Truncated Differentials and Skipjack

Lars R. Knudsen[1], M.J.B. Robshaw[2], and David Wagner[3]

[1] Department of Informatics, University of Bergen, N-5020 Bergen, Norway
larsr@ii.uib.no
[2] RSA Laboratories, 2955 Campus Drive, San Mateo, CA 94403, USA
matt@rsa.com
[3] University of California Berkeley, Soda Hall, Berkeley, CA 94720, USA
daw@cs.berkeley.edu

Abstract. We consider a range of attacks on reduced-round variants of the block cipher Skipjack. In particular we concentrate on the role of truncated differentials and consider what insight they give us into the design and long-term security of Skipjack. An attack on the full 32 rounds of Skipjack remains elusive. However we give attacks on the first 16 rounds of Skipjack that can efficiently recover the key with about 2^{17} chosen plaintexts and an attack on the middle sixteen rounds of Skipjack which recovers the secret key using only two chosen plaintexts. Several high-probability truncated differentials are presented the existence of which might best be described as surprising. Most notably, we show that the techniques used by Biham et al. can be presented in terms of truncated differentials and that there exists a 24-round truncated differential that holds with probability one.

1 Introduction

Skipjack is a 64-bit block cipher that is used in the Clipper Chip [11, 12] and was recently made public by the NSA [15, 16]. The length of the user-supplied key suggests that like other cryptographic proposals from the U.S. government [13, 14] the security level is intended to be 80 bits. Skipjack is a remarkably simple cipher and one interesting feature is the use of two different types of rounds. These are referred to as A-rounds and B-rounds and encryption with Skipjack consists of first applying eight A-rounds, then eight B-rounds, once again eight A-rounds and finally eight B-rounds.

The simplicity of Skipjack alone makes it an interesting cipher to study. However if we also recall the speculation and widespread distrust with which the cipher was first received [11, 12] then this once-secret cipher becomes particularly intriguing. In this paper we will consider some of the structural properties of Skipjack. In particular we note that the simple rounds of Skipjack seem to be particularly amenable to analysis using truncated differentials [7]. We will provide details of some particularly effective attacks on reduced-round versions of Skipjack and we will consider the applicability of these and other potentially more powerful attacks to an analysis of the full cipher.

A preliminary version of a report into the security of Skipjack was published on July 28, 1993 [9]. Written by five eminent cryptographers, the report reveals that while Skipjack was designed using techniques that date back more than forty years, Skipjack itself was initially designed in 1987. Since the cipher and design details were classified at the time of writing, the authors of the report were clearly restrained in what they could say. However, one phrase in the report is particularly interesting. There it is claimed that "[The design process] eliminated properties that could be indicative of vulnerabilities" [9]. In this paper we demonstrate that, in our opinion, this design goal was not attained. While we find no way to exploit the structural features that we highlight in a direct attack on the full Skipjack, we feel that the presence of these features could well be indicative of vulnerabilities. With more study, they could lead others towards an exploitable weakness in the cipher.

2 Description of Skipjack and other work

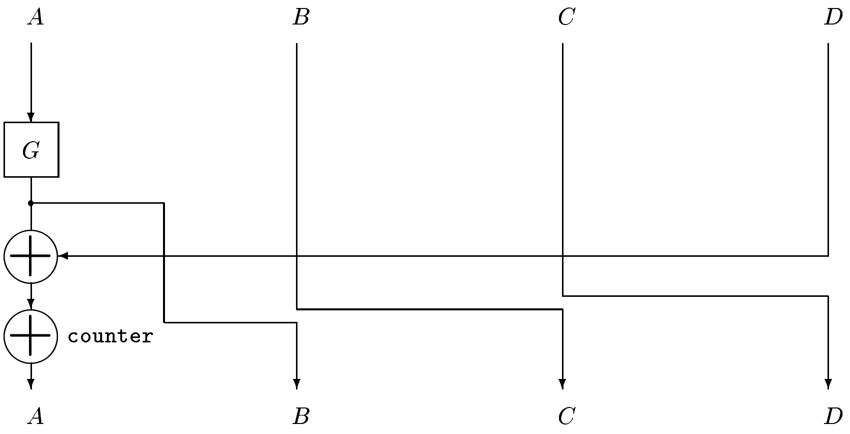
The 64-bit input block of Skipjack is split into four words of 16 bits. At the time of its initial design (1987) this approach was perhaps somewhat uncommon though RC2 [8] adopts a similar structure. In each round of Skipjack one of the words passes through a keyed permutation which we denote by G, and at most two words are modified during a single round. The function G has the structure of a four-round, byte-wise Feistel network. When needed, we will denote the round function (which uses a fixed, byte-wise substitution table S) by F. A counter, which is incremented at each round of encryption, is also used though it will be ignored throughout this paper since it has no cryptographic impact on our work. The rounds are illustrated in Figure 1.

The user-supplied key features during the G transformation. At each round four bytes of the 10 bytes of key material are used, with one byte being used at each step of the mini-Feistel network contained within G. If we denote the key by $k_0 \dots k_9$ then this key is simply replicated through the rounds, so that bytes $k_0, \dots, k_3$ are used in round one, bytes $k_4, \dots, k_7$ are used in round two, bytes k_8, k_9, k_0, k_1 are used in round three and so forth. We will sometimes write $G_{k_0 \dots k_3}$ to illustrate which key bytes are used in the G transformation.

