A TIMING ATTACK ON THE CIKS-1 BLOCK CIPHER

Michael Furlong
e-mail: michaelf@engr.mun.ca

Howard Heys
e-mail: howard@engr.mun.ca

Abstract

The use of data-dependent transformations has been an area of increasing interest for the designers of ciphers. In particular, data-dependent permutations (DDPs) provide a fast and simple cryptologic primitive when implemented in hardware. However, when a DDP block is naively implemented in software, it can reveal information about the Hamming weight of the control vector applied to it. Specifically, when a subkey is used as a control vector then information about the Hamming weight of the subkey can be directly obtained from timing information. This potentially leaves ciphers heavily dependent on DDPs vulnerable to timing attacks.

In this paper, we examine the application of a timing attack to the CIKS-1 symmetric block cipher. The analysis is motivated by the possibility that a naive implementation of the DDPs used in CIKS-1 would result in encryption taking a time that is a function of data. Such implementations are possible in software environments, typically in embedded systems such as smart cards. The methodology of deriving the Hamming weight of the key using a known plaintext attack based on timing information is outlined and is followed by a discussion of the results. Further, a simple means of thwarting the timing attack in a software implementation is presented.

Keywords

1 Introduction

The CIKS-1 encryption algorithm was introduced in [1] as a symmetric key block cipher which would have an efficient implementation in hardware and be able to thwart both linear and differential cryptanalysis. However, for some implementations, information regarding the nature of the key is revealed by the time it takes to encrypt a message.

The idea of a timing attack was first presented by Kocher in [2]. The underlying principle of the attack is that information about the time it takes to encrypt a message can reveal information about the key to the cipher. Kocher examined the applicability of the timing attack to asymmetric crypto-systems, but noted that symmetric systems may also be vulnerable to such attacks. For example, the RC5 block cipher is made vulnerable when implemented on a system that does not perform data-dependent rotations in constant time [3].

CIKS-1 has been subjected to linear [4] and differential [5] cryptanalysis. Both attempts have exploited weaknesses in the data-dependent permutations showing weaknesses in the cipher much greater than was anticipated in [1].

This paper will look at the weakness exposed in CIKS-1 due to the timing information resulting from the data-dependent permutation blocks. These DDP blocks compose the majority of the cryptologic primitives used in the CIKS-1 cipher. An attack is then presented which uses this timing information to determine the Hamming weight of the expanded key.

2 Description of the Cipher

The CIKS-1 cipher has a 64-bit block and consists of eight rounds involving the application of eight 32-bit subkeys. The operation of one round of the cipher is presented in Figure 1. We let L_0 and R_0 represent the left and right half of the plaintext input, respectively, each half consisting of 32 bits and we use the notation L_{i-1} and R_{i-1} to represent the input to the i-th round. The 32-bit subkeys, K_i, $1 \leq i \leq 8$, for each round are extended to 48 bits and are applied as control vectors to two permutations. In the figure, $X >>> p$ represents a circular right shift by p bits acting on vector X, Π_1 and Π_2 are fixed permutations, $+$ represents mod 2^2 addition of the bits in two vectors, and the boxes labelled as n/m we shall denote $DDP_{n/m}$ representing a data-dependent permutation acting on n bits and requiring a control vector of m bits in length.

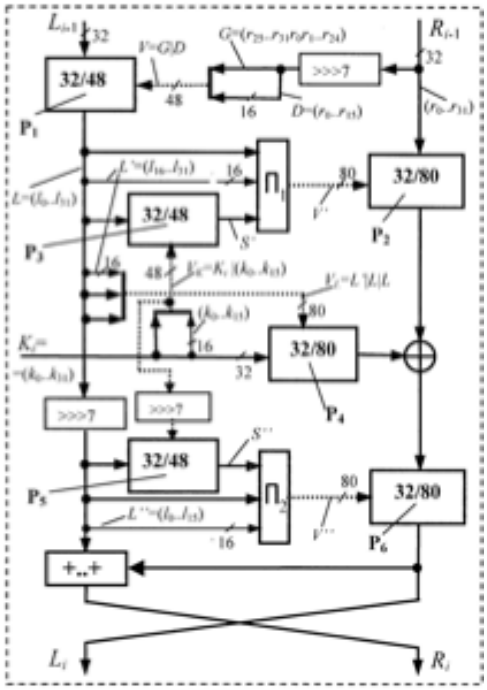

Figure 1: One round of the CIKS-1 cipher [1]

0-7803-8886-0/05/$20.00 ©2005 IEEE
CCECE/CCGEI, Saskatoon, May 2005

3 Cipher Components

The CIKS-1 cipher uses four distinct types of cryptologic primitives. Three of these, the fixed permutations, XOR, and mod 2^2 addition all occur in constant time and thus can be subtracted out of any timing information acquired. The data-dependent permutation operations do provide variable timing information in a naive implementation.

