Properties of the Building Blocks of Serpent*

Serge Mister (serge.mister@entrust.com)
Entrust Technologies

May 15, 2000

1 Introduction

In assessing the security of AES candidates, it is important to consider the amount of analysis that has been conducted. To this end, this note summarizes properties of the building blocks of Serpent (specifically the s-boxes and linear transformation) that were observed during analysis. Although none of these properties appear to lead to direct cryptanalytic attacks, they may suggest avenues for further exploration.

1.1 Serpent Structure

In [1], two implementations of Serpent are described; the "bit slice" version is used throughout this note. Serpent is a 32 round iterated block cipher with a 128 bit block size. The structure of the first 31 rounds of Serpent is illustrated in Figure 1. In the figure, "Input" is the input from the previous round (or the plaintext for Round 0), $\oplus$ denotes the bitwise XOR operation, i is the round number (numbered 0-31), k_i is the 128-bit i-th round subkey, S'_0-S'_7 are eight s-box transformations, and LT is the linear transformation layer. The last round differs from the previous ones in that the linear transformation LT is replaced with another XOR operation with the last round subkey k_{32}. Each s-box transformation S'_j shown in the diagram is actually 32 parallel applications of a single 4×4 invertible s-box S_j. The 128-bit input to the s-box transformation is partitioned into four 32-bit words. Then the k-th bit of each word is used as an input for the k-th instance of the s-box, and the k-th bit of each of the 32-bit output words is used to store

*Thanks to Carlisle Adams for his valuable input on this work.

the output of that s-box. For efficiency, this s-box transformation is implemented using a boolean expression for the s-box evaluated using the four 32-bit inputs.

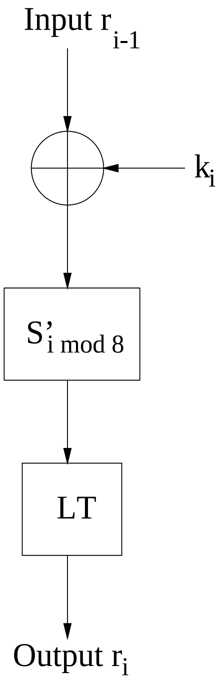

Figure 1: A round of Serpent

1.2 Serpent S-Boxes

In this section, properties of the s-boxes defined for use in Serpent are discussed.

1.2.1 S-box Specifications

The Serpent s-boxes can be described as shown in Table 1. The inverses of the s-boxes can be described as shown in Table 2.

The s-boxes were selected based on their resistance to differential [3] and linear cryptanalysis [2]. The following properties were required (quoted from [1]):

	Output							
Input	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
0	3	15	8	0	1	15	7	1
1	8	12	6	15	15	5	2	13
2	15	2	7	11	8	2	12	15
3	1	7	9	8	3	11	5	0
4	10	9	3	12	12	4	8	14
5	6	0	12	9	0	10	4	8
6	5	5	10	6	11	9	6	2
7	11	10	15	3	6	12	11	11
8	14	1	13	13	2	0	14	7
9	13	11	1	1	5	3	9	4
10	4	14	14	2	4	14	1	12
11	2	8	4	4	10	8	15	10
12	7	6	0	10	9	13	13	9
13	0	13	11	7	14	6	3	3
14	9	3	5	5	7	7	10	5
15	12	4	2	14	13	1	0	6

Table 1: S-box definitions

- each differential characteristic has a probability of at most 1/4, and a one-bit input difference will never lead to a one-bit output difference;

- each linear characteristic has a probability in the range $1/2 \pm 1/4$, and a linear relation between one single bit in the input and one single bit in the output has a probability in the range $1/2 \pm 1/8$;

- the nonlinear order of the output bits as a function of the input bits is the maximum, namely 3.

1.2.2 Invariance Properties

One general form of attack on a cipher is to submit two plaintexts x_1 and $x_2 = f(x_1)$ for encryption. Denote the output of the cipher to be y_1 and y_2 when x_1 and x_2 are encrypted, respectively. If it is sufficiently likely that $y_2 = g(y_1)$ for a known function g, an attack may be possible. Differential cryptanalysis fits into this attack model, with $f(x) = x \oplus \Delta_{input}$ and $g(y) = y \oplus \Delta_{output}$. In differential cryptanalysis, the XOR relationship holds with significant probability up to the last round of the cipher, and the last round

	Output							
Input	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
0	13	5	12	0	5	8	15	3
1	3	8	9	9	0	15	10	0
2	11	2	15	10	8	2	1	6
3	0	14	4	7	3	9	13	13
4	10	15	11	11	10	4	5	9
5	6	6	14	14	9	1	3	14
6	5	12	1	6	7	13	6	15
7	12	3	2	13	14	14	0	8
8	1	11	0	3	2	11	4	5
9	14	4	3	5	12	6	9	12
10	4	7	6	12	11	5	14	11
11	7	9	13	2	6	3	7	7
12	15	1	5	4	4	7	2	10
13	9	13	8	8	15	12	12	1
14	8	10	10	15	13	10	8	4
15	2	0	7	1	1	0	11	2

Table 2: S-box inverses

is used to deduce information about the round key used in that round.

