Towards Closing the Security Gap of Tweak-aNd-Tweak (TNT)

Chun Guo[1], Jian Guo[2], Eik List[3](✉), and Ling Song[4,5]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, China
chun.guo@sdu.edu.cn

[2] Division of Mathematical Sciences, School of Physical and Mathematical Sciences,
Nanyang Technological University, Singapore, Singapore
guojian@ntu.edu.sg

[3] Bauhaus-Universität Weimar, Weimar, Germany
eik.list@uni-weimar.de

[4] Jinan University, Guangzhou, China
songling.qs@gmail.com

[5] Institute of Information Engineering, Chinese Academy of Sciences,
Beijing, China

Abstract. Tweakable block ciphers (TBCs) have been established as a valuable replacement for many applications of classical block ciphers. While several dedicated TBCs have been proposed in the previous years, generic constructions that build a TBC from a classical block cipher are still highly useful, for example, to reuse an existing implementation. However, most generic constructions need an additional call to either the block cipher or a universal hash function to process the tweak, which limited their efficiency.

To address this deficit, Bao et al. proposed Tweak-aNd-Tweak (TNT) at EUROCRYPT'20. Their construction chains three calls to independent keyed permutations and adds the unmodified tweak to the state in between the calls. They further suggested an efficient instantiation TNT-AES that was based on round-reduced AES for each of the permutations. Their work could prove $2n/3$-bit security for their construction, where n is the block size in bits. Though, in the absence of an upper bound, their analysis had to consider all possible attack vectors with up to 2^n time, data, and memory. Still, closing the gap between both bounds remained a highly interesting research question.

In this work, we show that a variant of Mennink's distinguisher on CLRW2 with $O(\sqrt{n}2^{3n/4})$ data and $O(2^{3n/2})$ time from TCC'18 also applies to TNT. We reduce its time complexity to $O(\sqrt{n}2^{3n/4})$, show the existence of a second similar distinguisher, and demonstrate how to transform the distinguisher to a key-recovery attack on TNT-AES$[5, *, *]$ from an impossible differential. From a constructive point of view, we adapt the rigorous STPRP analysis of CLRW2 by Jha and Nandi to show $O(2^{3n/4})$ TPRP security for TNT. Thus, we move towards closing the gap between the previous proof and attacks for TNT as well as its proposed instance.

S. Moriai and H. Wang (Eds.): ASIACRYPT 2020, LNCS 12491, pp. 567–597, 2020.
https://doi.org/10.1007/978-3-030-64837-4_19

Keywords: Cryptanalysis · Block cipher · Tweakable block cipher · AES · Impossible differential

1 Introduction

Tweakable Block Ciphers (TBCs) differ from classical block ciphers in the sense that they take a public input called tweak that can increase the security or the performance of higher-level schemes effectively, e.g., in encryption modes [KR11,PS16], MACs [IMPS17,Nai15], or in authenticated-encryption schemes [IMPS17,JNP16]. Initially, TBCs have been built from classical block ciphers and universal hash functions, starting with Liskov et al.'s constructions [LRW02] LRW1 and LRW2. Various works enlarged the portfolio of generic TBC constructions, e.g. the cascade CLRW2 [LST12], Mennink's constructions $\widetilde{F}[1]$ and $\widetilde{F}[2]$ [Men15], XHX [JLM+17], XHX2 [LL18], or the constructions by Wang et al. [WGZ+16]. These proposals processed the tweak either with a universal hash function or an additional call to the classical block cipher.

As An Alternative Approach, several works proposed dedicated TBCs in the previous decade. In particular, the TWEAKEY framework [JNP14b] found wide adoption, e.g. in Deoxys-BC, Joltik-BC [JNP14b], or Skinny [BJK+16]. Though, since TWEAKEY treats key and tweak equally, any update needs a call to (significant parts of) the TWEAKEY schedule. However, tweak updates occur usually considerably more frequently than key updates. For example, modes like CTRT or ΘCB3 employ a different tweak in each primitive call. Thus, performant tweak-update functions can boost efficiency. KIASU-BC [JNP14a] or CRAFT [BLMR19] avoid tweak schedules, but need further analysis. Moreover, some applications cannot easily be equipped with novel dedicated TBCs but would profit rather from efficient transformations that turn an existing block-cipher implementation into a TBC. For this purpose, generic constructions such as CLRW2 are still relevant. Yet, it would be desirable if its internal hash function could be eliminated to avoid its implementation and the storage of its keys.

Tweak-aNd-Tweak (TNT) is a recent proposal by Bao et al. [BGGS20] for generating a TBC from three block ciphers $E_{K_1}, E_{K_2}, E_{K_3} : \mathcal{K} \times \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$, where $\mathbb{F}_2$ is the Galois Field of characteristic 2 and $\mathcal{K}$ a non-empty set of keys. The encryption of TNT is defined as

$$\mathsf{TNT}[E_{K_1}, E_{K_2}, E_{K_3}](T, M) \overset{\text{def}}{=} E_{K_3}(E_{K_2}(E_{K_1}(M) \oplus T) \oplus T).$$

where the tweak space is $\mathcal{T} = 2^n$. The intermediate values are illustrated in Fig. 1. We will use ΔM, ΔT, etc. to refer to the differences between two values M and M', T and T', and so on. This extends naturally to the other variables. Given ideal secret permutations $\pi_1, \pi_2, \pi_3 \in \mathsf{Perm}(\mathbb{F}_2^n)$, where $\mathsf{Perm}(\mathcal{X})$ is the set of all permutations over a set $\mathcal{X}$, Bao et al. [BGGS20] showed that TNT is a secure tweakable permutation for at least $O(2^{2n/3})$ queries.

TNT-AES instantiates the individual keyed permutations in TNT with round-reduced variants of AES. More precisely, TNT-AES$[r_1, r_2, r_3]$ denotes the version

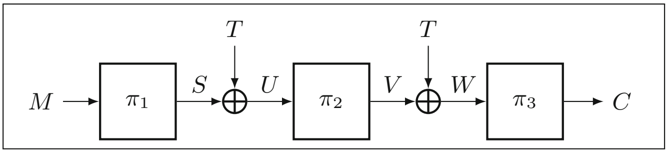


Fig. 1. The encryption of a message M under a tweak T with $\mathsf{TNT}[\pi_1, \pi_2, \pi_3]$.

where π_i uses r_i rounds of the AES, for $1 \le i \le 3$, and the tweak matches the state size of the AES, i.e. $\mathcal{T} = \mathcal{B} = (\mathbb{F}_2^8)^{4\times 4}$, and $n = 128$. The concrete proposal was $\mathsf{TNT\text{-}AES}[6, 6, 6]$ [BGGS20]. While the earlier proposal contained no explicit claim, it suggested that TNT should be treated as a secure tweakable block cipher for up to $O(2^{2n/3})$ queries and $\mathsf{TNT\text{-}AES}$ should provide n-bit security, even in the related-key chosen-tweak setting: "Following the proven security bound of TNT, $\mathsf{TNT\text{-}AES}$ offers $2n/3$-bit security, i.e., there exists no key-recovery attack, given that the data (the combination of tweak and plaintext with no restriction on individual input) and time complexities are bounded by $2^{2\cdot 128/3} \simeq 2^{85}$. Due to the fact that there is no attack against TNT matching the $2^{2n/3}$ bound, all our security analysis against $\mathsf{TNT\text{-}AES}$ are following the $2^n = 2^{128}$ bound for both data and time" [BGGS20, Sect. 5.2]. The best attack in [BGGS20] was a related-tweak boomerang distinguisher on $\mathsf{TNT\text{-}AES}[*, 5, *]$ with 21 active S-boxes. The asterisks indicate that the analysis holds for arbitrary values for r_1 and r_3.

Contribution. This work aims at narrowing the security gap from both sides. We show in Sect. 2 that a variant of Mennink's distinguisher on $\mathsf{CLRW2}$ [Men18] also applies to TNT, which yields a theoretical TPRP (i.e., chosen-tweak, chosen-plaintext) distinguisher in $O(\sqrt{n}2^{3n/4})$ time, data, and memory complexity. As improvements, we reduce the complexity of Mennink's information-theoretic distinguisher from $O(2^{3n/2})$ to $O(2^{3n/4})$ computations. More precisely, we show two similar TPRP distinguishers that we call parallel-road and cross-road distinguishers. We use one of them to mount a partial key-recovery attack on the instance $\mathsf{TNT\text{-}AES}[5, *, *]$ with an impossible differential in Sect. 3. Since it needs more message pairs, its complexity exceeds $O(2^{3n/4})$ but is still considerably below 2^n computations and data. We emphasize that we do not break the proposed version $\mathsf{TNT\text{-}AES}[6, 6, 6]$ of [BGGS20].

From a constructive point of view, we show that the rigorous STPRP (i.e., chosen plain- and ciphertext queries) analysis by Jha and Nandi on $\mathsf{CLRW2}$, that showed security for up to $O(2^{3n/4})$ queries, can be adapted to a TPRP proof of TNT with similar complexity. Thus, we move a considerable step towards closing the gap between proofs and attacks for TNT and its proposed instance.

Notation. We use uppercase characters for variables and functions, lowercase characters for indices, calligraphic characters for sets and distributions, and sans-serif characters for random variables. For $n \in \mathbb{N}$, let $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$ and $[0..n] \stackrel{\text{def}}{=} \{0, 1, \ldots, n\}$. For a bit string $X \in \mathbb{F}_2^n$, let $X = (X_{n-1}X_{n-2}\ldots X_0)$ be its individual bits. We assume that the most significant bit is the leftmost, and the least significant bit is the rightmost bit, s.t. the integer representation

x of X is $x = \sum_i 2^i \cdot X_i$. For $x < 2^n$, we will use $X = \langle x \rangle_n$ as conversion of an integer x into a n-bit string X that represents x. For non-negative integers $\leq n$ and $X \in \mathbb{F}_2^n$, we will use $\mathsf{lsb}_x(X)$ as function that returns the least significant x bits of X and $\mathsf{msb}_x(X)$ to return the most significant x bits of X. $(n)_k$ denotes the falling factorial $n!/(n-k)!$. For non-negative integers $x + y = n$ and $Z \in \mathbb{F}_2^n$, we will use $(X, Y) \xleftarrow{x,y} Z$ to denote that $X \parallel Y = Z$ where $|X| = x$ and $|Y| = y$. Similar to Perm, we define $\widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{X})$ as the set of all tweakable permutations $\widetilde{\pi} : \mathcal{T} \times \mathcal{X} \to \mathcal{X}$ over $\mathcal{X}$ with tweak space $\mathcal{T}$.

Practical Implications. While an STPRP proof is desirable, the implications of higher TPRP security already provide a valuable gain for TBC-based schemes that do not need the primitive's inverse. Considering authenticated encryption schemes, examples of such schemes include SCT [PS16], ZAE [IMPS17], ZOTR [BGIM19], or the TBC-based variants of OTR ($\mathbb{OTR}$) [Min14] and COFB (iCOFB) [CIMN17,CIMN20]. Considering MACs, there exist various such constructions, e.g. ZMAC [IMPS17] and its derivates [LN17,Nai18]. The security of those schemes is limited by the minimum of $O(2^{\min(n,(n+t)/2)})$ queries and the TPRP security of the underlying primitive. Since the latter is the bottleneck, its improvement yields directly higher security guarantees for the schemes.

AES. We assume that the reader is familiar with the AES. We use R to refer to the round function, X^i for the state after i rounds, starting with X^0 as the plaintext, and K^0 for the initial round key. We use X^i_{SB}, X^i_{SR}, and X^i_{MC} to refer to the state directly after the $\mathsf{SubBytes}$, $\mathsf{ShiftRows}$, and $\mathsf{MixColumns}$ operation in the i-th round, respectively. Moreover, $X^i[j]$ refers to the j-th byte of X^i. For $\mathcal{I} \subseteq \{0, 1, 2, 3\}$, we adopt the subspaces for diagonals $\mathcal{D}_{\mathcal{I}}$, columns $\mathcal{C}_{\mathcal{I}}$, inverse (or anti-)diagonals $\mathcal{ID}_{\mathcal{I}}$, and mixed spaces $\mathcal{M}_{\mathcal{I}}$ from Grassi et al. [GRR16].

2 Distinguishers on TNT

Here, we briefly describe two distinguishers on TNT with $O(\sqrt{n} \cdot 2^{3n/4})$ queries, which implies an upper bound on the (query) security of TNT of at most $O(q^4/(\sqrt{n} \cdot 2^{3n}))$. Our distinguishers are illustrated in Fig. 2. We do not claim that our observations are novel. Instead, both are applications of [LNS18] and [Men18]. The latter, however, is an information-theoretic distinguisher that uses $O(\sqrt{n} \cdot 2^{3n/4})$ queries, but the description by Mennink demands $O(2^{3n/2})$ offline operations to identify the required pairs.

We note that Sibleyras' work [Sib20] proposes generic key-recovery attacks for LRW2 and cascades that also hold for $\mathsf{CLRW2}$. Those attacks slightly reduce the time complexity of Mennink's attack, but require more queries, roughly $2^{2(n+k)/3}$, and are hence in $O(2^n)$ for plausible values of the key size $k \geq n/2$.

