A General Analysis of the Security of Elastic Block Ciphers

Debra L. Cook and Moti Yung and Angelos Keromytis
Department of Computer Science, Columbia University
{*dcook,moti,angelos*}*@cs.columbia.edu*

September 28, 2005

Abstract

We analyze the security of elastic block ciphers in general to show that an attack on an elastic version of block cipher implies a polynomial time related attack on the fixed-length version of the block cipher. We relate the security of the elastic version of a block cipher to the fixed-length version by forming a reduction between the versions. Our method is independent of the specific block cipher used. The results imply that if the fixed-length version of a block cipher is secure against attacks which attempt key recovery then the elastic version is also secure against such attacks.

Keywords: Elastic Block Cipher, Security Proofs.

1 Introduction

A method for transforming existing block ciphers into variable length block ciphers was defined in [1]. The variable length version of a block cipher is referred to an elastic block cipher. We analyze the security of elastic block ciphers in general by relating the security of the elastic version of a block cipher against key recovery attacks to that of the original fixed-length version of the block cipher. Our analysis is independent of the specific block cipher used. By forming a reduction from the elastic version to the fixed-length version, we are able to show that an attack which recovers the expanded key of the elastic version can also be used to recover the expanded key of the fixed-length version. As a result, if the fixed-length version of a block cipher is secure against key recovery attacks (including linear, differential, impossible differentials, higher order differential and square attacks, among others, which all attempt key recovery), then the elastic version is also secure against such attacks.

The remainder of the paper is organized as follows. In Section 2 we review the elastic block cipher construction. In Section 3 we show how any attack on an elastic block cipher which recovers the key can be converted into an attack on the original cipher. In Section 4 we state the implications of our results.

2 Background

The purpose of an elastic block cipher is to create a variable length block cipher from an existing block cipher. We briefly review the elastic block cipher construction presented in [1]. We include a full description of the key schedule properties because certain properties of the key schedule are necessary for our analysis.

Given an block cipher, G, that is structured as a series of r rounds and processes b bit blocks, a variable length block cipher, G', will be created that can process block sizes of $b + y$ bits where $0 \leq y \leq b$. The

number of rounds, r', in G' will be $r + \lceil yr/b \rceil$. We note that if G is a Feistel network, the round function of G will be viewed as consisting of one cycle of the Feistel network as opposed to just the function used within the Feistel network.

The data block processed by G' is treated as a left b bit block which is processed by the round function and a right y bit block which is omitted from the round function. Between rounds, the y bit block is XORed into the b bit block and y bits from the b bit block become the next y bit block. This is similar to an (unbalanced) Feistel network. The main difference is that in an elastic block cipher the swapping of bits between rounds involves bits from the outputs of the last two rounds; whereas in a Feistel network, the output of the last round and the input from the next to last round are XORed.

