Linear Cryptanalysis of Reduced Round Serpent[*] NES/DOC/TEC/WP3/008/1

Eli Biham[†] Orr Dunkelman[‡] Nathan Keller[§]

Abstract

Serpent is one of the 5 AES finalists. In this paper we present a 9-round linear approximation for Serpent with probability of $1/2 + 2^{-52}$. We use a variant of it to devise a linear attack on 10-round Serpent (with all key lengths) whose data complexity is 2^{118}, and whose running time is equivalent to the time of 2^{88} memory accesses. The main result of this paper uses another variant of this approximation, which is the first known attack against an 11-round Serpent with 256-bit keys. This attack requires 2^{118} known plaintexts and its running time is equivalent to the time of 2^{214} memory accesses.

1 Introduction

Serpent [1] is a block cipher designed by Ross Anderson, Eli Biham and Lars Knudsen as a candidate for the Advanced Encryption Standard (AES) [11]. It was selected to be among the five finalists.

In [5], a modified variant of Serpent in which the linear transformation was modified into a permutation was analyzed. The permutation allows one active S box to activate one S box in the consecutive round, a property that does not occur in Serpent. Thus, it is not surprising that this variant is much weaker than Serpent, and that it can be attacked with up to 34 rounds.

In [7], the amplified boomerang attack was applied to 9-round Serpent with 256-bit keys. The attack is based on building a 7-round distinguisher for Serpent

[*]The work described in this paper has been supported by the European Commission through the IST Programme under Contract IST-1999-12324.

[†]Computer Science department, Technion - Israel Institute of Technology, Haifa 32000, Israel, biham@cs.technion.ac.il, http://www.cs.technion.ac.il/~biham/.

[‡]Computer Science department, Technion - Israel Institute of Technology, Haifa 32000, Israel, orrd@cs.technion.ac.il, http://vipe.technion.ac.il/~orrd/me/.

[§]Mathematics department, Technion - Israel Institute of Technology, Haifa 32000, Israel, nkeller@tx.technion.ac.il.

and using it to attack up to 9 rounds of Serpent. The distinguisher is based on 4-round differential characteristic in rounds 1–4, and 3-round characteristic in rounds 5–7.

In [4], the rectangle attack was introduced and applied to 10-round Serpent with 256-bit keys. An attack on 7-round Serpent with 128-bit keys was also presented, and the best known attack on 8-round Serpent with 192-bit keys was presented as well. These attacks rely on the best known 3-round, 4-round, 5-round and 6-round differential characteristics of Serpent, which were also presented.

Linear cryptanalysis was first suggested by Matsui [9, 10, 8]. The attack is based on the bias of correlation between the parity of subset of the input bits to the parity of subset of the output bits. Linear cryptanalysis is applicable to many block ciphers, and for some ciphers (like DES) the best known attack is based on linear cryptanalysis.

In this paper we present the best known 9-round linear approximation of Serpent which has a probability of $1/2 + 2^{-52}$. We use a variant of this approximation (with a bias of 2^{-58}) in order to attack 10-round Serpent with all key lengths with data complexity of 2^{118} and running time of 2^{88} memory accesses. Using another variant of this approximation we attack up to 11-round Serpent with 256-bit keys. The attack requires the same amount of data and its analysis time is equivalent to 2^{214} memory accesses.

The paper is organized as follows: In Section 2 we give a short description of Serpent. In Section 3 we present the 9-round linear approximation of Serpent. In Section 4 we describe the search methodology we used to find the approximation. In Section 5 we present the linear attack on 10-round Serpent with all possible key lengths. In Section 6 we present the linear attack on 11-round Serpent with 256-bit keys. Section 7 summarizes the paper.

2 A Description of Serpent

Serpent [1] is an SP-network block cipher with block size of 128 bits and 0–256 bit keys. It is composed of alternating layers of key mixing, S boxes and linear transformation. Serpent has two equivalent descriptions, and we choose to deal only with the bitsliced description.

The key scheduling algorithm of Serpent accepts 128-bit, 192-bit and 256-bit keys[1]. Keys shorter than 256 bits are padded by 1 followed by as many 0's as needed to have a total length of 256 bits. The key is then used to derive 33 subkeys of 128 bits each.

We use the notations of [1]. Each intermediate value of round i is denoted by $\hat{B}_i$ (a 128-bit value). The 32 rounds are numbered 0–31.

[1]The key scheduling algorithm accepts keys with 0–256 bits, however, we focus on these 3 variants as they were considered during the AES

Serpent has a set of eight 4-bit to 4-bit S boxes. Each round function R_i ($i \in \{0, \ldots, 31\}$) uses a single S box 32 times in parallel. For example, R_0 uses S_0, 32 copies of which are applied in parallel. Each S box permutes one nibble, i.e., for $0 \leq j \leq 31$ we take the j'th bit of $X_3, \ldots, X_0$ and return the value according to the S box.

