Key-Schedule Cryptanalysis of DEAL

John Kelsey and Bruce Schneier

Counterpane Systems
{kelsey,schneier}@counterpane.com
101 E. Minnehaha Pkwy
Minneapolis, MN 55419

Abstract. DEAL is a six- or eight-round Luby-Rackoff cipher that uses DES as its round function, with allowed key lengths of 128, 192, and 256 bits. In this paper, we discuss two new results on the DEAL key schedule. First, we discuss the existence of equivalent keys for all three key lengths; pairs of equivalent keys in DEAL-128 require about 2^{64} DES encryptions to find, while equivalent keys in DEAL-192 and DEAL-256 require only six or eight DES encryptions to find. Second, we discuss a new related-key attack on DEAL-192 and DEAL-256. This attack requires 2^{33} related key queries, the same 3 plaintexts encrypted under each key, and may be implemented with a variety of time-memory tradeoffs; Given 3×2^{69} bytes of memory, the attack requires 2^{113} DES encryptions, and given 3×2^{45} bytes of memory, the attack requires 2^{137} DES encryptions. We conclude with some questions raised by the analysis.

1 Introduction

In June 1998 the National Institute of Standards and Technology (NIST) received fifteen candidate algorithms for the Advanced Encryption Standard (AES). The AES would eventually replace DES as a federal encryption standard, and hopefully would become a world-wide encryption standard as well.

One of the hardest aspects of cipher design is the key schedule. Numerous AES submissions have been attacked through their key schedule: SAFER+ [CMK+98] in [KSW99], Crypton [Lim98] in [Bor99], DFC [GGH+98] in [Cop98a,Cop98b], Frog [GLC98] in [WFS99], HPC [Sch98] in [Wag99,DBP+99], Magenta [JH98] in [BBF+99], MARS [BCD+98] and RC6 [RRS+98] in [Saa99]. These attacks have ranged from finding equivalent keys to weak key classes to related-key differential attacks [Bih94,KSW96,KSW97], and have generally not been serious. Still, equivalent or related keys can make the cipher unusable as a hash function (for example, in Davies-Meyer feed forward mode [Win84]), and can reduce the effective keyspace of the cipher[Knu93]; related-key differential attacks can cause vulnerabilities in applications where related-key queries are legitimate [KSW96,KSW97]. Weak key classes can mean that a percentage of the keys are vulnerable to attack.

One of the submissions was DEAL (Data Encryption Algorithm with Larger blocks) [Knu98]. Intended as the conservative choice, DEAL was designed to leverage the cryptographic confidence in DES while creating a new cipher with a 128-bit block and key lengths of 128-, 192-, and 256-bits. In this paper, we refer to DEAL with an n-bit key as DEAL-n. Thus, we have DEAL-128, DEAL-192, and DEAL-256.

In [Luc99], an attack was presented for DEAL-192, with a number of possible tradeoffs given between number of chosen-plaintext queries, and amount of work done for the attack. The best attack in terms of computational resources requires 2^{56} chosen plaintexts, about 2^{146} encryptions' worth of work, and about 2^{63} memory locations. With 2^{40} bits (2^{37} bytes) of memory, Lucks' best attack on DEAL-192 requires 2^{33} chosen plaintexts, and work equivalent to about 6×2^{189} DES encryptions (about 2^{189} DEAL encryptions).

In [Knu98], a number of impractical attacks are discussed on DEAL-192. There is a straightforward meet-in-the-middle attack on DEAL-192 requiring about 2^{168} work and 2^{173} bytes of memory, requiring only three known plaintexts. The memory requirements are totally unreasonable, and trading off time for memory does not yield an attack with reasonable memory requirements and less work than brute-forcing the key. There is also a general attack on 6-round Feis-

tel ciphers with bijective F-functions, based on a 5-round impossible truncated differential. Applying this attack to DEAL-192 gives an attack with 2^{119} work, 2^{70} chosen plaintexts, and 2^{68} bytes of memory. The chosen-plaintext requirements make this attack totally impractical. No attacks on DEAL-256 faster than exhaustive search were discussed.

1.1 Our Results

In this paper, we present the following results against DEAL:

- Equivalent keys for DEAL-192 and DEAL-256, with an algorithm to find them. The algorithm requires about six DES encryptions to find a set of 256 equivalent DEAL-192 keys, and eight DES encryptions to find a set of 256 equivalent DEAL-256 keys.
- Equivalent keys for DEAL-128, with an algorithm to find them. The algorithm requires about 2^{64} work to find a pair of equivalent keys.
- A related-key attack on DEAL-192 and DEAL-256, requiring three plaintexts under 2^{33} keys with a certain relationship, 3×2^{45} bytes of memory, and about 2^{137} DEAL encryptions' work, to find the last two rounds' subkeys for DEAL-192 and DEAL-256. (With more memory, this can be made faster.)
- A number of possible extensions to these attacks. DEAL-192 can be peeled down to four rounds, and then Biham's attack on four-round Ladder-DES can be applied[Bih97]; DEAL-256 can be peeled down to six rounds, and then Lucks' attack on six-round DEAL-192 can be applied. Alternatively, 64 bits can be recovered from the original key, and the remainder brute-force searched.

