ABSTRACT

In this project I will describe a method for integrating together cryptography and steganography through image processing. In particular, I will present a system able to perform steganography and cryptography at the same time using images as cover objects for steganography and New Pythagorean Triple Algorithm as keys for cryptography. I will show such system is an effective steganographic one and is also a theoretically unbreakable cryptographic one through the New Pythagorean Triple algorithm I can extend the definition of the Pythagorean Theorem which states that for any p and q (one of them is odd and the other even), there is only one fundamental solution (x,y,z). Using the New Pythagorean Triple algorithm formulas, this definition can be re-stated to: for any numbers p and q (one of them is odd and the other even) there are at least two fundamental solutions (x_1,y_1,z_1) and (x_2,y_2,z_2), but there are also special cases when even three fundamental solutions are possible (x_1,y_1,z_1), (x_2,y_2,z_2) and (x_3,y_3,z_3). Based on these solutions I can easily create the encryption and decryption key that can be used in a simple symmetric cryptosystem.

CHAPTER ONE

1.0 INTRODUCTION

Cryptography and steganography are well known and widely used techniques that manipulate information (messages) in order to cipher or hide their existence. These techniques have many applications in computer science and other related fields: they are used to protect e-mail messages, credit card information, corporate data, etc.

More specifically, steganography is the art and science of communicating in a way which hides the existence of the communication (Johnson and Jajodia, 1998). A steganographic system thus embeds hidden content in unremarkable cover media so as not to arouse an eavesdropper's suspicion (Provos and Honeyman, 2003). As an example, it is possible to embed a text inside an image or an audio file.

On the other hand, cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication (Menezes et al., 1996). In this paper we will focus only on confidentiality, i.e., the service used to keep the content of information from all but those authorized to have it.

Cryptography protects information by transforming it into an unreadable format. It is useful to achieve confidential transmission over a public network. The original text, or *plaintext*, is converted into a coded equivalent called *ciphertext* via an encryption algorithm. Only those who possess a secret key can decipher (*decrypt*) the ciphertext into plaintext.

Cryptography systems can be broadly classified into symmetric-key systems (see Fig. 1) that use a single key (i.e., a *password*) that both the sender and the receiver have, and public-key systems that use two keys, a public key known to everyone and a private key that only the recipient of messages uses. I will discuss only symmetric-key systems.

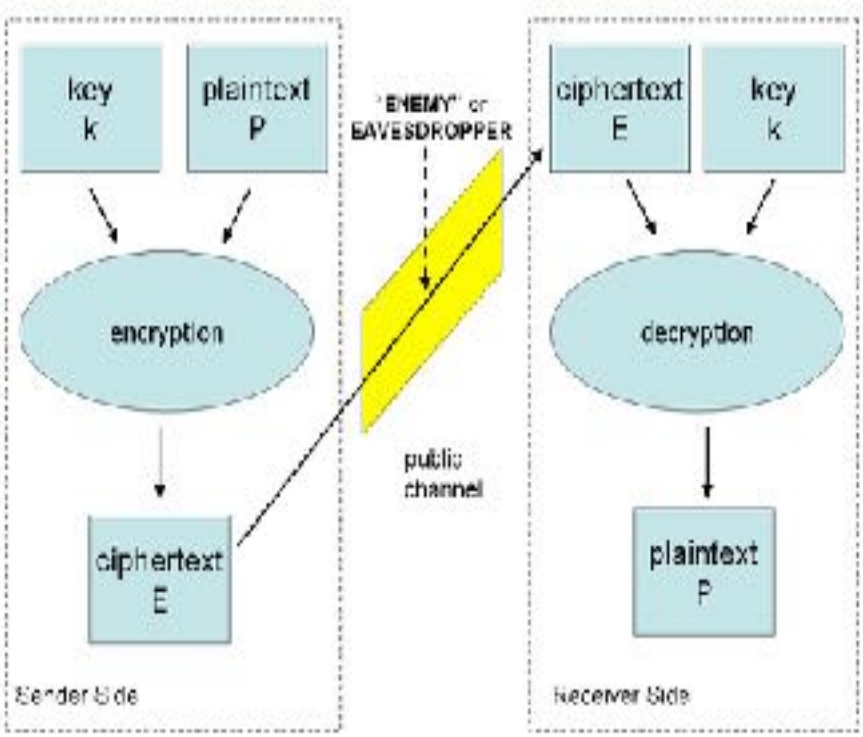


Figure 1: Symmetric-key Cryptographic Model.

1.1 BACKGROUND OF THE STUDY

Cryptography was used to assure only secrecy. Wax seals, signatures, and other physical mechanisms were typically used to assure integrity of the media and authenticity of the sender. With the advent of electronic funds transfer, the applications of cryptography for integrity began to surpass its use for secrecy. Electronic cash came into being from cryptography, and the electronic credit card and debit card sprung into widespread use. The advent of public key cryptography introduced the possibility of digital signatures, and other related

concepts such as electronic credentials. In the information age, cryptography has become one of the major methods for protection in all applications.

Cryptographic protocols have only recently come under intensive study, and as of this time, they are not sufficiently well developed to provide a great deal of assurance. There are several protocols that offer provable properties, primarily those designed for use with the OTP (One-Time-Password). The problem with proving properties of protocols under other schemes is that the mathematics is extremely complex for the RSA (Rivest-Shamir-Adleman) , and there is no sound mathematical basis for the DES (Data Encryption Standard). Much research is under way at this time in the field of protocol analysis and verification, and it is likely that once this field stabilizes, cryptographic protocols will follow suit.

Several special purpose cryptographic protocols have been developed and demonstrated sound. Most notably, the RSA key distribution protocol, a public key poker playing protocol, an OTP based dining cryptographers protocol, and the protocol used for verifying the nuclear test ban treaty.

A typical cryptographic protocol failure is encountered in the use of the RSA. It seems that if an attacker can choose the plaintext to be signed under an RSA signature system, observe the result of the signature, and then iterate the process, it is possible to get the signer to reveal the private key in a very small

number of signatures (about 1 signature per bit of the key). Thus an unmodified RSA signature system requires a sound protocol to be safely used.

Cryptography and steganography are cousins in the spy craft family: the former scrambles a message so it cannot be understood, the latter hides the message so it cannot be seen. A cipher message, for instance, might arouse suspicion on the part of the recipient while an invisible message created with steganographic methods will not.

In fact, steganography can be useful when the use of cryptography is forbidden: where cryptography and strong encryption are outlawed, steganography can circumvent such policies to pass message covertly. However, steganography and cryptography differ in the way they are evaluated: steganography fails when the ”enemy” is able to access the content of the cipher message, while cryptography fails when the ”enemy” detects that there is a secret message present in the steganographic medium (Johnson and Jajodia, 1998).

1.2 RESEARCH MOTIVATION

The disciplines that study techniques for deciphering cipher messages and detecting hide messages are called *cryptanalysis* and *steganalysis*. The former denotes the set of methods for obtaining the meaning of encrypted information, while the latter is the art of discovering covert messages.

