Truncated Differential Attacks: New Insights and 10-round Attacks on QARMA

Zahra Ahmadian[1], Akram Khalesi[1], Dounia M'foukh[2], Hossein Moghimi[1] and María Naya-Plasencia[2]

[1] Department of Electrical Engineering, Shahid Beheshti University, Tehran, Iran,
z_ahmadian@sbu.ac.ir,a_khalesi@sbu.ac.ir,h.moghimi@mail.sbu.ac.ir
[2] Inria, Paris, France,
dounia.mfoukh@inria.fr,maria.naya_plasencia@inria.fr

Abstract. Truncated differential attacks were introduced by Knudsen in 1994 [1]. They are a well-known family that has arguably received less attention than some other variants of differential attacks. This paper gives some new insight on truncated differential attacks and provides the best-known attacks on both variants of the lightweight cipher QARMA, in the single tweak model, reaching for the first time 10 rounds while contradicting the security claims of this reduced version. These attacks use some new truncated distinguishers as well as some evolved key-recovery techniques.

Keywords: Cryptanalysis · truncated differentials · QARMA · distinguisher · key recovery

1 Introduction

In the realm of modern cryptography, the design and analysis of secure block ciphers play a pivotal role in ensuring the confidentiality of sensitive data. The development of advanced encryption algorithms has been an ongoing endeavor, aiming to thwart increasingly sophisticated attacks while maintaining implementation efficiency. One of the key aspects in this evolution is the study of the variations of differential attacks, which are powerful techniques utilized by cryptanalysts to probe the vulnerabilities of cryptographic primitives. This paper focuses on the truncated differential attack, proposed first in 1994 by Knudsen [1], as a tool for the security evaluation of block ciphers.

Despite some instances of cryptanalysis based on truncated differential attacks as an independent attack [2, 3, 4], this attack has received less attention compared to other variations, such as the impossible differential attack, higher-order differential attacks, as well as boomerang and rectangle attacks. The primary utilization of an automated truncated differential path search has been targeted for discovering paths with minimal activation of S-boxes, to serve as a tool for high-probability concrete differential paths [5]. It also aids in identifying contradictions pertinent to impossible differential attacks [6].

In a recent work [7], a novel MILP (Mixed Integer Linear Programming) tool has been introduced for identifying the optimum truncated differential paths and applied on MIDORI, SKINNY, and CRAFT block ciphers covering a greater number of rounds with higher probabilities compared to their concrete differential counterparts.

This work subsequently garnered some interest in truncated differential attacks. In [8], considering that [7] has utilized certain approximations, an effective algorithm for accurately calculating the truncated differential path probability for a given truncated path is proposed. Another technique aimed at calculating the probability of truncated

differentials, considering the clustering effects (also referred to as the differential effect) has been outlined in [9]. Furthermore, certain studies have employed the methodology presented in [7] to automate the discovery of other distinguishers. These encompass the triangle attack [10] and mixture differential attacks [11], as examples. The significant advantages of truncated differential attacks, compared to concrete differential attacks, are as follows.

- **Simplicity.** The truncated differential attack utilizes a word-oriented variable definition. Moreover, this kind of attack does not inherently depend on the S-box details, hence its MILP model is free from the bottleneck of S-box modeling. Consequently, the truncated differential automatic search tools display enhanced efficiency and running time compared to the automatic tools for finding optimum concrete (bit-oriented) differentials.
- **Efficiency**. There are notable instances where the truncated differential distinguisher outperforms its concrete counterpart. Some examples include `KLEIN`, `MIDORI`, `SKINNY`, and `CRAFT`, for which truncated differential paths have been proposed for the number of rounds, proving that there are no concrete differential distinguishers [7, 2, 3].
- **Value-insensitivity.** The truncated differential distinguisher is inherently independent of the concrete value of the active words. This makes the key recovery part of the attack more flexible, potentially requiring less key material to be guessed, at the two edges of the distinguisher.

Reflection ciphers are a class of symmetric encryption algorithms that exhibit a unique property: the set of encryption functions is identical to the set of decryption functions. In other words, the encryption and decryption processes are the same, making the cipher "reflect" the input to produce the output. This design strategy aims to reduce the implementation cost of the cipher, by minimizing the overhead of decryption on top of the encryption.

`PRINCE` block cipher [12], an SPN cipher with FX construction, stands as one of the most renowned examples of a reflection cipher. To be precise, it possesses the α-reflection property, meaning that decryption is equivalent to the encryption with the related key of $K_{dec} = K_{enc} \oplus \alpha$, where α is a constant. In [13], a new attack called the reflection attack is proposed as a dedicated tool for cryptanalysis of `PRINCE`-like ciphers. It exploits the existence of too many fixed points in the intermediate rounds of the cipher and its extension to the full cipher.

Following in the footsteps of `PRINCE`, `MANTIS` [14] emerges as the subsequent reflection cipher again in the FX framework. It takes inspiration from `PRINCE`'s design while evolving into a tweakable block cipher. Notably, `MANTIS` integrates certain choices from `MIDORI`'s components [15] to enhance its structure. However, a practical attack on $\texttt{MANTIS}_5$ has been presented in [16], attributed to the `MANTIS`'s extremely lightweight components, including the tweak schedule, and the vulnerability resulting from the interaction between the `MIDORI`-inspired round function and the `PRINCE`-inspired inner rounds.

`QARMA` family of block ciphers is the most recent reflection cipher, boasting additional features such as being tweakable, lightweight, and low-latency [17]. Drawing inspiration from `PRINCE`, `MIDORI`, and `MANTIS`, `QARMA` exhibits notable differences both in the structure and in the choice of components. Unlike its predecessors, `QARMA` adopts a three-round Even-Mansour (EM) construction [18] rather than adhering to the FX construction. This departure from the FX construction was motivated by the cryptanalysis presented in [19]. Furthermore, `QARMA`'s decision to pivot to EM construction is motivated by the improved time, memory, and data complexities, which offer superior bounds compared to the FX construction.

Insights gleaned from the MITM and accelerated exhaustive search attacks on `PRINCE` [20], that exploited the unkeyed central construction of `PRINCE`, the designer of `QARMA`

included a key addition in the middle permutation of QARMA. Moreover, this middle permutation is non-involutory to avoid predictable differences at its two sides. Another innovation within the QARMA design pertains to the introduction of a family of almost MDS matrices defined over a ring with zero divisors. They allow to encode rotations in their operation while maintaining the minimal latency associated with binary matrices. The matrices used in QARMA are with the minimum and close to minimum fixed points for 64 and 128-bit versions, respectively. This property as well as suitable whitening keys around the middle permutation makes it secure against the reflection attacks [13].

Similar to PRINCE and MANTIS, QARMA claims a k-bit time-data trade-off security, where k is its key size, meaning that for any attack on QARMA, the product of time and data complexities is less than 2^k.

It is worth noting that the QARMA cipher has been the subject of various cryptanalysis efforts, most of which in the related tweak model, including MITM attack [21, 22], statistical saturation attack [23] and impossible differential attack [24, 25]. The only single-tweak attack [21] is a 10-round MITM attack, but it fails to meet the time-data tradeoff threshold. A review and discussion on the details of QARMA attacks, is provided in Sec. 5.2 of the paper. The designer of QARMA has proposed some security bounds against the differential attack by counting the minimum number of active S-Boxes using Mouha et al.'s MILP search method [26]. However, the resistance of this cipher against the truncated differential attack has not been evaluated, either by the designer or external cryptanalysts.

Contributions . This paper gives new insights into the theory of the -relatively less discussed- truncated differential attack and adds a new dimension to the cryptanalysis of QARMA by introducing the first valid single tweak truncated differential attack on both versions of 10-round QARMA. The contributions of this paper are as follows:

- Extension of truncated differential attack theory: The paper extends the theory of truncated differential attacks by providing a series of evidences and theorems, and formulating the complexities of the truncated differential attack. It is also proved that SPN ciphers with MDS MixColumns are resistant to this kind of attack.
- Discovering Optimal Truncated Differential Distinguishes for QARMA: The almost-MDS property of the MixColumns matrix within the QARMA cipher renders it susceptible to truncated differential analysis. Focusing this cipher, and by employing the automated MILP-based method proposed in [7], the paper identifies the optimum 6 and 4-round truncated differential distinguishers for QARMA family of ciphers.
- Attack on 10-round QARMA: Based on the identified distinguishers, the paper proposes the first valid attacks on both versions of the 10-round QARMA cipher with respect to the security claim trade-off given by the designer of QARMA; *i.e.* $\mathcal{DT} < 2^k$ with data and time complexities $\mathcal{D}$ and $\mathcal{T}$ and the key size k. The attack exploits some evolved key-recovery methods based on list merging techniques and precomputation.

2 Theoretical Background

In this paper, we consider a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^k \to \mathbb{F}_2^n$, with an n-bit block and a k-bit key. The input and output difference variables of E are denoted by ΔX and ΔY, respectively. We assume E is a Markove cipher [27], where the subkeys of iterative block cipher are assumed to be independent and uniformly random. This is the only assumption behind the truncated differential attack, which is a very common and widely accepted assumption in differential cryptanalysis.

Definition 1 (Concrete Differential Probability)**.** For block cipher E, the differential probability of the concrete input difference $\alpha \in F_2^n$ and output difference $\beta \in F_2^n$ is defined

as:

$$\Pr_{x,K}(\alpha \xrightarrow{E} \beta) = \Pr_{x,K}(\Delta Y = \beta | \Delta X = \alpha) = \Pr_{x,K}[E_K(x) \oplus E_K(x \oplus \alpha) = \beta] \tag{1}$$

The differential $(\alpha \xrightarrow{E} \beta)$ is called an *efficient* distinguisher if $\Pr(\alpha \xrightarrow{E} \beta) \gg 2^{-n}$.

Definition 2 (Truncated Differential Probability)**.** For block cipher E, the truncated differential probability with input truncated difference $\Delta_{in} \subseteq F_2^n$, and output truncated difference $\Delta_{out} \subseteq F_2^n$, is defined as:

$$\begin{aligned} \Pr_{x,K}(\Delta_{in} \xrightarrow{E} \Delta_{out}) &= \Pr_{x,K}(\Delta Y \in \Delta_{out} | \Delta X \in \Delta_{in}) \\ &= \Pr_{x,K}[E_K(x) \oplus E_K(x \oplus \alpha) \in \Delta_{out} | \alpha \in \Delta_{in}] \end{aligned} \tag{2}$$

Note that in the rest of the paper, all the probabilities are taken over independent and uniformly distributed random variables x and K. To streamline the presentation, we will omit x, K for simplicity.

Definition 3 (Efficient Truncated Differential)**.** The truncated differential $(\Delta_{in} \to \Delta_{out})$ is called efficient if it can distinguish cipher E from a Pseudo Random Permutation (PRP), which holds if:

$$\Pr(\Delta_{in} \xrightarrow{E} \Delta_{out}) > \Pr(\Delta_{in} \xrightarrow{PRP} \Delta_{out}) = \frac{|\Delta_{out}|}{2^n}. \tag{3}$$

The concept of efficient truncated differential has been introduced in [8] by defining the *Expected Differential Distinguishability* as the average differential probability over the output truncated differentials, which must be significantly larger than 2^{-n} to distinguish.

Property 1. *For block cipher E with concrete input-output differential pair (α, β), it holds that:*

$$\Pr(\alpha \xrightarrow{E} \beta) = \Pr(\beta \xrightarrow{E^{-1}} \alpha) \tag{4}$$

Lemma 1. *For block cipher E with truncated input-output differential pair $(\Delta_{in}, \Delta_{out})$ it holds that:*

$$\Pr(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in}) = \Pr(\Delta_{in} \xrightarrow{E} \Delta_{out}) \frac{|\Delta_{in}|}{|\Delta_{out}|} \tag{5}$$

Proof. By using Bayse theorem,

$$\begin{aligned} \Pr(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in}) &= \Pr(\Delta X \in \Delta_{in} | \Delta Y \in \Delta_{out}) \\ &= \Pr(\Delta Y \in \Delta_{out} | \Delta X \in \Delta_{in}) \frac{\Pr(\Delta X \in \Delta_{in})}{\Pr(\Delta Y \in \Delta_{out})} \\ &= \Pr(\Delta_{in} \xrightarrow{E} \Delta_{out}) \frac{|\Delta_{in}|}{|\Delta_{out}|} \end{aligned} \tag{6}$$

where the last equality holds by assuming uniform distributions for ΔX and ΔY. □

Example 1. Fig. 1 shows a 9-round truncated differential distinguisher for `Skinny-64` with the probability of 2^{-40} in the forward direction [7]. The reverse truncated differential in the backward direction is depicted by red arrows, which has the probability of 2^{-56}. Note that this trail is consistent with Lemma 1, where $|\Delta_{in}| = 2^4$ and $|\Delta_{out}| = 2^{20}$ and $P(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in}) = 2^{-40} \frac{2^4}{2^{20}} = 2^{-56}$.

