Development of Microprocessor Based Encoders for Secured Transmission

1. Introduction

In this age of universal electronic connectivity, of viruses and hackers, of electronic eavesdropping and electronic fraud, there is indeed no time at which security does not matter. Two trends have come together to make the topic of vital interest. First, the explosive growth in computer systems and their interconnections via networks has increased the dependence of both organizations and individuals on the information stored and communicated using these systems. This in turn, has led to a heightened awareness of the need to protect data and resources from disclosure, to guarantee the authenticity of data and messages. Second, the discipline of cryptography has matured leading to the development of practical, readily available applications to enforce security.

Here, a practical survey of principles and history of cryptography is presented. The literature of cryptography has a curious history. Secrecy, of course, has always played a central role, but until the First World War, important developments appeared in print in a more or less timely fashion and the field moved forward in much the same way other specialized disciplines. In 1920, one of the most influential cryptanalytic papers of twentieth century, William F. Friedman's monograph "The index of Coincidence and its applications in cryptography", appeared as a first research report of the private Riverbank Laboratories [1]. And this, despite the fact that works had been done as part of the war effort. In the same year Edward H. Hebern of Oakland, California filed the first patent for a rotor machine [2], the device destined to be a mainstay of military cryptography for nearly 50 years.

After the First World War, however, things began to change. U.S. army and Navy organizations working entirely in secret began to make fundamental advances in cryptography. During the thirties and forties, a few basic papers did appear in the open literature and several treatises on the subject were published, but the later were

farther and farther behind the state of art. By the end of war the transition was complete. With one notable exception, the public literature had died. The exception was Claude Shanon's paper, "The Communication Theory of Secrecy Systems", which appeared in the Bell System Technical Journal in 1949 [3]. It was similar to Friedman's paper, in that it grew out of wartime work of Shanon's.

From 1949 to 1967 the cryptographic literatures was barren. In that year a different sort of contribution appeared: David Kahn's history, The Codebreakers [4]. It did not contain any new technical ideas, but it did contain a remarkably complete history of what had gone before, including mention of some things that the Govt. of USA still considered secret. The significance of the Codebreakers lay not just its remarkable scope, but also in the fact that it enjoyed good sales and made tens of thousands of people, who had never given the matter a moment's thought, aware of cryptography. A trickle of new cryptographic began to written.

At about the same time, Horst Fiestel, who had earlier worked on identification of friend or foe devices for the Air Force, took his life long passion for cryptography to the IBM Watson Laboratory in Yorktown Heights, New York. There, he began development of what was to become the US Data Encryption Standard; by early 1970s several technical reports on this subject by Fiestel and his colleagues had been made public by IBM [5, 6, 7].

This was the situation when W. Diffie entered the field in late 1972. The cryptographic literature was not abundant, but there was included some very shiny nuggets.

Cryptography presents a difficulty not found in normal academic disciplines: the need for the proper interaction of Cryptography and Cryptanalysis. This arises out of the fact that in absence of real communications requirements, it is easy to propose a system that appears unbreakable. Many academic designs are complex that the would be cryptanalyst does not know where to start; exposing flaws in these designs is far harder than designing them in the first place. The result is that the competitive process, which is one strong motivations in academic research, can not take hold. In 1975, Hellman and W. Diffie proposed public Cryptography [8], one of the indirect aspects of their contribution was to introduce a problem that does not

even appear easy to solve. This enthused the number of people in Cryptography, the number of meetings held, and number of books and papers published.

W. Diffie told the following statements to the audience in his acceptance speech for the Donald E. Fink award – given for the best expository paper [9], "Privacy and Authentication: An Introduction to Cryptography", to appear in an IEEE Journal – which he received jointly with Hellman in 1980.

"I had an experience that I suspected was rare even among the prominent scholars who populate the IEEE award Ceremony: I had written the paper I had wanted to study, but could not find, when I first became seriously interested in Cryptography. Had I been able to go to the Stanford Bookstore and picked up a modern cryptography text, I would probably have learned about the fields years earlier. But only things available in the fall of 1972 were a few classic papers and some obscure technical reports."