Importance of the Results These results have both practical and theoretical interest.

DEAL is likely to see some use in the future in real-world systems. DEAL is an AES candidate, but even if it is not accepted as an AES finalist, it will almost certainly see some use. The general idea behind DEAL is a sound one, and has been proposed several times before [Rit94,Bih97]. As pointed out by Outerbridge at the first AES conference, widespread availability of DES hardware in many different environments makes DEAL relatively easy to implement in many different environments, at very low cost. A system designer in need of a 128-bit block cipher, and in possession of lots of DES-enabled devices, might do well to choose an algorithm like DEAL. (Certainly, he would be better off doing this than trying to design his own cipher from DES.)

In real-world use, the equivalent keys of DEAL have important practical implications– they make many standard hashing modes, e.g. Davies-Meyer mode, unsafe to use[1].

The related-key attacks are probably somewhat less practical, but may still be important in some applications. These attacks have the effect of peeling off the last two rounds of DEAL at the cost of about 2^{137} DEAL encryptions of work, using about 3×2^{45} bytes of memory, and requiring the same three plaintexts be encrypted under 2^{33} related keys. There are various time-memory tradeoffs available.

In the presence of 3×2^{69} bytes of random-access storage, the attack will run with about 2^{113} work, again recovering the last two round subkeys[2]. At that point, Biham's attack on 4-round Ladder-DES [Bih97] can be mounted, requiring another 2^{33} chosen plaintexts (under only one key) and 2^{88} time. The whole attack thus takes about 2^{113} work, 3×2^{69} bytes of random-access storage, 3 known plaintexts encrypted under 2^{33} related keys, and 2^{32} chosen plaintexts under one of those keys, to be selected after the rest of the attack has run its course. This compares with the best previously known attack, which required 2^{119} work, 2^{64} memory, and 2^{70} chosen plaintexts.

On a theoretical level, our results demonstrate an important fact: It is widely assumed that a key schedule that uses strong cryptographic components will, in practice, not be vulnerable to cryptanalysis. This assumption has motivated a number of ciphers' key schedules, including those of Khufu [Mer91], Blowfish [Sch94], and SEAL [RC98]. This assumption, unfortunately, isn't always true. In

[1] In [Knu98], it is noted that the slow key schedule of DEAL makes it a poor choice for hashing applications.

[2] This assumes that 3×2^{69} bytes of random-access storage can be found and used efficiently–in practice, this attack is of no practical significance, though variant attacks with lower memory requirements may be.

DEAL, a strong cipher is used in an apparently-reasonable way to process key material. However, the method used leaves the cipher vulnerable to related-key cryptanalysis, as well as allowing the existence of equivalent keys.

1.2 Guide to the Rest of the Paper

The rest of this paper is organized as follows: We first discuss the DEAL cipher and key schedule in the level of detail required for our attacks. We then discuss equivalent keys in DEAL-192 and DEAL-256, both how to find them and how many there appear to be. Next, we discuss equivalent keys in DEAL-128. After that, we discuss related-key differential attacks on DEAL-192 and DEAL-256. We conclude with a summary of our results, and some questions raised by them.

2 The DEAL Cipher and Key Schedule

DEAL is a cipher designed originally by Lars Knudsen [Knu98] and submitted for the AES by Richard Outerbridge. DEAL uses the DES as the round function of a larger balanced Feistel cipher in a Luby-Rackoff construction [LR88].

DEAL works as follows:

Let A, B be the left and right 64-bit halves of the input block, respectively. Let $R_{0..N-1}$ be the round subkeys, which are 64 bit blocks that are used as 56-bit DES keys, by ignoring the parity bits. Encryption is as follows: (Here, we show 8 rounds.)

$$\begin{aligned}
A &= A \oplus E_{R_0}(B) \\
B &= B \oplus E_{R_1}(A) \\
A &= A \oplus E_{R_2}(B) \\
B &= B \oplus E_{R_3}(A) \\
A &= A \oplus E_{R_4}(B) \\
B &= B \oplus E_{R_5}(A) \\
A &= A \oplus E_{R_6}(B) \\
B &= B \oplus E_{R_7}(A).
\end{aligned}$$

