Presented in SAC, August 1998

Initial Observations on Skipjack: Cryptanalysis of Skipjack-3XOR

Eli Biham* Alex Biryukov** Orr Dunkelman*** Eran Richardson† Adi Shamir‡

Abstract. Skipjack is the secret key encryption algorithm developed by the NSA for the Clipper chip and Fortezza PC card. It uses an 80-bit key, 128 table lookup operations, and 320 XOR operations to map a 64-bit plaintext into a 64-bit ciphertext in 32 rounds. This paper describes an efficient attack on a variant, which we call *Skipjack-3XOR* (Skipjack minus 3 XORs). The only difference between Skipjack and Skipjack-3XOR is the removal of 3 out of the 320 XOR operations. The attack uses the ciphertexts derived from about 500 plaintexts and its total running time is equivalent to about one million Skipjack encryptions, which can be carried out in seconds on a personal computer. We also present a new cryptographic tool, which we call the *Yoyo game*, and efficient attacks on Skipjack reduced to 16 rounds. We conclude that Skipjack does not have a conservative design with a large margin of safety.

Key words. Cryptanalysis, Skipjack, Yoyo Game, Clipper chip, Fortezza PC card.

1 Introduction

Skipjack is the secret key encryption algorithm developed by the NSA for the Clipper chip and Fortezza PC card. It was implemented in tamper-resistant hardware and its structure was kept secret since its introduction in 1993.

To increase confidence in the strength of Skipjack and the Clipper chip initiative, five well known cryptographers were assigned in 1993 to analyze Skipjack

* Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel, biham@cs.technion.ac.il, http://www.cs.technion.ac.il/~biham/.

** Applied Mathematics Department, Technion – Israel Institute of Technology, Haifa 32000, Israel.

*** Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel.

† Electrical Engineering Department, Technion – Israel Institute of Technology, Haifa 32000, Israel.

‡ Department of Applied Mathematics and Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel, shamir@wisdom.weizmann.ac.il.

and report their findings[4]. They investigated the strength of Skipjack using differential cryptanalysis[3] and other methods, and concentrated on reviewing NSA's design and evaluation process. They reported that Skipjack is a "representative of a family of encryption algorithms developed in 1980 as part of the NSA suite of "Type I" algorithms, suitable for protecting all levels of classified data. The specific algorithm, SKIPJACK, is intended to be used with sensitive but unclassified information." They concluded that "Skipjack is based on some of NSA's best technology" and quoted the head of the NSA evaluation team who confidently concluded "I believe that Skipjack can only be broken by brute force - there is no better way."

On June 24th, 1998, Skipjack was declassified, and its description was made public in the web site of NIST [7]. It uses an 80-bit key, $32 \cdot 4 = 128$ table lookup operations, and $32 \cdot 10 = 320$ XOR operations to map a 64-bit plaintext into a 64-bit ciphertext in 32 rounds.

This paper summarizes our initial analysis. We study the differential[3] and linear[6] properties of Skipjack, together with other observations on the design of Skipjack. Then, we use these observations to present a differential attack on Skipjack reduced to 16 rounds, using about 2^{22} chosen plaintexts and steps of analysis. Some of these results are based on important observations communicated to us by David Wagner [8].

We present a new cryptographic tool, which we call the *Yoyo game*, applied to Skipjack reduced to 16 rounds. This tool can be used to identify pairs satisfying a certain property, and be used as a tool for attacking Skipjack reduced to 16 rounds using only 2^{14} adaptive chosen plaintexts and ciphertexts and 2^{14} steps of analysis. This tool can also be used as a distinguisher to decide whether a given black box contains this variant of Skipjack, or a random permutation.

We then present the main result of this paper, which is an exceptionally simple attack on a 32-round variant, which we call *Skipjack-3XOR* (Skipjack minus 3 XORs). The only difference between the actual Skipjack and Skipjack-3XOR is the removal of 3 out of the 320 XOR operations. The attack uses the ciphertexts derived from about 500 plaintexts which are identical except for the second 16 bit word. Its total running time is equivalent to about one million Skipjack encryptions, which can be carried out in seconds on a personal computer. We thus believe that Skipjack does not have a conservative design with a large margin of safety.

