Analysis of Camellia

Lars R. Knudsen

April 28, 2000

1 Analysis of Camellia

The best attack we have found on Camellia is an attack based on truncated differentials. The attack is similar to an attack presented recently on the block cipher E2 in [2]. The differential analysis can be found in Sections 1.1 and 1.2. Also, we analyse the key-schedule of Camellia in Section 1.3 and comment on the S-boxes in Section 1.4 and on some other attacks in Section 1.5. We shall use some definitions and terms from the designers and refer to [3] for further definitions. In the appendix we give a compressed overview of the state-of-the-art of block cipher cryptanalysis.

1.1 Differential and Linear Cryptanalysis

In this section we consider conventional differential and linear cryptanalysis. The maximum probability of a differential through one S-box is 2^{-6}. The maximum bias of a linear approximation through one S-box is 2^{-4}. The question is how we can use these in combination to obtain differentials and linear approximations for several or many rounds. One possible tool to make such estimates is to estimate or measure the total number of *active S-boxes* in an analysis. For a few rounds of Camellia it is easy to find the minimum number of active S-boxes. For differential attacks with equal inputs to the round function in one round the number of active S-boxes is zero, and similarly for linear approximations the minimum number of active S-boxes is zero. However, when moving to more rounds, the number of active S-boxes will increase. It follows from a simple inspection of the structure in the round function of Camellia that for two rounds of Camellia the minimum total number of active S-boxes will be one, for three rounds the minimum total number of active S-boxes will be six. This number will increase rapidly for more than 3 rounds. A conservative estimate for both differential and linear attacks would be to expect at least 3 active S-boxes in each round on the average. This would mean a probability of 2^{-18} per round for differential attacks and a bias of 2^{-10} per round for linear attack. Iterated to six rounds this gives a probability of 2^{-108} for differential attacks and a bias of 2^{-55} for linear attacks. All in all, in an attack this would mean a requirement of 2^{108} pairs of chosen plaintexts for the differential attack, and 2^{110} known plaintexts for the linear attack. These estimates are very conservative. First of all, in the estimates we have used the maximum probabilities and biases for every active S-box. Furthermore we have estimated a maximum number of

three active S-boxes for every round, which may be quite optimistic. For both attacks, there is a sufficient level of security for all versions of Camellia.

1.2 Truncated Differentials

In this section we report on several truncated differentials for Camellia. We use the following notation. Let $(x_1 \cdots x_s)$ denote a vector of s bytes. With $s = 16$ this will be a plaintext block or a ciphertext block after i rounds of encryption. With $s = 8$ this will typically be the input or the output of the function F. The *difference* between two s-byte vectors, $s \in \{8, 16\}$ will be the exclusive-or of the individual bytes in the vectors. Also, we let

$$x = (x_1 \cdots x_8) \xrightarrow{F} (y_1 \cdots y_8) = y,$$

denote that a difference of x in two input vectors to F can result in a difference of y in the two output vectors with probability p. A one-round differential will be denoted

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8		x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}
x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\xrightarrow{F}$	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8
z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8		z_9	z_{10}	z_{11}	z_{12}	z_{13}	z_{14}	z_{15}	z_{16}

Here $x_1, \ldots, x_{16}$ denote the difference in the two 128-bit texts before the round, and $z_1, \ldots, z_{16}$ denote the difference in the texts after the round. Also, $z_{j+8} = x_j$ for $j = 1, \ldots, 8$, and $z_i = x_{i+8} \oplus y_i$ for $i = 1, \ldots, 8$.

One of the best ways to push information about differences through several rounds in an iterated cipher is to use what is called *iterative* differentials. These are differentials which can be concatenated with themselves any number of times. For Camellia we have found the following 1-round differential of probability 2^{-16}

x_1	0	x_3	0	x_5	0	x_7	0		x_9	0	x_{11}	0	x_{13}	0	x_{15}	0
x_1	0	x_3	0	x_5	0	x_7	0	$\xrightarrow{F}$	y_1	0	y_3	0	y_3	0	y_1	0
z_1	0	z_3	0	z_5	0	z_7	0		z_9	0	z_{11}	0	z_{13}	0	z_{15}	0