1.3 RESEARCH OBJECTIVE

The aim of this project is to suggest and implement a new method for integrating together cryptography and steganography through Pythagorean Triple and Image Processing. In particular, presenting a system able to perform steganography and cryptography at the same time. Such system as an effective steganographic one and is also a theoretically unbreakable cryptographic one.

1.4 RESEARCH METHODOLOGY

CHAPTER TWO

2.0 LITERATURE REVIEW

2.1 COMPUTER SECURITY CONCEPTS

2.11 Computer Security

The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

This definition introduces three key objectives that are at the heart of computer security:

Confidentiality: This term covers two related concepts:

Data confidentiality: Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

Privacy: Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.

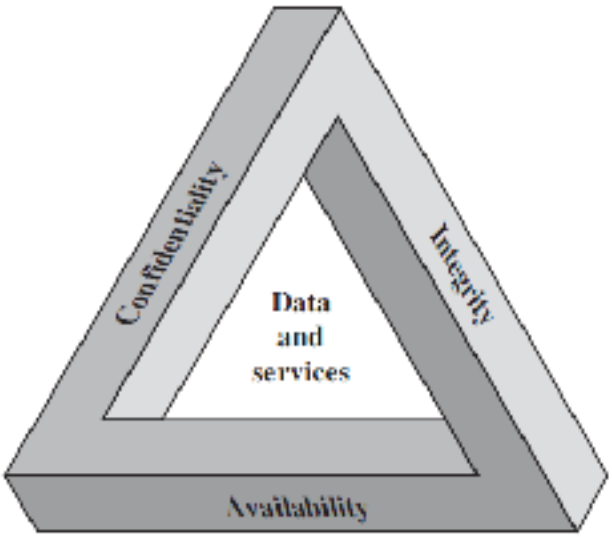
• **Integrity:** This term covers two related concepts:

Data integrity: Assures that information and programs are changed only in a specified and authorized manner.

System integrity: Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.

• **Availability:** Assures that systems work promptly and service is not denied to authorize users.

The three concepts embody the fundamental security objectives for both data and for information and computing services.



(*Standards for Security Categorization of Federal Information and Information Systems*) lists confidentiality, integrity, and availability as the three security

objectives for information and for information systems. FIPS 199 provides a useful characterization of these three objectives in terms of requirements and the definition of a loss of security in each category:

- **Confidentiality:** Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. A loss of confidentiality is the unauthorized disclosure of information.

- **Integrity:** Guarding against improper information modification or destruction, including ensuring information nonrepudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information.

- **Availability:** Ensuring timely and reliable access to and use of information. A loss of availability is the disruption of access to or use of information or an information system.

Although the use of the CIA triad to define security objectives is well established, some in the security field feel that additional concepts are needed to present a complete picture. Two of the most commonly mentioned are as follows:

- **Authenticity:** The property of being genuine and being able to be verified and trusted; confidence in the validity of a transmission, a

message, or message originator. This means verifying that users are who they say they are and that each input arriving at the system came from a trusted source.

• **Accountability:** The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity. This supports nonrepudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action. Because truly secure systems are not yet an achievable goal, we must be able to trace a security breach to a responsible party. Systems must keep records of their activities to permit later forensic analysis to trace security breaches or to aid in transaction disputes.

2.2 The Challenges of Computer Security

Computer and network security is both fascinating and complex. Some of the reasons follow:

1. Security is not as simple as it might first appear to the novice. The requirements seem to be straightforward; indeed, most of the major requirements for security services can be given self-explanatory, one-word labels: confidentiality, authentication, nonrepudiation, or integrity. But the mechanisms used to meet

those requirements can be quite complex, and understanding them may involve rather subtle reasoning.

2. In developing a particular security mechanism or algorithm, one must always consider potential attacks on those security features. In many cases, successful attacks are designed by looking at the problem in a completely different way, therefore exploiting an unexpected weakness in the mechanism.

3. Because of point 2, the procedures used to provide particular services are often counterintuitive. Typically, a security mechanism is complex, and it is not obvious from the statement of a particular requirement that such elaborate measures are needed. It is only when the various aspects of the threat are considered that elaborate security mechanisms make sense.

4. Having designed various security mechanisms, it is necessary to decide where to use them. This is true both in terms of physical placement (e.g., at what points in a network are certain security mechanisms needed) and in a logical sense [e.g., at what layer or layers of an architecture such as TCP/IP

(Transmission Control Protocol/Internet Protocol) should mechanisms be placed.

5. Security mechanisms typically involve more than a particular algorithm or protocol. They also require that participants be in possession of some secret information (e.g., an encryption key), which raises questions about the creation, distribution, and protection of that secret information. There also may be a reliance on communications protocols whose behavior may complicate the task of developing the security mechanism. For example, if the proper functioning of the security mechanism requires setting time limits on the transit time of a message from sender to receiver, then any protocol or network that introduces variable, unpredictable delays may render such time limits meaningless.

6. Computer and network security is essentially a battle of wits between a perpetrator who tries to find holes and the designer or administrator who tries to close them. The great advantage that the attacker has is that he or she need only find a single weakness, while the designer must find and eliminate all weaknesses to achieve perfect security.

7. There is a natural tendency on the part of users and system managers to perceive little benefit from security investment until a security failure occurs.

8. Security requires regular, even constant, monitoring, and this is difficult in today's short-term, overloaded environment.

9. Security is still too often an afterthought to be incorporated into a system after the design is complete rather than being an integral part of the design process.

10. Many users and even security administrators view strong security as an impediment to efficient and user-friendly operation of an information system or use of information.

2.21 Threats and Attacks

Threat

A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.

Attack

An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system.

The security attacks, mechanisms, and services. These can be defined briefly as

- Security attack: Any action that compromises the security of information owned by an organization.

• Security mechanism: A process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack.

• Security service: A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

2.22 Security Attacks

A useful means of classifying security attacks, in terms of *passive attacks* and *active attacks.* A passive attack attempts to learn or make use of information from the system but does not affect system resources. An active attack attempts to alter system resources or affect their operation.

Passive Attacks

Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted. Two types of passive attacks are the release of message contents and traffic analysis.

The **release of message contents** is easily understood (Figure 1.2a).A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information. We would like to prevent an opponent from learning the contents of these transmissions.

A second type of passive attack, **traffic analysis**, is subtler. Suppose that we had a way of masking the contents of messages or other information traffic so that opponents, even if they captured the message, could not extract the information from the message. The common technique for masking contents is encryption. If we had encryption protection in place, an opponent might still be able to observe the pattern of these messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged. This information might be useful in guessing the nature of the communication that was taking place.

Passive attacks are very difficult to detect, because they do not involve any alteration of the data. Typically, the message traffic is sent and received in an apparently normal fashion, and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern. However, it is feasible to prevent the success of these attacks, usually by means of encryption. Thus, the emphasis in dealing with passive attacks is on prevention rather than detection.

Active Attacks

Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories: masquerade, replay, modification of messages, and denial of service.