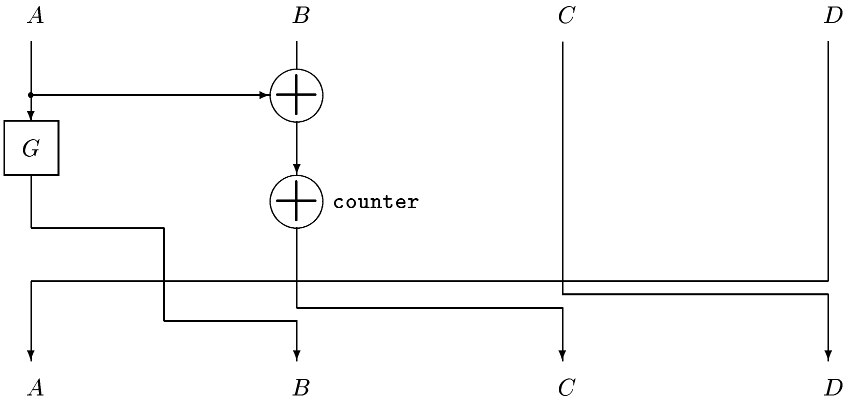
A first analysis by Biham et al. [1] studied some of the detailed properties of G and in particular some of the properties of the substitution table S. This provided a first description of some differential [6] and linear [10] cryptanalytic attacks on reduced-round versions of Skipjack. It was shown that reducing Skipjack to consist of the first 16 rounds (eight A-rounds followed by eight B-rounds) allowed one to mount a differential attack requiring about 2^{55} chosen plaintexts [4].

Independently of the authors of this paper, Biham et al. [2, 3] also considered the role of truncated differentials in Skipjack and some variants. All that is important for such attacks to be mounted is that the function G be a permutation. Further details about G (and therefore of the substitution box S) are

[4] It is important to note that this attack required that the key schedule be treated in a way that seems to conflicts with its intended use in the full cipher.



Skipjack A-round



Skipjack B-round

Fig. 1. The two rounds used in Skipjack. The counter is encrypted at each round and while it is included for completeness, it has no cryptanalytic significance with regards to the attacks in this paper.

immaterial. Most recently Biham et al. [5] derived attacks that are faster than exhaustive search for the key if Skipjack is reduced by at least one round. In this paper we consider alternative enhancements which offer interesting insights into the design of Skipjack. Currently these seem to be less effective than other attacks but we observe that there are opportunities for improvement and we outline promising avenues for further work that remain unexplored.

3 Truncated differentials of Skipjack

In a typical differential attack, the attacker chooses two plaintexts with a particular difference between them. During the encryption process the aim is to predict with some probability how the difference between these two quantities evolves. When the attacker is able to predict the difference towards the end of the encryption process with a sufficiently high probability, information about the user-supplied key can sometimes be derived.

When using truncated differentials, instead of trying to predict the evolution of some difference across an entire block, the cryptanalyst attempts to predict the difference across some fraction of this block. With Skipjack it is very natural to consider the difference across the four 16-bit words as we will now demonstrate. Let a, b, …, h denote any non-zero value to the difference[5] in a 16-bit word. We use r_A to denote an A-round and r_B to denote a B-round. One useful truncated differential characteristic allows us to cover the first 16 rounds of Skipjack:

$$(a, b, 0, c) \xrightarrow{8r_A} (e, e, 0, 0) \xrightarrow{8r_B} (g, h, f, 0), \tag{1}$$

The probability of the differential is 2^{-32} since $(a, b, 0, c) \xrightarrow{4r_A} (0, d, 0, 0)$ with probability 2^{-32}, $(0, d, 0, 0) \xrightarrow{4r_A} (e, e, 0, 0)$ always holds, and the last eight rounds of the characteristic $(e, e, 0, 0) \xrightarrow{8r_B} (g, h, f, 0)$ always holds. This differential will be useful to us in Section 4.1 where it is shown how to break the first 16 rounds of Skipjack with 2^{17} chosen plaintexts.

There are other interesting truncated differentials for Skipjack. The truncated differential (1) contains a truncated differential over eight B-rounds which holds with probability one. We found that there are at least two other truncated differentials over eight B-rounds which hold with the same probability. They are

$$(0, 0, a, 0) \xrightarrow{8r_B} (b, 0, c, d) \quad \text{and} \quad (0, a, 0, 0) \xrightarrow{8r_B} (0, b, c, d).$$

It is possible to add another four A-rounds to the latter differential while retaining the fact that the truncated differential holds with probability one. Thus, one gets the following twelve-round truncated differential with probability one

$$(0, a, 0, 0) \xrightarrow{8r_B} (0, b, c, d) \xrightarrow{4r_A} (h, h, f, g). \tag{2}$$

In Section 4.2 we will use this truncated differential to mount a particularly efficient truncated differential attack on the middle 16 rounds of Skipjack.

While a 12-round truncated differential with probability one seems remarkable enough, there is more and these results are described in Section 3.1. We also highlight some practical difficulties when using truncated differentials in Section 3.2 and we describe the semi-exhaustive search that we used to find these differentials in Section 3.3. We note here that the 16-round truncated differential (1) given above is indeed the best truncated differential for the first 16 rounds of Skipjack.

[5] While the most useful notion of difference can change depending on the cipher in question, for Skipjack we use bitwise exclusive-or.

3.1 Long truncated differentials

At least one truncated differential gives nontrivial information about the ciphertexts after 17 rounds of encryption. It goes through four A-rounds, eight B-rounds and five A-rounds and has the following form:

$$(0, a, 0, 0) \xrightarrow{4r_A} (b, b, 0, 0) \xrightarrow{8r_B} (c, d, e, 0) \xrightarrow{5r_A} (f, g, h, i),$$

where $f \neq g$ and h and i can take any values. A variant of Skipjack reduced to these 17 rounds can be distinguished from a randomly chosen permutation using only about $\sqrt{2} \cdot 2^8$ chosen plaintexts[6].

