On Elastic Block Ciphers and Their Differential and Linear Cryptanalyses

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

Motivated by applications such as databases with nonuniform field lengths, we introduce the concept of an *elastic block cipher*, a new approach to variable length block ciphers which incorporates fixed sized cipher components into a new network structure. Our scheme allows us to dynamically *"stretch"* the supported block size of a block cipher up to a length double the original block size, while increasing the computational workload *proportionally* to the block size. We show that traditional attacks against an elastic block cipher are impractical if the original cipher is secure. In this paper we focus on differential and linear attacks. Specifically, we employ an elastic version of Rijndael supporting block sizes of 128 to 256 bits as an example, and show it is resistant to both differential and linear attacks. In particular, employing a different method than what is employed in Rijndael design, we show that the probability of any differential characteristic for the elastic version of Rijndael is $\leq 2^{-(blocksize)}$. We further prove that both linear and nonlinear attacks are computationally infeasible for any elastic block cipher if the original cipher is not subject to such an attack and involves a block size for which an exhaustive plaintext search is computationally infeasible (as is the case for Rijndael).

Keywords: Block Cipher Design, Elastic Block Cipher, Variable Length Block Cipher, Linear Cryptanalysis, Differential Cryptanalysis.

1 Introduction

Block ciphers typically support a small number of fixed block sizes, usually just one, with most now operating on 128-bit blocks. However, applications are not designed around fixed block sizes. This results in the need for plaintext padding schemes. Although widely used in practice, padding imposes additional computational and space overhead to the encryption process. For example, most fields in databases are not defined to be an integral number of 16 bytes. Consider any financial or retail database, or tables in common database benchmarks [16]: lengths of 20,25,40, and 60 bytes are common for fields containing information such as name, address, transaction history, product description and order information. Encrypting individual fields, rows or columns to enable efficient querying requires padding. As a result, the amount of space that is "wasted" in a database due to encryption can be significant, especially when millions of records are stored. In database benchmarks for financial transactions and customer records of online retailers, encrypting individual fields with a variable length block cipher results in at least a 10% space savings over the entire database and over a 35% space savings for some tables. Similar issues arise when considering network traffic protection, *e.g.,* in IPsec [7], where a packet may be expanded by the encryption and require fragmentation, which degrades performance. A natural alternative is to use a stream cipher, but this is not appropriate for applications such as databases where numerous relatively small items must be encrypted and decrypted individually to allow for efficient querying. Furthermore, the use of a stream cipher is not always attractive since it sacrifices data and key diffusion, and requires synchrony between the sender and the receiver, which is an unsuitable assumption for many applications.

Previous proposals addressing this problem treat an existing block cipher as a black box and add operations around it to obtain support for variable sized blocks. We note that such proposals do not attempt to make the amount of work required proportional to the block size. Thus they are fundamentally different from our method. The method in [1] involves four applications of the cipher for block lengths between the original length and less than twice that length. Therefore, the resulting cipher requires at least four times the work of the original block cipher per block, regardless of the block size. An improved version of the same concept, discussed in [12], requires at a minimum two applications of the original block cipher plus an application of a hash function and additional operations regardless of the block size. Instead, we aim to have the computational workload gradually expand to twice that of the original block cipher as the block length expands, both as a pragmatic performance concern and as a more theoretical possibility motivated by a principle of security that one must assure as much security as needed without employing costly methods as an overkill.

We introduce the concept of an *elastic block cipher,* which allows us to "stretch" the supported block size of a block cipher up to a length double the original block size, while increasing the computational workload proportionally to the block size. We combine the length-preserving aspects of stream ciphers with the diffusion properties of block ciphers, and uses well analyzed components of existing block ciphers to leverage prior work as much as possible.[1] Our method permits block sizes to be set based on an application's requirements, allowing, for example, a non-traditional block size to be used for all blocks or a traditional block size to be used for all but the last block in a given mode of operation. Our intent is not to design an *ad hoc* new cipher (as was done in [6] and [15] and is always possible), but to systematically build upon existing block ciphers by creating a layered network that uses the round function from an existing block cipher and allows bits beyond the supported block size to be interleaved with bits in the supported block size. We note that since the original components were optimized for the fixed-size design, we do not expect to obtain the same efficiency that may be possible if a block cipher was designed from scratch and optimized for variable block sizes. However, our goal is to provide a method which allows continued use of existing algorithms, and to analyze and understand the issues with such an approach.

We treat the elastic version of a cipher as a new cipher in terms of the analysis required and therefore analyze it in terms of resistance to attacks and performance. However, we are able to reuse properties (such as differential characteristic bounds of the round function) from the original cipher along with properties of our network structure to assist in the analysis. In fact, our design enables an analysis which shows traditional attacks against an elastic block cipher are impractical if the original cipher is secure.

In this paper, we present the algorithm for creating an elastic block cipher given a fixed-size block cipher. We use Rijndael [11] with 128 bit blocks as a concrete example of applying the algorithm, resulting in a cipher that supports block sizes of 128 to 256 bits. We prove that the elastic version of Rijndael is resistant to both linear [9] and differential [2] attacks. We show that the probability of any differential characteristic for the elastic version of Rijndael is $\leq 2^{-(blocksize)}$. *We then prove in general that both linear and nonlinear attacks are computationally infeasible for an elastic block cipher if the original cipher is not subject to such an attack and involves a block size for which an exhaustive plaintext search is computationally infeasible.*

The contributions of this paper are:

[1] Although our approach still requires analysis, this is not done on a per-block-size basis but on the scheme as a whole.

- We propose a method for constructing a variable length block cipher from any existing block cipher such that the workload is proportional to the block size.
- We utilize a network structure into which the round function of a block cipher is inserted to create a variable length block cipher.
- We prove that linear and non-linear attacks on elastic block ciphers are computationally infeasible if the original cipher is immune to such an attack.
- We demonstrate a method for bounding the probability of differential characteristics of an elastic block cipher, using an elastic version of Rijndael as an example.

While we validated the elastic block cipher against the linear, non-linear and differential attack scenarios, further validation and investigations both on the design side and on the cryptanalysis side (*e.g.,* considering additional attacks such as Square [5] and higher order differentials [8]) are left as future work.

The remainder of the paper is organized as follows. In Section 2 we explain our approach for constructing elastic block ciphers. In Section 3 we prove that a differential attack on an elastic version of Rijndael is not possible. In Section 4 we show how to convert a linear attack on the elastic version of a cipher to a linear attack on the original cipher. We extend this result to cryptanalysis using nonlinear equations. In Section 5 we conclude the paper.