Property 2. *The truncated differential $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ is efficient, iff $(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in})$ is efficient.*

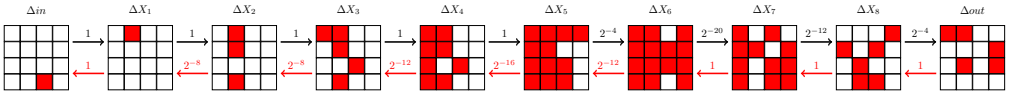

Figure 1: A 9-round truncated differential path for `Skinny-64`. The probability of black (forward) direction is 2^{-40} and for red (backward) direction is 2^{-56}

Proof.

$$\begin{aligned}(\Delta_{in} \xrightarrow{E} \Delta_{out}) \text{ is efficient} &\iff \Pr(\Delta_{in} \xrightarrow{E} \Delta_{out}) > \frac{|\Delta_{out}|}{2^n} \\ &\overset{(5)}{\iff} \Pr(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in}) > \frac{|\Delta_{in}|}{2^n} \\ &\iff (\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in}) \text{ is efficient.} \end{aligned} \tag{7}$$

□

Definition 4 (Optimum Truncated Differential)**.** The truncated differential $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ is called optimum if it is efficient and has the maximal $P(\Delta_{in} \xrightarrow{E} \Delta_{out})|\Delta_{in}|$.

Despite the concrete differential attack in which the data required for the distinguisher is only proportional to the inverse of the differential probability [28], this is not the case with the truncated differential distinguisher. This will be discussed more in Sec. 3.

Property 3. *The differential* $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ *is the optimum truncated differential for* E*, iff* $(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in})$ *is the optimum one for* E^{-1}*.*

Proof. This proposition is proved by contradiction. Suppose that $(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in})$ is not optimum. So, there is another efficient differential $(\mathcal{U} \xrightarrow{E^{-1}} \mathcal{V})$ such that $P(\mathcal{U} \xrightarrow{E^{-1}} \mathcal{V})|\mathcal{U}| > P(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in})|\Delta_{out}|$. By Lemma 1, it holds that:

$$\begin{aligned} P(\mathcal{U} \xrightarrow{E^{-1}} \mathcal{V})|\mathcal{U}| &> P(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in})|\Delta_{out}| \\ P(\mathcal{V} \xrightarrow{E} \mathcal{U})|\mathcal{V}| &> P(\Delta_{in} \xrightarrow{E} \Delta_{out})|\Delta_{in}| \end{aligned} \tag{8}$$

Eq. (8) means that $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ is not the optimum truncated differential distinguisher for E, which is a contradiction. The proof of the reverse direction is analogous to this proof. □

Property 4 (Link between concrete and truncated differential probabilities [8])**.** *For block cipher* E*, with input and output truncated differences* $\Delta_{in}, \Delta_{out} \subseteq F_2^n$*, it holds that:*

$$\Pr(\Delta_{in} \xrightarrow{E} \Delta_{out}) = \frac{1}{|\Delta_{in}|} \sum_{\substack{\alpha \in \Delta_{in}, \\ \beta \in \Delta_{out}}} \Pr(\alpha \xrightarrow{E} \beta) \tag{9}$$

Proof. Using the law of total probability and the rule of union of disjoint events, we can

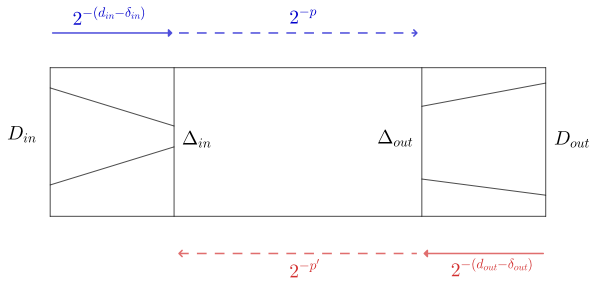

Figure 2: Truncated differential attack framework

write:

$$\begin{aligned}
\Pr(\Delta_{in} \to \Delta_{out}) &= \Pr(\Delta Y \in \Delta_{out} | \Delta X \in \Delta_{in}) \\
&= \sum_{\alpha \in \Delta_{in}} \Pr(\Delta Y \in \Delta_{out} | \Delta X \in \Delta_{in}, \Delta X = \alpha) \Pr(\Delta X = \alpha | \Delta X \in \Delta_{in}) \\
&= \frac{1}{|\Delta_{in}|} \sum_{\alpha \in \Delta_{in}} \Pr(\Delta Y \in \Delta_{out} | \Delta X = \alpha) \\
&= \frac{1}{|\Delta_{in}|} \sum_{\beta \in \Delta_{out}} \sum_{\alpha \in \Delta_{in}} \Pr(\Delta Y = \beta | \Delta X = \alpha) \\
&= \frac{1}{|\Delta_{in}|} \sum_{\substack{\alpha \in \Delta_{in}, \\ \beta \in \Delta_{out}}} \Pr(\alpha \to \beta)
\end{aligned}$$

□

Property 4 implies that the probability of truncated differential $(\Delta_{in} \to \Delta_{out})$ is neither greater nor smaller than each of its consistent concrete differentials $(\alpha \to \beta), \alpha \in \Delta_{in}, \beta \in \Delta_{out}$. Moreover, assume $(\alpha^* \to \beta^*)$ is the optimum (highest-probability) concrete differential (which practically corresponds to minimum or near to minimum active S-boxes), and $(\Delta^*_{in} \to \Delta^*_{out})$ is the optimum truncated differential. Then, according to Property 4, $(\alpha^* \to \beta^*)$ is not necessarily an instantiation of $(\Delta^*_{in} \to \Delta^*_{out})$, i.e. it does not necessitate that $(\alpha^* \to \beta^*) \in (\Delta^*_{in} \to \Delta^*_{out})$.

This means that the truncated differential can serve as an independent distinguisher, with the potential to surpass the performance of concrete differential distinguishers, in some cases.

3 Truncated Differential Attack

Let $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ be an r_d-round truncated differential of probability 2^{-p} for block cipher E. As shown in Fig. 2, we extend Δ_{in} in the backward direction for r_{in} rounds to get the difference D_{in} and Δ_{out} in the forward direction for r_{out} rounds to get D_{out}, both with probability 1. We denote $|D_x| = 2^{d_x}$ and $|\Delta_x| = 2^{\delta_x}$, where $x \in \{in, out\}$. According to Def. 3, $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ is an efficient distinguisher if

$$p < n - \delta_{out} \tag{10}$$

In the following, we formulate the parameters of the chosen plaintext attack constructed over the truncated differential distinguisher $(\Delta_{in} \xrightarrow{E} \Delta_{out})$.

Data Complexity. To generate the pairs required for the attack, we construct 2^s structures in plaintext, each of which is constant in non-active bits, and take all $2^{d_{in}}$ values in active bits of D_{in}. So, each structure can generate about $2^{2d_{in}-1}$ pairs with differences belonging to D_{in}.

Lemma 2. *The probability that a pair with plaintext difference D_{in} come up with a difference Δ_{in} after r_{in} rounds is $P_{filt} = 2^{-(d_{in}-\delta_{in})}$.*

Proof. We can assume the differential $(\Delta_{in} \xrightarrow{E^{-1}} D_{in})$ for the first r_{in} rounds of E in backward direction, with probability 1. Applying Lemma 1 to this differential, yields:

$$\begin{aligned} \Pr(D_{in} \xrightarrow{E^{-1}} \Delta_{in}) &= \Pr(\Delta_{in} \xrightarrow{E^{-1}} D_{in}) \frac{|\Delta_{in}|}{|D_{in}|} \\ &= 1 \cdot \frac{2^{\delta_{in}}}{2^{d_{in}}} = 2^{-(d_{in}-\delta_{in})} \end{aligned} \tag{11}$$

□

Therefore, the number of total pairs required for the attack must be equal to $(2^{-p} \times P_{filt})^{-1} = 2^{p+d_{in}-\delta_{in}}$. This gives the number of required structure as $2^{s+2d_{in}-1} = 2^{p+d_{in}-\delta_{in}}$ which yields $s = p - d_{in} - \delta_{in} + 1$. Finally, the data required for the attack would be as follows.

$$\mathcal{D} = 2^{s+d_{in}} = 2^{p-\delta_{in}+1} \tag{12}$$

Note that to minimize the data complexity, it is necessary to minimize the value of $p - \delta_{in}$, which is consistent with the definition of the optimum distinguisher, given in Def. 4

Time Complexity. The probability that a differential pair with difference Δ_{in} at round r_{in} have a difference belonging to D_{out} at the output is $P_{sieve} = 2^{-(n-d_{out})}$. So, the total number of sieved pairs supposed to be processed in the key recovery phase of the attack is $\mathcal{P} = 2^{p-n+d_{in}+d_{out}-\delta_{in}}$, and the time complexity of the attack is:

$$\mathcal{T} = (2^{p-\delta_{in}+1} + 2^{p-\delta_{in}+1}\frac{C_S}{C_E} + 2^{p-n+d_{in}+d_{out}-\delta_{in}}\frac{C_{KR}}{C_E})C_E \tag{13}$$

where C_E, C_S, and C_{KR} are the time complexities of the encryption, the sieving step, and the key recovery step, respectively.

The concrete differential attack is a symmetric attack, which means that if there is an attack in the forward direction, there is also another one with the same main parameters, using the reverse distinguisher in the backward direction [29]. In the following theorem, we show that the same case is valid for the truncated differential attack.

Theorem 1. *Suppose that there is a chosen plaintext truncated differential attack on block cipher E based on the distinguisher $(\Delta_{in} \xrightarrow{E} \Delta_{out})$ of probability 2^{-p}, with data complexity $\mathcal{D}$ and the total sieved pairs $\mathcal{P}$. we can construct a chosen ciphertext attack, using the reversed truncated differential $(\Delta_{in} \xleftarrow{E^{-1}} \Delta_{out})$, with the same data complexity $\mathcal{D}$ and total sieved pairs $\mathcal{P}$.*

Proof. Suppose that $P(\Delta_{out} \xrightarrow{E^{-1}} \Delta_{in}) = 2^{-p'}$. So, according to Lemma 1,

$$p' = p - \delta_{in} + \delta_{out} \tag{14}$$

This distinguisher is efficient if $p' < n - \delta_{in}$ which, given (14), is equivalent to the efficiency of the forward distinguisher (10). Suppose that we construct $2^{s'}$ structures in the output of the cipher, each of which contains $2^{d_{out}}$ ciphertexts, that gives $2^{2d_{out}-1}$ pairs of ciphertexts.

So, the total number of pairs would be $2^{s'+2d_{out}-1}$. The filtering probability from D_{out} to Δ_{out} is $P'_{filt} = 2^{-(d_{out}-\delta_{out})}$. So, the total pairs required for the attack is $2^{p'+d_{out}-\delta_{out}}$ which must be equal to $2^{s'+2d_{out}-1}$. In this way, the total number of structures would be obtained as $2^{s'} = 2^{p'-d_{out}-\delta_{out}+1}$. Therefore, the data complexity of the attack is

$$\mathcal{D}' = 2^{s'+d_{out}} = 2^{p'-\delta_{out}+1} = 2^{p-\delta_{in}+1} = \mathcal{D} \tag{15}$$

Finally, the sieving probability is $P'_{sieve} = 2^{-(n-d_{in})}$, and Total pairs after sieving is

$$\mathcal{P}' = 2^{p'-n+d_{in}+d_{out}-\delta_{out}} = 2^{p-n+d_{in}+d_{out}-\delta_{in}} = \mathcal{P} \tag{16}$$

The time complexity of the attack mainly depends on $\mathcal{D}'$ and $\mathcal{P}'$, as

$$\mathcal{T}' = (\mathcal{D}' + \mathcal{D}'\frac{C'_S}{C_D} + \mathcal{P}'\frac{C'_{KR}}{C_D})C_D \tag{17}$$

□

Note that the concrete differential attack can be regarded as a special case of truncated differential attack, in which $\delta_{in} = \delta_{out} = 0$.