Even more remarkably there are truncated differentials which give non-trivial information about the ciphertexts after up to 24 rounds of encryption. It is interesting to compare the following 24-round truncated differential with the 24-round "impossible differential" of Biham et al. [5]. They are identical, though the differential described here will be explained in the classical top-down fashion.

First consider the following 12-round differential that also features in [5]. The words a, b, c, d, e can be arbitrary nonzero values.

$$(0, a, 0, 0) \xrightarrow{4r_A} (b, b, 0, 0) \xrightarrow{8r_B} (c, d, e, 0) \tag{3}$$

The differential can be concatenated with the following differential over 8 A-rounds and 4 B-rounds.

$$(c, d, e, 0) \xrightarrow{8r_A} (j, k, l, m) \xrightarrow{4r_B} (r, s, t, u) \tag{4}$$

If we are careful to track how the differential evolves, we are able to place conditions on different words of the differential even if they are identified as being non-zero. A pair of inputs have equal values in the fourth word, but different values in the other three. The conditions at each round of the evolution of truncated differential (4) are given in Table 1. Note, as an example, that after the second A-round $f \neq e$ since $f = c \oplus e$ and $c \neq 0$. Likewise, after the fourth A-round $(i, g) \neq (0, 0)$. To see this, note that in the preceding round $h \neq g$, since $b \neq 0$. But $h \neq g$ implies that $(h, g) \neq (0, 0)$ and $(i, g) \neq (0, 0)$ since $i = 0 \Leftrightarrow h = 0$.

We can show (see Table 1) that the three rightmost words at the end of the last 12 rounds of the 24-round truncated differential cannot all be zero. Suppose to the contrary that $w = 0$, $\gamma = 0$, and $\beta = 0$. This implies $k = 0 \Rightarrow v = 0 \Rightarrow p = 0 \Rightarrow u = 0 \Rightarrow m = 0$, and we have a contradiction since $(m, k) \neq (0, 0)$. Altogether, this yields a 24-round truncated differential, where the differences in the three rightmost words of the ciphertexts cannot all be zero.

[6] To see this choose a pool of different plaintexts with equal values in the first, third and fourth words. Compute the exclusive-or of the first two words of all ciphertexts and look for a match in these values. Such a match will not be found for the Skipjack variant, but for a randomly chosen permutation a match is found with probability 2^{-16}.

Round	Difference	Properties
	$(a, b, c, 0)$	a, b, c nonzero
A1:	(d, d, b, c)	b, c, d nonzero
A2:	(f, e, d, b)	e, d, b nonzero, $f \neq e$
A3:	(h, g, e, d)	e, d nonzero, $h \neq g$
A4:	(j, i, g, e)	$e \neq 0$, $(i, g) \neq (0, 0)$, $j \neq i$
A5:	(l, k, i, g)	$(k, i) \neq (0, 0)$, $(i, g) \neq (0, 0)$, $l \neq k$
A6:	(n, m, k, i)	$(m, k) \neq (0, 0)$
A7:	(q, p, m, k)	$(m, k) \neq (0, 0)$
A8:	(s, r, p, m)	$s = k \oplus r$
B1:	(m, t, k, p)	$(m, k) \neq (0, 0)$
B2:	(p, u, α, k)	$\alpha = m \oplus t$
B3:	(k, v, β, α)	$\beta = p \oplus u$
B4:	$(\alpha, w, \gamma, \beta)$	$\gamma = k \oplus v$

Table 1. The last 12 rounds of the 24-round truncated differential.

3.2 Important practical details

Before proceeding it is worth highlighting two important features of a truncated differential if we wish to use it directly in an attack.

FILTERING. After accumulating what might be a vast amount of chosen plaintext-ciphertext pairs in an attack, the cryptanalyst needs to throw away as much erroneous data (pairs that do not follow the differential as intended) as possible. This is done by filtering. With the truncated differentials we consider, the structure we use for filtering is the presence of a zero difference in some word. In the 16-round attack of Section 4.1, the expected difference in the ciphertexts is $(g, h, f, 0)$, which means that only pairs of ciphertexts with equal fourth words will be left after filtering. The more zeros in the expected output difference, the greater the number of wrong pairs that can be filtered before starting to extract key material.

COUNTING. The second feature that is important to consider is where, in some input and output difference, the non-zero differences lie. While some truncated differentials might initially appear to be useful to the cryptanalyst, it is not always possible to extract key information. One example is the following truncated differential $(0, a, b, 0) \xrightarrow{8r_A} (c, d, e, 0) \xrightarrow{8r_B} (0, f, g, h)$ which holds with probability 2^{-32}. When using this differential in an attack it passes over the first round of encryption with probability one and it is not possible to distinguish the correct first-round subkey from the wrong ones.

The semi-exhaustive search described in Section 3.3 was completed for truncated differentials of the full 32-round Skipjack. The search revealed several truncated differentials of probability 2^{-64}. However for all of these it seems impossible to search for keys in both the first and the last round as would be needed to directly mount an attack.

3.3 The search for truncated differentials

The semi-exhaustive search for truncated differentials was done in the following manner. Represent each 16-bit word in Skipjack by a single bit and so four bits will be used to represent the internal state of the four 16-bit words in Skipjack. A zero in the i^{th} bit indicates that there is no difference in the values of the i^{th} of the pair of data that follow the differential. The value one is used to indicate a non-zero difference which results from the two words having different values. We can then specify a set of rules that describes the "encryption" of the four words of difference through the A-rounds and through the B-rounds. It is easy in this way to do a complete search for any number of A-rounds and similarly for any number of B-rounds.