2 Elastic Block Cipher Construction

2.1 Algorithm

We begin with a description of the algorithm (network) for expanding the encryption and decryption functions of existing block ciphers to accept blocks of size b to $2b$ bits, where b is the block size of the original block cipher. We neither modify the round function of the block cipher nor decrease the number of rounds applied to each bit; instead, we create a method by which bits beyond the supported block size can be interleaved with bits in the supported block size (while allowing any size up to double the original one). The result is a network which allows us to reuse the properties of the original block cipher's round function. We first present the steps for converting the fixed sized block cipher to a variable length block cipher. We explain the reasoning behind the specific steps in Section 2.2. The description we provide here is intended only to give the reader an understanding of the method and convey an overview of the reasoning behind specific steps, and is not our complete analysis of the approach. The general structure of the method is shown in Figure 1.

The following notation and terms will be used in the description and analysis of the elastic block cipher:
Notation:

- G denotes any existing block cipher that is structured as a sequence of rounds.
- r denotes the number of rounds in G.
- b denotes the block length of the input to G in bits.
- y is an integer in the range $[0, b]$.
- G' denotes the modified G with $b + y$ bit input for any valid value of y. G' will be referred to as the elastic version of G.
- r' denotes the number of rounds in G'.
- A bit (position) input to a block cipher is called *active* in a round if the bit is input to the round function. For example, in DES [10], half of the bits are active in each round, while in Rijndael all bits are active in each round.

- The round function will refer to one entire round of G. 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.

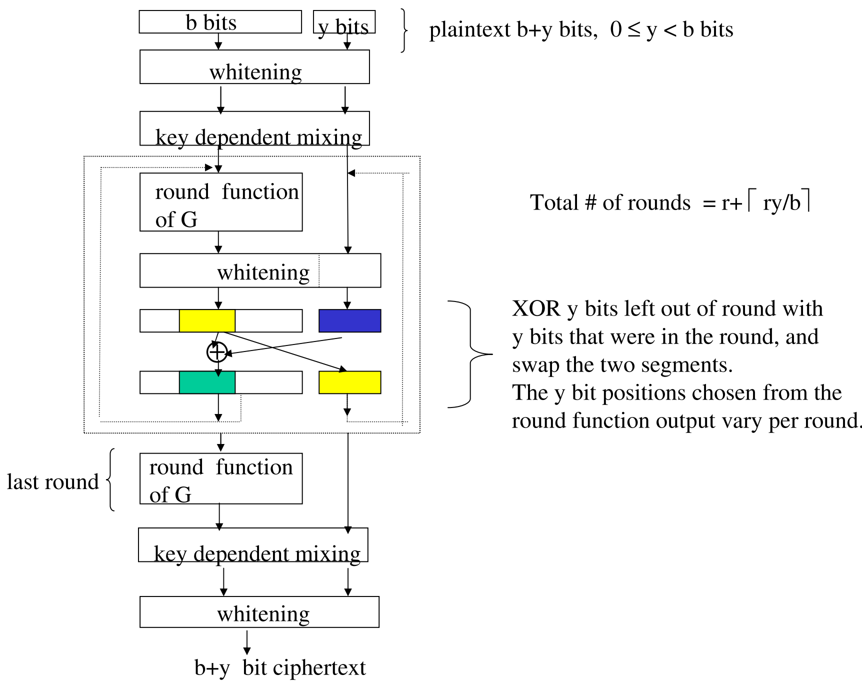

Figure 1: **Elastic Block Cipher Structure**

Given G and a plaintext of length $b+y$ bits, make the following modifications to G's encryption function to create the encryption function of G':

1. Set the number of rounds, r', such that each of the $b+y$ bits is input to and active in the same number of rounds in G' as each of the b bits is in G. $r' = r + \lceil yr/b \rceil$.
2. XOR all $b+y$ bits with key material as the first step. If G includes whitening as the first step prior to the first round, the step is modified to include $b+y$ bits (the original b bit "inner block" whitening for the leftmost bits, as well as "side whitening" of the extra y bits). If G does not have an initial whitening step, this step is added to G'. In either case, additional bits of expanded key material are required beyond the amount needed for G.
3. Apply a key-dependent mixing step that permutes or mixes the bits in a manner that any individual bit is not guaranteed to be in the rightmost y bits with a probability of 1. This will be referred to as the mixing step. Similarly, a key dependent mixing step is added at the end of the last round. In this implementations, a key dependent rotation is used. The leftmost b bits that are output from the initial mixing step are the input to the first round function.
4. If the round function of G includes XORing with key material at the end of the round and as a final step in the algorithm, the whitening is expanded to include all $b+y$ bits (inner-block as well as

side whitening). If G does not contain end of round whitening and whitening as the last step in the algorithm, add these whitening steps and apply them to all $y + b$ bits. In either case, additional bits of expanded key material are required beyond the amount required by G.

5. Alternate which y bits are left out of the round function by XORing the y bits left out of the previous round function with y bits from the round function's output, then swap the result with the y bits left out of the previous round. This step is performed after the end of round whitening. Specifically:

   (a) Let Y denote the y bits that were left out of the round function.

   (b) 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. We discuss this further in Section 2.2.

   (c) Set $Y \leftarrow X \oplus Y$.

   (d) Swap X and Y to form the input to the next round.

   This step will be referred to as "swapping" or the "swap step," and may be added to the last round if we require that all rounds be identical. However, having the swap step after the last round does not provide additional security.

The result, G', is a permutation on $b+y$ bits. Its inverse, the decryption function, consists of the network applied in the reverse and the round function replaced by its inverse.

2.2 Explanation of Algorithm

The method is designed for G' to be equivalent to G (with the possible addition of whitening and the key-dependent mixing steps) when the data is an integral number of b-bit blocks, while accommodating a range of b to $2b$-bit blocks. The construction allows us to reuse the round function of G and thus build upon the round function's properties, including its differential and linear bounds. The following is an overview of why specific steps are included in the construction.

Step 1: Each bit position of the input is required to be active in the same number of rounds in G' as the number of rounds in which each bit is active in G. This requirement allows the computational workload to increase proportionately to the block size while avoiding a reduced round attack on G from being applied to G'. Consider what happens if $y = b - 1$ and no rounds were added to G when creating G': $b - 1$ bits would be active in only $\frac{1}{2}$ of the rounds in which a bit is normally active in G. As y increases, the number of rounds increases gradually from $r + 1$ when $0 < \frac{ry}{b} \leq 1$ to $2r$ when $r - 1 < \frac{ry}{b} \leq r$.