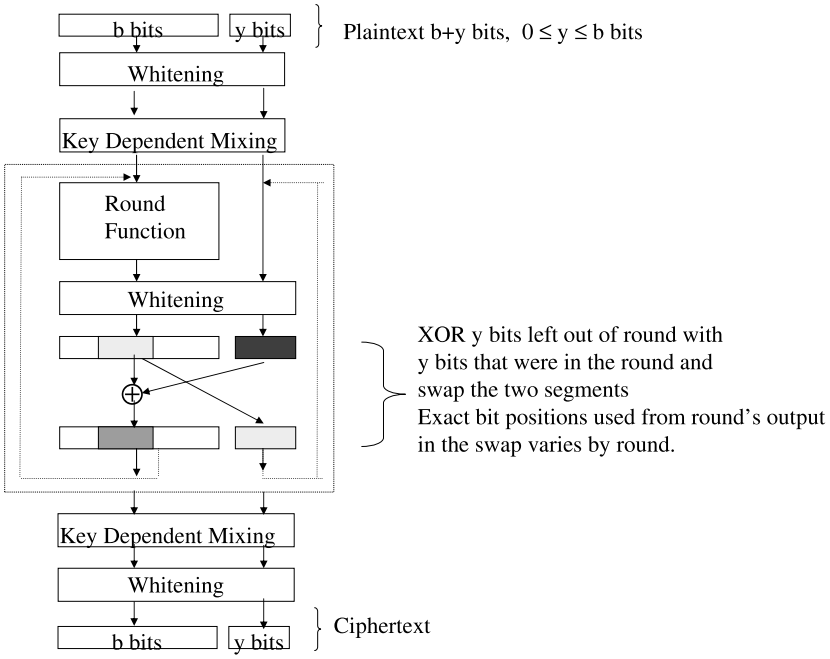


Figure 1: **Elastic Block Cipher Structure**

The general structure of G' is shown in Figure 1. G' consists of the following:

- An initial and final key dependent mixing step that permutes the $b + y$ bits.
- Initial whitening and end of round whitening.
- Between rounds, there is a swapping of bits from the left b bit block and the right y bit block in order to alternate which y bits are left out of the round function. This step is defined as follows:
  1. Let Y denote the y bits that were left out of the round function.
  2. Let X denote some subset of y bits from the round function's output of b bits. A different set of X bits (in terms of position) is selected in each round. How to best select X is dependent on the specific block cipher.
  3. Set $Y \leftarrow X \oplus Y$.

4. Swap X and Y to form the input to the next round.

The decryption function for G' consists of the network applied in reverse and the round function replaced by its inverse. If G is a Feistel network, the inverse of the round function is a cycle of G run in reverse.

In regards to the key schedule, G' requires more expanded key bits then G due to the whitening steps, the key dependent mixing steps and the increased number of rounds. Even if G includes an initial whitening steps and end of round whitening, additional whitening key bits are needed for the extra y bits and the additional rounds. Therefore, the key schedule of G cannot simply be reused for G'. In the definition of elastic block ciphers, some assumptions are made regarding the key schedule in order to avoid a highly structured key schedule with a direct relationship between expanded key bits, which is typical of existing block ciphers. Ideally, the expanded key bits should be pseudorandom. At a minimum, the whitening bits must be independent of any key bits used internal to the round function and the key bits used in the mixing steps must be independent of all other expanded key bits. It is also required that the key schedule is independent of the $b + y$ bit data block input to G, meaning the expanded key bits are fixed for a given key and do not vary based on the input to G'. This last requirement is a typical property of existing block ciphers.

3 Security of $G\,'$

3.1 Overview

For any concrete block cipher used in practice (and not treated as a pseudorandom permutation or a member of a family of functions), the cipher cannot be proven secure in a theoretical sense but rather is proven secure against known types of attacks. Thus, we can only do the same for the elastic version of such a cipher. In order to provide a general understanding of the security of our construction, we provide a method for reducing the security of the elastic version to that of the original version, showing that a security weakness in $G\,'$ implies a weakness in G. Our security analysis of $G\,'$ exploits the fact that there is an instance of G embedded in $G\,'$.

We concentrate on key-recovery attacks. We show how to reduce $G\,'$ to G in a manner that allows an attack which finds the round keys of $G\,'$ to be used to find the round keys for G. Security against key-recovery attacks does not by itself imply security (*e.g.,* the identity function which ignores the key is insecure while key recovery is impossible). However, all concrete attacks against real ciphers (differential, linear, *etc.*) attempt key recovery and thus practical block ciphers should be secure against such attacks.

In order to focus on the core components of the algorithm for creating $G\,'$ from G, we consider G without the initial and final key-dependent mixing steps. If present, these intuitively only serve to increase the security of G' since they prevent an attacker from knowing with probability 1 which bits are omitted from the first application of the round function. Furthermore, since the mixing steps are added steps (as opposed to modifications to components of G) using key material that is independent of the round and whitening key material (by the assumption on the key schedule), they do not impact our analysis.

