 - 2)!/16 = 2^{58.38}$.

In summary, for the whole attack, the data complexity is $2^{65.44} + 2^{58.38} = 2^{65.45}$, and the time complexity is $2^{65.44} + 2^{58.38} = 2^{65.45}$.

Remark 1. According to the previous analysis, we need to construct a ciphertext structure of size about 2^{30}. Next, we decrypt the ciphertexts, extend the corresponding plaintexts, and encrypt these plaintexts. Finally, we perform pairing on the ciphertexts to check if their differences are active only in the first 32 bits. However, for Yoroi-32, we tested about 2^{115} pairs, and found that the random probability is 2^{-96}, which means there is a significant noise in the correct pairs collected. We need to use the techniques introduced in Section 5.2.2 that swapping the input at a certain T-box of the collected pairs. The new extended plaintexts can also be obtained from the swapped pairs. Thus, the candidates of correct pair can be further filtered. The complexity introduced by this process is negligible.

Remark 2. It is worth mentioning that our blackbox phase is under the chosen-plaintext-ciphertext setting while the one used in the previous work [TI22] is a chosen-plaintext attack. To have a better comparison, we also devised an attack under the chosen-plaintext setting. However, its complexity is significantly higher than that of the previous work. Therefore, we do not consider it a primary contribution of this paper, and present this attack in Appendix C.

6 Experiments

We have verified the correctness of our theory through experiments. For Yoroi-16, in order to obtain 4 correct pairs, we need to construct a ciphertext structure of size approximately $2^{13.88}$ that only activates at most 16 bits. After obtaining the corresponding plaintexts by querying the decryption oracle, these plaintexts can theoretically form approximately 6.30 correct pairs, of which about 4 will be detected. We repeated the experiment 1000 times, and the average results are presented in Table 2.

Table 2: Comparison of the theoretical estimation and the experiment results.

Category	Size of structure	Number of correct pairs	N_c
Theoretical estimate	$2^{13.88}$	6.30	4
Experimental value	$2^{13.88}$	6.27	3.96

Subsequently, we performed experiments to verify the process of recovering correct permutations by using the correct pairs. Considering the large number of candidate permutations, we conducted experiments on Yoroi-32, as it has fewer branches. For convenience, we did not collect actual correct pairs. Instead, we randomly selected 16-bit input pairs satisfying the specific difference[2] of the linear layer and treated them as correct pairs. We conducted several experiments and, after filtering with each correct pair, the remaining number of permutations is shown in Table 3. For more results, please refer to Appendix D. These experiments successfully recovered the correct permutation using no more than 6 correct pairs, while in our attack on Yoroi-32, we provide an average of 8 correct pairs.

Table 3: The number of candidate permutations left.

trails \ N_c	0	1	2	3	4	5
1	$2^{36.34}$	$2^{24.60}$	$2^{13.88}$	6	1	-
2	$2^{36.34}$	$2^{24.63}$	$2^{12.82}$	5	1	-
3	$2^{36.34}$	$2^{24.60}$	$2^{13.30}$	6	2	1

7 Conclusion

In this paper, we propose a family of equivalent representations of Yoroi. We are able to recover an equivalent cipher of the Yoroi without any leaked information of the table entries. According to the complexity analysis in Section 5.3, under the premise of having sufficiently correct pairs, the complexity of recovering correct permutation grows at a super-exponential rate with the extension degree of the finite field in which the linear layer resides. This is of great significance for future attacks and designs.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. This research is supported by the National Key R&D Program of China(Grant No. 2024YFA1013000, 2023YFA1009500), the National

[2]The specific difference refers to the corresponding output difference of the linear layer is $(\Delta, 0, 0, 0)$.

Natural Science Foundation of China (Grant No. 62032014, U2336207), Department of Science & Technology of Shandong Province (No. SYS202201), Quan Cheng Laboratory (Grant No. QCLZD202301, QCLZD202306), the National Natural Science Foundation of China (Grant No. 62302293).

References

[BBIJ17] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, and Martin Bjerregaard Jepsen. Analysis of software countermeasures for whitebox encryption. *IACR Trans. Symmetric Cryptol.*, 2017(1):307–328, 2017.

[BBK14] Alex Biryukov, Charles Bouillaguet, and Dmitry Khovratovich. Cryptographic schemes based on the ASASA structure: Black-box, white-box, and public-key (extended abstract). In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 63–84. Springer, 2014.

[BCD06] Julien Bringer, Hervé Chabanne, and Emmanuelle Dottax. White box cryptography: Another attempt. *IACR Cryptol. ePrint Arch.*, page 468, 2006.

