A Polynomial-Time Universal Security Amplifier in the Class of Block Ciphers

John O. Pliam*

EPFL-DSC Technical Report No. DSC/2000/013

March 23, 2000

Abstract

We demonstrate the existence of an efficient block cipher with the property that whenever it is composed with any non-perfect cipher, the resulting product is strictly more secure, against an ideal adversary, than the original cipher. We call this property universal security amplification, and note that it holds trivially for a one-time pad (a stream cipher). However, as far as we are aware, this is the first efficient block cipher with this property. Several practical implications of this result are considered.

Keywords: product ciphers, cascade ciphers, security amplification, guessing entropy.

1 Introduction

It is often asked in cryptography whether the product of two ciphers might be more or less secure than one of the ciphers by itself. An *amplification* of security doesn't happen in general and important counterexamples have been identified. For example, if the permutations of a block cipher form a group (or more precisely, are uniformly distributed on a subgroup of the symmetric group on the set of message blocks), then two-key double encryption is no better than single encryption. Thus, it has been seen as important to rule out this pathology in the case of DES [4]. Furthermore, the security of a product can actually be less than that of the second cipher when the plaintext statistics are ill-behaved with respect to the permutations of the first cipher [12]. Nevertheless, depending on how security is measured and how the ciphers are modeled, other affirmative results have been advanced [18, 8, 1].

In this paper, we take a novel approach to this problem, raising a strong existence question about the security of product ciphers. Specifically, we ask: Is there an efficient block cipher which amplifies the security, against an ideal adversary, of every non-perfect cipher with which it is composed? By construction, we answer this question in the affirmative. The constructed cipher, as presented, would not be widely viewed as "practical" because it requires a variable length key which grows with the amount of plaintext encrypted (much like a one-time pad). Furthermore, the computation time for encrypting a single message block – though polynomial in the inputs – is slower than what has become expected in the modern arena of fast encryption. On the other hand, it may be possible to accelerate bulk encryption through parallelization techniques, and if a cryptographically strong substitute for the key were used (such as a key schedule, hash, or pseudo-random function), then a negation of our claim of security amplification would imply that an attacker had defeated the artificial keying mechanism.

There are other practical implications of our result. First of all, the techniques used here could facilitate the construction of computationally efficient S-boxes with provably strong security properties. More generally, if we are to understand, in more than purely heuristic terms, the security convergence of modern iterated cryptosystems, then our result establishes new limits on what can be accomplished in polynomial-time. Our construction might be modified and compromised to obtain faster ciphers with complementary security results.

*with the Security and Cryptography Laboratory (LASEC), Swiss Federal Institute of Technology, Lausanne (EPFL), E-mail: `john.pliam@epfl.ch`.

2 Preliminaries

A basic familiarity with random variables and probability spaces [10] is assumed. Some group theory [17] is also assumed, but in the next subsection, we shall review some important terminology about permutation groups [7].

2.1 Permutation Groups

Let $\mathscr{X}$ be any set. The collection of all invertible functions on $\mathscr{X}$ forms the *symmetric group* $\mathfrak{S}_{\mathscr{X}}$. Any subgroup $G \leq \mathfrak{S}_{\mathscr{X}}$ is called a *permutation group*, and we also say that G *acts on* $\mathscr{X}$ and that $\mathscr{X}$ is a *G-set*. The subgroup of G which fixes a point $x \in \mathscr{X}$ is called the *(point) stabilizer* of x, and is given by $\mathrm{Stab}_G(x) = \{h \in G \mid hx = x\}$.

When studying n-bit block ciphers, the finite set $\mathscr{M} = \{0,1\}^n$ of all n-bit binary strings (or equivalently the integers $\{0, 1, \ldots, 2^n - 1\}$) is the most natural G-set for some permutation group $G \leq \mathfrak{S}_{\mathscr{M}}$. But additionally for this paper, we will often consider two other actions of G on related sets. By $\mathscr{M}^{(\ell)}$ we mean the set of tuples of size ℓ with distinct elements in $\mathscr{M}$, G acting elementwise. If $p = (p_1, \ldots, p_\ell) \in \mathscr{M}^{(\ell)}$, the point stabilizer $\mathrm{Stab}_G(p)$ is sometimes written $\mathrm{Stab}_G(p_1, \ldots, p_\ell)$. By $\mathscr{M}^{\{m\}}$ we mean the set of subsets of $\mathscr{M}$ of size m, where $g \in G$ acts on $S \in \mathscr{M}^{\{m\}}$ by taking $S \mapsto gS$. The point stabilizer of $S \in \mathscr{M}^{\{m\}}$ is sometimes written $\mathrm{Stab}_G\{S\}$.

2.2 Shannon's Model and Product Ciphers

Following Shannon [18], we model an n-bit *block cipher* as a $\mathfrak{S}_{\mathscr{M}}$-valued random variable. If a cipher X only takes values in a subgroup $G \leq \mathfrak{S}_{\mathscr{M}}$, then X may be called a *G-cipher*. We may model a stream cipher in the same spirit (cf. [13]). Let $\{0,1\}^*$ denote the (infinite) set of finite binary strings, and let $H \leq \mathfrak{S}_{\{0,1\}^*}$ be the subgroup of length-preserving permutations. We shall call an H-valued random variable a *stream cipher*[1]. By a *cipher* we mean either a block cipher or a stream cipher.

Given two independent ciphers X and Y acting on the same message space, the cipher XY is called a *product cipher*, Y is called its *first component* and X is called its *second component*. The distribution of the product of two block ciphers is given by the *convolution*,

$$\mathsf{P}[XY = g] = x * y(g) \overset{\triangle}{=} \sum_{h \in G} x(gh^{-1})y(h), \tag{1}$$

where $x(g) = \mathsf{P}[X = g]$ and $y(g) = \mathsf{P}[Y = g]$. This representation of a product cipher will prove useful in the sequel.

The cipher U which is uniformly distributed on $\mathfrak{S}_{\mathscr{M}}$ is called the *perfect cipher*. For any subgroup $G \leq \mathfrak{S}_{\mathscr{M}}$ the G-cipher U_G which is uniformly distributed on G is called the *uniform G-cipher*. Given a infinite sequence of independent and uniformly random bits, $z_0, z_1, \ldots$, we may form a simple stream cipher, called the *one-time pad*, by mapping plaintext word m into $z_{|m|} \oplus m$, where $z_{|m|}$ is the word $z_0 \cdots z_{|m|}$.

2.3 The Computational Model

Shannon's model is a purely probabilistic one; it says very little about how a computer might transform plaintext into ciphertext and back. For a cipher X to be practical, there should be effective procedures for encryption (computing the action of X on plaintext) and decryption (computing the action of X^{-1} on ciphertext).