3.2 Round Key Recovery Attack

We use the fact that an instance of G is embedded in G' to create a reduction from G' to G. As a result of this reduction, an attack against G' that allows an attacker to determine some of the round keys implies an attack against G itself which is polynomially related in resources to the attack on G'. Assuming that G itself is resistant to such attacks, we conclude that G' does not reveal round-key bits to the attacker. The reduction

requires a set of (plaintext, ciphertext) pairs. This is not considered a limiting factor because in most types of attacks, whether they are known plaintext, chosen plaintext, adaptive plaintext, chosen ciphertext *etc.*, the attacker acquires a set of such pairs. We assume that the key schedule is independent of the plaintext and ciphertext. This assumption is true of block ciphers used in practice. We also assume G contains end of round whitening, as is the case with AES.

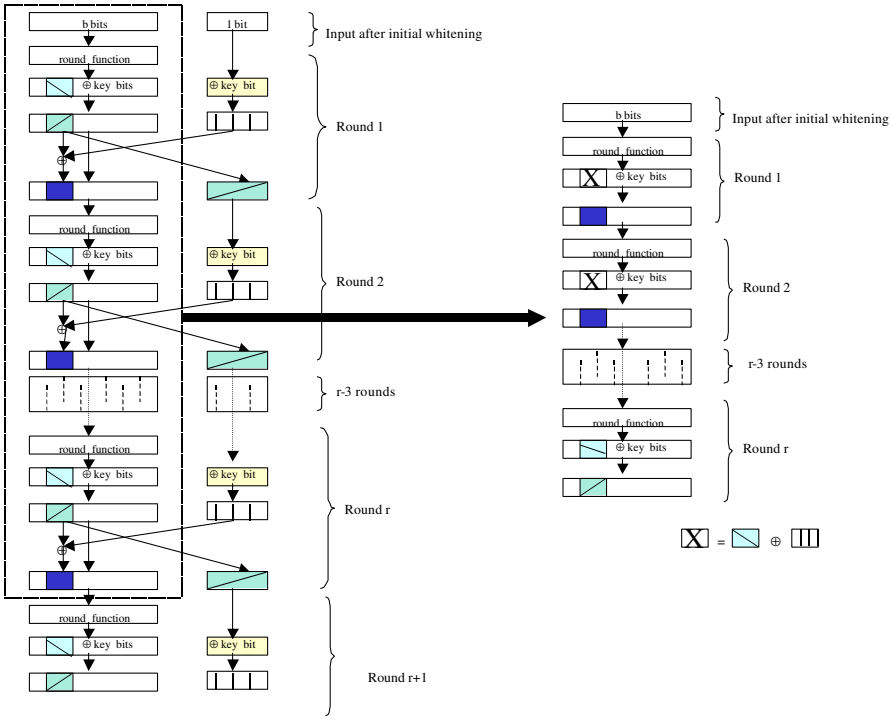


Figure 2: G **within** G'

In order to aid our analysis, we draw attention to the fact that the operations performed in G' on the leftmost b bit positions in r consecutive rounds is an application of G. This is depicted intuitively in Figure 2. This relationship can be used to convert an attack which finds the round keys for G' to an attack which finds the round keys for G. Let G_{rk} denote G using round keys rk. Specifically, if ${G_k}'(p \parallel x) = c \parallel z$, a set of round keys, rk, for G such that $G_{rk}(p) = c$ can be formed from the round keys and the round outputs in G' by collapsing the end-of-round whitening and swapping steps in G' into a whitening step. The leftmost b bits of the round key for the initial whitening are unchanged, and the rightmost y bits are dropped. The resulting whitening key bits will vary in up to y positions across the (plaintext, ciphertext) pairs due to the previous round's output impacting the end-of-round whitening step. However, it is possible to use these keys to solve for the round keys of G.

We state the following claim to assist the reader in understanding the linkage between G and G'. The claim shows that for any set of plaintext, ciphertext pairs encrypted under sets of round keys in G' where

