Article

Block Cipher in the Ideal Cipher Model: A Dedicated Permutation Modeled as a Black-Box Public Random Permutation

Yasir Nawaz and Lei Wang

Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; my_nawaz@sjtu.edu.cn (Y.N.); wanglei@cs.sjtu.edu.cn (L.W.)

Received: 3 November 2019; Accepted: 2 December 2019; Published: 5 December 2019

Abstract: Designing a secure construction has always been a fascinating area for the researchers in the field of symmetric key cryptography. This research aimed to make contributions to the design of secure block cipher in the ideal cipher model whose underlying primitive is a family of $n-\text{bit}$ to $n-\text{bit}$ random permutations indexed by secret key. Our target construction of a secure block ciphers denoted as $E[s]$ is built on a simple XOR operation and two block cipher invocations, under the assumptions that the block cipher in use is a pseudorandom permutation. One out of these two block cipher invocations produce a subkey that is derived from the secret key. It has been accepted that at least two block cipher invocations with XOR operations are required to achieve beyond birthday bound security. In this paper, we investigated the $E[s]$ instances with the advanced proof technique and efficient block cipher constructions that bypass the birthday-bound up to $2^{2n}$ provable security was achieved. Our study provided new insights to the block cipher that is beyond birthday bound security.

Keywords: pseudorandom permutation; block cipher; ideal cipher model; beyond birthday bound; provable security

1. Introduction

A block cipher encryption design is called beyond birthday bound (BBB) secure if the proven upper bound on the adversarial advantage is meaningful even if an adversary can process more than $2^{n/2}$ data blocks, where $n$ is the size of the block of a block cipher. The first time, Iwata proposed a BBB encryption mode cipher-based encryption (CENC) [1]. This was nonce based construction providing a solution through the invocation of more than one block cipher and simple XOR operation and achieved $2^{2n/3}$ security against all nonce respecting adversaries. Later on, Iwata proved CENC construction based on mirror theory technique [2], and achieved optimal security [3]. Bhattacharya and Nandi also gave the BBB security of CENC by analyzing the security bound of variable output length using the chi-squared method.

1.1. Pseudorandom Permutation and Pseudorandom Function with BBB

The conventional approach for designing the cryptography primitives based on symmetric cipher is to behave as a perfectly random function. The vast majority, in this case, is an encryption scheme [4], MAC encryption schemes [5,6], and authenticated encryption schemes [7], following this paradigm via pseudorandom functions (PRF). Patarin suggested the construction of permutation sum and proved that a variant of single permutation indistinguishable from a random function up to BBB [8]. In 2003, Patarin gave the result $2^{2n/3}$ security [9], like so, in 2005, achieved up to this security bound [10,11]. However, the PRF provides a solution for increasing the use of cryptography in a real-world application. The pseudorandom permutation (PRP) is the leading building block of the cryptographic design in spite of PRF [12–15]. If a block cipher is directly implemented as a PRF, which will have provable...
security limit birthday bound with a large block, this is often acceptable. But it is not acceptable in practice with a lightweight block cipher, which has relatively small block sizes. The PRF can be replaced by a PRP up to birthday bound queries [16–19]. Moreover, if the block size of a block cipher is large enough, then the security loss is sometimes acceptable. Whatever, there are many scenarios, such as lightweight applications, whose numbers have grown tremendously before some years that require higher security bound [20–26]. In recent years, various constructions have been proposed that achieve BBB security against more than \(2^{n/2} \) queries. We could categorize these constructions into XOR permutations based and truncation based. The XOR permutations is popular for BBB construction by taking the XOR of more than one independent PRP [20].

\[
XOR_{E_{k_1}, E_{k_2}}(x) = E_{k_1}(x) \oplus E_{k_2}(x)
\]

This construction was analyzed by Lucks [21]. The single variant of this construction provides the security up to \(2^{n/3} \) queries [27]. After that, Patarin revised this construction and improved the security bound up to \(2^{n/67} \) [28]. Later on, the results were generated by more than two independent PRP with XOR operation [29]. Dai et al. [30] using the chi-squared method verified the \(n \)-bit security of XOR construction, but the proof was provided by Bhattacharya and Nandi [31]. The XOR construction is acceptable for encryption, but it is not usable for authentication, because domain size is required to extend. This can be solved through hashing the message, but the XOR construction needs some precise combination with a double block hash function [32–34]. The truncation based solution was presented by Hall et al. [17]. Later on, it was proved that truncating \(n \)-bit permutation has security bound up to \(2^{n/3} \) queries [35]. Stam also derived these results in a non-cryptographic context [27]. Recently, another construction was proposed, which is known as Encrypted Davies Meyer (EDM) introduced by Cogliati and Seurin [36].

\[
EDM_{E_{k_1}, E_{k_2}}(x) = E_{k_2}(E_{k_1}(x) \oplus x)
\]

There are two independent permutations and it behaves like random function up to \(q^3 / 2^{2n} \) [36]. Afterward, Dai et al. [30] achieved \(q^4 / 2^{3n} \) using the chi-squared method. Now, a novel construction EDMD improved the security up to \(2^n / 67n \) by using mirror theory technique, which has almost an optimal security [37].

