On the Optimality of Non-Linear Computations of Length-Preserving Encryption Schemes

Mridul Nandi

Indian Statistical Institute
mridul.nandi@gmail.com

Abstract. It is well known that three and four rounds of balanced Feistel cipher or Luby-Rackoff (LR) encryption for two blocks messages are pseudorandom permutation (PRP) and strong pseudorandom permutation (SPRP) respectively. A **block** is n-bit long for some positive integer n and a (possibly keyed) **block-function** is a nonlinear function mapping all blocks to themselves, e.g. blockcipher. XLS (eXtended Latin Square) encryption defined over two block inputs with three blockcipher calls was claimed to be SPRP. However, later Nandi showed that it is not a SPRP. Motivating with these observations, we consider the following questions in this paper: *What is the minimum number of invocations of block-functions required to achieve PRP or SPRP security over ℓ blocks inputs*? To answer this question, we consider all those length-preserving encryption schemes, called **linear encryption mode**, for which only nonlinear operations are block-functions. Here, we prove the following results for these encryption schemes:

1. At least 2ℓ (or $2\ell - 1$) invocations of block-functions are required to achieve SPRP (or PRP respectively). These bounds are also tight.
2. To achieve the above bound for PRP over $\ell > 1$ blocks, either we need at least two keys or it can not be *inverse-free* (i.e., need to apply the inverses of block-functions in the decryption). In particular, we show that a single-keyed block-function based, inverse-free PRP needs 2ℓ invocations.
3. We show that 3-round LR using a single-keyed pseudorandom function (PRF) is PRP if we xor a block of input by a masking key.

Keywords: XLS, CMC, Luby-Rackoff, PRP, SPRP, Blockcipher.

1 Introduction

For all symmetric key algorithms, domains (sometimes, also ranges) are desired to be sets of bit-strings of variable sizes. However, almost all known methodologies, known as **modes**, use one or more (usually keyed) functions defined over small and fixed lengths (e.g., blockcipher, compression function, permutations in sponge constructions etc.) in a black-box manner. In this paper we consider only nonlinear keyed function, called **block-function**, from $I_n := \{0,1\}^n$ (elements of the set are called **blocks**) to itself, for a *fixed positive integer n*.

Linear Encryption Mode. A **linear mode** (LM) is defined by an oracle algorithm which interacts with nonlinear block-functions as oracles such that all inputs of the block-functions are computed through linear functions of the previous obtained responses. Finally, the output is also computed as a linear function of all responses of block-functions and the input. Most of the known symmetric key encryptions (e.g., Luby-Rackoff (LR) [22, 27], CMC [15], EME [16, 13] etc.) are examples of length-preserving (the number of blocks of input and output are same) linear encryption modes. An encryption scheme is said to be (S)PRP or (strong) pseudorandom permutation [32, 33] if it is secure against adversaries making only plaintext queries (or both plaintext, ciphertext queries respectively). The building block keyed block-function is assumed to be PRP or PRF (pseudorandom function [12]).

Our Contribution. It is well known that three (or four) round Luby-Rackoff is PRP (or SPRP respectively) [22]. Ristenpart et al. proposed XLS [41] invoking three blockcipher and claimed to have SPRP security. Later Nandi [30] showed that it is not SPRP. Among all linear mode based length-preserving SPRP (the formal description for linear-mode is given in section 3), the CMC and four-round Luby-Rackoff require only 2ℓ calls for encrypting ℓ blocks and others requires more. So understanding optimality of SPRP and PRP is our main motivation of this paper.

(1) Optimality in PRP and SPRP. In section 4, we show that a linear mode based length-preserving PRP (or SPRP) over ℓ blocks must invoke n-bit building-blocks at least $2\ell - 1$ (or respectively, 2ℓ) times. This justifies why XLS or three rounds of Luby-Rackoff are not SPRP. This bound is tight as we know that three and four-rounds LR, CMC (for arbitrary block messages) etc. achieve these bounds.

(2) Optimality in Single-key Inverse-Free PRP. Inverse-free encryptions [6, 17, 22, 19] are useful in terms of implementation as we do not need to implement the inverse of the building-block for the combined implementation of encryption and decryption. LR is an example of an inverse-free construction but it requires at least two independent PRF. Nandi [27] has shown that two independent PRF are also required even for some other variants of LR rounds. In section 5, we generalize his result and show that any linear-mode based inverse-free single key length-preserving PRP over ℓ blocks requires at least 2ℓ invocations (which is actually same for SPRP constructions). This shows that PRP and SPRP becomes equally costly for single-keyed inverse-free encryptions.

(3) Three-round single-PRF based LR with a masking is PRP. The above observation says that to achieve inverse-free double-block PRP with three invocations, we can use two independent PRF (e.g., the constructions in [27] are such examples). Two independent keyed PRF may be more costly than one as it may require key-scheduling or key set-up algorithms [10, 42]. In the later part of the section 5, we show that the single PRF based three round LR is indeed PRP if we simply mask one block of the input by a masking key.

2 Preliminaries

A **block matrix** is a binary square matrix of size n. Let $\mathbb{M}_n(a, b)$ denote the set of all partitioned matrices $E_{a\times b}$ (of size $a \times b$ as a block partitioned matrix and of size $an \times bn$ as a binary matrix) whose $(i, j)^{\text{th}}$ entry, denoted $E[i, j]$, is a block-matrix for all $i \in [1..a] = \{1, \ldots, a\}$ and $j \in [1..b]$. The transpose of E, denoted E^{tr}, is applied as a binary matrix. Thus, $E^{tr}[i, j] = E[j, i]^{tr}$. Conventionally, any matrix $E_{a\times b}$ is written as the following block-wise partition matrices

$$E = \begin{pmatrix} E[1,1] & E[1,2] & \cdots & E[1,b] \\ E[2,1] & E[2,2] & \cdots & E[2,b] \\ \vdots & \vdots & \vdots & \vdots \\ E[a,1] & E[a,2] & \cdots & E[a,b] \end{pmatrix} := \begin{pmatrix} E[1,*] \\ E[2,*] \\ \vdots \\ E[a,*] \end{pmatrix} := \begin{pmatrix} E[*,1] & E[*,2] & \cdots & E[*,b] \end{pmatrix}$$

where $E[i, *]$ and $E[*, j]$ denote i^{th} block-row and j^{th} block-column respectively. For $1 \le i \le j \le a$, we also write $E[i..j\ ;\ *]$ to mean the sub-matrix consisting of all rows in between i and j. We simply write $E[..j\ ;\ *]$ or $E[i..\ ;\ *]$ to denote $E[1..j\ ;\ *]$ and $E[i..a\ ;\ *]$ respectively. Similar notation for columns are defined.

Definition 1. *A (square) matrix $E \in \mathbb{M}_n(a, a)$ is called* **(block-wise) strictly lower triangular** *if for all $1 \le i \le j \le a$, $E[i, j] = \mathbf{0}$ (zero matrix).*

For all $x = (x_1, \ldots, x_a) \in I_n^a$, we define a linear function mapping a blocks to b blocks as $E\cdot x = (y_1, \ldots, y_b)$. Here, we consider x and y as binary column vectors (we follow this convention which should be understood from the context). So *the block matrix $E[i, j]$ represents the contribution of x_j to define y_i*. More formally,

$$y_i = E[i, 1] \cdot x_1 + E[i, 2] \cdot x_2 + \cdots + E[i, a] \cdot x_a, \quad 1 \le i \le b.$$

If E is a strictly lower triangular matrix then y_i is clearly functionally independent of $x_i, \ldots, x_a$, $1 \le i \le a$. So if we associate y_i uniquely to each x_i (e.g., $y_i = \rho(x_i)$ for some function ρ) then the choice of the vectors x and y satisfying $E\cdot x = y$ becomes unique. This observation is useful while we define intermediate inputs and outputs of a black-box based construction.