[BGE04] Olivier Billet, Henri Gilbert, and Charaf Ech-Chatbi. Cryptanalysis of a white box AES implementation. In Helena Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography, 11th International Workshop, SAC 2004, Waterloo, Canada, August 9-10, 2004, Revised Selected Papers*, volume 3357 of *Lecture Notes in Computer Science*, pages 227–240. Springer, 2004.

[BHMT16] Joppe W. Bos, Charles Hubain, Wil Michiels, and Philippe Teuwen. Differential computation analysis: Hiding your white-box designs is not enough. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *Lecture Notes in Computer Science*, pages 215–236. Springer, 2016.

[BI15] Andrey Bogdanov and Takanori Isobe. White-box cryptography revisited: Space-hard ciphers. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 1058–1069. ACM, 2015.

[BIT16] Andrey Bogdanov, Takanori Isobe, and Elmar Tischhauser. Towards practical whitebox cryptography: Optimizing efficiency and space hardness. In Jung Hee Cheon and Tsuyoshi 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 126–158, 2016.

[BU18] Alex Biryukov and Aleksei Udovenko. Attacks and countermeasures for white-box designs. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane,*

QLD, Australia, December 2-6, 2018, Proceedings, Part II, volume 11273 of *Lecture Notes in Computer Science*, pages 373–402. Springer, 2018.

[BU21] Alex Biryukov and Aleksei Udovenko. Dummy shuffling against algebraic attacks in white-box implementations. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 219–248. Springer, 2021.

[CCD+17] Jihoon Cho, Kyu Young Choi, Itai Dinur, Orr Dunkelman, Nathan Keller, Dukjae Moon, and Aviya Veidberg. WEM: A new family of white-box block ciphers based on the even-mansour construction. In Helena Handschuh, editor, *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, volume 10159 of *Lecture Notes in Computer Science*, pages 293–308. Springer, 2017.

[CEJvO02a] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. White-box cryptography and an AES implementation. In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers*, volume 2595 of *Lecture Notes in Computer Science*, pages 250–270. Springer, 2002.

[CEJvO02b] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. A white-box DES implementation for DRM applications. In Joan Feigenbaum, editor, *Security and Privacy in Digital Rights Management, ACM CCS-9 Workshop, DRM 2002, Washington, DC, USA, November 18, 2002, Revised Papers*, volume 2696 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2002.

[DLPR13] Cécile Delerablée, Tancrède Lepoint, Pascal Paillier, and Matthieu Rivain. White-box security notions for symmetric encryption schemes. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 247–264. Springer, 2013.

[ER59] P ERDdS and A R&wi. On random graphs i. *Publ. math. debrecen*, 6(290-297):18, 1959.

[FKKM16] Pierre-Alain Fouque, Pierre Karpman, Paul Kirchner, and Brice Minaud. Efficient and provable white-box primitives. In Jung Hee Cheon and Tsuyoshi 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 159–188, 2016.

[Kar10] Mohamed Karroumi. Protecting white-box AES with dual ciphers. In Kyung Hyune Rhee and DaeHun Nyang, editors, *Information Security and Cryptology - ICISC 2010 - 13th International Conference, Seoul, Korea, December 1-3, 2010, Revised Selected Papers*, volume 6829 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 2010.

[KI21] Yuji Koike and Takanori Isobe. Yoroi: Updatable whitebox cryptography. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(4):587–617, 2021.

[KLLM20] Jihoon Kwon, ByeongHak Lee, Jooyoung Lee, and Dukjae Moon. FPL: white-box secure block cipher using parallel table look-ups. In Stanislaw Jarecki, editor, *Topics in Cryptology - CT-RSA 2020 - The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, February 24-28, 2020, Proceedings*, volume 12006 of *Lecture Notes in Computer Science*, pages 106–128. Springer, 2020.

[KR11] Lars R Knudsen and Matthew Robshaw. *The block cipher companion*. Springer Science & Business Media, 2011.

[KSHI20] Yuji Koike, Kosei Sakamoto, Takuya Hayashi, and Takanori Isobe. Galaxy: A family of stream-cipher-based space-hard ciphers. In Joseph K. Liu and Hui Cui, editors, *Information Security and Privacy - 25th Australasian Conference, ACISP 2020, Perth, WA, Australia, November 30 - December 2, 2020, Proceedings*, volume 12248 of *Lecture Notes in Computer Science*, pages 142–159. Springer, 2020.

[LN05] Hamilton E. Link and William D. Neumann. Clarifying obfuscation: Improving the security of white-box DES. In *International Symposium on Information Technology: Coding and Computing (ITCC 2005), Volume 1, 4-6 April 2005, Las Vegas, Nevada, USA*, pages 679–684. IEEE Computer Society, 2005.