where it is assumed that $x_i \neq 0$ for $i \in \{1, 3, 5, 7\}$. The differential is iterative, which means that it can be concatenated with itself r times, yielding an r-round differential of probability 2^{-16r}. The probability is calculated as follows. A difference $x_5 \neq 0$ in two input bytes to the S-box 4 results in some difference y_3, and a difference $x_7 \neq 0$ in two input bytes to the S-box 2 results in some difference y_1. Then with an average probability of 2^{-8} a difference $x_1 \neq 0$ in two input bytes to the S-box 1 results in the difference $y_1 \oplus y_3$. Similarly, with an average probability of 2^{-8} a difference $x_3 \neq 0$ in two input bytes to the S-box 3 results in the difference $y_1 \oplus y_3$. Both these events will happen with probability 2^{-16}. The mixing of bytes at the end of the round function results in the difference indicated above.

In [2] similar differentials are exploited in cryptanalytic attacks on the block cipher E2. As noted in [2] the above individual probabilities are in fact $\frac{1}{255}$

rather than 2^{-8}, however since we are going to iterate this differential there will be a dependency between the different rounds and the exact probability will be hard to calculate. However, it seems plausible in the analysis of Camellia to assume independence between the rounds, just as the authors assumed in the analysis of E2 [2]. Therefore, for convenience, we shall use 2^{-8} as an approximation of the individual probabilities.

This differential iterated to five rounds looks as follows.

x_1	0	x_3	0	x_5	0	x_7	0		x_9	0	x_{11}	0	x_{13}	0	x_{15}	0
a_1	0	a_3	0	a_5	0	a_7	0	$\xrightarrow{F}$	A_1	0	A_3	0	A_3	0	A_1	0
b_1	0	b_3	0	b_5	0	b_7	0	$\xrightarrow{F}$	B_1	0	B_3	0	B_3	0	B_1	0
c_1	0	c_3	0	c_5	0	c_7	0	$\xrightarrow{F}$	C_1	0	C_3	0	C_3	0	C_1	0
d_1	0	d_3	0	d_5	0	d_7	0	$\xrightarrow{F}$	D_1	0	D_3	0	D_3	0	D_1	0
e_1	0	e_3	0	e_5	0	e_7	0	$\xrightarrow{F}$	E_1	0	E_3	0	E_3	0	E_1	0
y_1	0	y_3	0	y_5	0	y_7	0		y_9	0	y_{11}	0	y_{13}	0	y_{15}	0

Note that $a_i = x_i$ for $i \in \{1, 3, 5, 7\}$. Here it is assumed that $a_i \neq 0$, $b_i \neq 0$, $c_i \neq 0$, $d_i \neq 0$, $e_i \neq 0$, for $i \in \{1, 3, 5, 7\}$. In a chosen plaintext attack one can choose the plaintexts such that $a_i = x_i \neq 0$ for $i \in \{1, 3, 5, 7\}$. The probability that the difference in the four input words is nonzero in one round is $(\frac{255}{256})^4 \approx 0.984$. The total probability is therefore $(0.984)^4 2^{-80} \approx 2^{-80}$. In the following we shall ignore the factors 0.984.

Note that the exclusive-or of the difference in the plaintext vectors and the difference in the ciphertext vectors (after five rounds) has the following form.

$$z_1\ 0\ z_3\ 0\ z_3\ 0\ z_1\ 0\ \ z_9\ 0\ z_{11}\ 0\ z_{11}\ 0\ z_9\ 0 \tag{1}$$

This follows from the observation that the exclusive-or of the right half of the plaintext and the right half of the ciphertext equals the exclusive-or of the outputs of the F-function in the fourth and second rounds. And similarly, the exclusive-or of the left half of the plaintext and the left half of the ciphertext equals the exclusive-or of the outputs of the F-function in the fifth, third and first rounds. For a randomly chosen permutation the exclusive-or of the pairs of plaintexts and pairs of ciphertexts will have the form of (1) with a probability of 2^{-96}. For Camellia this happens with probability of at least 2^{-80}, since if the texts follow the differential we will have the form (1). Recently, we presented a new approach of how to use differentials to distinguish block ciphers from randomly chosen permutations [1]. In the following we shall apply this method to Camellia.

1.2.1 Distinguishing attacks

The above differential can be enhanced by letting the first round be a trivial round with probability one.

