Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent

John Kelsey[1], Tadayoshi Kohno[2⋆], and Bruce Schneier[1]

[1] Counterpane Internet Security, Inc.
{kelsey,schneier}@counterpane.com
[2] Reliable Software Technologies
kohno@rstcorp.com

Abstract. We introduce a new cryptanalytic technique based on Wagner's boomerang and inside-out attacks. We first describe this new attack in terms of the original boomerang attack, and then demonstrate its use on reduced-round variants of the MARS core and Serpent. Our attack breaks eleven rounds of the MARS core with 2^{65} chosen plaintexts, 2^{70} memory, and 2^{229} partial decryptions. Our attack breaks eight rounds of Serpent with 2^{114} chosen plaintexts, 2^{119} memory, and 2^{179} partial decryptions.

1 Introduction

MARS [BCD+98] and Serpent [ABK98] are block ciphers that have been proposed as AES candidates [NIST97a,NIST97b]. More recently, both were chosen as AES finalists. We have spent considerable time in the last few months cryptanalyzing both ciphers, with the bulk of our results appearing in [KS00,KKS00]. During our work on MARS, we developed a new class of attack based on David Wagner's boomerang and inside-out attacks [Wag99]. In this paper, we present this new class of attack, first in the abstract sense, and then in terms of specific attacks on reduced-round variants of the MARS core and of Serpent.

The MARS core provides an excellent target for these attacks. We know of no good iterative differential characteristics, nor of any good differentials of any useful length. However, there is a three-round characteristic and a three-round truncated differential each with probability one. Since these attacks allow concatenation of short differentials that don't connect in the normal sense of differential attacks, they are quite useful against the MARS core. Similarly, Serpent provides an excellent target for these attacks, because the main problem in mounting a differential attack on Serpent is keeping the differential characteristics used from spreading out to large numbers of active S-boxes; using boomerangs, amplified boomerangs, and related ideas, we can make use of differentials with relatively few active S-boxes, and connect them using the boomerang construction.

⋆ Part of this work was done while working for Counterpane Internet Security, Inc.

B. Schneier (Ed.): FSE 2000, LNCS 1978, pp. 75–93, 2001.

The underlying "trick" of the boomerang attack is to mount a differential attack with first-order differentials that don't normally connect through the cipher; these first-order differentials are connected by a second-order differential relationship in the middle of the cipher, by way of some adaptive-chosen-ciphertext queries. The underlying "trick" of the boomerang-amplifier attack is to use large numbers of chosen plaintext pairs to get that second-order differential relationship to appear in the middle of the cipher by chance. Extensions allow us to use structures of structures to get the second-order differential relationship in the middle for many pairs of texts at once.

1.1 Impact of the Results

The most important impact of our results is the introduction of a new cryptanalytic technique. This technique belongs to the same general class as the boomerang and miss-in-the-middle attacks; the attacker builds structures of a certain kind from right pairs for differentials through part of the cipher. Other members of this class of attacks are known to us, and research is ongoing into their uses and limitations.

Additionally, we provide the best known attack on the MARS core, breaking up to eleven rounds faster then brute-force search. This attack does not threaten full MARS; even if the cryptographic core had only eleven rounds, we know of no way to mount an attack on the core through the key addition/subtraction and unkeyed mixing layers present in the full MARS.

We also demonstrate this powerful new attack on a reduced-round variant of Serpent. Again, this attack does not threaten the full 32-round Serpent.

The attacks described in this paper are summarized below. However, these specific attacks are not the focus of the paper; instead, the focus is the new cryptanalytic technique.

Summary of Results

Cipher (rounds)	**Texts** (chosen plaintexts)	**Memory** (bytes)	**Work** (decryptions)
MARS Core (11)	2^{65}	2^{69}	2^{229} partial
Serpent (8)	2^{114}	2^{119}	2^{179} 8-round

2 Boomerangs, Inside-Out Attacks, and the Boomerang-Amplifier

2.1 Preliminaries

In [Wag99], Wagner introduces two new attacks: the boomerang attack and the inside-out attack. An understanding of both attacks is necessary to understand our new attack. In order to build on these concepts later, we briefly review the concepts from [Wag99].

Most of the attacks in this section make use of a block cipher E composed of two halves, e_0, e_1. That is, $E(X) = e_0(e_1(X))$. We also use the following notation to describe plaintext i as it is encrypted under E:

$$\begin{aligned} X_i &\leftarrow \text{ plaintext} \\ Y_i &\leftarrow e_0(X_i) \\ Z_i &\leftarrow e_1(Y_i) \text{ (ciphertext)} \end{aligned}$$

An important side-note: A normal differential always has the same probability through a cipher (or subset of cipher rounds) going forward and backward; by contrast a truncated differential can have different probabilities going forward and backward.

2.2 The Inside-Out Attack

Consider a situation in which we have probability one truncated differentials through both e_1 and through e_0^{-1}, both with the same starting difference. That is, we have

$$\begin{aligned} \Delta_0 &\rightarrow \Delta_1 \text{ through } e_1 \\ \Delta_0 &\rightarrow \Delta_2 \text{ through } e_0^{-1} \end{aligned}$$

In this case, we can mount an attack to distinguish E from a random permutation as follows:

1. Observe enough known plaintext/ciphertexts pairs that we expect R pairs of texts with the required difference in the middle. That is, we expect about R pairs (i, j) for which $Y_i \oplus Y_j = \Delta_0$.
2. Identify the pairs of inputs where $X_i \oplus X_j = \Delta_2$.
3. Identify the pairs of outputs where $Z_i \oplus Z_j = \Delta_1$.
4. Count the number of pairs that overlap (that is, the pairs that are right pairs in both input and output); if this count is substantially higher than would be expected from a random permutation, we distinguish E from a random permutation.