2.1 General Setup

Let $M^0, M^1 \in \mathbb{F}_2^n$ be two distinct messages and $\mathcal{T}^0$ and $\mathcal{T}^1$ be two sets of $q = 2^{3n/4+x}$ pairwise distinct random tweaks T_j for $0 \leq j < q$ in each set, where

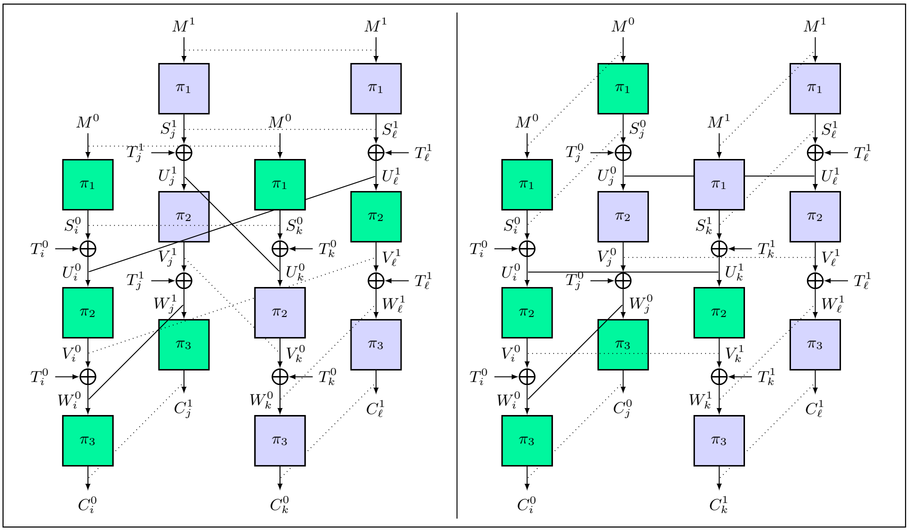


Fig. 2. Cross- (left) and parallel-road (right) distinguishers on TNT. Solid horizontal lines are probabilistic equalities that hold with probability around 2^{-n} each. Dotted lines hold either by choice or by design once the solid-line equalities are fulfilled.

$\mathcal{T}^i$ associates all tweaks with a fixed message M^i for $\mathcal{T}^i$ for $i = 0, 1$. We describe two ways to combine two pairs to quartets each that differ in the way where the messages are used. Figure 2 may illustrate why we call them parallel- and cross-road distinguishers. For both distinguishers, we want two pairs, (M_i, T_i) and (M_j, T_j) as well as (M_k, T_k) and (M_ℓ, T_ℓ), with the same tweak difference $\Delta T_{i,j} = T_i \oplus T_j = \Delta T_{k,\ell} = T_k \oplus T_\ell$, and for which $C_i = C_j$ and $C_k = C_\ell$.

2.2 Cross-Road Distinguisher

Here, we denote the queries and intermediate variables

- related to $(M^0, T_i^0) \in \mathcal{T}^0$ also as $(S_i^0, U_i^0, V_i^0, W_i^0, C_i^0)$,
- those related to $(M^1, T_j^1) \in \mathcal{T}^1$ also as $(S_j^1, U_j^1, V_j^1, W_j^1, C_j^1)$,
- those related to $(M^0, T_k^0) \in \mathcal{T}^0$ also as $(S_k^0, U_k^0, V_k^0, W_k^0, C_k^0)$, and
- those related to $(M^1, T_\ell^1) \in \mathcal{T}^1$ also as $(S_\ell^1, U_\ell^1, V_\ell^1, W_\ell^1, C_\ell^1)$.

Clearly, we want $i \neq k$ and $j \neq \ell$ as well as $(i, j) \neq (k, \ell)$.

Procedure. We define two construction functions:

$$\tau_0(i) \overset{\text{def}}{=} 0^{n/4-x} \parallel \langle i \rangle_{3n/4+x}, \quad \text{and} \quad \tau_1(j) \overset{\text{def}}{=} \langle j \rangle_{3n/4+x} \parallel 0^{n/4-x}.$$

The resulting tweak structures are illustrated in Fig. 3. The distinguisher procedure is given on the left-hand side of Algorithm 1. Let $\theta \geq 0$ be a threshold. The threshold depends on the desired error (and success) probability and will be discussed in Sect. 3.3. The distinguisher can be described as:

Algorithm 1. Distinguishers on TNT.

```
function crossRoad
    $K \twoheadleftarrow \mathbb{F}_2^k$
    $M^0 \twoheadleftarrow \mathbb{F}_2^n$
    $M^1 \twoheadleftarrow \mathbb{F}_2^n$
    $\mathsf{coll} \leftarrow 0$
    $\mathcal{L} \leftarrow [] \times [0..2^n - 1]$                  ▷ $2^n$ elements
    $\mathcal{D} \leftarrow 0 \times [0..2^n - 1]$                   ▷ $2^n$ elements
    for $i \leftarrow 0..q-1$ do                          ▷ $q$ iterations
        $T_i^0 \leftarrow \tau_0(i)$
        $C_i^0 \leftarrow \mathcal{E}_K(T_i^0, M^0)$
        $\mathcal{L}[C_i^0] \overset{\cup}{\leftarrow} \{T_i^0\}$
    for $j \leftarrow 0..q-1$ do                          ▷ $q$ iterations
        $T_j^1 \leftarrow \tau_1(j)$
        $C_j^1 \leftarrow \mathcal{E}_K(T_j^1, M^1)$
        $\mathsf{coll} \leftarrow \mathsf{coll}+\mathsf{findNumColls}(\mathcal{L}, \mathcal{D}, T_j^1, C_j^1)$
    return $\mathsf{coll} \geq \theta$

function findNumColls($\mathcal{L}$, $\mathcal{D}$, $T_j^1$, $C_j^1$)
    $c \leftarrow 0$
    for all $T_i^0$ in $\mathcal{L}[C_j^1]$ do
                              ▷ $2^{n/2}$ calls over all executions
        $\Delta T_{i,j} \leftarrow T_i^0 \oplus T_j^1$
        $c \leftarrow c + \mathcal{D}[\Delta T_{i,j}]$
        $\mathcal{D}[\Delta T_{i,j}] \leftarrow \mathcal{D}[\Delta T_{i,j}] + 1$
    return $c$

function parallelRoad
    $K \twoheadleftarrow \mathbb{F}_2^k$
    $M^0 \twoheadleftarrow \mathbb{F}_2^n$
    $M^1 \twoheadleftarrow \mathbb{F}_2^n$
    $\mathsf{coll} \leftarrow 0$
    $\mathcal{L} \leftarrow [] \times [0..2^n - 1]$    ▷ $2^n$ elements
    $\mathcal{D} \leftarrow 0 \times [0..2^n - 1]$     ▷ $2^n$ elements
    for $i \leftarrow 0..q-1$ do            ▷ $q$ iterations
        $T_i^0 \leftarrow \tau_0(i)$
        $C_i^0 \leftarrow \mathcal{E}_K(T_i^0, M^0)$
        for all $T_j^0$ in $\mathcal{L}[C_i^0]$ do
            $\Delta T_{i,j}^0 \leftarrow T_i^0 \oplus T_j^0$
            $\mathcal{D}[\Delta T_{i,j}^0] \leftarrow \mathcal{D}[\Delta T_{i,j}^0] + 1$
        $\mathcal{L}[C_i^0] \overset{\cup}{\leftarrow} \{T_i^0\}$
    $\mathcal{L} \leftarrow [] \times [0..2^n - 1]$    ▷ $2^n$ elements
    for $k \leftarrow 0..q-1$ do            ▷ $q$ iterations
        $T_k^1 \leftarrow \tau_1(k)$
        $C_k^1 \leftarrow \mathcal{E}_K(T_k^1, M^1)$
        for all $T_\ell^1$ in $\mathcal{L}[C_k^1]$ do
         ▷ $2^{n/2}$ calls over all executions
            $\Delta T_{k,\ell}^1 \leftarrow T_k^1 \oplus T_\ell^1$
            $\mathsf{coll} \leftarrow \mathsf{coll} + \mathcal{D}[\Delta T_{k,\ell}^1]$
        $\mathcal{L}[C_k^1] \overset{\cup}{\leftarrow} \{T_k^1\}$
    return $\mathsf{coll} \geq \theta$
```

1. Initialize two lists $\mathcal{L}$ and $\mathcal{D}$ and initialize a counter $\mathsf{coll} = 0$.
2. For $i \in [0..q-1]$:
   - Use $\tau_0(i)$ as tweak-construction function to generate queries (M^0, T_i^0). Encrypt them to obtain $C_i^0 \leftarrow \mathcal{E}_K(T_i^0, M^0)$. Insert T_i^0 to $\mathcal{L}[C_i^0]$.
3. For $j \in [0..q-1]$:
   - Use $\tau_1(j)$ as tweak-construction function to generate queries (M^1, T_j^1). Encrypt them to obtain $C_j^1 \leftarrow \mathcal{E}_K(T_j^1, M^1)$.

     3.1. For each $T_i^0 \in \mathcal{L}[C_j^1]$:
       - Derive $\Delta T_{i,j} = T_i^0 \oplus T_j^1$. Derive the number of pairs (T_k^0, T_ℓ^1) with the same tweak difference from $\mathcal{D}[\Delta T_{i,j}]$, add this number c to the total number of colliding quartets, coll, and increment $\mathcal{D}[\Delta T_{i,j}]$.
4. If $\mathsf{coll} > \theta$, return "real" and "random" otherwise.

Since the i-th and k-th query share the same message M^0, it follows that $S_i^0 = S_k^0$; a similar argument holds from $M_j^1 = M_\ell^1$ to $S_j^1 = S_\ell^1$. With probability 2^{-n}, it holds that $S_i^0 \oplus S_\ell^1 = T_i^0 \oplus T_\ell^1$. In this case, it follows that $U_i^0 = U_\ell^1$. By combination, there exist approximately $(2^{3n/4+x})^2 \cdot 1/(2^n - 1) \simeq 2^{n/2+2x}$ ordered collision pairs $U_i^0 = U_\ell^1$ between (M^0, T_i^0) and (M^1, T_ℓ^1). There exist $(2^{3n/4+x} - 1)^2 \cdot 1/(2^n - 1) \simeq 2^{n/2+2x}$ ordered collision pairs $U_k^0 = U_j^1$ between (M^0, T_k^0) and (M^1, T_j^1). Note that this is a conditional probability; since $S_k^0 = S_i^0$ and $S_j^1 = S_\ell^1$, it follows from $T_k^0 \oplus T_j^1 = S_k^0 \oplus S_j^1$ that $T_k^0 \oplus T_j^1 = T_i^0 \oplus T_\ell^1$. Those

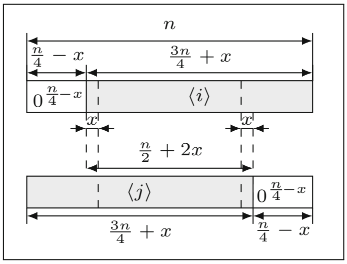


Fig. 3. Construction of the tweak sets.

will be mapped to $V_i^0 = V_\ell^1$ and $V_k^0 = V_j^1$. Thus, by combination, there are $\binom{2^{n/2+2x}}{2} \simeq 2^{n+4x-1}$ pairs of pairs (quartets) with $T_i^0 \oplus T_\ell^1$ and $T_k^0 \oplus T_j^1$. With probability 2^{-n}, a quartet has $V_i^0 \oplus V_\ell^1 = T_k^0 \oplus T_j^1$, which implies $W_i^0 = W_j^0$. Since $V_i^0 = V_\ell^1$ and $V_k^0 = V_j^1$, this implies that $W_k^0 = W_\ell^1$ also holds. We obtain

$$\binom{2^{n/2+2x}}{2} \cdot 2^{-n} \simeq 2^{4x-1} \text{ quartets.}$$

Similarly, we expect $(2^{3n/4+x})^2 \cdot 2^{-n} \simeq 2^{n/2+2x}$ pairs $C_i^0 = C_j^1$ formed by accident, which can be combined to

$$\binom{2^{n/2+2x}}{2} \cdot 2^{-n} \simeq 2^{4x-1} \text{ quartets.}$$

For a random tweakable permutation, only the latter events occur, whereas we have two sources in the real world. Thus, we can expect twice as many quartets in the real construction compared to the ideal world.

Experimental Verification. To improve the understanding, we followed Mennink's approach and also implemented the distinguisher for small permutations. We used TNT with three independent instances of Small-PRESENT-n [Lea10], the small-scale variants of PRESENT [BKL+07], with the original key schedule of PRESENT as proposed there, where the original round keys K^i are truncated to their rightmost (least significant) n bits for $n \in \{16, 20, 24\}$. We employed the full 31 rounds as for the original PRESENT cipher. For the real construction, we sampled uniformly and independently 1 000 random keys, one per experiment, and two random messages. The tweaks were constructed as in Fig. 3. For the ideal world, we sampled the ciphertexts uniformly and independently at random and verified that no message-pair for the same tweak amid any experiment collides. The results of our implementation are summarized in Table 1a. The source code of all experiments can be found freely available to the public.[1]

[1] https://gitlab.com/elist/tnt.

Table 1. Average #quartets for TNT with Small-PRESENT-n as permutations π_i ("real") and pseudorandom sampling ("ideal") over 1 000 experiments with random keys, two random messages, and 2^t tweaks per message in each experiment.

(a) Cross-road distinguisher.

n	t	Ideal	Real
16	11	0.026	0.061
16	12	0.485	1.009
16	13	7.967	15.970
16	14	127.458	255.133
20	14	0.032	0.055
20	15	0.494	0.960
20	16	8.087	16.162
20	17	128.057	255.739
24	17	0.034	0.066
24	18	0.482	1.009
24	19	7.979	16.174
24	20	127.941	255.661

(b) Parallel-road distinguisher.

n	t	Ideal	Real
16	11	0.015	0.050
16	12	0.232	0.787
16	13	4.076	12.127
16	14	64.274	192.275
20	14	0.024	0.057
20	15	0.274	0.749
20	16	3.892	11.952
20	17	64.405	191.398
24	17	0.016	0.063
24	18	0.233	0.726
24	19	4.016	12.170
24	20	63.686	191.599

2.3 Parallel-Road Distinguisher

Our second distinguisher is described in the right-hand side of Algorithm 1 and is illustrated on the right-hand side of Fig. 2. The core difference is the choice of sets for collisions. While the first distinguisher used collisions from different messages, the second one uses collisions from ciphertexts from the same set. Here, we denote the queries and intermediate variables

- related to $(M^0, T_i^0) \in \mathcal{T}^0$ also as $(S_i^0, U_i^0, \ldots)$,
- those related to $(M^0, T_j^0) \in \mathcal{T}^0$ also as $(S_j^0, U_j^0, \ldots)$,
- those related to $(M^1, T_k^1) \in \mathcal{T}^1$ also as $(S_k^1, U_k^1, \ldots)$, and
- those related to $(M^1, T_\ell^1) \in \mathcal{T}^1$ also as $(S_\ell^1, U_\ell^1, \ldots)$.