This paper is organized as follows: In Section 2 we describe the structure of Skipjack, and the main variants that we analyze in this paper. In Section 3 we present useful observations on the design, which we later use in our analysis. In Section 4 we describe a differential attack on a 16-round variant of Skipjack. The Yoyo game and its applications are described in Section 5. Finally, in Section 6 we present our main attack on Skipjack-3XOR.

Rule A	Rule B
$w_1^{k+1} = G^k(w_1^k) \oplus w_4^k \oplus counter^k$	$w_1^{k+1} = w_4^k$
$w_2^{k+1} = G^k(w_1^k)$	$w_2^{k+1} = G^k(w_1^k)$
$w_3^{k+1} = w_2^k$	$w_3^{k+1} = w_1^k \oplus w_2^k \oplus counter^k$
$w_4^{k+1} = w_3^k$	$w_4^{k+1} = w_3^k$

Fig. 1. Rule A and Rule B.

2 Description of Skipjack

The published description of Skipjack characterizes the rounds as either Rule A or Rule B. Each round is described in the form of a linear feedback shift register with an additional non linear keyed G permutation. Rule B is basically the inverse of Rule A with minor positioning differences. Skipjack applies eight rounds of Rule A, followed by eight rounds of Rule B, followed by another eight rounds of Rule A, followed by another eight rounds of Rule B. The original definitions of Rule A and Rule B are given in Figure 1, where *counter* is the round number (in the range 1 to 32), G is a four-round Feistel permutation whose F function is defined as an 8x8-bit S box, called *F Table*, and each round of G is keyed by eight bits of the key. The key scheduling of Skipjack takes a 10-byte key, and uses four of them at a time to key each G permutation. The first four bytes are used to key the first G permutation, and each additional G permutation is keyed by the next four bytes cyclically.

The description becomes simpler (and the software implementation becomes more efficient) if we unroll the rounds, and keep the four elements in the shift register stationary. In this form the code is simply a sequence of alternate G operations and XOR operations of cyclically adjacent elements. In this representation the main difference between Rule A and Rule B is the direction in which the adjacent elements are XORed (left to right or right to left).

The XOR operations of Rule A and Rule B after round 8 and after round 24 (on the borders between Rule A and Rule B) are consecutive without application of the G permutation in between. In the unrolled description these XORs are of the form

$$
\begin{aligned}
&W2 = G(W2, subkey) && \texttt{-- Rule A}\\
&W1 = W1 \oplus W2 \oplus 8 &&\\
&W2 = W2 \oplus W1 \oplus 9 && \texttt{-- Rule B}\\
&W1 = G(W1, subkey) &&
\end{aligned}
$$

which is equivalent to exchanging the words $W1$ and $W2$, and leaving $W2$ as the original $W1 \oplus 1$:

$W2 = G(W2, subkey)$
exchange W1 and W2
$W1 = W1 \oplus W2 \oplus 8$
$W2 = W2 \oplus 1$
$W1 = G(W1, subkey)$

(the same situation occurs after round 24 with the round numbers 8 and 9 replaced by 24 and 25). Figure 2 describes this representation of Skipjack (only the first 16 rounds out of the 32 are listed; the next 16 rounds are identical except for the counter values).

Also, on the border between Rule B and Rule A (after round 16), there are two parallel applications of the G permutation on two different words, with no other linear mixing in between.

Note that Rule A mixes the output of the G permutation into the input of the next G permutation, while Rule B mixes the input of a G permutation into the output of the previous G permutation (similarly in decryption of Rule A), and thus during encryption Rule B rounds add little to the avalanche effect, and during decryption Rule A rounds add little to the avalanche effect.

In this paper we consider variants of Skipjack which are identical to the original version except for the removal of a few XOR operations. We use the name *Skipjack-*$(i_1, \ldots, i_k)$ to denote the variant in which the XOR operations mixing two data words at rounds $i_1, \ldots, i_k$ are removed, and the name *Skipjack-3XOR* as a more mnemonic name for Skipjack-(4,16,17), which is the main variant we attack. Note that the removal of these XOR operations does not remove the effect of any other operation (as could happen if we removed the XORs of the Feistel structure of G, which would eliminate the effect of the corresponding F tables).