Suppose we had the following probabilities for a random i, j pair:

$$\begin{aligned} Pr[X_i \oplus X_j = \Delta_2] &= p_0 \\ Pr[Z_i \oplus Z_j = \Delta_1] &= p_1 \end{aligned}$$

That is, the probability of a randomly selected pair fitting the truncated difference Δ_2 is p_0, and its probability of fitting Δ_1 is p_1. In this case, we have:

$N =$ Number of total plaintext/ciphertext pairs.
$N_0 = N * p_0 =$ Expected number of counted right input pairs for random perm.
$N_1 = N * p_0 * p_1 =$ Expected number of those right input pairs
counted as right output pairs.

The number of pairs that are right pairs for both input and output is then binomially distributed, and can be approximated with a normal distribution with $\mu = N_1$ and $\sigma \approx \sqrt{N_1}$. When we have a right pair in the middle (that is, when $Y_i \oplus Y_j = \Delta_0$), then i, j must be a right pair for both inputs and outputs. We expect R right pairs in the middle; that means that for E, we expect about $N_1 + R$ right pairs for both input and output, while for a random permutation, we expect only about N_1. When R is much larger than $\sqrt{N_1}$, this becomes detectable with reasonably high probability. (For very low probabilities, we can use $R > 16\sqrt{N_1}$, which has an astronomically low probability.)

This gives us a way to attack a cipher even when there are no good differentials through the whole cipher, by waiting for the required difference to occur at random in the middle of the cipher. This idea can be extended to deal with differentials with lower probabilities; see [Wag99] for details.

2.3 Boomerangs

Another fundamental idea required to understand the boomerang-amplifier attack is the boomerang attack. Consider the same cipher $E(X) = e_1(e_0(X))$, but now suppose that there are excellent differentials through e_0, e_0^{-1}, and e_1^{-1}. For this discussion, we assume that these are normal differentials and that they have probability one. The attack works with lower-probability differentials, and with truncated differentials; for a full discussion of the additional complexities these raise, see [Wag99].

We thus have the following differentials:

$$\Delta_0 \rightarrow \Delta_1 \text{ through } e_0 \text{ and } e_1$$

with probability one. Now, despite the fact that $\Delta_1 \neq \Delta_0$ and despite a lack of any high-probability differential through E, we can still distinguish E from a random permutation as follows:

1. Request a right pair for e_0 as input, X_0, X_1, s.t. $X_0 \oplus X_1 = \Delta_0$.
2. After e_0, these have been encrypted to Y_0, Y_1, and have the relationship $Y_0 \oplus Y_1 = \Delta_1$. After e_1, these have been encrypted to Z_0, Z_1, with no predictable differential relationship.
3. We make two right pairs for e_1^{-1} from this pair, by requesting the decryption of $Z_2 = Z_0 \oplus \Delta_1$ and $Z_3 = Z_1 \oplus \Delta_1$.
4. Z_2, Z_3 are decrypted to Y_2, Y_3, with the relationships $Y_2 \oplus Y_0 = \Delta_0$ and $Y_3 \oplus Y_1 = \Delta_0$.
5. This determines the differential relationship between Y_2 and Y_3.

$$Y_0 \oplus Y_1 = \Delta_1; Y_0 \oplus Y_2 = \Delta_0; Y_1 \oplus Y_3 = \Delta_0$$
$$\text{thus: } Y_2 \oplus Y_3 = \Delta_1$$

6. Because $Y_2 \oplus Y_3 = \Delta_1$, we have a right output pair for e_0. Since we're dealing with a normal differential, we know that the differential must go the other direction, so that $X_2 \oplus X_3 = \Delta_0$.

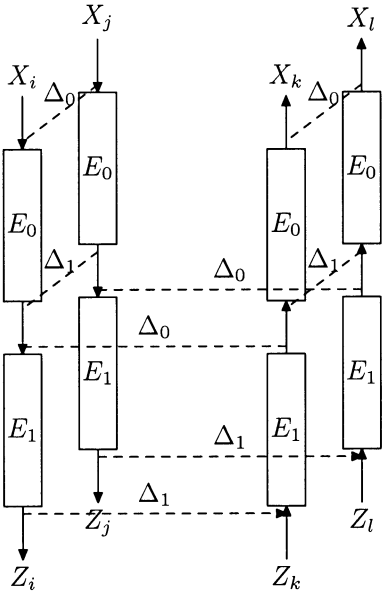


Fig. 1. The Boomerang Attack

7. A single instance of this working for a random permutation has probability 2^{-128} for a 128-bit block cipher, so this can be used very effectively to distinguish E from a random permutation.

Note that the really powerful thing about this attack is that it allows a differential type attack to work against a cipher for which there is no good differential through the whole cipher.

2.4 Turning the Boomerang into a Chosen-Plaintext Attack

We can combine the ideas of the inside-out and boomerang attacks to turn the boomerang attack into an attack that requires only chosen plaintext queries; unlike the boomerang attack, this new attack does not require adaptive-chosen-ciphertext queries. Note that we're dealing with the same cipher $E(X) = e_1(e_0(X))$.

Suppose we are dealing with a 128-bit block cipher, and are thus dealing with 128-bit differences. We request 2^{65} random chosen plaintext pairs X_{2i}, X_{2i+1} such that $X_{2i} \oplus X_{2i+1} = \Delta_0$. Since we are dealing with probability one differentials, this gives us 2^{65} pairs Y_{2i}, Y_{2i+1} such that $Y_{2i} \oplus Y_{2i+1} = \Delta_1$. We expect about two pairs (i, j) for which $Y_{2i} \oplus Y_{2j} = \Delta_0$. When we have an i, j pair of this kind