DEAL has 6 rounds for 128- and 192-bit keys, and 8 rounds for 256-bit keys. The key schedule works as follows, where $E(X)$ means X encrypted under a constant key used only for key scheduling, and $K_{0..3}$ are the four 64-bit blocks that make up a 256-bit key. (The key schedules for DEAL-192 and DEAL-128 are very similar to the key schedule shown below for DEAL-256, but with only six round keys generated, and only three or two 64-bit blocks of input key material.)

$$\begin{aligned}
R_0 &= E(K_0) \\
R_1 &= E(K_1 \oplus R_0) \\
R_2 &= E(K_2 \oplus R_1) \\
R_3 &= E(K_3 \oplus R_2) \\
R_4 &= E(K_0 \oplus R_3 \oplus 1) \\
R_5 &= E(K_1 \oplus R_4 \oplus 2) \\
R_6 &= E(K_2 \oplus R_5 \oplus 4) \\
R_7 &= E(K_3 \oplus R_6 \oplus 8).
\end{aligned}$$

The R_i values are used only as DES keys, and so their parity bits are ignored. This turns out to be very important for our analysis.

3 Equivalent Keys in DEAL-192 and DEAL-256

An encryption algorithm has *equivalent keys* when there are two or more keys, K, K^*, such that $K \neq K^*$ but $E_K(X) = E_{K^*}(X)$ for all X. Equivalent keys can reduce the effective keyspace of an algorithm in some cases, and if pairs of keys can be efficiently found, render the encryption algorithm unsafe to use in hashing modes.

We have an algorithm for finding sets of 256 equivalent keys in DEAL. For a special class of weak keys consisting of 2^{-64} of all keys of length 192 or 256, it is always possible to find sets of 256 equivalent keys. Further, an efficient algorithm exists to find weak keys of this type. Equivalent keys also exist for DEAL-128, but a very different algorithm is needed to find these keys, and they are discussed in the next section.

3.1 The Algorithm to Find Sets of Equivalent Keys

Consider the DEAL key schedule again:

$$R_0 = E(K_0)$$

$$
\begin{aligned}
R_1 &= E(K_1 \oplus R_0) \\
R_2 &= E(K_2 \oplus R_1) \\
R_3 &= E(K_3 \oplus R_2) \\
R_4 &= E(K_0 \oplus R_3 \oplus 1) \\
R_5 &= E(K_1 \oplus R_4 \oplus 2) \\
R_6 &= E(K_2 \oplus R_5 \oplus 4) \\
R_7 &= E(K_3 \oplus R_6 \oplus 8).
\end{aligned}
$$

Our general strategy will be as follows:

1. Find a "weak key" such that $R_0 = R_3 \oplus 2$ for 192-bit keys, or such that $R_1 = R_5 \oplus 4$.
2. Choose Δ active only in parity bits.
3. Let:

$$
\begin{aligned}
K_0^* &= K_0 \\
K_1^* &= K_1 \\
K_2^* &= D(R_2 \oplus \Delta) \oplus R_1 \\
K_3^* &= K_3 \oplus \Delta
\end{aligned}
$$

4. The result is a sequence of round subkeys such that:

$$
\begin{aligned}
R_0 &= R_0^* \\
R_1 &= R_1^* \\
R_2 &= R_2^* \oplus \Delta \\
R_3 &= R_3^* \\
R_4 &= R_4^* \\
R_5 &= R_5^* \\
R_6 &= R_6^* \oplus \Delta \\
R_7 &= R_7^*
\end{aligned}
$$

We choose K_2^*, K_3^* as:

$$
\begin{aligned}
K_3^* &= K_3 \oplus \Delta \\
K_2^* &= D(R_2 \oplus \Delta) \oplus R_1
\end{aligned}
$$

This gives us a pair of equivalent keys:

$$(K_0, K_1, K_2, K_3)(K_0, K_1, K_2^*, K_3^*).$$

In fact, for each Δ satisfying the above-mentioned requirements, we get a key equivalent to (K_0, K_1, K_2, K_3). The result is that we get a family of 256 equivalent keys, since there are 256 Δ values (including zero) that satisfy the requirements for Δ to be active only in parity bits.)

3.2 Why It Works

Let's consider the values of subkeys between the two related keys: $(K_0, K_1, K_2, K_3), (K_0, K_1, K_2^*, K_3^*)$.

Recall that:

$$
\begin{aligned}
R_1 &= R_5 \oplus 4 \\
K_3^* &= K_3 \oplus \Delta \\
K_2^* &= D(R_2 \oplus \Delta) \oplus R_1
\end{aligned}
$$

Also, recall that $R_i \oplus \Delta$ is equivalent to R_i, so long as Δ is active only in its parity bits.