The contemporary researcher has no such problem. The problem now is choosing where to start among the thousands of papers and dozens of books. It has been necessary to spend long hours hunting out and then studying the research literature before being able to design the sort of cryptographic utilities glibly described in popular articles. This gap has been filled by Bruce Schneier's "Applied Cryptography" [10]. Bruce Schneier has given a panoramic view of the fruits of 20 years (1976 – 1996) of public research. He has included an account of the world in which Cryptography is developed and applied, and discusses entities ranging from the International Association for Cryptologic Research (IACR) to the National Security Agency (NSA).

In early 80s NSA made several attempts to control Cryptography and issued a letter to the IEEE mentioning that the publication of Cryptographic material is a violation of the International Traffic Arms Regulations (ITAR). These viewpoint turned out not even to be supported by the regulation themselves – which contained an explicit exemption for published material.

Some essential terms and basic background of Cryptography are intended to present in section 1.1 to 1.8. Then the objective of the work will be presented.

1.1 Sender and Receiver:

A sender wants to send a message securely to a receiver so that eavesdropper can not read the message.

1.2 Messages, Encryption, Decryption and Related Terms:

A **message** is plaintext. The process of disguising a message in such a way as to hide its substance is **Encryption**. An encrypted message is **ciphertext**. The process of turning the ciphertext back into plaintext is **Decryption**. The art and science of keeping messages secure is **Cryptography**, and it is practiced by **Cryptographer**. The art and science of breaking ciphertext is **Cryptanalysis** and the practitioner is called **Cryptanalyst**. The branch of mathematics encompassing both Cryptography and Cryptanalysis is **Cryptology** and its practitioner is **Cryptologist**.

The encryption function (E) operated on a message (M) produces a ciphertext (C). Mathematically, C = E(M). In reverse process, the decryption function (D) operates on C to produce message (M). M=D(C). The whole process can be described as M = D(E(M)). The following security services are important requirements for cryptography. The security services are important requirements for cryptography. These are given in section 1.2.1 to 1.2.4.

1.2.1 Confidentiality: It is the protection of transmitted data from passive attacks.

1.2.2 Authentication: It is concerned with assuring that a communication is authentic. In the case of single message, such as a warning or alarm signal, the function of the authentication service is to assure the recipient that the message is from the source that it claims to be from.

1.2.3 Integrity: As with confidentiality, integrity can apply to a stream of messages, a single message or a selected field within a message. The most useful and straightforward approach is total stream protection. A connection oriented integrity service, one that deals with a stream of messages, assures that messages are received as sent, with no duplication, insertion, modification, reordering, replays or destruction of data.

1.2.4 Nonrepudiation: Nonrepudiation prevents either sender or receiver from denying a transmitted message.

1.3 Algorithms and Keys: A cryptographic algorithm is the mathematical function used for encryption and decryption. If the security of an algorithm is based on

keeping the way that algorithm works a secret. It is restricted algorithm. This type of algorithm, though popular, has become obsolete. Because someone of a group accidentally reveals the secret, everyone has to change their algorithm. Every time a user leaves the group, every one else has to switch to a different algorithm. The modern cryptography solves this problem with the introduction of a key, K. This key might be any one of large number of values. The range of possible values of key is called keyspace. Both encryption and decryption use this key. So the functions may written as

$$E_k(M) = C; \quad D_k(C) = M \quad \text{and} \quad D_k(E_k(M)) = M$$

The encryption function, E operated on message, M with key, K generates ciphertext, C; the decryption function, D operates on C with key, K generates the message; therefore the decryption function operates on encrypted message generates the message. Some algorithms use different keys, one (K_1) for encryption and other (K_2) for decryption. Then

$$E_{k1}(M) = C; \quad D_{k2}(C) = M \quad \text{and} \quad D_{k2}(E_{k1}(M)) = M$$

All of these security in these algorithms is based on the key (or keys). This can be shown in the following figure in.1.