By combination, we obtain about $(2^{3n/4+x})^2 \cdot 2^{-n} \simeq 2^{n/2+2x}$ collisions $U_i^0 = U_k^1$ between (M^0, T_i^0) and (M^1, T_k^1), and the approximately same number of collisions $U_j^1 = U_\ell^1$ between (M^1, T_j^1) and (M^1, T_ℓ^1). Those will be mapped to $V_i^0 = V_k^1$ and $V_j^0 = V_\ell^1$. We can form $\binom{2^{n/2+2x}}{2} \simeq 2^{n+4x-1}$ pairs of pairs (quartets). With probability 2^{-n}, a quartet has $V_i^0 \oplus V_j^0 = T_i^0 \oplus T_j^0$, which implies $W_i^0 = W_j^0$. Since $V_i^0 = V_k^1$ and $V_j^1 = V_\ell^1$, it follows that $W_k^1 = W_\ell^1$ holds. Thus, we obtain 2^{4x-1} quartets. Moreover, we expect $\binom{2^{3n/4+x}}{2} \cdot 2^{-n} \simeq 2^{n/2+2x-1}$ pairs $C_i^0 = C_j^0$ that are formed randomly and can be combined with $2^{n/2+2x-1}$ pairs $C_k^1 = C_\ell^1$. We obtain

$$(2^{n/2+2x-1})^2 \cdot 2^{-n} \simeq 2^{4x-2} \text{ quartets} \qquad (1)$$

Algorithm 2. More efficient variant of the parallel-road distinguisher on TNT.

```
function PARALLELROAD
    $K \twoheadleftarrow \mathbb{F}_2^k$
    $M^0 \twoheadleftarrow \mathbb{F}_2^n$
    $M^1 \twoheadleftarrow \mathbb{F}_2^n$
    coll ← 0
    $\mathcal{L} \leftarrow [] \times [0..q-1]$                ▷ $q$ elements
    $\mathcal{D} \leftarrow [] \times [0..q-1]$                ▷ $q$ elements
    for $i \leftarrow 0..q-1$ do                ▷ $q$ iterations
        $T_i^0 \leftarrow \tau_0(i)$
        $C_i^0 \leftarrow \mathcal{E}_K(T_i^0, M^0)$
        $(b_i^0, c_i^0) \xleftarrow{n/4,3n/4} C_i^0$
        for all $(T_j^0, b_j^0)$ in $\mathcal{L}[c_i^0]$ do
            if $b_i^0 = b_j^0$ then                ▷ $C_i^0 = C_j^0$
                $\Delta T_{i,j}^0 \leftarrow T_i^0 \oplus T_j^0$
                $(s_{i,j}^0, t_{i,j}^0) \xleftarrow{n/4,3n/4} \Delta T_{i,j}^0$
                $\mathcal{D}[t_{i,j}^0] \overset{\cup}{\leftarrow} \{s_{i,j}^0\}$
        $\mathcal{L}[c_i^0] \overset{\cup}{\leftarrow} \{(T_i^0, b_i^0)\}$
    $\mathcal{L} \leftarrow [] \times [0..q-1]$                ▷ $q$ elements
    for $k \leftarrow 0..q-1$ do                ▷ $q$ iterations
        $T_k^1 \leftarrow \tau_1(k)$
        $C_k^1 \leftarrow \mathcal{E}_K(T_k^1, M^1)$
        $(b_k^1, c_k^1) \xleftarrow{n/4,3n/4} C_k^1$
        for all $(T_\ell^1, b_\ell^1)$ in $\mathcal{L}[c_k^1]$ do
            if $b_k^1 = b_\ell^1$ then                ▷ $C_k^1 = C_\ell^1$
                $\Delta T_{k,\ell}^1 \leftarrow T_k^1 \oplus T_\ell^1$
                $(s_{k,\ell}^1, t_{k,\ell}^1) \xleftarrow{n/4,3n/4} \Delta T_{k,\ell}^1$
                for all $s_{i,j}^0$ in $\mathcal{D}[t_{k,\ell}^1]$ do                ▷ $\Delta T_{i,j}^0 = \Delta T_{k,\ell}^1$
                    if $s_{i,j}^0 = s_{k,\ell}^1$ then
                        coll ← coll + 1
        $\mathcal{L}[c_k^1] \overset{\cup}{\leftarrow} \{(T_k^1, b_k^1)\}$
return coll $\geq \theta$
```

formed at random. In sum, this yields

$$2^{4x-1} + 2^{4x-2} = 3 \cdot 2^{4x-2} \text{ quartets}$$

in the real construction, which implies that we can expect roughly three times as many quartets in the real construction compared to a random tweakable permutation wherin only the latter events occur.

Experimental Verification. We implemented the distinguisher with Small-PRESENT and state and tweak sizes of $n \in \{16, 20, 24\}$ bits. The results are summarized in Table 1b.

2.4 Efficiency

Mennink's [Men18] distinguisher evaluated the number of quartets for each tweak difference $\Delta \in \mathbb{F}_2^n$. From the choice of pairs given τ_0 and τ_1, there existed $2^{n/2+2x}$ possible pairs (C_i^0, C_j^1) for each tweak difference. Thus, the naive way needed $2^n \cdot 2^{n/2+2x} \simeq O(2^{3n/2+2x})$ operations to exhaust all 2^n possible tweak differences. To reduce the computational complexity below $O(2^n)$, we give an improved description of the parallel-road distinguisher.

The lists $\mathcal{L}$ and $\mathcal{D}$ needed to reserve 2^n cells each, which was the bottleneck. To reduce the complexity, we shrink $\mathcal{L}$ to a list of $2^{3n/4}$ sub-lists, where $\mathcal{L}[x]$ holds a sub-list of tweaks T_i^0 s.t. $\mathsf{lsb}_{3n/4}(C_i^0) = \langle x \rangle_{3n/4}$. This means that we truncate the $n/4$ most significant bits (MSB) of C_i^0. Additionally, we store also the $n/4$ truncated bits as part of the entry: $(T_i^0, \mathsf{msb}_{n/4}(C_i^0))$. Similarly, we no longer store a list of 2^n counters in $\mathcal{D}$. Instead, each entry will be a sub-list of full tweak differences. Thus, $\mathcal{D}[x]$ contains $2^{3n/4}$ slots, where $\Delta T_{i,j}$ is stored in the sub-list at location $\mathsf{lsb}_{3n/4}(\Delta T_{i,j}) = \langle x \rangle_{3n/4}$. Clearly, the length of the sub-list at $\mathcal{D}[x]$ equals the previous counter value that was stored in $\mathcal{D}[x]$ before.

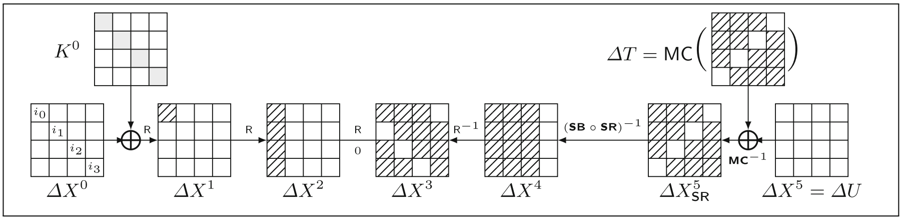


Fig. 4. Key recovery and impossible differential trail through 1 + 4 rounds of AES. Hatched bytes are active; filled bytes are targeted key bytes; indices in bytes denote that a set index is encoded into them.

On average, Line 22 of Algorithm 2 is called

$$\binom{2^{3n/4+x}}{2} \cdot 2^{-3n/4} \simeq 2^{3n/4+2x-1}$$

times. The second test in the if-statement on $n/4$ bits is fulfilled in about $2^{n/2+2x-1}$ calls. Thus, the first loop from Line 18 in Algorithm 2 has roughly $2^{3n/4+2x}$ operations on average. A similar argument holds for the second test in Line 33 of Algorithm 2. Thus, the second outer loop over q tweaks from Line 29 of Algorithm 2 also contains roughly $2^{3n/4+2x}$ operations on average. More detailed, the first $3n/4$-bit filter reduces again the number of pairs

$$\binom{2^{3n/4+x}}{2} \cdot 2^{-3n/4} \simeq 2^{3n/4+2x-1}$$

times. The second test in the if-statement on $n/4$ bits is fulfilled in $2^{n/2+2x-1}$ times on average. The $3n/4$-bit tweak-difference filter lets the check in Line 34 in Algorithm 2 be successful $2^{n/4+4x-2}$ times for $(2^{3n/4+2x-1})^2$ pairs. Thus, it will be called at most $2^{3n/4+x} + 2^{n/2+2x} + 2^{n/4+4x-2}$ and the overall computational complexity is in $O(2^{3n/4+2x})$.

3 An Impossible-Differential Attack on $\mathsf{TNT\text{-}AES}[5, *, *]$

We combine the well-known impossible differential on four-round AES for key-recovery attacks on versions of TNT-AES. We describe the key-recovery phase in the first round and both key recovery and impossible differential in π_1.

3.1 Core Idea

The core idea is based on the following assumption. The $O(\sqrt{n} \cdot 2^{3n/4})$-distinguisher works iff we can find pairs that collide in U. Let us consider the parallel-road distinguisher. It needs pairs (M^0, M^1) whose difference $\pi_1(M^0) \oplus \pi_1(M^1) = S^0 \oplus S^1$ equals the difference of their corresponding tweaks: $S^0 \oplus S^1 = T_i^0 \oplus T^1$, which implies that $U^0 = U^1$. The adversary can choose differences ΔT of its choice as well as plaintexts with certain input differences. If it can manage to exclude that ΔT occurs for the message inputs of its choice, then, the distinguisher cannot happen. This implies that $U^0 \neq U^1$ for all choices of M^0 and M^1. As a result, the values $V_i^0, V_j^0, V_k^1, V_\ell^1$ are pairwise unique for each quartet and the number of colliding pairs will then match that of a random tweakable permutation.

For this purpose, the adversary considers tweaks such that their differences ΔT are output differences of an impossible differential. Then, each correct quartet from the distinguisher is possible only if the message was not encrypted through the first (few) round(s) to an input difference of the impossible differential, which allows discarding all keys that would have encrypted it in this way. We need a sufficient number of pairs such that for all key candidates, we will expect a correct quartet (for TNT-AES), except for the correct key.

We use the impossible differential from Fig. 4, where ΔX^5_{MC} (the difference after five rounds) is identical to ΔT. Let $\mathcal{I} = \{0, 1, 2\}$ and let $\mathcal{M}_{\mathcal{I}}$ denote the mixed space after applying MixColumns to a vector space that is active in the first three inverse diagonals (cf. [GRR16]). Our choice leaves a space of $2^{96} - 1$ differences for $\Delta T \in \mathcal{M}_{\mathcal{I}}$ and call $\mathcal{T}$ the space of desired tweak differences.

3.2 Messages

We need message pairs with the impossible difference after π_1. Since the difference has 32 zero bits, a zero difference in the rightmost inverse diagonal has a probability of 2^{-32}. We try to recover $K^0[0, 5, 10, 15]$. For a message pair (M^i, M^j) that produces the impossible difference after π_1, we can discard all key candidates that would lead to a difference of $\Delta X^1 \stackrel{\text{def}}{=} \mathsf{R}(M^i) \oplus \mathsf{R}(M^j)$ that is active in only a single byte after the first round. On average, there exist $4 \cdot 2^8 = 2^{10}$ possible output differences ΔX^1. Since $M^i \oplus M^j$ is fixed, approximately one input-output mapping exists for the AES S-box on average. Hence, 2^{10} keys produce an impossible ΔX^1 on average and can be discarded. Assuming that the discarded keys are uniformly randomly and independently distributed, the probability that a key candidate can be discarded from a given pair (M^i, M^j) is 2^{-22}. Under standard assumptions, we need N_{pairs} pairs to reduce the number of key candidates to 2^{32-a}, where a is the advantage in bits:

$$(1 - 2^{-22})^{N_{\mathsf{pairs}}} \leq 2^{-a}. \tag{2}$$

Equation (2) yields approximately $2^{23.47}$, $2^{24.47}$, $2^{25.47}$, and $2^{26.47}$ necessary message pairs that fulfill the impossible difference after π_1 to obtain an advantage of $a = 4$, 8, 16, and 32 bits, respectively. If we fix the position of the inactive diagonal, we need $2^{26.47} \cdot 2^{32} \simeq 2^{58.47}$ message pairs, or $2 \cdot 2^{29.24} \simeq 2^{30.24}$ pairwise distinct messages. The number of message pairs with less than four active input bytes is negligible. We define $2^s \simeq \lceil 2^{29.24} \rceil$. We employ a space of a single plaintext diagonal, where we can focus on the first diagonal $\mathcal{D}_{\{0\}}$. The remaining diagonals are fixed to constants. We want a certain tweak difference that is zero in the final inverse diagonal. We add computational effort by choosing many

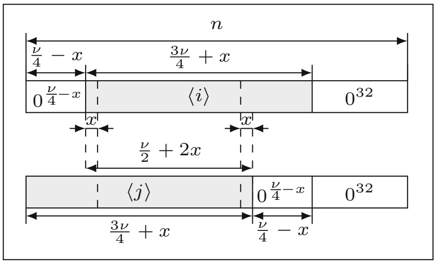


Fig. 5. Encoding the indices i and j into the tweaks to build the tweak sets $\mathcal{T}^i$ and $\mathcal{T}^j$ corresponding to the messages M^i and M^j.

messages that we partially compensate for by fixing those 32 bits to constants in all tweaks and define $\nu =^{\text{def}} n - 32 = 96$ for the AES.