1. There is no change in K_0, K_1, so there can be no change in R_0, R_1. That is,

We know that:

$$
\begin{aligned}
K_0 &= K_0^* \\
K_1 &= K_1^*
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
R_0 &= R_0^* \\
R_1 &= R_1^*
\end{aligned}
$$

2. $R_2 = R_2^* \oplus \Delta$ because

We know that:

$$
\begin{aligned}
K_2^* &= D(R_2 \oplus \Delta) \oplus R_1 \\
R_2 &= E(K_2 \oplus R_1)
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
R_2^* &= E(K_2^* \oplus R_1^*) \\
&= E(D(R_2 \oplus \Delta) \oplus R_1 \oplus R_1) \\
&= E(D(R_2 \oplus \Delta)) \\
&= R_2 \oplus \Delta
\end{aligned}
$$

3. $R_3 = R_3^*$ because:

We know that:

$$
\begin{aligned}
K_3^* &= K_3 \oplus \Delta \\
R_3 &= E(K_3 \oplus R_2) \\
R_2^* &= R_2 \oplus \Delta
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
R_3^* &= E(K_3^* \oplus R_2^*) \\
&= E(K_3 \oplus \Delta \oplus R_2 \oplus \Delta) \\
&= E(K_3 \oplus R_2) \\
&= R_3.
\end{aligned}
$$

4. R_4 and R_5 are unchanged, (that is, $R_4 = R_4^*, R_5 = R_5^*$) because R_4, R_5 are dependent only upon $K_0, K_1, and R_3$, and we have already established that those values are all unchanged.

$$\begin{aligned} R_4 &= E(K_0 \oplus R_3 \oplus 1) \\ &= R_4^* \\ R_5 &= E(K_1 \oplus R_4 \oplus 2) \\ &= R_5^* \end{aligned}$$

5. $R_6 = R_6^* \oplus \Delta$, because

   We know that:

$$\begin{aligned} R_5 &= R_1 \oplus 4 \\ R_1 &= R_1^* \\ R_5 &= R_5^* \\ R_6 &= E(K_2 \oplus R_5 \oplus 4) \\ &= E(K_2 \oplus R_1 \oplus 4 \oplus 4) \\ &= E(K_2 \oplus R_1) \\ &= R_2 \end{aligned}$$

   Therefore:

$$\begin{aligned} R_6^* &= E(K_2^* \oplus R_5 \oplus 4) \\ &= E(K_2^* \oplus R_1) \\ &= R_2^* \\ &= R_2 \oplus \Delta \end{aligned}$$

   And thus:

$$R_6^* = R_6 \oplus \Delta$$

6. Finally, $R_7 = R_7^*$ because

   We know that:

$$\begin{aligned} R_7 &= E(K_3 \oplus R_6 \oplus 8) \\ R_6^* &= R_6 \oplus \Delta \\ K_3 &= K_3^* \oplus \Delta \end{aligned}$$

   Therefore:

$$\begin{aligned} R_7^* &= E(K_3^* \oplus R_6^* \oplus 8) \\ &= E(K_3 \oplus \Delta \oplus R_6 \oplus \Delta \oplus 8) \\ &= E(K_3 \oplus R_6 \oplus 8) \\ &= R_7 \end{aligned}$$

3.3 Effect on the DEAL Keyspace

This set of equivalent keys has essentially no effect on the size of the effective keyspace, since it applies only to such a tiny fraction (about $3 * 2^{-64}$) of special keys.

3.4 Extensions

A variant of the same algorithm works with K_1, K_2 or K_0, K_1 as the active pair of key blocks. A variant of the algorithm can be carried out against DEAL-192. Against DEAL-128, a much more complex algorithm can be used to find equivalent keys, as will be discussed later in this paper.

3.5 Efficiently Finding Equivalent Keys

The naive algorithm for finding equivalent keys would be to try about 2^{64} different keys, waiting until $R_1 = R_5 \oplus 4$. This has complexity 2^{64}, and thus is no easier than looking for a collision in a 128-bit hash function, such as might be built from DEAL in Davies-Meyer hashing mode. However, the search for a class 256 of equivalent keys can be converted to a straightforward algebra problem, as follows:

1. Choose $K_{0,1,2}$ arbitrarily.
2. Derive:

$$\begin{aligned} R_0 &= E(K_0) \\ R_1 &= E(K_1 \oplus R_0) \\ R_2 &= E(K_2 \oplus R_1) \end{aligned}$$

3. Use the requirement that $R_5 = R_1 \oplus 4$ to derive:

$$\begin{aligned} R_5 &= R_1 \oplus 4 \\ &= E(K_1 \oplus R_4 \oplus 2) \\ &\quad \text{Thus:} \\ R_4 &= D(R_5) \oplus K_1 \oplus 2 \end{aligned}$$

4. Having learned R_4, we next compute R_3, and thus K_3:

$$\begin{aligned} R_4 &= E(R_3 \oplus K_0 \oplus 1) \\ &= D(R_5) \oplus K_1 \oplus 2 \\ &\quad \text{Thus:} \\ R_3 &= D(D(R_5) \oplus K_1 \oplus 2) \oplus K_0 \oplus 1 \\ &= E(R_2 \oplus K_3) \\ &\quad \text{Thus:} \\ K_3 &= D(R_3) \oplus R_2 \\ &= D(D(D(R_5) \oplus K_1 \oplus 2) \oplus K_0 \oplus 1) \oplus R_2 \end{aligned}$$

5. With $K_{0,1,2,3}$, we now have a "weak" key.

The process is nearly identical with DEAL-192.

