Lecture 1

Introduction to Cryptanalysis

This course is devoted to the methods of cryptanalysis. The field of cryptanalysis is fascinating, since it is artistic, and has not and will probably never become a routine. This is due to some fundamental paradoxes that lie in the basis of computation, and mathematics in general. Rephrasing Edgar Alan Poe: Puzzle created by human ingenuity, human ingenuity can solve. Essentially any one who has studied the introductory crypto course or who has read a popular crypto book (like [8, 9, 4]) can design a cipher. The `sci.crypt` newsgroup is a good proof of that, since each week a new cipher appears there (a recent AES contest [1] is an example as well). However it is much easier to build a cryptosystem, than to break one. And it is much easier to find weaknesses in cryptosystems, than to build a really strong one. During the long history of cryptography the best minds were involved in design and analysis of encryption methods[1].

In this lecture we review some fundamentals, that might help in following this course. Modern cryptography can be split into two large areas: private key cryptography and public key cryptography (sometimes called symmetric and asymmetric cryptography). In the private key cryptography, which will be the topic of this course the two interested parties (usually called Alice and Bob) share a secret key, which allows them to communicate securely over the non-trusted communication media. In the public key cryptography (which is a remarkable discovery of the late 70-s) the two parties need not share a common secret. The main idea is to separate the

[1]See an inspiring book by David Khan "The Codebreakers" [3] for a history of cryptographic lore from the ancient to the modern times.

encryption and the decryption processes and to distribute the encryption key publically, but to hold the decryption key private. This allows parties that have never met to communicate securely.

Before we continue a bit of terminology: the encryption function (algorithm) is usually called a *cipher*, its input is called *plaintext* or *clear* text and its output is called *ciphertext*. Ciphers are usually divided into two large classes: *stream ciphers* and *block-ciphers*. Stream ciphers try to imitate the ultimate one-time pad cipher and are actually supposed to be good pseudorandom generators capable of stretching a short secret seed (the secret key) into a long sequence of seemingly random bits. This "key-stream" is then XORed with the plaintext in order to obtain the ciphertext. One of the characteristic features of the stream ciphers is the presence of memory (called *state*), changing during the encryption of the stream. Another issue typical of stream ciphers is — re-synchronization in the case of transmission errors. Stream ciphers are usually very fast.

The second large class of ciphers are the block-ciphers. These will be the core of this course. The main idea behind a block-cipher is to divide the plaintext into blocks of symbols and to use a specially constructed function which mixes the block of plaintext with the secret key to produce the ciphertext. In order to avoid dictionary attacks (see Section 1.3.2 below) the blocks in modern block-ciphers are chosen to be fairly large: 64, 128 and more bits. The idea is to approximate a random permutation on a very large set of inputs (for example 2^{64} inputs). In a case of a random permutation on a set of 2^n inputs any n-bit string can be a result of any n-bit string. One of the methods is to use the product of several mixing transformations — a *product* cipher. Product of several ciphers is also a product cipher, such design is sometimes called a *cascade* cipher. In order to save the chip space in hardware and in order to simplify the design it is popular to build *iterative* block-ciphers which simply iterate the same mixing transformation (called round) sufficiently many times.

Among the main building blocks of modern block-ciphers are *substitutions* and *permutations*, which are primitive ciphers on their own. Substitution ciphers are known from ancient times and can be viewed simply as a change of names of the letters. For example in a cipher attributed to Julius Caesar each letter of the alphabet is exchanged by a letter standing three positions from it (A is encrypted as D, B as E, C as F, etc.). Of course in general the substitution need not have a simple "shift" structure as in Caesar's cipher. However, in-spite of an astronomical number of possible substitution ciphers over the English alphabet (26!), they are easily solvable, using the letter frequency analysis. As a bright illustration of this one can read Edgar Poe's fascinating story "The Golden Bug", or Conan Doyle's "The Dancing Men". A popular element of modern ciphers — a *substitution box* (S-box) takes a block of m bits as its input and outputs a block of n bits (m not necessarily equals n). S-box can perform any function on a set of its inputs; if $m = n$ it can be a permutation on a set of 2^m inputs, if $m > n$ it can be a collection of several permutations on a set of 2^n inputs. It can be a randomly chosen function, or a carefully designed function with special properties. It is desirable for an S-box to perform non-linear and non-affine function in order for the whole cipher to be a non-linear function[2]. Linearity in cipher's behavior is the end of a cipher, since it essentially means that information is leaked from the plaintext to the ciphertext. Both expanding ($m < n$) and contracting ($m > n$) S-boxes can be met in modern block-ciphers. Unless being calculated by a compact formula the memory required to store an S-box grows exponentially with the linear increase in the size of its input m. Thus the most typical sizes for S-box input are $m = 4, 6, 8, 12$ bits.