4 Potential Targets for Truncated Differential Attack

Truncated differential attacks are particularly effective on word-oriented Substitution-Permutation Network (SPN) ciphers. While their efficiency remains unlinked to the S-box specification, it becomes significantly reliant on the differential characteristics of the `MixColumns` matrix. In the following, we present the general structure of a word-oriented SPN cipher, a framework that has served as the foundation for various block cipher designs including `AES`, `MIDORI`, `SKINNY`, `QARMA`, and `CRAFT`. Subsequently, we establish a theorem that identifies a prerequisite condition for the effectiveness of the truncated differential distinguisher.

Definition 5 (Word-oriented SPN cipher)**.** The block cipher E, featuring an internal state matrix of $t \times t$ of m-bit words, is called a word-oriented SPN cipher, if it undergoes the following sequence of four operations in each round, in any order of execution:

- `Subkey addition`: XORs a subkey of size t^2 m-bit words to the internal state.
- `S-box`: applies m-bit S-boxes to each m-bit word of the internal state, in parallel.
- `Permutation`: applies the word-wise permutation π on Z_{t^2} within the internal state, *i.e.* $Y[i] = \pi(X[i]) = X[\pi(i)]$ for $i \in 0, \ldots, t^2 - 1$, where $X[i]$ denotes the i^{th} word of the internal state. Here, every column of $X[i]$ gets mapped to exactly to the t columns of $Y[i]$.
- `MixColumns`: multiplies matrix M to each column of the internal state, in parallel. where, M is a $t \times t$ matrix M over $\mathbb{F}_{2^m}$.

While MDS (Maximum Distance Separable) matrices do provide optimal diffusion for the `MixColumns` operation, this constraint has been intentionally relaxed in several ciphers to gain implementation advantages. In the next theorem, we show that an MDS `MixColumns` matrix in the word-oriented SPN ciphers is a sufficient condition for provable security against truncated differential attacks, under two assumptions: Markov cipher and uniformity of the output of MDS matrices. The former is a widely accepted assumption in different types of cryptanalysis of block ciphers, directly based on which the uniformity

and independence of the output difference of S-boxes (which is the input of `MixColumns`) is concluded [30]. The latter is discussed in the following.

We define $\mathsf{Tr} : \mathbb{F}_{2^m}^t \rightarrow \mathbb{F}_2^t$ as the m-bit truncation operation, i.e. assuming $\mathbf{x} = [x_1, \ldots, x_{t-1}]^\top, x_i \in \mathbb{F}_{2^m}$, then $\mathsf{Tr}(\mathbf{x}) = [r_0, \ldots, r_{t-1}]^\top, r_i \in \mathbb{F}_2$, where $r_i = 0$ iff $x_i = 0$. Let $\mathsf{Hw} : \mathbb{F}_2^t \rightarrow \{0, \ldots, t\}$ be the Hamming weight operator, i.e. $\mathsf{Hw}(\mathbf{a})$ is the number of non-zero elements of $\mathbf{a}$.

Lemma 3 (Near-Uniform Distribution of MDS Matrix Output)**.** *Let M be an MDS $t \times t$ matrix over $\mathbb{F}_{2^m}$. For the truncated difference vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}_2^t$, it holds that:*

$$\begin{aligned} \Pr(\mathbf{a} \xrightarrow{M} \mathbf{b}) &= \Pr_{\mathbf{x}}(\mathsf{Tr}(M\mathbf{x}) = \mathbf{b} | \mathsf{Tr}(\mathbf{x}) = \mathbf{a}) \\ &= \begin{cases} 1 & \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) = 0 \\ 0 & 1 \leq \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) \leq t \\ \approx 2^{-m(t-\mathsf{Hw}(\mathbf{b}))} & \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) \geq t+1 \end{cases} \end{aligned} \tag{18}$$

Assuming uniform distribution for $\mathbf{x}$.

Proof. The proof of this lemma is given in Appendix A. □

For all MDS matrices M, the accurate value of the transition probability $\Pr(\mathbf{a} \xrightarrow{M} \mathbf{b})$ is given by (33) and (32) of Appendix A, that can be approximated as (18). Both the accurate and approximated transition probabilities are independent of the MDS matrix description. Moreover, the transition probabilities depend only on $\mathsf{Hw}(\mathbf{a})$ and $\mathsf{Hw}(\mathbf{b})$. For, $t = 4, m = 4$, and $m = 8$ cases, the accurate and approximated transition probabilities of MDS matrices are reflected in Differential Branch Table (DBT) in Table 4 of Appendix B. It can be seen that the approximation works well, specifically for larger values for m.

Theorem 2. *Under the assumption of Markov cipher and uniform distribution for output of MDS matrix, There is no efficient truncated differential distinguisher for 3 rounds of a word-oriented SPN cipher with an MDS `MixColumns` matrix.*

Proof. Without loss of generality, we consider the order of operations within a round as given in Def. 5. Let X_i, Y_i, and Z_i be the truncated differences of the input state, input, and output of `MixColumns` for round i, respectively. Let's denote the number of zero columns in Z_i as c_i for $i = 1, 2, 3$, which is also equivalent to the number of zero columns in Y_i. Each column in $Z_i = X_{i+1} = \pi^{-1}(Y_{i+1})$ inherits at least c_{i+1} zero words from Z_{i+1}. Therefore, the count of zero words in Z_i, excluding those within zero columns, is at least $(t - c_i)c_{i+1}$. We use P_i to denote the truncated differential probability for round i. Considering the uniform distribution of the MDS matrix output (as stated in Lemma 3), it holds that:

$$P_i \leq 2^{-m((t-c_i)c_{i+1})} \quad i = 1, 2 \tag{19}$$

Let w_3 denote the number of zero words of Z_3, not belonging to a zero column so $P_3 = 2^{-mw_3}$. Therefore, the probability of the truncated differential path would be upper-bounded by $P_1P_2P_3 \leq 2^{-m(tc_2 - c_1c_2 + tc_3 - c_2c_3 + w_3)}$. The probability of Z_3 being the truncated differential pattern of the output of a PRP is $P_{PRP} = 2^{-m(tc_3 + w_3)}$. To prove the theorem, it suffices to show that the upper bound of $P_1P_2P_3$ is less than or equal to P_{PRP}, which holds if $c_2 = 0$ or $c_1 + c_3 \leq t$. In the former case, we have $P_1P_2P_3 \leq 2^{-mw_3} = P_{PRP}$, which proves the claim. In the latter, consider a nonzero column of Z_2 and Y_2, corresponding to the `MixColumns` operation of round 2. This `MixColumns` matrix has at least c_1 and c_3 input and output zero words. Since the `MixColumns` matrix is MDS, it follows that $c_1 + c_3$ must be less than t, thereby completing the proof. □

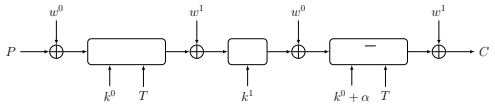

Figure 3: Overal scheme of QARMA [17]

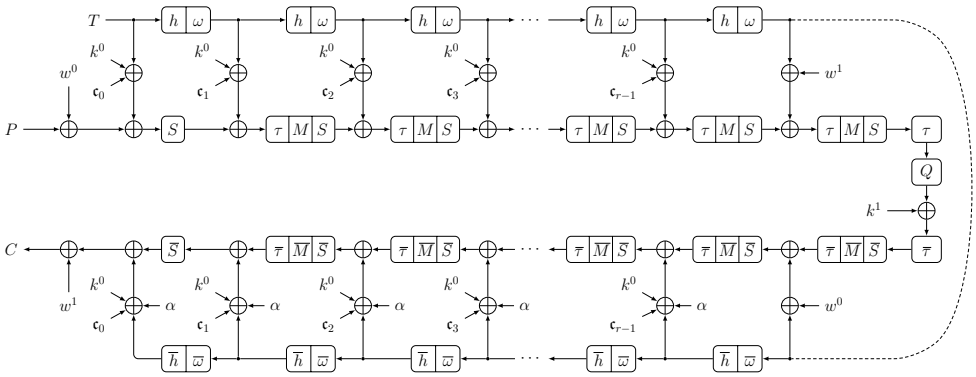

Figure 4: The structure of QARMA [17]

Theorem 2 implies that word-oriented SPN ciphers with non-MDS MixColumns matrices would be potentially vulnerable to truncated differential attacks. This assertion is corroborated by observations made during truncated differential cryptanalysis of MIDORI, SKINNY, and CRAFT [7], all of which use non-MDS MixColumns matrices. Within the QARMA family of block ciphers, to avoid the expensive implementation of MDS matrices, an almost-MDS matrix is selected as the MixColumns matrix. This motivated us to evaluate its security against truncated differential attack, which is outlined in the following Sections.

5 Introduction of QARMA Block cipher family

QARMA is a family of lightweight tweakable block ciphers proposed in 2017 [17], fitting to applications such as memory encryption, the generation of very short tags, and the construction of keyed hash functions. This family of block ciphers has two versions: QARMA-64 and QARMA-128. QARMA-64 has a 64-bit block size with a 128-bit encryption key, while QARMA-128 has a 128-bit block size with a 256-bit key. The design of QARMA was influenced by PRINCE [12] and MANTIS [14]. The cipher is intended for fully-unrolled hardware implementations with low latency, such as memory encryption.

5.1 Specifications of QARMA-64 and QARMA-128

QARMA is an Even-Mansour cipher that uses three stages at the middle, and whitening keys XORed in at the beginning and end of the cipher. The structure of this cipher is shown in Fig. 3 and 4. For QARMA-n, $n = 64$ or 128, the data is split into 16 m-bit words, $m = 4$ or 8 respectively, arranged in a 4×4 internal state matrix IS, which is shown as:

$$IS = \begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} \tag{20}$$

The key size of QARMA-n is $2n = 32m$ bits. The secret key K is divided into two halves of length n-bit, $K = w^0||k^0$, and extends to $k^1 = k^0$ and $w^1 = (w^0 \ggg 1) + (w^0 \gg (16m-1))$. For the sake of simplicity, in the rest of the paper, we refer to k^0 as k, w^0 as w, and w^1 as w'.

The tweak size of QARMA-n is $n = 16m$ bits denoted as $T = t_0||t_1||...||t_{15}$. The tweak update function consists of a permutation h as well as an LFSR ω. The permutation h is applied as $h(T) = t_{h(0)}||t_{h(1)}||...||t_{h(15)}$ where $h = [6, 5, 14, 15, 0, 1, 2, 3, 7, 12, 13, 4, 8, 9, 10, 11]$. The LFSR ω updates the tweak words $\{t_0, t_1, t_3, t_4, t_8, t_{11}, t_{13}\}$ as $(b_3, b_2, b_1, b_0) \rightarrow (b_0 \oplus b_1, b_3, b_2, b_1)$ for $m = 4$ and $(b_7, b_6, ..., b_0) \rightarrow (b_0 \oplus b_2, b_7, ..., b_1)$ for $m = 8$.

The *Forward Round Function* $\mathcal{R}(IS; tk)$ of QARMA-n is composed of the following layers:

1. `AddRoundTweakey`. The round tweakey tk is added to IS.
2. `ShuffleCells`. For both versions of QARMA, the internal state IS is shuffled according to the word permutation τ.

$$\begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} \xrightarrow{\tau} \begin{pmatrix} s_0 & s_{11} & s_6 & s_{13} \\ s_{10} & s_1 & s_{12} & s_7 \\ s_5 & s_{14} & s_3 & s_8 \\ s_{15} & s_4 & s_9 & s_2 \end{pmatrix} \tag{21}$$

3. `MixColumns`. The MixColumns matrix M_m, which is multiplied by IS in QARMA-n is:

$$M_4 = \begin{pmatrix} 0 & \rho & \rho^2 & \rho \\ \rho & 0 & \rho & \rho^2 \\ \rho^2 & \rho & 0 & \rho \\ \rho & \rho^2 & \rho & 0 \end{pmatrix}, \quad M_8 = \begin{pmatrix} 0 & \rho & \rho^4 & \rho^5 \\ \rho^5 & 0 & \rho & \rho^4 \\ \rho^4 & \rho^5 & 0 & \rho \\ \rho & \rho^4 & \rho^5 & 0 \end{pmatrix} \tag{22}$$

   which is defined over ring $R_m = \mathbb{F}_2[X]/(X^m + 1)$, and the multiplication by the image ρ of X in the ring R_m is just a simple circular left rotation of X. M_m is a symmetric and involutive matrix, i.e. $M_m = M_m^\top = M_m^{-1}$.
4. `SubCells`. A m-bit S-Box is applied to all words of IS in QARMA-n.

