J. Cryptology (1998) 11: 209–218

Journal of
CRYPTOLOGY

Two-Key Triple Encryption

Ivan B. Damgård and Lars R. Knudsen*
Matematisk Institut, Aarhus University,
Ny Munkegade 116, DK-8000 Aarhus C, Denmark

Communicated by James L. Massey

Received 22 June 1995 and revised 11 October 1996

Abstract. In this paper we consider multiple encryption schemes built from conventional cryptosystems such as DES. The existing schemes are either vulnerable to variants of meet-in-the-middle attacks, i.e., they do not provide security corresponding to the full key length used or there is no proof that the schemes are as secure as the underlying cipher. We propose a variant of two-key triple encryption with a new method of generating three keys from two. Our scheme is not vulnerable to the meet-in-the-middle attack and, under an appropriate assumption, we can show that our scheme is at least about as hard to break as the underlying block cipher.

Key words. Multiple encryption, Block ciphers, The Data Encryption Standard, Pseudorandom generators.

1. Introduction

Since its introduction in the late seventies, the American Data Encryption Standard (DES) has been the subject of intense debate and cryptanalysis. Like any other practical cryptosystem, DES can be broken by searching exhaustively for the key.

One natural direction of research is therefore to find attacks that will be faster than exhaustive search, measured in the number of necessary encryption operations. The most successful attack on DES known of this kind is the linear attack by Matsui [7], [8]. This attack requires about 2^{43} known plaintext blocks. Although this is less than the expected 2^{55} encryptions required for exhaustive key search, the attack is by no means more practical than exhaustive search. There are two reasons for this: first, in practice the time needed to obtain the information about the plaintext cannot be neglected; secondly, when doing exhaustive key search the enemy is free to invest as much in technology as he is capable of to make the search more efficient, in a known-plaintext attack he is basically restricted to the technology of the legitimate owner of the key, and to the frequency with which the key is used. In virtually any practical application, a single DES key will be

* Postdoctoral researcher sponsored by the Danish Technical Research Council.

applied to a lot less than 2^{43} blocks, even in its entire lifetime. The difference between the two kinds of attacks is illustrated in a dramatic way by the results of Wiener [14] who shows by concrete design of a key search machine that if the enemy is willing to make a $1 million investment, exhaustive key search for DES is certainly not infeasible and can be done in a few hours.

As a result, we have a situation where DES has proved very resistant over a long period to cryptanalysis and therefore seems to be as secure as it can be in the sense that by far the most practical attack is a simple brute-force search for the key. The only problem is that the key is too short given today's technology, and that therefore, depending on the value of the data being protecting, plain DES may not be considered secure enough anyway.

What can be done about this problem? One obvious solution is to try to design a completely new algorithm. This can only be a long term solution: a new algorithm has to be analyzed over a long period before it can be considered secure; also the vast number of people who have invested in DES technology will not like the idea of their investments becoming worthless overnight. An alternative is to devise a new system with a longer key using DES as a building block. This way existing DES implementations can still be used.

Thus we are in the situation where we have a block cipher that has proved to be very strong, the only problem being that the keys are too small and a simple brute-force attack has become possible. We therefore ask the following general question: Given cryptosystem $\mathcal{X}$, which cannot in practice be broken faster than exhaustive key search, how can we build a new system $\mathcal{Y}$, such that

1. keys in $\mathcal{Y}$ are significantly longer than keys in $\mathcal{X}$ (e.g., twice as long),
2. given an appropriate assumption about the security of $\mathcal{X}$, $\mathcal{Y}$ is provably almost as hard to break as $\mathcal{X}$ under any natural attack (e.g., ciphertext-only, known-plaintext, etc.),
3. it can be convincingly argued that $\mathcal{Y}$ cannot be broken faster than by an exhaustive key search, and is therefore in fact much stronger than $\mathcal{X}$?

Possible answers to this question have already appeared in the literature. The most well-known example is known as two-key triple encryption, where we encipher under one key, decipher under a second key, and finally encipher under the first key. Van Oorschot and Wiener [12] have shown, refining an attack of Merkle and Hellman [10], that this construction is not optimal: under a known-plaintext attack, it can be broken significantly faster than by an exhaustive key search. We propose a new variant of two-key triple encryption, which we conjecture has all the properties we require above. As indicated in our second demand we will not be able to prove that our new scheme is as secure as the underlying block cipher without an appropriate assumption about the latter, explicitly stated later in Hypothesis 1.