3 Useful Observations

3.1 Observations Regarding the Key Schedule

The key schedule is cyclic in the sense that the same set of four bytes of the subkeys (entering a single G permutation) are repeated every five rounds, and there are only five such sets. In addition, the key bytes are divided into two sets: the even bytes and the odd bytes. The even bytes always enter the even rounds of the G permutation, while the odd bytes always enter the odd rounds of the G permutation.

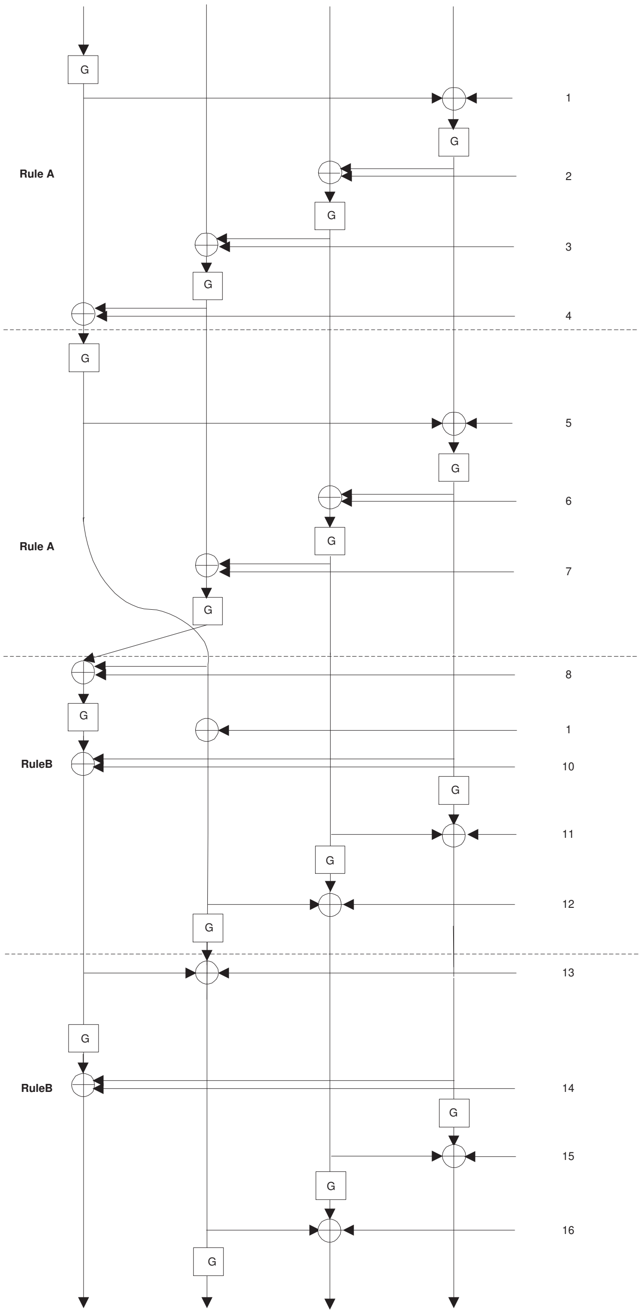

Fig. 2. Skipjack.

3.2 Decryption

As in most symmetric ciphers, decryption can be done using encryption with minor modifications. These modifications are (1) reordering the key bytes to $K^* = (cv_7, cv_6, ..., cv_0, cv_9, cv_8)$, (2) reversing the order of the round counters, and then (3) encrypting the reordered ciphertext $C^* = (cb_3, cb_2, cb_1, cb_0, cb_7, cb_6, cb_5, cb_4)$ gives the reordered plaintext $P^* = (pb_3, pb_2, pb_1, pb_0, pb_7, pb_6, pb_5, pb_4)$.

