Plaintext Recovery Attacks against XTS Beyond Collisions

Takanori Isobe[1,3] and Kazuhiko Minematsu[2]

[1] University of Hyogo, Japan. takanori.isobe@ai.u-hyogo.ac.jp
[2] NEC Corporation, Japan. k-minematsu@ah.jp.nec.com
[3] National Institute of Information and Communications Technology, Japan.

Abstract. XTS is an encryption scheme for storage devices standardized by IEEE and NIST. It is based on Rogaway's XEX tweakable block cipher and is known to be secure up to the collisions between the blocks, thus up to around $2^{n/2}$ blocks for n-bit blocks. However this only implies that the theoretical indistinguishability notion is broken with $O(2^{n/2})$ queries and does not tell the practical risk against the plaintext recovery if XTS is targeted. We show several plaintext recovery attacks against XTS beyond collisions, and evaluate their practical impacts.

Keywords: XTS, Storage encryption, Mode of operation, Even-Mansour Cipher

1 Introduction

XTS is a symmetric-key encryption scheme for storage devices such as HDD or USB memory sticks. It has been developed by IEEE Storage in Security Workgroup (SISWG) in 2007, based on a block cipher mode called XEX proposed by Rogaway in 2004 [19]. It has been standardized as IEEE standard for storage encryption (IEEE P1619) [11]. In 2010, NIST specified XTS as one of the recommended schemes with the explicit use of AES as the underlying block cipher. This scheme is called XTS-AES and is described in NIST SP800-38E [5] with a parameter restriction not presented in the IEEE document. XTS is quite widely deployed, such as Bitlocker, dm-crypt, and Truecrypt and its successors.

Since XTS is built on XEX, the security of XTS is basically inherited from that of XEX. Assuming the underlying n-bit block cipher is secure, XEX is provably secure as long as the number of processed blocks is sufficiently smaller than $2^{n/2}$ [19, 15]. This bound comes from a collision between two inputs to the block cipher, which is expected to happen with high probability for $2^{n/2}$ blocks due to the birthday paradox. Hence it is called "birthday bound". The security analysis of XTS largely follows this result, except the case that the last block in a sector is partial (i.e. not n bits) for which a variant of the classical ciphertext stealing is applied.

Most popular block cipher modes are secure up to the birthday bound. This implies that if we take n large enough the scheme is secure in practice. However, it is also important to study what will happen if the attacker can perform $2^{n/2}$

encryptions, that is, security analysis beyond the birthday bound. This allows to learn the limit of key lifetime and the danger of small-block ciphers. In the similar context, the security of CBC and CTR modes beyond the birthday bound have been studied [1, 13].

Our Contribution. In this paper, we study the security of XTS beyond the birthday bound, in particular, the security against plaintext recovery. We note that it is not hard to derive a collision-based distinguishing attack against XEX thus XTS, however, this only implies the tightness of the security bound. The collision attack needs $O(2^{n/2})$ encrypted blocks for one sector, and only reveals the block cipher mask used by this sector. Once the mask is known, XTS largely reduces to the basic ECB mode, hence the attack trivially reveals (a part of) plaintexts of the target sector when they are chosen or known with a low entropy. Unfortunately, this observation does not tell anything beyond, say how easy to recover the plaintext stored in sectors that was not the target of the collision attack. This is the problem we want to study.

Under a reasonable adversary model for storage encryption (e.g. [6, 12]) we derived several plaintext recovery attacks against XTS. Specifically, we classify the sectors of an XTS-encrypted storage into two categories, called reference sectors and target sectors. At the reference sectors, the adversary can encrypt a known plaintext and decrypt any ciphertext, that is, a combination of known-plaintext attack (KPA) and chosen-ciphertext attack. At the target sectors, the adversary can only perform a ciphertext-only attack or a partially-known-plaintext attack, where the definition of "partial" depends on the attack. The goal is to recover the plaintext at one of the target sectors for the corresponding ciphertext obtained by encryption queries to that sector.

Our attacks are not a trivial application of collision attack described above in that it does not need $O(2^{n/2})$ encrypted blocks of the sector for which the target plaintext is stored. The key observation of our attacks is a similarity between XTS and single-key Even-Mansour (SEM) ciphers: it allows us to convert attacks against SEM into XTS. Specifically, we show that once a mask is recovered by the collision attack at the reference sector, XTS at target sectors can be seen as a variant of SEM. Then, we propose plaintext-recovery attacks beyond collisions at target sectors in several practical settings. In a partially-known plaintext setting where a part of plaintext blocks at the target sector is known, e.g. fixed header files, we are able to recover the remaining unknown plaintext blocks by a variant of key recovery attacks on SEM [4]. For a 64-bit block cipher, our plaintext-recovery attack is feasible with only 2^8 known blocks in the target sectors, and 2^{56} local computations independently from the key size of the underlying block cipher. Besides, we show that this attack works with the almost same attack complexity even if there is no blocks that is completely known, e.g. only one byte in a block is known. Finally, we show that in the ciphertext-only setting where the adversary does not have any information of plaintext at target sectors, we are able to guess a target plaintext with a higher probability than random guessing.

Table 1. Summary of our attacks in the several attack settings.

Attack Setting	Attack Type	$n = 64$			$n = 128$		
		Time	Data	Memory	Time	Data	Memory
pKPA 1 (Sec. 7.1)	Plaintext Recovery	2^{56}	2^{8}	2^{8}	2^{96}	2^{32}	2^{32}
pKPA 2 (Sec. 7.2)	Plaintext Recovery	2^{55}	2^{10}	2^{57}	2^{98}	2^{32}	2^{122}
pKPA 2 (Sec. 7.2)	Plaintext Recovery	2^{57}	2^{10}	2^{10}	2^{120}	2^{32}	2^{32}
Ciphertext-only (Sec. 7.3)	Plaintext Recovery	2^{60}	2^{60}	2^{60}	2^{124}	2^{124}	2^{124}

pKPA 1: Partially-known plaintext setting where plaintext blocks of Data are known.
pKPA 2: Partially-known plaintext setting where only one byte of each block of Data is known.

Table 1 shows the summary of our attacks. Time and Memory are adversary's local computations and memory cost, respectively. Data is the number of required known plaintext/ciphertext blocks at target sectors in pKPA 1, and in pKPA 2, only one byte in a corresponding plaintext block is known. In the ciphertext-only setting, Data is the number of required ciphertexts at target sectors and there is no any information of plaintexts. For example, in pKPA 2 for $n = 64$, given 2^{10} known plaintext blocks in which only 1 byte is known at a target sector, all other plaintext blocks and unknown 7 bytes of 2^{10} known plaintext blocks in the target sector can be recovered with local 2^{57} computations and 2^{10} memory. In the ciphertext-only setting for $n = 60$, given 2^{60} ciphertexts, we can correctly guess one of them with time complexity of 2^{60} and 2^{60} memory, while it ideally should require 2^{64} ciphertexts to successfully guess a 64-bit plaintext.

We stress that our plaintext recovery attacks at target sectors are feasible with less than $O(2^{n/2})$ data unlike a trivial application of the collision attack to the target sectors which requires $O(2^{n/2})$ data. Note that collecting known plaintext blocks in the target sectors is the most difficult task for real world applications. In this sense, our attack is more practical. Especially, for $n = 64$, our plaintext recovery attacks at target sectors are successful with a small number of known plaintext blocks (e.g. 2^{8}) and practical local computations, independently from the key size. Therefore, our results reveal that 64-bit block ciphers with XTS mode, which is commercially deployed in some products, are practically insecure as storage encryption schemes.

2 Preliminaries

2.1 Basic notations

Let $\mathbb{N}$ be the set of positive integers. Let $[n]$ denote $\{1,\ldots,n\}$. The bit length of binary sequence X is written as $|X|$, and $|X|_n$ denotes $\lceil |X|/n \rceil$. Let $(M_1,\ldots,M_m) \xleftarrow{n} M$ denote the n-bit block parsing of M, where $m = |M|_n$, $|M_i| = n$ for $1 \le i \le m-1$ and $|M_m| \in [n]$. For a binary string X, its first s bits and last s bits are written as $\mathsf{msb}_s(X)$ and $\mathsf{lsb}_s(X)$.