Expected Number of Pairs. To each message M^i, we associate a tweak set $\mathcal{T}^i$, where we use the same tweaks for each message. Among the pairs in a single set (M, T^i) and (M, T^j), the probability for $C^i = C^j$ is approximately 2^{-n}.

Using 2^t tweaks in a set, we obtain $\binom{2^t}{2} \cdot 2^{-n} \simeq 2^{2t-n-1}$ pairs. Given two messages that do **not** have the desired tweak difference after π_1, we can combine the pairs, where each pair collides in its ciphertexts, to $(2^{2t-n-1})^2$ quartets, which have the correct tweak difference after π_1 with probability $2^{3n/4} = 2^{-96}$. Thus, the number of quartets become

$$(2^{2t-n-1})^2 \cdot 2^{-3n/4} \simeq 2^{4t-11n/4-2} \simeq 2^{4t-354}. \tag{3}$$

For messages that produce the desired difference after π_1, i.e., have 32 zero bits in the rightmost inverse diagonal, we can form $2^t \cdot 2^t \cdot 2^{-3n/4} \simeq 2^{2t-3n/4}$ pairs after π_1 since only the 96-bit tweak difference must match that of the message difference at that point. From those pairs, we can build quartets that collide with probability 2^{-n} after π_2. Thus, the number of quartets becomes

$$\binom{2^{2t-3n/4}}{2} \cdot 2^{-n} \simeq 2^{4t-5n/2-1} \simeq 2^{4t-321}. \tag{4}$$

Note that the number of quartets in Eq. (3) differs significantly from the 2^{4x-2} of Eq. (1) since we restrict the valid tweak differences. Here, we need more message pairs so that enough of them possess the desired 32-bit condition of the zero-difference anti-diagonal after π_1. Thus, the here-proposed attack is **less** efficient but allows us to recover a part of the secret key.

3.3 Success Probability, Advantage, and Data Complexity

Samajder and Sarkar [SS17] gave rigorous upper bounds on the data complexities for differential and linear cryptanalysis that improved previous results. For the parallel-road distinguisher, $2 \cdot 2^t$ message-tweak tuples in total produce

$2^{4t-5n/2-1} + 2^{4t-11n/4-2}$ quartets for the real world, and $2^{4t-11n/4-2}$ quartets in the ideal world on average. Thus, we can define for the probability of quartets

$$p_{\mathsf{cor}} \simeq 2^{-321} + 2^{-354} \quad \text{and} \quad p_{\mathsf{wrong}} \simeq 2^{-354}.$$

Let θ be a threshold and H_0 be the hypothesis that a given message pair M^i, M^j has the 32-bit zero difference after π_1 in the rightmost anti-diagonal. We say that H_0 holds if $N^{i,j}_{\mathsf{quartets}} > \theta$. Otherwise, we reject H_0.

Let $\alpha =^{\text{def}} \Pr[N^{i,j}_{\mathsf{quartets}} < \theta | M^i \oplus M^j \in \mathcal{T}]$ be the Type-I error, i.e., a pair with correct difference has too few quartets. This event is not essential, but yields more surviving wrong key candidates. Let $\beta =^{\text{def}} \Pr[N^{i,j}_{\mathsf{quartets}} \geq \theta | M^i \oplus M^j \notin \mathcal{T}]$ be the Type-II error, i.e., a pair with wrong difference after π_1 has more quartets than the threshold and is incorrectly classified as correct. The latter event is crucial since the pair might suggest the correct key as wrong and the attack will fail. Therefore, the success probability is given by

$$1 - \sum_{i<j} \Pr\left[N^{i,j}_{\mathsf{quartets}} \geq \theta\right] \cdot 2^{-22} \leq 1 - \left(2^{58.47} \cdot \Pr\left[N^{i,j}_{\mathsf{quartets}} \geq \theta\right] \cdot 2^{-22}\right).$$

Thus, β should be far below $2^{-36.5}$. From [SS17, Proposition 5.1], it follows that the number of quartets (for each message pair) should fulfill

$$N_{\mathsf{quartets}} \geq \frac{3\left(\sqrt{p_{\mathsf{cor}} \ln\left(\frac{1}{\alpha}\right)} + \sqrt{p_{\mathsf{wrong}} \ln\left(\frac{2}{\beta}\right)}\right)^2}{\left(p_{\mathsf{cor}} - p_{\mathsf{wrong}}\right)^2}. \tag{5}$$

Since the distinguisher produces $N_{\mathsf{quartets}} = 2^{4t-2}$ quartets, we can derive $t = (\log_2(N_{\mathsf{quartets}}) + 2)/4$. Results of t for plausible values of α and β are listed in Table 2. For Hypothesis 3, Samajder and Sarkar [SS17] suggest a threshold of

$$\theta = \sqrt{3 N_{\mathsf{quartets}} \cdot p_{\mathsf{wrong}} \cdot \ln\left(\frac{2}{\beta}\right)},$$

which is given in Table 2 for the sake of simplicity. Equation (5) targets single-differential key-recovery attacks.

Remark 1. We point out that Samajder and Sarkar also studied an upper bound for the data complexity of distinguishers in [SS17, Proposition 8.1]:

$$N_{\mathsf{quartets}} \geq \frac{v^2 \ln\left(\frac{1}{P_e}\right)}{2\left(D\left(\mathcal{P} \,\|\, \mathcal{Q}\right) + D\left(\mathcal{Q} \,\|\, \mathcal{P}\right)\right)^2}. \tag{6}$$

Though, [SS17, Sect. 10] showed that Eq. (5) yields a better upper bound for single-differential cryptanalysis. Details can be found in their work.

Table 2. Logarithmic data complexity per set t and logarithmic threshold values θ for varying error probabilities.

	$-\log_2(\beta)/\log_2(\theta)$					
$-\log_2(\alpha)$	38	39	41	45	53	69
1	80.882/–	80.882/–	80.882/–	80.882/–	80.882/–	80.882/–
2	81.382/−0.201	81.382/−0.201	81.382/−0.201	81.382/−0.201	81.382/−0.200	81.382/−0.200
4	81.882/3.204	81.882/3.204	81.882/3.204	81.882/3.204	81.882/3.204	81.882/3.204
8	82.382/5.730	82.382/5.730	82.382/5.730	82.382/5.730	82.382/5.730	82.382/5.730
16	82.882/8.012	82.882/8.012	82.882/8.012	82.882/8.012	82.882/8.012	82.882/8.012
32	83.382/10.183	83.382/10.183	83.382/10.183	83.382/10.183	83.382/10.183	83.382/10.183

Data Complexity. Choosing a sufficiently high threshold for the number of quartets allows identifying message pairs with the desired difference after π_1. Only those pairs are needed for subkey filtering. $t = 83.39$ gives approximately $2^{12.56}$ quartets on average, which implies $2 \cdot 2^{29.24} \cdot 2^{83.39} \simeq 2^{113.63}$ messages.

We employ Mennink's way of constructing tweaks. In each set, the tweaks iterate over $2^{83.39}$ values in the leftmost three anti-diagonals in the state X^5_{SR} before the $\mathsf{MixColumns}$ operation of Round 5 is applied to each tweak. We define that $\mu_0(i) : \mathbb{Z}_{2^{84}} \to (\mathbb{F}_{2^8})^{4\times4}$ encodes the integer i into the 12 bytes 0, 1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 15, from most to least significant bits and define $\mu_1(j) : \mathbb{Z}_{2^{84}} \to (\mathbb{F}_{2^8})^{4\times4}$ encodes $(j \lll 12)$ (left shift by 12 bits) into the 12 bytes 0, 1, 2, 4, 5, 7, 8, 10, 11, 13, 14 , 15, from most to least significant bits. This is illustrated in Figs. 4 and 5. In total, we need $2 \cdot 2^s \cdot 2^t \simeq 2^{113.63}$ message-tweak pairs.

3.4 Procedure

The attack proceeds as follows:

1. Zeroize 2^{2s} counters $N^{i,j}_{\mathsf{quartets}}$, and prepare lists $\mathcal{L}^0$, $\mathcal{D}^0$, $\mathcal{L}^1$, and $\mathcal{D}^1$. Initialize a list $\mathcal{K}$ of 2^{32} true flags that represent the values of $K^0[0, 5, 10, 15]$.
2. Construct the messages M^i and tweak sets $\mathcal{T}^i$ as described above and ask for the encryption of all tweak-message tuples. Each message-tweak set can be considered separately.
3. For 2^s messages M^i, $0 \le i < 2^s$:
   - 3.1 Call the first loop of the parallel-road distinguisher. For tweak set $\mathcal{T}^i$, store the results into $\mathcal{L}^{0,i}[c^{0,i}_k]$, for all $0 \le k < 2^t$. The 2^{2t-n-1} pairs are stored in $\mathcal{D}^{0,i}$.
4. For 2^s messages M^j, $2^s \le j < 2^{s+1}$:
   - 4.1 Call the second loop of the parallel-road distinguisher and store their results into $\mathcal{L}^{1,j}[c^{1,j}_k]$ for each tweak set $\mathcal{T}^j$ and $0 \le k < 2^t$. On average, 2^{2t-n-1} ciphertext pairs per tweak set need lookups in $\mathcal{D}^{1,j}$.
   - 4.2 For each message M^i:
     - i. Look up $\mathcal{D}^{0,i}$ for matches of the tweak difference. Increase the counter $N^{i,j}_{\mathsf{quartets}}$ if there are matches.

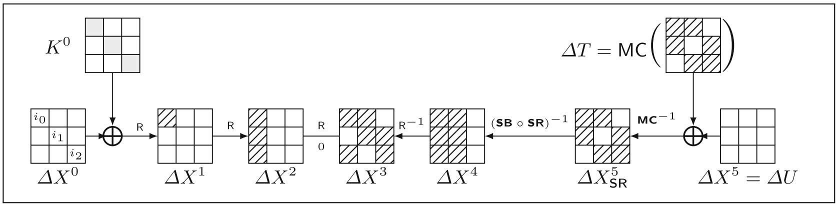


Fig. 6. Key recovery and impossible differential trail through $1 + 4$ rounds of Small-AES 36. Hatched bytes are active; filled bytes are targeted key bytes; indices in bytes denote that a set index is encoded into them.

5. For all counters $N^{i,j}_{\mathsf{quartets}}$ that are above the threshold θ, derive the $4 \cdot 2^8 \simeq 2^{10}$ round-key candidates $K^0[0, 5, 10, 15]$ that would encrypt $M^i \oplus M^j$ to a single-byte difference after the first round.
6. For all round-key candidates set the corresponding entry in $\mathcal{K}$ to false.
7. Output the entries of $\mathcal{K}$ that are still marked as true.

3.5 Computational and Memory Complexity

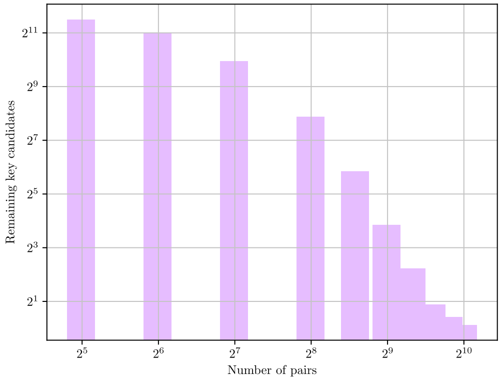
The total computational complexity is given by

1. $2 \cdot 2^{29.24} \cdot 2^{83.39} \simeq 2^{113.63}$ encryptions.
2. About $2^{29.24} \cdot 2^{83.39} \simeq 2^{112.63}$ memory insertions and lookups to obtain all pairs of equal ciphertexts in the sets $\mathcal{T}^{0,i}$ that are used to fill $\mathcal{D}^{0,i}$.
3. About $2^{29.24} \cdot 2^{83.39} \simeq 2^{112.63}$ memory insertions and lookups to obtain all pairs of equal ciphertexts in the sets $\mathcal{T}^{1,j}$.
4. About $2^s \cdot 2^{2t-1-n} \simeq 2^{29.24} \cdot 2^{2 \cdot 83.39-1-128} \simeq 2^{67}$ lookups into the sets $\mathcal{D}^{0,i}$.
5. We expect to have an advantage of at least $a \simeq 32$ bits. Thus, there will be at most 2^{96} remaining key candidates on average.

Thus, we have $2^{113.63} + 2^{96} \simeq 2^{113.63}$ encryptions and $2^{112.63} + 2^{112.63} + 2^{80.1} \simeq 2^{113.63}$ memory accesses. The memory complexity is upper bounded by storing $2^{112.63}$ ciphertext-tweak tuples in the lists $\mathcal{L}^{0,i}$ and $\mathcal{L}^{1,j}$ each and the same amount of tweak differences in $\mathcal{D}^{0,i}$ and $\mathcal{D}^{1,j}$, which is upper bounded by the memory for $2^{113.63}$ states and 2^{32} key candidates.

3.6 Experiments

For verification purposes, we considered a reduced version of the AES. A natural starting point is the 64-bit version, Small-AES [CMR05], where each cell is an element in $\mathbb{F}_{2^4}$. Since the complexity of $O(2^{3n/4}) = O(2^{48})$ operations and memory, multiplied by 100 keys is still hardly feasible, we reduced the cipher further to a 3×3-matrix structure of cells with 36-bit state, which we will denote as Small-AES36. We borrow almost all components from Small-AES, except for the MixColumns operation. In Small-AES36, MixColumns employs the circulant MDS matrix $\mathsf{circ}(1, 1, 2)$, with elements in the field $\mathbb{F}_{2^4}/p(\mathtt{x})$ with



	Key recovery		
#Pairs	μ	σ	a
32	2897.14	128.49	0.50
64	2025.67	131.33	1.02
128	992.78	90.57	2.04
256	234.67	38.19	4.13
384	57.34	13.82	6.16
512	14.44	5.25	8.15
640	4.71	2.30	9.76
768	1.85	0.97	11.11
896	1.34	0.66	11.58
1024	1.09	0.28	11.88

Fig. 7. Mean (μ) and standard deviation (σ) for the number of key candidates, as well as the advantage in bits (a), for 100 experiments of SMALL-AES36 each with varying numbers of message pairs with the desired difference ΔT after π_1 and random keys.