The `ShuffleCells` and `MixColumns` layers are omitted for the final round. The *Backward Round Function* $\bar{\mathcal{R}}(IS; tk)$ is the inverse of the forward round function $\mathcal{R}$. The *Pseudo-Reflector* function $\mathcal{P}(IS; tk)$ is positioned at the center of the cipher, which is

$$\mathcal{P} = \bar{\mathcal{R}}(\bar{\tau}(k \oplus Q_m(\tau(\mathcal{R}(IS))) \tag{23}$$

where $Q_m = M_m$, and the bar over a transformation denotes its inverse. Finally, the $(2r + 2)$-round QARMA$_r$-n is defined as $\bar{\mathcal{R}}^r(\mathcal{P}(\mathcal{R}^r(\cdot)))$.

5.2 Security Claims and Attacks on QARMA

Time-Data Trade off for QARMA-n. The designer of QARMA has claimed that: *Similarly to MANTIS and PRINCE, for QARMA-64 and QARMA-128, with $r = 7$ and $r = 11$ respectively, we claim that they attain n bits of tradeoff security.* This statement means that any attack on QARMA-n with data and time complexities $\mathcal{D}$ and $\mathcal{T}$, is a valid attack as far as $\mathcal{DT} < 2^{2n}$.

Number of rounds. For QARMA-64, r is chosen as 7, i.e. the total round of QARMA-64 would be 16 rounds. However, it is stated that the cipher is believed to be secure against practical attacks already for $r = 6$, i.e. 14 rounds, with some use cases even allowing for $r = 5$, i.e. 12 rounds. For QARMA-128, r is chosen as 11, i.e. the total round of QARMA-128 would be 24 rounds. However, it is stated that the cipher is believed to be secure against practical attacks already for $r = 8$, i.e. 18 rounds.

Table 1: Summary of the external cryptanalysis of QARMA

Cipher	Type	Model	Whitening	Symmetry	Rounds	Time	Data	Memory	Ref.
QARMA-64	MITM	RT	Yes	Yes	8	2^{90}	2^{16}	2^{90}	[22]
	MITM	RT	Yes	No	9	2^{89}	2^{16}	2^{89}	[22]
	SS	RT	Yes	Yes	10	2^{59}	2^{59}	$2^{29.6}$	[23]
	ID	RT	Yes	No	10	$2^{125.8}$	2^{62}	2^{37}	[31]
	ID	RT	No	No	11	2^{69}	$2^{58.38}$	$2^{63.38}$	[24]
	MITM	ST	No	No	10	2^{116}	2^{53}	2^{116}	[21]
	TD	**ST**	**Yes**	**Yes**	**10**	$\mathbf{2^{65.72}}$	$\mathbf{2^{49.39}}$	$\mathbf{2^{63.12}}$	**Sec. 6.2**
QARMA-128	MITM	RT	Yes	Yes	10	$2^{164.48}$	2^{88}	2^{97}	[24]
	MITM	RT	Yes	Yes	10	2^{156}	2^{88}	2^{145}	[22]
	ID	RT	Yes	No	10	$2^{120.94}$	$2^{104.02}$	$2^{94.50}$	[25]
	ID	RT	No	No	11	2^{137}	$2^{111.38}$	$2^{120.38}$	[24]
	ID	RT	No	No	11	$2^{145.98}$	$2^{102.54}$	$2^{135.54}$	[25]
	TDIB	RT	Yes	No	11	$2^{126.1}$	$2^{126.1}$	2^{71}	[23]
	MITM	RT	Yes	No	12	$2^{156.06}$	2^{88}	2^{154}	[24]
	MITM	ST	No	No	10	2^{232}	2^{105}	2^{232}	[21]
	TD	**ST**	**Yes**	**Yes**	**10**	$\mathbf{2^{137.84}}$	$\mathbf{2^{103.95}}$	$\mathbf{2^{134.51}}$	**Sec. 6.3**

MITM: Meet In the Middle
SS: Statistical Saturation
RT/ST: Related Tweak/Single Tweak
ID: Impossible Differential
TD: Truncated Differential
TDIB: Tweak Difference Invariant Bias

Cryptanalysis history. Most of the cryptanalytic work on QARMA-64 and QARMA-128 is in the related tweak model. In [22], the idea of two related tweaks in the MITM attacks on 8 and 9 rounds of QARMA-64, along with a related tweak on 10 rounds of QARMA-128, has been proposed. They are based on a 5-round MITM distinguisher demanding a δ-set on tweak variables. For QARMA-64, the $\mathcal{TD}$ is 2^{106} and 2^{105}, respectively, while QARMA-128 holds 2^{244}. Li et al. in [23] proposed a new cryptanalytic method that can be seen as a related-tweak statistical saturation attack by making a link between related-tweak statistical saturation distinguishers and the tweak difference invariant bias. By applying this approach, a related-tweak statistical saturation attack for 10-rounds of QARMA-64 and an 11-round attack on QARMA-128 were obtained.

In [24], two related-tweak impossible differential attacks on the 11 rounds of both versions of QARMA, without whitening keys, a MITM attack on the 10 rounds of QARMA-128 with whitening keys, and 12 rounds of QARMA-128 with the whitening keys are proposed. In[31] and [25], two related-tweak impossible differential attacks on QARMA-64 and QARMA-128 are proposed, respectively. The former, which is a 10-round key recovery attack with time and data complexity of $2^{125.8}$ and 2^{62}, violates the $\mathcal{TD}$ threshold claimed for QARMA-64.The latter is an 11-round atack on QARMA-128 that omits the outer whitening key with time complexity and data complexity of $2^{145.98}$ and $2^{102.54}$.

The only work in the single-tweak model is [21], which proposes 10-round MITM attacks for both versions of QARMA. However, it does not meet the time-data tradeoff threshold. For QARMA-64 the time complexity and data complexity were reported as 2^{70} and 2^{53} respectively, but its memory complexity, which is the lower bound of the time complexity, is 2^{116}. For QARMA-128, the time complexity and data complexity were 2^{141} and 2^{105} while, its memory complexity remains consistent at 2^{232}. Hence, both of these attacks do not satisfy the time-data tradeoff threshold and can not be considered as valid attacks of QARMA.

A summary of all the attacks on reduced-round QARMA, along with the new attacks presented in this paper, is given in Tab. 1.

6 Truncated Differential attack on QARMA-64 and QARMA-128

In this section, we present 10-round attacks on both versions of QARMA which are the best valid attacks on this cipher in the single-tweak model. We first introduce the optimum 6-round truncated distinguisher for this cipher, then based on the 4 inner rounds of which,

we propose the 10-round key recovery attack.

6.1 6-round Truncated Distinguisher for QARMA-n

The process of determining the optimal truncated differential path involves two steps: the identification of the optimal path and computing its accurate probability. To discover the optimal truncated differential path, we employ the Mixed-Integer Linear Programming (MILP) tool proposed in [7], which fully automates the search process, in an efficient way. This approach is built upon the assumption of independent and uniformly distributed variables for the output differences of the active S-boxes, a consequence of the Markov cipher assumption [27]. Under this assumption, the only part of the cipher that needs to be modeled is the Differential Branch Table (DBT) (defined in Def. 6 of Appendix B) associated with the MixColumns matrix M_m. The DBT of QARMA variants for $m = 4$ and 8 can be found in Appendix B. In [7], an approximated DBT is employed for its simplicity. In this approximation, all the transition probabilities through MixColumns matrix are rounded to the nearest power of 2^{-m}. However, when dealing with the DBT of M_4 and M_8 of QARMA, we calculate more precise values for transition probabilities, which span a broad range of non-integer values. Thus, we employ the MILP model proposed by Abdelkhalek in [5], specifically designed for the MILP modeling of large S-boxes with non-integer transition probabilities, which is well-suited to our case.

In accordance with Def. 4, we define the objective function as the minimization of $p - \delta_{in}$. To ensure that the resulting path is an efficient one, as stipulated in Def. 3, we incorporate the constraint $p \leq n - \delta_{out}$ into the model, for efficient distingusher.

Once we have identified the optimal path under the Markov cipher assumption, we employ Proposition 4 to determine the accurate value of the distinguisher probability. There are two approaches for computing this value. The first approach, introduced in [8], is an efficient and speedy algorithm. It calculates the truncated probability for a given truncated path (referred to as an "activity pattern" in [8]) by taking into account all the concrete paths consistent with the specified path. The second approach involves utilizing a SAT- or MILP-based automatic method to find all concrete differentials ($\alpha \to \beta$), where $\alpha \in \Delta^*_{in}$ and $\beta \in \Delta^*_{out}$, with $(\Delta^*_{in}, \Delta^*_{out})$ representing the input/output difference of the optimal truncated path obtained in the previous step. While this approach may not be as swift as the method proposed in [8], it offers the advantage of considering any potential differential effects, i.e. the concrete paths that are not necessarily confined to the optimal truncated path within the internal state differences. Therefore, we opt for the second approach to refine the probability of the optimal truncated differential distinguisher. Both the MILP- and SAT-based models were employed independently, and they provided perfectly matching solutions.

6-round distinguishers. We searched for the longest optimum truncated differential distinguisher for the middle part of QARMA-n and found a set of 16 distinguishers covering $\bar{\mathcal{R}}^2(\mathcal{P}(\mathcal{R}^2(\cdot)))$, for both QARMA-64 and QARMA-128. These distinguishers covers 6 full $\mathcal{R}$ (or $\bar{\mathcal{R}}$), though seven {ShuffleCells + MixColumns} layers are involved in. They are as follows:

$$\bar{\tau}(M(a \cdot \mathbf{e}_i)) \xrightarrow[2^{-p}]{\bar{\mathcal{R}}^2(\mathcal{P}(\mathcal{R}^2(\cdot)))} \bar{\tau}(M(b \cdot \mathbf{e}_i)), \quad a, b \in \mathbb{F}_2^m/\{0\}, \quad i = 0, \ldots, 15 \tag{24}$$

where $a \cdot \mathbf{e}_i$ is the 4×4 matrix whose i^{th} element is a, while all other elements are zero. All the 16 paths given in (24) exhibit a reflective pattern and share the following parameters:

- For QARMA-64, $\delta_{in} = 4$, and $\delta_{out} = 4$, and the approximated p merely based on the DBT is 51.8, which is refined to the accurate value $p = 49.7$, using SAT- and MILP-based automatic methods for computing (9), independently.

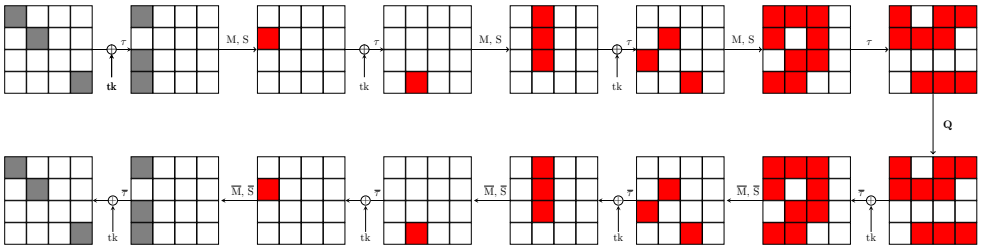

Figure 5: A 6-round truncated differential distinguisher for QARMA-64 and QARMA-128 with probability $P = 2^{-49.7}$ and $2^{-107.6}$, respectively. The red part is the 4-round path with the same probabilities.

- For QARMA-128, $\delta_{in} = 8$, and $\delta_{out} = 8$, and the approximated p merely based on the DBT is 108.77, which is refined to the accurate value $p = 107.60$ using a SAT-based automatic method.