Note that, in general, a block cipher $\mathcal{X}$ may have the property that successive encryptions under several keys is equivalent to encryption under one key (i.e., $\mathcal{X}$ is a group [5]). In [2] it was shown that the DES is not a group. Although the result we are about to show in fact holds in general, it is clear that the basic idea of multiple encryption is not useful if $\mathcal{X}$ is a group. We therefore assume in the following that $\mathcal{X}$ is not a group.

For the remainder of this paper we use as measurement of the time and memory needed

by an attack on a block cipher, the number of encryptions it requires and the number of plaintext or ciphertext blocks which need to be stored, respectively. Also, if not stated otherwise we mean by "breaking a cipher" that the secret key is found.

2. Multiple Encryption

In this section we look at methods for enhancing cryptosystems based on the idea of encrypting plaintext blocks more than once. Following the notation of the Introduction, we let $\mathcal{X}$ be the original system, and we let E_K and D_K denote encryption and decryption, respectively, in $\mathcal{X}$ under key K. We assume that the key space of $\mathcal{X}$ consists of all k-bit strings and that the block length of $\mathcal{X}$ is m. In a *cascade of ciphers* it is assumed that the keys of the component ciphers are independent. The following result was proved by Even and Goldreich.

Theorem 1 [3]. *A cascade of ciphers is at least as hard to break as any of the component ciphers in attacks where an attacker cannot make use of plaintext statistics.*

As seen, the result establishes a connection between the security of a cascade of ciphers and of the underlying ciphers. The following result covering all attacks was proved by Maurer and Massey.

Theorem 2 [9]. *Under any attack, a cascade of ciphers is at least as hard to break as the first cipher.*

The two results hold for any reasonable definition of breaking a cipher [3], [9], e.g., they hold for key-recovery attacks as well as for attacks that find a plaintext given a ciphertext.

A special case of a cascade of ciphers is when the component ciphers are equal, also called multiple encryption. In the following we consider different forms of multiple encryption.

2.1. *Double Encryption*

The simplest idea one could think of would be to encrypt twice using two keys K_1, K_2, i.e., let the ciphertext corresponding to P be $C = E_{K_2}(E_{K_1}(P))$. It is clear (and well known), however, that no matter how K_1, K_2 are generated, there is a simple meet-in-the-middle attack that breaks this system with a few known plaintexts using 2^k encryptions and 2^k blocks of memory, i.e., the same time complexity as key search in the original system. Also, in [13] van Oorschot and Wiener present a time/memory tradeoff of the meet-in-the-middle attack on double encryption. With 2^w words of memory the expected time complexity of the attack is $7 \times 2^{(3k-w)/2}$. Using DES as an example, an attack with $w = 30$, that uses 16 Gbytes of memory, will have an expected running time of 2^{72}.

Note that our goal is to build, based on $\mathcal{X}$, something on the same security level as a secure cipher designed from scratch to have key length $2k$. Attacks such as the above should certainly not be possible against such a cipher, and it is therefore clear that double encryption is not sufficient for us.

2.2. *Triple Encryption*

Triple encryption with three independent keys K_1, K_2, and K_3, where the ciphertext corresponding to P is $C = E_{K_3}(E_{K_2}(E_{K_1}(P)))$, is also not a satisfactory solution for a similar reason as for double encryption. A simple meet-in-the-middle attack will break this in time about 2^{2k} encryptions and 2^k blocks of memory with a few known plaintexts. Thus we do not get full return for our effort in tripling the key length—as stated in demand 3 in the Introduction, we would like attacks to take time close to 2^{3k} if the key length is $3k$. In addition to this, if $\mathcal{X} =$ DES, then a simple triple encryption would preserve the complementation property and preserve the existence of weak keys. Recently, it was shown that if an attacker can mount a so-called *related key* attack, triple encryption can be broken in time about 2^k [6]. The attack requires that the attacker can get the encryptions of a small number of known plaintexts under two sets of keys. The two triples of keys must differ only in the third keys with a difference known to the attacker.

It is clear, that no matter how the three keys in triple encryption are generated, the meet-in-the-middle attack mentioned is still possible, and so the time complexity of the best attack against *any* triple encryption variant is no larger than 2^{2k}. Such a security level should be obtainable with a key length of only $2k$. It therefore seems reasonable to try to generate the three keys from two independent $\mathcal{X}$-keys K_1, K_2.