One natural choice for the computational model is the standard *Turing machine* model [9]. Informally, we have an encryption algorithm Enc, which has as input arguments the plaintext m and the random key k, and which outputs ciphertext c. The corresponding decryption algorithm Dec is similarly defined. Formally in this model, we require a pair of deterministic Turing machines E and D, such that (under suitable encoding) $m = D(k, E(k, m))$, for all m and k. Notice that under this model, all randomness enters as an argument to the encryption and decryption algorithms, or equivalently as input data on the Turing machine tapes. Our view is that this model of computation

[1]In practice, a stream cipher will typically also have consistent *block prefix action*, i.e. for some integer n, it will be confined to permutations $h \in H$ such that when $|u| = |u'| \in n\mathbb{Z}$, $h(uw) = u'w'$ implies that for all v of length $|w|$, $h(uv) = u'v'$ for some v'.

is unnecessarily restrictive, because it fails to capture the simple idea that some ciphers (like the one-time pad) are "computationally efficient" even though they may require impractical amounts of key material to encrypt *every* possible plaintext.

Alternatively, we consider encryption and decryption algorithms which access key material as an auxiliary subroutine call. Formally, such a subroutine call is idealized by an *oracle function* $f : \{0,1\}^* \longrightarrow \{0,1\}$, and we are thus invoking the computational model of an *oracle Turing machine (OTM)* [9]. An OTM is a deterministic Turing machine augmented by an *oracle tape* and additional logic so that that at any time, the oracle tape with input α written on it can, in one step of computation, be transformed to have $f(\alpha)$ written on it. An OTM M with specific oracle function f will be denoted by M^f, and its time complexity is computed in the usual way (with oracle evaluation counting as one step). We may model uncertainty about the oracle function by treating it as an instance of a *random oracle function* $F : \{0,1\}^* \longrightarrow \{0,1\}$.

The next two definitions capture our intuitive notion of efficient encryption/decryption for block and stream ciphers, respectively.

Definition 1 (Efficient Block Ciphers) *A ensemble of block ciphers* $\{X_n\}_{n\in\mathbb{N}}$ *will be called* **computable in polynomial-time** *if there exists a random oracle function* F *and a pair of polynomial-time OTM's,* E *and* D*, such that for each* $n \in \mathbb{N}$*: (i). for each* $p \in \{0,1\}^n$, $p = D^F(E^F(p))$*, and (ii). the distribution of* E^F*, restricted to strings of length* n*, identical to that of* X_n*, and (iii). the distribution of* D^F*, restricted to strings of length* n*, is identical to that of* X_n^{-1}. □

By a mild but common abuse of notation, a block cipher X acting on $\{0,1\}^n$ will be called *computable in polynomial-time* if it is one of an ensemble of such ciphers, and any important properties hold for each representative.

Definition 2 (Efficient Stream Ciphers) *A stream cipher* X *will be called* **computable in polynomial-time** *if there exists a random oracle function* F *and a pair of polynomial-time OTM's,* E *and* D*, such that: (i). for each* $p \in \{0,1\}^*$, $p = D^F(E^F(p))$*, and (ii). the distribution of* E^F *is identical to that of* X*, and (iii). the distribution of* D^F *is identical that of* X^{-1}. □

Note that by Defs. 1 and 2, both the one-time pad and the Luby-Rackoff construction [13] are efficient. In fact, each is computable in linear time. Notice also that being computable in polynomial-time does not preclude that exponentially many bits may be necessary to completely describe the cipher's action on the entire message space. For example, each round of the Luby-Rackoff construction (a Feistel cipher with a perfectly random function acting on half-words) takes on one of

$$\left(2^{\frac{n}{2}}\right)^{2^{\frac{n}{2}}}$$

distinct permutations of an n-bit message space. Thus, for the common 3-round version of the construction, there must be $3n2^{\left(\frac{n-2}{2}\right)}$ bits to entirely describe it.

However, neither the one-time pad nor the Luby-Rackoff construction meets our objective. The one-time pad is not a block cipher. Furthermore, every permutation of the Luby-Rackoff construction is even and hence is confined to a proper subgroup (the alternating group, $\mathfrak{A}_{\mathscr{M}} \leq \mathfrak{S}_{\mathscr{M}}$), and we shall see from Lemma 1 below that it cannot be an universal security amplifier.

2.4 Optimal Chosen Plaintext Attacks

We now introduce the measure of security in terms of which strict security inequalities will be derived. Informally, it is just the average cost of the optimal (non-adaptive) chosen plaintext attack for an adversary in possession of an oracle which will answer the question, "is $X = g$?". There are two stages to the optimal strategy. First the adversary discards all permutations which are inconsistent with the acquired plaintext-ciphertext pairs. Then among the remaining permutations, he queries the oracle for the exact permutation in order of non-increasing probability. The adversary will obviously choose the plaintexts such that the average cost of this strategy is minimized. The difficulty of this attack is a direct and meaningful measure of the cipher's security.

To formally quantify this attack against a G-cipher X, $G \leq \mathfrak{S}_{\mathscr{M}}$, let us assume that the adversary has collected ℓ plaintexts and their corresponding ciphertexts into tuples $p, c \in \mathscr{M}^{(\ell)}$, respectively. The ciphertext tuple c is an instance of the random variable $C^\ell = Xp$, whose uncertainty is due exclusively to uncertainty about X. Now for any random variable Z the average cost of guessing its value is called the *guesswork*[2] of Z and is given by

$$W(Z) \triangleq \sum_{i=1}^{m} p_{[i]} i, \tag{2}$$

where Z takes on m values, and where the probabilities of Z have been arranged according to $p_{[i]} \geq p_{[j]}$ for all $i < j$. For fixed p and c, the *conditional guesswork* $W(X|c,p)$ is the guesswork of X as in Equation (2) after discarding all permutations $g \in G$ such that $c \neq gp$, and then rearranging and rescaling the probabilities accordingly. Now we must still account for the uncertainty about C^ℓ. Evidently, for a particular choice of plaintext tuple p, the cost of the attack must be weighted by the *a posteriori* probabilities $\omega(c|p) = \mathsf{P}\left[C^\ell = c \mid p\right]$, yielding

$$W(X|C^\ell, p) = \sum_{c \in \mathscr{M}^{(\ell)}} W(X|c,p)\omega(c|p). \tag{3}$$

The minimum value of $W(X|C^\ell, p)$ is the *optimal chosen plaintext attack work factor*, which will be denoted

$$\theta_\ell(X) = \min_{p \in \mathscr{M}^{(\ell)}} W(X|C^\ell, p). \tag{4}$$

For continuity we take $\theta_0(X)$ to be $W(X)$.

3 The Main Result

3.1 The Existence Theorem

We shall prove by construction the following theorem.

Theorem 1 *There is a cipher X, computable in polynomial-time, such that for each $0 \leq \ell \leq 2^n$ and every independent cipher Y, $\theta_\ell(XY) \geq \theta_\ell(Y)$. Furthermore, equality holds iff $\theta_\ell(Y) = \theta_\ell(U)$.*

It is easily seen (see e.g. [15]) that no non-perfect cipher Y can have $\theta_\ell(Y) = \theta_\ell(U)$, for all ℓ. Thus this theorem tells us in a very meaningful way, that every non-perfect cipher is brought closer to the the perfect cipher by left multiplication by X.