The second basic element — permutation (or *transposition*) cipher keeps plaintext characters as they are but arranges them in a different order. One of the oldest transposition methods was used by ancient Greeks: A leather belt is tightly wound around a cylinder and a message is written on the belt across the length of the cylinder. The belt is then worn by a messenger. The message can be decrypted by a party who has a cylinder of the same diameter as was used during the "encryption". Another famous transposition cipher, is the *rotating grille* cipher, which can be met in Jules Verne's novel "Mathias Sandorff". In order to encrypt, a square NxN grille with a few openings is placed over a sheet of paper and the message is written in the openings of the grill. The openings are cut in such a way that each time when the grille is rotated by 90 degrees, the openings cover new places in the square. Thus in order to encrypt a message the grill is placed on top of an empty NxN square and the openings are filled with the letters of the message. The grill is then rotated by 90 degrees and the process is repeated three times, till all the NxN square is filled with letters. See Figure 1.1 for an illustration of a 6x6 grille cipher. Breaking a basic permutation cipher is an easy task, especially if one knows a part of the encrypted plaintext. In modern ciphers permutations of bits are frequently used as one of the building blocks. They are usually intermixed with substitutions.

There are various *modes of operation* that a block-cipher can be used in. We describe some of them briefly. Let us denote the encryption of the plaintext P using the encryption function $E(\cdot)$ under a secret key K by $E_K(P)$. The result is denoted by C.

[2]By non-linearity and non-affinity of a function we mean that no output bit can be expressed as a linear or affine function of the input bits.

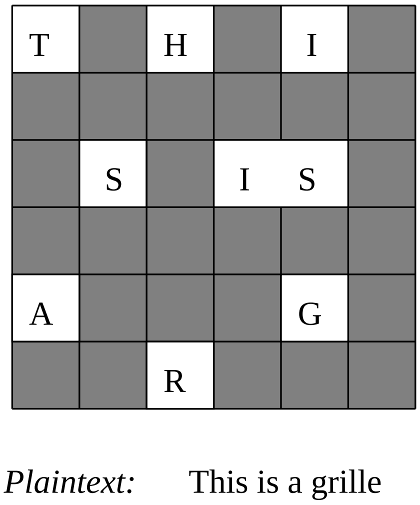

Figure 1.1: An example of a 6x6 grille cipher.

ECB This is the *electronic codebook* mode, which uses the cipher as it is. Given the plaintext block the encryption function is performed once and the result is used as a ciphertext: $C = E_K(P)$. Equal blocks are encrypted to equal blocks.

CBC This is the *cipher-block chaining* mode, which XORes the ciphertext of the previous encryption to the plaintext block before the next encryption is performed. Some *initial vector* (IV) (public or secret) is used as the first such XORed value. Chaining hides redundancy of the plaintext before it is encrypted, and binds separate blocks together. It is one of the most popular modes of operation in modern communications. Chaining goes well with sequential transmission of information, but creates difficulties in direct access or other non-sequential scenarios.

OFB This is the *output-feedback* mode, which produces a stream of re-encrypted blocks, starting with IV: $X_i = E_K(X_{i-1}), X_0 = IV$. The encryption then proceeds in a one-time pad like manner: $C_i = P_i \oplus X_i$.

CFB *Cipher-feedback* mode produces a similar stream as the OFB mode, but involves the previous ciphertext: $X_i = E_K(C_{i-1})$, C_0 is set to some initial value IV. The new ciphertext is obtained by: $C_i = P_i \oplus X_i$.

Cascade Several ciphers are sometimes used in cascade (one after another), in order to increase their strength. It is popular to use so called double- and triple- modes of operation which double- or triple- encrypt the plaintext under different keys. Notice that any product cipher, can be viewed as a long cascade of very simple ciphers.

Chaining modes CBC, OFB and CFB can be viewed as attempts to copy "stream cipher" behavior for encryption of messages that occupy several blocks. Various chaining modes have different error propagation properties in the case of transmission errors.