When combining A-rounds with B-rounds as required in Skipjack an extra "rule" is required. In the case where an A-round is followed by a B-round and where the output difference of the A-round has nonzero values assigned to the first two words, one needs to know if the difference in the first word is equal to the difference in the second word. This is of vital importance in the calculation of the probability of the differential in the B-round. However it is also easy to incorporate this consideration as a part of the search, since the differences in the two first output words from an A-round will be equal if, and only if, the fourth words of the inputs to the A-round are equal. Since no extra "rules" are needed in the transition from a B-round to an A-round, one can find truncated differentials for any number of A- and B-rounds.

In the following we report several findings of the search algorithm. In variants starting with eight B-rounds followed by eight A-rounds the following truncated differential

$$(0, a, 0, 0) \xrightarrow{4r_B} (0, b, c, 0) \xrightarrow{4r_B} (0, d, e, f) \xrightarrow{4r_A} (0, 0, g, h) \xrightarrow{4r_A} (0, 0, 0, i)$$

has component-wise probabilities of 1, 1, 2^{-16}, and 2^{-16} for the component four-round differentials respectively. Totally, the differential has probability 2^{-32} and a pair of texts following the differential can be found by taking all pairs generated from a pool of about 2^{17} chosen plaintext values. This makes it possible to effectively distinguish this variant of 16-round Skipjack from a random permutation using only around 2^{17} chosen plaintexts. For a random permutation two ciphertexts with equal values in the first three words (as in the above differential) occur with probability 2^{-48} and such a pair would normally be expected to occur after generating around $\sqrt{2} \cdot 2^{24}$ values.

The search revealed several truncated differentials for the full Skipjack with probability 2^{-64}. One example is the following differential where the words $a, \ldots, m$ can take any nonzero values.

$$(0, a, b, c) \xrightarrow{8r_A} (0, 0, 0, d) \xrightarrow{8r_B} (0, e, f, g) \xrightarrow{8r_A} (h, h, i, j) \xrightarrow{8r_B} (0, k, l, m),$$

This differential allows only for a very limited amount of filtering since only the form of the leftmost word of the ciphertext is restricted. (For all the 32-round truncated differentials with probability 2^{-64} that we have identified, only one of the words in the expected output differences is zero.) Furthermore, the leftmost word of the plaintext difference in all cases is zero, which means that key material cannot be extracted from analysis of the first round since all possible subkeys are equally likely. Thus, these differentials do not seem to be useful in an attack. However it is possible, at least theoretically, to mount an attack on the last 28 of the 32 rounds of Skipjack as we will show later.

4 Attacks using truncated differentials

4.1 The first sixteen rounds

We start with a truncated differential attack on the first 16 rounds of Skipjack that requires only 2^{17} chosen plaintexts and about 2^{34}–2^{49} time for the analysis. The range in the computational complexity comes from whether we treat the first and last round subkeys as independent or not.

We note that truncated differential cryptanalysis allows for significant improvements over an ordinary differential attack [1] due to two effects. First, the probability of the differential is sharply increased from $2^{-52.1}$, which was the probability of the differential [1] used in the conventional differential attack, to 2^{-32}. Second, the truncated differential allows us to extract more usable plaintext pairs from fewer chosen plaintexts because there is additional freedom in the construction of what are termed structures [6].

The attack uses the truncated differential (1) for the first 16 rounds of Skipjack. To generate suitable pairs for such a differential we choose 2^{17} plaintexts where the third words are fixed and obtain the corresponding ciphertexts. From these plaintexts one can form about 2^{33} pairs with the desired starting difference. With a high probability two right pairs will follow the truncated differential. Observing that the rightmost word has zero difference, we can immediately filter out many wrong pairs before moving on to the next stage of the analysis with 2^{17} pairs of data. In this second phase we will extract key material from the first and sixteenth rounds but the analysis will differ depending on whether the subkeys used in the outer two rounds are the same or different.

Independent subkeys. Here we treat the case where the subkeys used in the first and 16^{th} rounds are independently chosen. This seems more true to the intent of the Skipjack designers and is perhaps a better reflection of the style of attack that is needed for the full 32-round version of Skipjack.

Using the same truncated differential as before, each pair that survives filtering will suggest 2^{16} values for the four key bytes in the first round, and 2^{16} values for the four key bytes in the last round. It is possible to find these 2^{17} suggested values with offline work comparable to about 2^{17} G-box computations [4, 5]. (The trick is to use a precomputed table which, given differences y, z, allows us to find input x such that $F(x) \oplus F(x \oplus y) = z$ with one table lookup. We guess k_2, k_3, decrypt up by two layers of the G-box, and use the precomputed table to recover k_0, k_1, noting that z is known from the G-box input difference and y is known as a result of decrypting up two layers.) In total, we find that after filtering each remaining pair suggests about 2^{32} values for the eight key bytes used in the first and 16^{th} rounds. Naively we could simply count on those 64 key bits and look for a counter whose value exceeds one. By the birthday paradox, only about $2^{2\times 49}/2^{64+1} = 2^{33}$ wrong key values would remain, and each suggested value for the eight key bytes could be tested by exhaustive search over the remaining two unknown bytes. Thus, we could recover the key with about $(2^{17} \times 2^{32}) + (2^{33} \times 2^{16}) = 2^{50}$ work but the need for 2^{64} counters makes this approach totally impractical[7]. Instead we suggest the following technique.

Examine the plaintext pairs two at a time. For each two plaintext pairs use the calculation of G in the 16^{th} round to recover a list of possible values for the subkey. On average we expect to find about one possible subkey value. Similarly, the G computation in the first round is used to recover a possible value for another four key bytes. The suggested value for these eight key bytes can then be tested by exhaustive search over the remaining two unknown key bytes. There are about $2^{17} \cdot (2^{17} - 1)/2 \approx 2^{33}$ ways to choose two plaintext pairs, and each one requires about 2^{16} work, so with work equivalent to about 2^{49} encryptions we can recover the key.