2.1 Useful Properties of Matrices

It is well known that the maximum number of linearly independent (binary) rows and columns of a matrix $A \in \mathbb{M}_n(s, t)$ are same and this number is called rank of the matrix, denoted $\text{rank}(A)$. So clearly we have $\text{rank}(A) \le \min\{ns, nt\}$. By using Gaussian elimination method, denoted $x = \mathsf{solve}(A, b)$, we can solve for some x (not necessarily unique) of the system of solvable linear equations $A \cdot x = b$. By convention, whenever a non-zero solution exists it returns a non-zero solution. Note that if $w^{tr} = \mathsf{solve}(A^{tr}, b^{tr})$ then $w \cdot A = b$ (by applying transpose). The following results are straightforward and so we skip the proofs.

Lemma 1. *Let $A \in \mathbb{M}_n(s,t)$ and $r := \mathrm{rank}(A)$.*

(1) If $r < ns$ (i.e. presence of row-dependency) then $\mathsf{solve}(A^{tr}, 0)$ returns a non-zero element.

(2) Similarly for $r < nt$ (i.e. presence of column-dependency) $\mathsf{solve}(A, 0)$ returns a non-zero solution.

(3) Finally, let $r = nt$ (i.e., full column rank) and $b := A \cdot w$. Then, $\mathsf{solve}(A, b) = w$ (i.e., w is also the unique solution).

Lemma 2. *Suppose $A \in \mathbb{M}_n(s,s)$ is a non-singular matrix, i.e., rank(A) $= ns$. Let $t < s$ and*

$$B = \begin{pmatrix} A[..t,*] & \mathbf{0} \\ \mathbf{0} & A[..t,*] \\ A[t+1..,*] & A[t+1..,*] \end{pmatrix}$$

where $\mathbf{0}$ denotes the zero matrix of appropriate size. Then, $\mathrm{rank}(B) = n(s+t)$ (i.e., full row-rank).

2.2 Security Definitions

In this section we recall the security definitions of fixed length keyed constructions. One can also extend the definitions for variable length constructions.

PRF. We call an oracle algorithm $\mathcal{A}$ (t,q)-algorithm if it makes at most q queries and runs in time t. Let $\mathcal{K}$ be a key-space and $f : \mathcal{K} \times I_n^a \to I_n^b$ be a (keyed) function. We say that f is (q,t,ϵ)-PRF if for any (t,q)-algorithm $\mathcal{A}$ the **prf-distinguishing advantage**

$$\mathbf{Adv}_f^{\mathrm{prf}}(\mathcal{A}) := |\mathsf{Pr}[\mathcal{A}^{f_K} = 1; K \xleftarrow{\$} \mathcal{K}] - \mathsf{Pr}[\mathcal{A}^{g} = 1; g \xleftarrow{\$} \mathrm{Func}(a,b)]|$$

is at most ϵ where $\mathrm{Func}(a,b)$ denotes the set of all functions from I_n^a to I_n^b. We call randomly chosen g to be the (uniform) *random function*, denoted $\Gamma_{a,b}$. For notational simplicity, we skip the time parameter t which is irrelevant in this paper. We also simply write $\mathrm{Func} := \mathrm{Func}(1,1)$ and Perm to mean the set of all functions and permutations over I_n.

(S)PRP. A keyed permutation g over I_n^a is a function $g : \mathcal{K} \times I_n^a \to I_n^a$ such that for all key $k \in \mathcal{K}$, $g_k := g(K, \cdot) \in \mathrm{Perm}(a)$ (the set of all permutations over I_n^a). We denote the uniformly chosen permutation by Π_a and call *uniform random permutation.* A keyed permutation g is called (q,ϵ)-PRP if for any q-algorithm $\mathcal{A}$ the **prp-distinguishing advantage**

$$\mathbf{Adv}_g^{\mathrm{prp}}(\mathcal{A}) := |\mathsf{Pr}[\mathcal{A}^{g_K(\cdot)} = 1; K \xleftarrow{\$} \mathcal{K}] - \mathsf{Pr}[\mathcal{A}^{\Pi_a} = 1]|$$

is at most ϵ. By PRF-PRP switching lemma [4, 47], it is well known that $|\mathbf{Adv}_f^{\mathrm{prf}}(\mathcal{A}) - \mathbf{Adv}_f^{\mathrm{prp}}(\mathcal{A})| \le \binom{q}{2} 2^{-n}$. We define the **sprp-distinguishing advantage**

$$\mathbf{Adv}_f^{\mathrm{sprp}}(\mathcal{A}) := |\mathsf{Pr}[\mathcal{A}^{f_K, f_K^{-1}} = 1; K \xleftarrow{\$} \mathcal{K}] - \mathsf{Pr}[\mathcal{A}^{\Pi_a, \Pi_a^{-1}} = 1]|$$

and (q,ϵ)-SPRP.

2.3 Tools for Proving Security

Given a (t, q)-algorithm $\mathcal{A}$ interacting with an oracle $\mathcal{O}$ we denote the transcript $\tau(\mathcal{A}^{\mathcal{O}})$ by the random vector $((X_1, Y_1), \ldots, (X_q, Y_q))$ where $X_i \in I_n^a$ and $Y_i \in I_n^b$ are the i^{th} query made by and response obtained by $\mathcal{A}$ respectively. The following theorem, known as coefficient-H technique [34, 39] is very useful to show a construction is PRP or SPRP. It has also been adapted in [7, 24]

Theorem 1 (Coefficient-H Technique). *Let $f : \mathcal{K} \times I_n^a \to I_n^b$ be a keyed function and $\mathcal{V}_{\text{bad}} \subseteq (I_n^a \times I_n^b)^q$. Suppose*

1. *for all (t, q)-algorithm $\mathcal{B}$, $Pr[\tau(\mathcal{B}^{\Gamma_{a,b}}) \in \mathcal{V}_{\text{bad}}] \leq \epsilon_1$ and*
2. *for all $\tau = ((x_1, y_1), \ldots, (x_q, y_q)) \notin \mathcal{V}_{\text{bad}}$,*

$$Pr[f_K(x_1) = y_1, \ldots, f_K(x_q) = y_q; K \xleftarrow{\$} \mathcal{K}] \geq (1 - \epsilon_2) \times 2^{-nbq}.$$

Then, for all q-algorithm $\mathcal{A}$, $\mathbf{Adv}_f^{\text{prf}}(\mathcal{A}) \leq \epsilon_1 + \epsilon_2$.

3 Linear Mode

3.1 Linear Query and Mode

A block matrix $U \in \mathbb{M}_n(\ell, a+\ell)$ is called (a, ℓ)-**query function** if $U[*, a+1..]$ is block-wise strictly lower triangular. Here ℓ represents the number of queries and a represents the number of blocks in the input. For any such *query function*, an input $X \in I_n^a$, (and a tuple of ℓ functions $\tilde{\rho} = (\rho_1, \ldots, \rho_\ell)$ over I_n), we can *uniquely define* or associate u and v, called **intermediate input and output vector** respectively, satisfying (1) $U \cdot \binom{X}{v} = u$ and (2) $\tilde{\rho}(u) := (\rho_1(u_1), \ldots, \rho_\ell(u_\ell)) = v$. This can be easily shown by recursive definitions of u_i's and v_i's. More precisely, u_i is uniquely determined by $v_1, \ldots, v_{i-1}$ and X (through the linear function) and v_i is uniquely determined by u_i through ρ_i, for all $1 \leq i \leq \ell$. Informally, a (a, b, ℓ)-linear mode is a mode which takes a blocks input and returns b blocks output based on executing block-functions building blocks (see Fig 1 for an illustration of a linear mode). Formally, (a, b, ℓ)-linear mode is defined by a block matrix $E \in \mathbb{M}_n(\ell+b, \ell+a)$ where $E[1..\ell, *]$ is a (a, ℓ)-query function. For any ℓ-tuple of functions $\tilde{\rho} \in \text{Func}^\ell$, the corresponding linear-mode function $E^{\tilde{\rho}} : I_n^a \to I_n^b$ is defined as $E^{\tilde{\rho}}(X) = Y$ where

$$E \cdot \begin{pmatrix} X \\ v \end{pmatrix} = \begin{pmatrix} u \\ Y \end{pmatrix}, \quad \tilde{\rho}(u) = v.$$