A **masquerade** takes place when one entity pretends to be a different entity. A masquerade attack usually includes one of the other forms of active attack. For example, authentication sequences can be captured and replayed after a valid authentication sequence has taken place, thus enabling an authorized entity with few privileges to obtain extra privileges by impersonating an entity that has those privileges.

Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.

Modification of messages simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect. For example, a message meaning "Allow John Smith to read confidential file *accounts*" is modified to mean "Allow Fred Brown to read confidential file *accounts*."

The **denial of service** prevents or inhibits the normal use or management of communications facilities. This attack may have a specific target; for example, an entity may suppress all messages directed to a particular destination

2.3 Security Services

A security service as a service that is provided by a protocol layer of communicating open systems and that ensures adequate security of the systems or of data transfers.

2.31 Authentication

The authentication service is concerned with assuring that a communication is authentic. In the case of a 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. In the case of an ongoing interaction, such as the connection of a terminal to a host, two aspects are involved. First, at the time of connection initiation, the service assures that the two entities are authentic, that is, that each is the entity that it claims to be. Second, the service must assure that the connection is not interfered with in such a way that a third party can masquerade as one of the two legitimate parties for the purposes of unauthorized transmission or reception.

Two specific authentication services are defined as:

• **Peer entity authentication:** Provides for the corroboration of the identity of a peer entity in an association. Two entities are considered peers if they implement to same protocol in different systems; e.g., two TCP modules in two communicating systems. Peer entity authentication is provided for use at the establishment of, or at times during the data transfer phase of, a connection. It attempts to provide confidence that an entity is not performing either a masquerade or an unauthorized replay of a previous connection.

• **Data origin authentication:** Provides for the corroboration of the source of a data unit. It does not provide protection against the duplication or modification of data units. This type of service supports applications like electronic mail, where there are no prior interactions between the communicating entities.

2.32 Access Control

In the context of network security, access control is the ability to limit and control the access to host systems and applications via communications links. To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be tailored to the individual.

2.33 Data Confidentiality

Confidentiality is the protection of transmitted data from passive attacks. With respect to the content of a data transmission, several levels of protection can be identified. The broadest service protects all user data transmitted between two users over a period of time. For example, when a TCP connection is set up between two systems, this broad protection prevents the release of any user data transmitted over the TCP connection. Narrower forms of this service can also be defined, including the protection of a single message or even specific fields within a message. These refinements are less useful than the broad approach and may even be more complex and expensive to implement.

The other aspect of confidentiality is the protection of traffic flow from analysis. This requires that an attacker not be able to observe the source and destination, frequency, length, or other characteristics of the traffic on a communications facility.

2.34 Data Integrity

As with confidentiality, integrity can apply to a stream of messages, a single message, or selected fields within a message. Again, 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, or replays. The destruction of data is also covered under this service. Thus, the connection-oriented integrity service addresses both message stream modification and denial of service. On the other hand, a connectionless integrity service, one that deals with individual messages without regard to any larger context, generally provides protection against message modification only.

We can make a distinction between service with and without recovery. Because the integrity service relates to active attacks, we are concerned with detection rather than prevention. If a violation of integrity is detected, then the service may simply report this violation, and some other portion of software or human intervention is required to recover from the violation. Alternatively, there are mechanisms available to recover from the loss of integrity of data, as we will review subsequently. The incorporation of automated recovery mechanisms is, in general, the more attractive alternative.

2.35 Nonrepudiation

Nonrepudiation prevents either sender or receiver from denying a transmitted message. Thus, when a message is sent, the receiver can prove that the alleged sender in fact sent the message. Similarly, when a message is received, the sender can prove that the alleged receiver in fact received the message.

2.36 Availability Service

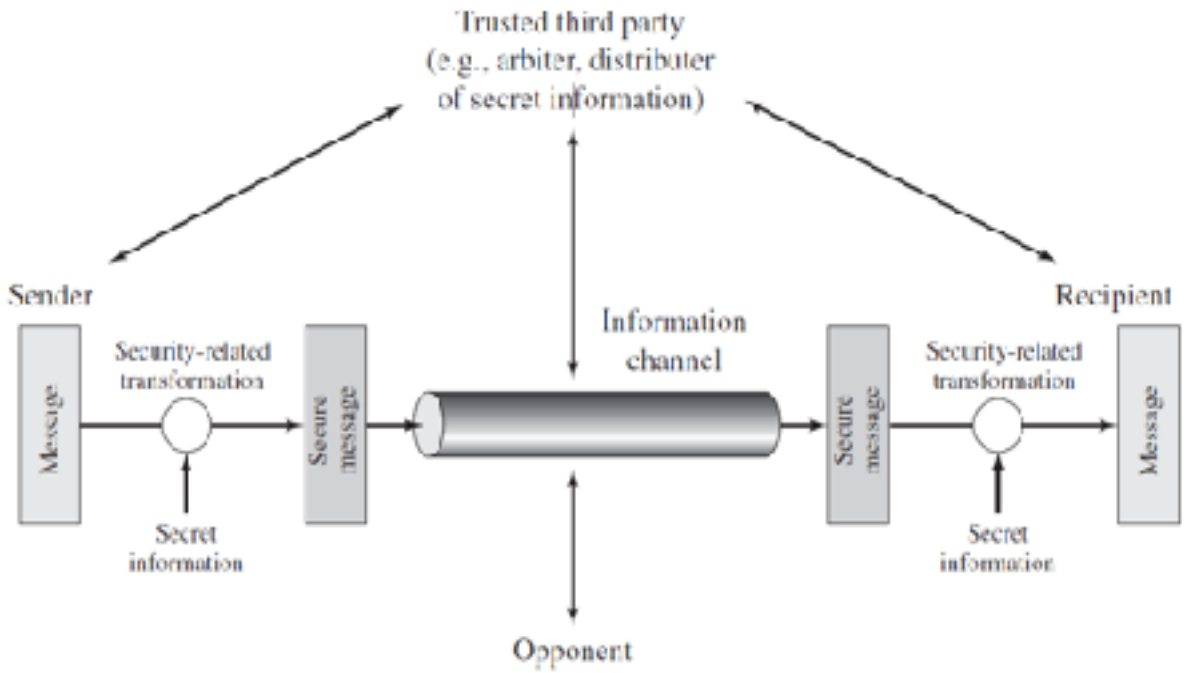
Availability is the property of a system or a system resource being accessible and usable upon demand by an authorized system entity, according to performance specifications for the system (i.e., a system is available if it provides services according to the system design whenever users request them). A variety of attacks can result in the loss of or reduction in availability. Some of these attacks are amenable to automated countermeasures, such as authentication and encryption, whereas others require some sort of physical action to prevent

2.4 Network Security

A message is to be transferred from one party to another across some sort of Internet service. The two parties, who are the *principals* in this transaction, must cooperate for the exchange to take place. A logical information channel is established by defining a route through the Internet from source to destination and by the cooperative use of communication protocols (e.g., TCP/IP) by the two principals.