0	0	0	0	0	0	0	0		x_9	0	x_{11}	0	x_{13}	0	x_{15}	0
0	0	0	0	0	0	0	0	$\xrightarrow{F}$	0	0	0	0	0	0	0	0
b_1	0	b_3	0	b_5	0	b_7	0	$\xrightarrow{F}$	B_1	0	B_3	0	B_3	0	B_1	0
c_1	0	c_3	0	c_5	0	c_7	0	$\xrightarrow{F}$	C_1	0	C_3	0	C_3	0	C_1	0
d_1	0	d_3	0	d_5	0	d_7	0	$\xrightarrow{F}$	D_1	0	D_3	0	D_3	0	D_1	0
e_1	0	e_3	0	e_5	0	e_7	0	$\xrightarrow{F}$	E_1	0	E_3	0	E_3	0	E_1	0
y_1	0	y_3	0	y_5	0	y_7	0		y_9	0	y_{11}	0	y_{13}	0	y_{15}	0

The probability of this differential is 2^{-64}. There are only 4 nonzero bytes in the plaintext difference. From one *structure* of 2^{32} plaintexts one can form 2^{63} pairs of plaintexts with the desired difference. Thus for Camellia reduced to 5 rounds, with 4 structures one would get 2^{65} pairs of plaintexts, and consequently one would get approximately two ciphertext pairs for which (1) is satisfied. For a randomly chosen permutation one would get approximately 2^{-31} ciphertext pairs with the desired difference. Thus with a high probability one can distinguish Camellia with five rounds from a randomly chosen permutation.

Fact 1 *The first five rounds of Camellia can be distinguished from a random permutation using about* 2^{34} *chosen plaintexts.*

We can extend this attack to an attack on six rounds of Camellia. The differential can be extended to six rounds by adding one round of probability 2^{-16}.

0	0	0	0	0	0	0	0		x_9	0	x_{11}	0	x_{13}	0	x_{15}	0
0	0	0	0	0	0	0	0	$\xrightarrow{F}$	0	0	0	0	0	0	0	0
a_1	0	a_3	0	a_5	0	a_7	0	$\xrightarrow{F}$	A_1	0	A_3	0	A_3	0	A_1	0
b_1	0	b_3	0	b_5	0	b_7	0	$\xrightarrow{F}$	B_1	0	B_3	0	B_3	0	B_1	0
c_1	0	c_3	0	c_5	0	c_7	0	$\xrightarrow{F}$	C_1	0	C_3	0	C_3	0	C_1	0
d_1	0	d_3	0	d_5	0	d_7	0	$\xrightarrow{F}$	D_1	0	D_3	0	D_3	0	D_1	0
e_1	0	e_3	0	e_5	0	e_7	0	$\xrightarrow{F}$	E_1	0	E_3	0	E_3	0	E_1	0
y_1	0	y_3	0	y_5	0	y_7	0		y_9	0	y_{11}	0	y_{13}	0	y_{15}	0

This differential has a probability of approximately 2^{-80}. For a pair of plaintexts following the differential one will have the form of (1) for the exclusive-or of the ciphertext and plaintext differences. To use this differential to distinguish Camellia from a randomly chosen permutation, one has to generate about 2^{81} pairs of plaintexts with the desired difference. So to generate 2^{81} pairs one would need about 2^{18} structures, totally about 2^{50} chosen plaintexts. The distinguishing technique is the same as above for five rounds.

Fact 2 *The first six rounds of Camellia can be distinguished from a random permutation using about* 2^{50} *chosen plaintexts.*

If we ignore the functions FL and FL^{-1} we can extend the attack to seven rounds of Camellia. First we extend the above six-round differential by adding another 1-round differential of probability 2^{-16}.

0	0	0	0	0	0	0	0		x_9	0	x_{11}	0	x_{13}	0	x_{15}	0
0	0	0	0	0	0	0	0	$\xrightarrow{F}$	0	0	0	0	0	0	0	0
a_1	0	a_3	0	a_5	0	a_7	0	$\xrightarrow{F}$	A_1	0	A_3	0	A_3	0	A_1	0
b_1	0	b_3	0	b_5	0	b_7	0	$\xrightarrow{F}$	B_1	0	B_3	0	B_3	0	B_1	0
c_1	0	c_3	0	c_5	0	c_7	0	$\xrightarrow{F}$	C_1	0	C_3	0	C_3	0	C_1	0
d_1	0	d_3	0	d_5	0	d_7	0	$\xrightarrow{F}$	D_1	0	D_3	0	D_3	0	D_1	0
e_1	0	e_3	0	e_5	0	e_7	0	$\xrightarrow{F}$	E_1	0	E_3	0	E_3	0	E_1	0
f_1	0	f_3	0	f_5	0	f_7	0	$\xrightarrow{F}$	F_1	0	F_3	0	F_3	0	F_1	0
y_1	0	y_3	0	y_5	0	y_7	0		y_9	0	y_{11}	0	y_{13}	0	y_{15}	0