2.3. *Two-key Triple Encryption*

One variant of this idea is well known as two-key triple encryption, proposed by Tuchmann [11]. The ciphertext C corresponding to the plaintext P is $C = E_{K_1}(D_{K_2}(E_{K_1}(P)))$. Compatibility with a single encryption can be obtained by setting $K_1 = K_2$. As can be seen, this uses a particular very simple way of generating the three keys from K_1, K_2.

The proof of the following result can be derived from that of Theorem 1 [3].

Theorem 3. *In attacks where an attacker cannot make use of plaintext statistics two-key triple encryption is at least as hard to break as it is to break a cryptosystem that uses a single decryption function of the underlying block cipher for encryption.*

Even though this result establishes some connection between the security of two-key triple encryption and single encryption, it does not hold for all attacks and still does not meet our second demand.

It is interesting to note that the related-key attack on a triple encryption scheme is not applicable to two-key triple encryption [6].

However, for the two-key triple encryption scheme, each of K_1 and K_2 only influences particular parts of the encryption process. Because of this, variants of the meet-in-the-middle attack are possible that are even faster than exhaustive search for K_1, K_2. In [10] Merkle and Hellman describe an attack on two-key triple DES encryption requiring 2^{56} chosen plaintext–ciphertext pairs and a running time of 2^{56} encryptions using 2^{56} words of memory. This attack was refined in [12] into a known-plaintext attack on the DES, which on input n plaintext–ciphertext pairs finds the secret key in time $2^{120}/n$ using n words of memory. The attacks can be applied to any block cipher. Therefore two-key triple encryption does not meet our third demand.

We therefore propose what we believe to be stronger methods for generating the keys.

Our main idea is to generate them *pseudorandomly* from two $\mathcal{X}$-keys, using a generator based on the security of $\mathcal{X}$. In this way, an enemy trying to break $\mathcal{Y}$ either has to treat the three keys as if they were really random which means he has to break $\mathcal{X}$, according to Theorem 2; or he has to use the dependency between the keys—this mean breaking the generator which was also based on $\mathcal{X}$! Thus, even though we have thwarted attacks like Merkle–Hellman and van Oorschot–Wiener by having a strong interdependency between the keys, we can still, if $\mathcal{X}$ is secure enough, get a connection between the security of $\mathcal{X}$ and $\mathcal{Y}$. In the following we concentrate on triple encryption schemes, but our results generalize to any n-fold schemes.

3. Multiple Encryption with Minimum Key

3.1. *General Description of* $\mathcal{Y}$

Let a block cipher $\mathcal{X}$ be given, as described above. The key length of $\mathcal{X}$ is denoted by k. By $E_K(P)$, we denote $\mathcal{X}$-encryption under K of block P, while $D_K(C)$ denotes decryption of C. We then define a new block cipher $\mathcal{Y}$ using a function G:

$$G(K_1, K_2) = (X_1, X_2, X_3)$$

which maps two $\mathcal{X}$-keys to three $\mathcal{X}$-keys. We display later a concrete example of a possible G-function. This is constructed from a few $\mathcal{X}$-encryptions. Keys in $\mathcal{Y}$ will consist of pairs (K_1, K_2) of $\mathcal{X}$-keys. Encryption in $\mathcal{Y}$ is defined by

$$E_{K_1,K_2}(P) = E_{X_3}(E_{X_2}(E_{X_1}(P))),$$

where $(X_1, X_2, X_3) = G(K_1, K_2)$. Decryption is clearly possible by decrypting using the X_i's in reverse order.

3.2. *Relation to the Security of* $\mathcal{X}$

We would like to be reasonably sure that we have taken real advantage of the strength of $\mathcal{X}$ when designing $\mathcal{Y}$. One way of stating this is to say that $\mathcal{Y}$ is at least as hard to break as $\mathcal{X}$. By Theorem 2, this would be trivially true if the three keys used in $\mathcal{Y}$ were statistically independent. This is of course not the case, since the X_i's are generated from only two keys. However, if the generating function G has a pseudorandom property as stated below, then the X_i's are "as good as random" and we can still prove a strong enough result.