The proof of Thm. 1 relies on three lemmas which treat different aspects of the problem. To express these lemmas succinctly, we introduce some additional terminology. First, the *support* of a G-cipher (or indeed any random variable) may be defined as $\operatorname{supp}(X) \triangleq \{g \in G \mid \mathsf{P}[X = g] \neq 0\}$. Second, it is useful to denote the size of the smallest ℓ-message stabilizer of a group by $M_G(\ell) \triangleq \min_{p \in \mathscr{M}^{(\ell)}} |\mathrm{Stab}_G(p)|$. It is easily seen that $\theta_\ell(U_G) = \frac{1}{2}[1 + M_G(\ell)]$.

The first lemma from [15] treats the case $\ell = 0$ but is also useful in establishing the other results.

Lemma 1 *Given $G \leq \mathfrak{S}_{\mathscr{M}}$, let X be a G-cipher. Every independent non-uniform G-cipher Y satisfies $W(XY) > W(Y)$, iff for each $g \in G$ and each subgroup $H \neq G$, $\operatorname{supp}(X) \not\subseteq gH$.*

The next lemma provides sufficient conditions for nearly universal amplification ($\ell > 0$) for ciphers in any permutation group.

Lemma 2 *For a permutation group $G \leq \mathfrak{S}_{\mathscr{M}}$, let X be a G-cipher such that $\operatorname{supp}(X) = G$. Then for each $1 \leq \ell \leq 2^n$ and every independent G-cipher Y, $\theta_\ell(XY) \geq \theta_\ell(Y)$. Furthermore, equality holds iff $\theta_\ell(Y) = \theta_\ell(U_G)$.*

The final lemma asserts the existence of a cipher suitable to translate Lemmas 1 and 2 into Thm. 1.

[2]Guesswork has sometimes been called *guessing entropy*, cf. [16] and [3].

Lemma 3 *There is a cipher X, computable in polynomial-time, with* $\operatorname{supp}(X) = \mathfrak{S}_{\mathscr{M}}$.

Assuming the validity of the above lemmas, the proof of Thm. 1 is immediate.

The proof of Lemma 2 is rather involved and is sketched in Sect. 4.4. Most of the rest of this paper is devoted to the construction of X and the proof of Lemma 3. Before diving into the precise details in Sect. 4, let us first take a slightly more informal look at the ideas underlying this construction.

3.2 An Intuitive Glimpse at the Construction

The symmetric group on the message space is truly enormous. It's size is approximated by

$$\log\log(2^n!) \approx n + \log(n) = O(n).$$

Because it takes two logarithms to bring $2^n!$ down to the polynomial n, our construction will exhibit two distinct sources of algorithmic efficiency:

1. *Recursion:* The cipher X will be recursively defined as the product of simpler ciphers. More precisely, the encryption algorithm Enc will itself be recursive but will also call another recursive algorithm $\mathsf{invSort}$. The decryption algorithm Dec will be similarly defined. The time complexity and recursion depth of each algorithm will be a polynomial in n.

2. *Oblivious Action*[3]*:* The cipher X will be representable as the product of a large number of random powers of transpositions (i.e. permutations of message blocks two at a time). Then Enc and Dec, the defining algorithms of X, will make use of polynomially many transpositions for every block encrypted.

Let $G \leq \mathfrak{S}_{\mathscr{M}}$ be any permutation group. There are many ways to construct a product cipher PQ which achieves every permutation in G, even though *both* P and Q are sparse on G. Indeed for any subgroup $H \leq G$, we may take Q which achieves every permutation in H and P which achieves one permutation in every left coset of H in G. It is easy to see that PQ achieves every permutation in G. For many large groups, it is possible to find subgroups satisfying $|G| \gg |H|$ and $|G| \gg [G:H]$. Formally, we have an amplification of support: $|\operatorname{supp}(PQ)| \gg |\operatorname{supp}(P)|$, and $|\operatorname{supp}(PQ)| \gg |\operatorname{supp}(Q)|$. Thus by exploiting the algebraic structure of the group, we may construct a densely distributed cipher as a product of very sparsely distributed ciphers.

Let's try to carry this idea even further. Consider a chain of subgroups of G

$$\{1\} = H_0 \leq H_1 \leq \cdots \leq H_m = G,$$

and for each i, an H_i-cipher P_i which contains one permutation in every left coset of H_{i-1} in H_i. Then by simple induction, the product cipher $P_m \cdots P_2 P_1$, would have complete support on G. For example in the symmetric group on 2^n symbols, consider the subgroups $H_i = \operatorname{Stab}(1, \ldots, 2^n - i)$, $0 \leq i \leq 2^n$. On the one hand, this choice of subgroups is promising because the number of cosets in $\mathfrak{S}_{2^n}$ of the largest proper subgroup is the polynomial n. Unfortunately however, there are 2^n subgroups in this chain, and so the number of terms in the product $P_m \cdots P_2 P_1$ grows exponentially with n. If we are to employ this technique, it may be inconvenient to use a chain of subgroups which fix collections of words in $\mathscr{M}$ – either as tuples or as sets – because any hierarchy of such collections would typically be as large as $\mathscr{M}$ itself.

It thus makes more sense to define subgroups which fix some feature of the words in $\mathscr{M}$. To that end define K_i to be the subgroup consisting of the permutations of $\mathfrak{S}_{\mathscr{M}}$ which preserve the first $n-i$ bits of each message block. We shall call K_i the $(n-i)$*-bit prefix stabilizer subgroup* of $\mathfrak{S}_{\mathscr{M}}$, and as i ranges from 0 to n these form the chain of subgroups

$$\{1\} = K_0 \leq K_1 \leq \cdots \leq K_n = \mathfrak{S}_{\mathscr{M}}. \tag{5}$$

[3]We borrow this term from [14] where it is used in the same context.

We will construct, for each $1 \leq i \leq n$, a K_i-cipher P_i which contains one permutation in every left coset of K_{i-1} in K_i. Then the cipher of Lemma 3 will be defined as

$$X = P_n \cdots P_2 P_1. \tag{6}$$

But let us compute the minimal support required of P_n. That is to say let us count the number of left cosets of K_{n-1} in $\mathfrak{S}_{\mathscr{M}}$. Since K_{n-1} permutes all but the most significant bit of words in $\mathscr{M}$, the left cosets of K_{n-1} are characterized by the rearrangements of $\mathscr{M}$ with distinct patterns of the most significant bit. There are precisely

$$[\mathfrak{S}_{\mathscr{M}} : K_{n-1}] = \binom{2^n}{2^{n-1}}$$

of these rearrangements. Observe that while we have reduced the number of permutations by a large number (by $(2^{n-1}!)^2$ in fact), on a doubly logarithmic scale we still have

$$\log\log \binom{2^n}{2^{n-1}} \approx n + 1 = O(n).$$

It may appear that we are right back where we started, yet we have transported the problem onto very fertile new ground.

