How to Forge DES-Encrypted Messages in 2^{28} Steps

Eli Biham[1]

Abstract

In this paper we suggest *key-collision attacks*, and show that the theoretic strength of a cipher cannot exceed the square root of the size of the key space. As a result, in some circumstances, some DES keys can be recovered while they are still in use, and these keys can then be used to forge messages: in particular, one key of DES can be recovered with complexity 2^{28}, and one key of (three-key) triple-DES can be recovered with complexity 2^{84}.

Keywords: Cryptanalysis, Data Encryption Standard (DES), Triple-DES, Multiple Encryption, Birthday Paradox, Key-Collision Attacks.

1 Introduction

Cryptographic keys are used as secret information during encryption, and whose knowledge is required to decrypt the ciphertexts. Most currently used ciphers use keys of 56 bits, 64 bits and 80 bits, and some ciphers use keys of 128 bits. Ciphers with k-bit keys can be used with 2^k distinct keys, and therefore, it is believed that the complexity of attacking ciphers with k-bit keys (or forging messages encrypted with such ciphers) is not less than 2^k, unless the design of the cipher is weak. It is also believed that forging messages is as difficult as breaking the cipher, and that frequent key replacements increase the security of the cipher, or at least cannot reduce its strength. In this paper we show that all these beliefs are wrong.

In the particular case of the Data Encryption Standard (DES)[21], several attacks were suggested: Differential cryptanalysis[5] requires 2^{47} chosen plaintexts and complexity in order to find the key, linear cryptanalysis[17] requires 2^{43} known plaintexts and complexity, and the improved Davies' attack[4] requires 2^{50} known plaintexts. However, the main threat for the security of DES is exhaustive search for the keys on

[1]Computer Science Department, Technion - Israel Institute of Technology, Haifa 32000, Israel. Email: biham@cs.technion.ac.il, WWW: http://www.cs.technion.ac.il/~biham/.

special purpose machines[10,27], which can try keys so fast so that all the 2^{56} possible keys can be searched within only a few hours.

In order to increase the strength of ciphers, multiple encryption was suggested, of which triple-encryption became the most popular variant, after double-encryption was shown to be theoretically no more secure than single encryption by meet in the middle attacks[18]. Triple-encryption is usually defined as $C = E_{K_3}(D_{K_2}(E_{K_1}(P)))$, where the key K of the triple encryption is formed from three independent single-encryption keys $K = (K_1, K_2, K_3)$. For sake of simplicity, and since it does not affect the results, we refer to triple encryption as doing encryption in all its components ($C = E_{K_3}(E_{K_2}(E_{K_1}(P)))$). A triple encryption variant which uses two keys[26] (where $K_3 = K_1$) was believed to be roughly as secure as three-key triple-encryption, but was shown to be theoretically (but not practically) no more secure than a single encryption[10,23]. Another (seemingly more secure) two-key triple-DES variant was suggested in [9]. Nowadays, triple-encryption and its two-key variant are candidates for an ANSI standard[1].

The most successful attacks on multiple encryption are meet in the middle attacks. Multiple encryption with an even number m of encryptions, such as double encryption ($m = 2$), or quadruple encryption ($m = 4$), can be analyzed with $2^{km/2}$ steps. However, multiple encryption with an odd m requires $2^{k(m+1)/2}$ steps, and in particular, triple-DES requires 2^{112} steps for a meet in the middle attack, although the key size is 168 bits; this complexity is the same as required for a similar attack on quadruple-DES with 224 key bits.

The ciphers are usually used in environments that require frequent key replacements. For example, bank communications transfer huge amounts of money, and the banks need to change keys before any of the used keys are to be found by an attacker, even if the attacker spends as much in the cryptanalysis as he might get by forging encrypted messages. Therefore, the attacker might receive many encrypted messages, encrypted under many distinct keys, while the amount of data encrypted under any particular key is small and hopefully does not help for the cryptanalysis of that key. However, in many such cases, the headers of the encrypted messages are similar, either due to added prefix in communication messages, or due to some standard prefix of a file format or a programming language (like the LaTeX `\documentstyle` or the PostScript `%!PS-Adobe-2.0` prefixes). A similar situation occurs when a disk contains many files. Current disks have more than a gigabyte space; in a typical such disk there might be an order of 100000 files. If all the files are encrypted (and especially if the files are automatically encrypted under random keys by a cryptographic file system), and all of them are written in the same format/language, we can also receive the prefix encrypted under many distinct keys.