\[
EDMD_{E_{k_1}, E_{k_2}}(x) = E_{k_2}(E_{k_1}(x)) \oplus E_{k_1}(x)
\]

Two independent keys are required for EDMD. The single key setting is significant for higher security bound and efficient construction, which was also performed in our construction. Anyways, this construction secures up to \(q / 2^{2n/3} \). Cogliati and Seurin also extended the EDMD construction called encrypted Wegman carter with davies meyer (EWCDM), which is nonce based BBB secure.

\[
EWCDM_{E_{k_1}, E_{k_2}, H_K}(N, M) = E_{k_2}(E_{k_1}(N) \oplus N \oplus H_K(M))
\]

where, \(H_K\) is a universal hash function, \(N\) denote the nonce, and \(M\) denote the message, which has an arbitrary length. The EWCDM achieved BBB up to \(2^{2n/3} \) MAC queries when it has nonce respecting setting. The use of internal state values of EWCDM construction makes their security analysis formally inapplicable [37]. Mennink presented the rationale relying on the EWCDM function, and simplified versions of the conversion method applied to the advanced encryption scheme (AES) [38]. The main proposal of AES-PRF, the AES with a feed forward of the middle state, achieved almost no optimal security. This construction was applied to GCM and GCM-SIV, and how it entails the significant security improvements was discussed. A little while back, Mennink presented a heuristic study to build BBB secure from public random permutation, showing that a single permutation call could not be secured BBB [39].

The above discussion shows that what to be tackled in PRF for BBB and where the goal is to build PRF, so that it is indistinguishable from a truly random function. However, our study aimed to build
block cipher in the ideal cipher model, under the assumption that the block cipher is a PRP out of
PRF, achieving full security. Moreover, the sum of even mansour (SoEM) construction achieves BBB up
to $2^{2n/3}$, that is built from two randomly drawn keys and two independent permutations; if either keys
or permutations are identical, then there is a birthday bound attack.

1.2. Our Construction

In this paper, we focused on a block cipher design based on a single key, which achieved BBB up
to $2^n$ security. The main motivation is by the scenarios where the block cipher only has block size of
32−bit, 48−bit, and 64−bit [40]. The target construction of block cipher depicted in Figure 1, defined
as $E[s] : K \times P \to P$, consists of two block cipher invocations and additional simple XOR operation.
Furthermore, a heuristic approach is carried out to examine the instances of $E[s]$ and, at last, $E1 – E32$
efficient construction is successfully found. In detail, the first invoke of block cipher produces a subkey
$y$ from the secret key $k$ such that $y = E(k, 0), y = E(0, k)$, and $y = E(k, k)$. The second invoke of a block
cipher encrypt and decrypt the plaintext $p$ and ciphertext $c$, respectively, with a key $k$ or $k \oplus y$. However,
we stress that the first block cipher invocation is precomputing and storing the subkey $y$. Thus, our
design only requires one invocation of a block cipher for encryption and decryption when the subkey
$y$ is precomputed and stored. We have designed this construction in the ideal cipher model that has
the main advantage of provable security up to $2^n$. The previously available block cipher has maximum
provable security up to $2^{2n/3}$. From the efficiency point of view, previous constructions required
more than one key, $s > 2$ block cipher invocations [20,36], and universal hash function invocations; in
the absence of these, their efficiency needed to be increased. The minimum number of block cipher
invocation with a single key is good for efficiency. Our design requires just a single secret key and one
block cipher invocation for encryption and decryption when the subkey is precomputed and stored.

2. Preliminaries

2.1. Notations

The $[0,1]^n$ denotes the set of bit strings of length $n$. We denote the bitwise addition $a \oplus b$, where $a, b \in [0,1]^n$. The $Y \leftarrow Z$ is the assignment of $Z$ to the variable $Y$. The $x \leftarrow X$ denotes
the uniform random selection of $x$ from $X$. The $|X|$ denotes the number of elements in $X$. Let $a \in [0, 1]$ and $b \in [0, 1]$, $a b$ denotes the multiplication of $a$ and $b$, if $a = 1$, then it is equal to $b$, and if $a = 0$,
then $a b$ equals to 0. The block cipher denotes as $E : K \times P \to P$, where $P$ is a plaintext/message space,
$K$ is the key space. Throughout the paper, we have fixed $K = P = [0,1]^n$. Let $E(k, \cdot)$ and $E^{-1}(k, \cdot)$
denote the encryption and decryption, respectively, with a secret key $k \in K$. Let $E^{k}(\cdot)$ involves $E(k, \cdot)$
and $E^{-1}(k, \cdot)$. Sometimes, we denote $E(k, \cdot)$ as $E_k(\cdot), E^{-1}(k, \cdot)$ as $E_k^{-1}(\cdot)$, and $E^{k}(\cdot)$ as $E_k(\cdot)$ and $E_k^{-1}(\cdot)$,
respectively. The $(u, w)$ are the input and output tuple of $E$ such that $w = E(u)$. The input-output tuple
of $E_k$ is denoted as $(p, c)$ such that $E_k(p) = c$. Let $\text{Perm}(n)$ denote the set of all permutations on $[0,1]^n$.
The function $\pi$ is said to be an ideal cipher model if randomly selected that is $\pi \leftarrow \text{Perm}(n)$. Similarly,
we define these notations $\pi(\cdot, \cdot), \pi^{-1}(\cdot, \cdot)$, and $\pi^k(\cdot, \cdot)$, respectively.