The mixings with the round numbers (counters) are often used to protect against related key attacks. In Skipjack, if these mixings are removed, the following stronger property would hold: Given a plaintext $P = (pb_0, pb_1, ..., pb_7)$, a key $K = (cv_0, ..., cv_9)$ and a ciphertext $C = (cb_0, ..., cb_7)$ such that $C = \text{Skipjack}_K(P)$, then decryption can be performed using encryption by $P^* = \text{Skipjack}_{K^*}(C^*)$, where $K^* = (cv_7, cv_6, ..., cv_0, cv_9, cv_8)$, $P^* = (pb_3, pb_2, pb_1, pb_0, pb_7, pb_6, pb_5, pb_4)$, and $C^* = (cb_3, cb_2, cb_1, cb_0, cb_7, cb_6, cb_5, cb_4)$.

This property could be used to reduce the complexity of exhaustive search of this Skipjack variant by a factor of almost 2 (26% of the key space rather than 50% in average) in a similar way to the complementation property of DES: Given the encrypted ciphertext $C1$ of some plaintext P, and the decrypted plaintext $C2$ of the related P^* under the same unknown key, perform trial encryptions with 60% of the keys K (three keys of each cycle of 5 keys of the rotation by two key bytes operations; efficient implementations first try two keys of each cycle, and only if all of them fail, they try the third keys of the cycles). For each of these keys compare the ciphertext to $C1$, and to $C2^*$ (i.e., $C2$ in which the bytes are reordered as above). If the comparison fails, the unknown key is neither K nor K^*. If it succeeds, we make two or three trial encryptions, and in case they succeed we found the key.

3.3 Complementation Properties of the G Permutation

The G permutation has $2^{16} - 1$ complementation properties: Let $G_{K0,K1,K2,K3}(x1, x2) = (y1, y2)$, where $K0, K1, K2, K3, x1, x2, y1, y2$ are all byte values, and let $d1, d2$ be two byte values. Then,

$$G_{K0 \oplus d1, K1 \oplus d2, K2 \oplus d1, K3 \oplus d2}(x1 \oplus d2, x2 \oplus d1) = (y1 \oplus d2, y2 \oplus d1).$$

G has exactly one fixpoint for every subkey (this was identified by Frank Gifford, and described in sci.crypt). Moreover, we observed that for every key and every value v of the form $(0, b)$ or $(b, 0)$ where 0 is a zero byte and b is an arbitrary byte value, G has exactly one value x for which $G(x) = x \oplus v$. It is unknown whether this property can aid in the analysis of Skipjack.

3.4 Differential and Linear Properties of the F Table

We generated the differential and linear distribution tables of the F table, and found that in the difference distribution table:

1. The maximal entry is 12 (while the average is 1).
2. 39.9% of the entries have non-zero values.
3. The value 0 appears in 39360 entries, 2 in 20559, 4 in 4855, 6 in 686, 8 in 69, 10 in 5, and 12 in 2 entries.
4. One-bit to one-bit differences are possible, such as $01_x \to 01_x$ (where the subscript x denotes a hexadecimal representation) with probability 2/256.

In the linear approximation table:

1. The maximal biases are 28 and -28 (i.e., probabilities of $1/2 + 28/256$ and $1/2 - 28/256$).
2. 89.3% of the entries have non-zero values.
3. The absolute value of the bias is 0 in 7005 entries, 2 in 12456, 4 in 11244, 6 in 9799, 8 in 7882, 10 in 6032, 12 in 4354, 14 in 2813, 16 in 1814, 18 in 1041, 20 in 567, 22 in 317, 24 in 154, 26 in 54, and 28 in 3 entries.
4. Unbalanced one-bit to one-bit linear approximations exist, such as $80_x \to 80_x$ with probability $1/2 + 20/256$.

3.5 Differential and Linear Properties of the G Permutation

Consider the F table, and let a and b be two byte values such that both $a \to b$ and $b \to a$ occur with a non-zero probability. We can prove that the best possible characteristic of G must be of the form: input difference: $(a, 0)$, output differences: $(0, b)$, with the intermediate differences $(a, 0) \to (a, 0) \to (a, b) \to (0, b) \to (0, b)$. There are 10778 pairs of such a and b, of which four have probability $48/2^{16} = 2^{-10.42}$. They are

1. $a = 52_x$, $b = f5_x$,
2. $a = f5_x$, $b = 52_x$,
3. $a = 77_x$, $b = 92_x$, and
4. $a = 92_x$, $b = 77_x$.