The set of eight S boxes is used four times. In round i we use $S_{i \bmod 8}$, i.e., S_0 is used in round 0, 8, etc. The last round (round 31) is slightly different than the others: apply S_7 on $\hat{B}_{31} \oplus \hat{K}_{31}$, and XOR the result with $\hat{K}_{32}$ rather than applying the linear transformation.

The cipher may be formally described by the following equations:

$$\begin{aligned} \hat{B}_0 &:= P \\ \hat{B}_{i+1} &:= R_i(\hat{B}_i) \\ C &:= \hat{B}_{32} \end{aligned}$$

where

$$\begin{aligned} R_i(X) &= LT(\hat{\mathcal{S}}_i(X \oplus \hat{K}_i)) & i &= 0, \ldots, 30 \\ R_i(X) &= \hat{\mathcal{S}}_i(X \oplus \hat{K}_i) \oplus \hat{K}_{32} & i &= 31 \end{aligned}$$

where $\hat{\mathcal{S}}_i$ is the application of the S box $S_{i \bmod 8}$ 32 times in parallel, and LT is the linear transformation.

The linear transformation mixes the four 32-bit words in the following manner:

$$\begin{aligned} X_0, X_1, X_2, X_3 &:= \mathcal{S}_i(\hat{B}_i \oplus \hat{K}_i) \\ X_0 &:= X_0 <<< 13 \\ X_2 &:= X_2 <<< 3 \\ X_1 &:= X_1 \oplus X_0 \oplus X_2 \\ X_3 &:= X_3 \oplus X_2 \oplus (X_0 << 3) \\ X_1 &:= X_1 <<< 1 \\ X_3 &:= X_3 <<< 7 \\ X_0 &:= X_0 \oplus X_1 \oplus X_3 \\ X_2 &:= X_2 \oplus X_3 \oplus (X_1 << 7) \\ X_0 &:= X_0 <<< 5 \\ X_2 &:= X_2 <<< 22 \\ \hat{B}_{i+1} &:= X_0, X_1, X_2, X_3 \end{aligned}$$

where $<<<$ denotes rotation, and $<<$ denotes shift.

In the last round, this linear transformation is replaced by an additional key mixing, thus $C = \hat{B}_{32} := S_7(\hat{B}_{31} \oplus \hat{K}_{31}) \oplus \hat{K}_{32}$.

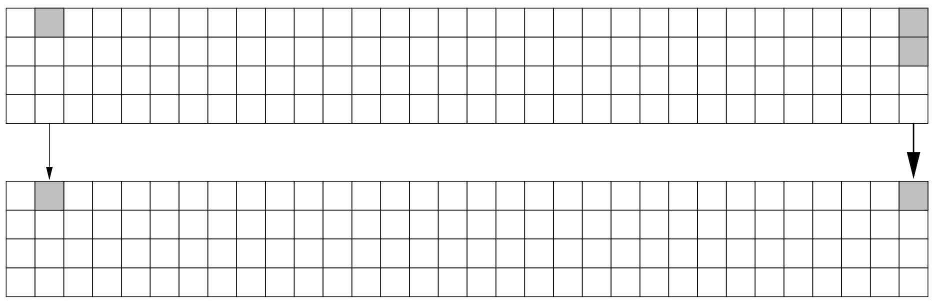


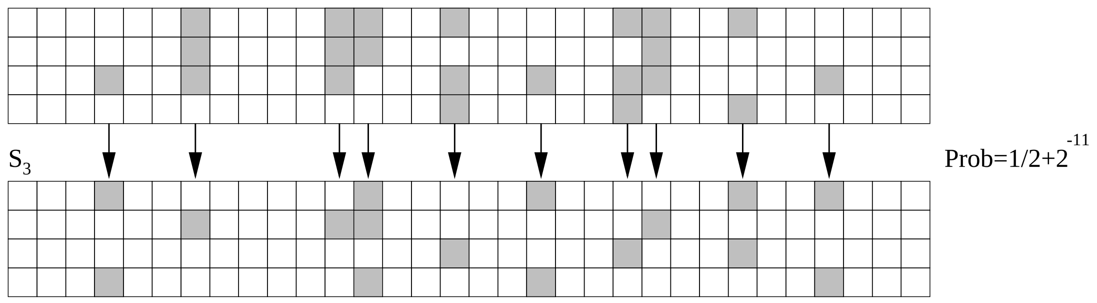
Figure 1: Linear Approximation Representation Example