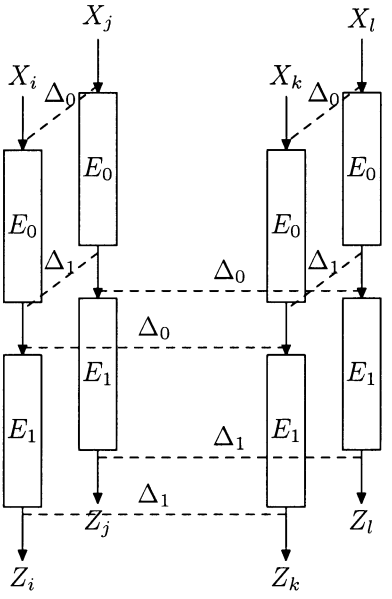


Fig. 2. The Boomerang-Amplifier Attack

we have the boomerang property:

$$Y_{2i} \oplus Y_{2i+1} = \Delta_1; Y_{2j} \oplus Y_{2j+1} = \Delta_1; Y_{2i} \oplus Y_{2j} = \Delta_0$$
$$\text{thus: } Y_{2i+1} \oplus Y_{2j+1} = \Delta_0$$
$$\text{and so: } Z_{2i} \oplus Z_{2j} = \Delta_1; Z_{2i+1} \oplus Z_{2j+1} = \Delta_1$$

There are about 2^{129} possible i, j pairs. The probability that any given pair will satisfy the last two equations is 2^{-256}. We can thus use the above technique to distinguish E from a random permutation.

We call this attack a boomerang-amplifier attack, because the boomerang structure "amplifies" the effect of a low-probability event ($Y_{2i} \oplus Y_{2j} = \Delta_0$) enough that it can be easily detected. By contrast, the inside-out attack amplifies such a low-probability event by detecting a signal from both input and output of the cipher.

2.5 Comparing Boomerangs and Boomerang Amplifiers

It is worthwhile to compare boomerang-amplifiers with the original boomerangs, in terms of attacks made possible. All else being equal, boomerangs require far fewer total queries than boomerang amplifiers, because in a boomerang-amplifier attack, we have to request enough right input pairs to expect the internal collision property that allows us to get the desired relationship between the pairs. Thus,

in the example above, we're trading off 2^{65} chosen plaintext queries for two adaptive chosen ciphertext queries.

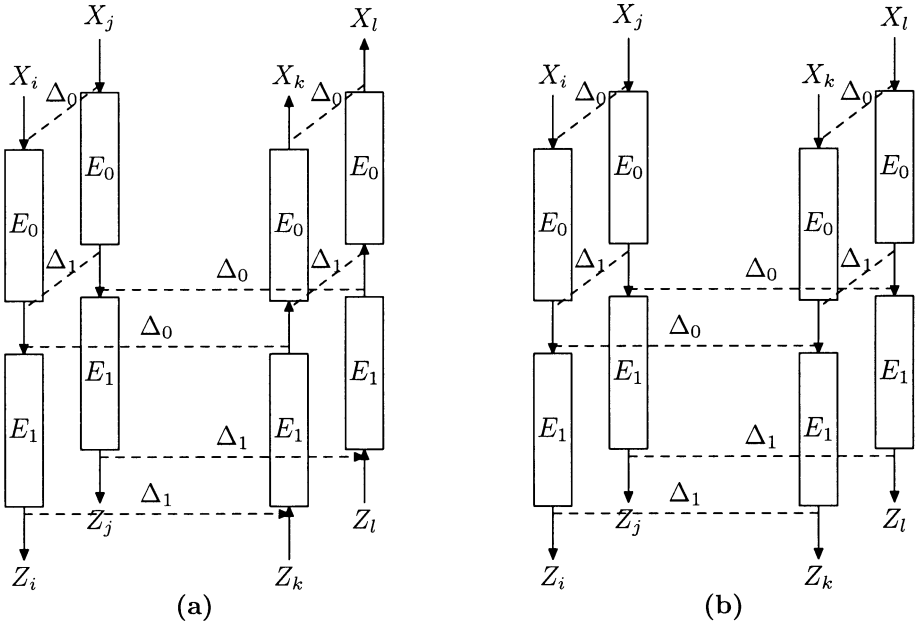


Fig. 3. Comparing Boomerangs and Boomerang-Amplifiers; Note Direction of Arrows

This might not look all that useful. However, there are three things that make this useful in many attacks:

1. When mounting an attack, we often need to guess key material on one end or the other of the cipher. With a chosen-plaintext/adaptive chosen-ciphertext attack model, we must increase our number of requested plaintexts/ciphertexts when we have to guess key material on either end. With a chosen-plaintext only attack, we can guess key material at the end of the cipher, and not have to increase our number of chosen plaintexts requested.
2. We can use the boomerang-amplifier, not just on pairs, but on k-tuples of texts.
3. We can use the boomerang-amplifier to get pairs (or k-tuples) of texts though part of the cipher, and then cover the remaining rounds of the cipher with truncated differentials or differential-linear characteristics. In this way, we can use truncated differentials that specify only a small part of the block, and couldn't be used with a standard boomerang attack.

2.6 Boomerang-Amplifiers with 3-Tuples

Consider a method to send 3-tuples of texts through e_0 with the property that when $X_{0,1,2}$ are chosen properly, Y_i, Y_i^*, Y_i^{**} have some simple XOR relationship, such as $Y_i^* = Y_i \oplus t^*$ and $Y_i^{**} = Y_i \oplus t^{**}$. We can carry out a boomerang-amplifier attack using these 3-tuples. Consider:

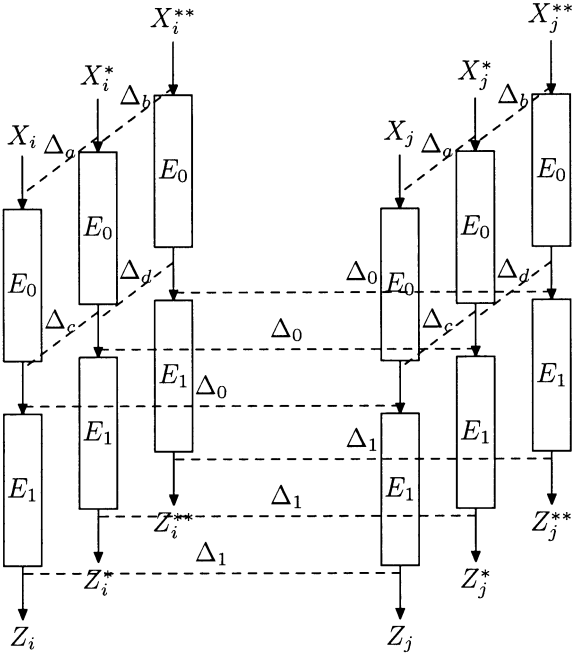


Fig. 4. A Boomerang-Amplifier with 3-Tuples

1. Request 2^{65} such 3-tuples, X_i, X_i^*, X_i^{**}.
2. We expect about one instance where $Y_i \oplus Y_j = \Delta_0$, by the birthday paradox.
3. For that instance, we get *three* right pairs through the cipher:

$$(Z_i, Z_j); (Z_i^*, Z_j^*); (Z_i^{**}, Z_j^{**})$$

This happens because:

$$Y_i \oplus Y_j = \Delta_0; Y_i^* = Y_i \oplus t^*; Y_j^* = Y_j \oplus t^*$$
$$\text{Thus: } Y_i^* \oplus Y_j^* = Y_i \oplus Y_j \oplus t^* \oplus t^* = Y_i \oplus Y_j = \Delta_0$$

The same idea works in a boomerang attack, but is of no apparent use. However, in a boomerang-amplifier attack, we are able to use this trick to get through more rounds. Because we're doing a chosen-plaintext attack, we can look for patterns that are apparent from k-tuples of right pairs, even through several more rounds of the cipher. Conceptually, we can use this to increase the "amplification" on the attack.

2.7 Detecting the Effects of the Boomerang-Amplifiers

A boomerang 4-tuple $(X_{i,j,k,l}, Z_{i,j,k,l})$ has the property that $X_{i,j}$ and $X_{k,l}$ are right input pairs, and $Z_{i,k}$ and $Z_{j,l}$ are right output pairs. When we mount a boomerang-amplifier attack, we know that $X_{i,j}$ and $X_{k,l}$ are right pairs, because we have chosen them to be right pairs. We thus detect a right pair of pairs by noting that $Z_{i,k}$ and $Z_{j,l}$ are right output pairs.

There is a straightforward trick for speeding up searches for these right pairs of pairs among large sets of pairs. It is easy to see that

$$\begin{aligned} Z_i \oplus Z_k &= \Delta_2 \\ Z_j \oplus Z_l &= \Delta_2 \\ Z_i \oplus Z_j &= Z_i \oplus Z_l \oplus \Delta_2 \\ &= Z_k \oplus Z_l \end{aligned}$$

This means that we can build a sorted list of the output pairs from a large set of input pairs, in which each entry in the list contains the XOR of the ciphertexts from one pair. When both $Z_{i,k}$ and $Z_{j,l}$ are right output pairs, then $Z_i \oplus Z_j = Z_k \oplus Z_l$. A variant of this technique works even when the output differences are truncated differences; in that case, the differential relationship works only in the fixed bits of the difference.

When we apply boomerang-amplifiers to larger blocks of texts, we must find other ways to detect them. For example, below we describe an attack in which we concatenate a differential-linear characteristic with probability one to the differences after several rounds resulting from right pairs in the middle. This means that each right pair in the middle gives us one bit that has to take on a certain value; we request batches of over 256 entries, and look for right pairs of batches. Each batch of N texts results, as ciphertext, in an N-bit string. We build a sorted list of these strings, and find the matches, which must come from right pairs of batches.

Alternatively, we can simply build sorted lists of all the Z_i and all the $Z_i \oplus \Delta_1$, and then look for matches.

3 Amplified Boomerangs and the MARS Core

3.1 The MARS Core

MARS [BCD+98] is a heterogenous, target-heavy, unbalanced Feistel network (to use the nomenclature from [SK96]). At the center are 16 core rounds: eight

forward core rounds and eight backward core rounds. Surrounding those rounds are 16 keyless mixing rounds: eight forward mixing rounds before the core, and eight backward mixing rounds after the core. Surrounding that are two whitening rounds: one at the beginning of the cipher and another at the end.

The design of MARS is fundamentally new; the whitening and unkeyed mixing rounds jacket the cryptographic core rounds, making any attacks on the core rounds more difficult. The core rounds, by contrast, must provide the underlying cryptographic strength, by giving a reasonable approximation of a random permutation family. This can be seen by considering the fact that if the cryptographic core is replaced by a known random permutation, there is a trivial meet-in-the-middle attack. Related results can be found in [KS00]. It thus makes sense to consider the strength of the MARS core rounds alone, in order to try to evaluate the ultimate strength of MARS against various kinds of attack.

Both forward and backward core rounds use the same E function, which takes one 32-bit input and two subkey words, and provides three 32-bit words. The only difference between forward and backward rounds is the order in which the outputs are combined with the words. For more details of the MARS core rounds, see [BCD+98,KS00].

Notation and Conventions. The notation we use here for considering the MARS core rounds differs from the notation used in [BCD+98]. One forward core round may be represented as follows:

1. $(A_{i-1}, B_{i-1}, C_{i-1}, D_{i-1})$ are the four 32-bit words input into round i.
2. (A_i, B_i, C_i, D_i) are the four 32-bit output words from round i.
3. The two round keys are K_i^+ and $K_i^\times$.
4. K_i^+ has 32 bits of entropy; $K_i^\times$ has just under 30 bits of entropy, because it is always forced to be congruent to 3 modulo 4. The full round thus has 62 bits of key material.
5. One forward core round may be expressed as:

$$
\begin{aligned}
F_i^\times &= ((A_{i-1} \lll 13) \times K_i^\times) \lll 10 \\
F_i^+ &= (A_{i-1} + K_i^+) \lll (F_i^\times \ggg 5) \\
F_i^s &= (S[\text{low nine bits } (A_{i-1} + K_i^+)] \oplus (F_i^\times \lll 5) \oplus F_i^\times) \lll F_i^\times \\
D_i &= A_{i-1} \lll 13 \\
A_i &= B_{i-1} + F_i^s \\
B_i &= C_{i-1} + F_i^+ \\
C_i &= D_{i-1} \oplus F_i^\times
\end{aligned}
$$

We find this notation easier to follow than the original MARS paper's notation, and so will use it for the remainder of this paper. On pages 12–13 of the original MARS submission document, these values are referred to as follows:

- F^s is referred to as either *out*1 or L.
- F^+ is referred to as either *out*2 or M.
- $F^\times$ is referred to as either *out*3 or R.
- K^+ is referred to as K.
- $K^\times$ is referred to as K'.