The computational complexity could be reduced using alternative techniques if more texts are available. We can form 2^{37} plaintext pairs from 2^{19} chosen plaintexts, and count on the last-round subkey. About 2^{21} pairs survive filtering and so incrementing the counters requires work equivalent to about 2^{37} computations of G. The right counter will be suggested about $2^5 + 2^5$ times, whereas the wrong counter will be suggested 2^5 times on average (with standard deviation $2^{2.5}$). Only about 32 wrong counters will exceed their mean value by $2^{2.5} \approx 5.66$ standard deviations or more, so only about 33 values for the last-round subkey will survive. Similarly, we can find 33 possibilities for the first-round subkey with another 2^{37} computations of G, so after time equivalent to $(2^{37} + 2^{37})/16 = 2^{34}$ trial encryptions we can recover 33^2 possibilities for 64 key bits. Finally, those 33^2 possibilities can be tested with an exhaustive search over the remaining two unknown key bytes. The total computational complexity is equivalent to $2^{34} + 33^2 \times 2^{16} \approx 2^{34}$ trial encryptions with 2^{19} chosen plaintexts and 2^{32} space.

Dependent subkeys. When the subkeys used in the first and 16^{th} rounds are the same[8], several optimizations may be applied to the truncated differential attack. In this case, the total amount of offline work required for the attack is roughly comparable to that needed for 2^{34} offline trial encryptions.

4.2 The middle sixteen rounds

It is interesting to observe that there is a very efficient way to break the middle 16 rounds of Skipjack, i.e. a version of Skipjack consisting of eight B-rounds

[7] Space requirements can be reduced to about $2^{49} \times 8 = 2^{52}$ bytes by using a hash table or sorted list to store the suggested key values, but this is still too large.

[8] This holds for Skipjack. However, we feel that this is somewhat artificial since it is highly likely that any designers of such a Skipjack variant would change the key schedule to avoid this eventuality.

followed by eight A-rounds. Of course Skipjack has many more rounds than the sixteen we are attacking here, but our work is interesting for two reasons.

First it demonstrates that there is an asymmetry in how the A-rounds and the B-rounds might combine together to resist the attacker. This might help provide some insight into the design rationale behind Skipjack. Second, the attack outlined makes use of the structure of the G computation, and most importantly, of the internal Feistel structure. As in the earlier attacks, the S-box itself is completely immaterial to our discussions, but the byte-wise nature of the G computation provides a real benefit to the attacker. It is possible that attacks depending on the word-oriented structure of Skipjack could be aided by also considering the byte-oriented structure of the G computation. As a demonstration of this, we show that with two chosen texts, we can break this reduced cipher with work equivalent to about 2^{47} trial encryptions; with three chosen texts, the complexity of the attack drops to about 2^{30} encryptions. This is surprisingly close to Skipjack's unicity distance (1.25 texts).

We shall number the rounds from 1 to 16, so that the first round uses $k_0, \ldots, k_3$ and so on. In this attack, we use the 12-round truncated differential (2) of probability one to cover rounds 1 to 12 of the reduced cipher.

First we obtain n independent pairs following the truncated differential by making $n + 1$ chosen-plaintext queries[9] with the first, third, and fourth words of the input fixed. The rest of this section describes how to analyze those n pairs. We will describe our attack in general terms, leaving the number n of pairs unspecified until the end. Afterwards, we will optimize over n to obtain the best possible results.

The analysis consists of seven phases. In each phase, we recover some portion of the key material, either by guessing or by deriving it from known quantities inside the cipher. We describe each of the seven phases in turn.

1. Guess k_0, …, k_3. For each pair, peel off the 16^{th} round to learn the value of h that this key guess suggests.
2. Recover k_9. A naive approach is to simply guess k_9; reversing three layers of the computation of G in round 13 (using k_1, k_0, k_9) will give the right half (low byte) of h in each pair if our guess for $k_{9\ldots3}$ was correct. This gives a filtering condition on $8n$ bits. In practice, this can be implemented efficiently using a precomputed lookup table; see Section 4.1 or [4, 5] for more details. With proper implementation, the work factor of this phase will be about 2^{32}, and we expect 2^{40-8n} values of $k_{9\ldots3}$ to remain.
3. Recover k_8. We can use the same technique as in the second phase, this time reversing a fourth layer of the G transformation in round 13. We predict that the exclusive-or of the values obtained should be the same as the left half (high byte) of h in each pair if our guess was correct. This gives a filtering

[9] One could use structures to obtain n pairs from $\sqrt{2n} + 1$ queries, but the resulting pairs would not be independent, and we do not expect the extra "dependent" pairs to provide any useful extra information. Furthermore, typically we only need $n = 2$ pairs, so the difference would be negligible in any case.

condition on $8n$ bits, so 2^{48-16n} possibilities for $k_8, \ldots, k_3$ will remain. With proper implementation, this phase takes about 2^{40-8n} work.

4. Guess k_4 and k_5. Now decrypt through the computation of G in round 14 (using $k_2, \ldots, k_5$) to learn g. This will suggest 2^{64-16n} values for $k_8, \ldots, k_5$, with a similar work factor.
5. Recover k_7. The outputs of the G transformation in round 10 are now known, and the inputs have known difference a. We can decrypt two layers of G in round 10 (using k_8 and k_9), and then derive k_7 (which is used in the next layer) with a precomputed lookup table, as above. With proper implementation, this phase takes 2^{64-16n} work, and we expect that about 2^{72-24n} possibilities for $k_7, \ldots, k_5$ will remain at the conclusion of this phase.
6. Recover k_6. Complete the analysis of the computation of G in round 10 by deriving k_6 from its known outputs and its known input difference a. With proper implementation, this phase takes 2^{72-24n} simple operations, and about 2^{80-32n} suggested values for the entire key $k_0, \ldots, k_9$ will be left.
7. Check suggested values. We can check each suggested value for the key in any of a number of ways. One simple way is to do a full trial decryption on a few of the texts. Alternately, one could encrypt through the G transformations in rounds two, three, and six to check the result against the known input to G in round 10. This will require only four G computations and thus can be quite a bit faster than a full trial decryption. We expect that this final phase will quickly eliminate all incorrect keys.

