# High Probability Linear Hulls in Q

Liam Keliher[1], Henk Meijer[1], and Stafford Tavares[2]

[1] Department of Computing and Information Science
Queen's University at Kingston, Ontario, Canada, K7L 3N6
{keliher,henk}@cs.queensu.ca
[2] Department of Electrical and Computer Engineering
Queen's University at Kingston, Ontario, Canada, K7L 3N6
tavares@ee.queensu.ca

**Abstract.** In this paper, we demonstrate that the linear hull effect is significant for the Q cipher. The designer of Q performs preliminary linear cryptanalysis by discussing linear characteristics involving only a single active bit at each stage [13]. We present a simple algorithm which combines all such linear characteristics with identical first and last masks into a linear hull. The expected linear probability of the best such linear hull over 7.5 rounds (8 full rounds minus the first $S$ substitution) is $2^{-90.1}$. In contrast, the best known expected differential probability over the same rounds is $2^{-110.5}$ [2]. Choosing a sequence of linear hulls, we get a straightforward attack which can recover a 128-bit key with success rate 98.4%, using $2^{97}$ known $\langle \text{plaintext}, \text{ciphertext} \rangle$ pairs and no trial encryptions.

**Keywords:** Q, linear cryptanalysis, linear hulls

## 1 Introduction

Q is a block cipher submitted to the NESSIE project by Leslie 'Mack' McBride [13]. Q has a straightforward SPN structure with s-boxes based on those in Rijndael [4] (the AES selection) and Serpent [1]. (The Serpent-like s-boxes can be implemented with an efficient bit-slicing technique [1]; for clarity, we will use the equivalent representation which involves bitwise permutations before and after application of these s-boxes).

The structure of the s-boxes and linear transformations allows the construction of linear characteristics with one active bit at each stage. We refer to such linear characteristics as *restricted characteristics.* Nyberg's *linear hull* concept [14] (the counterpart of differentials in differential cryptanalysis[10]) allows us to combine a large number of restricted linear characteristics into a single linear hull which can then be used to attack the cipher.

We present a simple algorithm for calculating the *expected linear probability* (ELP) of the linear hulls formed by this method over various numbers of rounds. The best such linear hull over 7.5 rounds (8 full rounds minus the first Rijndael s-box substitution) has ELP $2^{-90.1}$. In contrast, the best known expected differential probability over the same rounds is $2^{-110.5}$ [2].

## 2 Description of Q

### 2.1 Basic Components

The Q cipher is based on the substitution-permutation network (SPN) architecture [6,7,9]. Q has a block size of $N = 128$ bits. Q uses three different s-boxes, one $8 \times 8$ s-box named $S$ (this is the Rijndael s-box [4,5]), and two $4 \times 4$ s-boxes named $A$ and $B$ ($B$ is used in Serpent[1], and $A$ is "Serpent-like"). Each substitution stage uses multiple copies of a single s-box in parallel to process the 128-bit input (16 copies of $S$, or 32 copies of $A$ or $B$).

The Q key schedule generates twelve 128-bit subkeys named $KW1$, $KA$, $KB$, $K0, K1, \ldots, K7$, $KW2$. Key mixing is via bitwise XOR. See [13] for a complete description of the key schedule.

Before continuing, we need to clarify the convention used for numbering consecutive bytes and words, namely that numbering begins at 0 with the object in the lowest memory location—this is also the least significant object, since Q uses "little-endian" ordering. This convention extends to numbering the bits of bytes/nibbles, i.e., the least significant bit is numbered 0. Pictorially, numbering always increases from left to right (it follows that the bits in a 128-bit block are numbered $0 \ldots 127$, left to right).

Three linear transformations are used in the cipher. The permutation $P$ operates on a 128-bit block represented as four 32-bit words, $W_0, W_1, W_2, W_3$, as follows: $W_0$ is unchanged; $W_1$, $W_2$, and$W_3$ are right rotated by one byte, two bytes, and three bytes, respectively.