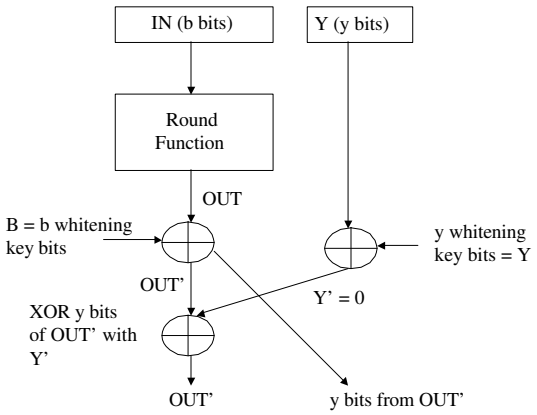
the rightmost y bits used for whitening in each round may vary amongst the sets and all other key bits are identical amongst the sets, there exists a corresponding set of plaintext, ciphertext pairs for G where the round keys used in G' for the round function and the leftmost b bits of each whitening step are the same as those used in G, the plaintexts used in G are the leftmost b bits of the plaintexts used in G', and the ciphertexts for G are the same as the leftmost b bits of output of the r^{th} round of G' prior to the swap step.

Claim I: Let $\{(pi, ci)\}$ denote a set of n plaintext, ciphertext pairs and let $|w| = |vi| = y$. If $G_k(pi) = ci$ then there exists n sets of round keys for the first r rounds of G' that are consistent with inputs $pi \parallel w$ producing $ci \parallel vi$ as the output of the r^{th} round prior to the swap at the end of the r^{th} round, for $i = 1$ to n, such that the following condition applies:

Condition I: The leftmost b bits used for whitening in each round are identical across the n sets and any bits used internal to the round function are identical across the n sets.

Furthermore, y may be any valid value. The bits in vi are not used and thus no restrictions are placed on their values.

Proof: Let $rk = \{rk_j$ for $j = 0$ to $r\}$ be the set of round keys corresponding to key k for G. rk_0 denotes the key bits used for initial whitening. For (pi, ci), form a set of the first r round keys for G' as follows: Pick a constant string, w, of y bits, such as a string of $0's$. Let $pi \parallel w$ be the input to G'. Let $rki' = \{rki'_j$ for $j = 0$ to $r\}$ denote the round keys for G' through the r^{th} round for the pair (pi, ci). Set any bits in rki'_j used internal to the round function to be the same as the corresponding bits in rk_j. Set the leftmost b bits used for whitening in rki'_j to the b bits used for whitening in rk_j. Set the rightmost y bits used for whitening in rki'_j to be the same as the y bits left out of the round function in round j of G'. This is illustrated by Figure 3. Notice that the leftmost b bits used for whitening in each round are identical across the n sets, and any bits used internal to the round function are identical across the n sets; specifically, they correspond to rk in each case, and the rightmost y bits used in each whitening step differ based on (pi, ci) across the n sets. The case in which G does not contain whitening steps corresponds to using 0's for the leftmost b bits of each whitening step in G'.



Converted round key for *G* = (B with y bits replaced by their XOR with Y') = B

Figure 3: **Converted Key Unchanged in Left b Bits**

The operations of G' on the leftmost b bits through round r, prior to the last swap, are identical to the operations in $G_k(pi)$ because the swap step in G' results in XORing y bits of a round function's output with y $0's$. Therefore, the leftmost b bits output from the r^{th} round prior to the swap in the r^{th} round is ci. Therefore, for $i = 1$ to n there exists a set of round keys, rki' for $G'_{rki'}$ such that $G'(pi)$ produces ci as the leftmost b bits in the r^{th} round prior to the swap step and Condition I holds, thus proving Claim I.

Theorem I: *If there exists an attack on G' that allows the round keys to be determined for r consecutive rounds, then there exists a polynomially time related attack on G with r rounds, assuming:*

- *G contains end-of-round whitening.*
- *No message-related round keys. Namely, if there are expanded key bits utilized in G, these expanded key bits depend only on the key and do not vary across inputs.*

The second condition is typical of existing block ciphers. With respect to the first condition placed on G in *Theorem I*, the condition may be removed if the attack on G' involves solving for the round key bits directly and allows the bits used in the whitening steps to be set to 0 for bit positions not swapped and to 0 or 1, as necessary, for bit positions swapped, to ensure the whitening on the leftmost b bits is equivalent to XORing with 0, which is the same as having no whitening in G. If the attack on G' finds all possible keys or sets of round keys, the attack must find the key(s) or set(s) of round keys corresponding to round keys that are equivalent to XORing with 0.