Galois Field. Any n-bit value may be taken as an element of $\mathrm{GF}(2^n)$ by seeing it as the sequence of coefficients of the polynomial. In particular, for $X, Y \in \{0,1\}^n$, we write $X \otimes Y$ to denote $\mathrm{GF}(2^n)$-multiplication of X and Y. A division of X by $Y \neq 0$ is written as X/Y.

2.2 Tweakable block cipher

A tweakable block cipher (TBC) is an extension of ordinary block cipher proposed by Liskov et. al [14]. A TBC is a keyed function $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ such that for each $(K,T) \in \mathcal{K} \times \mathcal{T}$, $\widetilde{E}(K,T,\cdot)$ is a permutation over $\mathcal{M}$. Here, K is the key and T is a public value called tweak. A conventional block cipher is a TBC with $\mathcal{T}$ being a singleton, and specifically written as $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$. The encryption of $X \in \mathcal{M}$ under key $K \in \mathcal{K}$ and tweak $T \in \mathcal{T}$ is $\widetilde{E}(K,T,X)$ and is also written as $\widetilde{E}_K(T,X)$ or $\widetilde{E}_K^T(X)$. For block cipher we write as $E_K(X)$. The decryption is written as $\widetilde{E}_K^{-1,T}(Y)$ for TBCs and $E_K^{-1}(Y)$ for block ciphers. For any $T \in \mathcal{T}$ and $K \in \mathcal{K}$, when $Y = \widetilde{E}_K^T(X)$ we have $\widetilde{E}_K^{-1,T}(Y) = X$.

We say $f : \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ is a tweakable permutation if $f(T,*)$ is a permutation for any $T \in \mathcal{T}$. Let $\widetilde{\mathsf{P}} : \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ be the ideal tweakable random permutation distributed uniformly over the set of all tweakable permutations : $\mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$. The security of a TBC $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ is measured by the computational indistinguishability from $\widetilde{\mathsf{P}}$ using chosen encryption queries (T,X) and chosen decryption queries (T,Y) [14, 19].

3 Specification of XTS

XTS is a tweakable encryption over message space $\mathcal{M} = \{0,1\}^*$ with tweak space $\mathcal{T}_{\mathsf{XTS}} = \{0,1\}^n$.

Let $E : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ be an n-bit block cipher, and let $K = (K_1, K_2) \in \mathcal{K}^2$ be a pair of keys of E. The core component of XTS is a single-block TBC written as $\mathsf{XEX2} : \mathcal{K}^2 \times \{0,1\}^n \times \mathcal{T}_{\mathsf{XEX2}} \rightarrow \{0,1\}^n$, where $\mathcal{T}_{\mathsf{XEX2}} = \mathcal{T}_{\mathsf{XTS}} \times \mathbb{N}$ is a tweak space. It encrypts n-bit plaintext block X to create ciphertext block Y using tweak $\overline{T} = (T,j) \in \mathcal{T}_{\mathsf{XEX2}}$ as

$$Y = S \oplus E_{K_1}(M \oplus S), \quad (1)$$

where $S = E_{K_2}(T) \otimes \alpha^j$. Here, α denotes the generator of the field, i.e. the polynomial $\mathtt{x}$, and α^j denotes the multiplication by α for j times.

XTS encrypts plaintext $M \in \{0,1\}^*$ with a tweak $T \in \mathcal{T}_{\mathsf{XTS}}$ by first (1) parsing M as $(M_1, \ldots, M_m) \overset{n}{\leftarrow} M$ and (2) encrypting M_j for $j \in [m-1]$ by XEX2 taking tweak (T, j). The encryption of the last block M_m depends on whether $|M_m| = n$ or shorter, in a similar manner to Ciphertext Stealing in CBC mode. See Figure 1.

3.1 LRW mode

Before XTS, IEEE SISWG considered a mode called LRW [9] named after the paper by Liskov et al. [14]. It can be seen as a predecessor of XEX. An encryption of LRW is the same as Eq. (1), however

$$S = L \otimes \overline{T}$$

is used instead, where $L \in \{0,1\}^n$ is the second key independent of K, and $\overline{T} \in \{0,1\}^n$ identifies the target block in the storage, which is typically considered as a combination of sector number and block index. LRW may look much slower than XTS as it involves a full multiplication over $\mathrm{GF}(2^n)$ for every update of tweak. However this is not the case if sector number and block index are properly encoded, e.g. when $\overline{T}$ is $(T \,\|\, j)$ with sector number $T \in \{0,1\}^{n/2}$ and block index $j \in \{0,1\}^{n/2}$, $L \otimes (T \,\|\, j+1)$ is obtained by $L \otimes (T \,\|\, j) \otimes \alpha$, hence basically the same cost as XTS. LRW mode was consequently not adapted as a standard, however, some encryption software still use it, in particular with 64-bit block ciphers (see Section 8).

4 Attack Model

Motivation. We first need to clarify the adversary model, i.e. how the adversary accesses to a storage encrypted by XTS and what is the goal of the adversary, in a way that reflects practical use cases, at least to some extent. This must be done first, as it is known that, unlike encryption or authenticated encryption, there is no widely accepted security notion that captures XTS beyond single-block tweakable block cipher, i.e., XEX2 (see e.g. Rogaway [18]). It is rather straightforward to derive a birthday attack to break Tweakable Strong Pseudo-random Permutation (TSPRP) notion of XEX2, however this is not sufficient for our purpose.

Informally, we classify the sectors of an XTS-encrypted storage into two categories, called reference sectors and target sectors. At the reference sectors, the adversary can encrypt a known plaintext and decrypt any ciphertext, that is, a combination of known-plaintext attack (KPA) and chosen-ciphertext attack (CCA)[4]. This implies that the plaintext recovery is trivial at these sectors if we

[4] Here it means an attack with decryption queries of any chosen ciphertext and does not mean a combination with chosen-plaintext attack.

```
Algorithm $\mathsf{XTS}_K.\mathrm{Enc}(M, T)$

 1. $(M_1, \ldots, M_m) \overset{n}{\leftarrow} M$
 2. for $i = 1$ to $m - 1$ do
 3.     $C_i \leftarrow \mathsf{XEX2}_K.\mathrm{Enc}(M_i, (T, i))$
 4. if $|M_m| = n$ then
 5.     $C_m \leftarrow \mathsf{XEX2}_K.\mathrm{Enc}(M_m, (T, m))$
 6. else
 7.     $C_m \leftarrow \mathsf{msb}_{|M_m|}(C_{m-1})$
 8.     $D \leftarrow \mathsf{lsb}_{n-|M_m|}(C_{m-1})$
 9.     $\widetilde{M}_m \leftarrow M_m \,\|\, D$
10.     $C_{m-1} \leftarrow \mathsf{XEX2}_K.\mathrm{Enc}(\widetilde{M}_m, (T, m))$
11. $C \leftarrow (C_1 \,\|\, C_2 \,\|\, \ldots \,\|\, C_m)$
12. return $C$
```

```
Algorithm $\mathsf{XTS}_K.\mathrm{Dec}(C, T)$

 1. $(C_1, \ldots, C_m) \overset{n}{\leftarrow} C$
 2. for $i = 1$ to $m - 1$ do
 3.     $M_i \leftarrow \mathsf{XEX2}_K.\mathrm{Dec}(C_i, (T, i))$
 4. if $|C_m| = n$ then
 5.     $M_m \leftarrow \mathsf{XEX2}_K.\mathrm{Dec}(C_m, (T, m))$
 6. else
 7.     $M_m \leftarrow \mathsf{msb}_{|C_m|}(M_{m-1})$
 8.     $D \leftarrow \mathsf{lsb}_{n-|M_m|}(M_{m-1})$
 9.     $\widetilde{C}_m \leftarrow C_m \,\|\, D$
10.     $M_{m-1} \leftarrow \mathsf{XEX2}_K.\mathrm{Dec}(\widetilde{C}_m, (T, m))$
11. $M \leftarrow (M_1 \,\|\, M_2 \,\|\, \ldots \,\|\, M_m)$
12. return $M$
```

```
Algorithm $\mathsf{XEX2}_K.\mathrm{Enc}(X, (T, j))$

 1. $(K_1, K_2) \overset{n}{\leftarrow} K$
 2. $S \leftarrow E_{K_2}(T) \otimes \alpha^j$
 3. $Y \leftarrow S \oplus E_{K_1}(X \oplus S)$
 4. return $Y$
```

```
Algorithm $\mathsf{XEX2}_K.\mathrm{Dec}(Y, (T, j))$

 1. $(K_1, K_2) \overset{n}{\leftarrow} K$
 2. $S \leftarrow E_{K_2}(T) \otimes \alpha^j$
 3. $X \leftarrow S \oplus E^{-1}_{K_1}(Y \oplus S)$
 4. return $X$
```