The other two linear transformations are bitwise permutations which we term PreSerpent( ) and PostSerpent( ), since they are located before and after each application of s-boxes $A$ and $B$. If we again view the 128-bit block as consisting of words $W_0, W_1, W_2, W_3$, PreSerpent( ) sends the bits of $W_0$ to the first (leftmost) input bits of the 32 identical $4 \times 4$ s-boxes, the bits of $W_1$ to the second input bits of these s-boxes, and so on. This is represented in Figure 1. PostSerpent( ) is simply the inverse of PreSerpent( ).

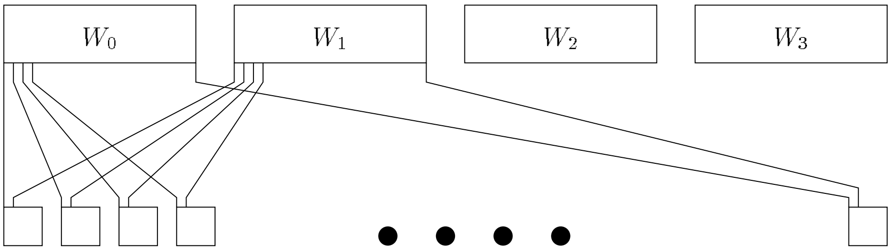

**Fig. 1.** PreSerpent( ) bitwise permutation

### 2.2 High-Level Structure

We will consider the version of Q using 8 full rounds and a 128-bit key (McBride also proposes a 9-round version for "high security applications" [13]). For $0 \le r \le 7$, round $r$ has the structure in Figure 2.

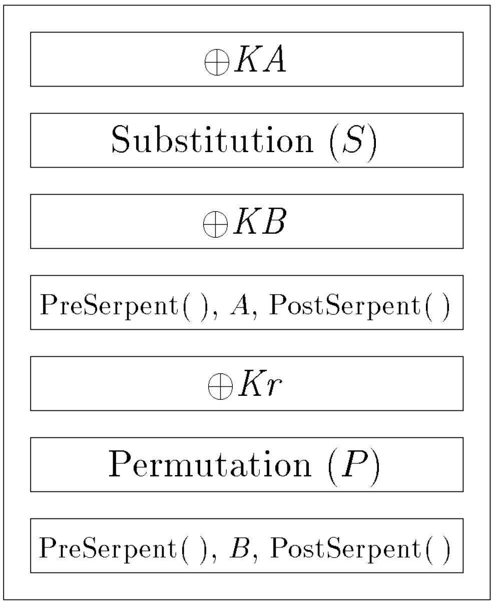
$\oplus KA$
Substitution $(S)$
$\oplus KB$
PreSerpent( ), $A$, PostSerpent( )
$\oplus Kr$
Permutation $(P)$
PreSerpent( ), $B$, PostSerpent( )

**Fig. 2.** Structure of full round

Note that a full round actually consists of three substitution stages ($S$, $A$, and $B$). The entire cipher is described by:

$$\oplus KW1,\ \text{Round0}, \ldots, \text{Round7},\ \oplus KA,\ \text{Substitution}(S),\ \oplus KB,\ \oplus KW2\ .$$

Therefore this version of Q contains a total of 25 substitution stages. The use of $KA$ and $KB$ can be viewed as making the substitution with $S$ key-dependent. However, Q also conforms to the standard SPN structure in which a subkey is XOR'd before each fixed substitution stage [9] (for Q, then, there are repeated subkeys).

## 3 Linear Probability

Given a bijective mapping $B : \{0,1\}^d \to \{0,1\}^d$, and *masks* $\mathbf{a}, \mathbf{b} \in \{0,1\}^d$, the associated *linear probability* (LP) value is defined as

$$LP(\mathbf{a}, \mathbf{b}) \overset{\text{def}}{=} \left(2 \cdot \text{Prob}\left\{\mathbf{a} \bullet \mathbf{X} = \mathbf{b} \bullet B(\mathbf{X})\right\} - 1\right)^2 ,$$

where $\mathbf{X}$ is a random variable uniformly distributed over $\{0,1\}^d$, and $\bullet$ denotes the inner product over GF(2). Note that $LP(\mathbf{a},\mathbf{b}) \in [0,1]$; nonzero LP values indicate a correlation between the input and output of $B$.