In Serpent, the s-boxes were specifically designed to protect against differential cryptanalysis, but it is possible that other choices of the functions f and g may lead to successful attack.

Consider a boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$. Let $\mathcal{Z} = \{x \in \{0,1\}^n | f(x) = 0\}$, the set of inputs that f maps to 0. Similarly, let $\mathcal{I} = \{x \in \{0,1\}^n | f(x) = 1\}$, the set of inputs that f maps to 1. Then the set of mappings π satisfying $f(\pi(x)) = f(x)$ for all x is the set of mappings that rearrange the inputs mapped to 0 amongst themselves and those mapped to 1 amongst themselves. More formally, the mappings are those defined by:

$$\pi(x) = \begin{cases} f(\pi_0(x)) & \text{if } x \in \mathcal{Z} \\ f(\pi_1(x)) & \text{if } x \in \mathcal{I} \end{cases}$$

where $\pi_0 : \mathcal{Z} \rightarrow \mathcal{Z}$ and $\pi_1 : \mathcal{I} \rightarrow \mathcal{I}$.

It can thus be observed that for all boolean functions, there exist transformations of the input that do not affect the output. It is interesting to consider whether or not such transformations could be useful in attacking Serpent. In any round of Serpent, the output of the previous round is xored with a round key before being submitted to the s-box. The output of the round is given by:

$$r_i = LT(S_i'(r_{i-1} \oplus k_i)) \ .$$

If a function f is applied to the input, the output would become:

$$r_i = LT(S_i'(f(r_{i-1}) \oplus k_i))$$

which, if f is linear in the XOR operation and k_i is not changed by the transformation, becomes

$$\begin{aligned} r_i &= LT(S_i'(f(r_{i-1}) \oplus f(k_i))) \\ &= LT(S_i'(f(r_{i-1} \oplus k_i))) \ . \end{aligned}$$

A program was written to search for linear transformations f for which at least one output bit of $S_i(x)$ is the same as the corresponding bit of $S_i(f(x))$ for all x. Table 3 lists, for each s-box S_i, the linear transformations f with the property that at least two output bits of $S_i(x)$ are the same as the corresponding output bits of $S_i(f(x))$. The linear mapping f is described in the table by four 4-bit binary numbers. The jth 4-bit number $(1 \leq j \leq 4)$ defines $f(x)$ for an input with only bit $4 - j$ set. By the linearity property $f(a \oplus b) = f(a) \oplus f(b)$, all other outputs can be determined. For example, consider calculating $f(1011)$ where f is the first transformation listed for S_4:

$$\begin{aligned} f(1011) &= f(1000) \oplus f(0010) \oplus f(0001) \\ &= 0101 \oplus 0010 \oplus 1100 \\ &= 1011 \end{aligned}$$

The last column indicates the number of times that the linear mapping results in a change to the output of the s-box.

The identity mapping (i.e. 1000 0100 0010 0001) does not provide a useful transformation, but it can be seen that S_1, S_4, S_5, and S_7 all have a single non-trivial mapping satisfying the invariance property. Similarly, S_0^{-1} and S_2^{-1} have nontrivial mappings satisfying the invariance property. In fact, S_2^{-1} has three such mappings. Note also that, because the s-boxes are invertible, a change in the input implies a change in the output. Thus, for example, for s-box S_1 the nontrivial linear transformation results in one of three output changes:

		Number of Output Changes				Total Output
S-Box	Linear Transformation	Bit 3	Bit 2	Bit 1	Bit 0	Changes
S_0	1000 0100 0010 0001	0	0	0	0	0
S_1	0110 0100 1100 1111	4	4	0	0	8
	1000 0100 0010 0001	0	0	0	0	0
S_2	1000 0100 0010 0001	0	0	0	0	0
S_3	1000 0100 0010 0001	0	0	0	0	0
S_4	0101 0100 0010 1100	0	4	6	0	8
	1000 0100 0010 0001	0	0	0	0	0
S_5	1000 0100 0010 0001	0	0	0	0	0
	1000 0100 0101 0110	6	4	0	0	8
S_6	1000 0100 0010 0001	0	0	0	0	0
S_7	0111 1011 1101 1110	6	6	0	0	8
	1000 0100 0010 0001	0	0	0	0	0
S_0^{-1}	0110 0100 1100 1111	0	4	0	4	8
	1000 0100 0010 0001	0	0	0	0	0
S_1^{-1}	1000 0100 0010 0001	0	0	0	0	0
S_2^{-1}	0011 0010 0100 1100	2	12	0	0	12
	0011 1111 1001 0001	4	8	0	0	8
	1000 0100 0010 0001	0	0	0	0	0
	1000 1001 1111 1100	6	4	0	0	8
S_3^{-1}	1000 0100 0010 0001	0	0	0	0	0
S_4^{-1}	1000 0100 0010 0001	0	0	0	0	0
S_5^{-1}	1000 0100 0010 0001	0	0	0	0	0
S_6^{-1}	1000 0100 0010 0001	0	0	0	0	0
S_7^{-1}	1000 0100 0010 0001	0	0	0	0	0