Steps 2 and 4: Whitening is a useful heuristic against attacks which combine rounds. The whitening steps allow rounds to work in isolation from each other in that the input to a round is unknown even when given the output of the previous round.

Step 3: A key-dependent permutation or mixing of bits prior to the first round increases the extent to which the first round contributes to preventing a differential attack. The mixing step will need to take less time than a single round; otherwise, an additional round can be added instead to decrease the probability of a specific differential occurring. A trivial mixing that prevents the attacker from knowing with probability 1 which y bits are excluded from the first round is a key-dependent rotation. Similarly, a key-dependent mixing step after the last round will prevent a single round differential from occurring with a probability of 1 in the first round of decryption.

Step 5: $X \oplus Y$ is performed instead of merely swapping X and Y in order to increase the rate of diffusion. If G does not have complete diffusion (meaning every bit impacts all other bits) in one round, then at the end of the first round there is some subset S of bits output from the round that have not been impacted by some of the bits in X. While the bits in Y may impact S in the second round, swapping X and Y would result in the bits in X having no impact in the second round; whereas, swapping X with $X \oplus Y$ will allow the bits in X to impact the second round. The selection of X depends on the round function of the specific block cipher. The bit positions selected for X should vary amongst the rounds to ensure that all bit positions are involved in both the b-bit and y-bit components, as opposed to always selecting the same y positions for use in X. Varying the positions also increases the diffusion amongst the bits. If all input bits to the round are utilized in the same manner, the bit positions chosen for X can be rotated across the rounds. For example, in Rijndael all bytes are processed by the same functions within the round. In that case, it is sufficient to select X to be consecutive bits starting at position $a_1 + a_2 * i (\bmod b)$ in round i for some constants a_1 and a_2. When G is such that the round function only processes a subset of the b bits in each round, we choose to add the swap step at the point in which all b bits have been processed by the round function. For example, when G is a Feistel network we add the swap step and whitening after every two rounds (a complete cycle) so a bit participates in the actions applied to each half of the b bit block once prior to potentially being swapped out regardless of its position.

By treating the round function of the original block cipher as a black box, the properties of the round function in regards to differential and linear relationships still hold. The number of rounds in the elastic cipher is set so each bit position passes through the round function at least the same number of times as in the original cipher. Furthermore, the XOR in the swap step results in each bit position influencing every round except the first round. The following fact regarding complete diffusion results from the network structure.

Fact I: *If complete diffusion occurs after q rounds in the original cipher, it occurs after at most $q + 1$ rounds in the elastic version.*

Proof: Ignoring the initial key dependent mixing step, by the end of the first round the y bits left out of the round have not impacted any other bits. The rate of diffusion for the b bits input to the round function is the same as in the original cipher. The inputs to the second through the last applications of the round function are influenced by all $b + y$ bits because of the XOR in the swap step. Thus beginning at the second round, complete diffusion will occur within q more rounds, and thus occurs by the end of the $q + 1^{st}$ round.

Key Schedule: Our options for a key schedule include modifying the key schedule of G to produce additional bytes, increasing the original key length and running the key schedule multiple times, or using an existing efficient stream cipher that is considered secure in practice (this also permits the key schedule to independent of the choice of G). At a minimum, we assume any expanded key bits which are external to the round function are independent of expanded key bits used within the round function and are created independently of the $b + y$ bit data block input to the cipher.

Decryption: The inverse of the round function, if it is not its own inverse, must be used for decryption. While the structure is similar to an unbalanced Feistel network [14], it is not a Feistel network due to bits output from the round function in the i^{th} round becoming the bits omitted from the $i+1^{st}$ round. In constrast, in an (unbalanced) Feistel network bits input to the round function in the i^{th} round become the bits omitted from the round function in the $i + 1^{st}$ round. Thus, it is not possible to perform decryption by simply running the ciphertext through the encryption function of G' with the round keys used in reverse order. Designing the elastic cipher in this manner increases the diffusion rate compared to that of an unbalanced Feistel network. We remind the reader that the swap step is added after a complete cycle when the original cipher is a Feistel network, thus the inverse of the "round" function in the elastic version is merely running the cycle in reverse, as is normally done in any block cipher which is a Feistel network.

2.3 Elastic Rijndael

The following are details regarding the elastic version of Rijndael. When G is Rijndael with a 128 bit block size, G' accepts blocks from 128 to 256 bits ($b = 128$ and $0 \leq y \leq 128$). The number of rounds in G' ranges from 10 when $y = 0$ to 20 when $y \geq 116$. We use a stream cipher (RC4 [13]) as the key schedule. The mixing steps are key dependent rotations of the data block. As mentioned previously, the bits selected from the rightmost b bits for the swap step are y sequential bits. We move the starting position to the right one byte each round so the y bits swapped out of the leftmost b bits after round i start at byte i mod 16. We expand the whitening steps in Rijndael to cover all $b + y$ bits. Notice that when the block size is 128 bits, if the original key schedule is used for the round keys, then the elastic version is Rijndael with the addition of the initial and final mixing steps.

3 Differential Cryptanalysis

3.1 Overview

We show that a differential attack on an elastic version of Rijndael is impossible for block sizes of $128 + y$ bits where $0 \leq y \leq 128$. We note that, similar to the linear attack analysis in Section 4, a general reduction between the elastic version of a cipher and the original version can be used to show that the elastic version of a cipher is immune to a differential attack if the original cipher is immune to such an attack. Our intent here is to show how the probabilities of the differential characteristics can be calculated or bounded in the elastic version of a block cipher when given the upper bound on the probability a differential characteristic for one application of the round function. Our analysis is performed using a computer model that tracks how many bytes contain a non-zero differential characteristic in each round. This allows us to form an upper bound on the probability a differential characteristic across all rounds of the cipher. The variable block size and the swap step significantly increase the number of cases to explore when determining the probability of a differential characteristic. This is the reason for why we decided to model and trace the differential characteristics as we describe here, and for why we did not use the approach from the differential trails analysis of Rijndael [4]. Our analysis results in the following claim.

Claim I: *For the elastic version of Rijndael accepting block sizes of* $128 + y$ *bits for* $0 \leq y \leq 128$*, the probability of a differential characteristic occurring is* $\leq 2^{-128-y}$.