The work required is about

$$2^{32} + 2^{32} + 2^{40-8n} + 2^{64-16n} + 2^{64-16n} + 2^{72-24n} + 4 \times 2^{80-32n}$$

simple operations. For $n = 1$ this gives 2^{51} steps and 2^{34} steps for $n = 2$. Of course, each step requires just a single G computation (often quite a bit less), so this is equivalent to about 2^{47} (respectively 2^{30}) trial encryptions. The result is a very sharp attack against the middle 16 rounds of Skipjack.

4.3 The last twenty-eight rounds

In this section we consider Skipjack reduced to the last 28 rounds and the following 28-round differential:

$$(a, b, 0, c) \xrightarrow{4r_A} (d, e, 0, 0) \xrightarrow{8r_B} (f, g, 0, h) \xrightarrow{8r_A} (i, i, 0, 0) \xrightarrow{8r_B} (j, k, l, 0),$$

where $(a, b, 0, c) \longrightarrow (d, e, 0, 0)$ is a four-round differential that starts in the fifth round, ends in the eighth round, and holds with probability 2^{-16}. The following eight-round differential has probability 2^{-16}, the next has probability 2^{-32}, and the final eight-round differential has probability 1. This gives a truncated differential over the last 28 rounds of Skipjack which holds with probability 2^{-64}.

To start the attack, choose 2^{41} plaintexts where the values of the third words are fixed to some arbitrary value. From these plaintexts we can form about 2^{81} pairs of which 2^{17} will be expected to follow the specified differential. Using the rightmost word of the ciphertexts we can filter out wrong pairs, leaving 2^{65} pairs. The extraction of key material that follows is similar to that given in Section 4.1.

INDEPENDENT SUBKEYS. First we assume that the keys in the first round of the differential (the fifth round of Skipjack) are independent of the keys in the last round. For each surviving pair we check which keys in the last round result in a difference that follows the differential after decryption by one round. About 2^{16} values of the 32-bit key will be suggested by this test for each pair. Similarly, for each surviving pair we check which keys result in differences that follow the differential after encryption by one round. With an efficient implementation, the suggested key values can be found in time comparable to 2^{17} evaluations of G (see Section 4.1 or [4, 5]). Overall we will find that $2^{65+16+16} = 2^{97}$ values for 64 bits of key material will be suggested. The expected value of the counter for a wrong value of the key is 2^{33}, whereas the expected value of the counter for the correct value of the key will be $2^{33} + 2^{17}$ since each of the 2^{17} right pairs will include the correct key value among the set of 2^{32} values suggested. This would mean that with a high probability the correct value of the key is among the 16% most suggested values. The total time for the analysis stage of this attack amounts to $2^{65+17} = 2^{82}$ G computations, a work effort that is equivalent to about 2^{77} encryptions. Thus, this attack is just faster than an exhaustive search for the key but the work effort required and the need for 2^{64} counters makes the attack totally impractical.

DEPENDENT SUBKEYS. If we assume that this reduced-round variant of Skipjack uses the key schedule specified in Skipjack then the attack will improve. The subkeys used in the fourth round are key bytes k_2, k_3, k_4, and k_5. The subkeys used in the last round are key bytes k_4, k_5, k_6, and k_7. The two sets of subkeys have a total entropy of only 48 bits. When taking this into account analysis of the data will suggest $2^{65+16} = 2^{81}$ values for a 48-bit key. The rest of the analysis is the same but the memory requirements have been reduced to 2^{48} counters.

We anticipate that similar attacks on Skipjack with fewer than 28 rounds will be much more efficient and that they can be used to find more information about the secret key. Furthermore, it might be possible to attack versions of Skipjack by counting on 64 key bits when the subkeys in the first two rounds and the last two rounds together have an entropy of 64 bits. We note that such a fortuitous key-scheduling coincidence occurs in the full 32-round Skipjack cipher.

5 Boomerang attacks

Here we consider the feasibility of boomerang attacks [17] on reduced-round variants of Skipjack. Boomerang attacks may be considered to be a close relative of miss-in-the-middle attacks [5], although these techniques were developed independently. Boomerang attacks on Skipjack are interesting because they allow us to improve on some of the existing miss-in-the-middle attacks by a factor of $2^{3.5}$–$2^{8.5}$. However, miss-in-the-middle attacks currently penetrate more rounds of Skipjack than boomerang attacks.

Boomerang attacks are chosen-plaintext, adaptive chosen-ciphertext attacks that work from the outside in. They use differential techniques to create a quartet structure inside the cipher by working from both ends of a cipher (the plaintext and ciphertext inputs) towards the middle. This quartet consists of four plaintexts P, P', Q, Q', along with their respective ciphertexts C, C', D, D' chosen as follows. We use a truncated differential $\Delta \to \Delta^*$ for the first half of the cipher, as well as the truncated differential $\nabla \to \nabla^*$ for the inverse of the last half of the cipher. The cryptanalyst picks P, P' so that $P \oplus P' \in \Delta$, encrypts to obtain C, C', then picks D, D' so that $C \oplus D \in \nabla$ and $C' \oplus D' \in \nabla$. The cryptanalyst then asks for the decryption of D, D' to obtain Q, Q'. We hope that the pair P, P' follows the differential $\Delta \to \Delta^*$, and that the pair C, D and the pair C', D' both follow the differential $\nabla \to \nabla^*$. If so, we have a quartet structure halfway through the cipher. If we have chosen Δ^*, ∇^* well, with good probability we obtain a difference of Δ^* halfway through the decryption of D, D', which lets us cover the remainder of the decryption with the backward differential $\Delta^* \to \Delta$. As a result, in a right quartet we will have the recognizable condition $Q \oplus Q' \in \Delta$. Many details have been omitted; a full description of the boomerang attack may be found in [17].