The method described here treats whitening key bits as if they are pseudorandom in that the whitening key bits can take on any value. If there is a relationship amongst the whitening key bits and/or between whitening key bits and key material used within the round function, such keys will be a subset of the possible keys found and can be determined by checking which of the potential keys corresponds to the key schedule.

When we refer to converting the round keys of G' into round keys for G we mean the following: In round j of G', let b_{ji} denote the i^{th} bit of the b bits output from the round function prior to the end of round whitening. Let kw_{ji} denote the end of round whitening key bit applied to b_{ji}. If b_{ji} is involved in the swap step at the end of round j, let y_{jh} denote the bit from the rightmost y bits with which b_{ji} is swapped and let kw_{jh} denote the whitening key bit applied to y_{jh}. Set the i^{th} whitening bit in round j of G to $kw_{ji} \oplus kw_{jh} \oplus y_{jh}$ if b_{ji} is involved in the swap step in G'. Set all other key bits used in G (both whitening and any internal to the round function) to be identical to the key bits used in G'. We refer to the initial whitening as round 0. The initial whitening for G' is converted to initial whitening for G by using the leftmost b expanded key bits of the initial whitening as the initial whitening in G. We describe here a method for converting the attack on G' to an attack on G. Without loss of generality, we use the first r rounds of G' as the r consecutive rounds for which the round keys are found. The attacks are presented in terms of solving for the round keys from the initial whitening to round r, but may also be performed by working from round r back to the initial whitening or by using any consecutive r rounds.

Proof:

This attack runs in quadratic time in the number of rounds of G. The attack on G' is used to solve for round keys 0 and 1 for G, then repeatedly solves for one round key of G at a time, using the output of one round of G as partial input to a reduced round version of G', running the attack on G' and converting the 1^{st} round key of G' to the round key for the next round of G. We assume that if an attack on G' with r rounds exists, then a reduced round attack on G' exists for any number of rounds $< r'$.

Given a set $\{(P^*, C^*)\} = \{(pi^*, ci^*)\}$ of n (plaintext, ciphertext) pairs for G, create a set $\{(P, C)\} = \{(pi^* \parallel 0, ci^* \parallel vi_r)\}$ of n (plaintext, ciphertext) pairs for an r round version of G'. Note: we only require

that the y bits appended to each pi^* when forming $\{(P, C)\}$ be a constant; we choose to use 0. The vi_r values appended to the $ci's$ are arbitrary and do not need to be identical. The r subscript in vi_r denotes the number of rounds. Our method runs reduced round attacks on G' and the vi_r's can vary each time. Solve G' for round keys 0 and 1. By the pseudorandomness of the round keys described in the definition of elastic block ciphers, sets of round keys exist that correspond to $\{(P, C)\}$ and which are identical in at least the first two rounds (the round keys across all n pairs may be identical in additional rounds, but we are only concerned with the first two rounds). Denote these as rk'_0 and rk'_1. Use the leftmost b bits of rk'_0 as round key 0, rk_0, for G. Since the rightmost y bits are identical across all inputs to G', when rk'_1 is converted to a round key for G, the result will be the same across all n elements of $\{(P, C)\}$. Use the converted round key as round key 1, rk_1, for G. For each pi^*, apply the initial whitening and first round of G using the two converted round keys. Let $p1i$ denote the output of the first round of G for $i = 1$ to n. Using a reduced round version of G' with $r - 1$ rounds and the initial whitening removed, set $\{(P, C)\} = \{(p1i \parallel 0, ci^* \parallel vi_{r-1})\}$ and solve for the first round key of G'. As before, convert the resulting round key(s) to a round key for G. Again, the converted round keys for G will be identical across all n values. Use the converted round key as the second round key for G. Repeat the process for the remaining rounds of G, each time using the outputs of the last round of G for which the round key has been determined as the inputs to G' and reducing the number of rounds in G' by 1, to sequentially find the round keys for G.