$p(\mathtt{x}) = (\mathtt{x}^4 + \mathtt{x} + \mathtt{1})$. We verified that the matrix is MDS in the given field with a python script.

The key-recovery phase targets the first diagonal of the first round key K^0. We iterate over all 2^{12} messages of the first diagonal and consider all message pairs (M^i, M^j) for distinct i, j that yield more than θ collisions for filtering. Each set $\mathcal{T}^{i,0}$ employs 2^t tweaks. Again, we use a variant of Mennink's tweak encoding: The t-bit tweaks $\langle i \rangle_{24} = (i_0, i_1, i_2, i_3, i_4, i_5)$ are encoded as $\mathsf{MC}(i_0, i_1, 0, i_2, 0, i_3, 0, i_4, i_5)$ in the cells 0-8, as shown in Fig. 6.

Expected Number of Messages. We experimented with varying numbers of message pairs that fulfilled the desired tweak differences ΔT. The results are illustrated in Fig. 7. We experimented with 1 000 random keys and 2^{12} messages that iterated over all values of the first diagonal and used a random value of the other cells. On average, we observed approximately $2^{11.1}$ message pairs with the desired difference after π_1, which yielded a probability of $2^{-11.9} \simeq 2^{12}$ that matches our expectation since we have 12 bit conditions in ΔX^5_{SR}.

Expected Number of Quartets. The distribution of quartets among message pairs with and without the desired difference is shown in Table 3.

Recall Eqs. (3) and (4). In our reduced AES version, we have a 24-bit tweak space, which must replace the $3n/4$ terms in those equations. In the following, we use $2^t = 2^{24+x}$. First, assume that $t \leq 24$ for a message pair that does **not** fulfill the correct difference after π_1. Then, we can combine $\binom{2^t}{2}$ tweaks pairs for one message and obtain 2^{-n} pairs that collide in their ciphertexts. We can combine those pairs for both messages to quartets, and have a probability of 2^{-24} that the tweak differences match for both pairs. If $t > 24$, we have $\binom{2^{24+x}}{2} \cdot 2^{-n}$ pairs per message whose ciphertexts collide. Building quartets, their tweak differences will match with probability $2^{-x} \cdot 2^{-24}$. Hence, we obtain

$$\begin{cases} (2^{2t-n-1})^2 \cdot 2^{-24} \simeq 2^{4t-96-2} \simeq 2^{4t-98} & \text{if } t \leq 24 \\ (2^{2t-n-1})^2 \cdot 2^{-x-24} \simeq 2^{4t-96-2-x} \simeq 2^{4t-98-x} & \text{otherwise.} \end{cases} \tag{7}$$

For a message pair that produces the desired difference after π_1, we have $2^{t-x} \cdot 2^{t-x}$ tweaks in their tweak sets that lead to a collision with probability 2^{-24} after π_1, and thus to $2^{2t-2x-24}$ pairs. Note that we can combine only the tweak sets that share the same 12-bit value in the anti-diagonal $\mathsf{MC}^{-1}(\varDelta T)[2,4,6]$. If $t = 24+x$ for non-negative x, there are 2^x times such pairs on average: $2^{2t-2x-24}$ for every value in the anti-diagonal, assuming 2^x is integer. Thus, we have

$$\begin{cases} \binom{2^{2t-24}}{2} \cdot 2^{-n} \simeq 2^{4t-48-36-1} \simeq 2^{4t-85} & \text{if } t \leq 24 \\ \binom{2^x \cdot 2^{2t-2x-24}}{2} \cdot 2^{-n} \simeq 2^{4t-2x-48-36-1} \simeq 2^{4t-2x-85} & \text{otherwise.} \end{cases} \tag{8}$$

quartets. For the messages with the desired difference after π_1, we observe approximately 2^3, 2^7, 2^{11}, 2^{13}, 2^{15}, and 2^{17} quartets with the standard deviation matching about the square root, for 2^t message-tweak tuples per message, and $t \in \{22, \ldots, 27\}$. This matches our expectations in Eq. (8) including the break at $t = 24$. For $t \leq 24$, one can observe an increasing factor of 2^4 quartets for each increment of t, which becomes 2^2 for $t > 24$.

For message pairs without the desired difference after π_1, the numbers of quartets are far below those of pairs with the desired difference, with means of 2^{-10}, 2^{-6}, 2^{-2}, 2^1, and 2^4, and 2^7. Again, the factor from t to $t+1$ changes from 2^4 if $t \leq 24$, to a factor of 2^3 times more quartets from t to $t+1$ when $t > 24$, as expected.

The standard deviations are about the square root of the expectations, which matches Bernoulli distributions. The major insight is that the gap in the number of quartets is huge enough – in the order of 2^{13}, 2^{12}, and 2^{11} for $t = 24, 25, 26$ – to reasonably choose a threshold and not have a single non-desired message pair that could mistakenly filter out the correct partial key.

4 Provable Security Preliminaries

4.1 Provable Security Notations

Given a sequence $\mathcal{X} = (X_1, \ldots, X_q)$, we use $\mathcal{X}^q$ to indicate that it consists of q elements; $\widehat{\mathcal{X}}^q = \{X_1, \ldots, X_q\}$ denotes their set and $\mu(\mathcal{X}^q, X)$ the multiplicity of an element X in $\mathcal{X}^q$. For an index set $\mathcal{I} \subseteq [q]$ and $\mathcal{X}^q$, $\mathcal{X}^{\mathcal{I}} =^{\text{def}} (X_i)_{i \in \mathcal{I}}$. For a pair of sequences $\mathcal{X}^q$ and $\mathcal{Y}^q$, $(\mathcal{X}^q, \mathcal{Y}^q)$ denotes the two-ary q-tuple $((X_1, Y_1), \ldots, (X_q, Y_q))$. An n-ary q-tuple is defined naturally. A two-ary tuple $(\mathcal{X}^q, \mathcal{Y}^q)$ is said to be permutation-compatible, denoted as $\mathcal{X}^q \leftrightsquigarrow \mathcal{Y}^q$, iff $X_i = X_j \Leftrightarrow Y_i = Y_j$. A three-ary tuple $(\mathcal{T}^q, \mathcal{X}^q, \mathcal{Y}^q)$ is said to be tweakable-permutation-compatible, denoted as $(\mathcal{T}^q, \mathcal{X}^q) \leftrightsquigarrow (\mathcal{T}^q, \mathcal{Y}^q)$, iff $(T_i, X_i) = (T_j, X_j) \Leftrightarrow (T_i, Y_i) = (T_j, Y_j)$. For any function $F : \mathcal{X} \to \mathcal{Y}$ and $\mathcal{X}^q$, $F(\mathcal{X}^q)$ denotes $(F(X_i), \ldots, F(X_q))$. For a set $\mathcal{X}$, $X \twoheadleftarrow \mathcal{X}$ means that X is sampled uniformly at random and independently from other variables from $\mathcal{X}$. Moreover, let $\exists^*$ mean "there exist distinct".

Table 3. Probabilities (μ) and standard deviations (σ) for #quartets of messages with the desired difference after π_1, from m experiments with random keys each and 2^t distinct tweaks per message.

		With desired difference?			
		With		Without	
t	m	$\log_2(\mu)$	$\log_2(\sigma)$	$\log_2(\mu)$	$\log_2(\sigma)$
22	10 000	2.994	1.511	−10.480	−5.241
23	1 000	6.997	3.550	−6.158	−2.991
24	100	11.005	5.502	−1.837	−0.907
25	100	12.998	6.479	1.233	0.664
26	100	15.001	7.437	3.986	2.097
27	100	17.002	8.395	6.987	3.497

A distinguisher $\mathbf{A}$ is an algorithm that tries to distinguish between two worlds $\mathcal{O}_{\text{real}}$ and $\mathcal{O}_{\text{ideal}}$ via black-box interaction with one of them chosen randomly and invisible from $\mathbf{A}$. At the end of its interaction, $\mathbf{A}$ has to output a decision bit. $\mathbf{Adv}_{\mathcal{O}_{\text{ideal}};\mathcal{O}_{\text{real}}}(\mathbf{A})$ denotes the advantage of $\mathbf{A}$ to distinguish between both. We consider information-theoretic distinguishers that are bounded only in terms of the number of queries and message material that they can ask to the available oracles. $\mathbf{Adv}_{\mathcal{O}_{\text{ideal}};\mathcal{O}_{\text{real}}}(q) \stackrel{\text{def}}{=} \max_{\mathbf{A}} \left\{ \mathbf{Adv}_{\mathcal{O}_{\text{ideal}};\mathcal{O}_{\text{real}}}(\mathbf{A}) \right\}$ denotes the maximum of advantages over all possible adversaries $\mathbf{A}$ that are allowed to ask at most q queries to its oracles. Later, we exclude trivial distinguishers, i.e., distinguishers who ask duplicate queries or queries to which the answer is already known.

4.2 Expectation Method

Let $\mathbf{A}$ be a computationally unbounded deterministic distinguisher that tries to distinguish between a real world $\mathcal{O}_{\text{real}}$ and an ideal world $\mathcal{O}_{\text{ideal}}$. The queries and responses of the interaction of $\mathbf{A}$ with its oracles are collected in a transcript τ. It may also contain additional information which would make the adversary only stronger. By Θ_{real} and Θ_{ideal}, we denote random variables for the transcript when $\mathbf{A}$ interacts with the real world or the ideal world, respectively. Since $\mathbf{A}$ is deterministic, the probability of $\mathbf{A}$'s decision depends only on the oracle and the transcript. A transcript τ is called attainable if its probability in the ideal world is non-zero.

The expectation method is a generalization of the popular H-coefficient method by Patarin [Pat08], which is a simple corollary of the following result.

Lemma 1 (Expectation Method [HT16]). Let Ω be a set of all transcripts that can be partitioned into two disjoint non-empty sets of good transcripts, GoodT and bad transcripts, BadT. For some $\epsilon_{\mathsf{bad}} > 0$ and a non-negative function $\epsilon_{\mathsf{ratio}} : \Omega \to [0, \infty)$, suppose $\Pr[\Theta_{\text{ideal}} \in \textsc{BadT}] \leq \epsilon_{\mathsf{bad}}$ and for any $\tau \in \textsc{GoodT}$, it holds that $\Pr[\Theta_{\text{real}} = \tau]/\Pr[\Theta_{\text{ideal}} = \tau] \geq 1 - \epsilon_{\mathsf{ratio}}$. Then, for any distinguisher $\mathbf{A}$ that tries to distinguish between $\mathcal{O}_{\text{real}}$ and $\mathcal{O}_{\text{ideal}}$, it holds:

$$\mathbf{Adv}_{\mathcal{O}_{\text{ideal}};\mathcal{O}_{\text{real}}}(\mathbf{A}) \leq \epsilon_{\mathsf{bad}} + \mathbb{E}\left[\epsilon_{\mathsf{ratio}}(\Theta_{\text{ideal}})\right].$$

4.3 Mirror Theory

Patarin [Pat10] defined the Mirror Theory as an approach to estimate the number of solutions of a linear system of equalities and linear inequalities in cyclic groups. He followed a recursive sophisticated proof [Pat08,Pat10] that was brought to the attention of a wider audience by Mennink and Neves [MN17]. Jha and Nandi [JN20] revisited it for a tight proof of CLRW2 [LRW02]. We follow their description that itself referred to Mennink and Neves' interpretation of the Mirror theory. For $q \geq 1$, let $\mathcal{L}$ be a system of linear equations of the form

$$\{\, e_1 : U_1 \oplus V_1 = \lambda_1, \quad \ldots, \quad e_q : U_q \oplus V_q = \lambda_q \,\},$$

where U_i and V_i are the unknowns, λ_i the knowns, and $U_i, V_i, \lambda_i \in \mathbb{F}_2^n$. We denote their sets as $\mathcal{U}^q$ and $\mathcal{V}^q$, respectively. Moreover, $\mathcal{L}$ contains a set of inequalities that uniquely determine $\widehat{\mathcal{U}}^q$ and $\widehat{\mathcal{V}}^q$, respectively. We assume that $\widehat{\mathcal{U}}^q$ and $\widehat{\mathcal{V}}^q$ are indexed in arbitrary order by index sets $[q_u]$ and $[q_v]$, where $q_u = |\widehat{\mathcal{U}}^q|$ and $q_v = |\widehat{\mathcal{V}}^q|$. Then, we can define two surjective index maps

$$\varphi_u : \begin{cases} [q] \to [q_u] \\ i \to j \text{ iff } U_i = \widehat{U}_j. \end{cases} \qquad \varphi_v : \begin{cases} [q] \to [q_v] \\ i \to j \text{ iff } V_i = \widehat{V}_j. \end{cases}$$

Thus, $\mathcal{L}$ is uniquely determined by $(\varphi_u, \varphi_v, \lambda^q)$ and vice versa. Let $\mathcal{G}(\mathcal{L}) =^{\text{def}} ([q_u], [q_v], \mathcal{E})$ be a labeled bipartite graph corresponding to $\mathcal{L}$, where

$$\mathcal{E} \stackrel{\text{def}}{=} \{(\varphi_u(i), \varphi_v(i), \lambda_i) : i \in [q]\}$$

is the set of edges and λ_i the edge labels. Thus, each equation in $\mathcal{L}$ corresponds to a unique labeled edge if there exist no duplicate equations in $\mathcal{L}$. We need three definitions to use the fundamental theorem of the Mirror Theory.