The profile of concrete paths and their probabilities are shown in Appendix C. Moreover, this validation also shows that the accuracy of the pure-truncated search [7] is well enough in case of QARMA. An instance of the introduced distinguisher for $i = 4$ for QARMA-64 is shown in Fig. 5.

$$\begin{pmatrix} a\rho & 0 & 0 & 0 \\ 0 & a\rho & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & a\rho^2 \end{pmatrix} \xrightarrow[2^{-49.7}]{\bar{\mathcal{R}}^2(\mathcal{P}(\mathcal{R}^2(\cdot)))} \begin{pmatrix} b\rho & 0 & 0 & 0 \\ 0 & b\rho & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & b\rho^2 \end{pmatrix} \tag{25}$$

Note that even though the output of these distinguishers has $3m$ bits of active words, $\delta_{out} = m$. This is because the three output nibbles are interdependent, as demonstrated in (25), for instance.

4-round distinguishers. The first and last rounds of transitions in the paths given in (24) are deterministic. So, if we omit these two rounds, we come up with a series of 4-round totally reflective distinguishers with the same probability, which is as follows.

$$a \cdot \mathbf{e}_i \xrightarrow[2^{-p}]{\bar{\mathcal{R}}(\mathcal{P}(\mathcal{R}(\cdot)))} b \cdot \mathbf{e}_i \tag{26}$$

where $p = 49.7$ and 107.60 for QARMA-64 and QARMA-128, respectively. An example of this distinguisher has been shown in Fig. 5, being differentiated from the 6-round distinguisher in red. In the next subsection, we use these 4-round paths as the underlying distinguisher for the proposed key recovery attacks.

6.2 Key Recovery Attack on QARMA-64

We first present the main attack procedure on QARMA-64, by which its key space is reduced by a factor of about 2^{12}. Then, we use it repeatedly to realize a full key recovery attack, satisfying the $\mathcal{TD}$ trade-off threshold.

6.2.1 Reducing the Key Space

We use an equivalent representation of QARMA-64, in which the AddRoundTweakey layer of all rounds, except for the first and last rounds, are replaced by an equivalent key

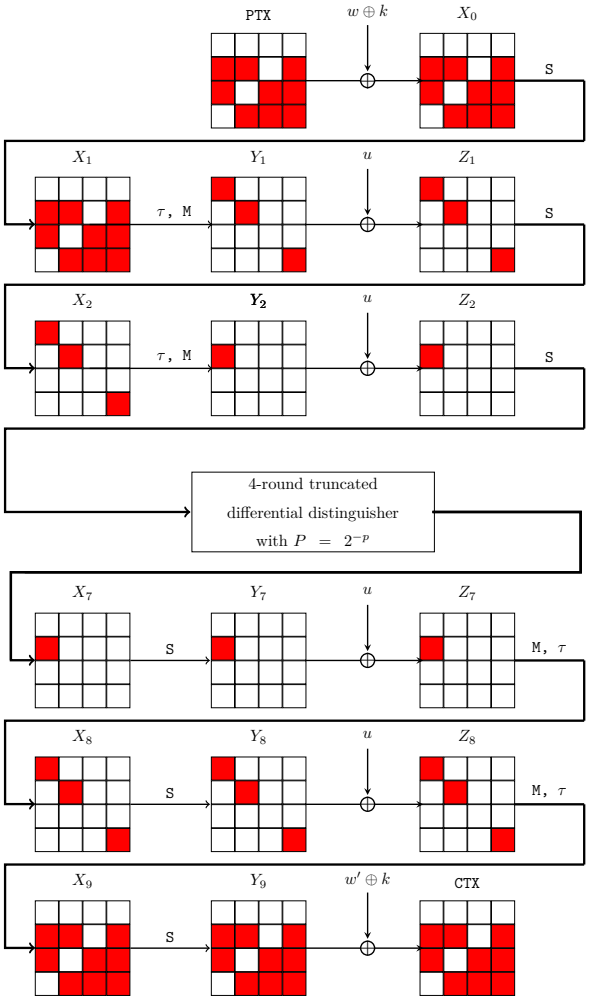

Figure 6: 10-round truncated differential attack on QARMA-64

$u = MC(\tau(k))$ XORed in after (before) the MixColumn layer in the forward (backward) round functions. We use the 4-round distinguisher given in (26) of probability $2^{-49.7}$ for $i = 4$, which is shown in Fig. 5, omitting the first and last rounds of the 6-round path. We extend this distinguisher for three rounds in each direction, resulting in a 10-round attack. This attack is shown in Fig. 6. For simplicity in this figure, we have omitted the tweaks, the constants $\mathfrak{c}_i$ and also α.

The propagation of active nibbles in the upper and lower parts is exactly the same. This causes all subkeys k or u involved in the attack to be the same in the upper and lower parts. The resulting differences in plaintext and ciphertext are active over the same nibbles, and finally $d_{in} = d_{out} = 36$.

Precomputation Phase. We will use the linear relations in the subkey bits to compute a precomputation table to reduce the time complexity of the attack by reducing the number of subkey bits guessed during the attack steps.

Thanks to the key schedule and the linear description of subkey bits involved in the attack given in Appendix D, we have the following linear relations between the subkey

bits of $(w \oplus k)$, u and $(w' \oplus k)$:

$$(w' \oplus k)[4]_3 + (w' \oplus k)[11]_2 + (w' \oplus k)[14]_2 = (w \oplus k)[4]_2 + (w \oplus k)[11]_1 + (w \oplus k)[14]_1 + u[5]_{1,4},$$

$$(w' \oplus k)[4]_4 + (w' \oplus k)[11]_3 + (w' \oplus k)[14]_3 = (w \oplus k)[4]_3 + (w \oplus k)[11]_2 + (w \oplus k)[14]_2 + u[5]_{1,2},$$

$$(w' \oplus k)[4]_{2,3,4} + (w' \oplus k)[11]_4 + (w' \oplus k)[14]_4 = (w \oplus k)[4]_{1,2,3} + (w \oplus k)[11]_3 + (w \oplus k)[14]_3 + u[5]_{2,3}.$$

Thus if we guess the 2^{12} possible values of $(w \oplus k)[4,11,14]$, the 2^4 possible values of $u[5]$, the 2^4 possible values of $(w' \oplus k)[4]$, and the 2^{24} possible values of $(C, C')[4, 11, 14]$, we would be able to compute $(w' \oplus k)[11]$ and $(w' \oplus k)[14]$, according to Step 1 of the attack below. Then, the three linear relations between the subkey bits apply a 3-bit filter on $(w' \oplus k)[4]$ and only two possible values of $(w' \oplus k)[4]$ remain for each triplet of $((w \oplus k)[4, 11, 14]$, $u[5]$, $(C, C')[4, 11, 14])$. We can now compute a precomputation table of the values of $(w' \oplus k)[4]$ for each possible $((w \oplus k)[4, 11, 14]$, $u[5]$, $(C, C')[4, 11, 14])$. Thus we have a size 2^{41} precomputation table of the 2 possible values of $(w' \oplus k)[4]$ for each of the 2^{40} possible values of $((w \oplus k)[4, 11, 14]$, $u[5]$, $(C, C')[4, 11, 14])$.

Similarly, we can compute the same precomputation table for the values of $(w' \oplus k)[5]$ and $(w' \oplus k)[7]$.

Generating Pairs. We follow the process discussed in Sec. 3 to accurately determine the data required for the attack. Each structure contains 2^{36} plaintexts, which are constant in the 7 non-active nibbles {0,1,2,3,6,9,12}, and take all possible values in the other nibbles. Since none of the differential pairs should share similar values in the active nibbles, the total number of pairs in each structure is $\frac{1}{2}(2^4(2^4-1))^9 = 2^{70.16}$. According to the differential branch table of M_4, that can be found in Appendix B, the filtering probability is $P_{Filt} = (2^{-7.81})^4 = 2^{-31.25}$, because of the three column transitions in the first `MixColumns`, and the one in the second. It must be held that $2^{s+70.16} = 2^{49.7+31.25}$ which gives $s = 10.8$. Therefore, the data required for the attack is $2^{s+36} = 2^{46.8}$. Finally, the probability of sieving the ciphertext pairs is $P_{sieve} = (2^{-4})^7 \times (\frac{15}{16})^9 = 2^{-28.84}$, and the total number of pairs after sieving is $2^{s+70.16-28.84} = 2^{52.12}$.

Attack Steps. For each of the $2^{52.12}$ candidate pairs, in order to verify which keys would allow to follow the differential path, the following steps are performed:

1. We first guess the nibble $(w \oplus k)[4]$ which implies the pair of values in $X_1[4]$. The `MixColumns` transition to column two in Y_1 implies that $\Delta X_1[11] = \Delta X_1[14] = \rho^{-1}\Delta X_1[4]$. On the other hand, we know the differences in nibbles $X_0[11]$ and $X_0[14]$ in the input of the S-box as they are given by the plaintext.

   There is a 2^{-p_n} probability of having a possible transition through the DDT and for each transition 2^{p_n} values make it possible. Thus, on average we associate one value of the nibbles $(w \oplus k)[11,14]$, per pair and per guess of $(w \oplus k)[4]$. The time complexity of this step is $2^{52.12} \times 2^4 = 2^{56.12}$.

2. We guess the nibble $u[5]$ to be able to use the precomputation table. Since we have P and P', we already have the needed bits of the ciphertexts, i.e. $(C, C')[4,11,14]$. Thus we can read on the precomputation table the 2 possible values of $(w' \oplus k)[4]$ and compute as in step 1, the value of $(w' \oplus k)[11]$ and $(w' \oplus k)[14]$. The time complexity after this step is $2^{56.12} \times 2^4 \times 2 = 2^{61.12}$.

3. Since we guessed $u[5]$, we can compute the pairs of values in $Y_8[5]$, and consequently $X_8[5]$. Due to the linear relation imposed by the `MixColumns` matrix, it holds that $\Delta Z_7[4] = \rho^{-1}\Delta X_8[5]$. Thanks to the reflective structure of the cipher, the same conditions apply to the upper part and we can compute $\Delta Z_2[4] = \rho^{-1}\Delta X_2[5]$.

4. We repeat Steps 1,2 and 3 with $((w' \oplus k)[5], u[0])$ and $((w' \oplus k)[7], u[15])$. The time complexity is now $3 \times 2^{61.12} = 2^{62.72}$.

5. We now have to match the three different candidates subkey bits we computed. For this, we will merge the list of the 2^9 differences $(\Delta_1 Z_2[4], \Delta_1 Z_7[4])$ computed in Step 3 using $u[5]$, the list of the 2^9 differences $(\Delta_2 Z_2[4], \Delta_2 Z_7[4])$ computed in Step 3 using $u[0]$ and the list of the 2^9 differences $(\Delta_3 Z_2[4], \Delta_3 Z_7[4])$ computed in Step 3 using $u[15]$.

   There is 2^8 possible values for the differences $(\Delta_i Z_2[4], \Delta_i Z_7[4])$, $i = 1, 2, 3$, therefore for each of the 2^9 possible values of $((w \oplus k)[4,11,14], u[5], (w' \oplus k)[4])$, there is $\frac{2^9}{2^8} = 2$ values of $((w \oplus k)[5,10,15], u[0], (w' \oplus k)[5,10,15])$ which will match. Similarly, for each of the two values of $((w \oplus k)[5,10,15], u[0], (w' \oplus k)[5,10,15])$, there will be two matches in the list of differences $(\Delta_3 Z_2[4], \Delta_3 Z_7[4])$ thus there will be 2 values of $((w \oplus k)[7,8,13], u[15], (w' \oplus k)[7,8,13])$ for each of the 2 values of $((w \oplus k)[5,10,15], u[0], (w' \oplus k)[5,10,15])$.

   Thus for each of the 2^9 guesses of $((w \oplus k)[4,11,14], u[5], (w' \oplus k)[4,11,14])$ we match two sets of subkey bits candidates $((w \oplus k)[5,10,15], u[0], (w' \oplus k)[5,10,15])$ and for each of those two sets we match two sets of subkey bits candidates of $((w \oplus k)[7,8,13], u[15], (w' \oplus k)[7,8,13])$. So overall, we get $2^{52.12} \times 2^9 \times 2 \times 2 = 2^{63.12}$ candidate triplets $(P, P', \text{information bits of key})$.