The primitives used in the cipher are described in the following sections.

3.1 Data-Dependent Permutations

The data-dependent permutation is the vulnerable component of the cipher. There is a direct relationship between the number of swaps that occur in a DDP block and the Hamming weight of the control vector. The DDP component does not change the Hamming weight of the data upon which it acts. This has been exploited by other attacks on CIKS-1 as seen in [6]. The DDP components are labelled P_1 to P_6 in Figure 1.

The DDP blocks can be of varying size and each DDP block is composed of smaller DDP blocks. The smallest DDP block has two inputs to be swapped, x_0 and x_1, and one control bit, v, and is called $DDP_{2/1}$. The general DDP block, $DDP_{2/1}$, and composition of other DDPs from the $DDP_{2/1}$ are illustrated in Figure 2.

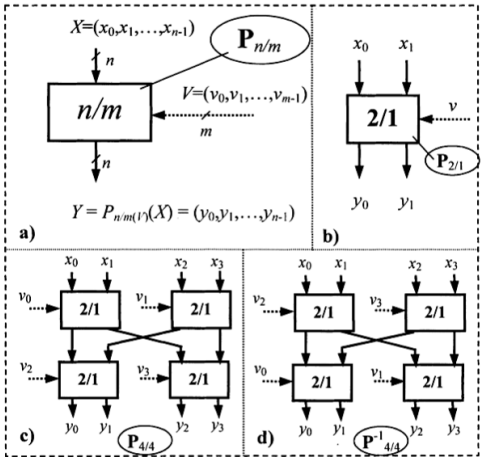

Figure 2: DDP blocks [1]

In the $DDP_{2/1}$, a control bit with a value of zero causes the swap of two bits, which takes time t_0, whereas a control bit of value one causes no swap and takes a time of t_1, where, as we shall see in Section 4, for a naive implementation $t_1 < t_0$. In this case, it can be seen that the number of swaps in any DDP is directly related to the Hamming weight of the control vector.

3.2 Fixed Permutations

Fixed permutations occur as Π_1 and Π_2, as well as three seven-bit circular shifts. Like the DDPs, these elements do not change the Hamming weight of the data upon which they act. This fact becomes useful when we remove extraneous timing information from the first and last rounds of the cipher. The fixed permutations themselves can be assumed to operate in constant time, irrespective of the data.

3.3 XOR

The XOR component is the bit-wise XOR of 2 vectors and is used to mix the key information in with the right side of the data. It can be assumed that this operation takes constant time to complete.

3.4 Addition

CIKS-1 uses mod 2^2 addition to combine the left and right data at the end of each round. In hardware, sixteen 2-bit addition components are used in parallel with the carry output being ignored. In a software implementation it could be implemented as a sequence of two bit additions. But whether it is done in parallel or sequentially, the addition can be assumed to take constant time.

4 Basic Principles of the Attack

The timing attack on CIKS-1 is applicable to implementations where the data-dependent permutations are executed in non-constant time and it is possible to measure accurately the timing associated with each encryption. For example, this may be possible for an implementation of the cipher in a single-threaded software environment, such as a microcontroller like the ones found in smart cards.

The subkeys for each round are used as the control vectors for the $DDP_{32/48}$ block. This requires the duplication of 16 bits of the key. Thus each subkey represents a 48-bit value, for a total key length of 384 bits over all 8 rounds. The 384-bit key is referred to as the expanded key, and it is the Hamming weight of this vector that is determined by the attack.

The weakness in CIKS-1 is exposed in an implementation of the $DDP_{2/1}$ where a swap is implemented as follows:

$$\text{if } (v == 0) \ \{\text{swap(x0, x1)}\}$$

where v is the control bit, and x0 and x1 are input bits to be considered for swapping. If the control bit indicates there should be a swap then the time to complete these instructions will include the time to evaluate the expression and the time it takes to execute the swap. Should the control bit indicate that there should not be a swap, then the time to execute the statements would only include the time to evaluate the boolean expression.

The underlying principle of the attack is that when the same key is exposed to sufficient data, the key-dependent data-dependent permutations will reveal information which is directly related to the Hamming weight of the expanded key.