The efficiency in our algorithms for P_i has its heritage in the closely related problem of card shuffling. In fact, both the security of a product cipher [15] and the fairness of a shuffled deck of cards [2, 6] is related to the uniformity of convolutions as in equation (1). In their now famous analysis riffle shuffles, Aldous and Diaconis remarked that "the lovely new idea here is to consider shuffling as inverse sorting." [2, Remark (a), p. 344]. Indeed it is quite natural to consider *encryption* as inverse sorting because the rearrangements of $\mathscr{M}$ which characterize the left cosets of $K_{n-1} \leq \mathfrak{S}_{\mathscr{M}}$ correspond precisely with the permutations which would be used in the *first* step of the obvious recursive sorting algorithm. In the reverse order, we may achieve all permutations of $\mathscr{M}$ by first achieving all rearrangements of the most significant bit, and then proceeding recursively with the less significant bits. What we claim is that sorting and inverse sorting on the most significant bit can be done in polynomial-time using both recursion and the oblivious action of transpositions. The rest is gravy.

Let us demonstrate this efficiency in a simple example with $n = 3$ and thus $\mathscr{M} = \{0, 1, \ldots, 7\}$. We start with a random arrangement $(6, 3, 5, 0, 7, 1, 4, 2)$ of the elements of $\mathscr{M}$, and attempt to sort this tuple on the most significant bit by the application of $n = 3$ rounds of involutions (recall that every involution is a product of disjoint transpositions). For reasons of efficiency we shall restrict ourselves to transpositions of the form $(j, j \oplus 2^i)$, with i constant for every round. The allowable round involutions are $(01)^{b_1}(23)^{b_2}(45)^{b_3}(67)^{b_4}$, $(02)^{b_5}(13)^{b_6}(46)^{b_7}(57)^{b_8}$ and $(04)^{b_7}(15)^{b_9}(26)^{b_{10}}(37)^{b_{11}}$, for rounds 0, 1 and 2, respectively. Table 1 below shows that we can indeed sort on the most significant bit of 2^n integers by carefully choosing the powers b_i in only only n rounds.

To overcome the limitations of having so few permutations, our strategy is as follows: the goal at the end of round 1, is to collect integers with leading 1 into the lowest part of the bottom half (those positions ≤ 3), and to collect integers with leading 0 into the lowest part of the top half (those positions ≥ 4). Then the powers of the transpositions in the final round (round 2) are determined by the sorting requirement. We claim that this strategy will work for all n.

4 The Construction Details

In the next two subsections we present the detailed construction of the cipher X of Lemma 3. In the following two subsections we prove Lemma 3 and sketch the proof of Lemma 2, respectively.

		round (transp./arrangemnt.)					
		0		**1**		**2**	
7	$2 = 010$	(67)	100	()	100	()	100
6	$4 = 100$	(67)	010	(46)	111	()	111
5	$1 = 001$	()	001	()	001	(15)	101
4	$7 = 111$	()	111	(46)	010	(04)	110
3	$0 = 000$	(23)	101	(13)	011	()	011
2	$5 = 101$	(23)	000	()	000	()	000
1	$3 = 011$	()	011	(13)	101	(15)	001
0	$6 = 110$	()	110	()	110	(04)	010

Table 1: A randomly chosen arrangement of $\{0, 1, \ldots, 7\}$ is sorted with respect to the most significant bit after the application of only 3 rounds of disjoint transpositions. The first two columns indicate the initial arrangement (position, value). The next three columns give, for each round, the transposition affecting the value at that position and subsequent arrangement.

4.1 Algebraic Details

We may encrypt by inverting the sorting procedure described in the previous section. Formally, for any j, define $R_i^{(j)}$ to be the product of independent and uniformly random powers of the 2^{n-1} distinct transpositions of the form $(k, k \oplus 2^i)$, with $0 \le k \le 2^n - 1$. Then let

$$P_i = R_0^{(i)} R_1^{(i)} \cdots R_{i-1}^{(i)},$$

and as before $X = P_n \cdots P_2 P_1$. Each random involution $R_i^{(j)}$ corresponds to a "round" as shown in Fig. 1 below. Note that while there is repetition (e.g. $R_i^{(j_1)}$ and $R_i^{(j_2)}$ are i.i.d. random variables), X is *not* a traditional iterated cryptosystems because the specific sequence of rounds is carefully chosen.

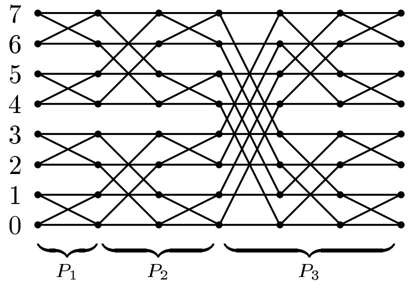

Figure 1: The structure of the cipher $X = P_3 P_2 P_1$ for $n = 3$. The rounds are applied left to right, and each round corresponds to a random involution shown as a vertical column of butterflies. Each butterfly in the diagram represents a random transposition of the form $(k, k \oplus 2^i)^b$, where parallel lines indicate $b = 0$ and a crossover indicates $b = 1$.

4.2 Algorithm Details

It is clear that we may recursively affect the actions of X and X^{-1} on any block, if we can carry out the rounds $R_i^{(j)}$ in the correct order and in such a way that the powers of all relevant transpositions are independent and equiprobable. Moreover, if we encounter the the same butterfly in two different executions, we must be able to reproduce the same random power of the corresponding transposition. This is easily accomplished if we consider the random bits to be indexed by $\mathscr{M} \times \mathbb{Z}$. The resulting function $f : \mathscr{M} \times \mathbb{Z} \longrightarrow \{0, 1\}$ is easily transformed into a random oracle function $F : \{0, 1\}^* \longrightarrow \{0, 1\}$ appropriate for Def. 1. We employ the convention that the power of any transposition $(k, k \oplus 2^i)$ is $f(m, r)$, where $m = \min\{k, k \oplus 2^i\}$ and r is the round. In other words, f is applied to the lower left hand corner of every butterfly in Fig. 1.

The next algorithm implements encryption. To encrypt a single plaintext block, the computational complexity will be $n(n+1)/2$ or $O\left(\frac{1}{2}n^2\right)$. For a block size of $n = 128$, this yields about $8,256$ operations.

Algorithm 1 *Defines recursive encryption functions* Enc *and* invSort*. The action of* $X = P_n \cdots P_2 P_1$ *on* $p \in \mathscr{M}$ *is affected by* $(q,r) = \mathsf{Enc}(p,n,1)$*, such that* $q = Xp$*. The action of* P_i *on* $p \in \mathscr{M}$ *is affected by* $(q,r) = \mathsf{invSort}(p, i-1, *)$*, such that* $q = P_i p$*.*

```
function Enc($p,i,r$):
    if $i > 1$ then
        $(q,r)$ = Enc($p,i-1,r$).
    endif
    return invSort($q,i-1,r$).
```

```
function invSort($p,j,r$):
    $q = p \oplus 2^j$.
    if $p < q$ then
        $b = f(p,r)$.
    else
        $b = f(q,r)$.
    endif
    if $b = 0$ then
        $q = p$.
    endif
    if $j > 0$ then
        return invSort($q,j-1,r+1$).
    else
        return $(q,r+1)$.
    endif
```

The decryption algorithm is easily obtained by performing the the transpositions in the reverse order. The necessary modifications are immediate, and we shall call the “reverse” of inverse-sorting fwdSort.