So v is the intermediate output vector associated to the input X and the final output $Y := E[\ell+1.., *] \cdot \binom{X}{v}$, a linear function of v and X. Now we state an useful differential property of linear mode. Note that the functions of $\tilde{\rho}$ are non-linear and would be secret for the adversaries. So to obtain any information about the intermediate input and output, we only can equate intermediate outputs whenever two inputs collide for same function. Given any vectors x, x' of same size, we write Δx to mean $x \oplus x'$ and $\Delta_{a.b} x$ to mean $(x_a \oplus x'_a, \ldots, x_b \oplus x'_b)$. We simply write $\Delta_t x$ to mean $\Delta_{1..t} x$ (the first t elements of Δx).

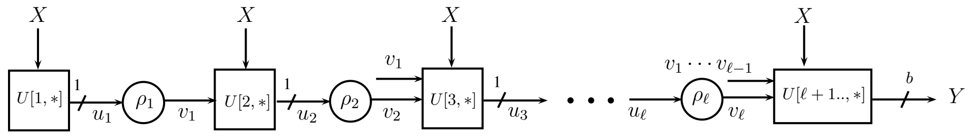

Fig. 1: Linear Mode: Here $U[i,*]$ means the i^{th} block row which maps $(X, v_1, \ldots, v_{i-1}, 0^{\ell-i+1})$ to u_i. Finally, $U[\ell+1..,*]$ maps the input X and intermediate output vector v to the output Y consisiting of b blocks.

Lemma 3. *Suppose $E[..t,*]\cdot X = E[..t,*]\cdot X'$ (i.e., $E[..t,*]\cdot \Delta X = 0$). Let $E^{\tilde{\rho}}(X) = Y$, $E^{\tilde{\rho}}(X') = Y'$. Let v, v' and u, u' denote intermediate outputs and inputs respectively associated with X and X' (for the function tuple $\tilde{\rho}$) respectively. Then, $\Delta_t u = \Delta_t v = 0^t$ and*

$$\Delta Y = E[\ell+1..\ ;\ ..a]\cdot \Delta X + E[\ell+1..\ ;\ a+t+1..]\cdot \Delta v_{t+1..}.$$

Proof. Due to choice of X and X', by induction one can show that $(u_1, v_1) = (u'_1, v'_1), \ldots (u_t, v_t) = (u'_t, v'_t)$ where u and u' denote the intermediate inputs associated with X and X' respectively (for the function tuple $\tilde{\rho}$). In other words, $\Delta_t u = \Delta_t v = 0^t$. Now, $Y = E[\ell+1..\ ;\ a+1..]\cdot v + E[\ell+1..\ ;\ ..a]\cdot X$ and similarly $Y' = E[\ell+1..\ ,\ a+1..]\cdot v' + E[\ell+1..\ ;\ ..a]\cdot X'$. The result is followed after we add these two equations and using that $\Delta_t v = 0^t$. □

3.2 Keyed Constructions Based on Linear Mode

Keyed Linear Mode. Let $\mathcal{F} = \mathcal{F}_1 \times \cdots \times \mathcal{F}_f$ and k be a non-negative integer where $\mathcal{F}_i \subseteq$ Func. A key-space $\mathcal{K}$ for any keyed function is of the form $I_n^k \times \mathcal{F}$. We call $\mathcal{F}$ the function-key space and I_n^k masking-key space. Any function g is also written as g^{+1}.

Definition 2. *Let $\mu : [1..\ell] \to [1..f]$, called* key-assignment function*, $\alpha := (\alpha_1, \ldots, \alpha_\ell) \in \{+1, -1\}^\ell$, called* inverse-assignment tuple*. For any function-key $\rho = (\rho_1, \ldots, \rho_f) \in \mathcal{F}$, we define $\rho_\mu^\alpha := (\rho_{\mu_1}^{\alpha_1}, \ldots, \rho_{\mu_\ell}^{\alpha_\ell})$. We denote the set of all functions ρ_μ^α by $\mathcal{F}_\mu^\alpha$.*

Here we implicitly assume that whenever $\alpha_i = -1$, ρ_{μ_i} is a permutation. If $\alpha = +1^\ell$, we simply skip the notation α. In general, the presence of inverse call of building blocks may be required when we consider decryption of keyed function. For the encryption, or a keyed function where decryption is not defined, w.l.o.g. we may assume that $\alpha = 1^\ell$.

Definition 3. *A (k, a, b) keyed linear mode with key-space $\mathcal{K}$, key-assignment function μ, is a $(a+k, b, \ell)$ linear mode E. For each key $\kappa := (L, \rho) \in \mathcal{K} := I_n^k \times \mathcal{F}$, we define a keyed function $E_\kappa(P) := E^{\rho_\mu}(L, P)$.*

Keyed linear mode E is actually a linear mode with a part of the input is the masking key and function tuples are also derived by reusing some keyed block-functions.

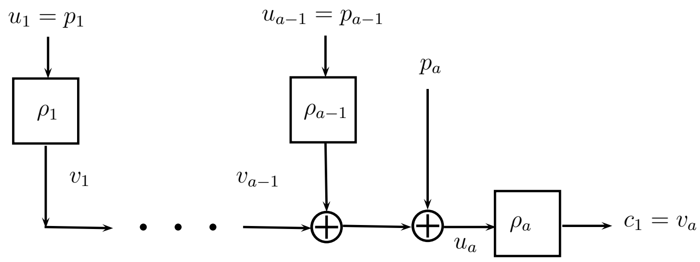

Fig. 2: PMAC. The input is $(p_1, \ldots, p_a)$ and the output is c_1.

Example 1. Consider the simple variant of PMAC [8, 43] defined over I_n^a. Let $(p_1, \ldots, p_a)$ be the input. For $1 \le i \le a-1$, $u_i = p_i$ and $u_a = p_a \oplus (\bigoplus_{i=1}^{a-1} v_i)$. Finally the output is defined as $c_1 = v_a$. Here $\ell = a$ and $b = 1$. There is no masking key, i.e. $k = 0$ and $f = a$ (all function-keys are independently chosen). In a single function-key version of PMAC (with independent masking key), we have $f = 1 = k$. The $u_i = \alpha^i \cdot L$ for $1 \le i < a$ and $u_a = p_a \oplus (\bigoplus_{i=1}^{a-1} v_i) \oplus L$.

Affine Domain Extension or ADE [28]. As defined in [28], affine domain extension over I_n^a is nothing but a $(a, 1, \ell)$-linear mode keyed function $\mathbf{E}$ such that the key-space is $\mathcal{K} = \mathcal{F} \subseteq$ Func, i.e., $f = 1$ (single function-key) and $k = 0$ (no masking key). Moreover, the final output is the response of the last oracle call, i.e. v_ℓ. One can consider an injective padding rule and sequence of such constructions indexed by a to incorporate variable length inputs. CBC-MAC [5], PMAC [8, 23, 31], TMAC [21], OMAC [18, 26], DAG-based constructions [20] etc. are some examples of ADE.

Length Preserving Linear Encryption Mode. A keyed linear mode E is called length-preserving (LP) encryption if E_κ is encryption scheme and $a = b$. In addition with these, we also assume that its decryption algorithm D is also a keyed linear mode which is indeed true for all known linear encryption modes. We first see an example below.