We use the remainder of this section to explain the computer model and how the result in Claim I was obtained. We note that our analysis is general in terms of block size but only considers a single method for selecting the bits to swap after each round as opposed to all possible ways of selecting y bits from 128 bits. It is computationally infeasible to include in the model all possible options for selecting the bits involved in the swap step. Therefore, we model the swap step to correspond to how we implemented it in our software version of the elastic cipher. In our implementation, the swap is performed by selecting y consecutive bits from the round function's output to swap with the y bits left out of the round function and the starting position of the bits selected rotates to the right one byte each round. Also, we omit the initial and final key dependent permutations from the model for simplicity since they will only decrease the probability of any differential characteristic. We first explain how the model is designed and then follow with specific results.

3.2 Model Description

We describe here how we modelled the elastic version of Rijndael and the impacts of each step on the differential characteristic. The following terms and notation are used:

- delta refers to the non-zero difference between two bit strings. Given two data blocks, a delta byte means there is a non-zero difference between the two bytes in a specific position in the two data blocks.
- exp: The probability a specific differential characteristic occurs (across some number of rounds) is 2^{-exp} where exp is a non-negative integer. If $exp \geq b + y$ for all differentials across r' rounds in the elastic version of a block cipher then the elastic version is immune to a differential attack regardless of the attacker's resources because more than 2^{b+y} plaintext, ciphertext pairs are required for an attack.
- $B||Y$ indicates the $b + y$ bit plaintext input to the elastic block cipher. For the elastic version of Rijndael, B refers to the leftmost 16 byte portion of the data block input to Rijndael's round function and Y refers to the rightmost y bits of the data block.
- $|\Delta B|$ and $|\Delta Y|$ refer to the number of delta bytes in B and Y, respectively.

In the model, the B portion of the data is viewed as a 4 by 4 matrix of bytes as defined in the description of AES [11]. Refer to Figure 2. The model tracks the number of bytes which differ per column of this matrix at the start of each application of the round function. We refer to this number as the column delta. For example, if bytes in the 1^{st},2^{nd} and 4^{th} rows of the second column contain a delta, the delta for column 2 is 3. We track the delta at the column level instead of the exact byte position due to the number of cases that would need to be considered if every byte is tracked. The lower bound of a probability for a differential characteristic is determined by adding up the number of delta bytes entering each application of Rijndael's round function and multiplying the result by 6 to obtain a lower bound for exp. The multiplication by 6 is due to the fact that the probability a specific difference in two one byte inputs to the S-Box produces a specific difference in the two outputs of the S-Box is 2^{-6} or 2^{-7}, depending on the specific difference (refer to pages 205-206 of [4]). The ShiftRows and MixColumns steps in Rijndael's round function, and the swap step and end of round whitening transform the differential in a fixed manner in that the specific input differential results in a specific output differential with probability 1 for each step. Therefore, the differential probability for a single round is $\leq 2^{-exp}$ where $exp = 6*$ (number of bytes with a delta).

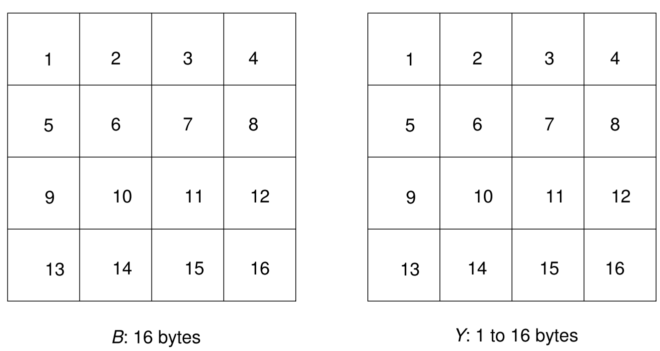

Figure 2: **Data Block as 4x4 Matrices**

The bytes in the Y portion of the data are also viewed as four columns of a 4 by 4 matrix as shown in Figure 2, with the matrix being filled in one row at a time. For example, if Y contains 7 bytes, there will be three columns containing two bytes each and one column containing one byte. The last byte is a partial byte if Y is not an integral number of bytes. The swap step swaps Y with y bits in $\lceil \frac{y}{8} \rceil$ sequential bytes from B, moving the starting position one byte to the right in B each round and wrapping around to the beginning of B as needed. Therefore, a column of Y is swapped with a column of B.

The column deltas are tracked through each step of the round function and through the swap step, with all possible resulting cases explored recursively. The following is an overview of how the tracking is performed. An input state consisting of the number of delta bytes per column for both B and Y along with the size of Y in bytes are input to the model. The model determines all possible states of B resulting from the Rijndael round function then, for each of these states, determines all possible states of $B||Y$ resulting from the swap step. These states become inputs to the next round. The specific impacts of each step are modelled as follows:

Rijndael's Round Function:

- SubBytes: The S-Box has no impact on the column delta because the model is not tracking the actual delta value but only the number of bytes with a delta. Each delta byte input to the SubBytes step produces a delta byte in the output of this step and does not alter the position of the delta byte.
- ShiftRows: This step causes each byte in the 2^{nd} through 4^{th} rows to change columns. Since we are tracking the number of delta bytes per column and not their exact positions, this step leads to multiple cases to investigate. For example, if one column has a single delta byte and the other three columns have zero deltas, the single delta byte may be in any of the four columns after ShiftRows is applied.
- MixColumns: Given the number of delta bytes in a column when entering MixColumns, we investigate all possible states which can result. If there are x delta bytes in a column at the start of MixColumns, there can be anywhere from $5 - x$ to 4 delta bytes in the column at the end of MixColumns because we do not know the specific delta values input to MixColumns. For example, two delta bytes in a column will result in either 3 or 4 delta bytes in the output column. This was determined by applying the MixColumns step to all 4-byte values for a single column.
- AddRoundKey: The whitening steps have no impact on the number of and positions of the delta bytes.

Swap Step: The impact of XORing a column of Y with a column of B and swapping the results is determined. Given that the specific positions of the delta bytes are not tracked, the swap step may lead to more than one state and all possible resulting states are investigated. For example, if a column of B has a delta of 2 after the Rijndael round function is applied, the swap impacts all bytes and the corresponding column in Y has a delta of 1, the resulting column in B may have a delta of 1,2 or 3 bytes and the resulting column in Y will have a delta of 2 bytes. A second example illustrates the cases where the swap does not impact all bytes of the column of B (this will happen to one or more columns of B when $y \leq 120$). If only one byte of a column in B is impacted by the swap, the column in B has a delta of 2 and the byte from Y being swapped into the column is a delta byte, the resulting columns in B and Y may have deltas of 1 and 1, 2 and 1, or 3 and 0 bytes, respectively. Since the exact delta values are not known, the model assumes that any delta byte in Y which is XORed with a delta byte in B can produce a zero (a non-delta byte).