Algorithm 2 *Defines recursive decryption functions* Dec *and* fwdSort*. The action of* $X^{-1} = P_1^{-1} P_2^{-1} \cdots P_n^{-1}$ *on* $p \in \mathscr{M}$ *is affected by* $(q,r) = \mathsf{Dec}(p, n, \frac{1}{2}n(n+1))$*, such that* $q = X^{-1}p$*. The action of* P_i^{-1} *on* $p \in \mathscr{M}$ *is affected by* $(q,r) = \mathsf{fwdSort}(p, i-1, *)$*, such that* $q = P_i^{-1}p$*.*

```
function Dec($p,i,r$):
    $(q,r)$ = fwdSort($q,i-1,r$).
    if $i > 1$ then
        return Dec($q,i-1,r$).
    else
        return $(q,r)$.
    endif
```

```
function fwdSort($p,j,r$):
    if $j > 0$ then
        $(p,r)$ = fwdSort($p,j-1,r$).
    endif
    $q = p \oplus 2^j$.
    if $p < q$ then
        $b = f(p,r)$.
    else
        $b = f(q,r)$.
    endif
    if $b = 0$ then
        return $(p,r-1)$.
    else
        return $(q,r-1)$.
    endif
```

Remark 1 *Notice how the round information is explicitly carried by input/output argument* r *through the entire recursion processed. During the execution of* Enc*, it is incremented, while during the execution of* Dec *it is decremented. This is necessary because encryption and decryption must agree on the random bits* $f(p,r)$ *which determine the appropriate powers of the various transpositions involved.* □

4.3 The Proof of Lemma 3

To prove Lemma 3 we must first develop some terminology and prove some preliminary results. Recall that the integers in $\mathscr{M}$ will have a dual role as n-bit strings. When treating prefixes and other substrings it is useful to have a *padding function* $\pi_i : \mathbb{Z} \longrightarrow \{0,1\}^i$ taking j to the binary representation of $j \bmod 2^i$ padded up to i bits. Also define a *prefix truncation function* $\tau_i : \{0,1\}^* \longrightarrow \{0,1\}^i$ taking binary word w to its first i bits (the most significant i bits).

It is natural for us to recursively partition $\mathscr{M}$ into disjoint subsets which share the same prefix. For example, let $S_0 = \{i \in \mathscr{M} \mid \tau_1(i) = 0\}$ and $S_1 = \{i \in \mathscr{M} \mid \tau_1(i) = 1\}$, so that $\mathscr{M}$ is the disjoint union $S_0 \cup S_1$. More generally, let $S_{\pi_i(j)} = \{k \in \mathscr{M} \mid \tau_i(k) = \pi_i(j)\}$ with $1 \leq j \leq 2^i - 1$, and again we partition $\mathscr{M}$ into disjoint subsets

$$\mathscr{M} = \bigcup_{j=0}^{2^i-1} S_{\pi_i(j)}.$$

The prefix stabilizers are naturally expressed in terms of these subsets, for example clearly $K_{n-1} = \mathrm{Stab}_{K_n}\{S_0\} \cap \mathrm{Stab}_{K_n}\{S_1\}$, and more generally

$$K_{n-i} = \bigcap_{j=0}^{2^i-1} \mathrm{Stab}_{K_n}\{S_{\pi_i(j)}\}.$$

The following proposition characterizes the left cosets of $K_{n-1} \leq K_n$.

Proposition 1 *A left coset of K_{n-1} in K_n is completely determined by the image of S_0 under the action of any left coset representative.*

Proof: First of all $K_{n-1} = \mathrm{Stab}_{K_n}\{S_0\} \cap \mathrm{Stab}_{K_n}\{S_1\} = \mathrm{Stab}_{K_n}\{S_0\}$, because anything which fixes S_0 must also fix S_1. Now $K_n = \mathfrak{S}_{\mathscr{M}}$ acts transitively on the set $\mathscr{M}^{\{2^{n-1}\}}$ of all subsets of $\mathscr{M}$ of half its size. By standard group action arguments [17, 7], the left cosets $\{gK_{n-1}\}$ are in one-to-one correspondence with the images $\{gS_0\}$, in a well-defined way. □

We shall derive presently a similar characterization of the left cosets of K_{n-i-1} in K_{n-i}. First let's agree that whenever $A \subset B$ we will consider $\mathfrak{S}_A$ to be a subgroup of $\mathfrak{S}_B$. Recall [17] that if a group G factors into product $G = HK$ of normal subgroups H and K, with $H \cap K = \{1\}$, then G is a *direct product* of H and K (it is literally isomorphic to the Cartesian product with the obvious group law). Clearly whenever B is a disjoint union of A_1 and A_2, $\mathfrak{S}_B$ contains the direct product $\mathfrak{S}_{A_1}\mathfrak{S}_{A_2}$. Visibly, $K_{n-1} = \mathfrak{S}_{S_0}\mathfrak{S}_{S_1}$, and if we write $\mathfrak{S}_{\pi_i(j)} = \mathfrak{S}_{S_{\pi_i(j)}}$, we also have that K_{n-i} is the direct product

$$K_{n-i} = \prod_{j=0}^{2^i-1} \mathfrak{S}_{\pi_i(j)}.$$

Proposition 2 *A left coset of K_{n-i-1} in K_{n-i} is completely determined by the images of $S_{\pi_i(j)0}$, $0 \leq j \leq 2^i - 1$, under the action of any left coset representative.*

Proof: Because K_{n-i} is the direct product given above, a left coset gK_{n-i-1} factors into a product of left cosets

$$\prod_{j=0}^{2^i-1} g_j \left(\mathrm{Stab}_{\mathfrak{S}_{\pi_i(j)}}\{S_{\pi_i(j)0}\} \cap \mathrm{Stab}_{\mathfrak{S}_{\pi_i(j)}}\{S_{\pi_i(j)1}\}\right).$$

However, we again have $\mathrm{Stab}_{\mathfrak{S}_{\pi_i(j)}}\{S_{\pi_i(j)0}\} \cap \mathrm{Stab}_{\mathfrak{S}_{\pi_i(j)}}\{S_{\pi_i(j)1}\} = \mathrm{Stab}_{\mathfrak{S}_{\pi_i(j)}}\{S_{\pi_i(j)0}\}$. Finally, 2^i invocations of Prop. 1 obtains the desired result. □

With this machinery in place, we may now prove Lemma 3.

Proof of Lemma 3: Recall that in order to facilitate the induction argument of Sect. 3.2, thereby establishing that $\mathrm{supp}(X) = \mathfrak{S}_{\mathscr{M}}$, we must show that (for each i) $\mathrm{supp}(P_{n-i})$ contains a representative of each left coset of K_{n-i-1} in K_{n-i}.