Most other characteristics of this form have probability 2^{-14} (6672 pairs) and 2^{-13} (3088 pairs). The remaining characteristics of this form have probabilities between 2^{-13} and $2^{-10.67}$.

Given a and b, there are additional characteristics with three active F tables (rather than only two), and for the above values of a and b the probabilities are between $2^{-15.4}$ and $2^{-15.83}$. These characteristics of G are $(0, b) \rightarrow (a, b)$ and $(a, b) \rightarrow (a, 0)$. We can combine these characteristics with the characteristics of the previous form and get cycles of three characteristics which have the form $(a, 0) \rightarrow (0, b) \rightarrow (a, b) \rightarrow (a, 0)$.

We studied the differential corresponding to these characteristics, and computed their exact probabilities by summing up the probabilities of all the characteristics with the same external differences. We found that the characteristic $(a, 0) \rightarrow (0, b)$ has the same probability as a differential and as a characteristic, as there are no other characteristics with the same external differences. $(0, b) \rightarrow (a, b)$ and $(a, b) \rightarrow (a, 0)$ with the same a and b as above have over a thousand small-probability counterparts with the same external differences, whose total probability is slightly smaller than the probabilities of the original characteristics. Thus, the probability of the differentials are almost twice that of the original characteristics (e.g., $137088/2^{32} = 2^{-14.94}$ instead of $73728/2^{32} = 2^{-15.83}$ in one of the cases).

We had also investigated other differentials of G. The characteristics we described with probability of around $2^{-10.42}$ (and other lower probability characteristics with zero differences in the first and fourth rounds of the G permutation) do not have any counterparts, and thus the corresponding differentials have the same probabilities as the characteristics. The best other differential we are aware of is $002A_x \rightarrow 0095_x$ with probability $2^{-14.715}$, and the best possible differential with the same input and output differences is $7F7F_x \rightarrow 7F7F_x$ with probability $2^{-15.84}$.

We next consider the case of linear cryptanalysis. As the characteristics are built in a similar way where XORs are replaced by duplications and duplications are replaced by XORs of the subsets of parity bits[1], we can apply the same technique for linear cryptanalysis. In this case we have 52736 possible pairs of a and b. The best linear characteristic of G is based on $a = b = 60_x$ and its probability is $1/2 + 2 \cdot 676/2^{16} = 1/2 + 2^{-5.6}$.

It is interesting to note that (due to its design) many criteria used in other ciphers are neither relevant to nor used in Skipjack. For example, a difference of one input bit in a DES S box cannot cause a difference of only one bit in its output, but there are many such instances in the F table of Skipjack.

Another observation is that due to the switchover from Rule A to Rule B iterations, the data between rounds 5 and 12 is very badly mixed. As mentioned earlier, on the border between the two rules (after rounds 8 and 24), the leftmost word is exchanged with word 2, and the new word 1 is XORed with the new word 2. We observed that the output of the G permutation in round 5 becomes the input to the G permutation in round 12, unaffected by other words (but

XORed with the fixed value $8 \oplus 9 = 1$). Thus, this word is not affected by any other word during 8 consecutive rounds. A similar property occurs in word 3 from round 7 to round 11, and in word 4 from round 6 to round 10. On the other hand, from round 5 to round 12 word 2 (renamed later to word 1) is affected several times by the other words, and the G permutation is applied to it several times, but it does not affect other words. Moreover, from round 13 to round 16, this word affects directly or indirectly only two of the three other words, and therefore, the input of the second word in round 5 never affects the fourth data word twelve rounds later.[6]

4 Cryptanalysis of Skipjack Reduced to 16 Rounds

4.1 Differential Cryptanalysis of Skipjack with Reduced Number of Rounds

The differential attack we describe here for 16-round Skipjack is considerably faster than exhaustive search. This attack is based on our original attack [2] with additional improvements based on Wagner's observations [8].