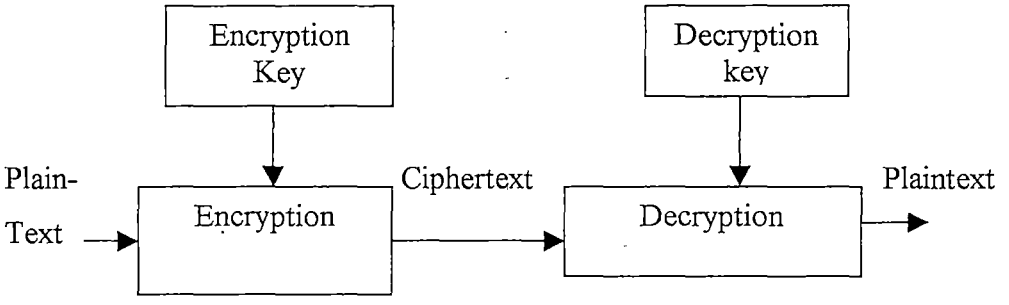

Fig 1.1: Encryption and decryption with keys

A **cryptosystem** is an algorithm, plus all possible plaintexts, ciphertexts and keys. According to the type of key used algorithms may be classified as

1. Symmetric algorithm

and 2. Public-key algorithm

1.3.1 Symmetric Algorithm: Symmetric Algorithms, also called conventional algorithms, are algorithms where the encryption key can be calculated from the

decryption key and vice-versa. In most symmetric algorithms, the encryption and decryption keys are the same. These algorithms, also called secret key algorithms, single key algorithms or one-key algorithms, requires that the sender and receiver agree on a key before they can communicate securely. The security rests on the key. As long as communication needs to remain secret, the key must remain secret. Symmetric algorithms can be divided into two categories. Some operate on plaintext a single bit at a time. These are called **Stream algorithms or stream ciphers.** Others operate on plaintext in groups of bits. The groups of bits are called blocks and algorithms are called **Block algorithms or Block ciphers.**

1.3.2 Public-key Algorithms: Public-key algorithms, also called Asymmetric algorithms, are designed so that the key used for encryption is different from the key used for decryption. The decryption key can not be calculated from the encryption key. The algorithms are called public-key algorithms, because the encryption key can be made public. A stranger can use the encryption key to encrypt a message, but only a specific person with corresponding decryption key can decrypt the message. The encryption is often called **public key** and the decryption key is often called the **private key (secret key)**. Sometimes messages are encrypted with the private key and decrypted with the public key. This used in digital signatures.

1.4 Security of Algorithms: Different algorithms offer different degrees of security. It depends on how hard they are to break. If the cost required to break an algorithm is greater than the value of encrypted data, then we are probably safe. If the time required to break the algorithm is longer than the time, the encrypted data must remain secret, then we are probably safe. If the amount of data encrypted with a single key is less than the amount of data necessary to break algorithm, then we are probably safe.

Lars Knudsen classified these different categories of breaking an algorithm in decreasing order of severity [11].

1. **Total Break**: A cryptanalyst finds the key, K, such that D_k (C) = M.
2. **Global Break**: A cryptanalyst finds an alternate algorithm, A equivalent to D_k(C), without knowing K.
3. **Local Deduction**: A Cryptanalyst finds the plaintext of an intercepted ciphertext.

4. **Information Deduction**: A Cryptanalyst gains some information about the key or plaintext. This information would be a few bits of the key, some information about the form of plaintext, and so forth.

An algorithm is unconditionally secure, if no matter how much ciphertext a cryptanalyst has, there is not enough information to recover the plaintext. In point of fact, only a **one-time pad** is unbreakable given infinite resources. All other cryptosystems are breakable in a ciphertext-only attack, simply by trying every possible key one by one and checking whether resulting plaintext is meaningful. This is called **Brute-force** attack.

Cryptography is more concerned with cryptosystems that are computationally infeasible to break. An algorithm is considered computationally secure, if it can not be broken with available resources, either current or future.

We can measure the complexity of an attack in different ways:

1. **Data Complexity**: The amount of data needed as input to the attack.
2. **Processing Complexity**: The time needed to perform the attack. This often called work-factor.
3. **Storage Requirement**: The amount of memory needed to do the attack.

Complexities are expressed as orders of magnitude. If an algorithm has a processing complexity of 2^{128}, then 2^{128} operations are required to break the algorithm. Still if we assume that we have enough computing speed to perform a million operations every second and we set a million parallel processors against task, it will still take over 10^{19} years to recover the key. That's a billion times the age of the Universe.