4 Finding Equivalent Keys in DEAL-128

In this section[3], we discuss an algorithm for finding equivalent keys in DEAL-128. Unlike the previous algorithm, this does not find classes of 256 equivalent keys, but instead pairs of equivalent keys. Also unlike the previous algorithm, this algorithm requires about 2^{64} runs of the DEAL key schedule to find a single pair of equivalent keys.

4.1 An Overview of Our Method

The goal is to find a pair of keys, K, K^*, such that $R_{0..5}$ and $R^*_{0..5}$ are all either equal or equivalent (equal in all bits except their parity bits, which will be ignored by the DES key schedule).

4.2 The Algorithm

1. For each Δ active only in parity bits:
   (a) For each K_0 value from 0 to $2^{64} - 1$:
      i. Compute $K^*_0 = D(E(K_0) \oplus \Delta)$
      ii. Compute $K_1 = D(1) \oplus E(K_0)$
      iii. Compute $K^*_1 = K_1 \oplus \Delta$
      iv. Use $K_{0,1}$ to compute $R_{0..5}$, and $K^*_{0,1}$ to compute $R^*_{0..5}$.
      v. Note that $R_{0..3}$ and $R^*_{0..3}$ are now equivalent:

$$\begin{aligned} R_0 &= R^*_0 \oplus \Delta \\ R_1 &= R^*_1 \\ R_2 &= R^*_2 \oplus \Delta \\ R_3 &= R^*_3 \end{aligned}$$

      .
      vi. Check to see whether $R_4 \oplus R^*_4 = \Delta$ This should happen with probability 2^{-64}
      vii. If so, we're done; R_5 will also equal R^*_5. If not, we must keep looking.

4.3 Why it Works

1. $R_0 = R^*_0 \oplus \Delta$ because

   We know that:

$$\begin{aligned} K^*_0 &= D(E(K_0) \oplus \Delta) \\ R_0 &= E(K_0) \end{aligned}$$

   Therefore:

$$\begin{aligned} R^*_0 &= E(K^*_0) \\ &= E(D(E(K_0) \oplus \Delta)) \\ &= E(K0) \oplus \Delta \\ &= R_0 \oplus \Delta \end{aligned}$$

2. $R^*_1 = R_1$ because:

   We know that:

$$\begin{aligned} K^*_1 &= K_1 \oplus \Delta \\ R^*_0 &= R_0 \oplus \Delta \\ R_1 &= E(R_0 \oplus K_1) \end{aligned}$$

   Therefore:

$$\begin{aligned} R^*_1 &= E(R^*_0 \oplus K^*_1) \\ &= E(R_0 \oplus \Delta \oplus K_1 \oplus \Delta) \\ &= E(R_0 \oplus K_1) \\ &= R_1 \end{aligned}$$

3. $R_1 = 1$, because

   We know that:

$$\begin{aligned} K_1 &= D(1) \oplus E(K_0) \\ &= D(1) \oplus R_0 \end{aligned}$$

   Therefore:

$$\begin{aligned} R_1 &= E(R_0 \oplus K_1) \\ &= E(R_0 \oplus E(K_0) \oplus D(1)) \\ &= E(R_0 \oplus R_0 \oplus D(1)) \\ &= E(D(1)) \\ &= 1 \end{aligned}$$

4. $R_1 = 1$ is necessary so that $R^*_2 = R_2 \oplus \Delta$:

   We know that:

$$\begin{aligned} R^*_0 &= E(K^*_0) \\ &= R^*_0 \\ R_1 &= 1 \\ &= R^*_1 \\ R_2 &= E(R_1 \oplus 1 \oplus K_0) \\ &= E(K_0) \\ &= R_0 \end{aligned}$$

   Therefore:

$$\begin{aligned} R^*_2 &= E(R^*_1 \oplus 1 \oplus K^*_0) \\ &= E(R_1 \oplus 1 \oplus K^*_0) \\ &= E(K^*_0) \end{aligned}$$

[3] We are indebted to David Wagner for pointing out the possibility of finding equivalent keys in DEAL-128, and proposing another, earlier method for finding them.