What we'll actually show, by an inner induction argument, is that for every subset $S \subset \mathscr{M}$ contiguous on each $S_{\pi_i(j)}$ $(0 \leq j \leq 2^i - 1)$ and every possible image T of S under the action of K_{n-i} (i.e., every T of the form gS for some $g \in K_{n-i}$), $\mathrm{supp}(P_{n-i})$ contains a permutation g taking $S \mapsto T$. Since each $S_{\pi_i(j)0}$ is trivially a contiguous subset of $S_{\pi_i(j)}$, we have the desired result by Prop. 2. Note also that if we can take an arbitrary contiguous set to an arbitrary image, then we can also take an arbitrary complement of a contiguous set to an arbitrary image.

Induction Base: Clearly K_1 is isomorphic to the direct product of 2^{n-1} symmetric groups on 2 elements (cyclic groups of order 2), and thus has size $|K_1| = 2^{2^{n-1}}$. Since $|\mathrm{supp}(P_1)| = 2^{2^{n-1}}$ also, the induction hypothesis holds trivially.

Induction Step: Without loss of generality, we consider the case $i = 0$. By hypothesis, $\mathrm{supp}(P_{n-1})$ contains an element of K_{n-1} taking any contiguous subset of S_0 to a desired image ($\subset S_0$, and of the same size), while simultaneously taking any contiguous subset of S_1 to a desired image (again $\subset S_1$, and of the same size). Choose arbitrary sets $U \subset S_0, V \subset S_1$, let $T = U \cup V$, and choose any contiguous set $S \subset \mathscr{M}$ of size $|T|$. Again without loss of generality, we may assume that $|S \cap S_0| \geq |U|$ (because otherwise $|S \cap S_1| \geq |V|$ and a completely symmetric argument applies). We must show that $\mathrm{supp}(P_n) = \mathrm{supp}(P_{n-1})\mathrm{supp}(R^{(n)}_{n-1})$ contains a g such that $gS = T$. Write $g = hk\alpha$, with $h \in \mathfrak{S}_{S_0}, k \in \mathfrak{S}_{S_1}$, and where α is some product of transpositions of the form $(j, j \oplus 2^{n-1})$. Evidently the real job of α is to send elements of $S \cap S_0$ in excess of $|U|$ across the most significant bit boundary into S_1, because $h, k \in \mathrm{Stab}_{K_n}\{S_0\}$ cannot do this later on. The transpositions in $\mathrm{supp}(R^{(n)}_{n-1})$, which flip the most significant bit, are perfect for this task. Let α be the product of the transpositions $(j, j \oplus 2^{n-1})$, with $j \in J$, where J consists of the highest $|S \cap S_0| - |U|$ elements of $S \cap S_0$. We claim that $(\alpha S) \cap S_0$ is a contiguous subset of S_0, and that $(\alpha S) \cap S_1$ is either a contiguous subset or the complement of a contiguous subset of S_1. Assuming that is true, then by the induction hypothesis, we may choose h taking $(\alpha S) \cap S_0 \mapsto U$ and k taking $(\alpha S) \cap S_1 \mapsto V$, so that $gS = hk(\alpha S) = T$.

Two cases naturally arise. (*Case 1:*) If S doesn't intersect with S_1 then α takes J contiguously to some image in the middle of S_1, and α leaves $S - J$ contiguously in the middle of S_0. (*Case 2:*) On the other hand, if S intersects non-trivially with S_1, then because S is contiguous, J is precisely the highest $|S \cap S_0| - |U|$ elements of S_0 itself, and furthermore $S \cap S_1$ consists of the lowest $|S \cap S_1|$ elements of S_1 which are left fixed by α. Therefore $(\alpha S) \cap S_1$ consists of the complement of a contiguous set (those elements between $S \cap S_1$ and αJ). But again α leaves $(S \cap S_0) - J$ contiguously in the middle of S_0. This completes the induction step for $i = 0$.

Applying this same argument within the appropriate direct product subgroups when $i > 0$ yields the inner induction step and thus completes the proof. □

Remark 2 *The previous proof seems harrowing with 2 cases nested inside 2 w.l.o.g.'s nested inside of 2 layers of induction. But, it is in essence just a rigorous form of the more intuitive sorting example given in the previous section (which may have seemed simpler at first glance).* □

4.4 Sketch of the Proof of Lemma 2

In this section, we shall sketch the proof of Lemma 2 with the help of some preliminary results.

The following inequality from [15] is essentially the translation into guesswork terminology of a nice result attributed to Day [5] about majorization for sums of real vectors (see [11] and [15]). Given any m vectors $x^{(1)}, \ldots, x^{(m)} \in \mathbb{R}^n_+$, m doubly stochastic $n \times n$ matrices $D_1, \ldots, D_m$, and m positive real numbers $\omega_1, \ldots, \omega_m$, we have

$$W\left(\sum_{i=1}^{m} \omega_i D_i x^{(i)}\right) \geq \sum_{i=1}^{m} \omega_i W(x^{(i)}). \tag{7}$$

Our technique for quantifying and comparing various performance values of the optimal chosen plaintext attack typically starts with a fixed ℓ and fixed $p \in \mathscr{M}^{(\ell)}$. We then proceed to study how the cipher's structure affects the expression of equation (3). A simple but useful observation is that the conditional guesswork $W(Y|c,p)$ is completely determined by the distribution of Y on some coset of the stabilizer $H = \mathrm{Stab}_G(p)$. Let $k = [G:H]$ and fix a set $\{g_i\}_{i=1}^{k}$

of left coset representatives of H in G. It is useful to treat the distribution $y(g) = \mathsf{P}[Y = g]$ as giant vector in $\mathbb{R}G$ (the real vector space spanned by G) which decomposes into to a direct sum of left coset component vectors (i.e. one component vector for each subspace of $\mathbb{R}G$ spanned by coset g_iH). For our purposes, a mathematically convenient way to deal with y (especially when handling the complicated bookkeeping involved with product ciphers) is to exploit the fact that each coset subspace $\mathbb{R}g_iH$ is isomorphic (as a vector space) to $\mathbb{R}H$. The resulting isomorphism naturally takes the form a tensor product (of $\mathbb{R}H$-modules) known as the *induced representation from H to G by $\mathbb{R}H$*:

$$\mathbb{R}G \cong \mathbb{R}G \otimes_{\mathbb{R}H} \mathbb{R}H = \bigoplus_{i=1}^{[G:H]} g_i \otimes \mathbb{R}H, \tag{8}$$

where the isomorphism takes $g_ih \mapsto g_i \otimes h$, and

$$\sum_{g\in G} y(g)g \mapsto \widehat{y} = \sum_{i=1}^{k} g_i \otimes y^{(i)}. \tag{9}$$

Note that $\{y^{(i)}\}_{i=1}^{k}$ are just the component distribution vectors of $y(g)$ on the cosets H, only in this form they are represented as vectors in $\mathbb{R}H$. It follows directly from the definition that

$$W(Y|C^\ell, p) = \sum_{i=1}^{k} W(y^{(i)}).$$

Now recall that the product $Z = XY$ of two independent G-ciphers X and Y has distribution $\mathsf{P}[XY = g] = x * y(g)$, which from equation (1) has the form of a matrix multiplication. Indeed using the direct sum decomposition of the induced representation given in equation (8), we shall derive the block structure of this matrix. Using this structure we shall compare the distribution within the appropriate cosets of H for XY vs. Y. The key is to represent Y by $\widehat{y}$ as in equation (9), but to leave X as a convex sum of the permutations in G weighted by $x(g) = \mathsf{P}[X = g]$. We aim to derive the form of Z represented by $\widehat{z} \in \mathbb{R}G \otimes_{\mathbb{R}H} \mathbb{R}H$, again as in equation (9).