Security aspects come into play when it is necessary or desirable to protect the information transmission from an opponent who may present a threat to confidentiality, authenticity, and so on. All the techniques for providing security have two components:

A security-related transformation on the information to be sent. Examples include the encryption of the message, which scrambles the message so that it is unreadable by the opponent, and the



addition of a code based on the contents of the message, which can be used to verify the identity of the sender

2.41 Model for Network Security

Some secret information shared by the two principals and, it is hoped, unknown to the opponent. An example is an encryption key used in conjunction with the transformation to scramble the message before transmission and unscramble it on reception.

A trusted third party may be needed to achieve secure transmission. For example, a third party may be responsible for distributing the secret information to the two principals while keeping it from any opponent. Or a third party may be needed to arbitrate disputes between the two principals concerning the authenticity of a message transmission.

This general model shows that there are four basic tasks in designing a particular security service:

1. Design an algorithm for performing the security-related transformation. The algorithm should be such that an opponent cannot defeat its purpose.

2. Generate the secret information to be used with the algorithm.

3. Develop methods for the distribution and sharing of the secret information.

4. Specify a protocol to be used by the two principals that makes use of the security algorithm and the secret information to achieve a particular security service.

2.5 Encryption Technics

2.51 Symmetric Encryption

Symmetric encryption, also referred to as conventional encryption or single-key encryption, was the only type of encryption in use prior to the development of public-key encryption in the 1970s. It remains by far the most widely used of the two types of encryption.

An original message is known as the **plaintext**, while the coded message is called the **ciphertext**.

The process of converting from plaintext to ciphertext is known as **enciphering** or **encryption**; restoring the plaintext from the ciphertext is **deciphering** or **decryption**.

The many schemes used for encryption constitute the area of study known as **cryptography**. Such a scheme is known as a **cryptographic system** or a **cipher**.

Techniques used for deciphering a message without any knowledge of the enciphering details fall into the area of **cryptanalysis**. Cryptanalysis is what the layperson calls “breaking the code.” The areas of cryptography and cryptanalysis together are called **cryptology**.

2.52 Symmetric Cipher

A symmetric encryption scheme has five ingredients:

• **Plaintext:** This is the original intelligible message or data that is fed into the algorithm as input.

• **Encryption algorithm:** The encryption algorithm performs various substitutions and transformations on the plaintext.

• **Secret key:** The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time. The exact substitutions and transformations performed by the algorithm depend on the key.

• **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts. The ciphertext is an apparently random stream of data and, as it stands, is unintelligible.

• **Decryption algorithm:** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.

2.6 Information Hiding

information hiding is the principle of segregation of the *design decisions* in a computer program that are most likely to change, thus protecting other parts of the program from extensive modification if the design decision is changed. The protection involves providing a stable interface which protects the remainder of

the program from the implementation (the details that are most likely to change).

Written another way, information hiding is the ability to prevent certain aspects of a class or software component from being accessible to its clients, using either programming language features (like private variables) or an explicit exporting policy.

The term *encapsulation* is often used interchangeably with information hiding. Not all agree on the distinctions between the two though; one may think of information hiding as being the principle and encapsulation being the technique. A software module hides information by encapsulating the information into a module or other construct which presents an interface.[1]

A common use of information hiding is to hide the physical storage layout for data so that if it is changed, the change is restricted to a small subset of the total program. For example, if a three-dimensional point (x,y,z) is represented in a program with three floating point scalar variables and later, the representation is changed to a single array variable of size three, a module designed with information hiding in mind would protect the remainder of the program from such a change.

In object-oriented programming, information hiding (by way of nesting of types) reduces software development risk by shifting the code's dependency on an uncertain implementation (design decision) onto a well-defined interface.

Clients of the interface perform operations purely through it so if the implementation changes, the clients do not have to change.

2.61 Encapsulation

Encapsulation can be define as "the process of compartmentalizing the elements of an abstraction that constitute its structure and behavior; encapsulation serves to separate the contractual interface of an abstraction and its implementation."

The purpose is to achieve potential for change: the internal mechanisms of the component can be improved without impact on other components, or the component can be replaced with a different one that supports the same public interface. Encapsulation also protects the integrity of the component, by preventing users from setting the internal data of the component into an invalid or inconsistent state. Another benefit of encapsulation is that it reduces system complexity and thus increases robustness, by limiting the interdependencies between software components.

In this sense, the idea of encapsulation is more general than how it is applied in OOP: for example, a relational database is encapsulated in the sense that its only public interface is a Query language (SQL for example), which hides all the internal machinery and data structures of the database management system. As such, encapsulation is a core principle of good software architecture, at every level of granularity.

Encapsulating software behind an interface allows the construction of objects that mimic the behavior and interactions of objects in the real world. For example, a simple digital alarm clock is a real-world object that a lay person can use and understand. They can understand what the alarm clock does, and how to use it through the provided interface (buttons and screen), without having to understand every part inside of the clock. Similarly, if you replaced the clock with a different model, the lay person could continue to use it in the same way, provided that the interface works the same.

In the more concrete setting of an object-oriented programming language, the notion is used to mean either an information hiding mechanism, a bundling mechanism, or the combination of the two. (See Encapsulation (object-oriented programming) for details.)

2.62 History of information hiding

The concept of information hiding was first described by David Parnas in Parnas (1972). Before then, modularity was discussed by Richard Gauthier and Stephen Pont in their 1970 book *Designing Systems Programs* although modular programming itself had been used at many commercial sites for many years previously – especially in I/O sub-systems and software libraries – without acquiring the 'information hiding' tag – but for similar reasons, as well as the more obvious code reuse reason.

63. Example of information hiding

Information hiding serves as an effective criterion for dividing any piece of equipment, software or hardware, into modules of functionality. For instance a car is a complex piece of equipment. In order to make the design, manufacturing, and maintenance of a car reasonable, the complex piece of equipment is divided into modules with particular interfaces hiding design decisions. By designing a car in this fashion, a car manufacturer can also offer various options while still having a vehicle which is economical to manufacture.

For instance, a car manufacturer may have a luxury version of the car as well as a standard version. The luxury version comes with a more powerful engine than the standard version. The engineers designing the two different car engines, one for the luxury version and one for the standard version, provide the same interface for both engines. Both engines fit into the engine bay of the car which is the same between both versions. Both engines fit the same transmission, the same engine mounts, and the same controls. The differences in the engines are that the more powerful luxury version has a larger displacement with a fuel injection system that is programmed to provide the fuel air mixture that the larger displacement engine requires.

In addition to the more powerful engine, the luxury version may also offer other options such as a better radio with CD player, more comfortable seats, a better suspension system with wider tires, and different paint colors.

With all of these changes, most of the car is the same between the standard version and the luxury version. The radio with CD player is a module which replaces the standard radio, also a module, in the luxury model. The more comfortable seats are installed into the same seat mounts as the standard types of seats. Whether the seats are leather or plastic, or offer lumbar support or not, doesn't matter.