If $B$ is parameterized by a key, $\mathbf{k}$, we write $LP(\mathbf{a},\mathbf{b};\mathbf{k})$, and the expected LP (ELP) over the uniform distribution of keys is denoted

$$ELP(\mathbf{a},\mathbf{b}).$$

### 3.1 LP Values for the Q S-boxes

In what follows, we will only be interested in LP values for the s-boxes of Q for masks of binary weight 1. We give these values in Tables 1, 2, and 3. Entry $[i,j]$ is the LP value for input (output) mask with 1 in position $i$ ($j$) and all other bits equal to 0. (Recall that we number bits from left to right starting at 0, with 0 indicating least significance.) For $S$, we denote this entry $LP_S[i,j]$ (entries for $A$ and $B$ are subscripted accordingly).

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | $(\frac{6}{64})^2$ | $(\frac{0}{64})^2$ | $(\frac{7}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{2}{64})^2$ | $(\frac{2}{64})^2$ | $(\frac{6}{64})^2$ |
| 1 | $(\frac{1}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{3}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{8}{64})^2$ | $(\frac{1}{64})^2$ |
| 2 | $(\frac{4}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{3}{64})^2$ | $(\frac{3}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{8}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{1}{64})^2$ |
| 3 | $(\frac{1}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{2}{64})^2$ | $(\frac{0}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{3}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{2}{64})^2$ |
| 4 | $(\frac{6}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{3}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{5}{64})^2$ | $(\frac{0}{64})^2$ | $(\frac{4}{64})^2$ |
| 5 | $(\frac{3}{64})^2$ | $(\frac{5}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{1}{64})^2$ | $(\frac{0}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{6}{64})^2$ |
| 6 | $(\frac{2}{64})^2$ | $(\frac{2}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{8}{64})^2$ | $(\frac{3}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{2}{64})^2$ |
| 7 | $(\frac{6}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{8}{64})^2$ | $(\frac{7}{64})^2$ | $(\frac{4}{64})^2$ | $(\frac{6}{64})^2$ | $(\frac{2}{64})^2$ | $(\frac{2}{64})^2$ |

**Table 1.** LP values for s-box S

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | $\frac{1}{16}$ | $\frac{1}{16}$ | $\frac{1}{16}$ |
| 1 | 0 | 0 | $\frac{1}{16}$ | 0 |
| 2 | 0 | $\frac{1}{16}$ | 0 | $\frac{1}{16}$ |
| 3 | 0 | 0 | 0 | 0 |

**Table 2.** LP values for s-box A

|  | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | $\frac{1}{16}$ | 0 | 0 | 0 |
| 1 | $\frac{1}{16}$ | $\frac{1}{16}$ | 0 | $\frac{1}{16}$ |
| 2 | 0 | $\frac{1}{16}$ | $\frac{1}{16}$ | 0 |
| 3 | 0 | 0 | $\frac{1}{16}$ | $\frac{1}{16}$ |

**Table 3.** LP values for s-box B

## 4 Linear Cryptanalysis

Linear cryptanalysis (LC) is a known-plaintext attack due to Matsui [11]; we use the version known as Algorithm 2. We do not give the details of LC here (see [9] for a description of LC applied to SPNs). It suffices to say that the attacker attempts to find one or more of the outermost subkeys by choosing input/output masks $\mathbf{a}, \mathbf{b} \in \{0,1\}^N$ (recall that $N$ is the block size) for a core subset of the substitution stages such that the corresponding value $LP(\mathbf{a}, \mathbf{b}; \mathbf{k})$ is relatively large.

In general it is infeasible to compute $LP(\mathbf{a}, \mathbf{b}; \mathbf{k})$ directly, so researchers have adopted the practice of using the expected value $ELP(\mathbf{a}, \mathbf{b})$ (over all independent and uniformly random subkeys) [15, 8, 9]. If

$$\mathcal{N}_L = \frac{c}{ELP(\mathbf{a}, \mathbf{b})}$$

is the number of known $\langle$plaintext, ciphertext$\rangle$ pairs used by the attacker (this is called the *data complexity*), the success rate of Algorithm 2 is given in Table 4. Note that this is the same as Table 3 in [11], except that the constant values differ by a factor of 4, since Matsui uses *bias* values, not LP values.[1]

| $c$ | 8 | 16 | 32 | 64 |
|---|---|---|---|---|
| Success rate | 48.6% | 78.5% | 96.7% | 99.9% |