Example 2. As an example, consider Luby-Rackoff (LR) keyed function with three rounds using two random functions ρ_1, ρ_2, i.e. $f = 2$, $a = b = 2$ and $\ell = 3$ (three invocations of the underlying block-functions). Consider the key-assignment function π with $\pi_1 = 1, \pi_2 = 1$ and $\pi_3 = 2$. So the function tuple after applying the key-assignment is (ρ_1, ρ_1, ρ_2). As there is no masking key, we have $k = 0$. So the key-space is Func^2. Given $(p_1, p_2) \in I_n^2$ we define

$$u_1 := p_1, v_1 = \rho_1(u_1), u_2 = v_1 + p_2, v_2 = \rho_1(u_2), u_3 = v_2 + p_1, v_3 = \rho_2(u_3).$$

Finally, the output is (c_1, c_2) where $c_1 := u_3$ and $c_2 = v_3 + u_2$. This is clearly decryptable. Consider u_i's, v_i's and p_i's as variables. The ciphertext provides two linear functions of these variables, namely u_3 and $v_3 + u_2$. So u_3 is in the span. As u_3 is in the span, v_3 is also computable. Thus u_2 is in the span of the extended ciphertext including v_3. Again v_2 is computable and hence $u_1 := p_1$ is in the extended span. Finally, p_2 is in the span after including v_1. So we see that that decryption algorithm is also linear mode.

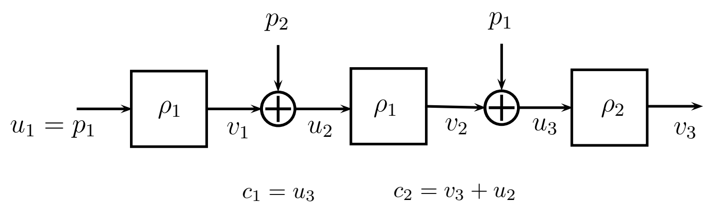

Fig. 3: LR with three round.

From the above example, it is clear that the intermediate input outputs for the building blocks would be same if we encrypt and then decrypt as we do in the correctness condition: $D_\kappa(E_\kappa(P)) = P$. Informally, if some input-output does not arise in the decryption then either this input-output is redundant in the encryption computation or the correctness condition does not hold (due to randomness of the output which has influence in the encryption but is not used in the decryption). We now describe the details of a length preserving linear encryption mode for which all invocations of block-function calls are not redundant.

Definition 4 (reordering of vectors). *Let* $\alpha := (\alpha_1, \ldots, \alpha_\ell) \in \{1, -1\}^\ell$, *and* $\beta = (\beta_1, \ldots, \beta_\ell)$ *be a permutation over* $[1..\ell]$. *A pair of vectors* $(w, z) \in I_n^{2\ell}$ *is* (a, β)*-reordering of a pair of vectors* $(u, v) \in I_n^{2\ell}$ *if*

$$(w_i, z_i) = \begin{cases} (u_{\beta_i}, v_{\beta_i}) & \text{if } \alpha_i = 1, \\ (v_{\beta_i}, u_{\beta_i}) & \text{if } \alpha_i = -1. \end{cases}$$

Definition 5. *A* $(k{+}a, a, \ell)$*-linear mode* E *is called* linear-mode length-preserving encryption *with key-space* $\mathcal{K} := I_n^k \times \mathcal{F}$ *and key-assignment* π *if the corresponding decryption algorithm* D *is also a* $(k + a, a, \ell)$*-linear mode with (1) an inverse assignment-tuple* $\alpha := (\alpha_1, \ldots, \alpha_\ell) \in \{1, -1\}^\ell$ *and (2) key-assignment* $\pi' := \pi \circ \beta$ *where* $\beta = (\beta_1, \ldots, \beta_\ell)$ *is a permutation over* $[1..\ell]$. *Moreover,* $\forall P \in I_n^a, L \in I_n^k, \rho = (\rho_1, \ldots, \rho_f) \in \mathcal{F}$,

$$E \cdot \begin{pmatrix} L \\ P \\ v \end{pmatrix} = \begin{pmatrix} u \\ C \end{pmatrix}, \ \rho_{\pi_1}(u_1) = v_1, \ldots \rho_{\pi_\ell}(u_\ell) = v_\ell \text{ if and only if}$$

$$D \cdot \begin{pmatrix} L \\ C \\ z \end{pmatrix} = \begin{pmatrix} w \\ P \end{pmatrix}, \ \rho^{\alpha_1}_{\pi'_1}(w_1) = z_1, \ldots, \rho^{\alpha_\ell}_{\pi'_\ell}(w_\ell) = z_\ell$$

where (w, z) *is* (a, β)*-reordering of* (u, v)*.*

The above definition implies that correctness condition of an encryption $D^{\rho^{\alpha}_{\pi'}}(L, E^{\rho}(L, P)) = P$. In addition with the correctness condition, the intermediate inputs and outputs for both encryption and decryption are simply reordered. In Example 2 (given above), we have $a = b = f = 2$, $\ell = 3$. For the decryption algorithm, we execute the function in the reverse order and so we set $\beta_1 = 3, \beta_2 = 2, \beta_1 = 3$. So the key-assignment function for the decryption is $\pi'_1 = 2, \pi'_2 = 2, \pi'_3 = 1$. We do not need to apply inverse for the decryption (it is called inverse-free) and so inverse-assignment tuple is 1^3. So if $(u_1, v_1), (u_2, v_2)$ and (u_3, v_3) are the intermediate input-output pairs for encryption then $(u_2, v_3), (u_2, v_2)$ and (u_1, v_1) (reordering of the previous pairs) are the intermediate input-output pairs for decryption.

EME [16], ELmE [11], AEZ [1], CMC [15] (these follow Encrypt-Mix-Encrypt paradigm), Luby-Rackoff with $a = b = 2$, unbalanced Fiestel [46, 17] etc. are some examples of length-preserving linear mode encryptions. HCBC1, HCBC2 [3], Modified-HCBC's, ELmD [1], MCBC [25], COPE [2] etc. are some examples of online computable length-preserving encryptions based on linear mode.

3.3 Known Results

Now we briefly revisit the related results known so far. Feistel structure is used to define different blockciphers e.g., Lucifer [48], DES etc. Later, Luby-Rackoff provides the PRP and SPRP security analysis of this type of ciphers and since then it is also popularly known as Luby-Rackoff (LR) cipher. There are many results known for security analysis of different rounds of LR and for different forms of Feistel structures [29, 6, 38, 37, 46]. Many results are known for reducing the key-sizes (i.e. reusing the round keys [35, 36, 40, 44]. Nandi [27] has characterized that all secure LR encryption schemes must have non-palindrome key-scheduling algorithms. Thus, we cannot use one single key.

XLS [41] is proposed to construct a generic encryption scheme which takes incomplete message blocks given that an encryption which can take complete message blocks. However, the result is shown to be wrong and some of implications (e.g., COPA [2] which use XLS) are shown [30].

This paper deals the optimality results for PRP and SPRP constructions. The optimality results for (delta) universal hash are known [29].

4 PRP and SPRP Distinguishing Attacks

Consider a length-preserving encryption scheme based on $(k+a, a, \ell)$ linear mode E. Now we show two main results in this section. Namely, we provide PRP and SPRP distinguishing attacks on the encryption scheme if $\ell \le 2a-2$. and $\ell \le 2a-1$ respectively. Thus, it gives lower bound on the number of invocations of building blocks for achieving PRP and SPRP security.