Even with the multiple cases to investigate in ShiftRows, MixColumns and the swap step, the amount of computation required is lower than what is required to track the exact position of each delta byte. The multiple cases to investigate due to MixColumns can be avoided only if the exact delta for each byte along with each delta byte's exact position was tracked, which requires computing all 2^{128+y} differentials. As a result of tracking the deltas only at the column level, the model will investigate states which cannot be reached. The inclusion of these unattainable states is not an issue because they can only lower exp and thus reduce the tightness of the bound. Among the unattainable states there is one set which we eliminated via manual analysis and which we describe in Section 3.3 during the discussion of Claim III.

3.3 Results

We ran the model for each case where Y is an integral number of bytes from 1 to 16. These cases cover all values of y for $0 < y \leq 128$. Since we are tracking only which byte positions involved deltas, the lower bound for $y = 8x$ where x is an integer such that $1 \leq x \leq 16$ covers the cases of y such that $8(x-1) < y \leq 8x$. For each case, the model was run through three rounds for all possible input states except for the state involving no differential in B since this produces a first round probability of 1 and involves the same analysis as the cases where B contains a delta and Y has a zero differential, only with one less round. States producing a three round bound which did not exclude the possibility of a differential attack were traced through subsequent rounds with the number of rounds depending on the exact size of Y and the probability produced after each round. An example of the our analysis is provided in Appendix A. For each case (value of Y tested), we found that the total exp across $r'-1$ rounds is at least $128+y$. Recall that r' depends on the size of Y. Therefore, the probability of a specific differential characteristic occurring is $\leq 2^{-128-y}$. The probability was computed across $r'-1$ rounds instead of r' rounds to cover the case of the first round producing a differential characteristic with a probability of 1 due to the key dependent permutations being omitted.

Claim II: *The elastic version of Rijndael with block size ranging from 129 to 224 bits in which any three round differential characteristic occurs with probability* $\leq 2^{-30}$.

Proof: For the cases where Y contains at most 12 bytes, the total exp across three rounds is ≥ 30. This was determined via the model for the cases in which B contains a delta in the first round's input. In the case where B contains no delta in the first round, exp is also at least 30 for the first three rounds regardless of the size of Y. This is because the delta input to the second round function is the delta in Y input to the cipher. If $1 \leq |\Delta Y| < 5$ and B has no delta in the plaintext, the delta output from the second application of the round function will be at least $5 - |\Delta Y|$ and the total $|\Delta B|$ across rounds 2 and 3 will be ≥ 5. If $|\Delta Y| \geq 5$ in round 1, then $|\Delta B| \geq 5$ in round 2. Therefore, the total exp across rounds two and three is at least 30.

For block sizes between 225 and 256 bits, we obtain the following result:

Claim III: *The elastic version of Rijndael with block size ranging from 225 to 256 bits in which any three round differential characteristic occurs either with probability* $\leq 2^{-30}$ *or with probability* 2^{-12}. *If the three round probability is* 2^{-12}*, the five round probability is* 2^{-132}.

Proof: For the block sizes where Y involves 13 or more bytes, the model produces a three round $exp \geq 30$ except in the scenario depicted in Figure 3. In this exception case, a three round $exp = 12$ is possible, but leads to a five round $exp = 132$. This 5 round exp is large enough such that when it is combined with the other possible three to five round exp values it produces a total $exp \geq 128 + y$ for $r'-1$ rounds. The three round $exp = 12$ can be produced by having a single byte delta in the 4^{th} row of the 4x4 matrix representing B. Let a denote the value of this single byte delta. Let a' be the value after SubBytes is applied. This will produce delta of 4, d, in a single column as a result of MixColumns. Suppose Y contains a delta which is entirely in the column which will be swapped with the 4 byte delta output from the round function. Also assume ΔY and d are identical in the three bytes in rows 1 to 3, and produce a delta byte = a in the byte in the 4^{th} row. Furthermore, assume the two bytes producing this difference of a are such that the SubBytes step will produce the same delta, a', as it did in the first round. This results in the input to the second round function involving a single delta byte with value a which is identical to the delta input to the first round function, with the exception that it will be in a different column. Since a is in the fourth row, it will be rotated 3 columns to the left during ShiftRows. The starting position of the swap step moves one byte to the right each round. In the second round, ΔY contains the 4 bytes swapped out of the first round function's output. These 4 bytes are in a single column whose value is $a', a', 3a', 2a'$. The delta output from the second round function will also be a single column containing $a', a', 3a', 2a'$. Thus the swap step after the second round will result in no delta input to the third round function and a ΔY of 4 bytes in a single column. Therefore, a 4 byte delta in a single column is input to the 4^{th} round, Y contains no delta in the 4^{th} round, a 16 byte delta is output from the 4^{th} round and a 16 byte delta is input to the 5^{th} round.

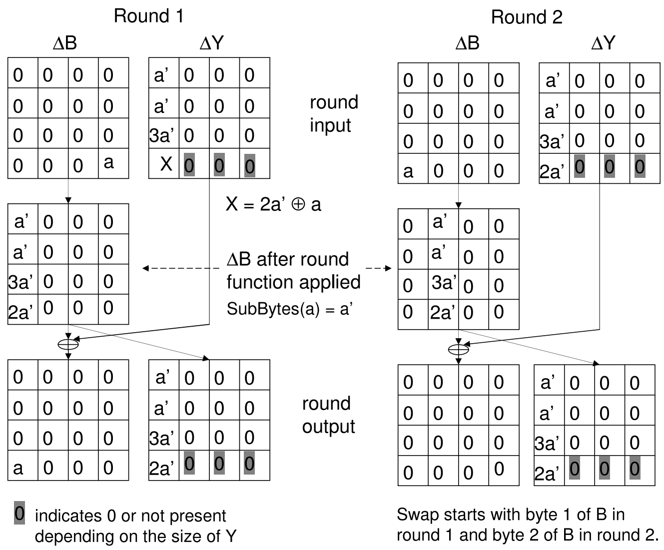

Figure 3: **Case With 3 Round** $exp = 12$ **and 5 Round** $exp = 132$

The exception case occurs only when Y is at least 13 bytes because it requires that Y contains enough bytes to impact an entire column of B in the swap step. Twelve bytes in Y populate three rows entirely (3 bytes in every column). Unattainable states involving a similar scenario with three delta bytes in a single column cancelling in the swap step were reached by the program when Y was less than 13 bytes due to the program not tracking the exact byte positions and exact byte values. These states were manually eliminated when determining which states needed to be traced through additional rounds.