**Table 4.** Success rates for Algorithm 2

### 4.1 Linear Characteristics and Linear Hulls

An efficient first approximation to the value $ELP(\mathbf{a}, \mathbf{b})$ can be found using *linear characteristics* (or simply *characteristics*). Let $T$ denote the number of core

[1] The corresponding table in [9] has an error, in that the constants have *not* been multiplied by 4 to reflect the use of LP values.

substitution stages over which ELP values are required. A $T$-stage characteristic is a $(T+1)$-tuple $\Omega = \langle \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_T, \mathbf{a}_{T+1} \rangle$. We consider $\mathbf{a}_t$ and $\mathbf{a}_{t+1}$ as input and output masks, respectively, for the $t^{\text{th}}$ substitution stage $(1 \leq t \leq T)$. The *expected linear characteristic probability* of $\Omega$, denoted $ELCP(\Omega)$, is defined as

$$ELCP(\Omega) \overset{\text{def}}{=} \prod_{t=1}^{T} ELP(\mathbf{a}_t, \mathbf{a}_{t+1}) \, . \tag{1}$$

Note that $\mathbf{a}_t$ and $\mathbf{a}_{t+1}$ determine input/output masks for each s-box in round $t$. Those s-boxes having nonzero input and output masks are called *active.* Moreover, the bits in any mask which are equal to 1 are called *active bits.* If a characteristic, $\Omega$, results in any s-box having zero input mask and nonzero output mask, or vice versa, it is easy to show that the ELP for that substitution stage is 0, and therefore $ELCP(\Omega) = 0$ by (1). For simplicity, we exclude all such characteristics from further consideration.

The attacker typically finds the characteristic, $\Omega = \langle \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_T, \mathbf{a}_{T+1} \rangle$, whose ELCP is maximal (called the *best* characteristic [12]), and then sets $\mathbf{a} = \mathbf{a}_1$, $\mathbf{b} = \mathbf{a}_{T+1}$, and uses the approximation

$$ELP(\mathbf{a}, \mathbf{b}) \approx ELCP(\Omega) \, . \tag{2}$$

However, more careful analysis requires the concept of *linear hulls,* due to Nyberg [14]. Given masks $\mathbf{a}$ and $\mathbf{b}$ for the $T$ stages under consideration, the corresponding linear hull, denoted $ALH(\mathbf{a}, \mathbf{b})$,[2] is the set of all $T$-stage characteristics whose first mask is $\mathbf{a}$ and whose final mask is $\mathbf{b}$. Nyberg then shows that

$$ELP(\mathbf{a}, \mathbf{b}) = \sum_{\Omega \in ALH(\mathbf{a}, \mathbf{b})} ELCP(\Omega).$$

It follows that (2) does not hold in general. The difference between the ELCP of a characteristic and the ELP value it is used to approximate is called the *linear hull effect.* If analysis is based on characteristics instead of linear hulls, the linear hull effect may result in an overestimation of the data complexity for a given success rate.

## 5 Computing Linear Hulls for Q

McBride performs preliminary linear cryptanalysis of Q by considering the formation of characteristics in which every $N$-bit mask contains only a single 1 [13]. We call these *restricted characteristics.* McBride estimates that the ELCP of the best characteristic over 8 rounds is in the range of $2^{-118}$ (i.e., bias value $= 2^{-60}$).

[2] Nyberg originally used the term *approximate linear hull*, hence the abbreviation ALH.

However, as we show below, it is straightforward to combine restricted characteristics into linear hulls[3] for which the ELP value is much higher than this, i.e., the linear hull effect is significant for Q.

In order to form linear hulls over $T$ core stages, our algorithm uses a 3-dimensional data structure DS[ ] of size $128 \times (T+1) \times 128$, in which each entry is a record of two values: an integer Count, and a real value ELP.[4] After running the algorithm, $\mathrm{DS}[i, t, j]$ contains information about the linear hull over stages $1 \ldots t$ whose first (last) mask contains a single 1 in position $i$ ($j$); namely, the Count field is the number of restricted characteristics in the linear hull, and the ELP field is the sum of the ELCP values of those restricted characteristics.