The information bits of subkeys involved in the attack have linear representations in key bits w and k, which are shown in Tab. 6 of Appendix D. The resulting linear system of equations has 84 equations (each corresponds to a guessed/implied subkey bit in the attack) in 77 variables (w and k bits), and its rank is 75. This means that the remaining triplets give $2^{63.12}$ possible values for 75 key bits, hence reducing the space for about 11.88 bits.

6.2.2 Recovering the Whole Key

If we repeat the attack steps once again with a new set of data, the data and time complexity will increase by a factor of 2, each. But, the key space is reduced by a factor of $2^{11.88}$. So, the remaining candidate keys will become $2^{63.12-11.88} = 2^{51.24}$.

In general, the time complexity of repeating the attack for N times is $2^{63.12} \times N$, and the complexity of the exhaustive search of the remaining key bits is $2^{128-11.88N}$, so the time complexity would be $2^{128-11.88N} + 2^{63.12} \times N$, which is minimized at $N = 6$. All in all, the time, data, and memory complexities of the attack are:

$$\begin{aligned} \mathcal{D} &= 6 \times 2^{46.8} = 2^{49.39} \\ \mathcal{T} &= 2^{63.12} \times 6 + 2^{128-11.88\times 6} = 2^{65.72} \quad (27) \\ \mathcal{M} &= 2^{63.12} \\ \mathcal{TD} &= 2^{115.11} < 2^{128}. \quad (28) \end{aligned}$$

6.3 Key Recovery Attack on QARMA-128

The attack on QARMA-128 shares many similarities with QARMA-64 and uses the same 4-round pattern for truncated distinguisher with probability $2^{-107.60}$. Therefore, in this section, we will focus on highlighting the distinctions between them to avoid repeating the details already discussed.

Precomputation Phase. For QARMA-128, we also have linear relations in the subkey bits which are used in the precomputation phase of the attack. Based on the linear description of subkey bits involved in the attack given in Appendix D, we have the following linear relations between the subkey bits of $(w \oplus k), u$ and $(w' \oplus k)$:

$$(w'\oplus k)[4]_2+(w'\oplus k)[11]_3+(w'\oplus k)[14]_7 = (w\oplus k)[4]_1+(w\oplus k)[11]_2+(w\oplus k)[14]_6+u[5]_{5,6}$$
$$(w'\oplus k)[4]_3+(w'\oplus k)[11]_4+(w'\oplus k)[14]_8 = (w\oplus k)[4]_2+(w\oplus k)[11]_3+(w\oplus k)[14]_7+u[5]_{6,7}$$
$$(w'\oplus k)[4]_4+(w'\oplus k)[11]_5+(w'\oplus k)[14]_{2\ldots8} = (w\oplus k)[4]_3+(w\oplus k)[11]_4+(w\oplus k)[14]_{1\ldots7}+u[5]_{7,8}$$
$$(w'\oplus k)[4]_5+(w'\oplus k)[11]_6+(w'\oplus k)[14]_2 = (w\oplus k)[4]_4+(w\oplus k)[11]_5+(w\oplus k)[14]_1+u[5]_{1,8}$$
$$(w'\oplus k)[4]_6+(w'\oplus k)[11]_7+(w'\oplus k)[14]_3 = (w\oplus k)[4]_5+(w\oplus k)[11]_6+(w\oplus k)[14]_2+u[5]_{1,2}$$
$$(w'\oplus k)[4]_7+(w'\oplus k)[11]_8+(w'\oplus k)[14]_4 = (w\oplus k)[4]_6+(w\oplus k)[11]_7+(w\oplus k)[14]_3+u[5]_{2,3}$$
$$(w'\oplus k)[4]_8+(w'\oplus k)[11]_{2\ldots8}+(w'\oplus k)[14]_5 = (w\oplus k)[4]_7+(w\oplus k)[11]_{1\ldots7}+(w\oplus k)[14]_4+u[5]_{3,4}$$

Thus if we guess the 2^{24} possible values of $(w \oplus k)$[4,11,14], the 2^8 possible values of u[5], the 2^8 possible values of $(w' \oplus k)$[4], and the 2^{48} possible values of $(C, C')[4, 11, 14]$, we would be able to compute $(w' \oplus k)$[11] and $(w' \oplus k)$[14], as in Step 1 of the attack. Then, the seven linear relations between the subkey bits apply a 7-bit filter on $(w' \oplus k)[4]$ and only two possible values of $(w' \oplus k)[4]$ remain for each triplet of $((w \oplus k)[4, 11, 14]$, $u[5]$, $(C, C')[4, 11, 14])$. We can now compute a precomputation table of the values of $(w' \oplus k)[4]$ for each possible $((w \oplus k)[4, 11, 14]$, $u[5]$, $(C, C')[4, 11, 14])$. Thus we have a size 2^{81} precomputation table of the 2 possible values of $(w' \oplus k)[4]$ for each of the 2^{80} possible values of $((w \oplus k)[4, 11, 14]$, $u[5]$, $(C, C')[4, 11, 14])$.

Similarly, we can compute the same precomputation table for the values of $(w' \oplus k)[5]$ and $(w' \oplus k)[7]$.

Generating Pairs. Each structure contains $2^{9m} = 2^{72}$ plaintexts, which are constant in 7 non-active words, taking all possible values in the other words. The total number of pairs in each structure is $\frac{1}{2}(2^m(2^m - 1))^9 = 2^{142.94}$. The filtering probability is $P_{Filt} = (2^{-15.99})^4 = 2^{-63.96}$. It must be held that $2^{s+142.94} = 2^{107.60+63.96}$ which gives $s = 28.62$. Therefore, the data required for the attack is $2^{s+72} = 2^{100.62}$. Finally, the probability of sieving the ciphertext pairs is $P_{sieve} = (2^{-8})^7 \times (\frac{255}{256})^9 = 2^{-56.05}$, and the total number of pairs after sieving is $2^{s+142.94-56.05} = 2^{115.51}$.

Attack Steps. Since the attack steps is very similar to QARMA-64's, which are performed for each of the $2^{101.68}$ pairs of data. In the following, we just report the time complexity of each step.

1. The time complexity of this step is $2^{115.51} \times 2^8 = 2^{123.51}$.
2. The time complexity after this step is $2^{123.51} \times 2^8 \times 2 = 2^{132.51}$.
3. In this step, it should hold that $\Delta Z_7[4] = \rho^{-5}\Delta X_8[5]$ and $\Delta Z_2[4] = \rho^{-5}\Delta X_2[5]$.
4. After repeating Steps 1,2 and 3 with $((w' \oplus k)[5], u[0])$ and $((w' \oplus k)[7], u[15])$, the time complexity is now $3 \times 2^{132.51} = 2^{134.10}$.
5. The three different candidate subkey bits should be matched by merging the list of the 2^{17} differences $(\Delta_i Z_2[4], \Delta_i Z_7[4])$, $1 \le i \le 3$, computed in Step 3 using $u[5]$, $u[0]$, and $u[15]$.

   Thus for each of the 2^{17} guesses of $((w \oplus k)[4,11,14], u[5], (w' \oplus k)[4,11,14])$ we match two sets of subkey bits candidates $((w \oplus k)[5,10,15], u[0], (w' \oplus k)[5,10,15])$ and for each of those two sets we match two sets of subkey bits candidates of $((w \oplus k)[7,8,13], u[15], (w' \oplus k)[7,8,13])$. So overall, we get $2^{115.51} \times 2^{17} \times 2 \times 2 = 2^{134.51}$ candidate triplets $(P, P', \text{information bits of key})$.

The linear representations of the information bits of subkeys involved in the attack in key bits w and k are shown in Tab. 7 of Appendix D. The resulting linear system of equations has 168 equations in 149 variables, with rank 147. This means that the remaining triplets give $2^{134.51}$ possible values for 147 key bits, hence reducing the space for about 12.49 bits.

6.3.1 Recovering the Whole Key

The time complexity of repeating the attack N times is $2^{134.51} \times N$, and the complexity of the exhaustive search of the remaining key bits is $2^{256-12.49N}$, so the time complexity would be $2^{256-12.49N} + 2^{134.51} \times N$, which is minimized at $N = 10$. All in all, the time, data, and memory complexities of the attack are:

$$\begin{aligned} \mathcal{D} &= 10 \times 2^{100.62} = 2^{103.95} \\ \mathcal{T} &= 2^{256-12.49\times 10} + 2^{134.51} \times 10 = 2^{137.84} \\ \mathcal{M} &= 2^{134.51} \\ \mathcal{TD} &= 2^{241.79} < 2^{256}. \end{aligned} \tag{29}$$

7 Conclusion

We have generalized and provided some new insight on truncated differential attacks, and we hope these results will be usefull for future research. We have analyszed the `QARMA` [17] block cipher proposed in 2017, with respect to truncated differential attacks and have been able to propose the best known attacks on this cipher, that reach up to 10 rounds and break the security claims of these reduced versions proposed by the designers (unlike the previous known attacks on 10-round `QARMA` that had a complexity higher than the security claims). For reaching this attacks we provide some new truncated distinguishers and an improved and dedicated key-recovery part, based on list merging techniques and precomputation, that allows to greatly reduce the time complexity.

Acknowledgment

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 714294 - acronym QUASYModo).

References

[1] L. R. Knudsen, "Truncated and higher order differentials," in *Fast Software Encryption: Second International Workshop Leuven, Belgium, December 14–16, 1994 Proceedings 2*, pp. 196–211, Springer, 1995.

[2] V. Lallemand and M. Naya-Plasencia, "Cryptanalysis of klein," in *International Workshop on Fast Software Encryption*, pp. 451–470, Springer, 2014.

[3] S. Rasoolzadeh, Z. Ahmadian, M. Salmasizadeh, and M. R. Aref, "An improved truncated differential cryptanalysis of klein," *Tatra Mountains Mathematical Publications*, vol. 67, no. 1, pp. 135–147, 2016.

[4] L. Li, K. Jia, X. Wang, and X. Dong, "Meet-in-the-middle technique for truncated differential and its applications to clefia and camellia," in *International Workshop on Fast Software Encryption*, pp. 48–70, Springer, 2015.

[5] A. Abdelkhalek, Y. Sasaki, Y. Todo, M. Tolba, and A. M. Youssef, "Milp modeling for (large) s-boxes to optimize probability of differential characteristics," *IACR Transactions on Symmetric Cryptology*, pp. 99–129, 2017.

[6] Y. Sasaki and Y. Todo, "New impossible differential search tool from design and cryptanalysis aspects: Revealing structural properties of several ciphers," in *Advances in Cryptology–EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30–May 4, 2017, Proceedings, Part III 36*, pp. 185–215, Springer, 2017.

[7] A. Ebrahimi Moghaddam and Z. Ahmadian, "New automatic search method for truncated-differential characteristics application to midori, skinny and craft," *The Computer Journal*, vol. 63, no. 12, pp. 1813–1825, 2020.

[8] M. Eichlseder, G. Leander, and S. Rasoolzadeh, "Computing expected differential probability of (truncated) differentials and expected linear potential of (multidimensional) linear hulls in spn block ciphers," in *Progress in Cryptology–INDOCRYPT 2020: 21st International Conference on Cryptology in India, Bangalore, India, December 13–16, 2020, Proceedings 21*, pp. 345–369, Springer, 2020.

[9] H. Guo, Z. Zhang, Q. Yang, L. Hu, and Y. Luo, "A new method to find all the high-probability word-oriented truncated differentials: Application to midori, skinny and craft," *The Computer Journal*, vol. 66, no. 5, pp. 1069–1082, 2023.

[10] X. Xie and T. Tian, "The triangle differential cryptanalysis," in *Australasian Conference on Information Security and Privacy*, pp. 72–88, Springer, 2023.

[11] X. Xie and T. Tian, "Structural evaluation of aes-like ciphers against mixture differential cryptanalysis," *Designs, Codes and Cryptography*, pp. 1–19, 2023.

[12] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, *et al.*, "Prince–a low-latency block cipher for pervasive computing applications," in *Advances in Cryptology–ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security*, pp. 208–225, Springer, 2012.

[13] H. Soleimany, C. Blondeau, X. Yu, W. Wu, K. Nyberg, H. Zhang, L. Zhang, and Y. Wang, "Reflection cryptanalysis of prince-like ciphers," *Journal of Cryptology*, vol. 28, pp. 718–744, 2015.

[14] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim, "The skinny family of block ciphers and its low-latency variant mantis," in *Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference*, pp. 123–153, Springer, 2016.