Definition 1 (Cycle-freeness). We call $\mathcal{L}$ cycle-free iff $\mathcal{G}(\mathcal{L})$ is acyclic.

Definition 2 (Maximal Block Size). Two equations e_i and e_j for distinct i, j are in the same component iff the corresponding edges (vertices) in $\mathcal{G}(\mathcal{L})$ are in the same graph component. The size of any component $\mathcal{C} \in \mathcal{L}$, denoted $\xi(\mathcal{C})$, is given by the number of vertices in the corresponding component of $\mathcal{G}(\mathcal{L})$. The maximal component size of $\mathcal{G}(\mathcal{L})$ is denoted by $\xi_{\max}(\mathcal{L})$ or short by $\xi_{\max}$.

Definition 3 (Non-degeneracy). $\mathcal{L}$ is called non-degenerate iff there exists no path of length ≥ 2 in $\mathcal{G}(\mathcal{L})$ such that the labels along its edges sum to zero.

Theorem 1 (Fundamental Theorem of the Mirror Theory [Pat10]). Let $\mathcal{L}$ be a system of equations over the unknowns $(\mathcal{U}^q, \mathcal{V}^q)$ that is (i) cycle-free, (ii) non-degenerate, and (iii) possesses a maximal component size of $\xi_{\max}$ with $\xi^2_{\max} \cdot \max\{q_u, q_v\} \leq 2^n$. Then, the number of solutions $(U_1, \ldots, U_{q_u}, V_1, \ldots, V_{q_v})$ of $\mathcal{L}$, denoted as h_q, such that $U_i \neq U_j$ and $V_i \neq V_j$ for all $i \neq j$, satisfies

$$h_q \geq \frac{(2^n)_{q_u} \cdot (2^n)_{q_v}}{(2^n)^q}. \tag{9}$$

h_q is multiplied by a factor of $(1 - \epsilon)$ for some $\epsilon > 0$ at the end. For $\xi \geq 2$ and $\epsilon > 0$, we denote as the (ξ, ϵ)-restricted Mirror-Theory theorem the variant with $\xi_{\max} = \xi$ and $h_q \geq (1 - \epsilon) \cdot h^*_q$, where h^*_q is the right-hand side of Eq. (9).

4.4 Transcript Graph

For TNT, a transcript τ will consist of the queries and responses (T_i, M_i, C_i) as well as intermediate values. We will later use a transcript of TNT as the tuple of tuples $(\mathcal{T}^q, \mathcal{M}^q, \mathcal{C}^q, \mathcal{X}^q, \mathcal{Y}^q, \mathcal{V}^q)$ that will collect the values T_i, M_i, etc., for $1 \leq i \leq q$, respectively. The roles of the individual variables are shown in Fig. 9.

Given a transcript τ, a transcript graph is a graph-isomorphic unique bipartite representation of the mappings in τ. For our purpose, the relevant transcript graph will reflect the mappings of $\mathcal{X}^q$ and $\mathcal{U}^q$. The transcript τ is therefore isomorphic to a graph on $(\mathcal{X}^q, \mathcal{U}^q)$.

Definition 4. A transcript graph $\mathcal{G} = (\mathcal{X}^q, \mathcal{U}^q, \mathcal{E}^q)$ that is associated with $(\mathcal{X}^q, \mathcal{U}^q)$ is denoted as $\mathcal{G}(\mathcal{X}^q, \mathcal{U}^q)$ and defined as $\mathcal{X} =^{\text{def}} \{(X_i, 0) : i \in [q]\}$, $\mathcal{U} =^{\text{def}} \{(U_i, 1) : i \in [q]\}$, and $\mathcal{E} =^{\text{def}} \{((X_i, 0), (U_i, 1)) : i \in [q]\}$. A label λ_i is associated with the edge $((X_i, 0), (U_i, 1)) \in \mathcal{E}$.

The resulting graph may contain parallel edges. The 0 and 1 in $(X_i, 0)$ and $(U_i, 1)$ will be dropped for simplicity. If for distinct $i, j \in [q]$, it holds that $X_i = X_j$ (or $U_i = U_j$), we denote that as shared vertex $X_{i,j}$ (or $U_{i,j}$). Since there is a bijection of each edge $(X_i, U_i) \in \mathcal{E}$ to i, we can also represent the edge by i.

4.5 Extended Mirror Theory

Jha and Nandi [JN20] applied the mirror theory to the tweakable-permutation setting. We briefly recall their main result and the necessary notations.

In an edge-labeled bipartite graph $\mathcal{G} = (\mathcal{Y}, \mathcal{V}, \mathcal{E})$, an edge (Y, V, λ) is isolated iff both Y and V have degree one. A component $\mathcal{S} \subseteq \mathcal{G}$ is called a star iff $\xi(\mathcal{S}) \geq 3$ (recall that $\xi(\mathcal{S})$ is the number of vertices in $\mathcal{S}$) and there is a unique vertex $V \in \mathcal{S}$ with degree $\xi(\mathcal{S}) - 1$. V is called the center of $\mathcal{S}$. $\mathcal{S}$ is called a $\mathcal{Y}$-star (or $\mathcal{V}$-star) if its center $Y \in \mathcal{Y}$ (or $V \in \mathcal{V}$). Consider an equation system $\mathcal{L}$

$$\{e_1 : Y_1 \oplus V_1 = \lambda_1, \quad e_2 : Y_2 \oplus V_2 = \lambda_2, \quad \ldots, \quad e_q : Y_q \oplus V_q = \lambda_q\},$$

such that each component in $\mathcal{G}(\mathcal{L})$ is either an isolated edge or a star. Let c_1, c_2, and c_3 denote the number of isolated, $\mathcal{Y}$-star, and $\mathcal{V}$-star components, respectively. Moreover, $q_1 = c_1$, q_2, and q_3 denote the number of their equations. The equations in $\mathcal{L}$ can be arranged in arbitrary order. The isolated edges are indexed first, followed by the star components. Jha and Nandi show the following:

Theorem 2 (Theorem 5.1 in [JN20]). Let $\mathcal{L}$ be as above with $q < 2^{n-2}$ and $\xi_{\max} q \leq 2^{n-1}$. Then, the number of tuples $(\mathcal{Y}^{q_Y}, \mathcal{V}^{q_V})$ that satisfy $\mathcal{L}$ with $Y_i \neq Y_j$ and $V_i \neq V_j$ for all $i \neq j$ satisfies

$$h_q \geq \left(1 - \frac{13q^4}{2^{3n}} - \frac{2q^2}{2^{2n}} - \left(\sum_{i=1}^{c_2+c_3} \eta_{c_i+1}^2\right) \frac{4q^2}{2^{2n}}\right) \cdot \frac{(2^n)_{q_1+c_2+q_3} \cdot (2^n)_{q_1+q_2+c_3}}{\prod_{\lambda' \in \lambda^2} (2^n)_{\mu(\lambda^q, \lambda')}},$$

where $\eta_j = \xi_j - 1$ and ξ_j denotes the number of vertices of the j-th component for $j \in [c_1 + c_2 + c_3]$.

4.6 Universal Hashing

Let $\mathcal{X}$ and $\mathcal{Y}$ be non-empty sets or spaces in the following, and let $\mathcal{H} = \{H | H : \mathcal{X} \to \mathcal{Y}\}$ be a family of hash functions.

Definition 5 (Almost-Universal Hash Function [CW79]). We say that $\mathcal{H}$ is ϵ-almost-universal (ϵ-AU) if, for all distinct $X, X' \in \mathcal{X}$, it holds that $\Pr[H(X) = H(X')] \leq \epsilon$, where the probability is taken over $H \twoheadleftarrow \mathcal{H}$.

Definition 6 (Almost-XOR-Universal Hash Function [Kra94,Rog95]). Let $\mathcal{Y} \subseteq \mathbb{F}_2^*$. We say that $\mathcal{H}$ is ϵ-almost-XOR-universal (ϵ-AXU) if, for all distinct $X, X' \in \mathcal{X}$ and arbitrary $\Delta \in \mathcal{Y}$, it holds that $\Pr[H(X) \oplus H(X') = \Delta] \leq \epsilon$, where the probability is taken over $H \twoheadleftarrow \mathcal{H}$.

Let $\mathcal{H} : \{H | H : \mathcal{T} \to \mathbb{F}_2^n\}$ be a family ϵ-almost-universal hash functions and $H \twoheadleftarrow \mathcal{H}$ be an instance. Let $\mathcal{X}^q =^{\text{def}} H(\mathcal{T}^q)$ be the sequence of outputs X_i from $H(T_i)$, for $i \in [q]$ queries. In the following, [JN20] defined, in an abstract way, variables ν_i for the number of occurrences of the hash value i, and defined coll for the number of colliding pairs in $\mathcal{X}^q$.

Lemma 2 (Lemma 4.3 in [JN20]). Since $\mathbb{E}[\mathsf{coll}] \leq \binom{q}{2}\epsilon$, it holds that

$$\mathbb{E}\left[\sum_{i=1}^{r} \nu_i^2\right] = 2 \cdot \mathbb{E}[\mathsf{coll}] + \sum_{i=1}^{r} \nu_i \leq 4 \cdot \mathbb{E}[\mathsf{coll}] \leq 2q^2\epsilon.$$

Thus, Lemma 2 says that the number of collisions is limited by $2q^2\epsilon$ on expectation. Furthermore, the corollary below upper bounds the number of occurrences of any single hash value. The proof in [JN20] stems from Markov's inequality.

Corollary 1 (Corollary 4.1 in [JN20]). Let $\nu_{\max} = \max\{\nu_i : i \in [r]\}$. Then, for some $a \geq 1$, it holds that $\Pr[\nu_{\max} \geq a] \leq \frac{2q^2\epsilon}{a^2}$.

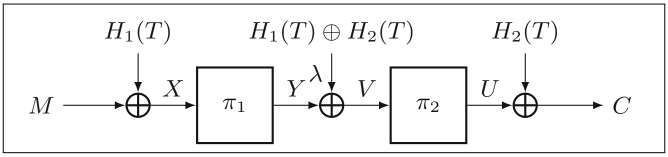


Fig. 8. CLRW2.

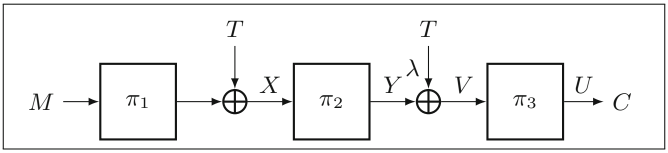


Fig. 9. TNT with relabeled variables.

The following lemma from [JN20] bounds the probability that four distinct inputs to two ϵ-AU hash functions yield three alternating collisions.

Lemma 3 (Alternating-collisions Lemma in [JN20]). Let $H_1, H_2 \twoheadleftarrow \mathcal{H}$ be independently sampled ϵ-AU hash functions with domain $\mathcal{X}$. Let $X_1, \ldots, X_q \in \mathcal{X}^q$ be pairwise distinct inputs. Then, it holds, over $H_1, H_2 \twoheadleftarrow \mathcal{H}$, that

$$\Pr\left[\exists^* i, j, k, \ell \in [q] : H_1(X_i) = H_1(X_j) \wedge H_2(X_j) = H_2(X_k) \wedge H_1(X_k) = H_1(X_\ell)\right]$$

is at most $q^2 \epsilon^{1.5}$.

5 TPRP Proof of TNT

We followed the footsteps of the STPRP proof of CLRW2 by [JN20] closely to show Theorem 3. We provide an extract that highlights where both constructions and proofs differ. Thus, we do not claim novelty of the proof approach but show that it applies also to TNT in encryption direction only with minor adaptions.

Theorem 3 (TPRP Security of TNT). Let $q \le 2^{n-2}$, and $E_{K_1}, E_{K_2}, E_{K_3} : \mathcal{K} \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be block ciphers with $K_1, K_2, K_3 \twoheadleftarrow \mathcal{K}$. Then,

$$\mathbf{Adv}^{\mathsf{TPRP}}_{\mathsf{TNT}[E_{K_1}, E_{K_2}, E_{K_3}]}(q) \le \frac{91q^4}{2^{3n}} + \frac{2q^2}{2^{2n}} + \frac{4q^2}{2^{1.5n}} + 3 \cdot \mathbf{Adv}^{\mathsf{PRP}}_{E}(q).$$

First, we can replace the secret-key block ciphers E_{K_1}, E_{K_2}, E_{K_3} with K_1, K_2, $K_3 \twoheadleftarrow \mathcal{K}$ by random permutations $\pi_1, \pi_2, \pi_3 \twoheadleftarrow \mathsf{Perm}(\mathbb{F}_2^n)$. For TNT, the advantage between both settings is upper bounded by

$$\mathbf{Adv}^{\mathsf{TPRP}}_{\mathsf{TNT}[E_{K_1}, E_{K_2}, E_{K_3}]} \le 3 \cdot \mathbf{Adv}^{\mathsf{PRP}}_{E}(q) + \mathbf{Adv}^{\mathsf{TPRP}}_{\mathsf{TNT}[\pi_1, \pi_2, \pi_3]}(q).$$

We consider the information-theoretic setting with a computationally unbounded distinguisher **A**. W.l.o.g., we assume that **A** is deterministic and non-trivial.

5.1 Oracle Descriptions

The Real Oracle. $\mathcal{O}_{\text{real}}$ runs $\mathsf{TNT}[\pi_1, \pi_2, \pi_3]$. The transcript random variable Θ_{real} yields the transcript as the tuple ($\mathcal{T}^q$, $\mathcal{M}^q$, $\mathcal{C}^q$, $\mathcal{X}^q$, $\mathcal{Y}^q$, $\mathcal{V}^q$) where for all queries $i \in [q]$, the values T_i, M_i, C_i, X_i, Y_i, V_i, U_i, λ_i refer to the variables as given in Fig. 9, which can be compared to those in CLRW2 in Fig. 8. The sets $\mathcal{U}^q = \mathcal{C}^q$ and $\lambda^q = \mathcal{T}^q$ can be derived directly from the transcript.