Any cipher is used in some setting or scenario, called *cryptographic protocol*. Among the various scenarios are: identification schemes (for ex. computer login protocol), digital signatures, authentication codes (for ex. they help to ensure integrity of the data flowing through the network), distribution and sharing of secrets and numerous other applications. A flaw in the logic of a protocol can lead to a break of a cryptosystem even without breaking the underlying cipher. Analysis and design of secure protocols is a very important issue in itself, however it is out of the scope of this course.

1.1 Classification of Attack Scenarios

We list here various cryptanalytic attack scenarios from those that require minimal assumptions on the power and knowledge of the attacker to the most unrealistic, hypothetic attacks. Note that a long tradition in cryptanalytic research is that the attacker has full knowledge of the encryption algorithm, and only the key of the cryptosystem is unknown. This assumption is called a *Kerckhoff's* principle, named after XIX-century German cryptanalyst who declared it first. There are many cryptosystems that are not broken under this condition, so why use one that is claimed to be secure, but is kept secret. This gives a false sense of security and in many cases tries to hide the lack of designer's expertise. Not once in the history of cryptology top secret designs were revealed[3].

The aim of the attacker is to read the encrypted messages, which in many cases is achieved by finding the secret key of the system. The efficiency of the attack is measured by the amount of plaintext-ciphertext pairs required, time spent for their analysis and the success probability of the attack. Usually the starting point of a cryptanalytic attack is the ability, to distinguish the output of a cipher from the output of a random permutation. This is the minimal goal for an attacker.

[3]One of the most spectacular stories is the cryptanalysis of the German Enigma cipher by Polish and British mathematicians during the World War II. Among the more recent stories is the reverse engineering of the RC4 cipher designed by Rivest for RSA [8].

Below we provide a **taxonomy of cryptanalytic attacks** based on various attack scenarios. The attacks are listed from the most practical to the hypothetical ones:

1.1.1 Ciphertext-Only

This is the most powerful cryptanalytic attack, since it requires only *passive* eavesdropping from the attacker in order to obtain the ciphertext. The knowledge of the plaintext is minimal and consists of some information about the distribution of the plaintexts. For example, an attacker may know that the encrypted plaintext is in English. Ciphers that succumb to this attack are useful examples of how not to build ciphers and as puzzles for cryptography students. Many of the classical ciphers (those that were used through the 5000 year old history of cryptography) fall into this category. However, solving these ciphers with a minimal amount of ciphertext and without a computer can be a non-trivial and exciting task. A good exercise for cryptanalysis students.

1.1.2 Known Plaintext

This scenario assumes that the attacker knows a portion of the encrypted text. The aim is either to derive from this known portion the secret key or at least to be able to read some unknown portion of the encrypted text. This scenario is still highly realistic, since it is hard to prevent the attacker from guessing part of the plaintext (something that in the good old days was called a "probable word method"), or from knowing the context of the communication.

1.1.3 Chosen Plaintext

In this case one assumes that the attacker has the ability to encrypt a text of his choice. In practice this can be achieved in the case when an encryption box with a securely embedded unknown secret key falls in the hands of the attacker or when it is possible to send chosen plaintext to the owner of the secret key and then eavesdrop to the transmission of this text in encrypted form to a third party. This scenario is less common since it requires *active* action of the attacker.

1.1.4 Chosen Ciphertext

This is similar to the previous case, but requires an ability to choose ciphertexts for a decryption device.

1.1.5 Adaptive Chosen Plaintext or Ciphertext

These chosen text attacks involve even more active action of the attacker, since they assume the ability of adaptive choice of the encrypted text based on the already received results of encryption (or decryption). In most cases this is unrealistic but is of high theoretic interest. If the attacker obtains the device with the unknown secret key this approach may be possible. From the "Learning Theory" point of view, cryptanalysis of a cipher can be seen as a problem of learning an unknown parameter k of a function $f(x, k)$. The aim of the attacker is thus to learn k using a minimal amount of queries to the function (adaptive queries are possible). Good ciphers are functions which can not be learned efficiently.

1.1.6 Related Key

This scenario assumes the knowledge (or the choice) of relations between keys in two different encryptions in addition to one of the scenarios described above. This attack may discover interesting theoretical weaknesses in the key scheduling algorithm of a cipher.