Fig. 1. XTS.

have ciphertext at a reference sector for unknown plaintext. However, at the target sectors, the adversary can only perform a ciphertext-only attack (COA) or a partially-known-plaintext attack (pKPA, the definition of "partial" depends on the attack) against the target sectors, and cannot perform decryption at all. The goal is to recover the plaintext at one of the target sectors for the corresponding ciphertext obtained by encryption queries to that sector.

The assumption of reference sector(s) follows the existing attack models for storage encryption, such as Ferguson [6, Section 2.7] and Khati et. al [12]. Intuitively, in many cases, there are encrypted sectors where we already know (some part of) the plaintext and can modify the ciphertext and somehow see the resulting plaintext. For example, Windows OS has a boot screen and the default value is known, and the result of a modification of the encrypted sectors that contains this boot screen image is visible by just booting OS and see how the boot screen has been corrupted. There should be other types of sectors that work as reference sectors, depending on applications and OSs. Individual analysis of each case is beyond the scope of our paper.

For the target sectors it is reasonable to assume that we have few knowledge of the plaintext.

Simplified Model. Let $T^{(r)}$ be the single reference sector and $T^{(t)}$ be the target sector. For simplicity, we assume the sector size is always mn bits for some positive integer m. We note that the size of sector and whether the last block is partial or not is irrelevant to our attacks. To simplify the description of attacks while capturing the core of our ideas, we assume the adversary only queries to

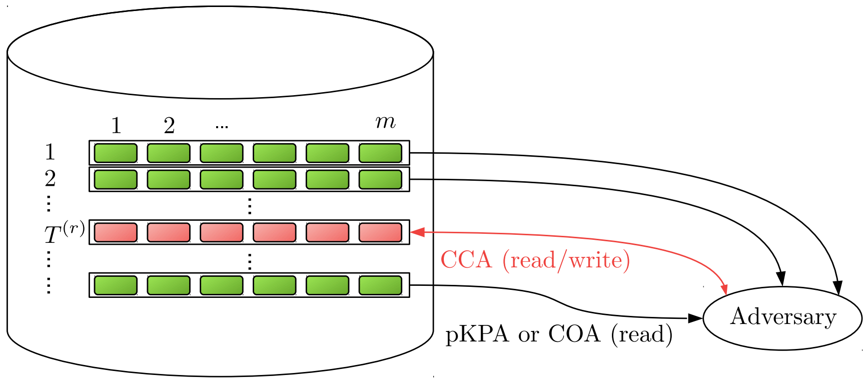


Fig. 2. Our Attack Model.

these two sectors, and each query is given directly to $\mathsf{XEX2}$ rather than XTS. Thus an encryption query is $(M, \overline{T})$ and a decryption query is $(C, \overline{T})$, both are elements of $\{0,1\}^n \times \mathcal{T}_{\mathsf{XEX2}}$.

As we described, the adversary can issue encryption queries $(M, (T^{(r)}, j))$ with some known (possibly random) M for any $j \in [m]$, and decryption queries $(C, (T^{(r)}, j))$ with any $C \in \{0,1\}^n$ and any $j \in [m]$. In fact, the condition for the adversary can be further reduced, so that we only require existence of two different blocks, $j, j' \in [m]$ in the reference sector that accept above queries.

Encryption queries to target sectors $(M, (T^{(t)}, j))$ can be done with unknown M (COA) or partially-known M (pKPA). When the adversary correctly guess M it means the win. Depending on the available queries at $T^{(t)}$, we derive different attacks.

All attacks are extended to the case where there are multiple target sectors. This captures the case that all sectors of HDD are target except the reference sector. The overview of our attack model is described in Fig. 2.

5 Overview of Our Attacks

Our attacks consist of two phases. In the first phase, we perform a *collision attack* at a reference sector $T^{(r)}$ to obtain a mask key $E_{K_2}(T^{(r)})$. This attack is a beyond-birthday attack in that it requires $2^n/2$ encryptions. Once $E_{K_2}(T^{(r)})$ is known, input/output pairs of the internal function E_{K_1} are available in the reference sector $T^{(r)}$. Note that the same internal function E_{K_1} is used in other sectors including the target sector $T^{(t)}$.

In the second phase, we mount *plaintext recovery attacks* at a target sector $T^{(t)}$. Due to a direct access to the internal function E_{K_1} via the reference sector $T^{(r)}$, XTS_K is regarded as a single-key Even-Mansour construction in the target sector $T^{(t)}$. Then, we are able to perform plaintext recovery attacks on the target sector $T^{(t)}$ in several practical attack settings.

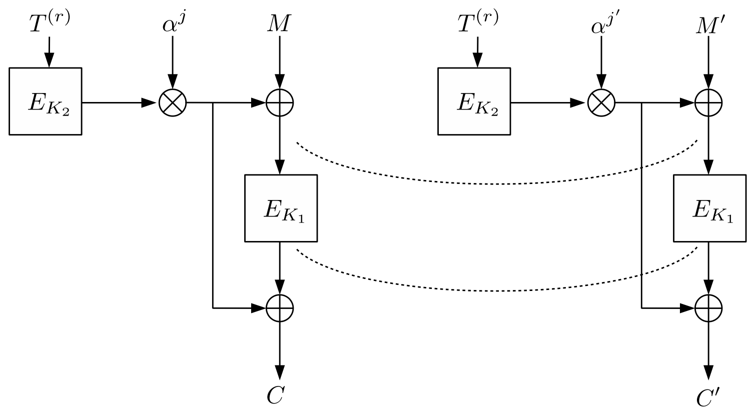


Fig. 3. Collision attack at the reference sector $T^{(r)}$.

Importantly, our plaintext recovery attack works independently from the key size of the internal function E_{K_1}. In other words, the key size of E_{K_1} does not affect the attack complexity.

6 Collision Attack at Reference Sector

We first describe a collision attack that obtains a mask key $E_{K_2}(T^{(r)})$ at a reference sector $T^{(r)}$.

Assumption and Goal. As mentioned in Section 4, the adversary can issue encryption queries $(M, (T^{(r)}, j))$ with some known M, and decryption queries $(C, (T^{(r)}, j))$ with any $C \in \{0,1\}^n$ for two distinct block j and j' in the reference sector $T^{(r)}$.

The purpose of this attack is to recover $E_{K_2}(T^{(r)})$ without knowing K_1 and K_2.