Definition 1. Consider the following experiment: an enemy B is presented with three k-bit blocks X_1, X_2, X_3. He then tries to guess which of two cases has occurred:

1. The X_i's are chosen independently at random.
2. The X_i's are equal to $G(K_1, K_2)$, for randomly chosen K_1, K_2.

Let p_1 be the probability that B guesses 1 given that case 1 occurs, and let p_2 be the probability that B guesses 1 given that case 2 occurs. The generator function G is said to be *pseudorandom*, if, for any strategy followed by B spending time equal to T encryption

operations,

$$|p_1 - p_2| \leq \frac{T}{V},$$

where V is the total number of keys in $\mathcal{X}$.

The intuition we want to express with this definition is that the generator function G should be at least as hard to break as it is to do exhaustive search for a key in system $\mathcal{X}$. Clearly, if the total number of keys is V, and resources for testing T randomly chosen keys are available, then the probability of finding the correct one is T/V. We therefore say that this also should be the maximum amount of success that can be achieved against G using time T. Definition 1 is inspired by the complexity-theoretic definition of a strong pseudorandom generator introduced by Blum and Micali [1]. An example of a conjectured pseudorandom generator is given in Section 3.3.

In the rest of this subsection we consider attacks against $\mathcal{X}$ and $\mathcal{Y}$ in a fixed scenario with a given plaintext distribution and a given form of attack, which can be ciphertext-only, known-plaintext, or chosen-plaintext. The attacks considered have a specific goal which can be either to find the key used, or to decrypt a given ciphertext. Each time an attack is executed, it has a certain probability of success, taken over the plaintext distribution, the choice of keys, and its own random choices. We do not specify the scenario further here because the reasoning below will work for any scenario of the form we have described. As usual, the time unit will be the number of encryption operations in system $\mathcal{X}$.

The next theorem shows the promised connection between security of $\mathcal{X}$ and $\mathcal{Y}$, i.e., in a given amount of time, an attack cannot do much better against $\mathcal{Y}$ than what is possible against $\mathcal{X}$.

Theorem 4. *Let p be the largest success probability that can be achieved by an attack against $\mathcal{X}$ running in time T. Assume now that an attack A against our new system $\mathcal{Y}$ runs in time T and has success probability $p + \varepsilon$. If the function G used to construct $\mathcal{Y}$ is pseudorandom, as defined in Definition* 1*, then*

$$\varepsilon \leq \frac{T}{V},$$

where V is the total number of keys in $\mathcal{X}$.

Proof. Let $\mathcal{Y}_0$ be the same system as $\mathcal{Y}$, but with independent keys X_i. It follows directly from Theorem 2 and by the definition of p, that any attack against $\mathcal{Y}_0$ running in time T will have a success probability of at most p. Therefore, A's probability of success against $\mathcal{Y}_0$ is at most p. We can use A as a subroutine in an algorithm B to decide whether some given X_1, X_2, X_3 were chosen independently at random or were the output of a pseudorandom generator, see Definition 1, whose inputs K_1 and K_2 were chosen independently at random. Given X_1, X_2, X_3, B uses these as keys in the triple encryption system and simulates A's attack. If A does not succeed, i.e., in the fixed scenario as discussed above, B will guess that the X_i's are independent, and if A succeeds, B will guess that they were generated from K_1, K_2. Since if case 1 of the

experiment in Definition 1 occurred, A will be attacking $\mathcal{Y}_0$, when X_1, X_2, X_3 were chosen independently at random, and if case 2 occurred, A will be attacking $\mathcal{Y}$, when X_1, X_2, X_3 were output from G, it is clear that for this B, the probabilities p_1 and p_2 of Definition 1 satisfy

$$p_1 \leq p, \tag{1}$$
$$p_2 = p + \varepsilon, \tag{2}$$

where (1) was argued above, and (2) follows from the assumption on A. Since G is pseudorandom by assumption, it follows from Definition 1 that

$$\varepsilon \leq |p_1 - p_2| \leq \frac{T}{V}. \qquad \square$$

To see more clearly what the statement of the theorem means, consider an ideal case, where the best an attack against $\mathcal{X}$ can do is to spend its time choosing random keys and test whether they fit with the information available. The success probability for time T would then be $T/2^k$ assuming a key can be tested in one encryption. This is one particular case, where the success probability is directly proportional to the running time of the attack. In general we get:

Corollary 1. *If the generator G used to construct $\mathcal{Y}$ is pseudorandom, see Definition* 1, *and if any attack against $\mathcal{X}$ running in time T has success probability at most δT, $\delta \geq 2^{-k}$, then any attack against $\mathcal{Y}$ running in time T has success probability at most $2\delta T$.*