4 Linear Cryptanalysis

When proving an elastic version of Rijndael is not subject to a practical linear attack, we are able to take a general approach that relates any elastic block cipher, G', to its original cipher, G. Therefore, we are able to obtain a result that applies to all elastic block ciphers as opposed to only the elastic version of Rijndael.

We will show that a linear relationship (characteristic) across r rounds of G' implies such a relationship across r rounds of G. If any such linear relationship holds with a probability such that fewer than 2^b plaintext, ciphertext pairs are required for an attack, then G is subject to a linear attack. If at least 2^b plaintext, ciphertext pairs are required for an attack on r rounds of G', then either the attack is infeasible on r rounds of G' from a practical perspective or G is subject to a brute force attack in practice. Note that we are dealing with an attack on only r rounds of G' and the probability of a linear relationship holding across $r' = r + \lceil \frac{ry}{b} \rceil$ rounds of G' will be less than that for r rounds. More specifically, if the attack on G' involves a maximum correlation between plaintext, ciphertext and key bits which occurs with probability $\leq 2^{-b}$ on r rounds (thus requiring in practice about $\geq 2^b$ plaintexts), then an attack on $2r$ rounds involves a maximum correlation that occurs with probability $\leq 2^{-2b}$. In this case, G' is practically secure against a linear attack when $\lceil \frac{ry}{b} \rceil = r$ regardless of the computational requirements.

Without loss of generality, we assume any linear relationship involves the expanded key bits as opposed to the original key input to the key schedule. We omit the initial and final key dependent permutations from our analysis because these permutations do not impact any linear relationship that exists across r rounds of G'. A direct implication of our result is that if G' is subject to nonlinear cryptanalysis, then so is G. From these results, we conclude that an elastic version of Rijndael is not subject to a practical linear attack (since Rijndael is not subject to such an attack based on the linear trails analysis in [3], pages 30-31, and the 128 bit block size is not subject to an exhaustive search in practice) and is subject to a nonlinear attack only if Rijndael is subject to such an attack.

Claim IV: *Given a block cipher G with a block size of b bits and r rounds, and its elastic version G' with a block size of $b + y$ bits for $0 \leq y \leq b$ and r' rounds where $r' = r + \lceil \frac{yr}{b} \rceil$, if G' is subject to a linear attack on r rounds then either G is subject to a linear attack or the resources exist to perform an exhaustive search on G over all plaintexts.*

Proof: We will show that if a linear attack exists for r rounds of G' then a linear attack exists for G. If the linear attack on r rounds of G' requires encrypting more than 2^b plaintexts then either the attack is computationally infeasible or G is insecure independent of the attack (since the attacker has the resources to encrypt 2^b plaintexts). To understand how a linear relationship (if one exists) between the plaintext, ciphertext and expanded key bits is determined for G', we first consider how a linear relationship is derived for a block cipher structured as a series of rounds with block length b and then add the impact of the whitening and swap step to these relationships. We number the rounds from 1 to r. We will refer to any initial whitening step that occurs prior to the first round as round 0 and the round function of round 0 is the initial whitening. The relationship between the output of the j^{th} round and the input to the $(j+1)^{st}$ round which we define via notation here is depicted in Figure 4 in Appendix B for both G and G'. We use the following notation:

- When we say two bits, $x1$ and $x2$, cancel each other in an equation we mean $x1 \oplus x2 = 0$ with probability 1.
- Let u_{ji} denote the i^{th} bit of the input to the round function in round j, $1 \leq i \leq b$, $0 \leq j \leq r$.
- Let v_{ji} denote the i^{th} bit of output from the round function in round j, $1 \leq i \leq b$, $0 \leq j \leq r$.
- Let n_j denote the number of expanded key bits used in the round function in round j, $0 \leq j \leq r$. This does not include any end of round whitening added to form G', but does include the end of round whitening if it is part of the round function of G (as is the case with Rijndael). The round function in round 0 is the identity function and $n_0 = 0$ if G does not contain initial whitening.
- Let ek_{ji} denote the i^{th} expanded key bit in the round function in round j, $1 \leq i \leq n_j$.
- Let $L_j([u_{j1}, ...u_{jb}] \oplus [v_{j1}, ...v_{jb}] \oplus [ek_{j1}, ...ek_{jn_j}])$ denote the set of linear equations (if any) relating the input, output and round key bits with non-negligible probability for the round function in round j, $0 \leq j \leq r$. We will abbreviate this as L_j. An equation in L_j holds with probability $\frac{1}{2} + \alpha$ for some non-negligible α such that $0 < \alpha \leq \frac{1}{2}$. We note that an equation which reflects a negative relationship, meaning the equation holds with probability $\frac{1}{2} - \alpha$, can be written as an equation holding with probability $\frac{1}{2} + \alpha$.
- Without loss of generality, the equations in L_j are in reduced form; for example, $u_{j2} \oplus u_{j2} \oplus u_{j2}$ will be reduced to a single u_{j2}.

A linear relationship across consecutive rounds is obtained by combining the linear equations for each of the rounds, with v_{ji} becoming $u_{(j+1)i}$. A linear relationship exists between plaintext, ciphertext and expanded key bits if the intermediate round inputs and outputs cancel when combining the per round equations, leaving equation(s) involving only u_{0i}'s, v_{ri}'s and expanded key bits.

We now consider how the steps between the rounds in G' impact the linear relationships across the rounds.

- Let Y denote the rightmost y bits of the data block for a $b + y$ bit data block.
- Let Γ' refer to the set of the equations used in a linear attack on r rounds of G' formed from combining the L_j's for the individual rounds along with the end of round whitening and swap steps.
- Let Γ refer to a set of linear equations for G formed from equations in Γ'.
- Let kw_{ji} denote the i^{th} key bit used for the whitening step added in round j when constructing G', $1 \leq i \leq b + y$. We set $kw_{ji} = 0$ for $1 \leq i \leq b$ if the round function of G includes end of round whitening because G' does not add whitening to the b bits when it is already present.
- Let w_{jl} denote the l^{th} bit of the Y portion of the data, for $1 \leq l \leq y$. $w_{jl} = v_{(j-1)h} \oplus kw_{(j-1)h}$ where $1 \leq h \leq b$ and h is the bit position swapped with bit position l in the previous swap.