This attack involves applying each round of G to n inputs for a total of rn rounds of G. $\frac{n(r+1)r}{2}$ rounds of G' are computed in the worst case if $A'_{G'}$ requires knowing the output of each round of the reduced round version of G' to find the first round key. r applications of $A'_{G'}$ on G' are needed on the reduced round versions of G' when solving for the round keys of G'. Let t_A denote the time to run $A'_{G'}$. The time to attack G is $O(nr^2 + rt_A)$.

In summary, the attack on G can be written as:

Input $\{(P^*, C^*)\} = \{(pi^*, ci^*)$ for $i = 1$ to $n\}$.
Create $\{(P, C)\} = \{(pi^* \parallel 0, ci^* \parallel vi_r)$ for $i = 1$ to $n\}$ for a r round version of G',
    where the $vi's$ are arbitrary.
Using $A_{G'}$, solve a r reduced round version of G' for rk'_0 and rk'_1.
Convert rk'_0 to rk_0 and rk'_1 to rk_1.
Set $p1i =$ first round output of G using rk_0 and rk_1, for $i = 1$ to n.
For $j = 1$ to $r - 1\{$
        $\{(P, C)\} = \{(pji \parallel 0, ci^* \parallel vi_{r-j})$ for $i = 1$ to $n\}$.
        Solve a $r - j$ reduced round version of G' for the first round key, rk'_1.
        Convert rk'_1 to form rk_{j+1}.
        $p(j + 1)i =$ output of round $j + 1$ of G on pji using rk_{j+1} for $i = 1$ to n.
$\}$

Another method for proving the theorem is provided in Appendix A. It is included because it requires fewer computations than the method described here and assists in explaining how the round keys of G' can be converted to round keys for G. When given the round keys for G' which correspond to n plaintext ciphertext pairs, the method described here produces round keys for G which correspond to n plaintext ciphertext pairs. Whereas, the alternative method described in the appendix produces the round keys for G for at most n plaintext, ciphertext pairs and requires $2^{y(r-2)}n$ plaintext, ciphertext pairs to guarantee the resulting round keys will correspond to at least n plaintext, ciphertext pairs. However, since the method in Appendix A requires fewer computations, it may useful when y is small relative to b.

4 Conclusions

We have described how any key recovery attack on an elastic block cipher can be converted into a key recovery attack on the original block cipher. The implications of this conversion is that if a block cipher is secure against a specific practical attack which attempts key recovery, then so is the elastic version. The attacks covered include linear, differential, impossible differentials, higher order differentials and square attacks, among others.

References

[1] D. Cook, M. Yung, and A. Keromytis. Elastic Block Ciphers. Cryptology ePrint Archive, 2004/128, 2004. http://eprint.iacr.org/.

Appendix A

We describe here an alternative method for proving Theorem I which requires fewer computations than the method described in the paper, but provides rounds keys for a smaller set of plaintext, ciphertext pairs.

This method produces an attack on G that runs in time polynomial in the attack on G' and r. The attack works as follows: Assume there exists a known (plaintext, ciphertext) pair attack on G' which produces the round keys either by finding the original key and then expanding it, or by finding the round keys directly. Using round keys for rounds 0 to r of G', convert the round keys into round keys for G one round at a time. For each round, extract the largest set of (plaintext, ciphertext) pairs used in the attack on G' that have the same converted round key. If there are n_j (plaintext, ciphertext) involved at round j, there will be at least $\frac{n}{2^y}$ pairs remaining for which the round keys are consistent after round j. The end result is a set of round keys for G that are consistent with a set of $\frac{n}{2^{y(r-2)}}$ b-bit (plaintext, ciphertext) pairs for G. We then describe how to take a set of (plaintext, ciphertext) pairs for G, convert them into a set of (plaintext, ciphertext) pairs for G' in order to run the attack on G' to find the round keys for G. Finally, we discuss the bounds on y for which this attack is more efficient than an exhaustive key search.

Let $\{(P, C)\} = \{(pi \parallel xi, ci \parallel zi)\}$ (for $i = 1$ to n) denote a set of n known $b + y$-bit (plaintext, ciphertext) pairs for G', where $|pi| = |ci| = b$ and $|xi| = |zi| = y$.