An hidden assumption is that not only it is difficult to recover keys of particular ciphertexts, it is also difficult to recover even one of the keys, since it is unacceptable that even one messages be forged successfully. The importance of this assumption is clear, especially in financial communications. This behavior motivates the following

definition:

We define the *theoretic strength* of a cipher as the minimal complexity t, such that given up to t plaintext/ciphertext pairs, possibly encrypted under different keys, an analysis taking up to t steps can recover (at least) one of the keys with a high probability.

Differential[5] and linear[17] cryptanalysis suggest that the theoretic strength of DES is bounded by 2^{47} and 2^{43} respectively. In the case of differential cryptanalysis, this result holds even if every structure of 2^{14} ciphertext blocks is encrypted under a different key; in this case the attack finds *one* of the keys, while it is still in use.

The use of the birthday paradox in cryptography is well known, but is usually limited to analysis of hash functions and compressed encodings[28,16,8,12]. This paradox suggests that in a class of 23 children, there is probability more than a half to have two children with the same birthday date. In general, if some property (the birthday) might get n distinct values, there is a high probability that even in about $\sqrt{n}$ entities there is a pair with the same value of the property. A variant of this paradox shows that given two classes, each of about $\sqrt{n}$ entities, there is a high probability that some entity of the first class has the same value of the property as some entity in the second class.

In this paper we use the birthday paradox to show that the theoretic strength of a cipher is bounded by the square root of the size of the key space. We describe a new attack which given about $2^{k/2}$ encrypted messages (whose headers are the same, and each is encrypted under a different key) can find one of the keys with complexity $2^{k/2}$. Most computation can be done in advance, so that only one table lookup is required online for each given ciphertext, and thus the key is found while it is still viable, and can be used to forge messages. We conclude that the theoretic strength of DES is not more than 2^{28} (encryption of 2^{28} plaintexts takes only 16 seconds on the DEC 1-giga-bit-per-second GaAs DES chip[11] or only 24 minutes in software on an Alpha computer). When keys are frequently changed (and the required information for the attack can be obtained), it might be possible in practice to precompute 2^{28} encryptions and to find one of the 2^{28} keys while it is still in use; this key can then be used to forge messages.

We suggest two tradeoffs between the number of given encrypted messages and the complexity of the attack, so that (1) if fewer messages are given, the complexity of finding one key grows by the same factor, and (2) if the number of encrypted messages is larger, the average investment in each key becomes smaller. In an extreme situation only one trial encryption is required in average for each found key.

We use this method together with the meet in the middle attack to show that the theoretic strength of multiple encryption is not more than $2^{mk/2}$, for any m, given only $2^{k/2}$ encrypted messages, and in particular the theoretic strength of three-key triple-DES is not more than 2^{84}, given 2^{28} encrypted messages.

The results of this paper show that if the same plaintext blocks are encrypted under many keys, the attacker can derive the key of one (or more) messages. Analogous results are already known for public key cryptosystems, and in particular for the RSA cryptosystem[24]: Assume e is small, and the same in e distinct public keys with moduli $n_1, n_2, \ldots, n_e$. Then, if the ciphertexts of some secret message m under the e keys are known, the attacker can derive $m^e \pmod{n_1 \cdot n_2 \cdot \ldots \cdot n_e}$ from the ciphertexts $m^e \pmod{n_i}$, and compute the secret message m as the (non-modular) e'th root of it[13].

We emphasis that the attacks described in this paper are applicable to all types of ciphers, and in particular to blockciphers and stream ciphers, and to any multiple encryption or even multiple mode of operation[2,3]. The attacks can be used with chosen plaintext, known plaintext, and even ciphertext only contexts, if the file, language or communication headers are fixed and publicly known. Although these attacks also apply to public key cryptosystems, the results in such systems are usually worse than directly solving (factoring) the keys of those systems.

In Section 2 we describe the attacks and their possible tradeoffs. In Section 3 we describe the extension to multiple encryption. Finally, we summarize the results, and describe possible immunitions.

2 The Attack

This attack works best when we can receive some plaintext block A, known to us, encrypted under many distinct keys. Such situations arise when a communication layer adds some fixed headers as prefix to the messages before encryption, when a standard fixed prefix is defined for some file format or a programming language, or when we can choose parts of the encrypted messages.