With the addition of the whitening and swap steps, the input to the round function is now defined as:

- $u_{(j+1)i} = v_{ji} \oplus kw_{ji}$ when v_{ji} is not involved in the swap step.
- $u_{(j+1)i} = v_{ji} \oplus kw_{ji} \oplus w_{jl} \oplus kw_{j(b+l)}$ when v_{ji} is involved in the swap step. When $j \geq 2$, this can be written as $u_{(j+1)i} = v_{ji} \oplus kw_{ji} \oplus v_{(j-1)h} \oplus kw_{(j-1)h} \oplus kw_{j(b+l)}$.

Notice that the steps between applications of the round function in G' maintain a linear relationship between the output of one round and the input of the next round.

Recall that the key schedule of $G\,'$ produces whitening bits which are created independently of the round function's input, output and the key bits used within the round function. Therefore, these whitening bits will cancel with any v_{ji}, u_{j+1} and/or ek_{ji} with probability $\frac{1}{2} + e$ for negligible e (*i.e.* there is no discernable relationship between these whitening bits and any of the plaintext, ciphertext and expanded key bits used internal to the round function by definition of the key schedule). Thus, the kw_{ji}'s added when forming G' will not increase the probability of a linear relationship between plaintext bits, ciphertext bits and expanded key bits used in the round function. We note that even if a different key schedule is used which does not guarantee independence amongst the kw_{ji}'s and which results in cancellation among some kw_{ji}'s, this is merely cancelling variables which are not present in the linear equations for the round function and thus will not simplify the equations or increase the probability that an equation holds across r applications of the round function (*i.e.* any equation in Γ' cannot have fewer variables than its corresponding equation in Γ as a result of the whitening added to G when forming $G\,'$).

Now we assume a set of equations exist for $G\,'$ and show how to convert them to a set of equations for G. Given the sets, L_j's, of linear equations for the round function in $G\,'$, these same sets of equations hold for G because the elastic version does not alter the round function. Combine these equations across rounds, but when forming the input to one round from the output of the previous round, set kw_{ji} to 0 for $0 \leq j \leq r$ and $1 \leq i \leq b+y$ so these whitening bits are omitted from the resulting equations. This will result in each input bit to the $(j+1)^{st}$ round function being either of the form $u_{(j+1)i} = v_{ji}$ or $u_{j+1} = v_{ji} \oplus v_{(j-1)h}$. Removing the kw_{ji}'s in this manner does not impact how the u_{ji}'s, v_{ji}'s and ek_{ji}'s cancel to form a set of equations involving only u_{0i}'s, v_{ri}'s and ek_{ji}'s. Therefore, the equations will combine to form a set of equations, Γ which is equal to Γ' with any kw_{ji}'s which appear in Γ' set to 0. The equations in Γ may contain at most y extra plaintext bits and at most y extra ciphertext bits since $G\,'$ processes $b + y$ bit blocks and G processes b bit blocks. The attacker can set these extraneous y plaintext bits to any value and the extra y ciphertext bits are identical to y of the bits output from the next to last round function. For any equation $Eq' \in \Gamma'$ which holds with probability $\frac{1}{2} + \alpha$, the corresponding equation, $Eq \in \Gamma$ formed by removing the $kw'_{ij}s$ from Eq' will also hold with probability $\frac{1}{2} + \alpha$. Furthermore, because we only potentially delete variables representing whitening key bits not present in G when converting Γ' to Γ and do not otherwise modify, remove or add equations, the computation and memory resources required to attack G using Γ is less than or equal to that required to attack G'. Thus, a linear attack on a r round version of $G\,'$ implies a linear attack exists on G. If the attack requires more than 2^b plaintext, ciphertext pairs (recall y bits chosen by the attacker can be appended to the b bit plaintexts for G), then either the attack is computationally infeasible or G is subject to a brute force attack and thus is insecure independent of the linear attack.

We generalize Claim IV to include both linear and nonlinear equations.

Claim V: *Given a block cipher G with a block size of b bits and r rounds, and its elastic version $G\,'$ with a block size of $b + y$ bits for $0 \leq y \leq b$ and r' rounds where $r' = r + \lceil \frac{yr}{b} \rceil$, if $G\,'$ is subject to a nonlinear attack on r rounds then either G is subject to a nonlinear attack or the resources exist to perform an exhaustive search on G over all plaintexts. Nonlinear refers to an attack involving a set of equations relating a single plaintext, its corresponding ciphertext and the expanded key bits.*

Proof: The proof follows directly from the proof to Claim IV by removing the qualification in Claim IV's proof that the equations in the L_j sets are linear. If a nonlinear attack exists on r rounds of $G\,'$ then an attack exists on G which runs in time less than or equal to the attack on $G\,'$.

5 Conclusions

We have presented a method for converting any existing block cipher which is structured as a series of rounds into a variable length block cipher which accepts block sizes between one and two times the block length of the original cipher. The method allows us to reuse the original block cipher's round function and results in the computational workload being proportional to the block size. By using a computer model to trace characteristics through multiple rounds of the elastic version of Rijndael, we have shown that the elastic version of Rijndael is not subject to a differential attack. We have also shown that any equations used in either a linear or nonlinear attack on the elastic version of a block cipher translate directly into equations for such an attack on the original cipher. Although several issues are open regarding elastic ciphers, requiring additional research, and we believe that this design and analysis will lead to various new and interesting future investigations.

References

[1] M. Bellare and P. Rogaway. On the Construction of Variable Length-Input Ciphers. In *Proceedings of Fast Software Encryption (FSE), LNCS 1636, Springer-Verlag*, 1999.

[2] E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, New York, 1993.

[3] J. Daemen and V. Rijmen. The Rijndael Block Cipher Version 2.0. Amended AES Proposal to NIST, March 1999.

[4] J. Daemen and V. Rijmen. *The Design of Rijndael: AES the Advanced Encryption Standard*. Springer-Verlag, Berlin, 2002.

[5] Ferguson, Kelsey, Lucks, Schneier, Stay, Wagner, and Whiting. Improved Cryptanalysis of Rijndael. In *Proceedings of Fast Software Encryption (FSE), LNCS 1978, Springer-Verlag*, 2000.