2.2. Security Definition

A computationally unbounded distinguisher $D$ is an algorithm that has adaptive access to
an oracle and outputs a bit 0 or 1. Let the two oracles $O_1$ and $O_2$ have the same interface, we can get
the distinguishing advantage of $D$ as follows.

$$Adv(D) = \Pr[D^{O_1} \Rightarrow 1] - \Pr[\Pr[D^{O_2} \Rightarrow 1]$$

A block cipher with a key space $K$ and message space $P$ is a mapping $E : K \times P \to P$ such that for
all key $k \in K$. The $E(K, P)$ is a permutation over $P$. We denote $E_k(p)$ for $E(k, p)$. The distinguisher $D$ is
having query access to \((O_1, E^\oplus)\): \(O_1\) is either \(E^\oplus_k(\cdot, \cdot)\) with \(k \overset{\$}{\leftarrow} K\) or \(\pi \overset{\$}{\leftarrow} \text{Perm}\). The \(E^\oplus\) is an underlying block cipher. The advantage of distinguisher \(D\) in distinguishing \(E\) and \(\pi\) is defined as:

\[
\text{Adv}^{\text{pp}}_{E}(D) = \left| \Pr[D^{E^\oplus_k(\cdot, \cdot)}, E^\oplus(\cdot) \Rightarrow 1] - \Pr[D^{\pi(\cdot), E^\oplus(\cdot) \Rightarrow 1} \right|
\]

Throughout this paper, we considered information as theoretical with computationally unbounded distinguishers \(D\) solely limited by the number of queries to the oracle. Overall, maximum is taken by distinguisher \(D\) that makes at most \(q\) queries to its oracles. Let \(\text{Adv}^{\text{pp}}_{E}(q) = \max_{D} \{ \text{Adv}^{\text{pp}}_{E}(D) \}\)

### 2.3. H-Coefficient Technique

Central to our proof is a H-Coefficient technique presented by Patarin [8,41]. As mentioned above, we considered the information as theoretical, with computationally unbounded distinguisher \(D\). Thus, we always assumed that distinguisher \(D\) is deterministic without the loss of generality. Let distinguisher \(D\) interacts with \(O_1\) and \(O_2\). The interaction of \(D\) with its oracles are recorded in a view \(v\). The \(X_{O_2}\) is the probability distribution of \(v\) when distinguisher \(D\) interacts with \(O_2\). The \(V\) is the set of all attainable views \(v\) when \(D\) interacting with \(O_2\), which is \(V = \{ \forall \left| \Pr[X_{O_2} = \bar{v}] > 0 \right\}\). The H-Coefficient technique states as follows:

Let \(0 \leq \varepsilon \leq 1\). Consider a partition \(V = V_{\text{good}} \cup V_{\text{bad}}\) set of attainable view such that:

1. \(\Pr[X_{O_2} \in V_{\text{bad}}]\)
2. For all \(v \in V_{\text{good}}\), \(\Pr[X_{O_2} = \bar{v}] \geq 1 - \varepsilon\)

Then, the distinguishing advantage satisfies:

\[
\text{Adv}(D) \leq \Pr[X_{O_2} \in V_{\text{bad}}] + \varepsilon
\]

The core idea of the H-coefficient technique is: a large number of views are almost equally likely in both oracles (real worlds and the ideal world), and the odd ones occur with a small probability. Note that the partitioning of \(V\) into \(\text{bad}\) and \(\text{good}\) views is directly reflected in the terms \(\Pr[X_{O_2} \in V_{\text{bad}}]\) and \(\varepsilon\) in the bound: if \(V_{\text{good}}\) is too large, \(\varepsilon\) will become large, whereas if \(V_{\text{bad}}\) is too large, \(\Pr[X_{O_2} \in V_{\text{bad}}]\) will become large.

### 3. Construction Limitations

In this section, we will discuss the construction limitations of secure block cipher in the ideal cipher model, which is built on dedicated block cipher invocations and simple XOR operation. The XOR operation has efficiency benefits. The target construction is denoted as \(E[s]\) and is built on \(s\) block cipher invocations. Let \(E\) denote the underlying block cipher with \(n - \text{bit block size and } n - \text{bit key size. Let } p, c, \text{ and } k\) denote the plaintext, ciphertext, and key, respectively, where all have \(n - \text{bit size. Let } a_{i,j}\) and \(b_{i,j}\) be one bit variable of being 0 or 1, where \(1 \leq i \leq s + 1\) and \(1 \leq j \leq i + 2\). The encryption of \(E[s]\) is shown in Algorithm 1. The target construction \(E[s]\) is depicted in Figure 1. In detail, this is a graphical view from which we would acquire the resultant block cipher construction. Moreover, all the \(s\) block cipher invocations are involved in the computation of the ciphertext \(c\). The ciphertext \(c\) must be invertible and efficiently decrypted from plaintext \(p\) and key \(k\). There are some limitations for \(E[s]\) to achieve our goal:

- The plaintext \(p\) should be involved in exactly one XOR operation. The \(p\) involves in XOR operation, which gives \(x_i\) and corresponding \(y_i\). So, both outputs \((x_i, y_i)\) are called plaintext dependent
variable. On the other side, if a variable \( y_i \) is used to compute another variable \( x_j \) which depends on \( y_i \), then \( x_j \) and corresponding \( y_j \) would also be plaintext dependent variable. So, we cannot use plaintext dependent variable to produce any key or subkey, otherwise, constructions will not be efficient.

There should be at most one plaintext dependent variable produced from the XOR operation. Otherwise, the decryption process cannot efficiently decrypt because there is more than one variable.

![Target Construction Diagram](image)

**Algorithm 1:** \( \mathbb{E}[s](\cdot, \cdot) \)

**Input:** \( k, p, E(\cdot, \cdot), \) variables \( a_{ij} \) and \( b_{ij} \)

**Output:** ciphertext \( x_1 = a_{11,k} \oplus b_{11,k} @ b_{12,p} \)

1. \( x_1 = a_{11,k} \oplus b_{11,k} @ b_{12,p} \)
2. *for* \( i = 1 \) to \( s - 1 \), *do*
3. \( y_i = E(a_{11,k}, x_i) \)
4. \( x_{i+1} = a_{10,1,k} @ \sum_{j=2}^{i+1} a_{10,j,y_j-1} @ b_{10,1,k} @ b_{10,2,p} @ \sum_{j=3}^{i+2} b_{10,j,y_j-2} \)
5. *end for*
6. \( y_s = E(k_s, x_s) \)
7. \( c = b_{s0,1,k} @ b_{s0,2,k} @ \sum_{j=3}^{s+2} b_{s0,j,y_j-2} \)
8. return ciphertext \( c \)
In order to achieve the above goals among the instances of target construction, we adopted a heuristic approach. For the instances of \( E[s] \), we invoked only two block cipher to achieve \( 2^n \) provable security because \( s = 1 \) for instances of \( E[s] \) had most \( 2^{n/2} \) security. Thus, at least two block cipher invocations are required to bypass the birthday bound barrier.

We continued to examine the instances of \( E[2] \) and would not analyze the \( E[s > 2] \) instances unless investigated all the instances of \( E[2] \) and none of them achieve \( 2^n \) security. In fact, if some instances of \( E[2] \) achieves \( 2^n \) security, then there is no need to examine the other instances of \( E[2] \). To follow the above strategy, we analyzed the target construction \( E[s] \) and found 32 instances with \( 2^n \) provable security.

3.1. \( E[2] \) Instances

According to the previous discussion, the plaintext \( p \) should be involved in exactly one XOR operation. There should be, at most, one plaintext dependent variable produced from the XOR operation. Otherwise, the decryption process cannot efficiently decrypt because there exists more than one variable. The plaintext dependent variable cannot be used to produce any key-value; otherwise, constructions will not be efficient. Following this strategy, we divided \( E[2] \) instances into three types on the basis of when plaintext \( p \) is XOR to compute \( x \) and \( c \), respectively.

- Type 1 instances: when \( p \) is XOR to compute \( x_1 \)
- Type 2 instances: when \( p \) is XOR to compute \( x_2 \)
- Type 3 instances: when \( p \) is XOR to compute \( c \)

3.1.1. Type 1 Instances

According to the above limitation, the plaintext dependent variables cannot be used to produce key value, so, \( a_{2,2} = 0 \). The plaintext \( p \) should be involved in exactly one XOR operation, so, \( b_{2,2} = 0 \) and \( b_{3,2} = 0 \). We set \( b_{2,3} = 1 \), which is the first block cipher invocation, and set \( b_{3,1} = 1 \), which is second block cipher invocation. If \( b_{2,3} = 0 \), it means two block ciphers' invocations are parallel, and these instances are involved in type 2. It also shows that \( x_2 \) and \( y_2 \) are plaintext variables. Then, we set \( b_{3,3} = 0 \) because \( y_2 \) is already used as a plaintext dependent variable. All of these simplified constructions of type 1 are shown in Figure 2. We examined the instances of type 1, and ciphertext is computed as follows.

\[
c = E(a_{2,1}, k, x_2) \oplus b_{3,1}, k
\]

![Figure 2. E[2] : Type 1 Construction.](image)

Instances with one block cipher Invocation of type 1.

We would show that any instance that makes only one block cipher invocation of type 1 construction could not achieve BBB security. Let \( E : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^n \) be a block cipher, shown in Figure 3. We showed that there exists a distinguisher \( D \) that can distinguish any such block cipher from random permutation using at most \( 2^{n/2} \) queries.

- When \( a_{1,1} = 0 \) and \( b_{1,1} = 1 \).