Proof. By Theorem 4 it follows that an attack against $\mathcal{Y}$ running in time T has a success probability of

$$p + \varepsilon \leq \delta T + \frac{T}{2^k} \leq 2\delta T. \qquad \square$$

For the case of attacks that break Y with probability 1, we get:

Corollary 2. *If the generator G used to construct $\mathcal{Y}$ is pseudorandom, see Definition* 1, *and if any attack against $\mathcal{X}$ running in time T has success probability at most $T/2^k$, then any algorithm breaking $\mathcal{Y}$ with certainty requires time at least 2^{k-1}.*

3.3. *A Concrete Two-Key Triple Encryption Construction*

We propose here a new construction for triple encryption, called **3-PEK** for triple encryption with pseudorandomly expanded keys. As before, the key length of $\mathcal{X}$ is k and the block length is m.

The keys X_1, X_2, X_3 are all used as keys for encryption. With $k = m$ we define this construction of $G(K_1, K_2) = (X_1, X_2, X_3)$ by

$$X_1 = E_{K1}(E_{K2}(IV_1)),$$
$$X_2 = E_{K1}(E_{K2}(IV_2)),$$
$$X_3 = E_{K1}(E_{K2}(IV_3)),$$

where IV_i are three different initial values, e.g., $IV_i = C + i$, where C is a constant. The above construction is easily extended to the case, where $k \neq m$ for $\mathcal{X}$. Simply generate $\lceil 3k/m \rceil$ ciphertexts using $\lceil 3k/m \rceil$ different initial values. The three $\mathcal{X}$ keys are constructed by selecting the $3k$ bits from these ciphertexts.

In the following we argue that our construction meets our three demands from the Introduction. So we assume that the underlying block cipher $\mathcal{X}$ cannot in practice be broken faster than an exhaustive search. Our first demand is met since the keys in our scheme are twice as long as in the underlying block cipher. It is seen that double encryption is used to generate the three keys. Therefore, based on Theorem 2, we make the following hypothesis.

Hypothesis 1. *Let $\mathcal{X}$ be a block cipher with key size k and block size m. If block cipher $\mathcal{X}$ cannot be broken faster than exhaustive key search under a known-plaintext attack using at most $\lceil 3k/m \rceil$ known plaintexts, then the generator G defined above is pseudorandom according to Definition* 1.

The reason for believing this hypothesis is that given X_1, X_2, X_3 and IV_1, IV_2, IV_3, it seems that an enemy would have to do a known-plaintext attack against double encryption with $\mathcal{X}$ in order to decide if the X_i's are random. By Theorem 2 and the assumption on $\mathcal{X}$, this requires at least as much time as an exhaustive key search in $\mathcal{X}$.

For the case of $\mathcal{X} =$ DES, we note that even though DES can in theory be broken faster than exhaustive key search, this requires an enormous number of known plaintexts (currently 2^{43}), and since in this case the enemy is given only three known plaintexts double encrypted with DES, we believe that the hypothesis is true for DES as well.

Under the hypothesis, Theorem 4 shows that our second demand is satisfied.

Note that although the hypothesis implies that $\mathcal{Y}$ is at least on the same security level as $\mathcal{X}$, this of course does not mean that it is not more secure. In fact, we conjecture that if the most efficient attack against $\mathcal{X}$ is a brute-force key search, then the most efficient attack against 3-PEK encryption is a brute-force search for the key of time complexity 2^{2k}. Attacks like the ones from [10] and [12] are applicable to ciphers for which the first and the third keys are equal. Since in our case every one of the three keys defined above are dependent on both master keys K_1 and K_2 in a complicated way (and are certainly different!) we conclude that these attacks are not applicable. Thus, it can be argued (but not proved) that our new scheme is much harder to break than the underlying block cipher and all our three demands from the Introduction are met.

In Table 1 a schematic overview is given of the time complexities of attacks against our new scheme and existing ones. The table contains time complexities for attacks that break the cipher in question with probability 1. For two-key triple encryption, we know of no lower bound on the complexities of any attack and the upper bound results from the meet-in-the-middle attacks described earlier in this paper. For three-key triple encryption the lower bound is by the result of Maurer and Massey and the upper bound is a simple meet-in-the-middle attack believed to be the best possible. For 3-PEK encryption the lower bound follows if the generator G is pseudorandom, see Definition 1 and Corollary 2, and the upper bound is our conjecture of the best attack. We challenge the reader to come up with attacks violating this upper bound.