[15] S. Banik, A. Bogdanov, T. Isobe, K. Shibutani, H. Hiwatari, T. Akishita, and F. Regazzoni, "Midori: A block cipher for low energy," in *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 21*, pp. 411–436, Springer, 2015.

[16] C. Dobraunig, M. Eichlseder, D. Kales, and F. Mendel, "Practical key-recovery attack on mantis5," *IACR Transactions on Symmetric Cryptology*, pp. 248–260, 2016.

[17] R. Avanzi, "The qarma block cipher family. almost mds matrices over rings with zero divisors, nearly symmetric even-mansour constructions with non-involutory central rounds, and search heuristics for low-latency s-boxes," *IACR Transactions on Symmetric Cryptology*, pp. 4–44, 2017.

[18] S. Even and Y. Mansour, "A construction of a cipher from a single pseudorandom permutation," *Journal of cryptology*, vol. 10, pp. 151–161, 1997.

[19] I. Dinur, "Cryptanalytic time-memory-data tradeoffs for fx-constructions with applications to prince and pride," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 231–253, Springer, 2015.

[20] S. Rasoolzadeh and H. Raddum, "Cryptanalysis of prince with minimal data," in *Progress in Cryptology–AFRICACRYPT 2016: 8th International Conference on Cryptology in Africa, Fes, Morocco, April 13-15, 2016, Proceedings 8*, pp. 109–126, Springer, 2016.

[21] R. Zong and X. Dong, "Meet-in-the-middle attack on qarma block cipher," *Cryptology ePrint Archive*, 2016.

[22] R. Li and C. Jin, "Meet-in-the-middle attacks on reduced-round qarma-64/128," *The Computer Journal*, vol. 61, no. 8, pp. 1158–1165, 2018.

[23] M. Li, K. Hu, and M. Wang, "Related-tweak statistical saturation cryptanalysis and its application on qarma," *Cryptology ePrint Archive*, 2019.

[24] Y. Liu, T. Zang, D. Gu, F. Zhao, W. Li, and Z. Liu, "Improved cryptanalysis of reduced-version qarma-64/128," *IEEE Access*, vol. 8, pp. 8361–8370, 2020.

[25] J. Du, W. Wang, M. Li, and M. Wang, "Related-tweakey impossible differential attack on qarma-128," *Science China Information Sciences*, vol. 65, no. 2, p. 129102, 2022.

[26] N. Mouha, Q. Wang, D. Gu, and B. Preneel, "Differential and linear cryptanalysis using mixed-integer linear programming," in *Information Security and Cryptology: 7th International Conference, Inscrypt 2011, Beijing, China, November 30–December 3, 2011. Revised Selected Papers 7*, pp. 57–76, Springer, 2012.

[27] X. Lai, J. L. Massey, and S. Murphy, "Markov ciphers and differential cryptanalysis," in *Advances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques Brighton, UK, April 8–11, 1991 Proceedings 10*, pp. 17–38, Springer, 1991.

[28] E. Biham and A. Shamir, "Differential cryptanalysis of des-like cryptosystems," *Journal of CRYPTOLOGY*, vol. 4, pp. 3–72, 1991.

[29] C. Boura, N. David, R. Heim Boissier, and M. Naya-Plasencia, "Better steady than speedy: full break of speedy-7-192," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 36–66, Springer, 2023.

[30] M. Eichlseder and D. Kales, "Clustering related-tweak characteristics: application to mantis-6," *IACR Transactions on Symmetric Cryptology*, pp. 111–132, 2018.

[31] R. Zong and X. Dong, "Milp-aided related-tweak/key impossible differential attack and its applications to qarma, joltik-bc," *IEEE Access*, vol. 7, pp. 153683–153693, 2019.

[32] S. Banerjee and A. Roy, *Linear algebra and matrix analysis for statistics*. Crc Press, 2014.

A Proof of Lemma 3

Let $N(\mathbf{a},\mathbf{b}) = \left|\{(\mathbf{x},\mathbf{y}) \in \mathbb{F}_{2^m}^t \times \mathbb{F}_{2^m}^t | \mathbf{y} = M\mathbf{x}, \mathsf{Tr}(\mathbf{x}) = \mathbf{a}, \mathsf{Tr}(\mathbf{y}) = \mathbf{b}\}\right|$. Since M is an MDS matrix, $N(\mathbf{a},\mathbf{b}) = 0$ for $1 \leq \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) \leq t$ and $N(\mathbf{a},\mathbf{b}) = 1$ for $\mathbf{a} = \mathbf{b} = 0$. Therefore, we have

$$\Pr(\mathbf{a} \to \mathbf{b}) = \begin{cases} 1 & \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) = 0 \\ 0 & 1 \leq \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) \leq t \\ \frac{N(\mathbf{a},\mathbf{b})}{(2^m-1)^{\mathsf{Hw}(\mathbf{a})}} & \mathsf{Hw}(\mathbf{a}) + \mathsf{Hw}(\mathbf{b}) \geq t+1 \end{cases}. \tag{30}$$

We first provide a recursive form for $N(\mathbf{a},\mathbf{b})$, then show how (30) approximately equals (18). Let $Q(\mathbf{a},\mathbf{b}) = \left|\{(\mathbf{x},\mathbf{y}) \in \mathbb{F}_{2^m}^t \times \mathbb{F}_{2^m}^t | \mathbf{y} = M\mathbf{x}, \mathsf{Tr}(\mathbf{x}) \preceq \mathbf{a}, \mathsf{Tr}(\mathbf{y}) \preceq \mathbf{b}, \mathbf{x} \neq \mathbf{0}\}\right|$, where $\mathsf{Tr}(\mathbf{x}) \preceq \mathbf{a}$ means that $\mathsf{Tr}(\mathbf{x})$ is component-wise less than or equal to $\mathbf{a}$. According to the definitions of N and Q, it holds that:

$$N(\mathbf{a},\mathbf{b}) = Q(\mathbf{a},\mathbf{b}) - \sum_{\substack{\mathbf{0} \prec \mathbf{a}' \prec \mathbf{a}, \\ \mathbf{0} \prec \mathbf{b}' \prec \mathbf{b}}} N(\mathbf{a}',\mathbf{b}') \tag{31}$$

It is more convenient to first compute $Q(\mathbf{a},\mathbf{b})$, then $N(\mathbf{a},\mathbf{b})$ according to (31). Let $\mathsf{Hw}(\mathbf{a}) = i$ and $\mathsf{Hw}(\mathbf{b}) = j$. Let $\mathcal{I}$ and $\mathcal{J}$ represent the index sets of zero components in $\mathbf{a}$ and $\mathbf{b}$, respectively. So, $|\mathcal{I}| = t - i$ and $|\mathcal{J}| = t - j$, and $\mathsf{Tr}(\mathbf{x}) \preceq \mathbf{a}$ implies $x_r = 0$ if $r \in \mathcal{I}$, and $\mathsf{Tr}(\mathbf{y}) \preceq \mathbf{b}$ implies $y_r = 0$ if $r \in \mathcal{J}$. All these constraints simplifies $Q(\mathbf{a},\mathbf{b})$ into $Q(\mathbf{a},\mathbf{b}) = \left|\{\mathbf{x}' \in \mathbb{F}_2^i | M'\mathbf{x}' = \mathbf{0}, \mathbf{x}' \neq \mathbf{0}\}\right|$, where $M' = M[\mathcal{J},\mathcal{I}^c]$, i.e. M' is a submatrix of M with dimensions $t - j \times i$, obtained by retaining rows in $\mathcal{J}$, and removing columns in $\mathcal{I}$.

So, $Q(\mathbf{a},\mathbf{b})$ would be the number of non-zero solutions for the homogeneous system of linear equations $M'\mathbf{x}' = \mathbf{0}$ over $\mathbb{F}_{2^m}$. Since M is an MDS matrix, any submatrix of that, including M' is non-singular. Therefore, according to Rouché–Capelli Theorem [32], it has a number of $2^{m(i-(t-j))} = 2^{m(i+j-t)}$ solutions, excluding $\mathbf{x}' = 0$, it would be

$$Q(\mathbf{a},\mathbf{b}) = 2^{m(i+j-t)} - 1 \tag{32}$$

According to (32), $Q(\mathbf{a},\mathbf{b})$ only depends on i and j, so for simplicity we can change the arguments $(\mathbf{a},\mathbf{b})$ of N and Q into (i,j), and (31) will be updated to

$$N(i,j) = 2^{m(i+j-t)} - 1 - \sum_{t+1 \leq i'+j' < i+j} \binom{i}{i'} \binom{j}{j'} N(i',j') \tag{33}$$

While (33) does not provide a direct formula for $N(i,j)$, it meets our needs adequately. Note that the terms within the summation, $N(i',j')$, are of order at most $2^{m(i+j-t-1)}$, much smaller than the first term $2^{m(i+j-t)}$, specifically for large m. Hence, we can approximate $N(i,j)$ by its dominant term, which is $2^{m(i+j-t)}$. by replacing $N(i,j) \approx 2^{m(i+j-t)}$ into (30), it holds that

$$\Pr(\mathbf{a} \to \mathbf{b}) \approx \frac{2^{m(i+j-t)}}{(2^m-1)^i} \approx 2^{-m(t-j)}, \tag{34}$$

which completes the proof.

B Differential Branch Tables

Definition 6 (Differential Branch Table (DBT))**.** For matrix $M_{t\times t}$ over $\mathbb{F}_{2^m}$, the Differential Branch Table is defined as a $2^t \times 2^t$ table whose entry $(\mathbf{a},\mathbf{b})$ reflects the base-2 logarithm of $P(\mathbf{a} \xrightarrow{M} \mathbf{b})$:

$$DBT(\mathbf{a},\mathbf{b}) = \log_2(\Pr_{\mathbf{x}}\{\mathsf{Tr}(M \cdot \mathbf{x}) = \mathbf{b} | \mathsf{Tr}(\mathbf{x}) = \mathbf{a}\}) \tag{35}$$

where $\mathbf{a} = [a_3, a_2, a_1, a_0]^\top, \mathbf{b} = [b_3, b_2, b_1, b_0]^\top \in \mathbb{F}_t^2$, and $\mathsf{Tr}(\cdot)$ is the m-bit truncation operator. The impossible transitions are shown by a "-".

Tables 2 and 3 show the Differential Branch Table (DBT) for QARMA-64/128 MixColumns Matrix $M(= Q)$. Table 4 includes the accurate and approximated DBT for all $t \times t$ MDS matrices for $t = 4$ and $m = 4, 8$. Since, for accurate MDS matrices the transition probabilities depend only on $\mathsf{Hw}(\mathbf{a})$ and $\mathsf{Hw}(\mathbf{b})$, the DBT reduces to a $(t+1) \times (t+1)$ table.