In this case, we can see the input or output of \( E \) is not related to \( p \) or \( c \). When \( b_{1,2} = 0 \), then distinguisher \( D \) selects arbitrary \( p \) and \( p' \) to get \( c \) and \( c' \). If the event \( c = c' \) occurs, then output is 1; otherwise, it is 0. The success probability of \( D \) is 1 when interacts with \( 1 - 2^{-n} \). The results are similar for \( b_{2,3} = 0 \).
When \( a_{1,1} = 0 \) and \( b_{1,1} = 0 \).

In this case, we can see the input or output of \( E \) is independent of the key. When \( b_{1,2} = 1 \), the distinguisher \( D \) selects arbitrary \( x_1 \) and \( x'_1 \) to get \( y_1 \) and \( y'_1 \); then, it puts \( p = b_{1,2}^{-1} x_1 \) and \( p' = b_{1,2}^{-1} x'_1 \) to get \( c \) and \( c' \). If the event occurs, then output is 1, otherwise 0.

\[
\text{Event} = \begin{cases} 
  c + c' = b_{2,2}.y_1 + b_{2,3}.y'_1 & \text{if } b_{2,2}.x_1 = 0 \\
  c + c' = b_{2,2}.y_1 + b_{2,3}.y'_1 & \text{if } b_{2,2}.x_1 \neq 0 
\end{cases}
\]

The success probability of \( D \) is 1 when interacts with \( 1 - 2^{-n} \). Similar is the case for \( b_{2,1} = 0 \).

- When \( b_{2,2} = 0 \).

In this case, there exists a distinguisher \( D \), distinguishing the real world oracle \((E^x, E^x)\) from the ideal world oracle \((\pi^x, E^x)\) with some probability. The distinguisher \( D \) makes \( 2^n/2 \) queries and operates as follows. For \( j = 1, \ldots, 2^n/2 \), the distinguisher \( D \) selects arbitrary \( p(i) \) to get \( c(i) \). If \( c(i) \neq c(j) \) for all queries and its indices \( j \neq j' \), then output 1, otherwise output 0.

At the end of type 1 instances, we can conclude that the plaintext added in the first XOR operation and the output value after the first invocation of block cipher are included in second block cipher invocation as a key that is a plaintext dependent variable, so the advantage of the adversary is at most around birthday bound.

#### 3.1.2. Type 2 Instances

Following the construction limitations, set \( b_{3,5} = 1 \). The plaintext \( p \) should be involved in exactly one XOR operation, so, \( b_{1,2} = 0 \) and \( b_{3,2} = 0 \). We set \( b_{2,3} = 1 \), that is, the first block cipher invocation, and thus, we set \( b_{3,4} = 1 \), that is, second block cipher invocation. It also shows that \( x_1 \) and \( y_1 \) are not plaintext dependent variables. All of these simplified constructions of type 1 are depicted in Figure 4.

Here, we examined the type 2 instances. For these instances, we computed ciphertext as follows.

\[
c = E(a_{2,1}.k \oplus b_{3,3}.y_1, x_2) \oplus b_{3,1}.k \oplus b_{3,3}.y_1
\]

The first block cipher invocation is \( y_1 = E(a_{1,1}.k, b_{1,1}.k) \). Throughout all the instances of type 2, we call \( y_1 \) as a subkey that is obtained from the secret key \( k \) for those instances with \((a_{1,1}, b_{1,1}) \neq (0, 0)\).
However, the computation from $p$ to $x_2$ is $x_2 = p \oplus b_{2,1} k \oplus b_{2,3} y_1$, and $\Delta x_2 = \Delta p$ always holds and $\Delta y_2 = \Delta c$, respectively. Moreover, for any plaintext and ciphertext pair $(p, c)$ and $(p', c')$, the adversary knows the internal variable differences $\Delta x_2$ and $\Delta y_2$. Therefore, according to the above constraint, we can find some conditions on the type 2 instances to achieve $BBB$.

- When $(a_{1,1}, b_{1,1}) \neq (0,0)$.

  If $(a_{1,1}, b_{1,1}) = (0,0)$, then it means $y_1 = E(0,0)$. Adversary makes a query $(0,0)$ to $E(\cdot, \cdot)$ to get $y_1$, and the first block cipher invocation kicks off. Then, the instances are based on only a single block cipher invocation in the adversary view. As we discussed in the previous sections, when $s < 2$, the construction achieves security up to birthday bound.

- When $(a_{2,1}, a_{2,2}) \neq (0,0)$.

  If $(a_{2,1}, a_{2,2}) = (0,0)$, then adversary regards $b_{2,1} k \oplus b_{2,3} y_1$ and $b_{3,1} k \oplus b_{3,3} y_1$. So, the instance gives essentially one step of [42].

- When $(b_{2,1}, b_{2,3}) \neq (0,0)$.