Idea for Collision Attack. We utilize an event where two inputs of E_{K_1} at two distinct blocks j and j' in the reference sector $T^{(r)}$ are the same as illustrated in Figure 3. When this event happens, we get the following equation with respect to inputs of E_{K_1}.

$$M \oplus (E_{K_2}(T^{(r)}) \otimes \alpha^j) = M' \oplus (E_{K_2}(T^{(r)}) \otimes \alpha^{j'}) \tag{2}$$

Since E_{K_1} is a permutation, we also have the following equation with respect to outputs of E_{K_1}.

$$C \oplus (E_{K_2}(T^{(r)}) \otimes \alpha^j) = C' \oplus (E_{K_2}(T^{(r)}) \otimes \alpha^{j'}) \tag{3}$$

From Eqs. (2) and (3), we obtain the equation regarding plaintexts and ciphertexts.

$$M \oplus C = M' \oplus C'. \tag{4}$$

Given $2^{n/2}$ sets of (M, C) and (M', C'), respectively, there exists two pairs of (M, C) and (M', C') that satisfy Eq. (4) with a high probability, in which, Eqs. (2) and (3) hold with probability of $1 - 2^{-n}$. Once such two pairs of (M, C) and (M', C') are found, we can recover a mask key $E_{K_2}(T^{(r)})$ from ciphertexts without knowing K_1 and K_2 as

$$E_{K_2}(T^{(r)}) = (C \oplus C')/(\alpha^j + \alpha^{j'}).$$

To find the collision pairs such that $M \oplus C = M' \oplus C'$, in a straight forward way, we should evaluate $2^n (= 2^{n/2} \times 2^{n/2})$ combinations of each $2^{n/2}$ sets of (M, C) and (M', C'). Using the meet-in-the-middle technique, we can efficiently find collision pairs. Specifically, we first make a table of $2^{n/2}$ pairs of (M, C) indexed by $M \oplus C$, and then access the table with $2^{n/2}$ values of $(M' \oplus C')$ to find pairs such that $M \oplus C = M' \oplus C'$. Then, the time complexity for finding collision pairs is estimated as about $2^{n/2}$ operations.

Attack Procedure. Based on the above idea of the collision attack, the procedure of the mask recovery attack is given as follow.

1. Choose two distinct block indexes, j and j' at a reference sector $T^{(r)}$.
2. Obtain $2^{n/2}$ pairs of (M, C) at j, and store $2^{n/2}$ tuples of $(C, M \oplus C)$ in a table.
3. Obtain $2^{n/2}$ pairs of (M', C') at j', and find a pair of (C, C') such that $M \oplus C = M' \oplus C'$ by accessing the table of $(C, M \oplus C)$ with values of $M' \oplus C'$.
4. Output $(C \oplus C')/(\alpha^j + \alpha^{j'})$ as a candidate for $E_{K_2}(T^{(r)})$.

Complexity Evaluation. $2^{n/2}$ known plaintext/ciphertext pairs are required in Step 2 and 3, respectively, and $2^{n/2}$ memory is required for the table in Step 2. The time complexity to find a collision of $M \oplus C = M' \oplus C'$ in Step 2 and 3 is bounded by $2^{n/2}$ encryption calls, assuming the cost of the sum of a single memory access in Step 2 and Step 3 and computations of $(M \oplus C)$ and $(M' \oplus C')$ is less than a single encryption function call.

In summary, the attack complexity is estimated as follows.

- Time $2^{n/2}$ encryptions
- Data: $2^{n/2+1}$ known plaintexts/ciphertexts
- Memory: $2^{n/2}$ blocks[5]

[5] Using Floyd's cycle-finding algorithm [7], this attack is feasible with the same complexity and negligible memory.

Example 1. For $n = 64$, our collision attack is feasible with 2^{33} known plaintext/ciphertexts, 2^{32} computations, and 2^{32} memory.

Example 2. For $n = 128$, our collision attack is feasible with 2^{65} known plaintext/ciphertexts, 2^{64} computations, and 2^{64} memory.

Especially, for $n = 64$, attack complexity is practical.

Reference Sector as Oracle. After the collision attack against the reference sector $T^{(r)}$ is succeeded, the adversary is able to obtain inputs X and outputs Y of E_{K_1} by knowledge of $E_{K_2}(T^{(r)})$ and a known pair of (M, C) as follows.

$$X = M \oplus (E_{K_2}(T^{(r)}) \otimes \alpha^{j})$$
$$Y = C \oplus (E_{K_2}(T^{(r)}) \otimes \alpha^{j'})$$

In our plaintext recovery attack at the target sectors $T^{(t)}$, we will use this reference sector as encryption/decryption oracle that output a Y/X, given X/Y. Since E_{K_1} is regarded as a public permutation by queries to the reference sector $T^{(r)}$, XTS_K at any target sector $T^{(t)}$ can be treated as a single-key Even-Mansour (SEM) cipher.

7 Plaintext-Recovery Attacks at Target Sector

In this section, we propose plaintext-recovery attacks at a target sector $T^{(t)}$ in several attack settings such as a partially-known plaintext setting and a ciphertext only setting. All attacks are performed after the collision attack in the reference sector $T^{(r)}$ described at Section 6, and the adversary is assumed to has access to the encryption/decryption oracle of E_{K_1}.

7.1 Partially-known Plaintext Attack 1

First, we describe a plaintext-recovery attack at a target sector $T^{(t)}$ in the partially-known-plaintext setting. Informally, we assume only some blocks in $T^{(t)}$ are known.

Assumption and Goal. The adversary is able to collect plaintext/ciphertext block pairs of indexes at some $j \in J^{kp}$ in a target sector $T^{(t)}$, where $J^{kp} \subset [m]$ is an index set of known plaintext blocks in $T^{(t)}$. That is, all the plaintext blocks at $(T^{(t)}, j)$ for any $j \in J^{kp}$ are known. Note that all ciphertexts at the target sector $T^{(t)}$ are available.

The purpose of this attack is to recover a set of unknown plaintext blocks $j \notin J^{kp}$ in the target sector $T^{(t)}$.

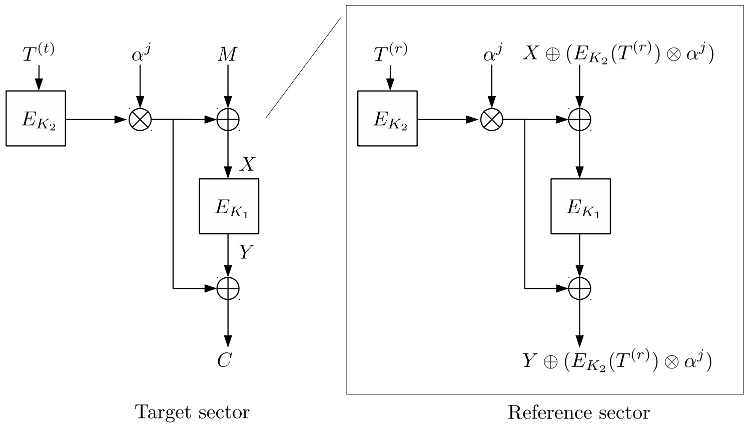


Fig. 4. Partially-known plaintext attack 1.

Attack Idea. The equation with respect to a mask key $E_{K_2}(T^{(t)})$ is given as follows.

$$E_{K_2}(T^{(t)}) = (\alpha^j)^{-1} \otimes (M \oplus X) = (\alpha^j)^{-1} \otimes (C \oplus Y)$$

This equation is rewritten as

$$(\alpha^j) \otimes E_{K_2}(T^{(t)}) = M \oplus X = C \oplus Y. \tag{5}$$

According to Eq. (5), a valid tuple of (M, C, X, Y) must satisfy the equation of $X \oplus Y = M \oplus C$. Given a valid tuple of (M, C, X, Y), $E_{K_2}(T^{(t)})$ is obtained as $E_{K_2}(T^{(t)}) = (\alpha^j)^{-1} \otimes (C \oplus Y)$ in the same manner to the key recovery attack against SEM by Dunkelman et al. [4].

Candidates of M and C are obtained at blocks of $j \in J^{kp}$ in the target sector $T^{(t)}$. As shown in Figure 4, X and Y are obtained in the reference sector $T^{(r)}$ by querying $X \oplus E_{K_2}(T^{(r)}) \otimes (\alpha^j)$ as a plaintext to an encryption oracle, and obtaining the answer (ciphertext) of $Y \oplus E_{K_2}(T^{(r)}) \otimes (\alpha^j)$ where $E_{K_2}(T^{(r)})$ and (α^j) are known values. Let q^t and q^r be the number of available $M \oplus C$ and $X \oplus Y$, respectively. When $q^t \times q^r \geq 2^n$, there exists a valid tuple of (M, C, X, Y) with a high probability.

After $E_{K_2}(T^{(t)})$ is found, with knowledge of $E_{K_2}(T^{(t)})$ and decryption oracle access to E_{K_1} in the reference sector $T^{(r)}$, all unknown plaintext blocks in $J^t \overset{\text{def}}{=} [m] \setminus J^{kp}$ can be recovered.