Historically, a code refers to a cryptosystem that deals with linguistic units: words, phrases, sentences and so forth. Codes are only useful for specialized circumstances. Ciphers are useful for any circumstance.

Steganography serves to hide secret messages in other languages, such that the secret's very existence is concealed. Generally the sender writes an innocuous message and then conceals a secret message on the same piece of paper. Historical tricks include invisible inks, tiny pin punctures on selected characters, minute differences between handwritten characters, pencil marks on the typewriter characters and so on. More recently, people are hiding secret messages in graphic

images. The least significant bit of each byte of the image is replaced with the bits of message. The graphical image won't change appreciably – most graphic standards specify more gradations of color than the human eye can notice and the messages can be stripped out at the receiving end. We can store a 64 kbyte message in 1024x 1024 grey-scale picture this way. Peter Wayner [12, 13] proposes hiding a message by using the least significant bits (lsb) of frames on a CD. The Kodak photo CD format's maximum resolution is 2048x3072 pixels, with each pixel containing 24 bits of RGB color information. The lsb of each 24 bit pixel can be changed without greatly affecting the quality of the image. The result is that we can hide 2.3 megabyte message in a single digital snapshot. There are number of software packages available and John in 1997 has developed one such package.

Now I examine the two basic building blocks of all encryption techniques or ciphers. These are basically

1. **Substitution Technique** and
2. **Transposition Technique**

A **substitution technique** is one in which each character is the plaintext is substituted for another character in the ciphertext. The receiver inverts the substitution on the ciphertext to recover the plaintext. There are 4 types of substitution ciphers:

a) A simple substitution cipher or monoalphabetic cipher: It is one in which each character of the plaintext is replaced with a corresponding character of ciphertext. The cryptograms in newspapers are of this kind.

b) Homophonic substitution cipher: It provides multiple substitutions, known as Homophones, for a single letter. For example, 'A' could correspond to either 5, 13, 25 or 56 , 'B' could correspond to either 7, 19, 31 or 42 and so on.

c) Polygram Substitution cipher: It is one in which blocks of characters are encrypted in groups. For example 'ABA' could correspond to 'RTQ'; 'ABB' could correspond to 'SLL' and so on.

d) Polyalphabetic Substitution Cipher: It is to use different monoalphabetic substitutions as one proceeds through the plaintext message. This cipher has the following features.

i) A set of related monoalphabetic substitution rules is used.

ii) A key determines which particular rule is chosen for a given transformation.

The best known and one of the simplest, such algorithm is referred to as the Vigenere cipher.

A famous Caesar cipher, in which a plaintext character is replaced by the character three to the right modulo 26, is a simple substitution cipher.

ROT13 is a simple encryption program commonly found on Unix system. Encrypting a file twice with ROT13 restores the original file.

P=ROT13(ROT13(P)). It is not intended for security.

Simple substitution ciphers can be easily broken because the cipher does not hide the underlying frequencies of different letters of plain text. Algorithms for solving these sorts of ciphers can be found in [14, 15, 16, 17, 18, 19, 20]. A good computer algorithm is [21].

Homophonic substitution ciphers were used as early as 1401 by Duchy of Mantua [22]. They are much more complicated to break than simple substitution ciphers. With a plaintext the ciphers are trivial to break. A ciphertext only attack is harder, but only takes a few seconds on a computer. Details are given in [23].

Polygram substitution ciphers are ciphers in which groups of letters are encrypted together. Playfair cipher was used by British during World War –I. It encrypts a pairs of letters together. The Hill cipher is another example of a polygram substitution cipher.

Polyalphabetic substitution ciphers were invented by Leon Battista in 1568 [22]. They were used by Union Army during American Civil War. The Vigenere cipher and Beaufort cipher are also examples of polyalphabetic substitution ciphers.

A Running Key Cipher – sometimes called a book cipher- in which one text is used to encrypt another text, is another example of this sort of cipher. This cipher can easily be broken, even though this cipher has a period the length of the text [22,24].