The Ideal Oracle. $\mathcal{O}_{\text{ideal}}$ implements $\widetilde{\Pi} \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathbb{F}_2^n, \mathbb{F}_2^n)$. Moreover, we treat the first permutation and tweak addition in TNT as equivalent to the first hash function in CLRW2. Thus, the ideal oracle samples $\pi_1 \twoheadleftarrow \mathsf{Perm}(\mathbb{F}_2^n)$ and gives all values X_i to $\mathbf{A}$ after $\mathbf{A}$ finished its interactions but before it outputs its decision bit. The transcript looks as before, where T_i, M_i, C_i are the inputs and outputs from $C_i = \widetilde{\Pi}(T_i, M_i)$ or $M_i = \widetilde{\Pi}^{-1}(T_i, C_i)$, $\lambda_i = T_i$, $X_i \leftarrow \pi_1(M_i) \oplus T_i$, $U_i \leftarrow C_i$. The values of the sets $\mathcal{X}^q$, $\mathcal{U}^q$, and $\mathcal{T}^q$ are defined honestly.

Jha and Nandi [JN20] characterized so-called bad hash keys. Given the partial transcript ($\mathcal{T}^q, \mathcal{M}^q, \mathcal{C}^q, \mathcal{X}^q$) – plus for CLRW2 also the hash functions H_1 and H_2 – they defined a number of conditions when (H_1, H_2) where considered good or bad, respectively, and defined the sets $\mathcal{H}_{\mathsf{good}}$ and $\mathcal{H}_{\mathsf{bad}}$ for this purpose. While TNT omits hash functions, the predicates were not conditions on the hash keys but instead on equalities of internal variables that can also occur in TNT. Therefore, we consider their cases analogously. A hash key was defined to be bad iff one of the following predicates was true:

1. badH_1: $\exists^* i, j \in [q]$ such that $X_i = X_j \wedge U_i = U_j$.
2. badH_2: $\exists^* i, j \in [q]$ such that $X_i = X_j \wedge T_i = T_j$.
3. badH_3: $\exists^* i, j \in [q]$ such that $U_i = U_j \wedge T_i = T_j$.
4. badH_4: $\exists^* i, j, k, \ell \in [q]$ such that $X_i = X_j \wedge U_j = U_k \wedge X_k = X_\ell$.
5. badH_5: $\exists^* i, j, k, \ell \in [q]$ such that $U_i = U_j \wedge X_j = X_k \wedge U_k = U_\ell$.
6. badH_6: $\exists k \geq 2^n/2q$, $\exists^* i_1, i_2, \ldots, i_k \in [q]$ such that $X_{i_1} = \cdots = X_{i_k}$.
7. badH_7: $\exists k \geq 2^n/2q$, $\exists^* i_1, i_2, \ldots, i_k \in [q]$ such that $U_{i_1} = \cdots = U_{i_k}$.

In the absence of hash keys, we cannot label those as H being bad or good. Thus, we call them bad and good hash equivalent instead.

Bad Hash Equivalent: If one of the events badH_1 through badH_7 occurs, the ideal oracle samples the values $\mathcal{Y}^q$ and $\mathcal{V}^q$ as $Y_i = V_i = 0$ for all $i \in [q]$.

Good Hash Equivalent: In the other case, it will be useful to study the transcript graph $\mathcal{G}(\mathcal{X}^q, \mathcal{U}^q)$ of the associations $(\mathcal{X}^q, \mathcal{U}^q)$ that arises from the transcript when no badH event occurs. Figure 10 shows all possible types of components in $\mathcal{G}(\mathcal{X}^q, \mathcal{U}^q)$. There, (star) components of the Types (2) and (3) contain exactly one vertex with a degree of ≥ 2. Components of Types (4) and (5) can contain one vertex with a degree of ≥ 2 in $\mathcal{U}$ and one such vertex in $\mathcal{X}$.

Lemma 4 (Lemma 6.1 in [JN20]). The transcript graph $\mathcal{G}(\mathcal{X}^q, \mathcal{U}^q)$ ($\mathcal{G}$ for short, hereafter) by a good hash equivalent has the following properties:

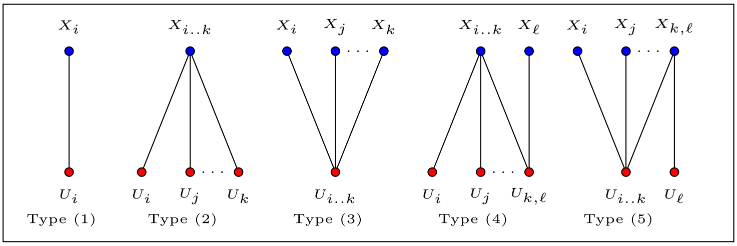


Fig. 10. Components types of a transcript graph corresponding to a good hash equivalent. Type (1) is the only component with a single edge. Types (2) and (3) are $\mathcal{X}$- and $\mathcal{U}$-star components, respectively. Types (4) and (5) are the only components that are neither isolated nor stars since they can have vertices of degree ≥ 2 in both $\mathcal{X}$ and $\mathcal{U}$.

1. $\mathcal{G}$ is simple, acyclic, and possesses no isolated vertices.
2. $\mathcal{G}$ has no two adjacent edges i and j such that $T_i \oplus T_j = 0$.
3. $\mathcal{G}$ has no component of size $\geq 2^n/2q$ edges.
4. $\mathcal{G}$ has no component with more than one vertex of degree ≥ 2 in neither $\mathcal{X}$ or $\mathcal{U}$ (though, it can have one vertex with degree ≥ 2 in $\mathcal{X}$ and one in $\mathcal{U}$).

The proof is given in [JN20].

For the sake of completeness, we describe the sampling process of $\mathcal{Y}^q$ and $\mathcal{V}^q$ in the case of a good hash equivalent. This is the same process as for CLRW2 in [JN20]. Therefore, this part is only a revisit and attributed to [JN20]:

The indices $i \in [q]$ are collected in index sets $\mathcal{I}_1, \ldots, \mathcal{I}_5$, corresponding to the edges in all Type-1, ..., Type-5 components, respectively. The five sets are disjoint and $[q] = \bigcup_{i=1}^{5} \mathcal{I}_i$. Let $\mathcal{I} = \bigcup_{i=1}^{3} \mathcal{I}_i$ and consider the system of equations

$$\mathcal{L} \stackrel{\text{def}}{=} \{Y_i \oplus V_i = T_i : i \in \mathcal{I}\},$$

where $Y_i = Y_j$ (respectively $V_i = V_j$) holds iff $X_i = X_j$ (respectively $U_i = U_j$) for all $i, j \in [q]$. The solution set of $\mathcal{L}$ is precisely the set

$$\mathcal{S} \stackrel{\text{def}}{=} \left\{(\mathcal{Y}^{\mathcal{I}}, \mathcal{V}^{\mathcal{I}}) : \mathcal{Y}^{\mathcal{I}} \leftrightsquigarrow \mathcal{X}^{\mathcal{I}} \wedge \mathcal{V}^{\mathcal{I}} \leftrightsquigarrow \mathcal{U}^{\mathcal{I}} \wedge \mathcal{Y}^{\mathcal{I}} \oplus \mathcal{V}^{\mathcal{I}} = \mathcal{T}^{\mathcal{I}}\right\}.$$

Given these definitions, the ideal-world oracle $\mathcal{O}_{\text{ideal}}$ samples $(\mathcal{Y}^q, \mathcal{V}^q)$ as follows:

- $(\mathcal{Y}^{\mathcal{I}}, \mathcal{V}^{\mathcal{I}}) \twoheadleftarrow \mathcal{S}$. This means, $\mathcal{O}_{\text{ideal}}$ samples uniformly one valid assignment from the set of all valid assignments.
- Let $\mathcal{G} \backslash \mathcal{I}$ denote the subgraph of $\mathcal{G}$ after the removal of edges and vertices corresponding to $i \in \mathcal{I}$. For each component $\mathcal{C} \subset \mathcal{G} \backslash \mathcal{I}$:
  - If $(X_i, U_i) \in \mathcal{C}$ corresponds to the edge in $\mathcal{C}$ where both X_i and U_i have a degree ≥ 2. Then, $Y_i \twoheadleftarrow \mathbb{F}_2^n$ and $V_i = Y_i \oplus T_i$.
  - For each edge $(X_{i'}, U_{i'}) \neq (X_i, U_i) \in \mathcal{C}$, either $X_{i'} = X_i$ or $U_{i'} = U_i$. Take the case that $X_{i'} = X_i$. Then, $Y_{i'} = Y_i$ and $V_{i'} = Y_{i'} \oplus T_{i'}$. In the other case $U_{i'} = U_i$. Then, $V_{i'} = V_i$ and $Y_{i'} = V_{i'} \oplus T_{i'}$.

Then, the transcript in the ideal world is completely defined, maintaining both the consistency of equations of the form $Y_i \oplus V_i = T_i$ as in the real world and the permutation consistency within each component for good hash equivalents. Still, there can be collisions among the values of $\mathcal{Y}$ or among the values of $\mathcal{V}$ from different components.

5.2 Definition of Bad Transcripts

The analysis of bad transcripts and of bad hash equivalents, in particular, is the core aspect wherein the analyses of CLRW2 and TNT differ. However, there can be collisions among the values of $\mathcal{Y}$ or among the values of $\mathcal{V}$ from different components that have to be treated in bad transcripts. Their treatment can be done similarly as in [JN20]. They are essential for the proof of TNT and listed in this subsection only for the sake of completeness, but we refer to [JN20] for their proof.

The set of transcripts Ω is the set of all tuples $\tau = (\mathcal{T}^q, \mathcal{M}^q, \mathcal{C}^q, \mathcal{X}^q, \mathcal{Y}^q, \mathcal{V}^q)$ defined as before. Recall that $\mathcal{U}^q = \mathcal{C}^q$ holds for TNT. Following [JN20], a bad transcript definition needs the following preprocessing steps:

1. Eliminate all tuples $(\mathcal{X}^q, \mathcal{U}^q, \mathcal{T}^q)$ such that both $\mathcal{Y}^q$ and $\mathcal{V}^q$ are trivially restricted by linear dependencies.
2. Eliminate all tuples $(\mathcal{X}^q, \mathcal{U}^q, \mathcal{V}^q, \mathcal{Y}^q)$ such that $\mathcal{X}^q \not\leftrightsquigarrow \mathcal{Y}^q$ or $\mathcal{U}^q \not\leftrightsquigarrow \mathcal{V}^q$.

A transcript τ is called a bad hash-equivalent transcript if one of the conditions badH_1 through badH_7 holds. We define a compound event $\mathsf{badH} =^{\text{def}} \bigcup_{i=1}^{7} \mathsf{badH}_i$ that ensures that the first requirement is fulfilled.

For the second requirement, all conditions that might lead to $\mathcal{X}^q \not\leftrightsquigarrow \mathcal{Y}^q$ or $\mathcal{U}^q \not\leftrightsquigarrow \mathcal{V}^q$ have to be addressed. The transcript is trivially inconsistent if one of them is fulfilled and we consider that badH does not hold in the following. If the transcript is still bad, it is called sampling-induced bad iff one of the following conditions from [JN20] holds, for some $\alpha \in \{1, \ldots, 5\}$ and $\beta \in \{\alpha, \ldots, 5\}$:

- $\mathsf{ycoll}_{\alpha,\beta}$: $\exists i \in \mathcal{I}_\alpha, j \in \mathcal{I}_\beta$ such that $X_i \neq X_j \wedge Y_i = Y_j$ and
- $\mathsf{vcoll}_{\alpha,\beta}$: $\exists i \in \mathcal{I}_\alpha, j \in \mathcal{I}_\beta$ such that $U_i \neq U_j \wedge V_i = V_j$,

where $\mathcal{I}_i$ is defined as before. It holds that

$$\mathsf{badsamp} \overset{\text{def}}{=} \bigcup_{\alpha \in [5], \beta \in \{\alpha, \ldots, 5\}} \left(\mathsf{ycoll}_{\alpha,\beta} \cup \mathsf{vcoll}_{\alpha,\beta} \right) .$$

By varying α and β over all 30 values, one obtains 30 conditions that could yield that $\mathcal{X}^q \not\leftrightsquigarrow \mathcal{Y}^q$ or $\mathcal{U}^q \not\leftrightsquigarrow \mathcal{V}^q$. Some of these conditions cannot be satisfied due to the sampling mechanism. Those are

$$\mathsf{ycoll}_{1,1}, \mathsf{ycoll}_{1,2}, \mathsf{ycoll}_{1,3}, \mathsf{ycoll}_{2,2}, \mathsf{ycoll}_{2,3}, \mathsf{ycoll}_{3,3},$$
$$\mathsf{vcoll}_{1,1}, \mathsf{vcoll}_{1,2}, \mathsf{vcoll}_{1,3}, \mathsf{vcoll}_{2,2}, \mathsf{vcoll}_{2,3}, \mathsf{vcoll}_{3,3}.$$

A transcript is called bad if it is a bad hash-equivalent or bad sampling-induced transcript. All other transcripts are called good and all good transcripts are attainable. It holds that

$$\Pr\left[\Theta_{\text{ideal}} \in \textsc{BadT}\right] \leq \Pr_{\Theta_{\text{ideal}}}\left[\mathsf{badH}\right] + \Pr_{\Theta_{\text{ideal}}}\left[\mathsf{badsamp}\right].$$

5.3 Analysis of Bad Transcripts

The analysis of bad transcript is the core point where the analysis of $\mathsf{CLRW2}$ and TNT differ. This is mainly because TNT lacks hash functions, but adds the unmodified tweak to the state between the permutation calls. As a result, hash collisions as in $\mathsf{CLRW2}$ cannot occur for distinct tweaks.