Attack Procedure. The attack procedure of the plaintext recovery attack in the partially-known plaintext setting is as follows.

1. Obtain q^r pairs of (X, Y) from the reference sector $T^{(r)}$, and store tuples of $(X, Y, X \oplus Y)$ to a table.

2. Obtain q^t known plaintext/ciphertext pairs (M, C) at the target sector $T^{(t)}$ such that $q^t \times q^r \approx 2^n$, and find a tuple of (M, C, X, Y) such that $X \oplus Y = M \oplus C$ by accessing the table of $(X, Y, X \oplus Y)$ with values of $(M \oplus C)$.
3. Output $(\alpha^j)^{-1} \otimes (C \oplus Y)$ as a candidate of $E_{K_2}(T)$.
4. Decrypt ciphertexts of block indexes in J^t by a decryption query to E_{K_1} at the reference sector $T^{(r)}$ and the knowledge of $E_{K_2}(T)$, and then obtain the corresponding plaintexts.

Complexity Evaluation. We estimate the number of queries to the reference sector $T^{(r)}$ as time complexity, as the adversary is able to locally compute input/output pairs of E_{K_1}, (X, Y), at the reference sector $T^{(r)}$ after the collision attack. We evaluate data complexity by the number of queries to the target sectors $T^{(r)}$, as partially-known plaintext/ciphertext blocks are given to the adversary in the target sectors. Thus, our attack is feasible by q^t known plaintext/ciphertext blocks at $T^{(t)}$ in Step 2 and q^r encryptions at $T^{(r)}$ in Step 1. The memory requirement is estimated as q^r blocks in Step 1. Note that in the case of $q^r > q^t$, by changing Step 1 and 2, the memory requirement is reduced while keeping the same time and data complexity, i.e. $(M, C, M \oplus C)$ is stored in a table instead of $(X, Y, X \oplus Y)$. Our attack is successful if $q^t \times q^r \geq 2^n$.

The attack complexity is estimated as follows.

– Time q^r encryptions
– Data: $q^t (= 2^n/q^r)$ known plaintexts/ciphertexts blocks
– Memory: $\min(q^r, q^t)$ blocks

Example 1. For $n = 64$, this plaintext-recovery attack is feasible with 2^8 known plaintext/ciphertext blocks, 2^{56} encryptions, and 2^8 memory.

Example 2. For $n = 128$, this plaintext-recovery attack is feasible with 2^{32} known plaintext/ciphertext blocks, 2^{96} encryptions, and 2^{32} memory.

For $n = 64$, this attack is practically feasible by the standard commercial computer resource in this time. For $n = 128$, time complexity is not very practical but it is still feasible by cloud-based computations and dedicated hardware. Note that our attacks does not need $O(2^{n/2})$ encrypted blocks of the sector for which the target plaintext is stored unlike a trivial application of collision attack in the target sector. It makes our attack more practical in the real world settings.

Multi-target Setting. Since the same key of K_1 is used at any sector, q^r pairs of (X, Y) of E_{K_1}, which are obtained in the reference sector $T^{(r)}$, can be used for plaintext recovery attacks at any sector. Thus, we are able to mount plaintext recovery attacks in multiple sectors at the same time. In the Multi-target setting where we try to compromise at least one of them [10, 8, 16, 2], given $q^r \geq 2^n/q^t$ plaintexts/ciphertexts from multiple target sectors, $E_{K_2}(T^{(t)})$ is recovered at the one of target sectors $T^{(t)}$, and then all plaintexts in this target sector can be recovered. This setting makes our attack more practical because collecting

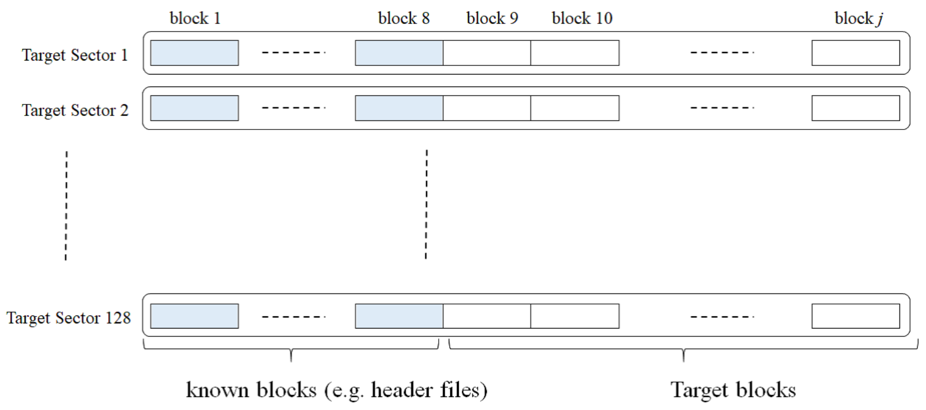


Fig. 5. Partially-known plaintext attack 1 in a multi-target setting.

known plaintexts/ciphertexts are most difficult task in the real world.

Example. For $n = 64$, our plaintext recovery attack is feasible with 2^{10} known plaintext/ciphertexts, and 2^{54} computations with 2^{10} memory. If the adversary knows the first eight blocks of plaintexts in target $2^7(= 128)$ sectors, e.g. fixed header files, as shown in Figure 5, our multi-target attack is successful, that is, unknown plaintext blocks of at least one sector out of 128 target sectors are recovered. This attack is more practical in the real world.

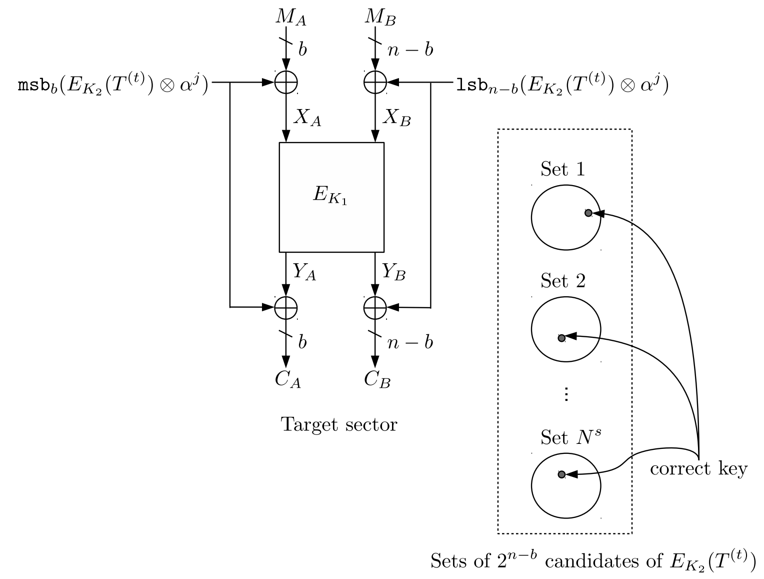
7.2 Partially-known Plaintext Attack 2

In this section, we propose another variant of partially-known plaintext recovery attacks in which there is no block in $T^{(t)}$ that is completely known.

Assumption and Goal. The adversary is able to collect plaintext/ciphertext pairs at blocks of index $j \in J^{kp} \subset [m]$ in a target sector $T^{(t)}$, however only b bits ($b < n$) out of n bits of each plaintext block of index $j \in J^{kp}$ are known and the other $n - b$ bits are unknown. For simplicity, we assume that first b bits of n-bit plaintext block M, denoted by $M_A = \mathsf{msb}_b(M)$, is known, the last $(n - b)$-bit $M_B = \mathsf{lsb}_{n-b}(M)$ is unknown at blocks of index $j \in J^{kp}$, where $M = M_A \parallel M_B$, while our attack is applicable when any part of b bits of each plaintext block is known.

The purpose of this attack is to recover a set of unknown plaintext blocks $j \notin J^{kp}$ and unknown bits of blocks of $j \in J^{kp}$ at the target sector.