Table 3: Linear transformations to which some s-box output bits are invariant

1. a change in output bit 3 (with probability 1/4),
2. a change in output bit 2 (with probability 1/4), or
3. no output change (with probability 1/2).

Bits 2 and 3 never change together since the total number of output changes is 8.

In explicitly choosing the s-boxes to be resistant to differential cryptanalysis, the designers of Serpent ensured that an input difference would lead to a specific output difference with a probability of at most $\frac{1}{4}$. However, as shown in Table 3, these same s-boxes admit linear transformations of the input that lead to output differences with probabilities greater than $\frac{1}{4}$. Although this property could possibly be exploited in the first round of the cipher, it is not clear that it could be extended to multiple rounds. This is because the output relationship is a simple XOR difference and not the original linear transformation relationship. The technique may, in general, be used to produce a higher-probability input difference to the second round of a cipher, but this does not appear possible for Serpent.

1.3 Linear Transformation

Labeling the linear transformation input by four 32-bit words X_0,X_1,X_2,X_3, with X_3 being the most significant (leftmost) 32-bits of the input, the linear transformation can be described by the following sequence of operations:

$$\begin{aligned}
X_0 &= X_0 <<< 13 \\
X_2 &= X_2 <<< 3 \\
X_1 &= X_1 \oplus X_0 \oplus X_2 \\
X_3 &= X_3 \oplus X_2 \oplus (X_0 \ll 3) \\
X_1 &= X_1 <<< 1 \\
X_3 &= X_3 <<< 7 \\
X_0 &= X_0 \oplus X_1 \oplus X_3 \\
X_2 &= X_2 \oplus X_3 \oplus (X_1 \ll 7) \\
X_0 &= X_0 <<< 5 \\
X_2 &= X_2 <<< 22
\end{aligned}$$

Consider an input to the linear transformation layer for which all four 32-bit words are the same. Is it ever the case that the output also has several equal 32-bit words? A search through the 2^{32} inputs for which all four input words are the same shows that the output is the same in at least two words exactly 11 times. It is interesting to note from Table 4 that it is most often the last two words of the output that are the same (8 times out of 11).

Input	Output Words			
00000000	**00000000**	**00000000**	**00000000**	**00000000**
04055ab0	4c10fbdf	1ef31f61	**4fc5983f**	**4fc5983f**
420a4b53	54ef7bbf	3664f310	**85a9e08c**	**85a9e08c**
460f11e3	18ff8060	2897ec71	**ca6c78b3**	**ca6c78b3**
547eb655	6c36b9ef	**42835ce0**	**42835ce0**	ef2843a0
78dd4a9d	**5533577a**	2ec962da	a47dda1b	**5533577a**
8011cc16	e5ce0ca1	723af941	**44965926**	**44965926**
804e8b77	2dde013a	**a6a84184**	0e3c3f36	**a6a84184**
841496a6	a9def77e	6cc9e620	**0b53c119**	**0b53c119**
c21b8745	b121771e	445e0a51	**c13fb9aa**	**c13fb9aa**
c61eddf5	fd318cc1	5aad1530	**8efa2195**	**8efa2195**

Table 4: Inputs for which at least two output words are the same in the linear transformation

2 Areas for Future Work

Many cryptographers currently design s-boxes using a “random” process, checking for properties required to obtain bounds on security against known attacks. It could then be asked if such s-boxes, especially if they are small, could exhibit some property not related to the known attacks, but that makes the cipher breakable. The properties observed above show that the Serpent s-boxes, though well-designed with respect to linear and differential cryptanalysis, may still have interesting (i.e. exploitable) properties, although none so far appear to cause weaknesses.

References

[1] R. Anderson, E. Biham, and L. Knudsen. Serpent: A proposal for the advanced encryption standard. Submission to AES Process.

[2] E. Biham. On Matsui's linear cryptanalysis. In *Advances in Cryptology - Proceedings of EUROCRYPT '94*, pages 341–355. Springer-Verlag, 1995.

[3] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. In *Advances in Cryptology: Proceedings of CRYPTO '90*, pages 1–21. Springer-Verlag, 1991.