3 A 9-round Linear Approximation

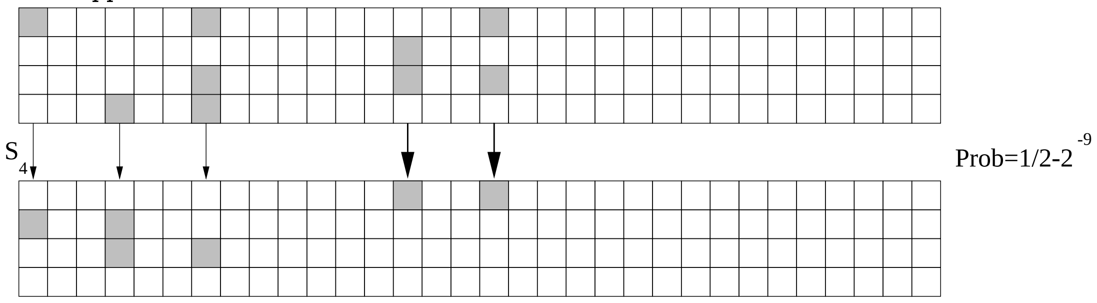
In the submission package of Serpent to the AES, the submitters claimed upper bounds on the probability of linear approximations for Serpent. Since then, no research on linear approximations of Serpent has been published.

We present the 9-round linear approximation we have found during our research which has probability of $1/2 + 2^{-52}$. In order to do so, we adopt a representation which is similar to the one used to describe differential characteristic in [6, 4]. In our representation, the rows are the bitsliced 32-bit words, where each column is the input to an S box. The upper row represents X_0 and the lower one represents X_3. The rightmost column represents the least significant bit, and the leftmost column represents the most significant bit. A square related to a bit which is taken into consideration in the approximation is marked in grey. A thin arrow means that for this S box, if we take the parity of the subset of the input bits marked in grey, the probability that we receive the same parity as the parity of the output subset is $\frac{1}{2} \pm \frac{1}{8}$. A wide arrow means the approximation has probability $\frac{1}{2} \pm \frac{1}{4}$. An example for our notation is presented in Figure 1, where in the first S box (related to least significant bits of $X_0, \ldots, X_3$) the parity of input subset 1 equals to the output subset 3 parity with probability $\frac{1}{2} \pm \frac{1}{4}$ (1/4 or 3/4), and in S box 30 the parity of input subset 3 equals parity of output subset 1 with probability $\frac{1}{2} \pm \frac{1}{8}$ (3/8 or 5/8).

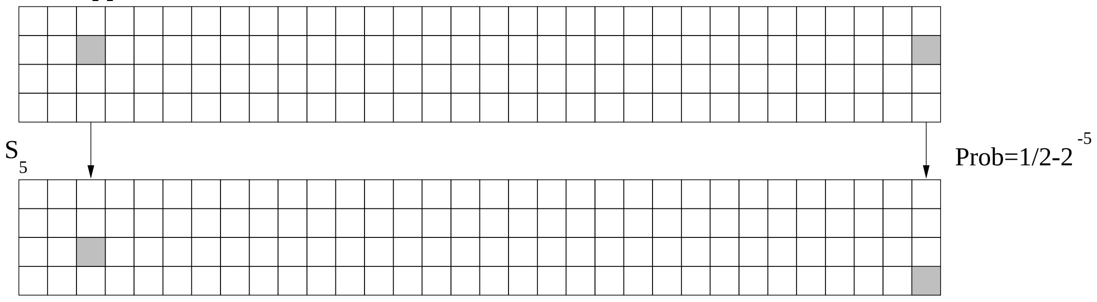
The 9-round linear approximation has a probability of $1/2 + 2^{-52}$. It starts in round 3 (or 11 or 19 or any other round using S_3), and the following approximation holds with bias of 2^{-11}:



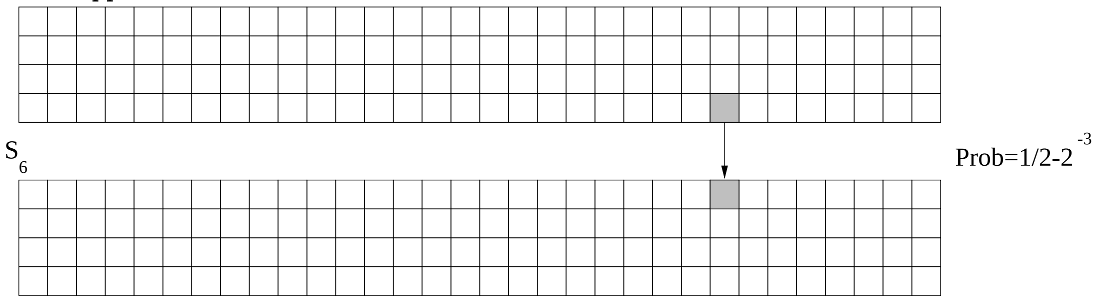
After the linear transformation and the application of S_4, we get the following linear approximation with bias of 2^{-9}:



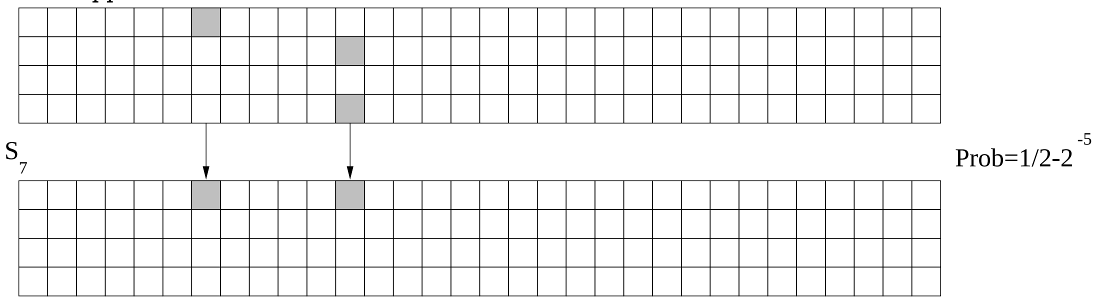
After the linear transformation and the application of S_5, we get the following linear approximation with bias of 2^{-5}:



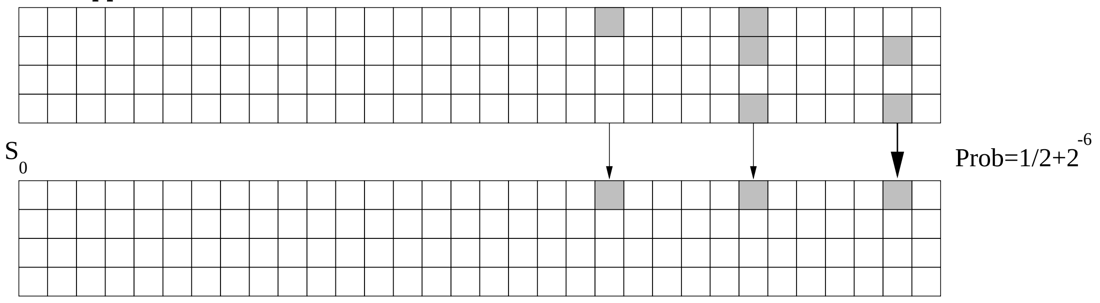
After the linear transformation and the application of S_6, we get the following linear approximation with bias of 2^{-3}:



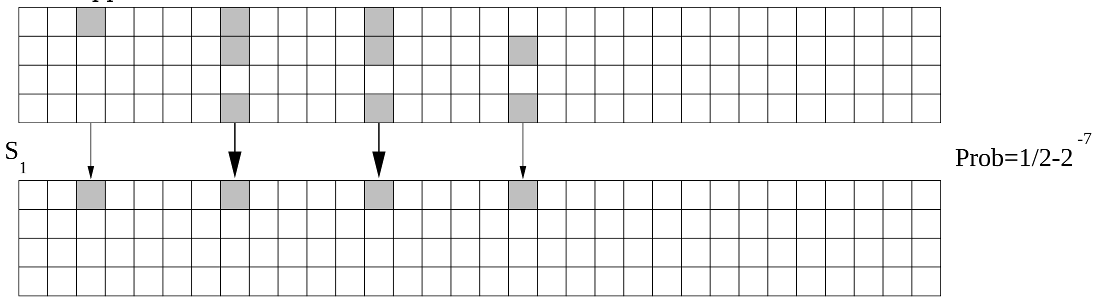
After the linear transformation and the application of S_7, we get the following linear approximation with bias of 2^{-5}:



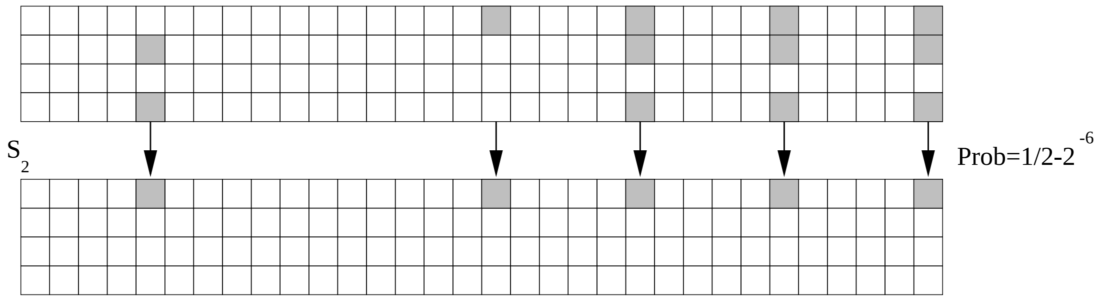
After the linear transformation and the application of S_0, we get the following linear approximation with bias of 2^{-6}:



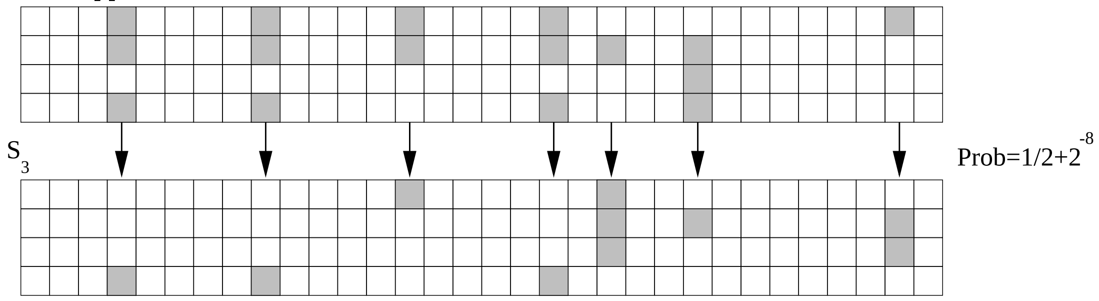
After the linear transformation and the application of S_1, we get the following linear approximation with bias of 2^{-7}:



After the linear transformation and the application of S_2, we get the following linear approximation with bias of 2^{-6}:



After the linear transformation and the application of S_3, we get the following linear approximation with bias of 2^{-8}:



The total bias of the approximation is therefore $2^8 \cdot 2^{-11} \cdot 2^{-9} \cdot 2^{-5} \cdot 2^{-3} \cdot 2^{-5} \cdot 2^{-6} \cdot 2^{-7} \cdot 2^{-6} \cdot 2^{-8} = 2^{-52}$.

The input and output subsets can be replaced with any subset which satisfies the biases of the approximations (for example, having $1 \to E_x$ instead of $1_x \to 6_x$ in S box 1 in the last round of the approximation). Therefore, there are 2^{17} linear approximations with bias 2^{-52}, half of them have probability $\frac{1}{2} + 2^{-52}$ and the other half have probability $\frac{1}{2} - 2^{-52}$.

These approximations can also be rotated by one or two bits to the left (but not three), due to the nature of the linear transformation. Thus, we have in total $3 \cdot 2^{17}$ linear approximations with bias 2^{-52}.

4 Search Methodology

The avalanche effect in Serpent is caused by the linear transformation, which mixes the intermediate words, with various offsets. Thus, we examined the connection between the input and the output of the linear transformation from the point of view of the linear cryptanalysis.

In order to find linear approximation with maximal bias, we used the following search method:

- We tried all the possibilities of a single active input bit to the linear transformation, and counted how many output bits become active. Moreover, we searched how many S boxes are activated in the next round. We have observed that bits from X_0 and X_2 affect 2–4 S boxes after the linear transformation, and the minimum 2 is achieved by one of several bits from X_0.

- We tried all the possibilities of a single active output bit from the linear transformation, and counted how many input bits are active. Moreover, we searched how many S boxes are activated in the previous round. We have observed that bits from X_1 and X_3 affect 2–4 S boxes before the linear transformation, and the minimum 2 is achieved by one of several bits from X_3.

- For S boxes 7, 8 and 9 (out of the parallel 32), the input can have only two active S boxes in the round before, and the output can activate only two S boxes in the next round. Thus, we found a 3-round approximation with 5 active S boxes. This, of course, requires that $1_x \rightarrow 8_x$ in the S box in the middle round, as we want 2 active S boxes in the previous round, and 2 active S boxes in the next round.

- We tried to add more rounds to the approximation, using the fact that when going backward to the previous round, it is better to take subsets of bits from X_1 and X_3, and while going forward it is better to take subsets of bits from X_0 and X_2.

- By iteratively adding rounds in the beginning and in the end of the approximation we have found a 9-round approximation.

As noted above we have found linear approximation for 3 to 9 rounds of Serpent. Details are given in Table 1. The approximations themselves can be easily derived from the 9-round approximation by removing the unnecessary rounds, and making the edge rounds (first and last round) of the approximation with maximal bias. From this table it becomes clear that the authors of Serpent made a mistake in their claimed bounds, as the biases found here are 4–8 times higher than the bounds in [1]. However, this mistake does not affect the security claims for the whole cipher, as even with the new approximations 16-round Serpent is still secure, not to mention 32-round Serpent.

5 Attacking 10-round Serpent

We attack 10-round Serpent (rounds 3–12) using a 9-round linear approximation. The 10-rounds are just like Serpent, besides the fact that after the last round (round 12) the linear transformation is omitted (as it has no cryptographic significance) and replaced by a key mixing operation (otherwise the attack gets additional round free of charge).