Assume the existence of an algorithm $A_{G'}$ that finds all possible keys, $\{k_j\}$, corresponding to $\{(P, C)\}$ in time less than a exhaustive search for the key. Let m denote the number of keys found. Without loss of generality, it is assumed the keys are available in expanded form.

Let $S = \{ek_j\}$ for $j = 1$ to m be the set of expanded keys used for whitening for which ek_j is from the expansion of key k_j and $G'_{k_j}(pi \parallel xi) = ci \parallel zi$ for $i = 1$ to n.

Let R_{int} denote any key material utilized within the round function. The values found for such key bits will be the same for the solutions derived by the attack for G' and G.

Let $\{(P, U)\} = \{(pi||xi, ui||vi)\}$ such that $ui||vi$ is the output of the r^{th} round of G', where $|ui| = b$ and $|vi| = y$.

Let $S' = \{ek'_j | ek'_j =$ bits of $ek_j \in S$ corresponding to rounds 0 to r used for whitening $\}$ be the set of expanded key bits used for whitening in rounds 0 to r of G'.

For each $ek_j \in S'$ and each $(pi \parallel xi, ui \parallel vi) \in \{(P, U)\}$, convert the round keys to round keys for G. Let ek'_{ij} be the converted key corresponding to the i^{th} element of $\{(P, U)\}$ and the j^{th} element of S'. The part of ek'_{ij} corresponding to round 0 will be identical across all elements. When the round keys are converted, at most y bits change in the leftmost b bits. Thus, the resulting round keys for round q, $0 < q \leq r$

can be divided for each of the y impacted bits into those that have a 0 in the affected bit and those that have a 1 in the affected bit. For $q = 1$ to r, define S'_{rnd_q} as the maximum-sized set of $ek'_{ij}s$ from $S_{rnd_{q-1}}$ that have identical round key(s) for round q, where $S'_{rnd_0} = S'$. Let $\{(P,U)_{rnd_q}\}$ be the corresponding elements of $\{(P,U)\}$. When forming $\{(P,U)_{rnd_q}\}$, at least $2^{-y}|\{(P,U)_{rnd_{q-1}}\}|$ of the elements from $\{(P,U)_{rnd_{q-1}}\}$ are included.

To illustrate how the sets S'_{rnd_q} and $\{(P,U)_{rnd_q}\}$ are created, consider the example shown in Figure 4 where $b = 4$, $y = 2$, and the leftmost 2 bits are swapped with the y bits in the swap step. The round number is q and $\{(P,U)_{rnd_{q-1}}\}$ contains three (plaintext, ciphertext) pairs. Suppose the outputs of the round function in the q^{th} of G' are $100101, 110011$ and 111111 and the whitening bits in the q^{th} round are 011010. The converted round keys corresponding to the three cases are $0110, 1110$ and 1110. Since 1110 occurs in the majority of the cases, set the q^{th} round key of G to 1110. S'_{rnd_q} contains the round keys for rounds 0 to $q-1$ from $S'_{rnd_{q-1}}$ and 0010, and $\{(P,U)_{rnd_q}\}$ contains the second and third (plaintext, ciphertext) pairs from $\{(P,U)_{rnd_{q-1}}\}$.