Attack Idea. Given q^t pairs of (M, C) and q^r pairs of (X, Y) such that $q^t \cdot q^r = 2^n$, there are $2^{n-b}(= 2^n/2^b)$ tuples of (M_A, C, X, Y) satisfying a corresponding



M_A
M_B
b
$n-b$
$\mathtt{msb}_b(E_{K_2}(T^{(t)}) \otimes \alpha^j)$
$\mathtt{lsb}_{n-b}(E_{K_2}(T^{(t)}) \otimes \alpha^j)$
X_A
X_B
E_{K_1}
Y_A
Y_B
C_A
C_B
Target sector
Set 1
Set 2
Set N^s
correct key
Sets of 2^{n-b} candidates of $E_{K_2}(T^{(t)})$

Fig. 6. Partially-known plaintext attack 2.

b-bit relation of $X_A \oplus Y_A = M_A \oplus C_A$, where X_A, Y_A, and C_A are lower b bits of X, Y and C, respectively. In this case, 2^{n-b} candidates of $E_{K_2}(T^{(t)}) = (\alpha^j)^{-1} \otimes (C \oplus Y)$ are obtained. Among them, there exists a correct $E_{K_2}(T^{(t)})$ with high probability because of $q^t \cdot q^r = 2^n$.

To efficiently sieve candidates of $E_{K_2}(T^{(t)})$, we obtain N^s sets of 2^{n-b} candidates of $E_{K_2}(T^{(t)})$ by preparing N^s sets of q^r pairs of (X, Y), and then find the duplicated one in all different N^s sets, assuming that the correct one is included in each set. Since the probability that a wrong key in a set 1 is included in the other $N^s - 1$ sets is $(2^{n-b}/2^n)^{N^s-1}$, the expected number of surviving wrong pairs is estimated as

$$2^{n-b} \cdot (2^{n-b}/2^n)^{N^s-1} = 2^{n-N^s b}.$$

If $2^{n-N^s b}$ is sufficiently small, we exhaustively test surviving key candidates of $1 + 2^{n-N^s b}$ to find the correct one. The overview of this attack is illustrated in Figure 6.

Attack Procedure. The attack procedure of the plaintext recovery attack in the partial known plaintext setting is as follows.

1. Obtain q^t known plaintext/ciphertext pairs (M, C) at the target sector $T^{(t)}$, and store tuples of $(M_A, C, M_A \oplus C_A)$ to a table.
2. Obtain N^s sets of q^r pairs of (X, Y) at $T^{(r)}$ such that $q^t \times q^r \approx 2^n$, and find N^s sets of 2^{n-b} tuples of (M_A, C, X, Y) satisfying the b-bit relation of

$X_A \oplus Y_A = M_A \oplus C_A$ by accessing the table of $(M_A, C, M_A \oplus C_A)$ with the values of $(X_A \oplus Y_A)$.

3. Output N^s sets of 2^{n-b} candidates of mask keys $E_{K_2}(T) = (\alpha^j)^{-1} \otimes (C \oplus Y))$ from valid tuples of (M_A, C, X, Y), and find the duplicated ones in all N^s sets. The expected number of remaining keys are estimated as $(1 + 2^{n-N^s b})$.
4. Exhaustively search $(1 + 2^{n-N^s b})$ surviving key candidates, and find the correct $E_{K_2}(T)$.
5. Decrypt ciphertexts of index in J^t by a decryption query to E_{K_1} at the reference sector $T^{(r)}$ and the knowledge of $E_{K_2}(T)$, and then obtain the corresponding unknown plaintexts.

Complexity Evaluation. The required data is q^t known plaintexts/ciphertexts in Step 1 where only b bits of each block are known. The time complexity is estimated as the sum of $N^s \times q^r$ encryptions in Step 2 and $1+2^{n-N^s b}$ encryptions in Step 4. The memory requirement is the sum of $N^s \times q^t$ (Step 1) and $N^s \times 2^{n-b}$ (Step 3). The attack complexity is estimated as follows.

- Time: $N^s \times q^r$ (Step 1) $+ 1 + 2^{n-N^s b}$ (Step 4)
- Data: $q^t (= 2^n / q^r)$ partial known plaintexts/ciphertexts in which only b bits of each plaintext are known.
- Memory: $N^s \times q^t$ (Step 1) $+ N^s \times 2^{n-b}$ (Step 3)

Note that the value N^s such that $N^s \times q^r = 1 + 2^{n-N^s b}$ is optimal for time complexity.

Table. 2 shows time and data complexity in each b with optimal values of Ns. Surprisingly, even when only b bits in q^t plaintexts are known, the key recovery attack is feasible with almost the same data and time complexity, that is, the product of time and data is around 2^n.

Example 1. For $n = 64$, $b = 8$ (1 byte) and $q^r = 2^{54}$, when $N^s = 2$ the expected number of surviving wrong pairs is $2^{48} (= 2^{64-8\cdot 2})$. The plaintext-recovery attack is feasible with 2^{10} known plaintext/ciphertexts, $2^{55} (= 2 \times 2^{54} + 1 + 2^{48})$ computations, and $2^{57} (= 2 \times 2^{10} + 2 \times 2^{56})$ memory.

Example 2. For $n = 128$, $b = 8$ (1 byte) and $q^r = 2^{96}$, when $N^s = 4$ the expected number of surviving wrong pairs is $2^{96} (= 2^{128-8\cdot 4})$. The plaintext-recovery attack is feasible with 2^{32} known plaintext/ciphertexts, $2^{98} (= 4 \times 2^{96} + 1 + 2^{96})$ computations, and $2^{122} (= 4 \times 2^{32} + 4 \times 2^{120})$ memory.

Multi-target Attack. This attack is naturally extended to a multi-target attack. It makes collecting (M, C) easier as discussed in the previous section. Figure 7 illustrates the case of only one byte is known ($b = 8$) in 1024 target sectors for $n = 64$. In this case, when $N^s = 2$, the expected number of surviving wrong pairs is $2^{48} (= 2^{64-8\cdot 2})$. The plaintext recovery attack in Multi-target setting is feasible with $2^{58} (= 2 \times 2^{57} + 1 + 2^{48})$ computations, and $2^{57} (= 2^7 + 2 \times 2^{57} + 1 + 2^{48})$ memory. The adversary recovers unknown plaintext blocks in one of 1024 target

Table 2. Time and data complexity in each b (the number of known bits in n-bit block) with optimal values of Ns of partially-known plaintext attack 2.

	$b=2$			$b=4$			$b=8$			$b=16$			$b=24$		
N^s	5	10	15	3	5	8	2	3	4	1	2	3	1	1	2
Data	2^{10}	2^{20}	2^{30}	2^{10}	2^{20}	2^{30}	2^{10}	2^{20}	2^{30}	2^{10}	2^{20}	2^{30}	2^{10}	2^{20}	2^{30}
Time	$2^{56.3}$	$2^{47.3}$	$2^{37.9}$	$2^{55.6}$	$2^{46.3}$	2^{37}	2^{55}	$2^{45.6}$	2^{36}	2^{54}	2^{45}	$2^{35.6}$	2^{54}	2^{44}	2^{35}
Data $\times$ Time	$2^{66.3}$	$2^{67.3}$	$2^{67.9}$	$2^{65.6}$	$2^{66.3}$	2^{67}	2^{65}	$2^{65.6}$	2^{66}	2^{64}	2^{65}	$2^{65.6}$	2^{64}	2^{64}	2^{65}

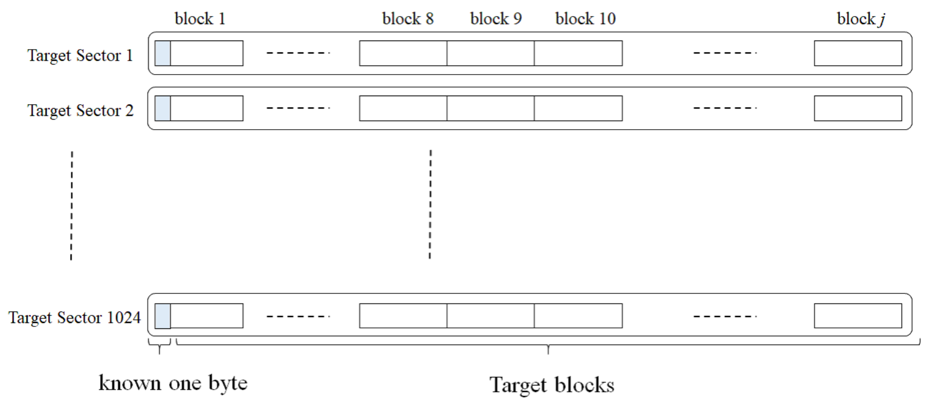


Fig. 7. Partial-known plaintext attack 2 in a Multi-target setting.

sectors.