Number of Rounds	Starting from	Number of Active S boxes	Bias	Claimed Best Bias [1]
3	S_5	5	2^{-7}	2^{-9}
4	S_5	8	2^{-12}	2^{-15}
5	S_5	12	2^{-18}	2^{-21}
6	S_4	17	2^{-25}	2^{-28}
6	S_5	17	2^{-25}	2^{-28}
7	S_4	22	2^{-32}	2^{-34}
8	S_4	29	2^{-39}	
9	S_3	39	2^{-52}	

Table 1: Our Best Linear Approximations for Serpent

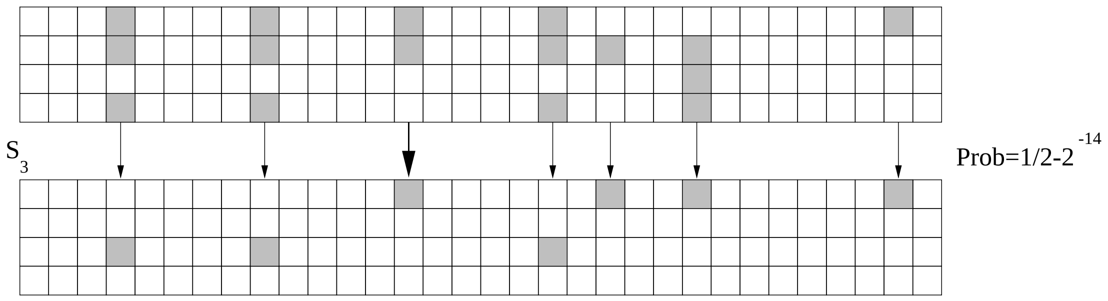


Figure 2: Last Round of the 9-Round Linear Approximation Used in the 10-Round Attack

The attack uses a 9-round which is similar to the one presented in Section 3. However, mounting a standard linear attack using the original 9-round approximation would result in too many active S boxes in round 12. In order to avoid this problem, we have altered the last round of the approximation (round 11) to the one presented in Figure 2. This approximation has probability of $1/2 - 2^{-58}$ (instead of $1/2 + 2^{-52}$), but we reduce by 12 the number of active S boxes in round 12, as this last round activates only 11 S boxes in round 12 instead of 23 as in the original approximation.

A basic attack can be devised based on this approximation. For each of the 2^{118} known plaintexts, decrypt each ciphertext through the 11 active S boxes under all possible 44-bit subkeys and calculate the parity of the input and output subsets. The right subkey is with high probability the one with maximal deviation from 1/2.

In order to retrieve the other key bits, we can use other approximations, which have the same probability (or a little better), which have additional S boxes in round 12. As we already found the subkey for 11 S boxes, we can use other approximations which activate these S boxes with a few additional S boxes (and thus retrieve more key bits).

However, this attack has time complexity of $2^{44} \cdot 2^{118} = 2^{162}$ which is much more than exhaustive search for 128-bit keys. But, we extract and consider only 44 bits from each ciphertext, and decrypt only those bits under all possible 44-bit subkeys, the same work is done many times (decrypting the same value under the same subkey).

Observing the fact that only a 44-bit value is extracted from each ciphertext, and only a 44-bit subkey is involved, we can perform the following improvement: We decrypt all the 44-bit ciphertext values, under all possible 44-bit subkeys, and keep in a table the parity of the output subset (as the actual value does not interest us). Now we just need to count how many times each of the 2^{44} possible values of the 44 ciphertext bits appear and what is the corresponding input subset parity.

Thus we use the following algorithm:

1. For any 44-bit ciphertext value, and for any 44-bit subkey, decrypt the partial ciphertext under the 11 active S boxes, and calculate the parity of the output subset (as defined by the linear approximation). Keep the value in a table.

2. Initialize array of 2^{45} counters (118 bits long).

3. For each plaintext P calculate the input subset parity, and extract the 44-bit ciphertext value. Advance the related counter (each counter counts how many time the specific 44-bit ciphertext value with the input parity was encountered).

4. For each of the 2^{45} counters and for each of the 2^{44} possible 44-bit subkeys, check the parity induced by the subkey.

5. Output the subkey which has maximal deviation from 1/2 (half of the plaintexts).

This way the table is accessed 2^{45} times for each possible subkey.

A trivial improvement can count only the 44-bit ciphertext value, and add/subtract one from the corresponding counter according to the parity of the input subset (as opposite parities balance each other). This way the table is accessed only 2^{44} times for each possible subkey.

The time complexity of the algorithm is the time needed to decrypt all the 44-bit ciphertext values under all 44-bit subkeys, through 11 S boxes (we ignore the time of extraction from the plaintexts, as we can ask in advance from the encryption function to give us only these bits). As we deal only with one round out of 10, the time complexity is about $2^{44} \cdot 2^{44} \cdot 1/10 = 2^{84.68}$ 10-round Serpent encryptions.