For the presentation of the algorithm, we strip off the first and last $S$-substitution stages, so $T = 23$. The algorithm is given in Figure 3 and Figure 4. (The pseudocode for subroutine ApplyB( ) is omitted, as it is symmetric to that for ApplyA( )—simply replace $\mathrm{LP}_A$ with $\mathrm{LP}_B$. Also, the pseudocode for subroutine PostSerpent( ) is omitted, as it is simply the inverse of PreSerpent( ).) Note that we use the shorthand $x \mathrel{+}= y$ to mean $x \leftarrow x + y$. The values which are important for our attack on Q will be stored in the entries $\mathrm{DS}[i, 23, j]$.

**Theorem 1.** *If* DS[ ] *is filled using the algorithm in Figures 3 and 4, then for* $0 \leq i, j \leq 127$ *and* $0 \leq t \leq 23$, $\mathrm{DS}[i, t, j]$.Count *is the number of restricted characteristics over the first t of the 23 substitution stages whose first (last) mask contains a single 1 in position i (j), and for which ELCP* $> 0$*; and* $\mathrm{DS}[i, t, j]$.ELP *is the sum of the ELCPs of these characteristics.*

*Proof.* Let $0 \leq i \leq 127$ be fixed. The theorem is easily proven using induction on $t$. Trivially, the base case ($t = 0$) is made true for all $j$ by the first For loop in the main program (Figure 3). We assume the statement holds for some $t \geq 0$ and demonstrate its truth for $t + 1$. Note that the truth of the statement is not affected by the linear transformations (bitwise permutations) in Q; we simply perform the "bookkeeping" of permuting the elements of $\mathrm{DS}[i, t, \cdot]$ accordingly.[5] Therefore, we need only consider the effect of the $(t+1)^{\mathrm{st}}$ substitution stage on $\mathrm{DS}[i, t, \cdot]$, and without loss of generality we can limit our consideration to substitution using $S$. Further, without loss of generality we will consider only the effect of the first (leftmost) copy of $S$, denoted $S_0$. The inputs/outputs for $S_0$

[3] We are abusing terminology somewhat by using the term *linear hull* to refer to a collection of *restricted* characteristics, omitting all other characteristics in the linear hull. However, this should not be a source of confusion.

[4] Our approach has strong similarities to the construction of differentials for Q by Biham et al. [2]; however, whereas Biham et al. use a linear algebraic approach, we opt for an algorithmic description.

[5] This works because given a mask *before* a linear transformation in Q, the corresponding mask *after* the linear transformation is obtained by processing the mask through the linear transformation. This applies to Q because all linear transformations are bitwise permutations. However, this does not hold in general—for an arbitrary linear transformation represented as a binary matrix, *output* masks are transformed to *input* masks via multiplication by the *transpose* of the linear transformation [3].

are bits $0 \ldots 7$ of the respective blocks (recall that bits in a block are numbered $0 \ldots 127$).

Let $\tilde{\jmath} \in \{0, \ldots, 7\}$. Consider all restricted characteristics over the first $(t+1)$ substitution stages whose first (last) mask has bit $i$ ($\tilde{\jmath}$) active. Clearly all these characteristics make $S_0$ active. Therefore, *prior* to substitution stage $(t+1)$, the active bit for each of these characteristics is in $\{0, \ldots, 7\}$. It follows that

$$\mathrm{DS}[i, t+1, \tilde{\jmath}].\mathrm{Count} \; = \; \sum_{j=0}^{7} \mathrm{DS}[i, t, j].\mathrm{Count}, \qquad (3)$$

with one proviso: if $\mathrm{LP}_S[j, \tilde{\jmath}] = 0$, then extending any $t$-stage restricted characteristic enumerated by $\mathrm{DS}[i, t, j]$.Count (for $0 \leq j \leq 7$) to $(t+1)$ stages will produce a value ELCP $= 0$ (by (1)). Therefore, we omit all such $j$ by modifying (3) as follows:

$$\mathrm{DS}[i, t+1, \tilde{\jmath}].\mathrm{Count} \; = \sum_{\substack{0 \leq j \leq 7 \\ \mathrm{LP}_S[j, \tilde{\jmath}] \neq 0}} \mathrm{DS}[i, t, j].\mathrm{Count} \qquad (4)$$

(this is done via the If statement in subroutine ApplyS( )). It is easily seen that $\mathrm{DS}[i, t+1, \tilde{\jmath}]$.ELP is correctly assigned the sum of the ELCP values of all characteristics enumerated by (4).