1.6 Transposition Ciphers: In a transposition cipher the plain text remains the same, but the order of the character is shuffled around. In a simple columnar transposition cipher, the plaintext is written horizontally into a piece of graph paper

of fixed width and the ciphertext is read off vertically. Decryption is a matter of writing the ciphertext vertically into a piece of graph paper of identical width and then reading the plaintext off horizontally. Cryptanalysis of these ciphers is discussed in [15,18]. Since the letters of ciphertext are the same as those of the plaintext, a frequency analysis on ciphertext would reveal that each letter has approximately the likelihood as in English. This is an idea to cryptanalysts. Putting ciphertext through a second transposition cipher greatly enhances security. These are even more complicated transposition ciphers, but computer can break them. The German ADFGVX cipher, used during World War –I, is a transposition cipher combined with a simple substitution. It was a complex algorithm, but was broken by George Painvin, a French Cryptanalyst [22].

Many modern algorithms use transposition; it requires a lot of memory and sometimes requires messages to be only certain lengths.

1.7 Rotor Machines: In the 1920s, various mechanical encryption devices were invented to automate the process of encryption. A rotor machine has a keyboard and series of rotors and implements a version of the Vigenere cipher. Each rotor is an arbitrary permutation of the alphabet, has 26 positions and performs a simple substitution. The output pins of one rotor are connected to the input pins of the next. It is the combination of several rotors and the gears moving them that make the machine secure. Because the rotors all move at different rates, the period for n-rotor machine is 26^n. It gives a frustrating picture to the cryptanalysts. The best known rotor device is the Enigma. The Enigma was used by the Germans during World War –II. The idea was invented by Arthur Scherbius [25] and Arvid Gerherd Damm in Europe. A team of Polish cryptographers broke the German Enigma and explained their attack to the British. The German modified the Enigma as the war progressed, and the British continued to cryptanalyze the new version. How rotor cipher works and how they were broken are available in [22, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35].

1.8 One-Time Pad: One-Time Pad is a perfect encryption scheme and was invented in 1917 by Major Joseph Mauborgne and AT & T's Gilbert Vernam [22]. One-Time Pad is nothing more than a large non-repeating set of truly random key letters, written on sheets of paper, and glued together in a pad. In its original form, it was a one-time tape for Teletypewriters. The sender uses each key letter on the pad to

encrypt exactly one plaintext character. Encryption is the addition modulo-26 of the plaintext character and the One-Time Pad key character. Each key letter is used exactly once, for only one message. The sender encrypts the message and then destroys the uses pages of the pad or used section of the tape. The receiver has an identical pad and uses each key on the pad, in turn, to decrypt each letter of the ciphertexts. The receiver destroys the same pad pages or tape section after decrypting the message.

Assuming an eavesdropper can't get access to the One-Time Pad used to encrypt the message, this scheme is perfectly secure.

A random key sequence added to a non-random plaintext message produces a completely random ciphertext message and no amount of computing power can change that.

The idea of a One-Time Pad can easily be extended to binary data. Instead of a One-Time Pad consisting of letters, a One-Time Pad of bits is used. Instead of adding the plaintext to the One-Time Pad an XOR is used. The ciphertext is XORed with the same One-Time Pad to decrypt the message. Everything else remains the same and the security is just as perfect. But One-Time Pad does not provide authenticity.

The available modern encryption algorithms are as follows.

1. Data Encryption Standard (DES)
2. Blowfish
3. RC5
4. CAST-128 (Carlisle Adams and Stafford Tavares)
5. RSA (Rivest Shamir Adelman)
6. MD5 (Message Digest)
7. RIPE MD -160 (RACE Integrated Project Evaluation Message Digest)
8. HMAC (Hash Message Authentication Code)
9. AES (Advanced Encryption Standard)
10. Serpent
11. IDEA
12. GOST

13. SEAL

Under the scenario, the work on message security based conventional method (i.e. Symmetric Block cipher) about three years back encouraged me and was started the work with a view to develop the encoding techniques to realize in hardware and can be connected to the computer as peripheral device. As a result 6 block cipher techniques are developed and tested and verified the techniques through assembly level code and data. Here encoding and decoding are used in place of encryption and decryption respectively and vice-versa. The encoders developed are given in the following.

1. Prime Position Encoder (PPE)
2. Recursive Pair Parity Encoder (RPPE)
3. Triangular Encoder (TE)
4. Modified Rotational Encoder (MRE)
5. Modified Johnson Encoder (MJE)
6. Bit Swapping Encoder (BSE)

Out of these six encoders, RPPE and TE are substitution type and the rests are transposition type. All the encoders up to 256 bit block length have been tested in steps (16, 32, 64, 128, 256). The encoding (encryption) as well as decoding (decryption) technique for each case has been presented.