Assume that the cipher has k key bits, and that the size of A is larger than the size of the keys. We choose $2^{k/2}$ keys at random and use them to encrypt A, keeping the results $(E_K(A), K)$ in a table, indexed by the first field. Each time we receive a new ciphertext C, we look in the table whether an entry (C, K) exists, for some K. If so, we conclude that the key K was used to encrypt C. This key is still in use, and can be used to forge messages. It is expected by the birthday paradox that one of the first $2^{k/2}$ ciphertexts discloses its encryption key.

This attack can be slightly modified, so that the number of ciphertexts and the number of random keys vary. Let us choose l random keys, and receive n ciphertexts. The probability to find one of the n keys is

$$1 - \left(1 - l \cdot 2^{-k}\right)^n = 1 - \left(1 - \frac{1}{2^k/l}\right)^n \geq 1 - e^{-ln/2^k}.$$

As long as we choose $ln \geq 2^k$, the probability of finding a key remains high. Table 1

Ciphertexts	1	2^8	2^{16}	2^{24}	$\mathbf{2^{28}}$	2^{32}	2^{40}	2^{48}	2^{56}
Complexity	2^{56}	2^{48}	2^{40}	2^{32}	$\mathbf{2^{28}}$	2^{24}	2^{16}	2^8	1

Table 1. The Tradeoff between the Complexity of Encrypting with the Random Keys to Attack DES and the Number of Required Ciphertexts.

Ciphertexts	2^{28}	2^{29}	2^{30}	2^{32}	2^{40}	2^{48}	2^{56}
Expected Number of Keys Found	1	4	16	2^8	2^{24}	2^{40}	2^{56}
Complexity per Found Key	2^{28}	2^{27}	2^{26}	2^{24}	2^{16}	2^8	1

Table 2. The Average Complexity Invested in Each Found Key of DES.

describes this tradeoff between the complexity of encrypting with the random keys, and the number of required ciphertexts. Moreover, in such situation, we find one of every $2^k/l$ keys, and as l grows, the fraction of found keys grows as well. Note that in the above description, l memory cells are required to keep the table. However, if $l \geq n$, a variant of the attack in which the key is found only after it is already obsolete (thus can be used only to decrypt old messages, rather than to forge messages), requires only n memory cells: we first receive the n ciphertexts, and keep them in the table, and then encrypt A under l random keys, till one of the results equals one of the ciphertexts in the table.

Another tradeoff is between the number of ciphertexts and the investment in analyzing each found key. In this case we rewrite the attack as follows: Each time we receive a ciphertext C, we

1. Enter the ciphertext C to a table T_1.
2. Choose a random key K, and enter $(E_K(A), K)$ into a table T_2, indexed by the first field.
3. Check whether C appears as the index in a pair (C, K') in T_2. If so, K' is expected to be the key which encrypted C.
4. Check if $E_K(A)$ appears in T_1. If so, K is expected to be the key which encrypted it.

In this attack, the first match is expected to appear after about $2^{k/2}$ ciphertexts, so the investment in the first key is about $2^{k/2}$ encryptions. This investment reduces for each additional key, till for 2^k keys the investment in each key is only one encryption. Table 2 describes the average complexity invested in each found key by this algorithm, when more than $2^{k/2}$ ciphertexts are given.

The attacks described in this section can be used even if some random initial value is prepended by the mode of operation (such as the cipher block chaining mode[22]).

In this case, the initial value can be viewed as a part of the key, and thus the birthday paradox is applied on both the key and the initial value.

Assume that the cipher has k key bits, and assume that the size of the initial value (usually the cipher blocksize) is m bits. We choose $2^{(m+k)/2}$ key/initial value pairs (K, IV) at random and encrypt the plaintext A with them by $C = E_K(IV, A)$, keeping the results (C, K, IV) in a table, indexed by the first field. Each time we receive a new ciphertext C, we look in the table whether an entry (C, K, IV) exists for some K and IV. If so, we conclude that the key K (and the initial value IV) was used to encrypt C. This key is still in use, and can be used to forge messages. It is expected by the birthday paradox that one of the first $2^{(m+k)/2}$ ciphertexts discloses its encryption key. The resulting theoretic strength $2^{(m+k)/2}$ is smaller than the 2^k complexity of exhaustive search, if $m < k$. Thus, if the blocksize is 64 bits, and the keysize is 128 bits, key-collision attacks require only 2^{96} steps, while exhaustive search requires 2^{128} steps.