### 5.1 Computational Results

We ran our algorithm for varying numbers of rounds by modifying the main program in Figure 3 appropriately. The best ELP values found are given in Table 5. For comparison, we include Table 6, which contains the corresponding best expected differential probability (EDP) values from [2]. The ELP values represent a minimum improvement in the exponent of approximately 17 relative to [2]; the improvement in the exponent is 20.4 for the case which is of primary interest to us: 7 full rounds with $A+B$ prepended, hereafter denoted $A+B+7$.

| Number of rounds | Full rounds only | With additional $S$ appended | With additional $A+B$ prepended |
|---|---|---|---|
| 6 | $2^{-72.3}$ | $2^{-77.2}$ | $2^{-78.8}$ |
| 7 | $2^{-83.7}$ | $2^{-88.6}$ | $2^{-90.1}$ |
| 8 | $2^{-95.1}$ | $2^{-100.0}$ | $2^{-101.5}$ |
| 9 | $2^{-106.4}$ | $2^{-111.3}$ | - |

**Table 5.** Best ELP values

```
Initialize all Count and LP entries in DS[ ] to 0
For $i = 0$ to 127
    DS$[i, 0, i]$.Count $\leftarrow 1$
    DS$[i, 0, i]$.ELP $\leftarrow 1$

For $i = 0$ to 127
    $t \leftarrow 0$
    ApplyA $(i, t)$;   $t$ += 1
    Permute $(i, t)$
    ApplyB $(i, t)$;   $t$ += 1

    For Round = 1 to 7
        ApplyS $(i, t)$;   $t$ += 1
        ApplyA $(i, t)$;   $t$ += 1
        Permute $(i, t)$
        ApplyB $(i, t)$;   $t$ += 1
```

```
Subroutine ApplyS $(i, t)$
    $\mathcal{J} \leftarrow \{j : \text{DS}[i, t, j].\text{Count} > 0\}$
    For $j \in \mathcal{J}$
        BoxIndex $\leftarrow j$ div 8
        InBit $\leftarrow j$ mod 8
        For OutBit = 0 to 7
            If   LP$_S$[InBit, OutBit] $\neq 0$
                $\tilde{\jmath} \leftarrow 8 \times$ BoxIndex + OutBit
                DS$[i, t+1, \tilde{\jmath}]$.Count += DS$[i, t, j]$.Count
                DS$[i, t+1, \tilde{\jmath}]$.ELP += DS$[i, t, j]$.ELP $\times$ LP$_S$[InBit, OutBit]
```