The best characteristics of 16-round Skipjack that we are aware of use the characteristics of the G permutation described above. The plaintext difference is $(a, 0, a, 0, 0, 0, 0, b)$ (where a, b and 0 are eight-bit values, and a, b are the values described in Section 3.5) and only six active G permutations (in which there are a total of 14 active F tables) are required to achieve the ciphertext difference $(0, b, 0, b, a, 0, 0, 0)$. There are four such characteristics with probabilities about $2^{-72.9}$. When we replace the characteristics by the corresponding differentials of G, the probability grows to about 2^{-71}. However, when we view the two G permutations in rounds 8 and 9 (unaffected by differences from other words) as one new permutation, its probability is about 2^{-16}, and thus the probability of the differential grows to about 2^{-58}.

Given the ciphertexts of many plaintext pairs with the difference $(a, 0, a, 0, 0, 0, 0, b)$, it is easy to identify and discard most of the wrong pairs in a 0R-attack. Such an attack requires about 2^{60} pairs. We observe that only a four-round characteristic of the first four rounds is required, with probability about 2^{-21}, and that when the characteristic holds, the truncated (word-wise) differences in rounds 5–16 are fixed. In this case we choose about 2^{22} chosen plaintext pairs, and can discard most of the wrong pairs, except for a fraction of 2^{-16} of them. Thus, about $2^6 = 64$ pairs remain.

Now we use a second observation that the same set of subkeys is used in the first and the 16th rounds. We try all the 2^{32} possible sets of subkeys and for each

[6] This property was found by Wagner[8].

remaining pair we encrypt the first round and verify that the characteristic of G holds, and decrypt the last round and verify whether the expected difference (i.e., the difference of the third ciphertext word) holds in the input of the last G permutation. The probability that a wrong set of subkeys does not discard a pair is $2^{-16} \cdot 2^{-10.4} = 2^{-26.4}$, and thus only the correct 32-bit subkey is expected to be proposed twice, by two different remaining pairs, and thus can be identified. This attack can be applied efficiently in 2^{16} steps for each analyzed pair, i.e., a total complexity of 2^{22} steps. Similar techniques (or even exhaustive search of the remaining 48 bits of the key) can complete the cryptanalysis.

4.2 Linear Cryptanalysis of Skipjack with Reduced Number of Rounds

Linear characteristics are built in a similar way where XORs are replaced by duplications and duplications are replaced by XORs of the subsets of parity bits[1]. As Rule A and Rule B differ essentially in this way, we can have similar analysis for linear cryptanalysis (except that we use linear characteristics rather than differentials). The probability of the best linear characteristic we found is about $1/2 + 2^{-35.5}$, and thus the attack seems to require more known plaintexts than the total number of possible plaintexts. However, this number can be reduced below 2^{64} by using shorter characteristics.

4.3 Modified Variants of Skipjack

Skipjack uses alternately eight rounds of Rule A and eight rounds of Rule B. In this section we investigate whether other mixing orders strengthen or weaken the cipher. A simple example of a modified design uses alternately four 'Rule A' rounds and four 'Rule B' rounds. We found an attack on this 16-round cipher which requires only about 2^{10} chosen plaintexts and about 2^{32} steps of analysis to find the subkey of round 3.

When Rule A rounds and Rule B rounds appear in reverse order (i.e., Rule B is applied first), and four rounds of each are applied consecutively, then only two pairs are required to find the last subkey.

These few examples indicate that the order of Rule A and Rule B rounds can have a major impact on the security of modified variants of Skipjack. Further study of modified variants will shed more light on Skipjack's design principles.

5 A New Cryptographic Tool: The Yoyo Game

Consider the first 16 rounds of Skipjack, and consider pairs of plaintexts $P = (w_1, w_2, w_3, w_4)$ and $P^* = (w_1^*, w_2^*, w_3^*, w_4^*)$ whose partial encryptions differ only in the second word in the input of round 5 (we will refer to it as the *property* from now on). As this word does not affect any other word until it becomes word 1 in round 12, the other three words have difference zero between rounds 5 and 12.

We next observe that given a pair with such a property, we can exchange the second words of the plaintexts (which cannot be equal if the property holds), and the new pair of plaintexts (w_1, w_2^*, w_3, w_4) and $(w_1^*, w_2, w_3^*, w_4^*)$ still satisfies the property, i.e., differs only in the second word in the input of round 5. Given the ciphertexts we can carry out a similar operation of exchanging words 1.