The engineers design the car by dividing the task up into pieces of work which are assigned to teams. Each team then designs their component to a particular standard or interface which allows the sub-team flexibility in the design of the component while at the same time ensuring that all of the components will fit together.

Motor vehicle manufacturers frequently use the same core structure for several different models, in part as a cost-control measure. Such a "platform" also provides an example of information hiding, since the floorpan can be built without knowing whether it is to be used in a sedan or a hatchback.

As can be seen by this example, information hiding provides flexibility. This flexibility allows a programmer to modify functionality of a computer program during normal evolution as the computer program is changed to better fit the needs of users. When a computer program is well designed decomposing the source code solution into modules using the principle of information hiding, evolutionary changes are much easier because the changes typically are local rather than global changes.

Cars provide another example of this in how they interface with drivers. They present a standard interface (pedals, wheel, shifter, signals, gauges, etc.) on which people are trained and licensed. Thus, people only have to learn to drive a car; they don't need to learn a completely different way of driving every time they drive a new model. (Granted, there are manual and automatic transmissions and other such differences, but on the whole cars maintain a unified interface.)

2.7 CRYPTOGRAPHY

Cryptographic systems are characterized along three independent dimensions:

1. The type of operations used for transforming plaintext to ciphertext. All encryption algorithms are based on two general principles: substitution, in which each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, and transposition, in which elements in the plaintext are rearranged. The fundamental requirement is that no information be lost (that is, that all operations are reversible). Most systems, referred to as *product systems*, involve multiple stages of substitutions and transpositions.

2. The number of keys used. If both sender and receiver use the same key, the system is referred to as symmetric, single-key, secret-key, or

conventional encryption. If the sender and receiver use different keys, the system is referred to as asymmetric, two-key, or public-key encryption.

3. The way in which the plaintext is processed. A *block cipher* processes the input one block of elements at a time, producing an output block for each input block. A *stream cipher* processes the input elements continuously, producing output one element at a time, as it goes along.

Cryptography systems can be broadly classified into symmetric-key systems that use a single key that both the sender and recipient have, and *public-key* systems that use two keys, a public key known to everyone and a private key that only the recipient of messages uses.

- Encryption is the translation of data into a secret code. Encryption is the most effective way to achieve data security. To read an encrypted file, you must have access to a secret key or password that enables you to *decrypt* it. Unencrypted data is called *plain text* ; encrypted data is referred to as *cipher text*.

There are two main types of encryption: asymmetric encryption (also called public-key encryption) and symmetric encryption.

2.71 Cryptographic algorithms and protocols can be grouped into four main areas:

• **Symmetric encryption:** Used to conceal the contents of blocks or streams of data of any size, including messages, files, encryption keys, and passwords.

• **Asymmetric encryption:** Used to conceal small blocks of data, such as encryption keys and hash function values, which are used in digital signatures.

• **Data integrity algorithms:** Used to protect blocks of data, such as messages, from alteration.

• **Authentication protocols:** These are schemes based on the use of cryptographic algorithms designed to authenticate the identity of entities.

2.72 Cryptanalysis and Brute-Force Attack

Typically, the objective of attacking an encryption system is to recover the key in use rather than simply to recover the plaintext of a single ciphertext. There are two general approaches to attacking a conventional encryption scheme:

• **Cryptanalysis:** Cryptanalytic attacks rely on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext–ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.

• **Brute-force attack:** The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.

If either type of attack succeeds in deducing the key, the effect is catastrophic. All future and past messages encrypted with that key are compromised.

Table below summarizes the various types of **cryptanalytic attacks** based on the amount of information known to the cryptanalyst. The most difficult problem is presented when all that is available is the *ciphertext only*. In some cases, not even the encryption algorithm is known, but in general, we can assume that the opponent does know the algorithm used for encryption. One possible attack under these circumstances is the brute-force approach of trying all possible keys. If the key space is very large, this becomes impractical. Thus, the opponent must rely on an analysis of the ciphertext itself, generally applying various statistical tests to it. To use this approach, the opponent must have some general idea of the type of plaintext that is concealed, such as English or French text, an EXE file, a Java source listing, an accounting file, and so on.

Types of Attacks on Encrypted Messages

Type of Attack	**Known to Cryptanalyst**

Ciphertext Only	• Encryption algorithm • Ciphertext
Known Plaintext	• Encryption algorithm • Ciphertext • One or more plaintext–ciphertext pairs formed with the secret key
Chosen Plaintext	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen Ciphertext	• Encryption algorithm • Ciphertext • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

Chosen Text	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

2.8 A Brief History of Cryptography

The earliest form of cryptography was the simple writing of a message, as most people could not read (New World, 2007). In fact, the very word cryptography comes from the Greek words kryptos and graphein, which mean hidden and writing, respectively (Pawlan, 1998).

Early cryptography was solely concerned with converting messages into unreadable groups of figures to protect the message's content during the time the message was being carried from one place to another. In the modern era, cryptography has grown from basic message confidentiality to include some phases of message integrity checking, sender/receiver identity authentication, and digital signatures, among other things (New World, 2007).

The need to conceal messages has been with us since we moved out of caves, started living in groups and decided to take this civilization idea seriously. As soon as there were different groups or tribes, the idea that we had to work against each other surfaced and was proliferated, along with rank violence, secrecy, and crowd manipulation. The earliest forms of cryptography were found in the cradle of civilization, which comes as no surprise, including the regions currently encompassed by Egypt, Greece and Rome.

As early as 1900 B.C., Egyptian scribes used hieroglyphs in a non-standard fashion, presumably to hide the meaning from those who did not know the meaning (Whitman, 2005). The Greek's idea was to wrap a tape around a stick, and then write the message on the wound tape. When the tape was unwound, the writing would be meaningless. The receiver of the message would of course have a stick of the same diameter and use it to decipher the message. The Roman method of cryptography was known as the Caesar Shift Cipher. It utilized the idea of shifting letters by an agreed upon number (three was a common historical choice), and thus writing the message using the letter-shift. The receiving group would then shift the letters back by the same number and decipher the message (Taylor, 2002).

The Caesar Shift Cipher is an example of a Monoalphabetic Cipher. It is easy to see why this method of encryption is simple to break. All a person has to do is to go down the alphabet, juxtapositioning the start of the alphabet to each succeeding letter. At each iteration, the message is decrypted to see if it makes

sense. When it does appear as a readable message, the code has been broken. Another way to break Monoalphabetic ciphers is by the use of what is known as frequency analysis, attributed to the Arabs circa 1000 C.E. (New World, 2007). This method utilizes the idea that certain letters, in English the letter "e," for instance, are repeated more often than others. Armed with this knowledge, a person could go over a message and look for the repeated use, or frequency of use, of a particular letter and try to substitute known frequently used letters (Taylor, 2002).

As for the Greek method of using a stick, once the method was known, it was a simple matter of trying out sticks of different diameters until the message became readable.