Useful Properties. The MARS core function is difficult to attack for many rounds. However, there are a number of useful properties which we have been able to use to good effect in analyzing the cipher. These include:

1. For $A = 0$, $F^{\times}$ is always zero, F^s is S[low nine bits of K^+], and F^+ is always K^+.
2. There is a three-round truncated differential $(0, 0, 0, \delta_0) \rightarrow (\delta_1, 0, 0, 0)$ with probability one.
3. The multiply operation can only propagate changes in its input toward its higher-order bits. This leads to a number of probability one linear characteristics for small numbers of rounds, one of which we will use in an attack, below.

Showing Differentials and Truncated Differentials. In the remainder of this section, we will represent differentials as 4-tuples of 32-bit words, such as $(0, 0, 0, 2^{31})$. We will represent truncated differentials in the same way, but with variables replacing differences that are allowed to take on many different values, as in $(0, 0, 0, x)$, which represents the a difference of zero in the first three words of the block, and an unknown but nonzero difference in the last word of the block. Differences within a word will be shown as sequences of known zero or one bits, "dont care" bits, or variables. Thus, to show a word whose high 15 bits are zeros, whose 16th bit may take on either value, and whose low 16 bits we don't care about, we would use $(0^{15}, a, ?^{16})$. If we have a difference in which the first three words are zero, and the last word has its high 15 bits zero, its 16th bit able to take on either value, and with all other bits unimportant for the difference, we would show this as $(0, 0, 0, (0^{15}, a, ?^{16}))$.

Sending a Counter Through Three Rounds of the MARS Core. Here is one additional property of the MARS core that turns out to be very useful: We can send a counter through three rounds of MARS core by choosing our inputs correctly.

Consider a set of inputs $(0, t, u, i)$, where t, u are random 32-bit words, and i is a counter that takes on all 2^{32} possible values. After three rounds, this goes to $(i + v, w, x, y)$, where v, w, x, y are all 32-bit functions of t, u.

When t, u are held constant, we get a set of 2^{32} texts whose A_3 values run through all possible values in sequence. We don't know the specific values, but for any additive difference, δ, we can identify 2^{32} pairs with that difference after three rounds.

Similarly, we can choose a restricted set of i values. For example, if we run i through all values between 0 and 255, we get 128 different pairs with a difference of 128. This will prove useful in later attacks.

In 3.3, the property described here will be exploited to mount a far more powerful attack on the MARS core.

3.2 Attacking MARS with a Simple Amplified Boomerang

The MARS core has a three-round differential characteristic with probability one: $(0, 0, 0, 2^{31}) \rightarrow (2^{31}, 0, 0, 0)$. It also has a three-round truncated differential with probability one: $(0, 0, 0, \alpha) \rightarrow (\beta, 0, 0, 0)$. We can use these two characteristics to mount a boomerang-amplifier attack through six rounds of the cipher.

We request about 2^{48} input pairs X_{2i}, X_{2i+1}, such that $X_{2i} \oplus X_{2i+1} = (0, 0, 0, 2^{31})$. As we described above, these pairs are encrypted to pairs Y_{2i}, Y_{2i+1} such that $Y_{2i} \oplus Y_{2i+1} = (2^{31}, 0, 0, 0)$.

After we have about 2^{48} such pairs, we expect to have one pair (i, j) such that $Y_{2i} \oplus Y_{2j} = (0, 0, 0, \alpha)$ for any $\alpha \neq 0$. For this pair, we can solve for $Y_{2i+1} \oplus Y_{2j+1}$; we get $(0, 0, 0, \alpha)$ in that difference as well.

We thus get two right input pairs after round three, and two right output pairs from round six. Among 2^{48} right input pairs, we have about 2^{95} pairs of right input pairs. Since the probability of randomly getting an output pair with difference $(\beta, 0, 0, 0)$ for any $\beta \neq 0$ is 2^{-96}, and since we expect one 4-tuple with two such output pairs, we will easily distinguish six rounds of MARS core from a random permutation.

3.3 A Boomerang-Amplified Differential-Linear Attack on Eleven Rounds

We can combine the above idea with two other properties of the MARS core to build a much more powerful attack, which is properly classified as either a boomerang-amplified differential-linear attack, or a boomerang-amplified truncated-differential attack. Our attack consists of the following:

1. We choose inputs so that we get batches of 280 texts following the pattern

$$(s, t, u, v), (s + 1, t, u, v), (s + 2, t, u, v), ..., (s + 279, t, u, v)$$

   We use the technique described in section 3.1 to do this.
2. We request 2^{57} such batches, so that we can expect a pair of batches with a truncated difference $\delta = (0, 0, 0, (?^{13}, 0^{17}, ?^2))$ between each corresponding pair of texts in the batch. That is, after round three, the *first* elements of one pair of batches have the following relationship:

$$v^* - v = \delta$$
$$s^* - s = t^* - t = u^* - u = 0$$

   It follows by simple arithmetic that the ith elements of the pair of batches have difference $(0, 0, 0, \delta)$. Note that this is an *additive* difference.
3. When we get this right pair of batches, we get 280 pairs with this additive difference into the output of round three. This means we get 280 right pairs in the output of round six.
4. We are then able to cover two more rounds with a linear characteristic with $p \approx 1$.