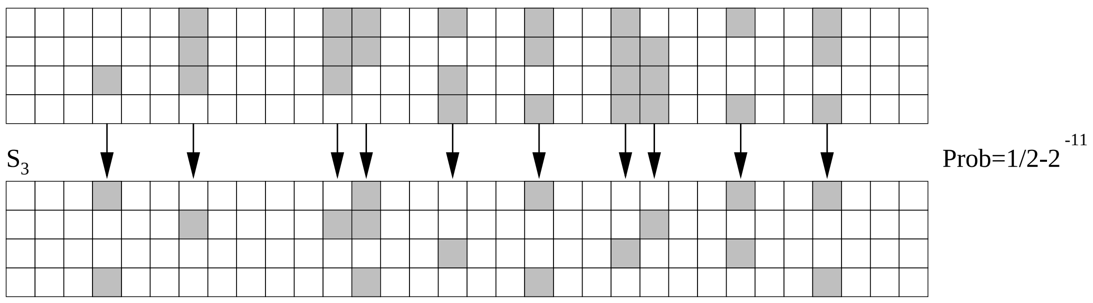


Figure 3: First Round of the 9-Round Linear Approximation Used in the 11-Round Attack

Again we can use other approximations to find the remaining bits of the last, and for key sizes larger than 128 bits, we can peel off the last round, and use better characteristics to find more subkey material (which is translated into knowledge about the key).

We can also use other linear approximations. There is a linear approximation with bias 2^{-57}, with only 12 active S boxes in the last round. Using this approximation instead would require 2^{116} plaintexts, 2^{48} counters, and $2^{48} \cdot 2^{48} = 2^{96}$ times decryption through 12 S boxes, which are less than $2^{96} \cdot 1/10 = 2^{92.68}$ 10-round Serpent encryptions.

For 192-bit and 256-bit keys we can use the original 9-round linear approximation, using 23 S boxes and reducing the required plaintexts to 2^{106}. The memory complexity in such a case would be 2^{92} counters, and the time needed is equivalent to $2^{92} \cdot 2^{92} = 2^{184}$ times encrypting 1-round of Serpent (actually 23 S boxes).

6 Attacking 11-round 256-bit key Serpent

We attack 11-round Serpent (rounds 2–12) using a 9-round linear approximation. The 11 rounds are again just like Serpent, besides the fact that after the last round (round 12) the linear transformation is omitted (as it has no cryptographic significance).

The attack is based on a 9-round linear approximation (rounds 3–11). It is similar to the one used in Section 5, besides a small change in the first round of the approximation which is presented in Figure 3. This modification reduces the number of active S boxes in round 2 by 5 from 29 to 24, without changing the probability of the approximation ($1/2 - 2^{-58}$).

The basic attack is similar to the basic attack from Section 6. We encrypt all 2^{118} plaintexts through the 24 active S boxes in round 2 and decrypt them through the 11 active S boxes in round 12. As we check 2^{140} possible subkeys, the time complexity of such an attack is $2^{118} \cdot 2^{140} = 2^{258}$, but as we encrypt/decrypt only in 2 out of 11 rounds, the time complexity is about the time complexity of exhaustive search (for 256-bit keys).

But again we observe that the same operation is done more than once for each subkey. This time we perform the counting according to the 96 bits from the plaintext, and take the output subset parity from the ciphertext. However, we need to know the parity in round 11. In order to do so, we just repeat the attack for each of the possible 2^{44} possible subkeys in round 12.

The attack goes as follows:

1. For each of the possible 96-bit plaintext values, and 96-bit subkeys (24 active S boxes in round 2) calculate the encryption of the value under the subkey, and compute the parity of the active bits in the approximation. Store the value in a 2^{192} 1-bit entry table.
2. For each possible 44-bit subkey in round 12 do the following:
   (a) Decrypt all 2^{118} ciphertexts through the 11 active S boxes in round 12, and calculate the parity.
   (b) Extract the 96-bit plaintext value corresponding to the 24 active S boxes in round 2 and add/subtract the corresponding counter (according to the parity of the output subset).
   (c) For each of the 96-bit subkeys calculate the bias according to the precomputed table (step 1) and the number of 96-bit plaintext value (according to the parity step 2b).
   (d) Choose the key with maximal absolute value, and keep this value aside.
3. For all the 44-bit subkeys, check what is the maximal absolute value among the values kept aside. Choose the 44-bit subkey which is related to the 96-bit subkey chosen.

It takes $2^{96} \cdot 2^{96} = 2^{192}$ encrypting through 24 S boxes to create the large table which hold the parities. For each subkey of the last round (2^{44}) we decrypt all ciphertexts (2^{118}) and count them. Moreover, for each 96-bit subkey of the first round we need to go over the table of 2^{96} plaintext values, which takes 2^{192} memory accesses. Thus, the time complexity of the algorithm is $2^{192} + 2^{44} \cdot (2^{118} + 2^{192}) = 2^{236}$ memory accesses.