$$= R_0^*$$
$$= R_0 \oplus \Delta$$
$$= R_2 \oplus \Delta$$

5. $R_3^* = R_3$ because

   We know that
   $$R_2^* = R_2 \oplus \Delta$$
   $$K_1^* = K_1 \oplus \Delta$$
   Therefore:
   $$R_3^* = E(R_2^* \oplus 2 \oplus K_1^*)$$
   $$= E(R_2 \oplus \Delta \oplus 2 \oplus K_1 \oplus \Delta)$$
   $$= E(R_2 \oplus 2 \oplus K_1)$$
   $$= R_3$$

6. We keep trying different values for (K_0, K_1) until we see $R_4^* = R_4 \oplus \Delta$.
7. $R_5^* = R_5$ because

   We know that:
   $$R_4^* = R_4 \oplus \Delta$$
   $$K_1^* = K_1 \oplus \Delta$$
   Therefore:
   $$R_5^* = E(R_4^* \oplus 8 \oplus K_1^*)$$
   $$= E(R_4 \oplus d \oplus 8 \oplus K_1 \oplus \Delta)$$
   $$= E(R_4 \oplus 8 \oplus K_1)$$
   $$= R_5$$

5 Related-Key Attacks on DEAL-256 and DEAL-192

Consider the algorithm for finding equivalent keys in DEAL-256. If we applied the algorithm without the special key property that $R_1 = R_5 \oplus 4$, we would end up with *nearly equivalent* keys: key with the same subkeys for all but the last two rounds. We could then mount an attack based on this fact, given encryptions from the two keys.

Here, we will discuss a related-key attack based on finding a pair of nearly-equivalent keys. We will discuss several issues with this attack, and then present the whole attack:

- How to detect that we have a pair of nearly-equivalent keys.
- How to use detection to learn information about the key.
- How to extract the last two rounds' subkeys when this property holds.
- How to mount the full attack.

5.1 Detecting Nearly-Equivalent Keys

Given three plaintext/ciphertext pairs from a pair of keys, (K, K^*) believed to be nearly-equivalent, we can determine whether they have this property with very high probability of being right, at the cost of about 2^{64} work and about 3×2^{56} memory locations. We mount something very similar to the meet-in-the-middle attack on double DES encryption.

Consider one text, broken into two 64-bit halves, (A_0, B_0). All but the last two rounds of encryption are identical between the keys, so after the identical rounds, we get (C_0, D_0) for this plaintext under both keys. The last two rounds are different, so we get (Y_0, Z_0) from K, and (Y_0*, Z_0*) from K^*.

Note that:

$$Z_0 = D_0 \oplus E_{R_7}(Y_0)$$
$$Z_0* = D_0 \oplus E_{R_7^*}(Y_0*).$$

We know three plaintext,ciphertext pairs, so we know three different sets of Y_0, Y_0*, and Z_0, Z_0* values. We can mount a DES keysearch effort on R_7 and R_7^*. We try all 2^{56} possible values of R_7, and for each one, we get candidate D_0 values from all three plaintexts. We do the same for all possible values of R_7^*. We get two tables of 2^{56} different 192-bit values, which must be sorted. We then find the matches between the two tables. For 192-bit keys, the keysearch would be on R_5 and R_5^*.

If we find a pair of matching values, it is overwhelmingly likely that we have found the right values for R_7, R_7^* (or R_5, R_5^*).

This shows how to determine whether a pair of keys is nearly-equivalent, but not how to find which pair in a batch of 2^{33} of them is nearly-equivalent.

Imagine a situation in which we had unlimited memory resources. We could do the same kind of meet-in-the-middle computation described above, but on all 2^{33} keys. This would take $2^{56} \times 2^{33} = 2^{89}$ encryptions, 89×2^{89} swap operations, and about 2^{95} bytes of memory. At

the end, we would sweep through the 2^{89} 192-bit blocks computed from three ciphertexts under each key, and look for duplicates. We would not expect to see any duplicates (though it wouldn't be totally surprising to see them) unless there is a pair of nearly-equivalent keys. Any duplicates that came either from the same key, or from keys with the same Δ value would simply be ignored.

In practice, we have limited memory resources, and so we consider time-memory tradeoffs.

The time-memory tradeoffs available here can be summarized as follows[4]:

Memory (bytes)	**Work** (DES encryptions) *Updated*
3×2^{69}	2^{113}
3×2^{61}	2^{121}
3×2^{53}	2^{129}
3×2^{45}	2^{137}
3×2^{37}	2^{145}

5.2 Extracting the Rest of the Key

Once we know R_7, R_7^*, we can mount the same kind of attack to get R_6, R_6^*. We have then peeled off the last two rounds, and have a six-round cipher remaining to attack. (In the case of DEAL-192, we have a four-round cipher remaining to attack.) In the case of DEAL-256, knowing R_6 and R_7 allows us to find K_3. In the case of DEAL-192, knowing R_4 and R_5 allows us to find K_2. This leaves us with a 192-bit search to break DEAL-256, or a 128-bit search to break DEAL-192.