**Fig. 3.** Pseudocode for computation of linear hulls over 23 core stages

### 5.2 Recovering the Full Key

Each linear hull over $A+B+7$ can be used to attack the key bytes associated with two $S$ s-boxes, one in the first substitution stage, and one in the last. We guess the corresponding bytes of $KW1$ and $KA$ for the former, and the corresponding bytes of $KW2$ and $KB$ for the latter, requiring a total of $2^{32}$ counters. We opt for a 99.9% success rate by using $\frac{64}{ELP}$ known $\langle$plaintext, ciphertext$\rangle$ pairs (see Table 4). Using a series of 16 linear hulls, we can systematically recover each

```
Subroutine ApplyA $(i, t)$

    PreSerpent $(i, t)$
    $\mathcal{J} \leftarrow \{j : \mathrm{DS}[i,t,j].\mathrm{Count} > 0\}$
    For $j \in \mathcal{J}$
        BoxIndex $\leftarrow j$ div 4
        InBit $\leftarrow j$ mod 4
        For OutBit $= 0$ to 3
            If $\mathrm{LP}_A[\mathrm{InBit}, \mathrm{OutBit}] \neq 0$
                $\tilde{\jmath} \leftarrow 4 \times \mathrm{BoxIndex} + \mathrm{OutBit}$
                $\mathrm{DS}[i, t+1, \tilde{\jmath}].\mathrm{Count}$ += $\mathrm{DS}[i,t,j].\mathrm{Count}$
                $\mathrm{DS}[i, t+1, \tilde{\jmath}].\mathrm{ELP}$ += $\mathrm{DS}[i,t,j].\mathrm{ELP} \times \mathrm{LP}_A[\mathrm{InBit}, \mathrm{OutBit}]$
    PostSerpent $(i, t+1)$

Subroutine PreSerpent $(i, t)$

    For $j = 0$ to 127
        Temp$[j] \leftarrow \mathrm{DS}[i,t,j]$
    For $j = 0$ to 127
        $\tilde{\jmath} \leftarrow 4 \times (j \bmod 32) + (j \operatorname{div} 32)$
        $\mathrm{DS}[i,t,\tilde{\jmath}] \leftarrow \mathrm{Temp}[j]$

Subroutine Permute $(i, t)$

    Partition $\mathrm{DS}[i,t,\cdot]$ into 4 "words" of size 32 $(W_0, W_1, W_2, W_3)$:
        $W_s \leftarrow \langle \mathrm{DS}[i,t,32s], \ldots, \mathrm{DS}[i,t,32s+31] \rangle$, for $s = 0 \ldots 3$
    Leave $W_0$ unchanged
    Right rotate $W_1$ by 8, $W_2$ by 16, $W_3$ by 24
```

**Fig. 4.** Pseudocode for other subroutines

byte of $KW2$. Assuming the success rates of the 16 attacks are independent, the overall success rate is $.999^{16} \approx 98.4\%$. Once $KW2$ is known, running the key schedule backwards yields the other subkeys (and the original key).

Using our algorithm, we found the best linear hull for attacking each byte of $KW2$. These are given in Table 7. Since the *smallest* of the 16 ELP values is

| Number of rounds | Full rounds only | With additional $S$ appended | With additional $A+B$ prepended |
|---|---|---|---|
| 6 | $2^{-92.9}$ | $2^{-105.35}$ | $2^{-95.5}$ |
| 7 | $2^{-107.9}$ | $2^{-120.35}$ | $2^{-110.5}$ |
| 8 | $2^{-122.9}$ | $2^{-135.35}$ | $2^{-125.5}$ |
| 9 | $2^{-137.9}$ | $2^{-150.35}$ | - |

**Table 6.** Corresponding best EDP values from Biham et al. [2]

approximately $2^{-91}$, the overall data complexity is $2^{97}$. We do not perform key ranking [11], so no trial encryptions are required.

## 6 Conclusion

We have considered the resistance of Q to linear cryptanalysis based on linear hulls. We present a straightforward algorithm which combines all characteristics consisting of masks with binary weight 1 (termed restricted characteristics) into the corresponding linear hulls, and we compute the expected linear probability (ELP) of each such linear hull (actually, since we limit our consideration to restricted characteristics, the values we obtain are lower bounds on the corresponding ELP values). The ELP of the best such linear hull over 7.5 rounds (8 full rounds minus the first $S$ substitution) is $2^{-90.1}$, a significant improvement over the best known expected differential probability (EDP) for the same rounds, namely $2^{-110.5}$ [2]. We can use the linear hulls found by our algorithm to recover a 128-bit key with success rate 98.4%, using $2^{97}$ known $\langle\text{plaintext}, \text{ciphertext}\rangle$ pairs and no trial encryptions.

There are a number of reasons for the success of our approach. First, each of the three s-boxes in Q has multiple nonzero LP values corresponding to input/output masks of weight 1. In contrast, one of the design criteria for $A$ and $B$ was that no single-bit input *difference* can produce a single-bit output difference [13]. This is why our ELP values are superior to the EDP values in [2]. Secondly, the linear transformations in Q have low diffusion, allowing a mask with weight 1 to be transformed into a mask also having weight 1. Finally, the cryptanalyst's job is made easier by the fact that finding a 128-bit subkey such as $KW2$ allows the original key to be recovered. This could be avoided by building a one-way property into the key schedule.