Low-Memory Attack. When $N_s = 1$, we can mount low-memory attacks at the cost of increased time complexity. In this case, 2^{n-b} candidates in Step 3 are exhaustively searched without the duplication check. Thus, we do not need to store 2^{n-b} key candidates and the memory requirement of Step 3 to store key candidates are not necessary. The attack complexity is estimated as follows.

- Time: q^r (Step 1) $+ 1 + 2^{n-b}$
- Data: $q^t (= 2^n/q^r)$ partial known plaintexts/ciphertexts in which only b bits of each plaintext are known.
- Memory: q^t

When $q^r > 1 + 2^{n-b}$, the attack is feasible with the almost same complexity and data requirements, namely the product of time and data is around 2^n, otherwise it becomes less efficient.

Example 1. For $n = 64$, $b = 8$ (1 byte) and $q^r = 2^{54}$, the expected number of surviving wrong pairs is $2^{56} (= 2^{64-8})$ to be exhaustively searched. The plaintext recovery attack is feasible with 2^{10} known plaintext/ciphertexts, $2^{57} (= 2^{54}+1+2^{56})$ computations, and 2^{10} memory.

Example 2. For $n = 128$, $b = 8$ (1 byte) and $q^r = 2^{96}$, the expected number of surviving wrong pairs is $2^{120} (= 2^{128-8})$ to be exhaustively searched. The plaintext recovery attack is feasible with 2^{32} known plaintext/ciphertexts, $2^{120} (= 2^{96} + 1 + 2^{120})$ computations, and 2^{32} memory.

7.3 Ciphertext-only Attack

In this section, we propose a plaintext recovery attack in the weakest, ciphertext-only setting, where the adversary does not have any information about plaintexts.

Assumption and Goal. The adversary is able to collect ciphertexts at a target sector $T^{(t)}$, and does not have any information about plaintexts.

The purpose of the attack is to guess plaintext at the target sector $T^{(t)}$ with higher probability than random guessing.

Attack Idea. From Eq. 5, an n-bit equation with respect to a plaintext is given as

$$M = C \oplus (X \oplus Y (= E_{K_1}(X))).$$

Interestingly, a plaintext M is expressed as C, X and Y without including the value of E_{K_2} as shown in Figure 8. These are obtained in the target sector $T^{(t)}$ and the reference sector $T^{(r)}$, respectively, even in the ciphertext-only setting,

We utilize a *multi collision technique* in $(X \oplus Y (= E_{K_1}(X)))$ to guess a plaintext M from only a corresponding ciphertext C with higher probability than random guessing, while this technique was originally used for key-recovery attacks on a 2-round single-key Even-Mansour cipher [17, 3],

Specifically, we exploit t-way collisions of $F(X) = (X \oplus E_{K_1}(X))$ for a value of v such that

$$X^{(1)} \oplus E_{K_1}(X^{(1)}) = X^{(2)} \oplus E_{K_1}(X^{(2)}) = \cdots = X^{(t)} \oplus E_{K_1}(X^{(t)}) = v,$$

i.e., t input values of $X^{(1)}, \ldots, X^{(t)}$ map to the same output value v through the function F. Assuming that there are t-way collisions with respect to v in $F(X)$, the probability of $M = C \oplus v$ is estimated as $t/2^n$, which is t times higher than the expected $1/2^n$.

Attack Procedure. Our plaintext recovery attack in the ciphertext-only setting consists of two phases. In the first phase, we find t-way collisions of $F(X) = (X \oplus E_{K_1}(X)) = v$ by accessing the reference sector $T^{(r)}$. In the second phase, we guess a plaintext as $M = P \oplus v$. The attack procedure using t-way collisions is given as follows.

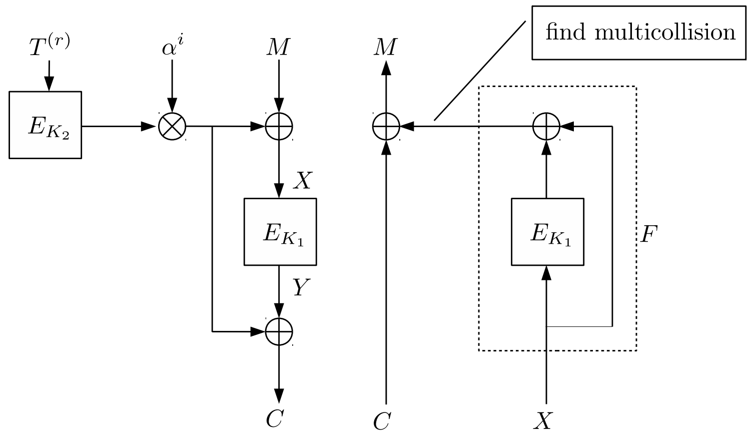


Fig. 8. Ciphertext-only Attack.

1. Obtain q^r pairs of (X, Y) at the reference sector $T^{(r)}$ and evaluate $(X \oplus Y)$.
2. Find t-way collisions of $F(X) = (X \oplus Y(= E_{K_1}(X)))$, which map to the same value v.
3. Guess a values of a target plaintext M at $T^{(t)}$ from a corresponding ciphertext C as $M = C \oplus v$.

Complexity Evaluation. For the accurate estimation of the cost for finding t-way collisions of $F(X)$ in the Step 2, we assume that in-degree of an element in the range of $F(X)$ is distributed according to the Poisson distribution with an expectation λ, which is equal to the average in-degree, i.e., $\lambda = q^r/2^n$ is the ratio between the sizes of the domain and range as with Dinur et al. [3]. For a parameter t, the probability that an arbitrary element v will have an in-degree of t is $(\lambda^t \cdot e^{-\lambda})/t!$. Since there are 2^n elements in the range, it is expected that about $(2^n \cdot \lambda^t \cdot e^{-\lambda})/t!$ vertexes have an in-degree of t.

Assuming that a value of v in has t-way collisions, the success probability of our plaintext recovery attack is estimated as $t/2^n$. The time complexity is estimated as q^r for finding t-way collisions at the reference sector $T^{(r)}$.

Example 1. For $n = 64$ and $q^r = 2^{60}$, which implies $\lambda = 2^{60}/2^{64} = 2^{-4}$, and $t = 10$, the number of 10-way collisions is estimated as $num = (2^n \cdot \lambda^t \cdot e^{-\lambda})/t! = (2^{64} \cdot (2^{-4})^{10} \cdot e^{-2^{-4}}/10!) = 4$. With these parameters, the time complexity is estimated as 2^{60} encryptions and its memory complexity is 2^{60}. Then, the success probability in Step 3 is $10/2^{64} = 2^{-60.68}$ while ideally it should be 2^{-64}.

Example 2. For $n = 128$ and $q^r = 2^{124}$, which implies $\lambda = 2^{124}/2^{128} = 2^{-4}$, and $t = 25$, the number of 18-way collisions is estimated as $num = (2^n \cdot \lambda^t \cdot e^{-\lambda})/t! = (2^{128} \cdot (2^{-4})^{18} \cdot e^{-2^{-4}}/18!) = 4$. With these parameters, time complexity

Table 3. The expected number of vertexes for each q^r computations and the success probability of our ciphertext-only plaintext recovery attacks using multiple t-way collision for $n = 64$.

q^r	Vertex	Expected Number of Vertexes	Success Probability	Ideal Probability
2^{62}	12	1787	$2^{-49.6}$	$2^{-53.2}$
	10	$2^{21.8}$	$2^{-38.9}$	$2^{-42.2}$
2^{60}	10	4	2^{-60}	2^{-62}
	8	$2^{16.6}$	$2^{-44.4}$	$2^{-47.4}$
	6	$2^{30.4}$	$2^{-31.0}$	$2^{-33.6}$
	4	$2^{43.3}$	$2^{-18.7}$	$2^{-20.7}$
2^{56}	6	90	$2^{-54.9}$	$2^{-57.5}$
	4	$2^{27.4}$	$2^{-32.0}$	$2^{-34.0}$
2^{52}	4	2739	$2^{-51.6}$	$2^{-53.6}$

is estimated as 2^{124} encryptions and its memory complexity is 2^{124}. Then, the success probability in Step 3 is $18/2^{128} = 2^{-123.83}$ while ideally it should be 2^{-128}.