Now any $g \in G$ acts by left multiplication on any $g_j \otimes v \in \mathbb{R}G \otimes_{\mathbb{R}H} \mathbb{R}H$ according to $g(g_j \otimes v) = g_i \otimes hv$, where $gg_jH = g_iH$, so that $h \in H$ is uniquely determined by $gg_j = g_ih$. Thus we have that

$$\widehat{z} \;=\; \sum_{i=1}^{k} g_i \otimes z^{(i)} \;=\; \left(\sum_{g\in G} x(g)g\right)\left(\sum_{j=1}^{k} g_j \otimes y^{(j)}\right) \;=\; \sum_{j=1}^{k}\left(\sum_{g\in G} x(g)g\right)(g_j \otimes y^{(j)}).$$

For any particular i we may collect together contributions to direct summand $g_i \otimes \mathbb{R}H$,

$$\begin{aligned} g_i \otimes z^{(i)} &= \sum_{j=1}^{k} \sum_{g\, g_j\, H = g_i\, H} x(g)\, g\, (g_j \otimes y^{(j)}) \\ &= \sum_{j=1}^{k} \sum_{g\in\Gamma_{ij}} x(g)(g_i \otimes h_{ij}(g) y^{(j)}) \\ &= g_i \otimes \left(\sum_{j=1}^{k}\left(\sum_{g\in\Gamma_{ij}} x(g) h_{ij}(g)\right) y^{(j)}\right), \end{aligned}$$

where $\Gamma_{ij} = \{g \in G \,|\, g\, g_jH = g_iH\}$ and $h_{ij}(g) = g_i^{-1}gg_j$. Thus,

$$z^{(i)} = \sum_{j=1}^{k} \omega_{ij} D_{ij} y^{(j)}, \tag{10}$$

where

$$D_{ij} = \sum_{g\in\Gamma_{ij}} \frac{x(g)}{x(\Gamma_{ij})} h_{ij}(g), \tag{11}$$

and where $\omega_{ij} = x(\Gamma_{ij})$. Notice that the sum in equation (11) is a convex sum of permutations in H, hence each D_{ij} takes the form of a doubly stochastic matrix under a suitable ordering of H (the basis vectors of $\mathbb{R}H$). Furthermore, it can easily be shown that the values of ω_{ij} are the elements of a doubly stochastic matrix [15]. Also note that $\Gamma_{ii} = H^{g_i} \leq G$ for each i. The true core of Lemma 2 is the following proposition.

Proposition 3 *For a permutation group $G \leq \mathfrak{S}_{\mathscr{M}}$, let X and Y be independent G-ciphers such that* $\operatorname{supp}(X) = G$. *For any $p \in \mathscr{M}^{(\ell)}$ such that Y is non-uniform on at least one left coset of* $\operatorname{Stab}_G(p)$, *we have $W(XY|C^\ell, p) > W(Y|C^\ell, p)$.*

Proof: Let p satisfy the assumption of the proposition, and let $\hat{z}$ represent the distribution of the product $Z = XY$ as above. Let $y^{(j)}$ be non-uniform and consider $D_{jj}y^{(j)}$. That is to say, let us focus on this one submatrix block on the diagonal of the larger doubly stochastic matrix representing the convolution $z = x * y$.

Since $\Gamma_{jj} = H^{g_j}$, we may rewrite D_{jj} as

$$D_{jj} \quad = \quad \sum_{g\in\Gamma_{jj}} \frac{x(g)}{x(\Gamma_{jj})} g^{g_j^{-1}} \quad = \quad \sum_{h\in H} \frac{x(h^{g_j})}{x(H^{g_j})} h.$$

Thus we see that by scaling appropriately, $D_{jj}y^{(j)}$ has the form of a product of two independent H-ciphers $\widetilde{X}\widetilde{Y}$, with $\mathsf{P}\left[\widetilde{X} = h\right] = \mathsf{P}\left[X = h^{g_j}\right]/x(H^{g_j})$, and $\mathsf{P}\left[\widetilde{Y} = h\right] = \mathsf{P}\left[Y = g_j h\right]/y(g_j H)$. But since $\operatorname{supp}(X) = G$ and conjugation by g_j yields an isomorphism of $H \longleftrightarrow \Gamma_{jj}$, $\operatorname{supp}(\widetilde{X})$ is not confined to any proper coset of H, and we may invoke Lemma 1 to obtain $W(\widetilde{X}\widetilde{Y}) > W(\widetilde{Y})$ or more importantly for our purposes, $W(D_{jj}y^{(j)}) > W(y^{(j)})$.

Note that we may bound any $W(z^{(i)})$ by the inequality of equation (7) as

$$W(z^{(i)}) \quad = \quad W\left(\sum_{m=1}^{k} \omega_{im} D_{im} y^{(m)}\right) \quad \geq \quad \sum_{m=1}^{k} \omega_{im} W(y^{(m)}),$$

but by using $W(D_{jj}y^{(j)}) > W(y^{(j)})$ and equation (7) again, we may strictly bound $W(z^{(j)})$ as follows

$$\begin{aligned} W(z^{(j)}) &= W\left(\sum_{m=1}^{k} \omega_{jm} D_{jm} y^{(m)}\right) \\ &\geq \sum_{m=1}^{k} \omega_{jm} W(D_{jm} y^{(m)}) \\ &> \sum_{m=1}^{k} \omega_{jm} W(y^{(m)}). \end{aligned}$$

(Note that in both cases we have used, for doubly stochastic D, the inequality $W(Dv) \geq W(v)$ which follows from simple majorization arguments [15]). Combining these bounds on $W(z^{(i)})$ we obtain a strict bound on $W(Z|C^\ell, p)$ as follows

$$\begin{aligned} W(Z|C^\ell, p) \quad = \quad \sum_{i=1}^{k} W(z^{(i)}) &> \sum_{i=1}^{k}\sum_{m=1}^{k} \omega_{im} W(y^{(m)}) \\ &= \sum_{m=1}^{k} W(y^{(m)}) \sum_{i=1}^{k} \omega_{im} \\ &= \sum_{m=1}^{k} W(y^{(m)}) \quad = \quad W(Y|C^\ell, p), \end{aligned}$$

which was to be proved. ☐

Remark 3 *Evidently, in the previous proposition, we could weaken the condition* $\mathrm{supp}(X) = G$ *to: For every* $p \in \mathscr{M}^{(\ell)}$, $\mathrm{supp}(X) \cap \mathrm{Stab}_G(p)$ *is not confined to a proper coset of* $\mathrm{Stab}_G(p)$. *However, for our purposes in this paper, it was not necessary to use the weaker condition.* ☐

The next proposition provides an important interpretation of the situation when a cipher is uniform on every coset of an ℓ-message stabilizer.