5. We guess our way past the last two full rounds, and past part of the third round from the end. This gives us a part of the output from the eighth round. This requires 62 bits for each full round, and 39 bits for the partial guess, thus a total of 163 bits guessed.
6. For each of the 2^{57} batches, we extract the linear characteristic from the output of the eighth round of each of the 280 texts. We thus get a 280-bit string from each batch, for each partial key guess.
7. For each key guess, we build a sorted list of these 280-bit strings, and find the match. The probability of finding a match of a 280-bit string in 2^{57} texts is about 2^{-167}; we are trying to find a match for 2^{163} different key guesses. Thus, we expect to have no false matches, and we are vanishingly unlikely to have more than a small number of false matches.

Getting the Right Difference Between the Batches. Consider only the first element in each batch. There are 2^{57} such elements. We need one pair such that after round three, it has a truncated difference of $(0, 0, 0, (?^{13}, 0^{17}, ?^2))$; that is, with all but 15 of its bits zeros. The probability of a random pair of texts having this difference is 2^{-113}. There are about 2^{113} pairs of these texts, and so we expect this difference to happen about once.

When this difference happens between the first elements of a pair of batches, it is easy to see that it must also occur between the second elements, and the third, and so on. We thus get a right pair of batches, yielding 280 right pairs in the output from round three.

The Linear Characteristic/Truncated Differential. Consider a single pair of texts with the difference $(0, 0, 0, (?^{12}, a, 0^{17}, ?^2))$ at the output of round three, where a is a single unknown bit. We care only about bit a in our attack. When we originally developed this attack, we thought in terms of a differential-linear characteristic with probability one. In this case, this is equivalent to a truncated differential with only one bit specified. Here, we describe this in terms of the truncated differential attack.

First, we note that with probability of very nearly one $(1 - 2^{-17})$, a will be unchanged for any given pair of corresponding texts after round six, in the output truncated difference $((?^{12}, a, ?^{19}), 0, 0, 0)$. The probability that this doesn't change for any corresponding pair of texts in the right pair of batches is about 0.998. (The number of times a changes is binomially distributed, with $n = 280, p = 2^{-17}$.)

In the seventh round, bit a is rotated to the low-order bit input into the multiply operation; this means that bit ten of $F_7^{\times}$ is a. This is a "backward" core round, so the output from the seventh round leaves us with truncated difference $((?^{21}, a, ?^{10}), ?, ?, ?)$. In the next round, the leftmost word is changed only by being rotated. We thus get the following truncated difference in the output from the eighth round: $(?, ?, ?, (?^7, a, ?^{23}))$.

This single bit appears as a constant in the differences for all 280 corresponding pairs of the right pair of batches.

Guessing Key Material. We guess the low nine bits of K_9^+, and the full 30-bit $K_9^\times$. This allows us to backtrack through f_9^s, and thus to recover bit a for all the texts. We then guess our way past rounds ten and eleven by guessing 62 bits for each round.

Required Resources for the Attack. We attack eleven rounds of MARS core; five forward and six backward.

We request 2^{57} batches of 280 texts each, and thus must request a total of about 2^{65} chosen plaintexts. We get a batch of 2^{65} ciphertexts back out, which we must store and then examine to mount our attack.

We must guess our way past two full MARS core rounds (at a cost of 62 bits each), plus part of a third MARS core round (at a cost of 39 bits). We thus must guess a total of 163 bits. We must try 2^{163} times to find a match, once per key guess.

For each key guess, we have to do the following steps:

1. Do the partial decryption on about 2^{65} ciphertexts, and extract one bit per ciphertext, at a cost of about 2^{65} partial decryptions.
2. Arrange the resulting bits as 2^{57} 280-bit strings.
3. Sort the 280-bit strings, at a cost of about $57 \times 2^{57} \approx 2^{63}$ swapping operations' work.

To simplify our analysis, we assume that the work of doing the 2^{65} partial decryptions dominates the work of sorting the 280-bit strings; we thus require about $2^{163} \times 2^{65} = 2^{228}$ partial decryptions' work to mount the attack.

The total memory required for the attack is 2^{70} bytes, sufficient for 2^{66} ciphertexts.

4 Boomerang-Amplifiers and Serpent

Serpent is a 32-round AES-candidate block cipher proposed by Ross Anderson, Eli Biham, and Lars Knudsen [ABK98]. In this section we show how one can apply the amplified boomerang technique to reduced-round Serpent variants. Additional attacks against reduced-round Serpent can be found in [KKS00]. Unlike those used in MARS, the differentials used in our attacks on Serpent do not have probability one.

4.1 Description of Serpent

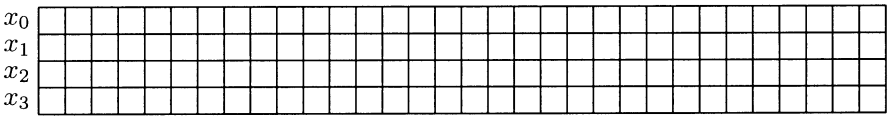
Serpent is a 32-round block cipher operating on 128-bit blocks. The Serpent design documentation describes two versions of Serpent: a bitsliced version and a non-bitsliced version. Both versions are functionally equivalent. The difference between the bitsliced and non-bitsliced versions of Serpent is the way in which the data is represented internally. In this document we shall only consider the bitsliced version of Serpent.

Let B_i represent Serpent's intermediate state prior to the ith round of encryption; B_0 is the plaintext and B_{32} is the ciphertext. Let K_i represent the 128

bit ith round subkey and let S_i represent the application of the ith round S-box. Let L represent Serpent's linear transformation (see [ABK98] for details). Then the Serpent round function is defined as:

$$\begin{aligned} X_i &\leftarrow B_i \oplus K_i \\ Y_i &\leftarrow S_i(X_i) \\ B_{i+1} &\leftarrow L(Y_i) \qquad i = 0, \ldots, 30 \\ B_{i+1} &\leftarrow Y_i \oplus K_{i+1} \; i = 31 \end{aligned}$$

In the bitsliced version of Serpent, one can consider each 128-bit block X_i as the concatenation of four 32-bit words x_0, x_1, x_2, and x_3. Pictorially, one can represent Serpent's internal state X_i using diagrams such as the following: Serpent uses eight S-boxes S_i where the indices i are reduced modulo 8; e.g.,



$S_0 = S_8 = S_{16} = S_{24}$. Each S-box takes four input bits and produces four output bits. The input and output nibbles of the S-boxes correspond to the columns in the preceding diagram (where the most significant bits of the nibbles are the bits in the word x_3).