Table 2: DBT for QARMA-64 MixColumns Matrix M_4

a / b	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-
0x2	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-
0x3	-	-	-	-6.229	-	-	-	-4.229	-	-	-	-4.229	-	-	-	-0.184
0x4	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-
0x5	-	-	-	-	-	-3.907	-	-	-	-	-	-	-	-	-	-0.099
0x6	-	-	-	-	-	-	-6.229	-4.229	-	-	-	-	-	-	-4.229	-0.184
0x7	-	-	-	-8.135	-	-	-8.135	-4.181	-7.814	-	-	-4.091	-	-4.006	-4.091	-0.408
0x8	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-
0x9	-	-	-	-	-	-	-	-	-	-6.229	-	-4.229	-	-4.229	-	-0.184
0xa	-	-	-	-	-	-	-	-	-	-	-3.907	-	-	-	-	-0.099
0xb	-	-	-	-8.135	-7.814	-	-	-4.091	-	-8.135	-	-4.181	-	-4.091	-4.006	-0.408
0xc	-	-	-	-	-	-	-	-	-	-	-	-	-6.229	-4.229	-4.229	-0.184
0xd	-	-	-7.814	-	-	-	-	-4.006	-	-8.135	-	-4.091	-8.135	-4.181	-4.091	-0.408
0xe	-	-7.814	-	-	-	-	-8.135	-4.091	-	-	-	-4.006	-8.135	-4.091	-4.181	-0.408
0xf	-	-	-	-7.998	-	-7.913	-7.998	-4.314	-	-7.998	-7.913	-4.314	-7.998	-4.314	-4.314	-0.368

Table 3: DBT for QARMA-128 MixColumns Matrix M_8

a / b	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-
0x2	-	-	-	-	-	-	-	-	-	-	-	-	-	0	-	-
0x3	-	-	-	-14.404	-	-	-	-8.011	-	-	-	-8.011	-	-	-	-0.011
0x4	-	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-
0x5	-	-	-	-	-	-7.994	-	-	-	-	-	-	-	-	-	-0.005
0x6	-	-	-	-	-	-	-14.404	-8.011	-	-	-	-	-	-	-8.011	-0.011
0x7	-	-	-	-16.007	-	-	-16.007	-8.011	-15.99	-	-	-8.06	-	-8	-8.006	-0.022
0x8	-	-	-	-	-	-	-	0	-	-	-	-	-	-	-	-
0x9	-	-	-	-	-	-	-	-	-	-14.404	-	-8.011	-	-8.011	-	-0.011
0xa	-	-	-	-	-	-	-	-	-	-	-7.99	-	-	-	-	-0.005
0xb	-	-	-	-16.007	-15.99	-	-	-8.006	-	-16.007	-	-8.011	-	-8.006	-8	-0.022
0xc	-	-	-	-	-	-	-	-	-	-	-	-	-14.404	-8.011	-8.011	-0.011
0xd	-	-	-15.99	-	-	-	-	-8	-	-16.007	-	-8.006	-16.007	-8.011	-8.006	-0.022
0xe	-	-15.99	-	-	-	-	-16.007	-8.006	-	-	-	-8	-16.007	-8.006	-8.011	-0.022
0xf	-	-	-	-16	-	-15.995	-16	-8.017	-	-16	-15.995	-8.017	-16	-8.017	-8.017	-0.022

Table 4: Accurate (left) and approximated (right) DBTs for a 4×4 MDS Matrix for $m = 4$ and $m = 8$, $i = \mathsf{Hw}(\mathbf{a})$ and $j = \mathsf{Hw}(\mathbf{b})$.

$m = 4$, accurate

i/j	0	1	2	3	4
0	0	-	-	-	-
1	-	-	-	-	0
2	-	-	-	-3.907	-0.447
3	-	-	-7.814	-4.354	-0.363
4	-	-11.721	-8.261	-4.269	-0.374

$m = 4$, approximated

i/j	0	1	2	3	4
0	0	-	-	-	-
1	-	-	-	-	0
2	-	-	-	-4	0
3	-	-	-8	-4	0
4	-	-12	-8	-4	0

$m = 8$, accurate

i/j	0	1	2	3	4
0	0	-	-	-	-
1	-	-	-	-	0
2	-	-	-	-7.994	-0.023
3	-	-	-15.989	-8.017	-0.023
4	-	-23.98	-16.01	-8.017	-0.023

$m = 8$, approximated

i/j	0	1	2	3	4
0	0	-	-	-	-
1	-	-	-	-	0
2	-	-	-	-8	0
3	-	-	-16	-8	0
4	-	-24	-16	-8	0

C Profile of Concrete paths in terms of Probability

Table 5: Number of concrete differential paths $N(\tilde{p})$ of probability $\tilde{p}$ consistent with the 6-round truncated path given in (24), for QARMA-64 (left) and QARMA-128 (right)

$\tilde{p}$	$N(\tilde{p})$	$\frac{1}{\lvert\Delta_{in}\rvert} \log_2(\sum_{i \leq \tilde{p}} 2^{-i} N(i))$
55	2	-57.9068
56	20	-55.3218
57	44	-54.3832
58	189	-53.3637
59	512	-52.6098
60	1624	-51.8913
61	3784	-51.3483
62	9943	-50.8505
63	21134	-50.4608
64	40528	-50.1657
65	66060	-49.9631
66	96836	-49.8308
67	119046	-49.7551
68	123644	-49.7173
69	99734	-49.7063
70	64261	-49.6967
71	29406	-49.6965
72	10514	-49.6963

$\tilde{p}$	$N(\tilde{p})$	$\frac{1}{\lvert\Delta_{in}\rvert} \log_2(\sum_{i \leq \tilde{p}} 2^{-i} N(i))$
109	2	-115.9943
110	5	-114.8243
111	9	-114.2394
112	38	-113.4707
113	155	-112.5891
114	688	-11.5785
115	2479	-110.5740
116	10174	-109.6268
117	38190	-108.6982
118	133167	-108.0414
119	75479	-107.8413
120	91853	-107.7319
121	123217	-107.6629
122	122651	-107.6298
123	120829	-107.6137
124	118665	-107.6103
125	112937	-107.6022
126	114523	-107.6003
127, . . . , 144	2268082	-107.6003

D Linear description of Subkey bits

The information bits guessed/implied during the attack have linear representations in key bits w and k which are shown in Tab. 6 and Tab. 7. In this paper the bit indices are arranged according to the following patterns. For QARMA-64

$$\begin{pmatrix} \mathtt{x}_0 \dots \mathtt{x}_3 & \mathtt{x}_4 \dots \mathtt{x}_7 & \mathtt{x}_8 \dots \mathtt{x}_{11} & \mathtt{x}_{12} \dots \mathtt{x}_{15} \\ \mathtt{x}_{16} \dots \mathtt{x}_{19} & \mathtt{x}_{20} \dots \mathtt{x}_{23} & \mathtt{x}_{24} \dots \mathtt{x}_{27} & \mathtt{x}_{28} \dots \mathtt{x}_{31} \\ \mathtt{x}_{32} \dots \mathtt{x}_{35} & \mathtt{x}_{36} \dots \mathtt{x}_{39} & \mathtt{x}_{40} \dots \mathtt{x}_{43} & \mathtt{x}_{44} \dots \mathtt{x}_{47} \\ \mathtt{x}_{48} \dots \mathtt{x}_{51} & \mathtt{x}_{52} \dots \mathtt{x}_{55} & \mathtt{x}_{56} \dots \mathtt{x}_{59} & \mathtt{x}_{60} \dots \mathtt{x}_{63} \end{pmatrix}, \tag{36}$$

and for QARMA-128

$$\begin{pmatrix} \mathtt{x}_0 \dots \mathtt{x}_7 & \mathtt{x}_8 \dots \mathtt{x}_{15} & \mathtt{x}_{16} \dots \mathtt{x}_{23} & \mathtt{x}_{24} \dots \mathtt{x}_{31} \\ \mathtt{x}_{32} \dots \mathtt{x}_{39} & \mathtt{x}_{40} \dots \mathtt{x}_{47} & \mathtt{x}_{48} \dots \mathtt{x}_{55} & \mathtt{x}_{56} \dots \mathtt{x}_{63} \\ \mathtt{x}_{64} \dots \mathtt{x}_{71} & \mathtt{x}_{72} \dots \mathtt{x}_{79} & \mathtt{x}_{80} \dots \mathtt{x}_{87} & \mathtt{x}_{88} \dots \mathtt{x}_{95} \\ \mathtt{x}_{96} \dots \mathtt{x}_{103} & \mathtt{x}_{104} \dots \mathtt{x}_{11} & \mathtt{x}_{112} \dots \mathtt{x}_{119} & \mathtt{x}_{120} \dots \mathtt{x}_{127} \end{pmatrix}. \tag{37}$$

Table 6: QARMA-64 subkey linear description in w and k

guessed/ computed subkeys		linear description		guessed/ computed subkeys		linear description		guessed/ computed subkeys		linear description	
subkey	bit	w	k	subkey	bit	w	k	subkey	bit	w	k
	16	16	16		16	15	16				
	17	17	17		17	16	17		0	-	22,41,61
	18	18	18		18	17	18				
	19	19	19		19	18	19				
	20	20	20		20	19	20		1	-	23,42,62
	21	21	21		21	20	21				
	22	22	22		22	21	22				
	23	23	23		23	22	23		2	-	20,43,63
	28	28	28		28	27	28				
	29	29	29		29	28	29				
	30	30	30		30	29	30		3	-	21,40,60
	31	31	31		31	30	31				
	32	32	32		32	31	32				
	33	33	33		33	32	33		20	-	18,45,57
	34	34	34		34	33	34				
	35	35	35		35	34	35				
	40	40	40		40	39	40		21	-	19,46,58
$(w+k)$	41	41	41	$(w'+k)$	41	40	41	(u)			
	42	42	42		42	41	42				
	43	43	43		43	42	43		22	-	16,47,59
	44	44	44		44	43	44				
	45	45	45		45	44	45				
	46	46	46		46	45	46		23	-	17,44,56
	47	47	47		47	46	47				
	52	52	52		52	51	52				
	53	53	53		53	52	53		60	-	30,33,53
	54	54	54		54	53	54				
	55	55	55		55	54	55				
	56	56	56		56	55	56		61	-	31,34,54
	57	57	57		57	56	57				
	58	58	58		58	57	58				
	59	59	59		59	58	59		62	-	28,35,55
	60	60	60		60	59	60				
	61	61	61		61	60	61				
	62	62	62		62	61	62		63	-	29,32,52
	63	63	63		63	0,62	63				

Table 7: QARMA-128 subkey linear description in w and k

guessed/ computed subkeys		linear description		guessed/ computed subkeys		linear description		guessed/ computed subkeys		linear description		guessed/ computed subkeys		linear description		guessed/ computed subkeys		linear description	
subkey	bit	w	k	subkey	bit	w	k	subkey	bit	w	k	subkey	bit	w	k	subkey	bit	w	k
	32	32	32		84	84	84		32	31	32		84	83	84				
	33	33	33		85	85	85		33	32	33		85	84	85		0	-	44,81,125
	34	34	34		86	86	86		34	33	34		86	85	86		1	-	45,82,126
	35	35	35		87	87	87		35	34	35		87	86	87		2	-	46,83,127
	36	36	36		88	88	88		36	35	36		88	87	88		3	-	47,84,120
	37	37	37		89	89	89		37	36	37		89	88	89		4	-	40,85,121
	38	38	38		90	90	90		38	37	38		90	89	90		5	-	41,86,122
	39	39	39		91	91	91		39	38	39		91	90	91		6	-	42,87,123
	40	40	40		92	92	92		40	39	40		92	91	92		7	-	43,80,124
	41	41	41		93	93	93		41	40	41		93	92	93				
	42	42	42		94	94	94		42	41	42		94	93	94		40	-	36,93,113
	43	43	43		95	95	95		43	42	43		95	94	95		41	-	37,94,114
	44	44	44		104	104	104		44	43	44		104	103	104		42	-	38,95,115
	45	45	45		105	105	105		45	44	45		105	104	105		43	-	39,88,116
	46	46	46		106	106	106		46	45	46		106	105	106		44	-	32,89,117
	47	47	47		107	107	107		47	46	47		107	106	107		45	-	33,90,118
	56	56	56		108	108	108		56	47	56		108	107	108		46	-	34,91,119
$(w+k)$	57	57	57	$(w+k)$	109	109	109	$(w'+k)$	57	56	57	$(w'+k)$	109	108	109	(u)	47	-	35,92,112
	58	58	58		110	110	110		58	57	58		110	109	110		120	-	60,69,105
	59	59	59		111	111	111		59	58	59		111	110	111		121	-	61,70,106
	60	60	60		112	112	112		60	59	60		112	111	112		122	-	62,71,107
	61	61	61		113	113	113		61	60	61		113	112	113		123	-	63,64,108
	62	62	62		114	114	114		62	61	62		114	113	114		124	-	56,65,109
	63	63	63		115	115	115		63	62	63		115	114	115		125	-	57,66,110
	64	64	64		116	116	116		64	63	64		116	115	116		126	-	58,67,111
	65	65	65		117	117	117		65	64	65		117	116	117		127	-	59,68,104
	66	66	66		118	118	118		66	65	66		118	117	118				
	67	67	67		119	119	119		67	66	67		119	118	119				
	68	68	68		120	120	120		68	67	68		120	119	120				
	69	69	69		121	121	121		69	68	69		121	120	121				
	70	70	70		122	122	122		70	69	70		122	121	122				
	71	71	71		123	123	123		71	70	71		123	122	123				
	80	80	80		124	124	124		80	71	80		124	123	124				
	81	81	81		125	125	125		81	80	81		125	124	125				
	82	82	82		126	126	126		82	81	82		126	125	126				
	83	83	83		127	127	127		83	82	83		127	0,126	127				