4.1 PRP Distinguishing Attack on E with $\ell = 2a - 2$

Let us assume $\ell = 2a - 2$. The attack can be trivially extended to all those constructions with $\ell < 2a - 2$. We recall that $E_L^{\tilde{\rho}}(P) = C$ if and only if

$$E \cdot \begin{pmatrix} L \\ P \\ v \end{pmatrix} = \begin{pmatrix} u \\ C \end{pmatrix}, \quad \tilde{\rho}(u) = v.$$

Distinguisher D_{prp} against $(k+a, a, 2a-2)$-linear mode E.

1. **step-1** (finding a suitable difference in a pair of plaintext queries): Let $d \in I_n^a$ be the non-zero solution of $\mathsf{solve}(E[..a-1, ..a], 0)$, i.e. $E[..a-1\ ;\ k+1..k+a] \cdot d = 0$. Such a non-zero solution exists as the number of columns is more than that of rows (see lemma 1).
2. **step-2** (make the queries with the difference obtained in step-1): Now the distinguisher makes two queries 0^a and d and obtains corresponding responses $c = E_L^{\tilde{\rho}}(0)$ and $c' = E_L^{\tilde{\rho}}(d)$. Let

   $$u_1, v_1, \ldots, u_{2a-2}, v_{2a-2}, \text{ and } u'_1, v'_1, \ldots, u'_{2a-2}, v'_{2a-2}$$

   denote the intermediate inputs outputs for the two queries respectively. By lemma 2, we have $1 \le i \le a-1$, $u_i = u'_i, v_i = v'_i$ and

   $$\Delta c = E[2a-1..\ ;\ k+1..(a+k)] \cdot d + E[2a-1..\ ;\ 2a+k..] \cdot \Delta v_{a..}$$

   while it is interacting with the keyed construction.
3. **step-3** (find a nullifier of unknown intermediate values): As the matrix $E[2a-1..\ ;\ 2a+k..]$ is $a \times (a-1)$ matrix, we find a non-zero binary vector $w \in \{0,1\}^{na}$ such that $w \cdot E[2a-1..\ ;\ 2a+k..] = 0$. In particular, $w = \mathsf{solve}(E[2a-1.., 2a+k..]^{\mathrm{tr}}, 0)$.
4. **step-4** (the distinguisher event): If $w \cdot \Delta c = w \cdot E[2a-1..\ ;\ k+1..(a+k)] \cdot d$ then it returns 1 (decision for the keyed construction), else returns 0 (decision for uniform random permutation).

The distinguishing advantage of the above distinguisher D is at least $1/2$ since for a random permutation $w \cdot \Delta c = w \cdot E[2a-1..\ ;\ k+1..(a+k)] \cdot d$ with probability $1/2$ whereas we have seen this holds with probability one for the keyed construction. When $a = 2$, we know that LR with three rounds is PRP. This shows the bound is tight at least for $a = 2$.

A General Distinguisher D_{prp} against $(k+a,a,\ell)$-linear mode E. Now we define a distinguisher against $(k+a,a,\ell)$-linear mode E assuming certain singularities in the sub-matrices.

Assumption: Suppose there exists an integer t such that

1. $\mathrm{rank}(E[..t, ..a]) < na$ and
2. $\mathrm{rank}(E[\ell+1.. \;;\; a+k+t+1..]) < na$.

Note the above assumption always holds for $t = a-1$ when $\ell \leq 2a-2$. However, if $\ell \geq 2a-1$, the both conditions not necessarily hold.

Whenever the assumptions hold, we have the following similar distinguisher as mentioned before. This distinguisher would be used later on while describing SPRP distinguishers.

Due to the assumptions. we can find d and w such that $E[..t, ..a]) \cdot d = 0$ and $w \cdot E[\ell+1.. \;;\; a+k+t+1..] = 0$. Then we make two queries 0 and d and obtain responses c and c'. The distinguisher returns 1 if $w \cdot \Delta c = w \cdot E[\ell+1..|k+1..(a+k)] \cdot d$, else 0.

4.2 SPRP Distinguishing Attack on E with $\ell = 2a-1$

Now we show that if $\ell < 2a$ then we have a SPRP distinguisher. In other words, $2a$ many invocations is minimum to achieve SPRP and which is tight as it is achieved in CMC. Consider a length-preserving $(k,a,2a-1)$-encryption scheme based on $(k+a,a,2a-1)$-linear mode E. Let us denote the $(k+a,a,2a-1)$-linear mode for its decryption by D. We describe three distinguishers depending on cases.

Case 1: rank($E[2a.. \;;\; 2a+k..]$) $< na$

In this case, the two assumptions, mentioned above, hold for $t = a-1$. So we have our general PRP distinguisher.

Case 2: rank($D[..a \;;\; k+1..k+a]$) $< na$

In this case, the two assumptions also hold for $t = a-1$ for the decryption function. So we have our general PRP distinguisher applied to the decryption function works.

Case 3: rank($D[..a \;;\; k+1..k+a]$) $= na$, rank($E[2a.. \;;\; 2a+k..]$) $= na$

Here we describe a SPRP distinguisher. Briefly, it works as follows. It first makes two queries as in step-2 (the first $a-1$ intermediate input and outputs are identical for two encryption queries). Using the invertible property we can actually obtain all the differences of intermediate values. As the computation of decryption algorithm must use same internal input and outputs of the building blocks, we also know the differences of intermediate inputs and outputs if we decrypt the first two encryption queries. Now we find another decryption query for which the first a intermediate input and output difference with one of the first two queries are fixed. So we can nullify the unknown $a-1$ differences and obtain a distinguishing event. The details are described below.

Distinguisher D_{sprp} **against** $(k+a, a, 2a-2)$**-linear mode** E.

1. **step-1** (make two queries with a certain difference, same as PRP distinguisher): Let $d \in I_n^a$ be the non-zero solution of $\mathsf{solve}(E[..a-1\ ;\ ..a], 0)$, i.e. $E[..a-1\ ;\ ..a] \cdot d = 0$. It makes two queries 0^a and d and obtains corresponding responses $c = E_L^{\tilde{\rho}}(0)$ and $c' = E_L^{\tilde{\rho}}(d)$.
   Let $u_1, v_1, \ldots, u_{2a-1}, v_{2a-1}$ and $u'_1, v'_1, \ldots, u'_{2a-1}, v'_{2a-1}$ denote the intermediate inputs outputs for the two queries respectively. By lemma 3, we have $1 \le i \le a-1$, $u_i = u'_i, v_i = v'_i$ and
   $$\Delta c = E[2a-1..\ ;\ k+1..(a+k)] \cdot d + E[2a..\ ;\ 2a+k..] \cdot \Delta v_{a..}$$
   while it is interacting with the keyed construction.
2. **step-2** (solve for Δu, Δv): Using the invertible property of $E[2a..\ ;\ 2a+k..]$, we can actually solve $\Delta v_{a..}$ and hence $\Delta u_{a..}$. Thus, we know Δu and Δv. Suppose we make two (redundant) decryption queries c and c' (whose responses must be 0 and d) and let $w_1, z_1, \ldots, w_{2a-1}, z_{2a-1}$ and $w'_1, z'_1, \ldots, w'_{2a-1}, z'_{2a-1}$ denote the intermediate inputs outputs for the two queries respectively. Then by the definition of decryption algorithm we also know Δw, Δz which are nothing but (β, π)-reordering of $(\Delta u, \Delta v)$.
3. **step-3** (find a difference for the final decryption query): Now we find a difference d' such that
   $$D[..a\ ;\ k+1..k+a+1] \cdot \begin{pmatrix} d' \\ \Delta z_1 \end{pmatrix} = \begin{pmatrix} \Delta w_1 \\ 0^{a-1} \end{pmatrix}.$$
   We can solve for a non-zero d'. This can be solved assuming that $\Delta w_1 \neq 0$ (see the remark below). Note that the matrix $D[..a\ ;\ k+1..k+a]$ is invertible. Now we make the decryption query $c'' = c' + d'$.. Let $w''_1, z''_1, \ldots, w''_{2a-1}, z''_{2a-1}$ denote the intermediate inputs outputs for the query and p denotes the response. By choice of d' we know that $z'_1 = z_1$ and $\Delta z'_{2..a} = 0^{a-1}$.
4. **step-4** (find a nullifier of unknown intermediate values, same as PRP distinguisher): As $D[2a..\ ;\ 2a+k..]$ is $a \times (a-1)$ matrix, we find a non-zero binary vector $w \in \{0,1\}^{nb}$ such that $w \cdot D[2a.., 2a+k..] = 0$.
5. **step-5** (the distinguisher event): If $w \cdot (p \oplus d) = w \cdot D[2a-1..\ ;\ k+1..(a+k)] \cdot d'$ then it returns 1 (decision for the keyed construction), else returns 0 (decision for uniform random permutation).