Multiple *t*-way Collision. For $n = 64$ and $q^r = 2^{60}$, we are able to find $2^{16.6}$ 8-way collisions. Given a ciphertext, the probability that the corresponding plaintext is included in a set of $P = C \oplus v$, $v \in V^s$, where V^s is a set of $2^{16.6}$ 8 vertexes, is estimated as $2^{44.4} (= 2^{16.6} \times 8/2^{64})$, while ideally it should be $2^{47.4} (= 2^{16.6}/2^{64})$. Table 3 shows the expected number of vertexes for each q^r, the success probability of our plaintext recovery attack which is estimated as the probability that a plaintext is included in a set of $P = C \oplus v'$, where v' is a set of corresponding vertexes.

Multi-target Setting. Once t-way collisions are found in Step 2, plantext recovery attacks are applicable to any ciphertext at any target sector $T^{(t)}$ with the same success probability (namely multi-target attack). Thus, in the above case, there are $2^{44.4}$ ciphertexts in the while target disc, the attack is successful one of them with a high probability.

8 Practical Impact

Since our attacks are based on collisions, the data complexity is at least $2^{n/2}$ for one key of XTS, which is not considered as an urgent risk when $n = 128$ including the case of XTS-AES. However, we stress that the storage encryption is crucially different from encryption of communication that allows rekeying for each session to thwart the attacks that need a large amount of data per key. A key in a storage encryption is hard to renew, since this implies total re-encryption of storage devices. Hence it is likely to be used very long time (or forever), and the risk of birthday attack is larger than the encryption for communications. In case of 64-bit block ciphers however, as demonstrated by Sweet32 [1], a collision can occur around 2^{32} 8-byte blocks which is about 32 Gbyte. In this case our attack has a practical complexity.

We performed a survey on existing encryption software that employ 64-bit block ciphers with LRW or XTS. The survey is on specification documents and the product website, and we did not look into the source codes even if available.

Fortunately, most of popular products with active developments, such as VeraCrypt [6], solely use XTS-AES or XTS with strong 128-bit block ciphers (e.g. AES finalists) in addition to AES. Nevertheless, we can find some examples of storage/file encryption software that use 64-bit block ciphers with LRW. We note that our attacks against XTS are also applicable to LRW with minor modifications, since our attacks are independent of the mechanism for deriving each masks

Examples.

- BestCrypt [7] is a popular encryption software and it employs 64-bit block ciphers such as Blowfish, CAST, and GOST 28147-89 with LRW. We have informed our findings to the developper of BestCrypt.
- Old version of TrueCrypt [8], specifically the version up to 4.2, supported Blowfish, CAST-128, and TDES with LRW.
- A successor of TrueCrypt called CipherShed [9] used Blowfish, Cast, and DES with LRW.
- Some popular encryption systems on linux such as dm-crypt [10] may support 64-bit ciphers and LRW. The supported ciphers will depend on the kernel.

9 Conclusion

In this paper, we have studied the security of XTS storage encryption scheme from the aspects of plaintext recovery. The provable security result of the core of XTS (XEX2) suggests that the attack is not feasible without data of $2^{n/2}$, and indeed it is rather easy to derive an attack based on collision, using that amount of data. However this attack only breaks the indistinguishability. Starting from this simple collision attack, we have shown several plaintext recovery attacks beyond collision, based on the popular adversary model against storage encryption. Our main observation is that, once a collision attack was successful, XTS

[6] `https://www.veracrypt.fr/en/Home.html`
[7] `https://www.jetico.com/data-encryption`
[8] `http://truecrypt.sourceforge.net/`
[9] `https://www.ciphershed.org/`
[10] `https://wiki.archlinux.org/index.php/dm-crypt`

can be seen as a variant of Single-key Even-Mansour (SEM) cipher, therefore we can adapt the known attacks against SEM. To our knowledge, our work is the first to study the plaintext recovery security of XTS. Since all attacks have $2^{n/2}$ to even close to 2^n complexity, the attacks do not show a practical threat against the standard XTS–AES for its $n = 128$-bit block. However, as we observed, there still exist systems that use 64-bit block ciphers with XTS or LRW for storage encryption, where our attacks can be a practical concern.

Acknowledgments The authors would like to thank the anonymous referees of SAC 2019 for their insightful comments and suggestions. Takanori Isobe is supported by Grant-in-Aid for Scientific Research (B) (KAKENHI 19H02141) for Japan Society for the Promotion of Science.

References

[1] Karthikeyan Bhargavan and Gaëtan Leurent. On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 456–467. ACM, 2016.

[2] Eli Biham. How to Decrypt or Even Substitute DES-Encrypted Messages in 2^{28} steps. *Inf. Process. Lett.*, 84(3):117–124, 2002.

[3] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Key Recovery Attacks on Iterated Even-Mansour Encryption Schemes. *J. Cryptology*, 29(4):697–728, 2016.

[4] Orr Dunkelman, Nathan Keller, and Adi Shamir. Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 336–354. Springer, 2012.

[5] Morris Dworkin. Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices. Standard, National Institute of Standards and Technology., 2010.

[6] Niels Ferguson. AES-CBC + Elephant diffuser – A Disk Encryption Algorithm for Windows Vista –. 2006.

[7] Robert W. Floyd. Nondeterministic Algorithms. *J. ACM*, 14(4):636–644, October 1967.

[8] Pierre-Alain Fouque, Antoine Joux, and Chrysanthi Mavromati. Multi-user Collisions: Applications to Discrete Logarithm, Even-Mansour and PRINCE. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 420–438. Springer, 2014.

[9] Shai Halevi. Storage Encryption: A Cryptographer's View. *Invited Talk at SCN 2008*, 2008.

[10] Viet Tung Hoang, Stefano Tessaro, and Aishwarya Thiruvengadam. The Multi-user Security of GCM, Revisited: Tight Bounds for Nonce Randomization. In

Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, pages 1429–1440, New York, NY, USA, 2018. ACM.

[11] Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices. Standard, IEEE Security in Storage Working Group.

[12] Louiza Khati, Nicky Mouha, and Damien Vergnaud. Full Disk Encryption: Bridging Theory and Practice. In Helena Handschuh, editor, *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, volume 10159 of *Lecture Notes in Computer Science*, pages 241–257. Springer, 2017.

[13] Gaëtan Leurent and Ferdinand Sibleyras. The Missing Difference Problem, and Its Applications to Counter Mode Encryption. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 745–770. Springer, 2018.

[14] Moses Liskov, Ronald L. Rivest, and David A. Wagner. Tweakable Block Ciphers. In Moti Yung, editor, *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2002.

[15] Kazuhiko Minematsu. Improved Security Analysis of XEX and LRW Modes. In Eli Biham and Amr M. Youssef, editors, *Selected Areas in Cryptography, 13th International Workshop, SAC 2006, Montreal, Canada, August 17-18, 2006 Revised Selected Papers*, volume 4356 of *Lecture Notes in Computer Science*, pages 96–113. Springer, 2006.

[16] Nicky Mouha and Atul Luykx. Multi-key Security: The Even-Mansour Construction Revisited. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 209–223. Springer, 2015.

[17] Ivica Nikolic, Lei Wang, and Shuang Wu. Cryptanalysis of Round-Reduced LED. In Shiho Moriai, editor, *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2013.

[18] Philip Rogaway. Evaluation of Some Blockcipher Modes of Operation. *CRYPTREC Report*, 2011.

[19] Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Pil Joong Lee, editor, *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.