The intension of designing of microprocessor based system [123, 124, 125, 126] may lead to realize the tested techniques in microcontroller based system or in VLSI design of the same techniques in order to implement hardware-wise.

The tested techniques are analyzed using two methods. One is to find out the distribution of the characters in encoded file with respect to the source file. Another one is to find out the chi-square value for the test of homogeneity using a statistical method[122]. If the encoded message does not satisfy the homogeneity i.e. non-homogeneous with respect to the source message, the encoded message may supply the security in encoding. To implement the first method the frequency distribution graph for both source file and the encoded file is drawn along with the RSA encoded file for comparison. The principle of the second one is described here with an example and its application in encoding is also discussed.

Let a population be classified according to two attributes A and B into k and l classes respectively, say,

$A_1, A_2, \ldots\ldots\ldots\ldots A_k$

$B_1, B_2, \ldots\ldots\ldots\ldots B_l$

Let P_{ij} be the proportion of members of the population belonging simultaneously to the i-th class of A (i.e A_i) and the j-th class of B (i.e B_j). The structure of the population will then be as follows in table 1.1.

Table 1.1: Proportion of members of population

		B – Class					
		B_1	B_2	B_3	.....	B_l	Total
A-class	A_1	P_{11}	P_{12}	P_{13}	.....	P_{1l}	P_{1o}
	A_2	P_{21}	P_{22}	P_{23}	.....	P_{2l}	P_{2o}
	A_3	P_{31}	P_{32}	P_{33}	.....	P_{3l}	P_{3o}
	:	:	:	:	:	:	
	A_k	P_{k1}	P_{k2}	P_{k3}	.....	P_{kl}	P_{ko}
	Total	P_{o1}	P_{o2}	P_{o3}	.....	P_{ol}	1

The proportion P_{ij} defines the joint distribution of A and B. The marginal totals

$$P_{io} = \sum_{j=1}^{l} P_{ij}$$

give the marginal distribution of A while the other marginal totals

$$P_{oj} = \sum_{i=1}^{k} P_{ij}$$

give the marginal distribution of B. When P_{ij} are unknown, we may enquire whether A & B are independent.

We have then to test the hypothesis H_o : $P_{ij} = P_{io} \times P_{oj}$ for all i,j

Let a random sample of side n be drawn from the population, the drawing being mutually independent. If we denote by f ij the number of members of the sample that belong to the i-th class of A and to the j-th class of B then, under the hypothesis H_o ;

$$\sum_{i=1}^{k} \sum_{j=1}^{l} (f_{ij} - n P_{io} P_{oj})^2 / n P_{io} P_{oj} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

will be distributed as approximately a κ^2 with df = kl -1. where df is degree of freedom.

This statis tics could be used to test Ho if Pio and Poi were known quantities.

In the present case, however, they are unknown and have to be estimated from the sample itself.

The proper estimator of P_{io} which is the population proportion in the class Ai, is the corresponding sample proportion f_{io}/n, where $f_{io} = \sum_j f_{ij}$

Similarly, the proper estimator of Poj is f_{oj}/n, where $f_{oj} = \sum_i f_{ij}$

Substituting these estimators for Pio and Poj in equation (1), we get the statistics

$$\sum_i \sum_j (f_{ij} - f_{io} f_{oj} /n)^2 / (f_{io} f_{oj} /n) \quad = n \sum_i \sum_j f_{ij}^2 / f_{io} f_{oj} - n \quad \ldots\ldots (2)$$

We are using (k+l) estimators, of which (k-1) + (l-1) are independent.

For, given (k-1) of the estimators for Pio, the other automatically follows; and similarly, given (l-1) of the estimators for Poj, other is automatically determined. Hence equation (2) is distributed as approximately a κ^2 with

$$(kl -1)- (k-1) - (l-1) = (k-1) (l-1) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

degrees of freedom.