3 Extension for Multiple Encryption

The results of the previous section hold for any cipher, including multiple encryption, where k is replaced by the total size of all the (independent) keys mk. These results show that the complexity of finding one key, in multiple encryption schemes is not more than $2^{mk/2}$, given $2^{mk/2}$ encrypted messages. In this section we show that the same complexity can be reached with only $2^{k/2}$ encrypted messages, when the attack is combined with the meet in the middle attack[18]. When m is even, the meet in the middle attack already requires only $2^{mk/2}$ steps to analyse a given plaintext/ciphertext pair. In this section we concentrate on attacking odd m's, whose best previous attack (the meet in the middle attack) requires $2^{(m+1)k/2}$ steps (see [25, page 360]).

The meet in the middle attack on multiple encryption is as follows. We assume without loss of generality that all the multiple encryption components encrypt (rather than decrypt); otherwise the roles of encryption and decryption in these components are exchanged in the following discussion. Assume that the number of components is m. We divide them into two parts of lengths m_1 and m_2, where $m_1 + m_2 = m$: the first m_1 components and the last m_2 components. When a plaintext/ciphertext pair (P, C) (whose lengths are at least the key size mk) is received, we apply

1. Exhaustively try all the $2^{m_1 k}$ keys $K' = K_1 K_2 \dots K_{m_1}$ of the first m_1 components, and keep the results $(E_{K_{m_1}}(E_{K_{m_1-1}}(\dots(E_{K_1}(P)))), K_1 \dots K_{m_1})$ in a table T, indexed by the first field.

2. Exhaustively try all the $2^{m_2 k}$ keys $K'' = K_{m_1+1} \dots K_m$ of the last m_2 components, each of them is used to decrypt C. If one of the decrypted results $D_{K_{m_1+1}}(D_{K_{m_1+2}}(\dots(D_{K_m}(C))))$ appears as an index in the table T, together with a key K', then the key of (P, C) is expected to be $K = K'K''$.

This method works best when m is even and m_1 and m_2 are chosen as $m_1 = m_2 = m/2$, in which case the complexity of the attack is $2^{mk/2}$ (step 1 takes $2^{mk/2}$ half-encryptions, and step 2 takes $2^{mk/2}$ half-encryptions). For odd m's, the best result is $m_1 = (m-1)/2$ and $m_2 = (m+1)/2$, and the complexity is therefore $2^{(m+1)k/2}$.

We describe now a combination of our attack from Section 2 with the meet in the middle attack, which has the same low complexity as of our attack $2^{mk/2}$, but requires a much smaller number of ciphertexts for odd m's. We assume that the block A, which we later receive encrypted under many keys, is known, either since we choose it in the context of a chosen plaintext attack, or since A is the fixed header of the communication, language, or some file format. The size of A must be larger than the total size of the key, in order to allow the algorithm to find a unique solution. We choose $m_1 = (m+1)/2$ and $m_2 = (m-1)/2$ and do

1. Exhaustively try $2^{mk/2}$ (random) keys $K' = K_1 K_2 \ldots K_{m_1}$ out of the $2^{(m+1)k/2}$ possible keys of the first m_1 components, and keep records of the results and the keys $(E_{K_{m_1}}(E_{K_{m_1-1}}(\ldots(E_{K_1}(A)))), K_1 \ldots K_{m_1})$ in a table T, indexed by the first field.

2. Given a ciphertext C of A under an unknown key, exhaustively try all the $2^{(m-1)k/2}$ keys $K'' = K_{m_1+1} \ldots K_m$ of the last m_2 components, each of them is used to decrypt C. If the result $D_{K_{m_1+1}}(D_{K_{m_1+2}}(\ldots(D_{K_m}(C))))$ appears as an index in the table T, together with a key K', then the key of that encrypted C is expected to be $K = K'K''$.

3. Given about $2^{k/2}$ ciphertexts of A, the last step is applied $2^{k/2}$ times. With a high probability one of the keys of these ciphertexts is found.