This differential also has a probability of approximately 2^{-96}. As before, we will generate many plaintext pairs and count the number of pairs for which (1) hold. However, note that the probability of the differential is the same as the probability that for a randomly chosen permutation (1) will hold. For Camellia with seven rounds, ignoring the functions FL and FL^{-1}, (1) will hold for all pairs which follow the differential. Clearly for all pairs the first round of the differential will hold. A pair of plaintexts will not follow the second round with probability $1 - 2^{-16}$. If we assume, which seems plausible, that the ciphertext differences in these cases look random, then (1) will hold also in these cases with a probability of 2^{-96}. All together, the probability for this seven-round version of Camellia that the plaintext and ciphertext difference will satisfy (1) is approximately $2^{-96} + (1 - 2^{-16})2^{-96} \simeq 2^{-95}$. (Actually, this probability also covers the cases where the pair of texts follow the differential in the first round, but not the second, and the cases where the pair of texts follow the differential in the first two rounds, but not in the third, and so on. However, the sum of the probabilities of all these cases are small compared to the above probabilities.) To distinguish this version of Camellia from a randomly chosen permutation we pick 2^{36} structures (as above) which gives us 2^{99} pairs of plaintexts with the desired differences. For the Camellia-version this yields about 16 pairs of texts for which (1) holds, while the expected number for a randomly chosen permutation is 8. Since the standard deviation in the first case is 4 and around 3 in the last case, with a high probability we will be able to distinguish between the two. Totally this attack needs 2^{68} chosen plaintexts and we have the following result.

Fact 3 *The first seven rounds of Camellia without the functions FL and FL^{-1} can be distinguished from a random permutation using about 2^{68} chosen plaintexts.*

We could try and extend this one round further by adding another round of probability 2^{-16} to the above differential. The probability will be 2^{-112}. However, in this case the difference in probabilities of obtaining texts on the form of (1) will be very small. For the Camellia-version the total probability of getting

such pairs will be approximately $2^{-112} + (1 - 2^{-16})2^{-96} = 2^{-96}$ which is what one can expect also for a randomly chosen permutation.

It does not seem possible to extend the approach of distinguishing Camellia from a randomly chosen permutation further than to versions reduced to 7 rounds. However, there exist differentials for Camellia for even more number of rounds.

1.2.2 The existence of differentials

Consider the above differentials for five, six and seven rounds. There are $2^{16} - 2^8$ pairs of bytes with a nonzero exclusive-or difference, and 2^8 pairs of bytes with zero exclusive-or. The differences x_9, x_{11}, x_{13}, and x_{15} can take any value, thus totally 2^{32} values and consequently there are approximately 2^{63} possible pairs for all four bytes together. Thus, there are approximately $2^{63} \times (2^8)^{12} = 2^{159}$ pairs of plaintexts with the desired difference. If we ignore the keyed functions FL and FL^{-1} in Camellia we can iterate the above differential to any number of rounds, with a decrease in probability of a factor of 2^{-16} for every additional round. Iterated to eleven rounds the probability is approximately $(2^{-16})^{10} = 2^{-160}$. Therefore, for this version of Camellia for about one in every two keys, one can expect to find one pair of plaintexts, which will follow the expected (zero) values specified in the differential in every round.