5.3 Selecting the Keys

Let K be the original key. Let K_i be the ith additional key requested. We request Δ keys such that:

- Start with initial targeted key, K, and Δ active in parity bits only.
- For $i = 0$ to 255, do
  - Let Δ_j = next delta active in parity bits only.
  - For $j = 0$ to 2^{25}, do

$$K[0]_i = K[0]$$
$$K[1]_i = K[1] \oplus Random\_Block_j$$
$$K[2]_i = K[2] \oplus \Delta_i$$

$$R_2[i] = R_2[j] \oplus \Delta.$$

by the birthday paradox. So, we will have Δ pairs of keys to test, of which we expect one pair to be nearly-equivalent.

5.4 The Full Attack

The full attack is thus carried out as follows:

1. We request 2^{33} related keys according to the pattern described above. We expect one pair of these keys to be nearly-equivalent, but we don't yet know which pair.
2. We request the same three chosen plaintexts to be encrypted under each key. (We don't have to be able to choose anything about them, but the same three plaintexts must be encrypted under each key.)
3. We apply our test to the whole set of ciphertexts from the related keys. Given 3×2^{45} bytes of memory, we will have to carry out 2^{137} DES encyptions.
4. Let K, K^* be the pair of nearly-equivalent keys, which we have now detected. In detecting the property, we have learned the last round's subkey. We now apply the same meet-in-the-middle attack to find the next-to-last round's subkey. (In DEAL-256, this isR_6; in DEAL-192, this is R_4.)
5. We may now either apply some other attack on the cipher with two fewer rounds, or we may use knowledge of the last two rounds' subkeys to learn 64 bits of the input key, and then brute-force the remaining key.
6. Assuming we just brute-force the remaining key, the attacks on DEAL-192 and DEAL-256 both require 2^{33} related-key queries, the same three chosen-plaintexts requested under each key, and 3×2^{53} bytes of memory. The attack on DEAL-192 then requires 2^{129} work, and the attack on DEAL-256 requires 2^{192} work.

[4] These computational cost estimates assume memory available with no additional costs for random accesses. If the attack were implemented with tape memory, for example, then the actual time taken for the attack would go up substantially.

7. There may be improved attacks that exploit weaknesses in four- or six-round DEAL once we have discovered the last two round keys. For example, Biham's attack against Ladder-DES can also be applied to DEAL-192, once the last two rounds have been peeled off.

6 Conclusions

In this paper, we have demonstrated a weakness in the key schedule of DEAL, leading to both equivalent keys and vulnerability to related-key attacks. While the related-key attacks are of primarily academic interest (requiring 2^{128} DEAL encryptions worth of work for the cheapest attack), the equivalent keys are of immediate interest for anyone using DEAL in certain hashing modes. The important lessons we draw from this analysis are:

1. Simply using a cryptographic primitive in a reasonable-looking way to design a key schedule does not guarantee resistance to attacks on the key schedule.
2. In the specific case of DEAL, ignoring the parity bits of the keys sent in allowed nearly-equivalent keys to be found. A special class of keys were then found for which, instead of nearly-equivalent keys, these keys would be equivalent. Had those bits been immediately used, our attacks would not work.

Unfortunately, we don't have a general design principle we can pull out of this analysis; designing key schedules is hard, and there aren't any sure-fire shortcuts. This is borne out by the long list of AES candidates cryptanalyzed based on their key schedules which appears in the introduction.

6.1 Open Questions

A number of questions are raised by this research:

1. Are there key schedules we can build from cryptographic mechanisms that are provably secure against various forms of attack?
2. In the absence of these, can we at least find some useful design principles for cryptographic key schedules?
3. Are there similar attacks on other cryptographic key schedules, e.g., those of Khufu, Blowfish, and SEAL?

7 Acknowledgements

Thanks to David Wagner, Stefan Lucks, Ron Rivest, Adi Shamir, Richard Outerbridge, and Eli Biham for useful questions and comments during and after the FSE '99 rump session in Rome. Thanks to the anonymous referees for useful comments on the original submission of this paper.

References

[BBF+99] B. Biham, A. Biryukov, N. Ferguson, L. Knudsen, B. Schneier, and A. Shamir, "Cryptanalysis of Magenta," *Second AES Candidate Conference*, Mar 99.

[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.

[Bih94] E. Biham, "New Types of Cryptanalytic Attacks Using Related Keys," *Journal of Cryptology*, v. 7, n. 4, 1994, pp. 229–246.