[LRM+13] Tancrède Lepoint, Matthieu Rivain, Yoni De Mulder, Peter Roelse, and Bart Preneel. Two attacks on a white-box AES implementation. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 265–285. Springer, 2013.

[MRP12] Yoni De Mulder, Peter Roelse, and Bart Preneel. Cryptanalysis of the xiao - lai white-box AES implementation. In Lars R. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*, volume 7707 of *Lecture Notes in Computer Science*, pages 34–49. Springer, 2012.

[MWP10] Yoni De Mulder, Brecht Wyseur, and Bart Preneel. Cryptanalysis of a perturbated white-box AES implementation. In Guang Gong and Kishan Chand Gupta, editors, *Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings*, volume 6498 of *Lecture Notes in Computer Science*, pages 292–310. Springer, 2010.

[TI22] Yosuke Todo and Takanori Isobe. Hybrid code lifting on space-hard block ciphers application to yoroi and spnbox. *IACR Trans. Symmetric Cryptol.*, 2022(3):368–402, 2022.

[WMGP07] Brecht Wyseur, Wil Michiels, Paul Gorissen, and Bart Preneel. Cryptanalysis of white-box DES implementations with arbitrary external encodings. *IACR Cryptol. ePrint Arch.*, page 104, 2007.

[XL09] Yaying Xiao and Xuejia Lai. A secure implementation of white-box aes. In *2009 2nd International Conference on Computer Science and its Applications*, pages 1–6, 2009.

A Approximate the Value of R_{p_w}

Algorithm 1: Approximate the value of R_{p_w}.

Input: N_c correct pairs, n_1 random 4-bit permutations and the total number of candidate permutations N_{p_w}.

Output: The value of R_{p_w}.

```
sum = 0
forall correct pair (P_{r1}, P_{r2}) do
    forall random permutation π_i do
        δ = (π_i(F_𝔸(p_1^{r1}) ⊕ F_𝔸(p_1^{r2})), ··· , π_i(F_𝔸(p_ℓ^{r1}) ⊕ F_𝔸(p_ℓ^{r2}))) · M_t
        if lsb_{t×(ℓ−1)}(δ) = 0 then
            sum = sum + 1
R_{p_w} = sum/N_c/n_2 × N_{p_w}
return R_{p_w}
```

B Input Difference of Linear Layer in the First Round from Right Paris

Table 4: Maximal linearly independent group from correct pairs.

$(\zeta_1, \cdots, \zeta_\ell)$	Maximal linearly independent group on the 4-bit from $(\zeta_2, \cdots, \zeta_\ell)$
(7,3,14,11,8,1,6,12)	(1,3,8,12)
(14,6,15,5,3,2,12,11)	(3,2,5,11)
(9,5,1,14,11,3,10,7)	(1,3,7,11)
(15,12,13,10,6,4,11,5)	(4,5,6,12)
(8,15,3,1,14,5,13,9)	(1,3,5,9)
(1,10,2,15,5,6,7,14)	(2,6,14,15)
(6,9,12,4,13,7,1,2)	(1,2,4,12)
(13,11,9,7,12,8,5,10)	(8,9,10,12)
(10,8,7,12,4,9,3,6)	(4,8,9,12)
(3,13,6,2,15,10,9,1)	(1,2,6,9)
(4,14,8,9,7,11,15,13)	(8,9,15,13)
(2,7,4,13,10,12,14,15)	(4,12,14,15)
(5,4,10,6,2,13,8,3)	(2,3,4,8)
(12,1,11,8,9,14,2,4)	(1,2,4,8)
(11,2,5,3,1,15,4,8)	(1,2,3,8)

For Yoroi-16, if we obtain a correct pair, we can obtain a maximal linearly independent group from $(\zeta_2, \cdots, \zeta_\ell)$. For Yoroi-32, we can obtain a maximal linearly independent group from two different correct pairs. The input difference is shown in Table 5.

Table 5: Input difference of linear layer in the first round from the right pairs of Yoroi-32.

(14,9,13,11),(15,1,9,5),(1,8,4,14),(13,2,1,10),(3,11,12,1)
(2,3,8,15),(12,10,5,4),(9,4,2,7),(7,13,15,12),(6,5,11,2)
(8,12,6,9),(4,6,3,13),(10,15,14,6),(11,7,10,8),(5,14,7,3)

C The Attack Against Yoroi in the Chosen-Plaintext Scenario

We observe that the process of our blackbox phase can be implemented under the chosen-plaintext setting, except for the chosen-ciphertext process in step 2 in Section 5.2.1. It chooses the ciphertexts to obtain the plaintext pairs where the corresponding ciphertexts of the two elements differ only in the most significant n_{in} bits. Moreover, we found that these ciphertexts also can be obtained by choosing plaintext and performing collisions through the following process.