5 Proposed Attack

The weakness in the CIKS-1 cipher lies with the data-dependent permutations. By obtaining timing information

about the execution of a DDP, the number of swaps and, hence, the Hamming weight of its control vector can be determined. So if we know the timing information for a plaintext/ciphertext pair, we can directly relate it to the number of swaps that took place in the DDP blocks of the encryption process and, subsequently, we can use this to determine the Hamming weight of the expanded key.

Because we know the plaintexts and the resultant ciphertexts, we can calculate how many swaps occurred in DDPs P_1, P_2, P_4, and P_6 in the first round, and P_2, P_4, and P_6 in the last round. We cannot determine the swaps that occur in DDPs P_3 and P_5 in any round because they depend directly on the unknown key.

Although we do not know the number of swaps for DDPs P_1, P_2, P_4, and P_6 in rounds two to seven, we do have information on the statistical properties of the swaps. For example, for $DDP_{32/80}$, assuming the 80-bit control vector is generated by replicating 16 random bits three times and 16 random bits twice, the expected number of swaps is $\mu = 40$ and the variance is given by $\sigma^2 = 52$.

For each plaintext/ciphertext/timing tuple, we maintain a quantity Γ_i that represents the square of the differences between the recorded number of swaps that occurred during encryption and the estimated number of swaps that would occur on average as a result of our guess at the Hamming weight of the expanded key. (From the recorded number of swaps that occurred we can account for the number of swaps that we know in the first and last rounds, to remove some uncertainty.) We can view the recorded number of swaps as being composed of two quantities: w, the number of swaps due to data being used as the control vector, and s_{HW}, the number of swaps due to the key being used as the control vector. Similarly, we can view the estimated number of swaps being composed of two quantities: μ_w, the expected value of the total number of swaps due to data being used as the control vector, and $\tilde{s}_{HW}$ the number of swaps that would occur due to our guess at the Hamming weight of the key. Hence, we may write Γ_i as:

$$\Gamma_i = [(w + s_{HW}) - (\mu_w + \tilde{s}_{HW})]^2$$

For each of N total plaintext/ciphertext/timing tuples, we calculate Γ_i. We then can compute the expected value of Γ_i, which we label ϕ, such that

$$\phi = \frac{1}{N} \sum_{i=1}^{N} \Gamma_i$$

Since ϕ is the expected value of Γ_i, we can then write:

$$\begin{aligned} E\{\Gamma_i\} &= E\{[(w + s_{HW}) - (\mu_w + \tilde{s}_{HW})]^2\} \\ &= E\{[(w - \mu_w) + (s_{HW} - s_{\tilde{HW}})]^2\} \\ &= E\{(w - \mu_w)^2 + 2(w - \mu_w)(s_{HW} - s_{\tilde{HW}}) \\ &\quad + (s_{HW} - s_{\tilde{HW}})^2\} \\ &= E\{(w - \mu_w)^2\} + E\{2(w - \mu_w)(s_{HW} - s_{\tilde{HW}})\} \\ &\quad + E\{(s_{HW} - s_{\tilde{HW}})^2\} \end{aligned}$$

We can see that the first term of the expected value is just the variance of w, σ_w^2, that the second term will go to zero since the expected value of the difference between a random quantity and its mean is zero, and that the third term is simply a constant. Hence,

$$\phi = \sigma_w^2 + (s_{HW} - \tilde{s}_{HW})^2$$

Now, for the correct guess at the key Hamming weight, $s_{HW} = \tilde{s}_{HW}$ and $(s_{HW} - \tilde{s}_{HW})^2 = 0$. In all other cases, $(s_{HW} - \tilde{s}_{HW})^2 > 0$, since it is a square, and thus a positive quantity. We may then conclude that ϕ will be at a minimum value when the correct Hamming weight is guessed. Through the application of this attack, we can determine the Hamming weight of the expanded key.

In order to estimate the number of swaps that occurred in the encryption due to data-based control vectors, we must calculate the expected values of the number of swaps in data-controlled DDP blocks. As can be seen from Figure 1, there are only two types of DDPs used: $DDP_{32/48}$ and $DDP_{32/80}$. The value μ_w is the sum of the expected values of the different DDP blocks, which are simply equal to half the number of control vector bits. The expected number of swaps that occur in all the relevant DDP blocks in the cipher that are not controlled by subkey bits is, hence, $\mu_w = 888$. Note that σ_w^2 can also be computed, accounting for the dependencies of control vector bits in the DDPs. However, the value of σ_w^2 is not dependent on the key guess and, since the attack only requires the identification of the smallest value of ϕ, the value of σ_w^2 is not needed in the attack.