5.1 The middle twenty-four rounds

Consider a simplified 24-round Skipjack variant obtained by deleting four rounds from both ends of the real Skipjack cipher. This variant is intended to be relatively representative of a Skipjack cipher weakened to 24 rounds, in that it retains the symmetry between encryption and decryption.

Observe that there is a truncated differential of probability one through four A-rounds and eight B-rounds: $\Delta = (0, a, 0, 0) \xrightarrow{4r_A} (b, b, 0, 0) \xrightarrow{8r_B} (c, d, e, 0) = \Delta^*$. Due to the fact that the structure of an A-round is almost the inverse of the structure of a B-round, we also obtain a truncated differential of probability one for decryption through four B-rounds and eight A-rounds, specifically $\nabla = (f, 0, 0, 0) \xrightarrow{4r_B^{-1}} (g, g, 0, 0) \xrightarrow{8r_A^{-1}} (i, h, 0, j) = \nabla^*$. (Here $a, b, \ldots, j$ can take on any non-zero value.) Finally, we use the backward differential $\Delta^* \to \Delta$ of probability 2^{-32} for decrypting through the first half of the cipher. This gives a success probability[10] of $\Pr[\Delta \to \Delta^*] \times \Pr[\nabla \to \nabla^*]^2 \times 2^{-16}(1 - 2^{-16})^2 \times \Pr[\Delta^* \to \Delta] = 1 \times 1^2 \times 2^{-16}(1 - 2^{-16})^2 \times 2^{-32} \approx 2^{-48}$.

To mount a boomerang attack first construct a plaintext pair P, P' with $P \oplus P' \in \Delta$. Denote the ciphertexts C, C'. Next obtain 2^{16} ciphertexts D by varying the first word in C, and in a similar manner obtain 2^{16} ciphertexts D'

[10] Here the factor of 2^{-16} comes from the requirement that we get a difference of Δ^* halfway through the decryption of D, D', which happens when the fourth words of the two ∇^* differences are equal. In other words, if the ∇^*-difference is $(i, h, 0, j)$ in the C, D pair and $(i', h', 0, j')$ in the C', D' pair, we require that $j = j'$ so that $(i, h, 0, j) \oplus (i', h', 0, j') \oplus (c, d, e, 0)$ will take the form of a Δ^* truncated difference. Finally, one must add a correction factor of $(1 - 2^{-16})^2$, because the differential $\Delta^* \to \Delta$ is not valid when $i \oplus i' \oplus c = 0$ or $h \oplus h' \oplus d = 0$.

by modifying C'. Note that the truncated differentials $\nabla \to \nabla^*$ for C, D and $\nabla \to \nabla^*$ for C', D' are simultaneously obeyed, so we get 2^{32} possible quartets, of which $2^{32} \times 2^{-16}(1 - 2^{-16})^2 \approx 2^{16}$ have a difference of the form Δ^* halfway through the decryptions of D, D'.

Each structure of 2^{17} texts contains a right quartet with probability $2^{16} \times 2^{-32} = 2^{-16}$. Right quartets can be recognized by the difference Δ in the plaintexts Q, Q'. This allows us to filter out all but 2^{-48} of the wrong quartets and after repeating the attack 2^{16} times, we expect to obtain one right quartet and one wrong quartet.

To reduce the number of texts required we can choose $2^{8.5}$ plaintexts by varying the second word and holding the first, third, and fourth words fixed; then for each of the $2^{8.5}$ resulting ciphertexts, we generate 2^{16} more variant ciphertexts and decrypt. Each pair of plaintexts then gives a structure, so 2^{16} structures can be obtained from this pack of $2^{8.5}$ plaintexts, and thus we expect to see the first right quartet after only $2^{24.5}$ chosen texts.

While we cannot recover key information from this 24-round version of Skipjack using these techniques, we are able to distinguish this version from a random cipher with about $2^{8.5}$–$2^{9.5}$ chosen plaintexts and $2^{24.5}$–$2^{25.5}$ chosen ciphertexts. The same ideas can be applied to the inverse cipher to get a similar attack that uses $2^{24.5}$–$2^{25.5}$ chosen ciphertexts and $2^{8.5}$–$2^{9.5}$ chosen plaintexts.

5.2 The middle twenty-five rounds

Consider a Skipjack variant obtained by deleting the first three and last four rounds from the real Skipjack. We can use $2^{34.5}$ chosen texts to break 25 rounds of Skipjack with a $2^{61.5}$ work factor. One can use structures to bypass the first round subkey: vary the first and fourth words, and hold the middle two words fixed. With $2^{18.5}$ such plaintexts, one expects to find 2^{20} pairs of plaintexts which satisfy the desired relationship after the first round. After another $2^{34.5}$ chosen ciphertexts, one should find about 16 right quartets.

We then guess the first round subkey, and peel off the first round, checking for right quartets in the same way as in our 24-round attack. In this way, for each guess at the subkey we expect only about 16 of the wrong quartets to survive the filtering phase. This allows us to distinguish a right guess at the first round subkey from a wrong guess with good probability. In the former case 32 quartets will survive the filtering phase and in the latter only 16 quartets are expected to survive, which is a difference of four standard deviations. The analysis procedure can be performed with about $2^{34.5} \times 2^{32} = 2^{66.5}$ computations of G, which is a workload roughly equivalent to $2^{61.5}$ trial encryptions. In all, the attack recovers 32 key bits after $2^{61.5}$ work; the remaining 48 key bits can be found by trial decryption.