The art and science of cryptography showed no major changes or advancements until the Middle Ages. By that time, all of the western European governments were utilizing cryptography in one form or another. Keeping in touch with ambassadors was the major use of cryptography. One Leon Battista Alberti was known as “The Father of Western Cryptology,” most notably due to his development of polyalphabetic substitution. His method was to use two copper disks that fit together. Each one of them had the alphabet inscribed on it. After every few words, the disks were rotated to change the encryption logic, thereby limiting the use of frequency analysis to crack the cipher (Cohen, 1990). Polyalphabetic substitution went through a variety of changes and is most notably attributed to Vigenere, although Rubin claims that he in fact had

nothing to do with its creation. Rubin further points out that the use of the cipher disks continued in the Civil War, with the South using brass cipher disks, although the North regularly cracked the messages (2008).

Gilbert Vernam worked to improve the broken cipher, creating the Vernam-Vigenere cipher in 1918, but was unable to create one of significantly greater strength. His work did lead to the *one time pad*, which uses a key word only once, and it proved to be near unbreakable (Rubin, 2008). Whitman reports that criminals used cryptography during prohibition to communicate with each other.

Additionally, it is important to mention the recently popularized "windtalkers." The Navajo's used their own language as a basis for cryptography (2005). The code was never broken and was instrumental in the victory in the Pacific Theatre during WWII. An argument could be made that the spoken language was not technically cryptography, but it should be noted that at every communication, the message was written down as a matter of procedure.

In modern times, the public key method of cryptography has seen wide adoption. The use of a common public key and a private key held only by the sender is in use today as a form of asymmetric encryption; one of the uses of this method is for the sender to use the private key to encrypt the message and then anyone who receives the message uses the public key to decipher it. In this way, the receiver knows who the message had to come from.

This method makes up the backbone of the Digital Signature. Problems arise when communications between multiple organizations require the use of many public keys and knowing when to use which one. No matter which method is used, a combination of methods applied one after the other will give the best result (Whitman, 2005).

In conclusion, it is somewhat surprising how limited the history of this very important topic is. No doubt cryptography and in a greater sense, cryptology, has played an enormous role in the shaping and development of many societies and cultures. While history may paint a different picture, the fact that the winners often write history is worth noting. If an army has a strong weapon that was instrumental in providing information that led to success, how apt are they to reveal it in the records of the wars? Instead, it may seem better to have idolized heroes than to reveal the cloak and dagger methods that actually led to success.

Crpytography, by its very nature, suggests secrecy and misdirection; therefore, the fact that the history of this topic is short and somewhat inaccessible is of no great surprise. Perhaps it is itself coded in what is has already been written.

2.9 APPLICATION OF CRYPTOGRAPHY

2.91 Secrecy in Transmission

Most current secrecy systems for transmission use a private key system for transforming transmitted information because it is the fastest method that operates with reasonable assurance and low overhead.

If the number of communicating parties is small, key distribution is done periodically with a courier service and key maintenance is based on physical security of the keys over the period of use and destruction after new keys are distributed.

If the number of parties is large, electronic key distribution is usually used. Historically, key distribution was done with a special key-distribution-key (also known as a master-key) maintained by all parties in secrecy over a longer period of time than the keys used for a particular transaction. The "session-key" is generated at random either by one of the parties or by a trusted third party and distributed using the master-key.

The problem with master-key systems is that if the master-key is successfully attacked, the entire system collapses. Similarly, if any of the parties under a given master-key decides to attack the system, they can forge or intercept all messages throughout the entire system. Many complex private-key systems for reducing some of these problems have been proposed and used for various applications.

With the advent of public-key systems, secrecy can be maintained without a common master-key or a large number of keys. Instead, if Bob wants to communicate with Alice, Bob sends Alice a session-key encrypted with Alice's public key. Alice decrypts the session-key and uses that over the period of the transaction.

These are examples of cryptographic protocols, methods for communicating while attaining a particular cryptographic objective. These protocols are used primarily to deal with key management and system misuse problems. Many other protocols are applied to eliminate other attacks on these systems.

2.92 Secrecy in Storage

Secrecy in storage is usually maintained by a one-key system where the user provides the key to the computer at the beginning of a session, and the system then takes care of encryption and decryption throughout the course of normal use. As an example, many hardware devices are available for personal computers to automatically encrypt all information stored on disk. When the computer is turned on, the user must supply a key to the encryption hardware. The information cannot be read meaningfully without this key, so even if the disk is stolen, the information on it will not be useable.

Secrecy in storage has its problems. If the user forgets a key, all of the information encrypted with it becomes permanently unuseable. The information

is only encrypted while in storage, not when in use by the user. This leaves a major hole for the attacker. If the encryption and decryption are done in software, or if the key is stored somewhere in the system, the system may be circumvented by an attacker. Backups of encrypted information are often stored in plaintext because the encryption mechanism is only applied to certain devices.

2.93 Integrity in Transmission

Many of the users of communication systems are not as much concerned about secrecy as about integrity. In an electronic funds transfer, the amount sent from one account to another is often public knowledge. What the bank cares about is that only proper transfers can take place. If an active tapper could introduce a false transfer, funds would be moved illicitly. An error in a single bit could literally cause millions of dollars to be erroneously credited or debited. Cryptographic techniques are widely used to assure that intentional or accidental modification of transmitted information does not cause erroneous actions to take place.

A typical technique for assuring integrity is to perform a checksum of the information being transmitted and transmit the checksum in encrypted form. Once the information and encrypted checksum are received, the information is again check summed and compared to the transmitted checksum after decryption. If the checksums agree, there is a high probability that the message

is unaltered. Unfortunately, this scheme is too simple to be of practical value as it is easily forged. The problem is that the checksum of the original message is immediately apparent and a plaintext message with an identical checksum can be easily forged. Designing strong cryptographic checksums is therefore important to the assurance of integrity in systems of this sort.

The key distribution problem in a one-key system is as before, but an interesting alternative is presented by the use of public keys. If we generate a single public-key for the entire system and throw away the private key that would go with it, we can make the checksum impossible to decrypt. In order to verify the original message, we simply generate a new checksum, encrypt with the public key, and verify that the encrypted checksum matches. This is known as a one-way function because it is hard to invert.

Actual systems of this sort use high quality cryptographic checksums and complex key distribution and maintenance protocols, but there is a trend towards the use of public keys for key maintenance.

2.94 Integrity in Storage

Integrity against random noise has been the subject of much study in the fields of fault tolerant computing and coding theory, but only recently has the need for integrity of stored information against intentional attack become a matter for cryptography.

The major mean of assuring integrity of stored information has historically been access control. Access control includes systems of locks and keys, guards, and other mechanisms of a physical or logical nature. The recent advent of computer viruses has changed this to a significant degree, and the use of cryptographic checksums for assuring the integrity of stored information is now becoming widespread.

As in the case of integrity in transmission, a cryptographic checksum is produced and compared to expectations, but storage media tends to have different properties than transmission media. Transmitted information is typically more widely available over a shorter period of time, used for a relatively low volume of information, and accessed at a slower rate than stored information. These parameters cause different tradeoffs in how cryptosystems are used.