  If $(b_{2,1}, b_{2,3}) = (0,0)$, then $p = x_2$, i.e., the adversary knows and can control the $x_2$ value. A distinguisher $D$ is launched and fixes two distinct $p$ and $p'$. The distinguisher $D$ queries to $E[2]_k(\cdot, \cdot)$ and gets ciphertext $c$ and $c'$ and stores $(c \oplus c')$, respectively. The $D$ makes a query for $E(\cdot, \cdot)$ and receives $\omega$ and $\omega'$, respectively, and matches $\omega \oplus \omega'$ to stored $c \oplus c'$. The distinguisher $D$ recovers $a_{2,1} k \oplus a_{2,2} y_1$. For any plaintext-ciphertext pair $(p, c)$ and $(p', c')$, the distinguisher $D$ can compute $z$ (such that $a_{2,1} k \oplus a_{2,2} y_1 = z$) and $z'$ and query $(z, p)$ and $(z', p')$ to $E(\cdot, \cdot)$, recovering $y_2$ and $y'_2$, respectively. So, the output of distinguisher $D$ is 1 if $c \oplus c' = y_2 \oplus y'_2$, otherwise, compute 0. When interacting with $E[2]$, then the output of distinguisher $D$ is 1 until it recovers $a_{2,1} k \oplus a_{2,2} y_1$. Thus, the success probability is $1 - (1 - 2^{-n})^2$.

- When $(b_{3,1}, b_{3,3}) \neq (0,0)$.

  This is a similar analysis which is presented above, where the adversary knows and has control over the value of $y_2$ and he fixes the ciphertext $c$ and $c'$ and queries to $E[2]^{-1}_k(\cdot, \cdot)$.

- When $(b_{2,1}, b_{2,3}) \neq (a_{2,1}, a_{2,2})$.

  If $(b_{2,1}, b_{2,3}) = (a_{2,1}, a_{2,2})$, it has $(b_{2,1} k \oplus b_{2,3} y_1) = (a_{2,1} k \oplus a_{2,2} y_1)$, which is denoted by $g$ and $x_2 \oplus z_2 = g \oplus p \oplus g = p$. Thus, the adversary knows and can control $x_2 \oplus z$. A distinguisher $D$ is launched and gives queries to $E[2]_k(\cdot, \cdot)$ and receives $c$ and $c'$ and stores $(c \oplus c')$, respectively. Moreover, $D$ sends distinct queries to $E(\cdot, \cdot)$ and receives $\omega$ and $\omega'$, respectively, and stores $(\omega \oplus \omega')$. Then, he matches $(\omega \oplus \omega')$ and $(c \oplus c')$. The $D$ can compute $x_2$ and $z$ for any plaintext-ciphertext and receive $y_2$ from $E(\cdot, \cdot)$. Moreover, the distinguisher $D$ just needs to make some extra queries. Thus, the success probability is trivially $1 - (1 - 2^{-n})^2$.

- When $(b_{3,1}, b_{3,3}) \neq (a_{2,1}, a_{2,2})$.

  This is also having a similar analysis as shown above.

Putting all the above properties of type 2 instances together, we got 32 instances, denoted by $E_1, E_2, \ldots, E_{32}$ and depicted in Figure 5. We investigated these constructions and found $2^n$ provable security. We used the H-Coefficient technique for proof, which is discussed in Section 4.
We use some extra queries. Thus, when interacting with the adversary, the distinguisher can compute $s'$ for any ciphertext and receive $c$. The success probability is $\mathbb{E}(c = \text{matches})$.

Figure 5. Cont.
Symmetry 2019, 11, x FOR PEER REVIEW 10 of 14

Figure 5. The $E_1, E_2, \ldots, E_{32}$ efficient construction: the internal variable $y$ is referred to as a subkey for these constructions.
3.1.3. Type 3 Instances

When \( p \) is XOR to compute \( c \), then \( b_{3,2}, p = 1 \), \( b_{1,2}, p = 0 \), and \( b_{2,2}, p = 0 \). The constructions of type 3 are depicted in Figure 6. In this construction, it could be seen that \( p \) and \( c \) are linearly related, and distinguisher \( D \) can distinguish by only two queries to \( E[2]_{\Delta_k}(\cdot, \cdot) \) with distinct plaintext \( p \) and \( p \oplus \Delta \), verifying \( \Delta c = \Delta \). Hence, the discussion of type 3 instances is omitted here.

\[
\begin{align*}
E &\xrightarrow{a_{1,1}, k} b_{1,1}, k \\
&\xrightarrow{b_{2,1}, k} a_{2,1}, k \oplus a_{2,2}, y_1 \\
&\xrightarrow{b_{3,4}, y_2} c
\end{align*}
\]

\( E[2] : \) Type 3 Construction.