Thus we see that, although the problem is different from that in the solution of each is formally same, the κ^2 statistics used in each case being of the form

(Grand Total) $\sum_i \sum_j$ [(class frequency)^2 / (row total)(column total)] - (Grand Total)

with degree of freedom, df = (no of rows -1) (no of column -1)

Therefore, $\kappa^2 = n [\sum_i \sum_j f_{ij}^2 / f_{io} f_{oj} - 1]$ with df =(k-1)(l-1)

Application in encryption : Let a population be classified according to attributes A and B, into k and l classes

$A_1, A_2, \ldots\ldots\ldots\ldots A_k$

$B_1, B_2, \ldots\ldots\ldots\ldots B_l$

It means the presence of characters in source and encoded files respectively. Let fij lie the frequency of occurrence of members in the population belonging simultaneously to the i-th class of A (i.e. A i) and the j-th class of B (i.e B_j), then structure of the frequency distribution will then be as followsin the table1.2.

Table 1.2: Occurrence of characters in the file

		Characters in file					
		Ch1	Ch2	Ch3	.....	Chl	Total
Files	File1	f_{11}	f_{12}	f_{13}	.....	f_{1l}	f_{1o}
	File2	f_{21}	f_{22}	f_{23}	.....	f_{2l}	f_{2o}
	File3	f_{31}	f_{32}	f_{33}	.....	f_{3l}	f_{3o}
	:	:	:	:	:	:	
	filek	f_{k1}	f_{k2}	f_{k3}	.....	f_{kl}	f_{ko}
	Total	f_{o1}	f_{o2}	f_{o3}	.....	f_{ol}	Total(N)

Now let us consider the situation where the source file and the encoded file will tested for homogeneity. Hence the number of files will be two. Again in each file there may be maximum 256 characters in each file. Hence the frequencies may be represented as given in the table.

The formula of chi-square is

$$\kappa^2 = n\left[\sum_i \sum_j f_{ij}^2 / f_{io} f_{oj} - 1\right] \quad \text{with df} = (k-1)(l-1)$$

Here k=2 and l=256

Under Ho, Expected frequency $Ei = (f_{io} f_{oj})/N$

and observed frequency, $Oi^2 = f_{1i}^2 + f_{2i}^2$

and the chi-square, $\kappa^2 = \sum (O_i^2 / E_i) - N$

Frequencies will be clubbed together if either observed frequencies are less than expected frequencies and values are less than 5, corresponding df will be reduced by 1.

Now $\kappa^2 = (f_{11}^{\ 2} + f_{21}^{\ 2}) / [(f_{o1} + f_{1o})/N] + (f_{12}^{\ 2} + f_{22}^{\ 2}) / [(f_{o2} + f_{1o})/N]$

$+ (f_{13}^{\ 2} + f_{23}^{\ 2}) / [(f_{o3} + f_{1o})/N] + \ldots\ldots + (f_{11}^{\ 2} + f_{21}^{\ 2}) / [(f_{o1} + f_{1o})/N] - N$

If calculated κ^2 > tabulated $\kappa^2_{\alpha,(k-1)(l-1)}$ then the files are non-homogeneous. Therefore, if calculated κ^2 > tabulated $\kappa^2_{0.01,(2-1)(256-1)}$, the two files non-homogeneous with uncertainty limit of 1%.

Consider an example where five samples are taken. The table 1.3 shows the observed frequencies and expected frequencies (in bracket).

Table 1.3: A sample of five characters

	Symbols					
File	A	B	C	D	E	Total
Source file	10 (8)	20 (16)	15 (16.5)	8 (4.5)	4 (12)	57
Encoded file	6 (8)	12 (16)	18 (16.5)	1 (4.5)	20 (12)	57
	16	32	33	9	24	114

Calculated $\kappa^2 = (10^2 + 6^2)/ [(57x16)/114] + (20^2 + 12^2)/ [(57x32)/114]$

$+(15^2 + 18^2)/ [(57x33)/114] + (8^2 + 1^2)/ [(57x9)/114]$

$+ (4^2 + 20^2)/ [(57x24)/114] - 114 = 22.81$

Tabulated $\kappa^2_{0.01,(2-1)(5-1)} = 9.488$

Since calculated κ^2 > tabulated $\kappa^2_{0.01,(k-1)(l-1)}$, the encoded file is non-homogeneous with respect to the source file.