1.1.7 Partial Knowledge of the Key

This is a scenario, in which the attacker possesses partial knowledge of the secret key (for example due to a flaw in the randomization procedure, which generates encryption keys). In a good cipher, partial knowledge of the key should not make finding the rest of it any easier. Otherwise the exhaustive search on such cipher will be made much easier.

1.2 The Goals of Cryptanalytic Attacks

Below we provide a classification of attacks by their effect on the cryptosystem:

1.2.1 Distinguishing Attacks

These are the most basic cryptanalytic attacks. Distinguishing attack is successful if it is possible to distinguish an output of a cipher from an output of a random permutation. These are closely related to "cipher detection" attacks, which can tell that a certain encryption algorithm is used in the communication. In many cases distinguishing attacks open doors for more soffisticated key-recovery attacks (see below).

1.2.2 Partial Knowledge of the Plaintext

In this attack partial knowledge of the plaintext (any predicate of the cipher's input) is gained. As an example consider a cryptosystem, which preserves the parity of the plaintext in the ciphertext.

1.2.3 Decryption

In this case the attacker is capable of decrypting part of the encrypted traffic. Note that this ability may be reached without obtaining the secret key of the system. As an example consider any encryption system, which is an involution (self-inverse function). Decryption in such system can be achieved by re-encryption of the ciphertext (this is possible in adaptive chosen plaintext attack scenario).

1.2.4 Encryption (Forgery)

In this case the attacker is able to encrypt meaningful messages under unknown secret key. This does not necessary means that the attacker possesses the knowledge of the secret key. Cipher vulnerable to this attack is not suitable for authentication/identification purposes.

1.2.5 Partial Key Recovery

In this attack some predicate of the secret key is gained by the attacker. While the remaining part of the key, may still be too large to allow for an attack, this property is highly undesirable in ciphers. This is usually just a first step for a total key recovery attack.

1.2.6 Total Key Recovery

This is the most fatal cryptanalytic attack for a cryptosystem. With the knowledge of the secret key the attacker can perform all the disastrous effects described above.

1.3 Methods of Cryptanalysis

What does it mean to break a cipher?

Depending on the protocol a cipher is used in, threat model and computational power of the attacker it can mean different things. It can mean investing a quarter of a million dollars in hardware and breaking DES with exhaustive search [2] in one day. It can mean doing pen and paper analysis of letter frequencies of a simple substitution. It can mean writing a program to search for statistic patterns in a cipher. The cipher can be considered broken if its output can be distinguished from a random permutation under a chosen plaintext/ciphertext attack. It can be considered broken if the secret key is found in a ciphertext-only attack. It can be considered broken if it is possible to derive secret elements of a cipher under a chosen key attack. In short, a cipher is broken, if a person who uses it decides to stop doing so because he/she does not trust its security anymore.

In the world we are living in, exhaustive search attacks are obviously the most straightforward methods of cryptanalysis, they give us insight on how fast modern computers are, on how large the Internet is and how many people are willing to donate free cycles of their home computers to experimental distributed computation projects. However one can disregard the exhaustive search threat when dealing with a serious 128-bit key cipher. In general, people expect that a good cipher is a one for which the best attack is an exhaustive search for the key. Therefore, if one can save, say, a 2^{28} factor in the search – which still leaves the complexity at an infeasible level of 2^{100} – it indicates the existence of defects in the design that the designers were not taking into account. This, although not a "full break", serves as a first warning with respect to the security of the cipher. Pen and paper are still very powerful weapons in the hands of cryptanalyst as long as a cipher has a simple and clean structure. In many cases however paper analysis needs feedback from computer analysis of a cipher: search for special statistical properties and irregularities. From a researcher point of view a cipher is considered broken if a weakness in it is found which requires the changes of the design. The change of design may vary from adding more rounds to a cipher and to the change of key-schedule and internal tables and wires. The cipher is totally broken if it is easier to build a new cipher from the scratch than to repair the broken one, or if the mere foundation of its design were found to be flawed.

When a Cipher is Secure?

A short answer is: never. For a more serious view one can consider two models of the power of the attacker: information-theoretic and computational. In a information-theoretic setting the computational power of the adversary is unbounded. That means that the attacker can for example exhaustively check all the secret keys of the system. Under this assumption only one cryptosystem is perfectly secure: the so called One-Time Pad (OTP). This cipher picks a stream of random bits as long as the plaintext and XORs it to the message. The same stream is never used for any other message. Impracticality of such approach is obvious: before starting secure communication with the OTP method one has to pass to the other side securely the length of the key equal to the future length of communication. OTP approach is used in the top secret communications (for ex. between presidents of different countries).