Lemma 5. For TNT, it holds in the ideal world that

$$\Pr\left[\mathsf{badH}\right] \leq \frac{4q^2}{2^{1.5n}} + \frac{32q^4}{2^{3n}}.$$

Proof. We study the probabilities of the individual events badH in the following. Prior, we note that $F(T_i, M_i) =^{\text{def}} \pi_1(M_i) \oplus T_i$ is ϵ-AU for $\epsilon \leq 1/(2^n - 1) \leq 2^{1-n}$, and at most $1/(2^n - (q-1))$ if $q-1$ values M_i had been queried before. Since $q \leq 2^{n-2}$, it holds that $\epsilon \leq 4/(3 \cdot 2^n)$.

- badH_1**.** This event holds if for some distinct i, j both $X_i = X_j$ and $U_i = U_j$. If $T_i = T_j$, it must hold that $M_i \neq M_j$, which implies that $X_i \neq X_j$ and the event cannot hold. If $T_i \neq T_j$, $X_i = X_j$ implies $Y_i = Y_j$ and $U_i = U_j$ implies $V_i = V_j$. Thus, it would have to hold that $T_i = T_j$, which is a contradiction. Hence, the probability is zero.
- badH_2**.** This event holds if for some distinct i, j both $X_i = X_j$ and $T_i = T_j$. Since $T = T$, it must follow that $M_i = M_j$. Though, since $\mathbf{A}$ does not ask duplicate queries, this implies that $X_i \neq X_j$. So, the probability is zero.
- badH_3**.** This event holds if for some distinct i, j both $U_i = U_j$ and $T_i = T_j$. Again, the latter condition implies that $M_i \neq M_j$. $U_i = U_j$ implies that $V_i = V_j$, which implies that $Y_i = Y_j$, $X_i = X_j$, and $\pi_1(M_i) = \pi_1(M_j)$, which is a contradiction and therefore has zero probability.
- badH_4**.** This event holds if for some distinct i, j, k, ℓ, $X_i = X_j$, $U_j = U_k$, and $X_k = X_\ell$. The values of X are results from an ϵ-universal hash function. The values U are sampled uniformly at random in the ideal world from a set of at least $2^n - q$ values for the current tweak. Thus, its sampling process can be interpreted to be ϵ-AU with $\epsilon \leq 1/(2^n - q)$. We can apply Lemma 3 to obtain

$$\Pr\left[\mathsf{badH}_4\right] \leq q^2\epsilon^{1.5} \leq \frac{4^{1.5}q^2}{(3 \cdot 2^n)^{1.5}} \leq \frac{2q^2}{2^{1.5n}}.$$

- badH_5**.** This event holds if for some distinct i, j, k, ℓ, $U_i = U_j$, $X_j = X_k$, and $U_k = U_\ell$. From a similar argumentation as for badH_4, it holds that

$$\Pr\left[\mathsf{badH}_5\right] \leq \frac{2q^2}{2^{1.5n}}.$$

- badH_6. This event holds if there exist distinct $i_1, \ldots, i_k \in [q]$ for $k \geq 2^n/2q$ such that $X_{i_1} = \cdots = X_{i_k}$. Since $(T_i, M_i) \neq (T_j, M_j)$ for none of the indices, we can use Corollary 1 with $a = 2^n/2q$ to upper bound it by

$$\Pr[\mathsf{badH}_6] \leq \frac{8q^4\epsilon}{2^{2n}} \leq \frac{16q^4}{2^{3n}}.$$

- badH_7. This event holds if there exist distinct $i_1, \ldots, i_k \in [q]$ for $k \geq 2^n/2q$ such that $U_{i_1} = \cdots = U_{i_k}$. From a similar argumentation as for badH_6, we get

$$\Pr[\mathsf{badH}_7] \leq \frac{16q^4}{2^{3n}}.$$

Lemma 5 follows then from the sum of probabilities of all badH events. □

Lemma 6. For TNT, it holds in the ideal world that

$$\Pr[\mathsf{badsamp}] \leq \frac{14q^4}{2^{3n}}.$$

The proof is exactly as in [JN20] and is deferred to the full version of this work.

5.4 Analysis of Good Transcripts

Lemma 7. For an arbitrary good transcript τ, it holds that

$$\frac{\Pr[\Theta_{\text{real}} = \tau]}{\Pr[\Theta_{\text{ideal}} = \tau]} \geq 1 - \frac{45q^4}{2^{3n}} - \frac{2q^2}{2^{2n}}.$$

Again, the proof can follow a similar argumentation as the analysis of good transcripts in [JN20] and is therefore deferred to the full version of this work.

6 Summary and Discussion

This work tried to conduct a step towards closing the security gap of TNT. We showed in Sect. 2 that a variant of Mennink's distinguisher from [Men18] also applies to TNT, which yields a theoretical distinguisher in $O(\sqrt{n} \cdot 2^{3n/4})$ time, data, and memory complexity. For this purpose, we reduce the complexity of Mennink's information-theoretic distinguisher from $O(2^{3n/2})$ to $O(2^{3n/4})$ computations and show that at least two similar distinguishers exist. Thereupon, we use the distinguisher to mount a partial key-recovery attack on the instance TNT-AES$[5, *, *]$ from an impossible differential. This attack is described in Sect. 3. Since it needs multiple pairs, its complexity is higher than $O(2^{3n/4})$. We emphasize that our analysis does not break the proposed version of TNT-AES$[6, 6, 6]$ from [BGGS20].

From a constructive point of view, we followed the rigorous analysis by Jha and Nandi on CLRW2. We show in Sect. 5 that their STPRP security proof of CLRW2 for up to $O(2^{3n/4})$ queries can be adapted to an TPRP proof of TNT with similar complexity. We could build on the approach by Jha and Nandi on CLRW2 since we restricted the adversary's queries to the forward direction only. Thus, the first permutation and tweak addition masks the inputs, similar to the first hash function in CLRW2. Since an equivalent is missing at the ciphertext side, one cannot directly derive STPRP security. However, a four-round variant of TNT would possess such hash-function-like masking at the ciphertext-side. This implies that a four-round variant that adds a fourth independent permutation π_4 and encrypts M under T as

$$\mathsf{TNT4}[\pi_1, \pi_2, \pi_3, \pi_4](T, M) \overset{\text{def}}{=} \pi_4(\pi_3(\pi_2(\pi_1(M) \oplus T) \oplus T) \oplus T),$$

would directly inherit the $O(2^{3n/4})$ STPRP security from CLRW2. Still, it remains a highly interesting work to conduct an STPRP analysis of the three-round construction TNT. In particular, the Mirror-theory approach seems not easily adaptable since the sampling process in the ideal world is unclear.

From our studies, we see strong indications that TNT is STPRP-secure for approximately $O(2^{3n/4})$ queries if the primitives are secure – although, we were not able to show it at this point of time. However, we found the problem of sampling the variables from both sides consistently in the middle non-trivial. An alternative strategy could be a more precise, but also considerably more sophisticated, study of the original χ^2-based proof of TNT from [BGGS20].

Acknowledgments. We are highly thankful to Zhenzhen Bao and Mridul Nandi for the fruitful discussions as well as the reviewers from Asiacrypt 2020 for their inspiring comments, all of which lead to significant improvements in this work. This research has been partially supported by Nanyang Technological University in Singapore under Grant 04INS000397C230, Singapore's Ministry of Education under Grants RG18/19 and MOE2019-T2-1-060, the National Natural Science Foundation of China (No. 61961146004, 61802399, 61802400, 61732021 and 61772519) and the Youth Innovation Promotion Association CAS.

References

[BGGS20] Bao, Z., Guo, C., Guo, J., Song, L.: TNT: how to tweak a block cipher. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12106, pp. 641–673. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45724-2_22

[BGIM19] Bao, Z., Guo, J., Iwata, T., Minematsu, K.: ZOCB and ZOTR: tweakable blockcipher modes for authenticated encryption with full absorption. IACR Trans. Symmetric Cryptol. **2019**(2), 1–54 (2019)

[BJK+16] Beierle, C., et al.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 123–153. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_5

[BKL+07] Bogdanov, A., et al.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74735-2_31

[BLMR19] Beierle, C., Leander, G., Moradi, A., Rasoolzadeh, S.: CRAFT: lightweight tweakable block cipher with efficient protection against DFA attacks. IACR Trans. Symmetric Cryptol. **2019**(1), 5–45 (2019)

[CIMN17] Chakraborti, A., Iwata, T., Minematsu, K., Nandi, M.: Blockcipher-based authenticated encryption: how small can we go? In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 277–298. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66787-4_14

[CIMN20] Chakraborti, A., Iwata, T., Minematsu, K., Nandi, M.: Blockcipher-based authenticated encryption: how small can we go? J. Cryptol. **33**(3), 703–741 (2019). https://doi.org/10.1007/s00145-019-09325-z

[CMR05] Cid, C., Murphy, S., Robshaw, M.J.B.: Small scale variants of the AES. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 145–162. Springer, Heidelberg (2005). https://doi.org/10.1007/11502760_10

[CW79] Carter, L., Wegman, M.N.: Universal classes of hash functions. J. Comput. Syst. Sci. **18**(2), 143–154 (1979)

[GRR16] Grassi, L., Rechberger, C., Rønjom, S.: Subspace trail cryptanalysis and its applications to AES. IACR Trans. Symmetric Cryptol. **2016**(2), 192–225 (2016)

[HT16] Hoang, V.T., Tessaro, S.: Key-alternating ciphers and key-length extension: exact bounds and multi-user security. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 3–32. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_1

[IMPS17] Iwata, T., Minematsu, K., Peyrin, T., Seurin, Y.: ZMAC: a fast tweakable block cipher mode for highly secure message authentication. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 34–65. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_2

[JLM+17] Jha, A., List, E., Minematsu, K., Mishra, S., Nandi, M.: XHX – a framework for optimally secure tweakable block ciphers from classical block ciphers and universal hashing. In: Lange, T., Dunkelman, O. (eds.) LATINCRYPT 2017. LNCS, vol. 11368, pp. 207–227. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-25283-0_12

[JN20] Jha, A., Nandi, M.: Tight security of cascaded LRW2. J. Cryptol. **33**, 1272–1317 (2020). https://doi.org/10.1007/s00145-020-09347-y

[JNP14a] Jean, J., Nikolić, I., Peyrin, T.: KIASU v1.1 (2014). First-round submission to the CAESAR competition

[JNP14b] Jean, J., Nikolić, I., Peyrin, T.: Tweaks and keys for block ciphers: the `TWEAKEY` framework. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 274–288. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45608-8_15

[JNP16] Jean, J., Nikolić, I., Peyrin, T.: Deoxys v1.41, 12 October 2016. Third-round submission to the CAESAR competition. http://competitions.cr.yp.to/caesar-submissions.html

[KR11] Krovetz, T., Rogaway, P.: The software performance of authenticated-encryption modes. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 306–327. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21702-9_18

[Kra94] Krawczyk, H.: LFSR-based hashing and authentication. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 129–139. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48658-5_15

[Lea10] Leander, G.: Small scale variants of the block cipher PRESENT. IACR Cryptology ePrint Archive 2010/143 (2010)

[LL18] Lee, B.H., Lee, J.: Tweakable block ciphers secure beyond the birthday bound in the ideal cipher model. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018. LNCS, vol. 11272, pp. 305–335. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03326-2_11

[LN17] List, E., Nandi, M.: ZMAC+ - an efficient variable-output-length variant of ZMAC. IACR Trans. Symmetric Cryptol. **2017**(4), 306–325 (2017)

[LNS18] Leurent, G., Nandi, M., Sibleyras, F.: Generic attacks against beyond-birthday-bound MACs. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 306–336. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_11

[LRW02] Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 31–46. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_3

[LST12] Landecker, W., Shrimpton, T., Terashima, R.S.: Tweakable blockciphers with beyond birthday-bound security. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 14–30. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_2

[Men15] Mennink, B.: Optimally secure tweakable blockciphers. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 428–448. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48116-5_21

[Men18] Mennink, B.: Towards tight security of cascaded LRW2. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11240, pp. 192–222. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03810-6_8

[Min14] Minematsu, K.: Parallelizable rate-1 authenticated encryption from pseudorandom functions. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 275–292. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_16

[MN17] Mennink, B., Neves, S.: Encrypted Davies-Meyer and its dual: towards optimal security using mirror theory. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 556–583. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_19

[Nai15] Naito, Y.: Full PRF-secure message authentication code based on tweakable block cipher. In: Au, M.-H., Miyaji, A. (eds.) ProvSec 2015. LNCS, vol. 9451, pp. 167–182. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26059-4_9

[Nai18] Naito, Y.: On the efficiency of ZMAC-type modes. In: Camenisch, J., Papadimitratos, P. (eds.) CANS 2018. LNCS, vol. 11124, pp. 190–210. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00434-7_10

[Pat08] Patarin, J.: The "coefficients H" technique. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 328–345. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04159-4_21

[Pat10] Patarin, J.: Introduction to mirror theory: analysis of systems of linear equalities and linear non equalities for cryptography. IACR Cryptology ePrint Archive 2010/287 (2010)

[PS16] Peyrin, T., Seurin, Y.: Counter-in-tweak: authenticated encryption modes for tweakable block ciphers. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 33–63. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_2

[Rog95] Rogaway, P.: Bucket hashing and its application to fast message authentication. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 29–42. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-44750-4_3

[Sib20] Sibleyras, F.: Generic attack on iterated tweakable FX constructions. In: Jarecki, S. (ed.) CT-RSA 2020. LNCS, vol. 12006, pp. 1–14. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-40186-3_1

[SS17] Samajder, S., Sarkar, P.: Rigorous upper bounds on data complexities of block cipher cryptanalysis. J. Math. Cryptol. **11**(3), 147–175 (2017)

[WGZ+16] Wang, L., Guo, J., Zhang, G., Zhao, J., Gu, D.: How to build fully secure tweakable blockciphers from classical blockciphers. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10031, pp. 455–483. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53887-6_17