The complexity of the first step in this attack is $2^{mk/2}$. The complexity of the second step for each given ciphertext is $2^{(m-1)k/2}$, and given $2^{k/2}$ ciphertexts, the total complexity of this step is $2^{mk/2}$. The first given ciphertext whose first m_1 encryptions are keyed by one of the random choices chosen in the first step leads to the solution of the key. Since we try a fraction of $2^{-k/2}$ of these keys in the first step, we need in average about $2^{k/2}$ ciphertexts to find one key. Therefore, we end with complexity about $2^{mk/2}$ for finding one of the keys of the given $2^{k/2}$ ciphertexts.

As in the case of single encryption, we have the same tradeoffs: increasing the complexity allows to decrease the number of required ciphertexts, and increasing the number of ciphertexts reduce the average investment in each found key. The last tradeoff is also valid for even m's.

Key Size	Theoretic Strength	Ciphers
56	2^{28}	DES
40	2^{20}	US exportable ciphers
64	2^{32}	LOKI[7,6], Feal-N[20], SAFER-SK64[15]
80	2^{40}	Skipjack
128	2^{64}	Feal-NX[19], SAFER-SK128[15], IDEA[14]
k	$2^{k/2}$	Any cipher

Table 3. The Complexities of Attacking Ciphers.

Scheme	Key Size	Theoretic Strength	Required Ciphertexts
Double-DES*	112	2^{56}	1
Two-key triple-DES	112	2^{56}	2^{56}
3-MAK DES[9]	112	2^{56}	2^{56}
Three-key triple-DES	168	2^{84}	2^{28}
Any scheme	k	$2^{k/2}$	$2^{k/2}$
Any double encryption with two keys*	$2k$	2^{k}	1
Any triple encryption with two keys	$2k$	2^{k}	2^{k}
Any triple encryption with three keys	$3k$	$2^{3k/2}$	$2^{k/2}$
Multiple encryption with m encryptions			
odd m	mk	$2^{mk/2}$	$2^{k/2}$
even m*	mk	$2^{mk/2}$	1

* Using the meet in the middle attack[18].

Table 4. The Complexities of Attacking Multiple Encryption Schemes.

4 Summary

The attack we suggested works against any cipher, and requires about $2^{k/2}$ steps. Table 3 summarizes the theoretic strengths of many ciphers against this attack. It shows that the key sizes of most used blockciphers are too small for many applications.

When multiple encryption schemes are attacked, we can use much smaller numbers of required ciphertexts without increasing the complexity of the attack, by extending this attack with the meet in the middle attack. In such schemes we need only $2^{mk/2}$ steps to find one key, given only $2^{k/2}$ ciphertexts, when m is odd. The tradeoffs between the number of ciphertexts and the complexity, and the average investment for each found key are valid also in this case. Table 4 summarizes the complexities of attacking many multiple encryption schemes.

When modes of operation with random initial values are used, and the keysize is

Key Size	Steps	Ciphers
80	2^{72}	Skipjack
112	2^{88}	3-MAK DES
128	2^{96}	Feal-NX[19], SAFER-SK128[15], IDEA[14]

Table 5. The Complexities of Attacking Modes (such as CBC) with Random Initial Values.

larger than the blocksize, the attack has the results described in Table 5.

The simplest countermeasure against this attack is to reduce the frequency of key replacement; this solution is however unacceptable, since it increases the probability of success of other attacks (such as differential and linear cryptanalysis). Another possible countermeasure avoids encrypting fixed common blocks in messages by using some chaining mode with random initial values. This solution helps, but as we showed, if the initial values are shorter than the keys, still key-collision attacks require less steps than exhaustive search. Thus, to prevent key-collision attacks, we suggest to increase the key size by a factor of two, as was done in hash functions for the hashed results. This solution increases the security of ciphers against exhaustive search as well.

Acknowledgments

It is a pleasure to acknowledge Ross Anderson, Don Coppersmith, Jennifer Seberry, and Adi Shamir for their various remarks and suggestions which helped to improve the exposition and results of this paper. Ross Anderson pointed out the observation at the end of Section 2 that the described attacks are applicable even when modes of operation with random initial values are used. This research was supported by the fund for the promotion of research at the Technion.

References

[1] ANSI draft X9.52-19XX, *Triple DEA Modes of Operation*, Revision 5.1a, March 1996.