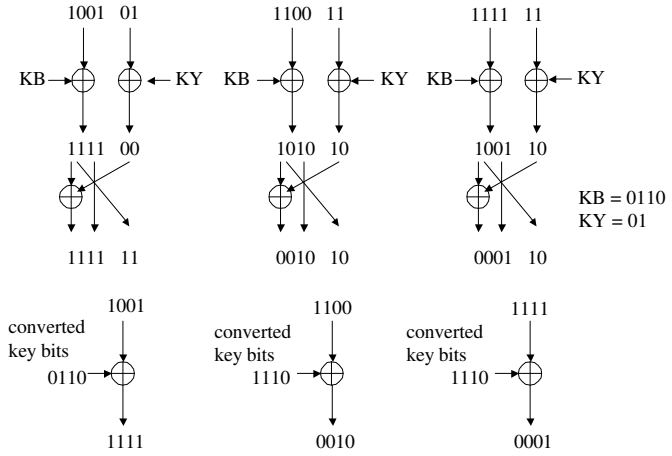


Figure 4: **Forming** S'_{rnd_q}

Let $\{(P,C)_G\} = \{(pi,ci)|(pi \parallel yi, ui \parallel vi) \in \{(P,U)_{rnd_r}\}\}$. $|\{(P,C)_G\}| \geq n/2^{yr}$. $\{(P,C)_G\}$ is a set of (plaintext, ciphertext) pairs for which $G_{rk}(pi) = ci \ \forall \ (pi,ci) \in \{(P,C)_G\}$ with the whitening round keys of $rk \in S'_{rnd_q}$ and any additional key material utilized by the rounds is the same as that for G', namely R_{int}.

Let t_r denote the time to run r rounds of G', and t_A denote the time to run $A_{G'}$. In the case of obtaining at least one set $\{(P,U)_{rnd_r}\}$ of size $\geq \frac{n}{2^{yr}}$, the time required beyond t_A consists of: nmt_r time to obtain the outputs of the first r rounds for each $\{(P,U)\}$, $O(nmr)$ time to perform the conversion of the round keys from G' to round keys for G and $O(nmr)$ time to form the S'_{rnd_r} sets. Thus, the additional time required to attack G (beyond the time required to attack G'_{b+y} is $O(nm(r+t_r))$. The only unknown value is m, the number of keys produced by the attack on G'_{b+1}. If m is large enough, to the extent that it approaches the average number of keys to test in a brute force attack on G', then this contradicts the assumption that an efficient attack exists on G' because the attacker is left with a large set of potential keys for decrypting

additional ciphertexts.

To perform the attack on G when given a set of (plaintext, ciphertext) pairs for G, convert the pairs into a set of (plaintext, ciphertext) pairs for G' and find the round keys for G' then for G as follows: Let rk_{r+l}, where $1 \leq l \leq r' - r$, denote a set of randomly chosen round keys for rounds $r+1$ to the last round of G' that will be held constant, and let RF_{end} denote the last $r' - r$ rounds of G' using these round keys. Given a set $\{(P^*, C^*)\} = \{(pi^*, ci^*)\}$ for $i = 1$ to n known (plaintext, ciphertext) pairs for G, create the set $\{(P, C)\}$ of (plaintext, ciphertext) pairs to use in the attack on G' by setting $pi \parallel xi = pi^* \parallel 0$ and $ci \parallel zi = RF_{end}(ci^* \parallel 0)$ for $i = 1$ to n. This choice of $ci \parallel zi$ corresponds to an output of $ci^* \parallel 0$ in the r^{th} round of G'. For the set of (P, C) pairs are created, $\{P, U)\} = \{(pi^* \parallel 0, ci^* \parallel 0)\}$. Apply the attack on G' to solve for the round keys of G' then produce the sets $\{P, U)_{rnd_r}\}$ and S_{rnd_r}. The sets of round keys in S_{rnd_r} will be consistent with the (plaintext, ciphertext) pairs in $\{P, U)_{rnd_r}\}$.

We now discuss how the number of (plaintext, ciphertext) pairs required. Recall that so far we have defined a method which produces a set of at least $\frac{n}{2^{yr}}$ (plaintext, ciphertext) pairs which are consistent with the round keys. This lower bound on the number of plaintext, ciphertext pairs can be slightly increased to $\frac{n}{2^{y(r-2)}}$ by using $b + y$ bit plaintexts that are the same in the rightmost y bits, and by defining the ui values representing the ciphertext output of G in the r^{th} round of G' to be the output of the r^{th} round prior to the swapping step. This will result in $|S'_{rnd_1}| = n$ and $|S'_{rnd_r}| = |S'_{rnd_{r-1}}|$, thus in first and r^{th} rounds the set of (plaintext, ciphertext) pairs is not reduced. The number of (plaintext, ciphertext) pairs produced for G that are consistent with the round keys for G is $\geq \frac{n}{2^{y(r-2)}}$. The number of possible plaintexts for G is 2^b; therefore, it is necessary for $y(r-2) \leq b$ for the attack on G' to be applied to G. Our method defined in Section 3 overcomes this restriction at the cost of increased computation.