Remark 1. In the above attack we assume that $\Delta w_1 \neq 0$ since otherwise we do not get a non-zero d'. Note that Δw_1 can be written as a function of c and c'. So for a random permutation, a function of c and c' become zero has low probability. So we may assume that the $\Delta w_1 \neq 0$.

5 Security Analysis of Inverse-free Single Key Construction

5.1 PRP Attack of Single-Key Inverse-free Constructions without Masking

In the last section, we have seen that to obtain PRP, we need at least $2a - 1$ invocations and this is tight as three rounds of LR achieves this bound. Note that the three calls of the building block can not have same key. In [27], it is also shown that three rounds of LR-type rounds with same key building block can not be PRP. However, their result is applicable to a specific form of encryption schemes. Now, we generalize this result and show that any inverse-free single function-key (and no masking key) PRP requires at least $2a$ calls. In [27], there is a construction of inverse-free SPRP over two blocks invoking underlying function (single keyed) four times. So the bound is tight. Interestingly, the cost of PRP and SPRP become same when we want inverse-free single function-key constructions.

Consider a length-preserving encryption scheme based on $(a, a, 2a-1)$-linear mode E. Let us denote the $(a, a, 2a-1)$-linear mode for its decryption by D. Since it is inverse-free the inverse-assignment for the decryption is $\beta = (1, 1, \ldots, 1)$. As it is based on single function-key, the key-assignment is a constant function, i.e., $\pi_i = \pi'_i = 1$. However, there exists a permutation β over $[1..2a - 1]$. such that w and z are π-reordering of u and v respectively where u, v denote the intermediate input and output, respectively for $E^\rho(P) = C$ and similarly w, z for $D^\rho(C) = P$. We first briefly describe how we can construct a PRP-distinguisher (as like SPRP). The attack is similar to SPRP but we can not make decryption queries. We see how we can manage even if we are not allowed to make decryption queries.

We make two encryption queries such that $\Delta_{a-1}u = \Delta_{a-1}v = 0^{a-1}$. This is possible as we have a many plaintext blocks. Assuming some invertible property, we can find out the whole differences Δu and Δv for these two queries. For these two queries, if we look at the decryption computation then the first inputs, say w_1, w'_1 and their corresponding output differences Δz_1 (not the exact outputs) for both decryption are known (as there is no masking key). So now we make two encryption queries with the the following restrictions on intermediate values $\overline{u}, \overline{v}, \overline{u}'$ and $\overline{v}'$: $\overline{u}_1 = w_1, \overline{u}'_1 = w'_1$, $\Delta_{2..a}\overline{u} = \Delta_{2..a}\overline{u}'$, $\Delta_{2..a}\overline{v} = \Delta_{2..a}\overline{v}'$. As we have obtained differences for the first a inputs in a determined manner, we can nullify the remaining $a - 1$ intermediate differences and obtain a distinguishing event. The more details of the attack is given below depending on different cases. Note that the matrix $E \in \mathbb{M}_n(3a - 1, 3a - 1)$.

Case 1: rank(rank$\boldsymbol{E}$[$\boldsymbol{2a..}$; $\boldsymbol{2a..}$]) $<$ $\boldsymbol{na}$ In this case, the two assumptions, mentioned before, hold for $t = a - 1$. So we have our general PRP distinguisher.

Case 2: rank(rank$\boldsymbol{E}$[$\boldsymbol{1..a}$; $\boldsymbol{..a}$]) $<$ $\boldsymbol{na}$ In this case, the two assumptions also hold for $t = a$. So we have our general PRP distinguisher.

Case 3: rank(rank$\boldsymbol{E}$[$\boldsymbol{1..a}$; $\boldsymbol{..a}$]) $=$ $\boldsymbol{na}$, rank(rank$\boldsymbol{E}$[$\boldsymbol{2a..}$; $\boldsymbol{2a..}$]) $=$ $\boldsymbol{na}$ Here we describe a PRP distinguisher which works similar to SPRP distinguisher and as described above.

Distinguisher D_{prp} against $(a, a, 2a-1)$**-linear-mode** E **(with corresponding decryption mode** D.

1. **step-1** (make two queries with a certain difference, same as PRP distinguisher): Let $d \in I_n^a$ be the non-zero solution of $\mathsf{solve}(E[..a-1, ..a], 0)$, i.e. $E[..a-1, ..a] \cdot d = 0$. It makes two queries 0^a and d and obtains corresponding responses $c = E^\rho(0)$ and $c' = E^\rho(d)$.
   Let $u_1, v_1, \ldots, u_{2a-1}, v_{2a-1}$ and $u'_1, v'_1, \ldots, u'_{2a-1}, v'_{2a-1}$ denote the intermediate inputs outputs for the two queries respectively. By lemma 3, we have $1 \le i \le a-1$, $u_i = u'_i, v_i = v'_i$ and
   $$\Delta c = E[2a-1..|k+1..(a+k)] \cdot d + E[2a.., 2a+k..] \cdot \Delta v_{a..}$$
   while it is interacting with the keyed construction.
2. **step-2** (solve for Δu, Δv): Using the invertible property of $E[2a.., 2a..]$, we can actually solve $\Delta v_{a..}$ and hence $\Delta u_{a..}$. Thus, we know Δu and Δv. Now note that the first input of decryption D is only based on c and c'. Let β be the permutation corresponding to the reordering of intermediate input outputs for decryption. So the values of u_{β_1} and u'_{β_1} are known (as they depend only on c and c' due to no masking keys and inverse-free property). Moreover, we know Δv_{β_1}. Here we assume the difference Δu_{β_1} is non-zero, otherwise, we can have a different distinguishing event as zero difference can occur with low probability for random permutation.
3. **step-3** (find a difference for two more encryption queries): Now we find a solution p and p' such that
   $$\begin{pmatrix} E[1,*] & \mathbf{0} \\ \mathbf{0} & E[1,*] \\ E[2..a,*] & E[2..a,*] \end{pmatrix} \cdot \begin{pmatrix} p \\ p' \end{pmatrix} = \begin{pmatrix} u_{\beta_1} \\ u'_{\beta_1} \\ \mathbf{0} \end{pmatrix}.$$
   This can be solved as it has full column rank (see Lemma 2). Now we make two encryption queries p and p' and obtain outputs $\overline{c}$ and $\overline{c}'$. Let $\overline{u}, \overline{v}, \overline{u}'$ and $\overline{v}'$ be the intermediate inputs and outputs for these two queries respectively. So $\overline{u}_1 = u_{\beta_1}, \overline{u}'_1 = u'_{\beta_1}$, $\Delta \overline{v}_1 = \Delta v_{\beta_1}$ and $\Delta_{2..a}\overline{u} = \Delta_{2..a}\overline{v} = 0^{a-1}$. Thus, the a block output difference $\Delta \overline{c}$ depends only on the $a-1$ blocks of the intermediate output difference $\Delta \overline{v}_{a+1..}$.