We use X' to represent an XOR difference between two values X and X^*.

4.2 Distinguishing Seven Rounds of Serpent

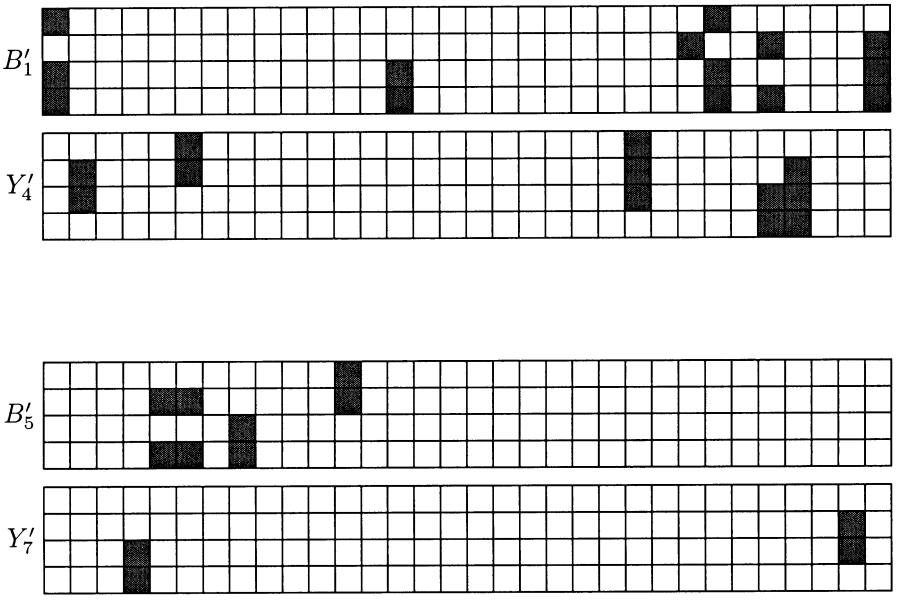
Let us consider a seven-round Serpent variant $E_1 \circ E_0$ where E_0 corresponds to rounds one through four of Serpent and E_1 corresponds to rounds five through seven of Serpent.

There are several relatively high-probability characteristics through both halves of this seven round Serpent variant. Let us consider two such characteristics $B'_1 \rightarrow Y'_4$

and $B'_5 \rightarrow Y'_7$

where $B'_1 \rightarrow Y'_4$ is a four-round characteristic through E_0 with probability 2^{-31} and $B'_5 \rightarrow Y'_7$ is a three-round characteristic through E_1 with probability 2^{-16}. Additional information on these characteristics can be found in [KKS00].

We can immediately combine these two characteristics to form a seven round boomerang distinguishing attack requiring 2^{95} chosen plaintext queries and 2^{95} adaptive chosen ciphertext queries. Using the amplified boomerang technique, however, we can construct a chosen-plaintext only distinguishing attack requiring 2^{113} chosen plaintext queries.



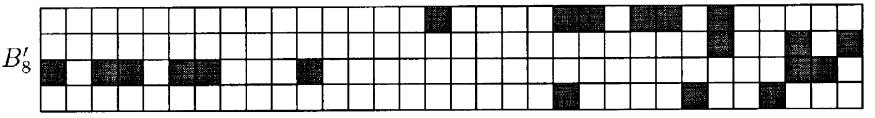
The details of the attack are as follows. We request 2^{112} plaintext pairs with our input difference B'_1. After encrypting with the first half of the cipher E_0, we expect roughly 2^{81} pairs to satisfy the first characteristic $B'_1 \rightarrow Y'_4$. There are approximately 2^{161} ways to form quartets using these 2^{81} pairs. We expect there to be approximately 2^{33} quartets (Y_4^0, Y_4^1) and (Y_4^2, Y_4^3) such that $Y_4^0 \oplus Y_4^2 = L^{-1}(B'_5)$. However, because (Y_4^0, Y_4^1) and (Y_4^2, Y_4^3) are right pairs for the first half of the cipher, and $Y_4^0 \oplus Y_4^1 = Y_4^2 \oplus Y_4^3 = Y'_4$, we have that $Y_4^1 \oplus Y_4^3$ must also equal $L^{-1}(B'_5)$. In effect, the randomly occurring difference between Y_4^0 and Y_4^2 has been "amplified" to include Y_4^1 and Y_4^3.

At the input to E_1 we expect approximately 2^{33} quartets with a difference of (B'_5, B'_5) between the pairs. This gives us approximately two quartets after the seventh round with an output difference of (Y'_7, Y'_7) across the pairs. We can identify these quartets by intelligently hashing our original ciphertext pairs with our ciphertext pairs XORed with (Y'_7, Y'_7) and noting those pairs that collide. In a random distribution, the probability of observing a single occurrence of the cross-pair difference (Y'_7, Y'_7) is approximately 2^{-33}.

4.3 Eight-Round Serpent Key Recovery Attack

We can extend the previous distinguishing attack to an eight-round key-recovery attack requiring 2^{113} chosen plaintext pairs, 2^{119} bytes of memory, and work equivalent to approximately 2^{179} eight-round Serpent encryptions. This attack covers rounds one through eight of Serpent. If we apply the linear transformation

L to Y'_7 we get the difference:



Given 2^{113} chosen plaintext pairs with our input difference B'_1, we expect approximately eight pairs of pairs with cross-pair difference (Y'_7, Y'_7) after the seventh round. This corresponds to eight pairs of pairs with difference (B'_8, B'_8) entering the eighth round. By guessing 68 bits of Serpent's last round subkey K_9, we can peel off the last round and perform our previous distinguishing attack.