We note, for the case of $\mathcal{X} =$ DES, it is unlikely that 3-PEK has any weak keys, and

Table 1. Bounds on the time complexities of attacks on the proposed scheme and the existing ones.

Scheme*	Key size	Lower bound (all attacks)	Upper bound (best known attack)
(1) Block cipher $\mathcal{X}$	k	2^k	2^k
(2) Two-key triple $\mathcal{X}$	$2k$	Unknown	2^k
(3) Three-key triple $\mathcal{X}$	$3k$	2^k	2^{2k}
(4) 3-PEK $\mathcal{X}$	$2k$	2^{k-1}	2^{2k}

*(1) By assumption on $\mathcal{X}$. (2) and (3) Upper bound requires storage of 2^k words. (4) Lower bound holds, assuming G is pseudorandom, see Definition 1.

the complementation property [4] does not hold. Finally, we note that the related-key attack on triple-DES, described in [6], is not applicable to our scheme.

4. Conclusion

We considered multiple encryption schemes built from conventional cryptosystems. We showed how to build triple encryption schemes with a security strongly connected to the security of the underlying block cipher and not vulnerable to the special meet-in-the-middle attacks on existing schemes. We focused on triple encryption schemes, but it is clear that our ideas can be extended to n-fold schemes.

Acknowledgment

We would like to thank the referees of this paper for useful comments.

References

[1] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13:850–864, 1984.

[2] K.W. Campbell and M.J. Wiener. DES is not a group. In E.F. Brickell, editor, *Advances in Cryptology—CRYPTO '92*, LNCS 740, pages 512–520. Springer-Verlag, Berlin, 1993.

[3] S. Even and O. Goldreich. On the power of cascade ciphers. *ACM Transactions on Computer Systems*, 3:108–116, 1985.

[4] M.E. Hellman, R. Merkle, R. Schroeppel, L. Washington, W. Diffie, S. Pohlig, and P. Schweitzer. Results of an initial attempt to cryptanalyze the NBS Data Encryption Standard. Technical Report, Stanford University, Stanford, CA, September 1976.

[5] B.S. Kaliski, R.L. Rivest, and A.T. Sherman. Is the data encryption standard a group? (Results of cycling experiments on DES.) *Journal of Cryptology*, 1(1):3–36, 1988.

[6] J. Kelsey, B. Schneier, and D. Wagner. Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES. In N. Koblitz, editor, *Advances in Cryptology—Proc. CRYPTO '96*, LNCS 1109, pages 237–251. Springer-Verlag, Berlin, 1996.

[7] M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology—Proc. Eurocrypt '93*, LNCS 765, pages 386–397. Springer-Verlag, Berlin, 1993.

[8] M. Matsui. The first experimental cryptanalysis of the Data Encryption Standard. In Y.G. Desmedt, editor, *Advances in Cryptology—Proc. Crypto '94*, LNCS 839, pages 1–11. Springer-Verlag, Berlin, 1994.

[9] U. Maurer and J.L. Massey. Cascade ciphers: the importance of being first. *Journal of Cryptology*, 6(1):55–61, 1993.
[10] R. Merkle and M. Hellman. On the security of multiple encryption. *Communications of the ACM*, 24(7):465–467, 1981.
[11] W. Tuchman. Hellman presents no shortcut solutions to DES. *IEEE Spectrum*, 16(7):40–41, July 1979.
[12] P.C. van Oorschot and M.J. Wiener. A known-plaintext attack on two-key triple encryption. In I.B. Damgård, editor, *Advances in Cryptology—Proc*. *Eurocrypt* '90, LNCS 473, pages 318–325. Springer-Verlag, Berlin, 1990.
[13] P.C. van Oorschot and M.J. Wiener. Improving implementable meet-in-the-middle attacks of orders of magnitude. In N. Koblitz, editor, *Advances in Cryptology—Proc*. *CRYPTO* '96, LNCS 1109, pages 229–236. Springer-Verlag, Berlin, 1996.
[14] M.J. Wiener. Efficient DES key search. Technical Report TR-244, School of Computer Science, Carleton University, Ottawa, Ontario, May 1994. Presented at the Rump Session of Crypto '93.