The Yoyo game starts by choosing an arbitrary pair of distinct plaintexts P_0 and P_0^*. The plaintexts are encrypted to C_0 and C_0^*. We exchange the first words of the two ciphertexts as described above, receiving C_1 and C_1^*, and decrypt them to get P_1, P_1^*. Now we exchange the second words of the plaintexts, receiving P_2 and P_2^*, and encrypt them to get C_2 and C_2^*. The Yoyo game repeats this forever.

In this game, whenever we start with a pair of plaintexts which satisfies the property, all the resultant pairs of encryptions must also satisfy the property, and if we start with a pair of plaintexts which does not satisfy the property, all the resultant encryptions cannot satisfy it.

It is easy to identify whether the pairs in a Yoyo game satisfy the above property, by verifying whether some of the pairs achieved in the game have a non-zero difference in the third word of the plaintexts or in the fourth word of the ciphertexts. If one of these differences is non-zero, the pair cannot satisfy the property. On the other hand, if the pair does not satisfy the property, there is only a probability of 2^{-16} that the next pair in the game has difference zero, and thus it is possible to stops games in which the property is not satisfied after only a few steps. If the game is not stopped within a few steps, we conclude with overwhelming probability that the property is satisfied.

This game can be used for several purposes. The first is to identify whether a given pair satisfies the above property, and to generate many additional pairs satisfying the property.

This can be used to attack Skipjack reduced to 16 rounds in just 2^{14} steps. For the sake of simplicity, we describe a suboptimal implementation with complexity 2^{17}. In this version we choose 2^{17} plaintexts whose third word is fixed. This set of plaintexts defines about 2^{33} possible pairs, of which about 2^{17} candidate pairs have difference zero in the fourth word of the ciphertexts, and of which about one or two pairs are expected to satisfy the property. Up to this point, this attack is similar to Wagner's attack on 16-round Skipjack [8]. We then use the Yoyo game to reduce the complexity of analysis considerably. We play the game for each of the 2^{17} candidate pairs, and within a few steps of the game discard all the pairs which do not satisfy the property. We are left with one pair which satisfies the property, and with several additional pairs generated during the Yoyo game which also satisfy the property. Using two or three of these pairs, we can analyze the last round of the cipher and find the unique subkey of the last round that satisfies all the requirements with complexity about 2^{16}. The rest of the key bytes can be found by similar techniques.

This game can also be used as a distinguisher which can decide whether an unknown encryption algorithm (given as an oracle) is Skipjack reduced to 16 rounds or a random permutation.

The above Yoyo game keeps three words with difference zero in each pair. We note that there is another (less useful) Yoyo game for Skipjack reduced to 14 rounds (specifically, rounds 2 to 15), which keeps only one word with difference zero. Consider pairs of encryptions $P = (w_1, w_2, w_3, w_4)$ and $P^* = (w_1^*, w_2^*, w_3^*, w_4^*)$ which have the same data at the leftmost word in the input of round 5. As this word is not affected by any other word until it becomes word 2 in round 12, we can conclude that both encryptions have the same data in word 2 after round 12. Given a pair with such an equality in the data, we can exchange the first word of the plaintexts, and the new pair of plaintexts (w_1^*, w_2, w_3, w_4) and $(w_1, w_2^*, w_3^*, w_4^*)$ still has the same property of equality at the input of round 5. Moreover, if the first words of the plaintexts are equal (i.e., $w_1 = w_1^*$ and thus exchanging them does nothing) we can exchange the second words (w_2 with w_2^*) and get the same property. If they are also equal, we can exchange w_3 with w_3^* and get the same property. If they are also equal, we exchange w_4 with w_4^*. However, if the property holds, this last case is impossible, as at least two words of the two plaintexts must be different. Given the ciphertexts we can carry out a similar operation of exchanging words 2. If words 2 are equal, exchange words 1, then words 4, and then words 3. Also in this case a difference of only one word ensures that the property is not satisfied. This Yoyo game is similar to the previous game, except for its modified exchange process, and it behaves similarly with respect to the new difference property.

6 Cryptanalysis of Skipjack-3XOR

In this section we analyze Skipjack-3XOR, which is identical to the original 32-round Skipjack except for the removal of the three XOR operations which mix 16-bit data words with their neighbors at rounds 4, 16 and 17. We show that this version is completely insecure, since it can be broken in one million steps using only about 500 chosen plaintexts.