However, here we assumed that the functions FL and FL^{-1} were not used. Let us analyse these functions with respect to the differences used above. If two 64-bit texts have a difference of $(a\ 0\ b\ 0\ b\ 0\ a\ 0)$ in the input to FL (or FL^{-1}) then the differences in the outputs will be of the form $(f\ 0\ e\ 0\ c\ 0\ d\ 0)$ with some high probability. However, the values of f and e will not be equal nor will the values of c and d. To see why this is the case, let us take a closer look at FL. The first operation is to bitwise 'AND' the left half of the input with a subkey, rotate the result by one position to the left, and exclusive-or this result to the right half of the input. Thus, if the left halves of two inputs has a difference $(a\ 0\ b\ 0)$, the difference after the 'AND' of a key will be $(a'\ 0\ b'\ 0)$, where a' and b' have Hamming weights at most those of a and b respectively. If the most significant bits of the leftmost bytes of the inputs (the bytes with difference a) are equal, they will be equal after the 'AND' operation; if the bits are different they will be equal after the 'AND' operation with a probability of 1/2, where the probability is taken over all keys. Similar observations for the most significant bits of the third leftmost bytes (the bytes with difference b). If the involved bits are equal, the one-bit rotation leaves the difference in the second and fourth bytes zero. Note that in that case the right halves of the output of FL will have the form $(c\ 0\ d\ 0)$. The 'OR' operation in the second "round" of FL means that the left halves of the outputs of FL will have the form $(f\ 0\ e\ 0)$. In total the differential

$$(a\ 0\ b\ 0\ b\ 0\ a\ 0) \stackrel{FL}{\rightarrow} (f\ 0\ e\ 0\ c\ 0\ d\ 0)$$

has probability at least 1/4 for all keys, probability at least 1/2 for three of four keys, and probability 1 for one in four keys, where the probability is taken over all input texts. A similar observation can be made about the function FL^{-1}.

In an attempt to specify differentials for more rounds of Camellia, one could allow both the left and the right halves of the plaintexts to take any values, and modify the differential accordingly. The differential would have the following form.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8		x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}
a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	$\xrightarrow{F}$	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
b_1	0	b_3	0	b_5	0	b_7	0	$\xrightarrow{F}$	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
c_1	0	c_3	0	c_5	0	c_7	0	$\xrightarrow{F}$	C_1	0	C_3	0	C_3	0	C_1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
m_1	0	m_3	0	m_5	0	m_7	0	$\xrightarrow{F}$	M_1	0	M_3	0	M_3	0	M_1	0
y_1	0	y_3	0	y_5	0	y_7	0		y_9	0	y_{11}	0	y_{13}	0	y_{15}	0

Here it is assumed that $x_{i+8} = A_i$ for $i \in \{2, 4, 6, 8\}$ and that $a_i = B_i$ for $i \in \{2, 4, 6, 8\}$. The probability for r rounds, $r > 2$ is $2^{-16r-32}$ with an addition decrease in probability of a factor of 2^{-4} for every layer of FL and FL^{-1}. Such a differential would have probabilities 2^{-32} in the first two rounds, and as before, a probability of 2^{-16} in subsequent rounds. This would give approximately 2^{255} pairs of plaintexts available in the analysis. However, for such pairs of plaintexts and ciphertexts, pairs satisfying the differential can no longer be assumed to have the form of (1) for neither halves. However, every second byte of the ciphertext would still be expected to be zero.

This means that for Camellia (including FL and FL^{-1}) reduced to 13 rounds, the above differential has a probability of $2^{-16 \cdot 13-32-4-4} = 2^{-248}$. Thus, there will exist pairs of plaintexts following the expected values in the differential for up to 13 rounds. However, it is very difficult to see the applications of such (good) pairs in cryptanalytic attacks.

Summing up, there exist truncated differentials for every fixed key for up to 13 rounds of Camellia, specifying 64 bits of information after every round. We showed that these differentials can be used to distinguish Camellia when reduced to 6 rounds from a randomly chosen permutation. We expect that this 6-round distinguisher can be extended to a key-recovery attack on Camellia reduced to 7 rounds. Also, we showed that for Camellia reduced to 7 rounds but without the functions FL and FL^{-1} there is a distinguishing attack requiring 2^{68} chosen plaintexts.

1.3 The Key-schedule

The key-schedule takes a 128-bit, a 192-bit or a 256-bit key, K, as input. In the first phase the key-schedule defines two keys K_L and K_R each of 128 bits, and then computes two other keys of 128 bits each, K_A and K_B, as a function of the user-selected key. The key K_B is used only for the 192-bit and 256-bit key versions.