[2] Eli Biham, *Cryptanalysis of Multiple Modes of Operation*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of ASIACRYPT'94, pp. 278–292, 1994.

[3] Eli Biham, *Cryptanalysis of Triple Modes of Operation*, in preparation.

[4] Eli Biham, Alex Biryukov, *An Improvement of Davies' Attack on DES*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'94, pp. 461–467, 1994.

[5] Eli Biham, Adi Shamir, *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag, 1993.

[6] Lawrence Brown, Matthew Kwan, Josef Pieprzyk, Jennifer Seberry, *Improving Resistance to Differential Cryptanalysis and the Redesign of LOKI*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of ASIACRYPT'91, pp. 36–50, 1991.

[7] Lawrence Brown, Josef Pieprzyk, Jennifer Seberry, *LOKI - A Cryptographic Primitive for Authentication and Secrecy Applications*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of AUSCRYPT'90, pp. 229–236, 1990.

[8] Don Coppersmith, *Another Birthday Attack*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'85, pp. 14–17, 1985.

[9] Ivan B. Damgaard, Lars Ramkilde Knudsen, *Multiple Encryption with Minimum Key*, proceedings of Cryptography: Policy and Algorithms, Brisbane, Queensland, Australia, Lecture Notes in Computer Science 1029, pp. 156–164, July 1995.

[10] W. Diffie, M. E. Hellman, *Exhaustive Cryptanalysis of the NBS Data Encryption Standard*, Computer, Vol. 10, No. 6, pp. 74–84, June 1977.

[11] Hans Eberle, *A High-speed DES Implementation for Network Applications*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'92, pp. 521–539, 1992.

[12] Marc Girault, Robert Cohen, Mireille Campana, *A Generalized Birthday Attack*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'88, pp. 129–156, 1988.

[13] J. Hastad, *On Using RSA with Low Exponent in a Public Key Network*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'85, pp. 403–408, 1985.

[14] Xuejia Lai, James L. Massey, Sean Murphy, *Markov Ciphers and Differential Cryptanalysis*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'91, pp. 17–38, 1991.

[15] James L. Massey, *Announcement of a Strengthened Key Schedule for the Cipher SAFER*, Revision of 15 September 1995.

[16] Carl H. Meyer, Stephen M. Matyas, *Cryptography: A New Dimension in Computer Data Security*, John Wiley and Sons, 1982.

[17] Mitsuru Matsui, *Linear Cryptanalysis Method for DES Cipher*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'93, pp. 386–397, 1993.

[18] R. C. Merkle, M. E. Hellman, *On the Security of Multiple Encryption*, Communications of the ACM, Vol. 24, No. 7, pp. 465–467, July 1981.

[19] Shoji Miyaguchi, *The FEAL cipher family*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'90, pp. 627–638, 1990.

[20] Shoji Miyaguchi, Akira Shiraishi, Akihiro Shimizu, *Fast Data Encryption Algorithm FEAL-8*, Review of electrical communications laboratories, Vol. 36, No. 4, pp. 433–437, 1988.

[21] National Bureau of Standards, *Data Encryption Standard*, U.S. Department of Commerce, FIPS pub. 46, January 1977.

[22] National Bureau of Standards, *DES Modes of Operation*, U.S. Department of Commerce, FIPS pub. 81, December 1980.

[23] Paul C. van Oorschot, Michael J. Wiener, *A Known Plaintext Attack on Two-Key Triple Encryption*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'90, pp. 318–325, 1990.

[24] Ronald L. Rivest, Adi Shamir, Leonard M. Adleman, *A Method for Obtaining Digital Signatures and Public Key Cryptosystems*, CACM, Vol. 21, No. 2, Feb. 1978.

[25] Bruce Schneier, *Applied Cryptography, Protocols, Algorithms, and Source Code in C*, second edition, John Willey & Sons, 1996.

[26] Walter Tuchman, *Hellman Presents no Shortcut Solutions to the DES*, IEEE Spectrum, Vol. 16, No. 7, pp. 40–41, July 1979.

[27] Michael J. Wiener, *Efficient DES Key Search*, technical report TR-244, School of Computer Science, Carleton University, Ottawa, Canada, May 1994. Presented at the Rump session of CRYPTO'93, August 1993.

[28] Gideon Yuval, *How to Swindle Rabin*, Cryptologia, Vol. 3, No. 3, pp. 187–189, 1979.