[Bih97] E. Biham, "Cryptanalysis of Ladder-DES," *Fast Software Encryption, Fourth International Workshop*, Springer-Verlag, 1997.

[Bor99] J. Borst, "Weak Keys of Crypton," *Second AES Candidate Conference*, rump session presentation, Mar 99.

[CMK+98] L. Chen, J.L. Massey, G.H. Khachatrian, and M.K. Kuregian, "Nomination of SAFER+ as Candidate Algorithm for the Advanced Encryption Standard (AES)," NIST AES Proposal, Jun 98.

[Cop98a] D. Coppersmith, "DFC Weak Keys," Note to NIST AES Discussion Group, 10 Sep 98.

[Cop98b] D. Coppersmith, "Re: DFC Weak Keys," Note to NIST AES Discussion Group, 22 Oct 98.

[DBP+99] C. D'Halluin, G. Bijnens, B. Preneel, V. Rijmen, "Equivalent keys of HPC, draft paper, 1999. Available online at: `http://www.esat.kuleuven.ac.be/čosicart/ps/VR-9900.ps.gz`

[GGH+98] H. Gilbert, M. Girault, P. Hoogvorst, F. Noilhan, T. Pornin, G. Poupard, J. Stern, and S. Vaudenay, "Decorrelated Fast Cipher: an AES Candidate," NIST AES Proposal, Jun 98.

[GLC98] D. Georgoudis, D. Lerous, and B.S. Chaves, "The 'Frog' Encryption Algorithm," NIST AES Proposal, Jun 98.

[JH98] M.J. Jacobson and K. Huber, "The MAGENTA Block Cipher Algorithm," NIST AES Proposal, Jun 98.

[KSW96] J. Kelsey, B. Schneier, and D. Wagner, "Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES," *Advances in Cryptology — CRYPTO '96 Proceedings*, Springer-Verlag, 1996, pp. 237–251.

[KSW97] J. Kelsey, B. Schneier, and D. Wagner, "Related-Key Cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA," *Information and Communications Security, First International Conference Proceedings*, Springer-Verlag, 1997, pp. 203–207.

[KSW99] J. Kelsey, B. Schneier, and D. Wagner "Key Schedule Weaknesses in SAFER+," *Second AES Candidate Conference*, Mar 99.

[Knu93] L. Knudsen, "Cryptanalysis of LOKI '91," *Advances in Cryptography — AUSCRYPT '92 Proceedings*, Springer-Verlag, 1993.

[Knu98] L Knudsen, "DEAL — A 128-bit Block Cipher," NIST AES Proposal, Jun 98.

[Lim98] C.H. Lim, "CRYPTON: A New 128-bit Block Cipher," NIST AES Proposal, Jun 98.

[LR88] M. Luby, and C. Rackoff, "How to construct pseudorandom permutations from pseudorandom functions," SIAM Journal of Computing, 17: #2, 373-386, 1988.

[Luc99] S. Lucks, "On the Security of the 128-bit Block Cipher DEAL," *Fast Software Encryption, Sixth International Workshop*, Springer-Verlag, 1999, to appear.

[Mer91] R.C. Merkle, "Fast Software Encryption Functions," *Advances in Cryptology — CRYPTO '90 Proceedings*, Springer-Verlag, 1991, pp. 476–501.

[RC98] P. Rogaway and D. Coppersmith, "A Software-Optimized Encryption Algorithm," *Journal of Cryptology*, v. 11, n. 4, 1998, pp. 273–287.

[Rit94] T. Ritter, *Ladder-DES: A Proposed Candidate to Replace DES,* appeared in the Usenet newsgroup sci.crypt, Feb 1994.

[RRS+98] R. Rivest, M. Robshaw, R. Sidney, and Y.L. Yin, "The RC6 Block Cipher," NIST AES Proposal, Jun 98.

[Saa99] M. Saarinen, "A Note Regarding the Hash Function Use of MARS and RC6," available online from `http://www.jyu.fi/ mjos/`, 1999.

[Sch94] B. Schneier, "Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)," *Fast Software Encryption, Cambridge Security Workshop Proceedings*, Springer-Verlag, 1994, pp. 191–204.

[Sch98] R. Schroeppel "Hasty Pudding Cipher Specification," NIST AES Proposal, Jun 98.

[Wag99] D. Wagner, "Equivalent keys for HPC," *Second AES Candidate Conference*, rump session presentation, Mar 99.

[WFS99] D. Wagner, N. Ferguson, and B. Schneier, "Cryptanalysis of FROG," *Second AES Candidate Conference*, Mar 99.

[Win84] R.S. Winternitz, "Producing One-Way Hash Functions from DES," *Advances in Cryptology: Proceedings of Crypto 83*, Plenum Press, 1984, pp. 203–207.