4. **step-4** (find a nullifier of unknown intermediate values, same as PRP distinguisher): As $E[2a.., 2a+1..]$ is $a \times (a-1)$ matrix, we find a non-zero binary vector $w \in \{0,1\}^{nb}$ such that $w \cdot E[2a.., 2a+1..] = 0$.
5. **step-5** (the distinguisher event): If $w \cdot (p \oplus d) = w \cdot D[2a-1..|k+1..(a+k)] \cdot d'$ then it returns 1 (decision for the keyed construction), else returns 0 (decision for uniform random permutation).

5.2 PRP security of Single-Key Luby-Rackoff with Masking

Define one round Luby-Rackoff $\mathsf{LR}^f(a,b) = (b \oplus f(a), a)$ where $a, b \in I_n$ and $f \in \mathrm{Func}(a,a)$. In [27] it was shown that three rounds of some variants LR rounds with single function key is not PRP secure. In last section we have also generalized and showed that any encryption making three calls over two blocks input with key space $\mathcal{K} = \mathcal{F} = \mathrm{Func}(a)$ is not PRP secure. However, we now show that a simple variant of LR with a masking key becomes PRP secure.

Definition 6. *For any $f \in \mathrm{Func}(a)$, $L \in I_n$, we define*

$$\mathsf{LR}_L^{f,3}(a,b) = \mathsf{LR}^f(\mathsf{LR}^f(\mathsf{LR}^f(a+L,b))).$$

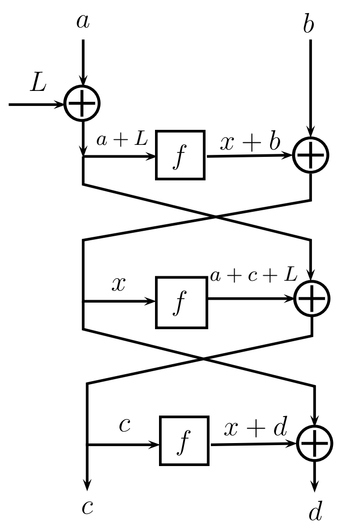

Fig. 4: LR-three rounds single function-key and one masking key.

Now we show that the above construction with key-space $\mathcal{K} = I_n \times \mathrm{Func}$ is PRP. Note that we have constant key-assignment (i.e., we reuse the PRF for all invocations) and also inverse assignment tuple is 1^3. Let f denote the uniform random function on I_n. Given a tuple of elements $c = (c_1, \ldots, c_t)$ we say that the event $\mathsf{coll}(c)$ holds if there exits $i \neq j$ such that $c_i = c_j$. We define

$$\mathcal{V}_{bad} = \{((a_1, b_1, c_1, d_1), \ldots (a_q, b_q, c_q, d_q)) \in I_n^{4q} : \mathsf{coll}(c)\}.$$

It is easy to see that for random function Γ_2 and a q-algorithm $\mathcal{A}$,

$$\mathsf{Pr}[\tau(\mathcal{A}^{\Gamma_2}) \in \mathcal{V}_{bad}] \leq \binom{q}{2} 2^{-n}.$$

Now we show the high interpolation probability of the variant of 3 round LR construction.

Proposition 1. *For all* $((a_1, b_1, c_1, d_1), \ldots (a_q, b_q, c_q, d_q)) \notin \mathcal{V}_{bad}$*, we have*

$$\mathsf{Pr}[\mathsf{LR}_L^{f,3}(a_i, b_i) = (c_i, d_i), 1 \leq i \leq q] \geq (1 - \epsilon)2^{-2nq}$$

where $\epsilon = 2q(q-1)2^{-n}$.

Proof. We say that a tuple $(L_0, (x_i)_{1 \leq i \leq q})$ is *admissible* if

1. $L_0 \notin \{a_i + c_j; 1 \leq i, j \leq q\} \cup \{a_i + x_j; 1 \leq i, j \leq q\}$,
2. x_i's are distinct and $x_i \neq c_j$, $1 \leq i, j \leq q$ and
3. whenever $a_i = a_j$, we have $x_i + x_j = b_i + b_j$.

Let $\mathcal{A}$ denote the set of admissible tuples. Let q_1 be the number of distinct a_i's. The number of $(L_0, x = (x_1, \ldots, x_q))$, denoted $N_{1,3}$, satisfying only (1) and (3) is at least $(2^n - q(q-1)) \times 2^{nq_1}$. So the number of admissible tuple is at least

$$(2^n - q(q-1)) \times 2^{nq_1} - (2^n - q(q-1)) \times 2^{n(q_1 - 1)} q(q-1).$$

We mainly subtract the number of tuples satisfying (1) and (3) and not satisfying (2) from $N_{1,3}$. So the number of admissible tuple is at least $2^{n(q_1+1)}(1-\epsilon)$ where $\epsilon = q(q-1)2^{-n+1}$.

Now, for any $\tau = ((a_1, b_1, c_1, d_1), \ldots (a_q, b_q, c_q, d_q)) \notin \mathcal{V}_{bad}$ we have

$$\mathsf{Pr}[\tau] \geq \sum_{(L_0, x) \in \mathcal{A}} \mathsf{Pr}[\tau, X_i = x_i, L = L_0] = \sum_{(L_0, x) \in \mathcal{A}} 2^{-n(q_1 + 2q + 1)}.$$

By using the lower bound of the number of admissible tuples we have

$$\mathsf{Pr}[\mathsf{LR}_L^{f,3}(a_i, b_i) = (c_i, d_i), 1 \leq i \leq q] \geq (1 - \epsilon)2^{-2nq}$$

where $\epsilon = 2q(q-1)2^{-n}$. □

Theorem 2. *For any q-adversary, the PRP advantage* $\mathbf{Adv}^{\mathrm{prp}}_{\mathsf{LR}_L^{f,3}}$ *against* $\mathsf{LR}_L^{f,3}$ *is at most* $\frac{2.5q(q-1)}{2^n}$.

Proof. Armed with the above result and using Coefficient-H technique the theorem follows. □

6 Conclusion

In this paper, we justify formally why we do not have any length-preserving PRP constructions more efficient than LR three rounds and length-preserving SPRP constructions more efficient than CMC or four round LR (in terms of the number of building block calls). We note that this optimality holds for all linear modes. In other words, it excludes the constructions which use field multiplications or nonlinear operations other than blockcipher calls (e.g., HCTR [49, 9], TET [14], HEH [45] etc.). We show that any such linear mode based constructions over ℓ blocks requires at leat $2\ell - 1$ blockcipher calls against chosen plaintext adversaries and at least 2ℓ blockcipher calls against chosen plaintext-ciphertext adversaries. This bounds are clearly tight as we know some constructions achieving the bound. Then we look into inverse-free single-key PRP constructions. Nandi has shown that three blockcipher call is no longer sufficient for LR-type constructions over two blocks (note that three call is sufficient using two independent PRF). We extend this result and show that any ℓ-block single-key inverse-free PRP must require 2ℓ calls like SPRP constructions. However, if we are allowed to use one masking key then we can have inverse-free PRP construction invoking only three blockcipher calls. We actually show that the three round LR using same keyed PRF is PRP if we mask a plaintext block by a masking key.

References

1. CAESAR submissions, 2014. http://competitions.cr.yp.to/caesar-submissions.html.
2. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. Parallelizable and authenticated online ciphers. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 424–443. Springer, 2013.
3. M. Bellare, A. Boldyreva, L. Knudsen, and C. Namprempre. On-line ciphers and the hash-cbc constructions. In *Advances in Cryptology –Crypto 2001*, number 2139 in Lecture Notes in Computer Science, pages 292–309, Berlin, 2001. Springer.
4. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Advances in Cryptology – Eurocrypt 2006*, number 4004 in Lecture Notes in Computer Science, pages 409–426, Berlin, 2006. Springer.
5. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining. In Yvo Desmedt, editor, *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 341–358. Springer, 1994.
6. Thierry P Berger, Marine Minier, and Gaël Thomas. Extended generalized feistel networks using matrix representation. In *Selected Areas in Cryptography–SAC 2013*, pages 289–305. Springer, 2014.
7. Daniel J. Bernstein. A short proof of the unpredictability of cipher block chaining (2005).