For the 128-bit key version $K_L = K$ and $K_R = 0$. For the 192-bit key version K_L is the leftmost 128 bits of K, and the remaining 64 bits are assigned the left half of K_R. The right half of K_R is the bitwise negated value of its left half. For the 256-bit key version, K_L is the leftmost 128 bits of K, and K_R the

rightmost 128 bits of K. K_A is computed as follows. Let $C^i = C^i_L \mid C^i_R$ and let $C^0 = K_L \oplus K_R$. Then we compute

$$\begin{aligned}
C^1_L &= F(C^0_L, \sigma_1) \oplus C^0_R && (2)\\
C^1_R &= C^0_L && (3)\\
C^2_L &= F(C^1_L, \sigma_2) \oplus C^1_R && (4)\\
C^2_R &= C^1_L && (5)\\
C^2 &= C^2 \oplus K_L && (6)\\
C^3_L &= F(C^2_L, \sigma_3) \oplus C^2_R && (7)\\
C^3_R &= C^2_L && (8)\\
C^4_L &= F(C^3_L, \sigma_4) \oplus C^3_R && (9)\\
C^4_R &= C^3_L, && (10)
\end{aligned}$$

and define $K_A = C^4$. Set $C^4 = C^4 \oplus K_R$, compute

$$\begin{aligned}
C^5_L &= F(C^4_L, \sigma_5) \oplus C^4_R && (11)\\
C^5_R &= C^4_L && (12)\\
C^6_L &= F(C^5_L, \sigma_6) \oplus C^5_R && (13)\\
C^6_R &= C^5_L, && (14)
\end{aligned}$$

and define $K_B = C^6$. The values σ_i are fixed constants. For the 128-bit key version the 26 round keys of each 64 bits are computed from the keys K_L and K_A. For the 192-bit key and 256-bit versions the 34 round keys of each 64 bits are computed from the keys K_L, K_R, K_A and K_B.

First, we are convinced that the above key-schedule makes related-key attacks very difficult. In these attacks an attacker must be able to get encryptions under several related keys. If the relation between, say, two keys is known then if the corresponding relations between the round-keys can be predetermined, sometimes one can predict how the keys encrypt a pair of different plaintexts. However, since the round-keys depend on K_A and K_B, which are results of encryptions, these round-key relations will be very hard to control and predict. Also, the slide-attacks seems to be very unlikely to succeed for Camellia.

Our second observation is that the above description of the key-schedules can be simplified somewhat. Consider the version of Camellia with 128-bit keys. Let $K_L = K_{LL} \mid K_{LR}$ and set $C^0 = 0$. Then we compute

$$\begin{aligned}
C^1_L &= F(C^0_L, K_{LL} \oplus \sigma_1) \oplus C^0_R && (15)\\
C^1_R &= C^0_L && (16)\\
C^2_L &= F(C^1_L, K_{LR} \oplus \sigma_2) \oplus C^1_R && (17)\\
C^2_R &= C^1_L && (18)\\
C^3_L &= F(C^2_L, \sigma_3) \oplus C^2_R && (19)\\
C^3_R &= C^2_L && (20)\\
C^4_L &= F(C^3_L, \sigma_4) \oplus C^3_R && (21)\\
C^4_R &= C^3_L. && (22)
\end{aligned}$$

Then again $K_A = C^4$. Note that this definition saves one exclusive-or operation and further the structure is now of a classical Feistel-type. All in all, the key K_A is the ciphertext of the plaintext zero in a four-round Feistel encryption scheme using the Camellia round function. The round keys in this Feistel scheme are dependent on the halves of the key K_L in the first two rounds, but they are fixed in the final two rounds. This has the effect that if an attacker is able, by some means, to find the value of K_A, then he can compute the key K_L in a straightforward manner. Also, if an attacker is able to find K_{AR} and K_{LL} (totally 128 bits), then he can compute all of K_A and K_L. In both cases, the attacker can compute the values of all round-keys. Since K_A is the ciphertext of a 4-round Feistel cipher depending on K_L, the above properties are somewhat surprising.

Consider next the version of Camellia with 192-bit and 256-bit keys. Let $K_L = K_{LL} \mid K_{LR}$, $K_R = K_{RL} \mid K_{RR}$, and set $C^0 = 0$. Then we compute