## References

1. R. Anderson, E. Biham, and L. Knudsen, *Serpent: A flexible block cipher with maximum assurance,* The First Advanced Encryption Standard Candidate Conference, Proceedings, Ventura, California, August 1998.

| Byte of $KW2$ | Active bits (input,output) | ELP | Number of characteristics in linear hull |
|---|---|---|---|
| 0 | (31, 3) | $2^{-91.1}$ | 94,726,326 |
| 1 | (7, 11) | $2^{-91.1}$ | 94,726,326 |
| 2 | (15, 19) | $2^{-91.1}$ | 94,726,326 |
| 3 | (23, 27) | $2^{-91.1}$ | 94,726,326 |
| 4 | (31, 35) | $2^{-90.1}$ | 191,795,706 |
| 5 | (7, 43) | $2^{-90.1}$ | 191,795,706 |
| 6 | (15, 51) | $2^{-90.1}$ | 191,795,706 |
| 7 | (23, 59) | $2^{-90.1}$ | 191,795,706 |
| 8 | (23, 67) | $2^{-90.2}$ | 188,281,125 |
| 9 | (31, 75) | $2^{-90.2}$ | 188,281,125 |
| 10 | (7, 83) | $2^{-90.2}$ | 188,281,125 |
| 11 | (15, 91) | $2^{-90.2}$ | 188,281,125 |
| 12 | (7, 99) | $2^{-90.2}$ | 183,092,934 |
| 13 | (15, 107) | $2^{-90.2}$ | 183,092,934 |
| 14 | (7, 115) | $2^{-90.2}$ | 183,092,934 |
| 15 | (15, 123) | $2^{-90.2}$ | 183,092,934 |

**Table 7.** Best linear hulls for attacking bytes of $KW2$

2. E. Biham, V. Furman, M. Misztal, and V. Rijmen, *Differential cryptanalysis of Q,* Fast Software Encryption (FSE 2001), To be published in Lecture Notes in Computer Science, Springer-Verlag.
3. J. Daemen, R. Govaerts, and J. Vandewalle, *Correlation matrices,* Fast Software Encryption : Second International Workshop, LNCS 1008, Springer-Verlag, pp. 275–285, 1995.
4. J. Daemen and V. Rijmen, *AES proposal: Rijndael,* `http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf`.
5. J. Daemen and V. Rijmen, *AES (Rijndael) reference code in ANSI C,* `http://csrc.nist.gov/encryption/aes/rijndael/`.
6. H. Feistel, *Cryptography and computer privacy,* Scientific American, Vol. 228, No. 5, pp. 15–23, May 1973.
7. H.M. Heys, *The design of substitution-permutation network ciphers resistant to cryptanalysis,* Ph.D. Thesis, Queen's University, Kingston, Canada, 1994.
8. S. Hong, S. Lee, J. Lim, J. Sung, and D. Cheon, *Provable security against differential and linear cryptanalysis for the SPN structure,* Fast Software Encryption (FSE 2000), LNCS 1978, Springer-Verlag, pp. 273–283, 2001.
9. L. Keliher, H. Meijer, and S. Tavares, *New Method for Upper Bounding the Maximum Average Linear Hull Probability for SPNs,* Advances in Cryptology—EUROCRYPT 2001, LNCS 2045, Springer-Verlag, pp. 420–436, 2001.

10. X. Lai, J. Massey, and S. Murphy, *Markov ciphers and differential cryptanalysis,* Advances in Cryptology—EUROCRYPT'91, LNCS 547, Springer-Verlag, pp. 17–38, 1991.
11. M. Matsui, *Linear cryptanalysis method for DES cipher,* Advances in Cryptology—EUROCRYPT'93, LNCS 765, Springer-Verlag, pp. 386–397, 1994.
12. M. Matsui, *On correlation between the order of s-boxes and the strength of DES,* Advances in Cryptology—EUROCRYPT'94, LNCS 950, Springer-Verlag, pp. 366–375, 1995.
13. L. McBride, *Q: A Proposal for NESSIE v2.00,* First NESSIE Workshop, Leuven, Belgium, November 2000.
14. K. Nyberg, *Linear approximation of block ciphers,* Advances in Cryptology—EUROCRYPT'94, LNCS 950, Springer-Verlag, pp. 439–444, 1995.
15. S. Vaudenay, *On the security of CS-Cipher,* Fast Software Encryption (FSE'99), LNCS 1636, Springer-Verlag, pp. 260–274, 1999.