2.95 Authentication of Identity

Authenticating the identity of individuals or systems to each other has been a problem for a very long time. Simple passwords have been used for thousands of years to prove identity. More complex protocols such as sequences of keywords exchanged between sets of parties are often shown in the movies or on television. Cryptography is closely linked to the theory and practice of using passwords, and modern systems often use strong cryptographic transforms in

conjunction with physical properties of individuals and shared secrets to provide highly reliable authentication of identity.

Determining good passwords falls into the field known as key selection. In essence, a password can be thought of as a key to a cryptosystem that allows encryption and decryption of everything that the password allows access to. In fact, password systems have been implemented in exactly this way in some commercial products.

The selection of keys has historically been a cause of cryptosystem failure. Although we know from Shannon that H(K) is maximized for a key chosen with an equal probability of each possible value (i.e. at random), in practice when people choose keys, they choose them to make them easy to remember, and therefore not at random. This is most dramatically demonstrated in the poor selection that people make of passwords.

On many systems, passwords are stored in encrypted form with read access available to all so that programs wishing to check passwords needn't be run by privileged users. A side benefit is that the plaintext passwords don't appear anywhere in the system, so an accidental leak of information doesn't compromise system wide protection.

A typical algorithm for transforming any string into an encrypted password is designed so that it takes 10 or more msec/transformation to encode a string. By simple calculation, if only capital letters were allowed in a

password, it would take .26 seconds to check all the one letter passwords, 6.76 seconds to check all the 2 letter passwords, 4570 seconds for the 4 letter passwords, and by the time we got to 8 letter passwords, it would take about 2*10**9 seconds (24169 days, over 66 years).

For passwords allowing lower case letters, numbers, and special symbols, this goes up considerably. Studies over the years have consistently indicated that key selection by those without a knowledge of protection is very poor. In a recent study, 21% of the users on a computer system had 1 character passwords, with up to 85% having passwords of 1/2 the maximum allowable length, and 92% having passwords of 4 characters or less. These results are quite typical, and dramatically demonstrate that 92% of all passwords could be guessed on a typical system in just over an hour.

Several suggestions for getting unpredictable uniform random numbers include the use of low order bits of Geiger counter counts, the use of the time between entries at a keyboard, low order bits of the amount of light in a room as measured by a light sensitive diode, noisy diode output, the last digit of the first phone number on a given page of a telephone book, and digits from transcendental numbers such as Pi.

2.96 Credentialing Systems

A credential is typically a document that introduces one party to another by referencing a commonly known trusted party. For example, when credit is

applied for, references are usualy requested. The credit of the references is checked and they are contacted to determine the creditworthiness of the applicant. Credit cards are often used to credential an individual to attain further credit cards. A driver's license is a form of credential, as is a passport.

Electronic credentials are designed to allow the credence of a claim to be verified electronically. Although no purely electronic credentialing systems are in widespread use at this time, many such systems are being integrated into the smart-card systems in widespread use in Europe. A smart-card is simply a credit-card shaped computer that performs cryptographic functions and stores secret information. When used in conjunction with other devices and systems, it allows a wide variety of cryptographic applications to be performed with relative ease of use to the consumer.

A typical cryptographic protocol failure is encountered in the use of the RSA. It seems that if an attacker can choose the plaintext to be signed under an RSA signature system, observe the result of the signature, and then iterate the process, it is possible to get the signer to reveal the private key in a very small number of signatures (about 1 signature per bit of the key). Thus an unmodified RSA signature system requires a sound protocol to be safely used.

2.97 Electronic Signatures

Electronic signatures, like their physical counterparts, are a means of providing a legally binding transaction between two or more parties. To be as

useful as a physical signature, electronic signatures must be at least as hard to forge, at least as easy to use, and accepted in a court of law as binding upon all parties to the transaction.

The need for these electronic signatures is especially accute in business dealings wherein the parties to a contract are not in the same physical vacinity. For example, in an international sale of an airplane, signatures are typically required from two bankers, two companies, two insurance agencies, two attorneys, two governments, and often several other parties. The contracts are hundreds of pages long, and signatures must be attained within a relatively shoert period of time on the same physical document. Faxcimily signatures are not legally binding in all jurisdictions, and the sheer length of a document precludes all parties reading the current copy as they meet at the table to affix signatures. Under current law, all parties must meet in one location in order to complete the transaction. In a transatlantic sale, $100,000 in costs can easily be encurred in such a meeting.

An effort in Europe is currently underway to replace physical signatures with electronic signatures based on the RSA cryptosystem. If this effort succeeds, it will allow many millions of dollars to be saved, and launch the era of digital signatures into full scale motion. It will also create a large savings for those who use the system, and therefore act to force others to participate in order to remain competitive.

2.98 Electronic Cash

There are patents under force throughout the world today to allow electronic information to replace cash money for financial transactions between individuals. Such a system involves using cryptography to keep the assets of nations in electronic form. Clearly the ability to forge such a system would allow national economies to be destroyed in an instant. The pressure for integrity in such a system is staggaring.

2.99 Threshold Systems

Thresholding systems are systems designed to allow use only if a minimal number of parties agree to said use. For example, in a nuclear arms situation, you might want a system wherein three out of five members of the Joint Chiefs of Staff agree. In a banking situation, a safe might only be opened if 4 out of the authorized 23 people allowed to open the safe were present. Such systems preclude a single individual acting alone, while allowing many of the parties to a transaction to be absent without the transaction being halted.

Most threshold systems are based on encryption with keys which are distributed in parts. The most common technique for partitioning a key into parts is to form the key as the solution to N equations in N unknowns. If independent equations are known, the key can be determined by solving the simultaneous equations. If less than N equations are known, the key can be any value since there is still an independent variable in the equations. Any number

can be chosen for N and equations can be held by separate individuals. The same general concept can be used to form arbitrary combinations of key requirements by forming ORs and ANDs of encryptions using different sets of keys for different combinations of key holders. The major difficulties with such a system lie in the key distribution problem and the large number of keys necessary to achieve arbitrary key holder combinations.

2.910 Systems Using Changing Keys

Shannon has shown us that given enough reuse of a key, it can eventually be determined. It is thus common practice to regularly change keys to limit the exposure due to successful attack on any given key. A common misconception is that changing a key much more often than the average time required to break the cryptosystem, provides an increased margin of safety.

If we assume the key is chosen at random, and that the attacker can check a given percentage of the keys before a key change is made, it is only a matter of time before one of the keys checked by the attacker happens to correspond to one of the random keys. If the attacker chooses keys to attack at random without replacement over the period of key usage, and begins again at the beginning of each period, it is 50% likely that a currently valid key will be found by the time required to try 50% of the total number of keys, regardless of key changes. Thus if a PC could try all the DES keys in 10 years, it would be 50% likely that a successful attack could be launched in 5 years of effort. The real benefit of key

changes is that the time over which a broken key is useful is limited to the time till the next key change. This is called limiting the exposure from a stolen key.