$$
\begin{aligned}
C_L^1 &= F(C_L^0, K_{LL} \oplus K_{RL} \oplus \sigma_1) \oplus C_R^0 && (23)\\
C_R^1 &= C_L^0 && (24)\\
C_L^2 &= F(C_L^1, K_{LR} \oplus K_{RR} \oplus \sigma_2) \oplus C_R^1 && (25)\\
C_R^2 &= C_L^1 && (26)\\
C_L^3 &= F(C_L^2, \sigma_3 \oplus K_{RL}) \oplus C_R^2 && (27)\\
C_R^3 &= C_L^2 && (28)\\
C_L^4 &= F(C_L^3, \sigma_4 \oplus K_{RR}) \oplus C_R^3 && (29)\\
C_R^4 &= C_L^3 && (30)\\
C_L^5 &= F(C_L^4, \sigma_5) \oplus C_R^4 && (31)\\
C_R^5 &= C_L^4 && (32)\\
C_L^6 &= F(C_L^5, \sigma_6) \oplus C_R^5 && (33)\\
C_R^6 &= C_L^5 && (34)
\end{aligned}
$$

Then $K_A = C^4 \oplus K_R$ and $K_B = C^6$. Here one can compute $K_A \oplus K_R$ from K_B and similarly from K_B one can compute $K_A \oplus K_R$.

We do not think that the above considerations imply a speed-up of an exhaustive key search, only that knowledge of some round-keys gives immediate knowledge of other round-keys. It seems that such properties could be avoided in a re-design of the key-schedule.

1.4 The S-boxes

Camellia uses four S-boxes, s_1, s_2, s_3, and, s_4. s_1 is derived from an inversion function in $GF(2^8)$ together with an affine transformation in the outputs and in the inputs. The other S-boxes are derived from s_1 by a simple rotation by one position of either the input or the output. In more detail, $s_2(x) = s_1(x) <<< 1$, $s_3(x) = s_1(x) >>> 1$, and $s_4(x) = s_1(x <<< 1)$. Clearly, the four S-boxes are very related. The advantages of deriving all S-boxes from one S-box are clear for implementation reasons. The disadvantages are not as clear, however, if an attacker would find a weakness in one of the S-boxes, then there is a high

probability that this weakness would appear in all S-boxes. We have not found any reason to suspect that cryptographic weaknesses are present nor will be detected in any of the S-boxes. Also, ciphers like the AES-candidate Rijndael make a more simple use of only one S-box. The S-box in Rijndael is derived in a manner similar to that of Camellia and is used throughout the cipher. So far, no reports have been published that this is a weakness for Rijndael. Thus there is no immediate indication that this poses any threat for the security of Camellia.

1.5 Other attacks

The S-boxes used in Camellia have the highest possible nonlinear order of 8-bit S-boxes, namely seven. Therefore it can be expected that higher order differential attacks will have only limited applications.

Since the round function of Camellia is bijective, the non-surjective attack will not be applicable.

The interpolation attacks work particularly well for ciphers for which the nonlinear components have a simple mathematical description. For Camellia, the S-boxes used have a complex description. Furthermore the use of the functions FL and FL^{-1} are likely to destroy any mathematical structure from the S-boxes. In our opinion it is very unlikely that the interpolation attack will be of any threat to Camellia.

The mod-n cryptanalysis works particularly well for ciphers using a mix of rotations and modular additions. Since Camellia uses the exclusive-or operation and no modular additions, the mod-n analysis will not be applicable to Camellia.

Finally, there are the trivial brute-force attacks. First, an exhaustive key search will be applicable to Camellia, but because of the size of the keys, such searches are unlikely to be feasible for at least the next 30 years. The matching ciphertext attack works for all block ciphers and depends only on the block size n. With $2^{n/2}$ ciphertext blocks one can expect to see at least two equal ciphertext blocks. This gives information about some plaintext blocks in three of the four standard modes of operation. For Camellia, this means that a maximum of 2^{64} blocks should be encrypted using one key.

2 Concluding Remarks

We have analysed Camellia and found no important weaknesses. The cipher has a conservative design and any practical attacks against Camellia would require a major breakthrough in the area of cryptanalysis of encryption systems. We think that Camellia is a very strong cipher, which matches the security of the best block ciphers today.

References

[1] L.R. Knudsen. A detailed analysis of SAFER. *The Journal of Cryptology*, 13(4):417–436, 2000.

[2] M. Matsui and T.Tokita. Cryptanalysis of a reduced version of the block cipher E2. In L. R. Knudsen, editor, *Fast Software Encryption, Sixth International Workshop, Rome, Italy, March 1999, LNCS 1636*, pages 71–79. Springer Verlag, 1999.

[3] NTT, Mitsubishi. Camellia - 128-bit block cipher.