6 Experimental Results

We investigated the attack by executing an experiment where 1000 random keys were generated. For each of the keys, a number (N) of plaintexts was randomly generated. Each of the plaintexts was encrypted and timing information about that encryption was recorded. Trial attacks were executed on plaintext sets of size N = 100, 1,000, 10,000, and 100,000. Using the recorded information, Γ and ϕ were calculated and the key Hamming weight with the minimum value of ϕ was selected as the correct value. It should be noted that the Hamming weight of the key is for the fully expanded 384-bit key, which is likely derived from a smaller initial cipher key.

As can be seen in Figure 3, an increase in the number of ciphertexts available increases the likelihood of correctly guessing the Hamming weight of the expanded key. Approximately a 94% success rate can be obtained with only 10^4 plaintext/ciphertext pairs. Given 10^5 plaintext/ciphertext pairs, 100% of the Hamming weights were successfully identified.

7 Timing Attack Safe Implementation

The weakness in the cipher arises from a naive software implementation of the DPP blocks. In order to thwart the timing attack, one simply needs to restructure the implementation of the $DDP_{2/1}$ function such that it executes

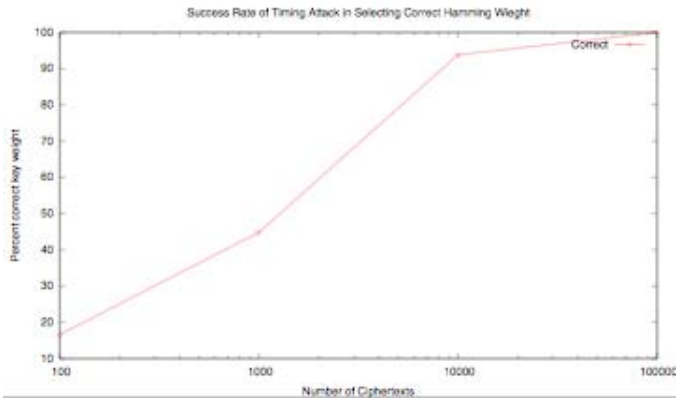

Figure 3: Attack success rate

in constant time. Initially one might be tempted to use the boolean algebra provided by many programming languages. However this may still reveal timing information as many compilers allow for "short-circuiting" when computing boolean expressions. In this case, should an expression evaluate to be true based on the first condition, the remainder of the expression will not be computed and thus would take less time, again revealing timing information. The following solution is recommended for use in C or C-like programming languages:

```
void ddp2_1(int& x0, int& x1, int v) {
  int y0 = (x0 * v) + (x1 * ( 1 - v ));
  int y1 = (x0 * ( 1 - v )) + (x1 * v);
  x0 = y0;
  x1 = y1;
}
```

This approach executes in constant time and thus does not reveal information about the data x0, x1, and v. There may be more efficient solutions that also hide information about the Hamming weight of the control vector, but this solution will fit into the interface provided by the naive implementation studied in this paper.

8 Conclusion

In this paper, we have described a weakness in the CIKS-1 block cipher. We have demonstrated the effectiveness of a timing attack on an implementation of CIKS-1 where the time to execute a data-dependent permutation takes non-constant time. This situation could arise where the cipher has been implemented in software with a naive, straightforward implementation of the data-dependent permutations. Such an implementation may occur on a microcontroller in an embedded system such as a smart card.

The timing attack presented in this paper does rely on accurate timing measurements on individual encryptions. This information would be difficult to obtain in a mutlithreaded environment like most modern general purpose operating systems. However, it is quite conceivable that the attack can be modified and implemented in an environment where the timing measurements are noisy. In any case, this paper shows a vulnerability which designers must be aware of when implementing CIKS-1, as indeed must be the designers of any cipher using DDPs as cryptologic primitives. Fortunately this concern can be eliminated by ensuring the DDPs occur in constant time, thus protecting the cipher from this timing attack.

References

[1] A. Moldovyan and N. Moldovyan, "A Cipher Based on Data-Dependent Permutaitons," , vol. 15, pp. 61–72, 2001.

[2] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in . Springer-Verlag, 1996, pp. 104–113.

[3] H. Handschuh and H. M. Heys, "A Timing Attack on RC5," in . Springer-Verlag, 1999, pp. 306–318.

[4] C. Lee, D. Hong, S. Lee, S. Lee, H.-J. Yang, and J. Lim, "A Chosen Plaintext Linear Attack on Block Cipher CIKS-1," in . Springer-Verlag, 2002, pp. 456–468.

[5] B. J. Kidney, H. M. Heys, and T. S. Norvell, "A Differential Attack on the CIKS-1 Block Cipher," in . IEEE, 2004.

[6] ——, "A Weight Based Attack on the CIKS-1 Block Cipher," in , Nov 2003.