Proposition 4 *Let Y be a G-cipher, for a permutation group $G \leq \mathfrak{S}_{\mathscr{M}}$. For any $p \in \mathscr{M}^{(\ell)}$, write $H = \mathrm{Stab}_G(p)$ and we have*

$$W(Y|C^\ell, p) \leq \frac{1+|H|}{2},$$

with equality holding iff Y is uniform on each coset of H.

Proof: For $c \in \mathscr{M}^{(\ell)}$ with $\omega(c|p) \neq 0$,

$$1 \leq W(Y|c,p) \leq \frac{1+|H|}{2},$$

because $W(Y|c,p)$ is the guesswork on a coset of size $|H|$. Furthermore, equality in the upper bound is achieved iff Y has constant probability on that particular coset [15]. Now since $\sum_{c \in \mathscr{M}^{(\ell)}} \omega(c|p) = 1$, the sum from equation (3)

$$W(Y|C^\ell, p) = \sum_{c \in \mathscr{M}^{(\ell)}} W(Y|c,p)\omega(c|p)$$

is convex and therefore achieves its maximum of $\frac{1}{2}(1+|H|)$ iff Y is constant on each coset of H (Y will of course have the constant probability 0 on those cosets corresponding to $\omega(c|p) = 0$). ☐

By tying together the previous two propositions, we may finally prove Lemma 2.

Proof of Lemma 2: Again, let us write $Z = XY$. Suppose there is a $p \in \mathscr{M}^{(\ell)}$ such that $\theta_\ell(Z) = W(Z|C^\ell, p)$ and Y is non-uniform on at least one coset of $\mathrm{Stab}_G(p)$. Then we may invoke Prop. 3 to obtain

$$\theta_\ell(Z) = W(Z|C^\ell, p) > W(Y|C^\ell, p) \geq \theta_\ell(Y).$$

On the other hand, suppose that for every $p \in \mathscr{M}^{(\ell)}$ satisfying $\theta_\ell(Z) = W(Z|C^\ell, p)$, Y is uniform on each coset of $\mathrm{Stab}_G(p)$. Let $H = \mathrm{Stab}_G(p)$ for any such p. By equation (10), Z is uniform on each coset of H as well, and by Prop. 4,

$$\theta_\ell(Z) = W(Z|C^\ell, p) = \frac{1+|H|}{2}$$

Now choose any $\widehat{p}$ with $|\mathrm{Stab}_G(\widehat{p})| = M_G(\ell)$ and hence

$$\theta_\ell(Z) = \frac{1+|H|}{2} \leq W(Z|C^\ell, \widehat{p}) \leq \frac{1+M_G(\ell)}{2},$$

forcing $|H| = M_G(\ell)$, and thus $\theta_\ell(Z) = \theta_\ell(U_G)$. Then, either $\theta_\ell(Y) \neq \theta_\ell(U_G)$, in which case $\theta_\ell(Z) > \theta_\ell(Y)$, or $\theta_\ell(Y) = \theta_\ell(U_G)$.

To summarize what we have proved thus far, $\theta_\ell(Z) \geq \theta_\ell(Y)$ and if equality holds then $\theta_\ell(Y) = \theta_\ell(U_G)$. However conversely, if $\theta_\ell(Y) = \theta_\ell(U_G)$, then $\theta_\ell(U_G) \geq \theta_\ell(Z) \geq \theta_\ell(Y) = \theta_\ell(U_G)$, forcing equality $\theta_\ell(Z) = \theta_\ell(Y)$, which completes the proof. ☐

5 Conclusion

The issue of security amplification by product composition remains a complex one. In this paper, we have added to the number of situations where a definite answer can be given. Specifically, Thm. 1 asserts that there exists efficient cipher X such that the security of XY is strictly greater than Y unless Y is perfect. There is room for further improvement in this result. For example, a more efficient cipher might be constructed which makes use of a weakened form of Lemma 2 as discussed in Remark 3. Additionally, our implementation might be optimized for bulk encryption.

The cipher we construct to prove Thm. 1 is costly in some ways but has other desirable properties. Unlike a one-time pad, if the key were replaced by a pseudo-random source, a known plaintext-ciphertext block would *not* trivially betray the key used for that block. This property could be useful in constructing provably secure practical encryption systems. Also observe that our construction is *not* an iterated cryptosystem but rather a product of independent rounds with a *carefully chosen order*. The techniques employed here might be a useful new paradigm for practical cryptosystems with key schedules instead of a truly random source of key material.

Acknowledgments. I would like to thank Serge Vaudenay for his many insightful comments, and in particular for suggesting the formal computational model of Sect. 2.

References

[1] W. Aiello, M. Bellare, G. Di Crescenzo, and R Venkatesan. Security amplification by composition: The case of doubly-iterated, ideal ciphers. In H. Krawczyk, editor, *Advances in Cryptology - CRYPTO '98*, Berlin, 1998. Springer-Verlag.

[2] David J. Aldous and Persi Diaconis. Shuffling cards and stopping times. *Amer. Math. Monthly*, 93:333–348, 1986.

[3] Christian Cachin. *Entropy Measures and Unconditional Security in Cryptography.* PhD thesis, ETH Zürich, 1997.

[4] Keith W. Campbell and Michael J. Wiener. DES is not a group. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92*, pages 512–517, Berlin, 1992. Springer-Verlag.

[5] P. W. Day. Rearrangement inequalities. *Canad. J. Math.*, 24:930–943, 1972.

[6] Persi Diaconis. *Group Representations in Probability and Statistics.* Institute of Mathematical Statistics, Hayward, CA, 1988.

[7] John D. Dixon and Brian Mortimer. *Permutation Groups.* Springer-Verlag, New York, 1996.

[8] S. Even and O. Goldreich. On the power of cascade ciphers. *ACM Transactions on Computer Systems*, 3(2), 1985.

[9] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, New York, 2nd edition, 1979.

[10] G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes.* Oxford University Press, Oxford, 2nd edition, 1992.

[11] Albert W. Marshall and Ingram Olkin. *Inequalities: Theory of Majorization and Its Applications.* Academic Press, San Diego, 1979.

[12] Ueli M. Maurer and James L. Massey. Cascade ciphers: The importance of being first. *Journal of Cryptology*, 6:55–61, 1993.

[13] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, Boca Raton, 1997.

[14] Moni Naor and Omer Reingold. On the construction of pseudorandom permutations: Luby-Rackoff revisited. *Journal of Cryptology*, 12:29–66, 1999.

[15] John O. Pliam. *Ciphers and their Products: Group Theory in Private Key Cryptography.* PhD thesis, University of Minnesota, July 1999.

[16] John O. Pliam. Guesswork and variation distance as measures of cipher security. In *Selected Areas in Cryptography - SAC'99*, LNCS 1758, pages 62–77, Berlin, 2000. Springer-Verlag.

[17] Joseph J. Rotman. *An Introduction to the Theory of Groups.* Wm. C. Brown, Dubuque, IA, 3rd edition, 1988.

[18] Claude E. Shannon. Communication theory of secrecy systems. *Bell System Tech. Jour.*, 28:656–715, 1949.