4. Security Proof

Let \( E_1, E_2, \ldots, E_{32} \) is any instance, and \( E \) is an underlying block cipher. Let there be any distinguisher \( D \) that has access to oracles \( O_1 \) and \( O_2 \), either \( E_{\Delta_k}^\pm(\cdot, \cdot) \) with \( k \overset{\$}{\leftarrow} K \) or \( \pi_{\Delta_k}^\pm(\cdot, \cdot) \), \( E_{\pi_{\Delta_k}}^\pm(\cdot, \cdot) \). The distinguisher \( D \) is computationally unbounded and deterministic, making \( q \) queries when interacting with \( O_1 \) and \( O_2 \). We defined distinguisher queries to \( O_1 \) and \( O_2 \) as \( q_1 \) and \( q_2 \), respectively: \( q = q_1 + q_2 \) and do not contain duplicate queries. When distinguisher \( D \) interacts with \( O_1 \) and \( O_2 \), the queries response are \( v_1 = \{(p_1, c_1), \ldots, (p_{q_1}, c_{q_1})\} \) and \( v_2 = \{(w_1, \pi_{q_1}^{1}(\cdot)), \ldots, (w_{q_2}, \pi_{q_2}^{2}(\cdot))\} \), respectively. The \( v \) is the view denoting the transcripts, and in the end, the distinguisher \( D \) obtains a view \( v = (v_1, v_2) \). The distinguisher \( D \), based on the \( v \), computes its decision bit. Accordingly, the decision bit probability distribution of distinguisher \( D \) relies on the probability distribution of \( v \). The \( X \) and \( Y \) are the probability distribution on \( v \) when interacts with \( (E_{\pi_{\Delta_k}^\pm}(\cdot, \cdot), E_{\Delta_k}^\pm(\cdot, \cdot)) \) and \( (\pi_{\Delta_k}^\pm(\cdot, \cdot), E_{\pi_{\Delta_k}^\pm}(\cdot, \cdot)) \), respectively. We used \( V \) as an attainable view when \( D \) interacts with \( O_1 \), which is \( V = \{v | \text{Pr}[Y = v] > 0\} \) and \( V = V_{\text{good}} \cup V_{\text{bad}} \). The main goal of the proof is to disclose the subkey \( y \) and secret key \( k \) after interacting with \( O_1 \) and \( O_2 \).

In \( (\pi_{\Delta_k}^\pm(\cdot, \cdot), E_{\pi_{\Delta_k}^\pm}(\cdot, \cdot)) \) as \( (O_1, O_2) \), we chose \( k \overset{\$}{\leftarrow} K \) and got corresponding subkey \( y \) by querying \( E_{\pi_{\Delta_k}^\pm} \). The distinguisher \( D \) can easily derive query response \( (u, w) \) of \( E_{\pi_{\Delta_k}^\pm}(\cdot, \cdot) \) invocations for each query response \( (p_j, c_i) \) in view \( v_1 \). The query responses of a block cipher \( E \) for each view \( v = (v_1, v_2) \in V \) is divided into three tables. The first one consists of a single query response of block cipher \( E \):

\[
T^1 = \{(u^1_1, w^1_1) = y\}
\]

The second table consists of the other queries’ responses of block cipher \( E \) derived from \( v_1 \):

\[
T^2 = \{(u^2_1, w^2_1), \ldots, u^2_{q_2}, w^2_{q_2}\}
\]

The last table consists of all queries’ responses from \( v_2 \):

\[
T^3 = \{(u^3_1, w^3_1), \ldots, u^3_{q_2}, w^3_{q_2}\}
\]

4.1. Bad Events

\( v \in V_{\text{bad}} \) if there are following queries:

\[
T^1 = \{(u^1_1, w^1_1) = y\}, \quad T^2 = \{(u^2_1, w^2_1), \ldots, u^2_{q_2}, w^2_{q_2}\}, \quad \text{and} \quad T^3 = \{(u^3_1, w^3_1), \ldots, u^3_{q_2}, w^3_{q_2}\}
\]

such that the following condition holds: there exists \( (u^j_i, w^j_i) \) in table \( T^i \) and \( (u^j'_i, w^j'_i) \) in table \( T^i' \) such that \( (u^j_i, w^j_i) = (u^j'_i, w^j'_i) \) where \( i \neq i' \), then \( v \) causes bad event.

4.2. \( \text{Pr}[Y \in V_{\text{bad}}] \)

According to our construction, we gave here the exact definition of \( V_{\text{bad}} \), which also ensures the \( V_{\text{good}} \). The \( V_{\text{good}} \) does not cause bad event. Here, we defined the \( V_{\text{bad}} \) of \( E_1 \) only due to the limited space. At least, one event defines the \( V_{\text{bad}} \) if it exists.
(a) \((p_i, c_i) \in v_1\) such that \(p_i = y\);
(b) \((p_i, c_i) \in v_1\) such that \(c_i = k\);
(c) \((p_i, c_i) \in v_1\) and \((u_j, w_j) \in v_2\) such that \((u_j = p_i \oplus y)\)
(d) \((p_i, c_i) \in v_1\) and \((u_j, w_j) \in v_2\) such that \((w_j = c_i \oplus y \oplus k)\)

The subkey \(y\) and secret \(k\) are uniformly selected at random from a set of size of at least \(2^n - q - 1\). We get

\[
\Pr[(a)] \leq q/2^n - q - 1;
\]
\[
\Pr[(b)] \leq q/2^n - q - 1;
\]
\[
\Pr[(c)] \leq q/2^n - q - 1;
\]
\[
\Pr[(d)] \leq q/2^n - q - 1;
\]