5 Conclusions

In this paper, we have introduced a new kind of attack that is closely related to the boomerang attack of Wagner [Wag99]. We have applied this attack to reduced-round versions of both the MARS core and of Serpent.

5.1 Related Attacks

There is a set of related attacks, including the miss-in-the-middle, boomerang, and amplified boomerang attacks, which deal with pairs of differentials that reach to the same point in the intermediate state of the cipher, but which don't connect as needed for conventional attacks. A miss-in-the-middle attack gives us three texts in the middle that *can't* fit a certain second-order differential relationship. A boomerang or amplified boomerang gives us a 4-tuple that does fit a certain second-order differential relationship. However, these are different from standard higher-order differential attacks in that the second-order differential relationship doesn't continue to exist through multiple rounds. Instead, this relationship serves only to connect pairs of texts with a first-order differential relationship in the middle of the cipher.

In some sense, this is similar to the way structures are used with higher-order differential relationships, in order to use first-order differentials more efficiently. Thus, we might have two good differentials through the first round that will get us to our desired input difference:

$$\Delta_0 \rightarrow \Delta_1$$
$$\Delta_2 \rightarrow \Delta_1$$

It's a common trick to request $X, X \oplus \Delta_0, X \oplus \Delta_2, X \oplus \Delta_0 \oplus \Delta_2$, which will give us four right pairs, two for each differential, for the price of only four texts. We're requesting a 4-tuple of texts with a second-order differential relationship,

but it doesn't propagate past the first round. This is a second-order differential attack in exactly the same sense as the boomerang and amplified boomerang attacks are second-order differential attacks.

The boomerang and boomerang-amplifier attacks are, in some sense, a new way of building structures of texts *inside* the middle of the cipher. These structures have a second order relationship, which allows the four texts to take part in four right pairs. However, in the boomerang and boomerang-amplifier attacks, two of the right pairs go through the first half of the cipher, and two go through the second half of the cipher.

5.2 Applying the Attack to Other Algorithms

We have not yet applied this attack to other algorithms besides MARS and Serpent. However, there is a common thread to situations in which the attack works: We need to be able to get through many rounds with some differential that has reasonably high probability. In the case of the MARS core, there are probability one differentials for three rounds, simply due to the structure of the cipher. In the case of Serpent, the probability of a differential characteristic is primarily a function of the number of S-boxes in which the difference is active across all rounds of the characteristic. Differences spread out, so that it is possible to find reasonably good characteristics for three or four rounds at a time, but not for larger numbers of rounds, since by then the differences have spread to include nearly all the S-boxes.

Applying this general class of attack to other ciphers will be the subject of ongoing research.

It is worth repeating, however, that this technique does not endanger either Serpent or MARS. In the case of MARS, the cryptographic core is jacketed with additional unkeyed mixing and key addition/subtraction layers, which would make chosen-plaintext attacks like this one enormously more expensive (more expensive than exhaustive search), even if our attack worked against the full cryptographic core. In the case of Serpent, the large number of rounds prevents our attack from working against the full cipher.

Acknowledgements. The "extended Twofish team" met for two week-long cryptanalysis retreats during Fall 1999, once in San Jose and again in San Diego. This paper is a result of those collaborations. Our analysis of MARS and Serpent has very much been a team effort, with everybody commenting on all aspects. The authors would like to thank Niels Ferguson, Mike Stay, David Wagner, and Doug Whiting for useful conversations and comments on these attacks, and for the great time we had together. The authors also wish to thank the reviewers, for useful comments and suggestions, and Beth Friedman, for copyediting the final paper.

References

ABK98. R. Anderson, E. Biham, and L. Knudsen, "Serpent: A Proposal for the Advanced Encryption Standard," NIST AES Proposal, Jun 1998.

BCD+98. C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S.M. Matyas, L. O'Connor, M. Peyravian, D. Safford, and N. Zunic, "MARS — A Candidate Cipher for AES," NIST AES Proposal, Jun 98.

BS93. E. Biham and A. Shamir, *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag, 1993.

Knu95b. L.R. Knudsen, "Truncated and Higher Order Differentials," *Fast Software Encryption, 2nd International Workshop Proceedings*, Springer-Verlag, 1995, pp. 196–211.

KS00. J. Kelsey and B. Schneier, "MARS Attacks! Cryptanalyzing Reduced-Round Variants of MARS," *Third AES Candidate Conference*, to appear.

KKS00. T. Kohno, J. Kelsey, and B. Schneier, "Preliminary Cryptanalysis of Reduced-Round Serpent," *Third AES Candidate Conference*, to appear.

LH94. S. Langford and M. Hellman, "Differential-Linear Cryptanalysis," *Advances in Cryptology — CRYPTO '94*, Springer-Verlag, 1994.

Mat94. M. Matsui, "Linear Cryptanalysis Method for DES Cipher," *Advances in Cryptology — EUROCRYPT '93 Proceedings*, Springer-Verlag, 1994, pp. 386–397.

NIST97a. National Institute of Standards and Technology, "Announcing Development of a Federal Information Standard for Advanced Encryption Standard," *Federal Register*, v. 62, n. 1, 2 Jan 1997, pp. 93–94.

NIST97b. National Institute of Standards and Technology, "Announcing Request for Candidate Algorithm Nominations for the Advanced Encryption Standard (AES)," *Federal Register*, v. 62, n. 117, 12 Sep 1997, pp. 48051–48058.

SK96. B. Schneier and J. Kelsey, "Unbalanced Feistel Networks and Block Cipher Design," *Fast Software Encryption, 3rd International Workshop Proceedings*, Springer-Verlag, 1996, pp. 121–144.

Wag99. D. Wagner, "The Boomerang Attack," *Fast Software Encryption, 6th International Workshop*, Springer-Verlag, 1999, pp. 156–170.