1. Randomly select 2^r plaintexts, query the encryption oracle and obtain the corresponding ciphertexts.
2. Store all the **ciphertexts** in a hash table indexed by the least significant $n_{in} \times (\ell - 1)$ bits of the corresponding ciphertexts.
3. For each index, pair the plaintexts and obtain the desired pairs.

The subsequent process is exactly the same.

Complexity Increase This process results in an increase in complexity. On one hand, due to the low probability of collisions, a large number of query encryptions are required to obtain a desired pair, while in the original version, only two decryptions are needed. On the other hand, since the pairs do not form a structure, each pair is independent, which introduces further complexity in the process of sieving for correct pairs. More precisely, assume 2^d desired pairs are required. In the chosen-plaintext scenario, the total number of distinct elements is 2^{d+1}. By comparison, in the chosen-plaintext-ciphertext scenario, since the ciphertext is chosen from the structure, only $2^{\frac{d+1}{2}}$ elements are needed. Furthermore, during the phase of extending elements and sieving for correct pairs (as detailed in Section 5.2.1), the number of queries increases with the total number of elements, which makes the chosen-plaintext scenario more complex.

The analysis of the extra complexity is as follows. In this process, 2^r random plaintexts can form approximately 2^{2r-1} pairs. The probability of a collision between ciphertext pairs is $2^{-n_{in}\times(\ell-1)}$. Thus, the total number of pairs collected is $2^{2r-1} \times 2^{-n_{in}\times(\ell-1)}$, of which the number of correct pairs is $2^{2r-1} \times 2^{-n_{in}\times(\ell-1)} \times 2^{-t\times(R-2)} \times \frac{15}{16}$. Therefore, in order to obtain N_c correct pairs, we have

$$N_c = 2^{2r-1} \times 2^{-n_{in}\times(\ell-1)} \times 2^{-t\times(R-2)} \times \frac{15}{16} \times Pr,$$

where Pr is the probability that a correct pair can be detected. Additionally, to check the collected pairs, each pair needs to be extended to 2^{m+1} plaintexts. Thus, the total data and time complexity of this process is

$$2^r + 2^{2r-1} \times 2^{-n_{in}\times(\ell-1)} \times 2^{m+1} = \sqrt{\frac{2 \times N_c \times 16 \times 2^{t\times(R-2)} \times 2^{n_{in}\times(\ell-1)}}{Pr \times 15}} + \frac{N_c \times 16 \times 2^{t\times(R-2)} \times 2^{m+1}}{Pr \times 15}.$$

For Yoroi-16, we set $N_c = 4$, yielding a data and time complexity of $2^{69.88}$, while for Yoroi-32, with $N_c = 8$, the complexity is $2^{88.76}$. The complexity of this process is significantly higher than that of other processes. Consequently, the total complexity of the attack on Yoroi-16 remains $2^{69.88}$, while for Yoroi-32, it is $2^{88.76}$.

D Experimental Results

Table 6: The number of candidate permutations left.

trails \ N_c	0	1	2	3	4	5	6
1	$2^{36.34}$	$2^{24.78}$	$2^{13.39}$	24	1	-	-
2	$2^{36.34}$	$2^{24.77}$	$2^{13.47}$	4	2	1	-
3	$2^{36.34}$	$2^{24.76}$	$2^{12.93}$	4	1	-	-
4	$2^{36.34}$	$2^{24.56}$	$2^{12.89}$	12	1	-	-
5	$2^{36.34}$	$2^{24.64}$	$2^{13.52}$	4	2	1	-
6	$2^{36.34}$	$2^{24.80}$	$2^{13.50}$	24	2	2	1
7	$2^{36.34}$	$2^{24.62}$	$2^{13.10}$	2	2	1	-
8	$2^{36.34}$	$2^{24.64}$	$2^{12.94}$	24	1	-	-
9	$2^{36.34}$	$2^{24.65}$	$2^{13.40}$	24	2	1	-
10	$2^{36.34}$	$2^{25.22}$	$2^{13.90}$	378	6	2	1
11	$2^{36.34}$	$2^{25.22}$	$2^{13.34}$	6	2	2	1
12	$2^{36.34}$	$2^{24.63}$	$2^{12.83}$	16	2	1	-
13	$2^{36.34}$	$2^{25.60}$	$2^{13.90}$	222	6	1	-
14	$2^{36.34}$	$2^{24.55}$	$2^{13.49}$	12	1	-	-
15	$2^{36.34}$	$2^{24.78}$	$2^{13.64}$	12	2	2	1