[6] J. Reeds, III. Cryptosystem for Cellular Telephony. US Patent 5,159,634, 1992.

[7] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. Request for Comments (Proposed Standard) 2401, Internet Engineering Task Force, Nov. 1998.

[8] L. Knudsen. Truncated and Higher Order Differentials. In *Proceedings of FSE, LNCS 1008, Springer-Verlag*, pages 196–211, 1995.

[9] Matsui. Linear Cryptanalysis Method for DES Cipher. In *Proceedings of Advances in Cryptology - Eurocrypt '93, LNCS 0765, Springer-Verlag*, 1993.

[10] NIST. FIPS 46-3 Data Encryption Standard (DES), 1999.

[11] NIST. FIPS 197 Advanced Encryption Standard (AES), 2001.

[12] S. Patel, Z. Ramzan, and G. Sundaram. Efficient Constructions of Variable-Input-Length Block Ciphers. In *Proceedings of Selected Areas in Cryptography - SAC 2004, LNCS, Springer-Verlag*, 2004.

[13] B. Schneier. *Applied Cryptography*. John Wiley and Sons, New York, 1996.

[14] B. Schneier and J. Kelsey. Unbalanced Feistel Networks and Block Cipher Design. In *Proceedings of Fast Software Encryption (FSE), LNCS 1039, Springer-Verlag*, 1996.

[15] R. Schroeppel. Hasty Pudding Cipher. http://www.cs.arizona.edu/rcs/hpc, 1998.

[16] Transaction Processing Performance Council. TPC Benchmarks. http://www.tpc.org/, 2005.

Appendix A: Differential Bounds for the Elastic Version of Rijndael with 152 Bit Block Size

As an example of how the output of the model was used to calculate the differential bounds for the elastic version of Rijndael, we include here the results from the case of Y containing three bytes. When Y is 3 bytes, the total block size is 152 bits (19 bytes) and 12 rounds are applied. We compute the bound for 11 rounds to accommodate the case when B does not contain a delta in round 1. Notice that a differential attack is not possible if each series of three rounds has a $exp \geq 54$ because this produces a $exp \geq 162$ across 9 rounds. States producing a three round bound ≤ 48 were evaluated through 4 or 5 rounds as needed. The resulting bounds for exp are indicated in Table 1. For simplicity, we do not list cases in the table with a three round $exp \leq 48$ and a four round exp which precludes a differential attack.

3 Round exp	4 Round exp	minimum 5 Round exp
30	54	144
42	60	126
42	84	132
42	108	not computed
48	66	132
$\geq$ 54	not computed	not computed

Table 1: **Minimum exp Across 4 and 5 Rounds when Y is 3 Bytes**

If a total $exp \geq 152$ occurs in 11 or fewer rounds, a differential attack cannot occur. To bound the value of exp, we consider what happens based on the number of 3 round sequences that have an $exp \geq 54$ and fill in the remaining rounds with the values from Table 1. Specifically, we view the 11 rounds as 3,4 and 5 round sequences such as (3 rounds, 4 rounds, 4 rounds). As we previously stated, three sequences of three rounds with $exp \geq 54$ in each sequence eliminates a differential attack. If two series of three rounds have an $exp \geq 54$, the minimum exp for 4 rounds in the remaining 5 rounds is 54; therefore, the total exp across 10 rounds is ≥ 162. If one series of three rounds has an $exp \geq 54$, there are two remaining series of 4 rounds, each with an $exp \geq 54$ and the total exp across 11 rounds is ≥ 162. If each series of 3 rounds has a $exp \leq 48$, then there is either a 5 round $exp \geq 126$, in which case any other 4 or 5 round exp from Table 1 will produce a 9 or 10 round $exp \geq 234$, or there is a 4 round $exp \geq 108$ and any two 3 round exp values combined with the 108 will produce a $exp \geq 168$ in 10 rounds. In all cases, the total exp exceeds 152. Thus the probability of a specific differential characteristic occurring is $< 2^{-152}$ when Y is 3 bytes, implying a differential attack is not possible for a 152 bit block size.

Appendix B: Diagram for Linear Cryptanalysis

The linear relationship between the round input and output for G and G' as defined in Section 4 is depicted in Figure 4. The variables are defined in Section 4.

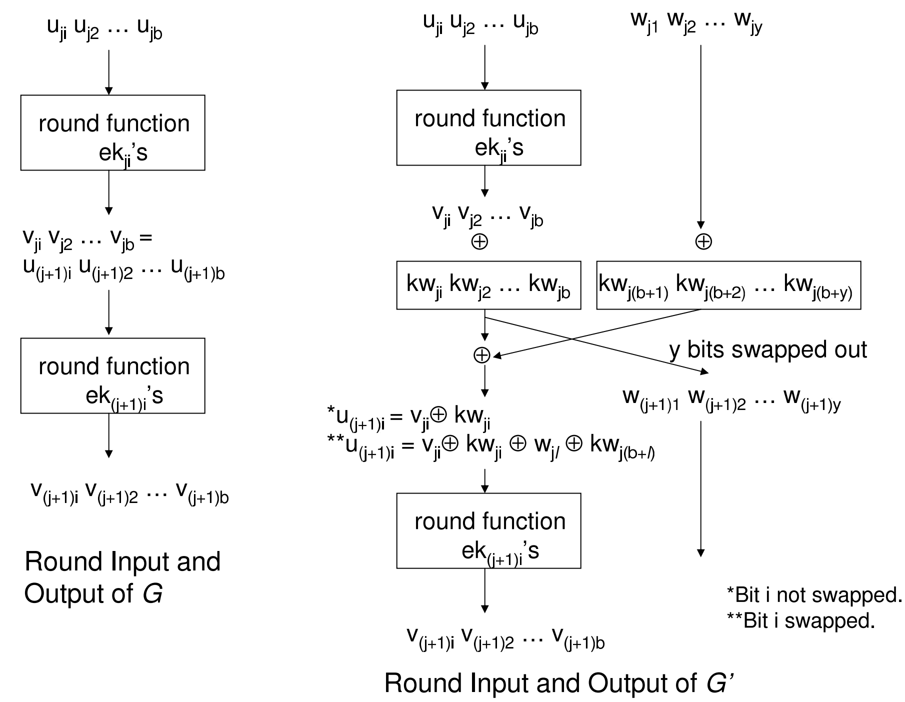

Figure 4: **Linear Relationship between j^{th} Round's Output and $(j+1)^{st}$ Round's Input**