8. J. Black and P. Rogaway. A block-cipher mode of operations for parallelizable message authentication. In *Advances in Cryptology – Eurocrypt 2002*, number 2332 in Lecture Notes in Computer Science, pages 384–397, Berlin, 2002. Springer.
9. Debrup Chakraborty and Mridul Nandi. An improved security bound for HCTR. In Kaisa Nyberg, editor, *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 of *Lecture Notes in Computer Science*, pages 289–302. Springer, 2008.
10. Joan Daemen and Vincent Rijmen. The Design of Rijndael: AES - The Advanced Encryption Standard., 2002. http://csrc.nist.gov/CryptoToolkit/aes/rijndael/Rijndael-ammended.pdf.
11. Nilanjan Datta and Mridul Nandi. Misuse resistant parallel authenticated encryptions. *IACR Cryptology ePrint Archive*, 2013:767, 2013.
12. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986.
13. Shai Halevi. EME*: Extending EME to handle arbitrary-length messages with associated data. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT*, volume 3348 of *Lecture Notes in Computer Science*, pages 315–327. Springer, 2004.
14. Shai Halevi. Invertible Universal Hashing and the TET Encryption Mode. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 412–429. Springer, 2007.
15. Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer, 2003.
16. Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004.
17. Viet Tung Hoang and Phillip Rogaway. On generalized feistel networks. In *Advances in Cryptology–CRYPTO 2010*, pages 613–630. Springer, 2010.
18. T. Iwata and K. Kurosawa. Omac: One-key cbc mac. In *Fast Software Encryption, 10th International Workshop – FSE 2003*, number 2887 in Lecture Notes in Computer Science, pages 129–153, Berlin, 2003. Springer.
19. Tetsu Iwata and Kan Yasuda. Btm: A single-key, inverse-cipher-free mode for deterministic authenticated encryption. In *Selected Areas in Cryptography*, pages 313–330. Springer, 2009.
20. C. S. Jutla. Prf domain extension using dag. In *Theory of Cryptography: Third Theory of Cryptography Conference – TCC 2006*, number 3876 in Lecture Notes in Computer Science, pages 561–580, Berlin, 2006. Springer.
21. K. Kurosawa and T. Iwata. Tmac: Two-key cbc mac. In *Topics in Cryptology – CT-RSA 2003: The Cryptographers' Track at the RSA Conference 2003*, number 2612 in Lecture Notes in Computer Science, pages 33–49, Berlin, 2003. Springer.
22. M. Luby and C. Rackoff. How to construct pseudo-random permutations from pseudo-random functions. In *Advances in Cryptology – Crypto 1985*, number 218 in Lecture Notes in Computer Science, page 447, New York, 1984. Springer-Verlag.
23. K. Minematsu and T. Matsushima. New bounds for pmac, tmac, and xcbc. In *Fast Software Encryption – FSE 2007*, number 4593 in Lecture Notes in Computer Science, pages 434–451, Berlin, 2007. Springer.
24. M. Nandi. A simple and unified method of proving indistinguishability. In *Progress in Cryptology – Indocrypt 2006*, number 4329 in Lecture Notes in Computer Science, pages 317–334, Berlin, 2006. Springer.

25. Mridul Nandi. Two new efficient cca-secure online ciphers: Mhcbc and mcbc. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT*, volume 5365 of *Lecture Notes in Computer Science*, pages 350–362. Springer, 2008.
26. Mridul Nandi. Improved security analysis for OMAC as a pseudorandom function. *J. Mathematical Cryptology*, 3(2):133–148, 2009.
27. Mridul Nandi. The characterization of luby-rackoff and its optimum single-key variants. In Guang Gong and Kishan Chand Gupta, editors, *INDOCRYPT*, volume 6498 of *Lecture Notes in Computer Science*, pages 82–97. Springer, 2010.
28. Mridul Nandi. A unified method for improving PRF bounds for a class of blockcipher based macs. In Seokhie Hong and Tetsu Iwata, editors, *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*, pages 212–229. Springer, 2010.
29. Mridul Nandi. On the minimum number of multiplications necessary for universal hash functions. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, volume 8540 of *Lecture Notes in Computer Science*, pages 489–508. Springer, 2014.
30. Mridul Nandi. XLS is not a strong pseudorandom permutation. 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 478–490. Springer, 2014.
31. Mridul Nandi and Avradip Mandal. Improved Security Analysis of PMAC. Cryptology ePrint Archive, Report 2007/031, 2007. http://eprint.iacr.org/.
32. Moni Naor and Omer Reingold. A pseudo-random encryption mode. Manuscript available from www.wisdom.weizmann.ac.il/˜naor.
33. Moni Naor and Omer Reingold. On the construction of pseudorandom permutations: Luby-Rackoff revisited. *J. Cryptology*, 12(1):29–66, 1999.
34. J. Patarin. Etude des Générateurs de Permutations Basés sur le Schéma du D.E.S. Phd Thèsis de Doctorat de l'Université de Paris 6, 1991.
35. Jacques Patarin. New results on pseudorandom permutation generators based on the des scheme. In *Advances in Cryptology?CRYPTO?91*, pages 301–312. Springer, 1992.
36. Jacques Patarin. How to construct pseudorandom and super pseudorandom permutations from one single pseudorandom function. In *Advances in Cryptology?EUROCRYPT?92*, pages 256–266. Springer, 1993.
37. Jacques Patarin. Generic attacks on feistel schemes. In *Advances in CryptologyASIACRYPT 2001*, pages 222–238. Springer, 2001.
38. Jacques Patarin. Security of random feistel schemes with 5 or more rounds. In *Advances in Cryptology–CRYPTO 2004*, pages 106–122. Springer, 2004.
39. Jacques Patarin. The ?coefficients h? technique. In *Selected Areas in Cryptography*, pages 328–345. Springer, 2009.
40. Josef Pieprzyk. How to construct pseudorandom permutations from single pseudorandom functions. In *Advances in Cryptology?EUROCRYPT?90*, pages 140–150. Springer, 1991.
41. Thomas Ristenpart and Phillip Rogaway. How to enrich the message space of a cipher. In Alex Biryukov, editor, *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 101–118. Springer, 2007.
42. Ronald L. Rivest, Matthew J. B. Robshaw, and Yiqun Lisa Yin. RC6 as the AES. In *AES Candidate Conference*, pages 337–342, 2000.

43. Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes ocb and pmac. In Pil Joong Lee, editor, *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.
44. Babak Sadeghiyan and Josef Pieprzyk. A construction for super pseudorandom permutations from a single pseudorandom function. In *Advances in Cryptology?EUROCRYPT?92*, pages 267–284. Springer, 1993.
45. Palash Sarkar. Improving upon the tet mode of operation. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC*, volume 4817 of *Lecture Notes in Computer Science*, pages 180–192. Springer, 2007.
46. Bruce Schneier and John Kelsey. Unbalanced feistel networks and block cipher design. In *Fast Software Encryption*, pages 121–144. Springer, 1996.
47. Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/.
48. Arthur Sorkin. Lucifer, a cryptographic algorithm. *Cryptologia*, 8(1):22–42, 1984.
49. Peng Wang, Dengguo Feng, and Wenling Wu. HCTR: A variable-input-length enciphering mode. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *CISC*, volume 3822 of *Lecture Notes in Computer Science*, pages 175–188. Springer, 2005.