The starting point of the attack is Wagner's observation[8] that the differential characteristic we used in the previous section can use truncated (i.e., word-wise) differences [5]. The attack uses the following characteristic of Skipjack-3XOR: For any 16-bit non-zero value a, the plaintext difference $(0, a, 0, 0)$ leads to the difference $(b, c, 0, 0)$ after round 16 with probability 1, which in turn leads to a difference $(d, 0, 0, 0)$ after round 28 with probability 2^{-16}, for some unspecified non-zero values b, c, and d. This difference leads to some difference $(e, f, g, 0)$ in the ciphertexts, for some e, f, and g.

The attack requires two pairs of plaintexts with such a differential behavior. To get them, encrypt $2^9 = 512$ distinct plaintexts which are identical except at their second word. They give rise to about $2^{18}/2 = 2^{17}$ pairs, and each pair has the required property with probability 2^{-16}. The two right pairs can be easily recognized since the two ciphertexts in each pair must be equal in their last 16 bits.

The basic steps of the attack are:

1. We know the input differences and the actual outputs of the 32nd G permutation. Each right pair yields a subset of about 2^{16} possible key bytes $cv_4, \ldots, cv_7$, and the intersection of the two subsets is likely to define these 32 key bits (almost) uniquely. This part can be implemented in about 2^{16} evaluations of G.
2. The 29th G permutation shares two key bytes cv_4, cv_5 with the 32nd G permutation, which are already known. 2^{16} possible combinations of the two key bytes cv_2, cv_3 and the inputs to the 30th G permutation in both pairs can be found. A careful implementation of this step requires a time complexity which is equivalent to 2^{17} evaluations of G.
3. For each of the 2^{16} combinations we still miss the key bytes cv_8, cv_9 entering the last two F tables in round 30, and the key bytes cv_0 and cv_1 entering the first two F tables in round 31. Together they are equivalent to a single G, which we call G'. In each right pair, the two encryptions have the same values in G'. We view both right pairs as a super pair of two G' evaluations, whose actual inputs and outputs are known. The analysis of G' takes about the equivalent of 2^9 G evaluations, and thus the total complexity is equivalent to about 2^{25} G evaluations.

Since each Skipjack encryption contains $2^5 = 32$ G evaluations, the total time complexity of this cryptanalytic attack is equivalent to about one million Skipjack encryptions, and can be carried out in seconds on a personal computer.

Acknowledgments

We are grateful to David Wagner, Lars Knudsen, and Matt Robshaw for sharing various beautiful observations and results with us. We are also grateful to Rivka Zur, the Technion CS secretary, for preparing Figure 2.

References

1. Eli Biham, *On Matsui's Linear Cryptanalysis*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'94, pp. 341–355, 1994.
2. Eli Biham, Alex Biryukov, Orr Dunkelman, Eran Richardson, Adi Shamir, *Initial Observations on the Skipjack Encryption Algorithm*, June 25, 1998, available at `http://www.cs.technion.ac.il/~biham/Reports/SkipJack/`.
3. Eli Biham, Adi Shamir, *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag, 1993.
4. Ernest F. Brickell, Dorothy E. Denning, Stephen T. Kent, David P. Maher, Walter Tuchman, *SKIPJACK Review, Interim Report, The SKIPJACK Algorithm*, July 28, 1993. Available at `http://www.austinlinks.com/Crypto/skipjack-review.html`.
5. Lars R. Knudsen, Thomas A. Berson, *Truncated differentials of SAFER-K64*, proceedings of Fast Software Encryption, Cambridge, Lecture Notes in Computer Science, pp. 15–26, 1996.
6. Mitsuru Matsui, *Linear Cryptanalysis Method for DES Cipher*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'93, pp. 386–397, 1993.
7. *Skipjack and KEA Algorithm Specifications*, Version 2.0, 29 May 1998. Available at the National Institute of Standards and Technology's web page, `http://csrc.nist.gov/encryption/skipjack-kea.htm`.
8. David Wagner, *Further Attacks on 16 Rounds of SkipJack*, private communication, July 1998.