The second approach is more reasonable, since there is no such thing as all powerful attacker. One can estimate the computational power of the adversary and use a cipher that is, say 2^{30} harder to break than the estimate. Such security margin is very important since one has to anticipate the yet undiscovered attacks as well as the growing computational power. To our knowledge not a single modern cipher has a proof that it is computationally secure up to a certain bound, since finding the lower bounds on complexity of algorithms in general, is a very hard task. On the other hand, given a cryptanalytic attack on a cipher we obtain an upper bound on complexity of breaking it. One of the popular approaches is to build a cipher that is provably secure against certain very powerful methods of attack (these methods will be shown later in this course). This is the best that can be done at our present state of knowledge and of course it does not prevent a discovery of a new method of cryptanalysis which will break the "provably" secure ciphers. It is very hard to fight the unknown.

Here we present a list of typical cryptanalytic methods:

1.3.1 Exhaustive Search

This is the most obvious and straightforward method of attacking ciphers. It checks all the possible secret keys against a short known plaintext/ciphertext sample (or ciphertext-only sample if the plaintext redundancy is high). The correct key will produce the correct ciphertext from a known plaintext. The key-size of modern ciphers is picked large enough in order to make this method of attack impossible (128 bits or more). One of the major weaknesses of the "Data Encryption Standard" cipher which will be described later in this course, was its short key size (56 bits) which allows an exhaustive search attack. Simple as it is, an element of **partial guessing** is present in almost any cryptanalytic attack.

Just to give a feeling of it let us quote some numbers: 2^{30} simple operations are easily performed on a modern personal computer in several seconds. It takes several thousand computers on the net 10 days to perform 2^{55} DES encryptions. Thus the power of all computers on the Internet can be roughly estimated as some 2^{70} simple operations per day. In order to compare these numbers with our everyday experiences note that[4]: the

[4]Sited from [8].

odds of being killed by the lightning (per day) are 1 in 2^{33}, the odds of being killed in an automobile accident (per year, US) are 1 in 2^{13}. The odds of winning top prize in US state lottery are 1 in 2^{28}. The time till the next ice age is 2^{14} years. The time till sun goes nova is 2^{30} years. The number of atoms on Earth is about 2^{170}. The number of atoms in the galaxy is 2^{223}.

1.3.2 Dictionary Attacks

This is another simple but important type of attack on block-ciphers. If the size of the block in a cipher is small the attacker can collect many blocks and analyze frequencies of different blocks. An extreme case of a cipher vulnerable to such attack is a simple substitution cipher, where each letter is exchanged for another letter of the alphabet.

Another type of dictionary attack is to store encryptions of a single fixed block under all possible keys in a sorted dictionary and to wait till this block is transmitted encrypted with the secret key. This secret key is then found by a single look up in a dictionary. Note that storage devices with 2^{36}-byte capacity are reality (1999).

1.3.3 Equivalent Description

In some cases designer of the system (in the quest to make things very complicated) overlooks simple equivalent descriptions of his cryptosystem (or of some parts of it). For example applying a sequence of bit-permutations one by one to a block of plaintext is equivalent to applying a single bit-permutation which is a product of all of them. If the permutations are unknown to the attacker, the fact that there are l of them does not make life of the attacker harder. He just needs to discover the product permutation. This is for example the case, if at some point in the cryptosystem there is an operation: $P \oplus K_1 \oplus K_2 \oplus \ldots \oplus K_l$ which is just equivalent to $P \oplus K'$ for $K' = K_1 \oplus K_2 \oplus \ldots \oplus K_l$. Similar situation can happen in a product of several substitutions. In many cases cryptanalytic attack is preceded by a discovery of a simplifying equivalent description.

1.3.4 Periodicity or Search for Invariants

Invariants, i.e features which are preserved during the encryption (for example if encryption preserves the parity of the plaintext) are an undesirable property of a cipher. If cryptanalyst manages to discover an invariant of a cipher, or even an approximate invariant, a distinguishing attack can be mounted. Any type of periodic behavior, or correlation between sets of encryptions should be avoided. Moreover, any special properties of the internal building blocks of the system (like cycles in permutations, high/low probability relations) can be very useful to the attacker.