We can perform the same algorithm, but this time trying all 96-bit subkeys, and receive an algorithm with time complexity of $2^{88} + 2^{96} \cdot (2^{118} + 2^{88}) = 2^{214}$ memory accesses. The memory needed is 2^{88} entries for the precomputed parity table, plus 2^{44} counters. Note that the maximal bias can be kept in one memory block (the bias + 96-bit subkey + 44 bit subkey). Thus, 2^{88} bits of memory are sufficient.

We conclude that this attack is better than exhaustive search for 256-bit keys.

Rounds	Key Size	Complexity			Source
		Data	Time	Memory	
7	256	2^{122}	2^{248}	2^{126}	[6] - Section 3.5
	all	2^{85}	2^{86} MA	2^{52}	[4] - Section 3
8	192 & 256	2^{128}	2^{163}	2^{133}	[6] - Section 4.2
	192 & 256	2^{110}	2^{175}	2^{115}	[6] - Section 5.3
	256	2^{85}	2^{214} MA	2^{85}	[4] - Section 3.1
9	256	2^{110}	2^{252}	2^{212}	[6] - Section 5.4
10	256	$2^{126.4}$	2^{208}	$2^{131.4}$	[4] - Section 4
	256	$2^{126.4}$	$2^{204.8}$	$2^{195.5}$	[4] - Section 4.1
	all	2^{118}	$2^{88}MA$	2^{44} bits	This paper
	all	2^{116}	$2^{92}MA$	2^{48} bits	This paper
	192 & 256	2^{106}	2^{184}MA	2^{184} bits	This paper
11	256	2^{118}	$2^{214}MA$	2^{88} bits	This paper

MA - Memory Accesses

Table 2: Known Attacks on Serpent

7 Summary

In this paper we presented the first known linear cryptanalysis of Serpent, along with the best known 9-round linear approximation for Serpent with probability $1/2 + 2^{-52}$, and explained our search method.

Using the approximations we found, we devised attacks against 10-round Serpent with all key lengths with time complexity 2^{88} memory accesses and 2^{118} known plaintexts. We have also presented an attack on 11-round Serpent with 256-bit keys with time complexity of 2^{214} memory accesses with the same data complexity (2^{118}) and 2^{88} bits of memory.

8 Acknowledgment

The authors would like to thank Tal Mor and Vladimir Furman for their many comments and suggestions which improved the results and presentation of this paper.

References

[1] Ross Anderson, Eli Biham, Lars R. Knudsen, *Serpent: A Proposal for the Advanced Encryption Standard*, NIST AES Proposal 1998. Available on-line at *http://www.nist.gov/aes/*, 1997.

[2] Eli Biham, *A Note on Comparing the AES Candidates*, Second AES Candidate Conference, 1999. Available on-line at *http://www.nist.gov/aes/*, 1997.

[3] Eli Biham, Adi Shamir, *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag, 1993.

[4] Eli Biham, Orr Dunkelman, Nathan Keller, *The Rectangle Attack – Rectangling the Serpent*, proceeding of Eurocrypt 2001, to appear. Available on-line at *http://vipe.technion.ac.il/~orrd/crypt/*.

[5] Orr Dunkelman, *An Analysis of Serpent-p and Serpent-p-ns*, presented at the rump session of the Second AES Candidate Conference, 1999. Available on-line at *http://vipe.technion.ac.il/~orrd/crypt/*.

[6] Tadayoshi Kohno, John Kelsey, Bruce Schneier, *Preliminary Cryptanalysis of Reduced-Round Serpent*, Third AES Candidate Conference, 2000. Available on-line at *http://www.nist.gov/aes/*, 1997.

[7] John Kelsey, Tadayoshi Kohno, Bruce Schneier, *Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent*, proceedings of Fast Software Encryption 7, 2001, to appear.

[8] Mitsuru Matsui, Atsuhiro Yamagishi, *A New Method for Known Plaintext Attack of FEAL Cipher*, proceedings of Eurocrypt 1992, Springer Verlag LNCS 658, pp. 81–91.

[9] Mitsuru Matsui, *Linear Cryptanalysis Method for DES Cipher*, proceedings of Eurocrypt 93, Springer Verlag LNCS 765, pp. 386–397, 1994.

[10] Mitsuru Matsui, *The First Experimental Cryptanalysis of the Data Encryption Standard*, proceedings of Crypto 1994, Springer Verlag LNCS 839, pp. 1–11, 1994.

[11] NIST, *A Request for Candidate Algorithm Nominations for the AES*. Available on-line at *http://www.nist.gov/aes/*, 1997.