Thus, we get

\[
\Pr[Y \in V_{bad}] \leq \Pr[(a)] + \Pr[(b)] + \Pr[(c)] + \Pr[(d)]
\]

Let \(q < 2^{n-1}\) and using above values, we get

\[
\Pr[Y \in V_{bad}] \leq \frac{4q}{2^n-1}
\]

4.3. Ratio for \(V_{good}\)

First of all, \(\Pr[X = v]\). The \(X\) is a random variable that is defined on the probability space of all possible underlying block cipher \(E\) and all possible secret key \(k\). The probability space of \(X\) is denoted as \(all_X\). Correspondingly, the \(|all_X|\) is equal to \(2^n (2^n!)^{2^n}\). In \(all_X\), an element \(\pi\) getting along with \(v\) is taken, if \(\pi\) gives exactly the same responses for all queries. The \(\text{comp}_X(v)\) is defined as all the elements in \(all_X\) compatible with \(v\).

\[
\Pr[X = v] = \frac{|\text{comp}_X(v)|}{|all_X|}
\]

Similarly, \(Y\) is defined on the probability space of \(E1\), underlying block cipher \(E\), and key \(k\). On defining \(\text{comp}_Y(v)\) and \(all_Y\), respectively, we have

\[
\Pr[Y = v] = \frac{|\text{comp}_Y(v)|}{|all_Y|}
\]

\(all_Y\) is \(2^n (2^n!)^{2^n} (2^n!)^{2^n}\), that is the number of keys times, the number of block ciphers. We next computed \(|\text{comp}_X(v)|\) and \(|\text{comp}_Y(v)|\). We knew that the view \(v\) contains the key \(k\) value, that is, at the end of the interaction, it is disclosed to distinguisher \(D\). A set of input outputs of underlying block cipher \(E\) are derived and separately stored in tables \(T^1\), \(T^2\), and \(T^3\). The number of input-output of \(E\) with the key value \(i\) is denoted as \(a_i\) and \(\beta_i\) in \(T^2\) and \(T^3\), respectively, where \(0 \leq i \leq 2^n - 1\). The \(\gamma_i\) denotes the number of queries to \(O_1\) with key value. There is no collision between any two tables, so \(v\) is good. Secondly, the distinguisher \(D\) never makes duplicate queries. Therefore, all the inputs and outputs of \(E\) in \(T^1\), \(T^2\), and \(T^3\) are distinct, showing that \(\gamma_i = a_i\). The query response \((u_{11}^1, w_{11}^1)\) of \(E\) in \(T^1\) has \(u_{11}^1 = k\) or \(w_{11}^1 = 0\) (\(E1\) to \(E20\) have \(u_{11}^1 = k\) and others \(u_{11}^1 = 0\)). On assuming \(u_{11}^1 = k\), we got

\[
|\text{comp}_X(v)| = (2^n - a_k - \beta_k - 1)! \prod_{i=0}^{k-1} (2^n - a_i - \beta_i)! \prod_{i=k+1}^{2^n-1} (2^n - a_i - \beta_i)!
\]
From \((2^n - \alpha)(2^n - \beta)\) \leq (2^n - \alpha - \beta)!(2^n)!, we have

\[\left|\text{comp}_{\gamma}(v)\right| \leq (2^n - \alpha_k - \beta_k - 1)! (2^n)^{2^n}\]

We can compute

\[
\frac{\left|\text{comp}_{\chi}(v)\right|}{\left|\text{comp}_{\gamma}(v)\right|} \geq \frac{(2^n - \alpha_k - \beta_k - 1)! \prod_{i=0}^{k-1} (2^n - \alpha_i - \beta_i)! \prod_{i=k+1}^{2n-1} (2^n - \alpha_i - \beta_i)!}{(2^n - \alpha_k - \beta_k - 1)! (2^n)^{2^n} \prod_{i=0}^{k-1} (2^n - \alpha_i - \beta_i)! \prod_{i=k+1}^{2n-1} (2^n - \alpha_i - \beta_i)!} = \frac{1}{(2^n)^{2^n}}
\]

Finally, we can compute

\[
\Pr[X = v] = \frac{\left|\text{comp}_{\chi}(v)\right|}{\left|\text{comp}_{\gamma}(v)\right|} \times \frac{\text{all}_{\gamma}}{\text{all}_{\chi}} 
\geq \frac{1}{(2^n)^{2^n}} \times \frac{2^n (2^n)!^{2^n} (2^n)!^{2^n}}{2^n (2^n)!^{2^n}} = 1
\]

Thus, it gives a ratio for \(V_{\text{good}} = 0\)

Combining both 4.2 and 4.3,

\[
\text{Adv}_{E_1}^{prp}(q) \leq \frac{4q}{2^n-1}
\]

**Author Contributions:** L.W. conceptualized the idea, Y.N. performed analysis, and both the authors wrote manuscript in coordination with each other.

**Funding:** National Nature Science Foundation of China, Youth Project.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

28. Patarin, J. Security in $O(2^n)$ for the xor of two random permutations$\setminus$proof with the standard h technique. IACR Cryptol. ePrint Arch. 2013, 2013, 368.