1.3.5 Birthday Paradox

This is a very important probabilistic paradox with uncountable applications in modern cryptology: from block-ciphers to public key cryptography.

Consider a set X of cardinality $|X| = 2^n$ and a random function $F(x)$ defined on the elements of this set. Define a notion of *collision* of the function F: A pair $(x, y), x \in X, y \in X$ is called a collision of function F if $F(x) = F(y)$. The question is: how large should be two subsets A and B of X so that *collision* between them happens with probability more than half? The probability of no collision for sets A, B can be written as:

$$\left(1 - \frac{|A|}{|X|}\right)^{|B|},$$

which in case $|A| << |X|$, is approximately:

$$e^{-\frac{|A|\cdot|B|}{|X|}}.$$

The required condition is satisfied, when $|A|\cdot|B| \approx |X|$, and thus in the case of equal cardinalities $|A| = |B|$ we receive a surprising answer: $|A| \approx 2^{n/2}$. Similar result can be received when there is one set: $A = B$.

How this paradox applies to cryptography? Suppose that for a known-plaintext attack on a 64-bit block cipher the adversary needs to obtain two plaintext-ciphertext pairs which differ by a certain predefined constant. Due to a birthday paradox-like argument a pool of only about 2^{32} known plaintexts with high probability will contain the required pair. We will describe applications of this paradox in greater detail later in this course.

1.3.6 Meet-in-the-Middle Attack

This method of cryptanalysis divides the encryption algorithm into upper and lower parts. Then by means of partial guessing it tries to perform partial encryption from top to the middle and partial decryption from bottom to the middle. The results are then compared and if they agree the key candidate is saved for further analysis, otherwise the key guess is wrong. The most notable applications of this method are in the analysis of cascade ciphers [5, 6, 7].

1.3.7 Statistic Approaches

These methods search for statistic patterns, that are leaked to the ciphertext from the plaintext. If such patterns can be observed a cipher can be distinguished from a random permutation. Such distinguishing attack is the first step for attack on any cipher. Note that statistic approaches can concentrate both on finding *high frequency* (probability) events, as well as *low frequency* or even *impossible events*. It is clear that any bias from the random behavior is an undesirable property for a cipher.

1.3.8 Weak Keys with Respect to a Particular Attack

In some cases it is possible to point a class of keys that facilitate the analysis of the cipher with a specific attack method. A subset W of keys is called *weak*, if testing whether the current key is from W (membership test, which is an attack under the assumption that the key is weak) can be done in much less than $|W|$ steps. Such keys should be avoided. If a cryptosystem possesses a high ratio of weak keys it is a serious drawback which requires a redesign of the cryptosystem. Consider, for example, a cipher in which every key in a thousand is weak. Suppose that this cipher is used in a setting which requires the change of the key every 10 seconds[5]. If the keys are chosen at random an attacker will have to listen to the communication for an hour and a half on the average till he can decrypt a portion of the traffic.

[5]Note that modern networks have speeds of 10 to 100 megabits per second and thus up to 1 Gbit of data can be transmitted in that time.

Bibliography

[1] Advanced Encryption Standard Development Effort,
http://csrc.nist.gov/encryption/aes/aes_home.htm.

[2] Electronic Frontier Foundation,
http://www.eff.org/pub/Privacy/Crypto_misc/DESCracker/
http://www.cryptography.com/des/index.html

[3] D. Khan, *The Codebreakers, The Story of Secret Writing*, Macmillan, 1967.

[4] A. Menezes, P.C. Van Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[5] R. Merkle, M. Hellman, *On the Security of Multiple Encryption*, Communications of the ACM, Vol.24, No.7, pp.465-467, 1981.

[6] P. C. van Oorschot, M. J. Wiener, *A Known-Plaintext Attack on Two-Key Triple Encryption*, proceedings of EUROCRYPT'90, LNCS 473, pp.318–325, 1990.

[7] P. C. van Oorschot, M. J. Wiener, *Improving Implementable Meet-in-the-Middle Attacks by Orders of Magnitude*, proceedings of CRYPTO'96, pp.229–236.

[8] B. Schneier, *Applied Cryptography Second Edition*, John Wiley & Sons, New York, NY, 1996.

[9] D. R. Stinson, *Cryptography: theory and practice*, CRC Press, 1995.