5.3 Comparison with miss-in-the-middle attacks

It is interesting to compare the complexity of boomerang attacks to Biham et al.'s miss-in-the-middle attacks [5] on the same reduced-round variants of Skipjack.

For 24 rounds, a boomerang attack needs $2^{24.5}$–$2^{25.5}$ chosen texts to distinguish the cipher from a random permutation, whereas the miss-in-the-middle attack needs 2^{33}–2^{35} chosen texts. For 25 rounds, our boomerang attack uses $2^{34.5}$ texts and $2^{61.5}$ work to recover the key, whereas the miss-in-the-middle attack [5] uses 2^{38} chosen texts and 2^{48} work. With regards to the data requirements, it appears that boomerang attacks compare favorably to miss-in-the-middle attacks for these reduced-round variants of Skipjack. However Biham et al. have demonstrated that miss-in-the-middle attacks can be used to analyze 31 rounds of Skipjack whereas boomerang attacks are currently restricted to no more than 25 rounds.

The boomerang attacks were aided by the fact that the $(4A, 8B)$ round structure (as found in the first half of the 24-round cipher) is weaker against truncated differentials in the encryption direction than in the decryption direction, while the $(8A, 4B)$ structure is weaker in the reverse direction. This property makes it easy to probe the $(4A, 8B, 8A, 4B)$ cipher from both ends at once with a boomerang attack. We suspect for similar reasons that a $(16B, 16A)$ structure might be easier to analyze with boomerang techniques than a $(16A, 16B)$ structure, which suggests that the ordering of the A-rounds and B-rounds may be quite important to the security of the Skipjack cipher.

6 Conclusion

In this paper we have described several interesting truncated differentials for Skipjack. These can be used in a variety of attacks, including particularly efficient attacks on reduced-round versions of Skipjack. The existence of such attacks and the effectiveness of truncated differentials demonstrates that Skipjack has unusual and surprising structural features. We also demonstrate the effectiveness of boomerang attacks on Skipjack. While they cannot be extended to attack 31 rounds of Skipjack like miss-in-the-middle attacks, for those reduced-round versions of Skipjack that can be compromised using both techniques, boomerang attacks are typically more effective than miss-in-the-middle attacks. We feel that attempts to extend existing boomerang attacks to more rounds could lead to more efficient attacks on Skipjack than are currently available. We leave it as a challenge to use our findings to find more efficient attacks on 16- to 31-round variants of Skipjack. Currently, an attack on the full 32 rounds of the cipher (other than by a brute force search for the key) remains elusive.

References

1. E. Biham, A. Biryukov, O. Dunkelmann, E. Richardson, A. Shamir. Initial Observations on the Skipjack Encryption Algorithm. June 25, 1998. Available at `http://www.cs.technion.ac.il/~biham/Reports/SkipJack/`.
2. E. Biham, A. Biryukov, O. Dunkelmann, E. Richardson, and A. Shamir. Cryptanalysis of Skipjack-3XOR in 2^{20} time and using 2^9 chosen plaintexts. July 2, 1998. Available at `http://www.cs.technion.ac.il/~biham/Reports/SkipJack/`.

3. E. Biham, A. Biryukov, O. Dunkelmann, E. Richardson, and A. Shamir. Cryptanalysis of Skipjack-4XOR. June 30, 1998. Available at `http://www.cs.technion.ac.il/~biham/Reports/SkipJack/`.
4. E. Biham, A. Biryukov, and A. Shamir. Initial Observations on Skipjack: Cryptanalysis of Skipjack-3XOR. SAC'98. Available at `http://www.cs.technion.ac.il/~biham/Reports/SkipJack/`.
5. E. Biham, A. Biryukov, and A. Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In J. Stern, editor, *Advances in Cryptology - Eurocrypt '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23, 1999. Springer Verlag. Also available at `http://www.cs.technion.ac.il/~biham/Reports/SkipJack/`.
6. E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard.* Springer-Verlag, New York, 1993.
7. L.R. Knudsen. Applications of higher order differentials and partial differentials. In B. Preneel, editor, *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211, 1995. Springer Verlag.
8. L.R. Knudsen, V. Rijmen, R.L. Rivest, and M.J.B. Robshaw. On the design and security of RC2. In S. Vaudenay, editor, *Fast Software Encryption*, volume 1372 of *Lecture Notes in Computer Science*, pages 206–219, 1998. Springer Verlag.
9. E.F. Brickell, D.E. Denning, S.T. Kent, D.P. Maher, and W. Tuchman. Skipjack Review Interim Report. July 28, 1993.
10. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology - Eurocrypt '93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397, 1993. Springer Verlag.
11. B. Schneier and D. Banisar. *The Electronic Privacy Papers.* John Wiley & Sons, 1997.
12. W. Diffie and S. Landau. *Privacy on the Line.* MIT Press, 1998.
13. National Institute of Standards and Technology (NIST). FIPS Publication 186: Digital Signature Standard (DSS). May 1994.
14. National Institute of Standards and Technology (NIST). FIPS Publication 180-1: Secure Hash Standard (SHS). May 1994.
15. National Security Agency. Skipjack and KEA algorithm specifications. May 1998. Available at `http://csrc.ncsl.nist.gov/encryption/skipjack-1.pdf`.
16. National Security Agency. NSA Releases Fortezza Algorithms, Press Release, June 24, 1998. Available at `http://csrc.ncsl.nist.gov/encryption/nsa-press.pdf`.
17. D. Wagner. The boomerang attack. In L. Knudsen, editor, *Fast Software Encryption*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170, 1999. Springer Verlag.