2.10 STEGANOGRAPHY

A plaintext message may be hidden in one of two ways. The methods of **steganography** conceal the existence of the message, whereas the methods of cryptography render the message unintelligible to outsiders by various transformations of the text. A simple form of steganography, but one that is time-consuming to construct, is one in which an arrangement of words or letters within an apparently innocuous text spells out the real message. Various other techniques have been used historically; some examples are the following:

- **Character marking:** Selected letters of printed or typewritten text are overwritten in pencil. The marks are ordinarily not visible unless the paper is held at an angle to bright light.

- **Invisible ink:** A number of substances can be used for writing but leave no visible trace until heat or some chemical is applied to the paper.

Pin punctures: Small pin punctures on selected letters are ordinarily not visible unless the paper is held up in front of a light.

• **Typewriter correction ribbon:** Used between lines typed with a black ribbon, the results of typing with the correction tape are visible only under a strong light.

Although these techniques may seem archaic, they have contemporary equivalents. A proposes hiding a message by using the least significant bits of frames on a CD. For example, the Kodak Photo CD format's maximum resolution is 2048-3072 pixels, with each pixel containing 24 bits of RGB color information. The least significant bit of each 24-bit pixel can be changed without greatly affecting the quality of the image. The result is that you can hide a 2.3-megabyte message in a single digital snapshot. There are now a number of software packages available that take this type of approach to steganography.

2.101 Problem of Steganography

Steganography has a number of drawbacks when compared to encryption. It requires a lot of overhead to hide a relatively few bits of information, although using a scheme like that proposed in the preceding paragraph may make it more effective. Also, once the system is discovered, it becomes virtually

worthless. This problem, too, can be overcome if the insertion method depends on some sort of key.

Alternatively, a message can be first encrypted and then hidden using steganography. The advantage of steganography is that it can be employed by parties who have something to lose should the fact of their secret communication (not necessarily the content) be discovered. Encryption flags traffic as important or secret or may identify the sender or receiver as someone with something to hide.

Steganography is also the art of hiding the fact that communication is taking place, by hiding information in other information. It is the art of concealing a message in a cover without leaving a remarkable track on the original message.

It Pronounced "ste-g&-'nä-gr&-fE" and Derived from Greek roots

"Steganos" = covere "Graphie" = writing

Its ancient origins can be traced back to 440 BC. In *Histories* the Greek historian Herodotus writes of a nobleman, Histaeus, who used steganography first time.

The goal of Steganography is to mask the very presence of communication making the true message not discernible to the observer. As steganography has very close to cryptography and its applications, we can with

advantage highlight the main differences. Cryptography is about concealing the content of the message. At the same time encrypted data package is itself evidence of the existence of valuable information. Steganography goes a step further and makes the ciphertext invisible to unauthorized users.

The goal of Steganography is to mask the very presence of communication making the true message not discernible to the observer. As steganography has very close to cryptography and its applications, we can with advantage highlight the main differences. Cryptography is about concealing the content of the message. At the same time encrypted data package is itself evidence of the existence of valuable information. Steganography goes a step further and makes the ciphertext invisible to unauthorized users. Two other technologies that are closely related to steganography are watermarking and fingerprinting. These technologies are mainly concerned with the protection of intellectual property. But steganography is concern with the hiding of text in another information like image, text, audio, video.

2.102 Type of steganography*:* There are 4 different types of steganography:

a. Image

b. Audio

c. Video

d. Protocol

2.103 Text steganography using digital files is not used very often since text files have a very small amount of redundant data.

Audio/Video steganography is very complex in use. **Image steganography** is widely use for hiding process of data. Because it is quite simple and secure way to transfer the information over the internet. Image steganography has following types:

a. Transform domain

i. Jpeg

b. Spread spectrum

ii. Patch work

a. Image domain

i. LSB and MSB in BMP

ii. LSB and MSB in JPG

It is most efficient (in term of data hiding) method of image steganography. Because the intensity of image is only change by 1 or 0 after hiding the information. Change in intensity is either 0 or 1 because the change at last bit.

Disadvantage of LSB
We noticed that in the approach discussed above, the time taken for generating the random numbers depends on the size of the key. In our approach it means that it also depends on the cover-image size.

Although the LSB embedding methods hide data in such a way that the humans do not perceive it, such schemes can be easily destroyed by an opponent such as using lossy compression algorithms or a filtering process.

Any process that modifies the values of some pixels, either directly or indirectly, may result in degrading of the quality of the original object.

2.11 A brief history of steganography

The first description of the use of steganography dates back to the Greeks. Herodotus tells how a message was passed to the Greeks about Xerses' hostile intentions underneath the wax of a writing tablet, and describes a technique of dotting successive letters in a cover text with a secret ink, due to Aeneas the Tactician.

Pirate legends tell of the practice of tattooing a secret information, such as a map, on the head of someone, so that the hair would conceal it.

Kahn tells of a trick used in China of embedding a code ideogram at a prearranged position in a dispatch; a similar idea led to the grille system used in medieval Europe, where a wooden template would be placed over a seemingly innocuous text, highlighting an embedded secret message.

During WWII the grille method or some variants were used by spies. In the same period, the Germans developed microdot technology, which prints a clear, good quality photograph shrinking it to the size of a dot.

There are rumors that during the 1980's Margareth Thatcher, then Prime Minister in UK, became so irritated about press leaks of cabinet documents, that she had the word processors programmed to encode the identity of the writer in the word spacing, thus being able to trace the disloyal ministers.

During the "Cold War" period, US and USSR wanted to hide their sensors in the enemy's facilities. These devices had to send data to their nations, without being spotted.

Today, steganography is researched both for legal and illegal reasons. Among the first ones there is war telecommunications, which use spread spectrum or meteor scatter radio in order to conceal both the message and its source.

In the industry market, with the advent of digital communications and storage, one of the most important issues is copyright enforcement, so digital

watermarking techniques are being developed to restrict the use of copyrighted data.

Another important use is to embed data about medical images, so that there are no problems with matching patient's records and images.

Among illegal ones is the practice of hiding strongly-encrypted data to avoid controls by cryptography export laws.

CHAPTER THREE

3.0 SYSTEM DESIGN

3.1 Cryptography Fundamentals

3.11 Mathematics

Cryptography is nothing more than applied mathematics. You don't have to trust other people to tell you that something is secure. You can know it for yourself.

12. Inverse functions

To prove my identity, I can generate two functions. One function will be the inverse of the other. That means that if I were to compute a value with one function, I could compute the original input with the inverse.

I can use this to prove my identity like this. I give you one function, but I keep the inverse to myself. Then you can make up a secret and run it through the function I gave you. Only I, with my inverse function, could turn it back into the original secret.

13. Modulo arithmetic

Have you ever used a secret decoder ring? It maps each of the 26 letters of the alphabet to another letter. The mapping depends upon how far you turn the ring. But the ring only has 26 settings. Once you go past the last setting, you wrap around to the beginning.

Let's number those settings starting at zero. Setting 0 is the way the ring came out of the box. Move it one space, and you are at setting 1. Move it another space and you are at setting 2. Keep going. Once you get to setting 25, you've gone through all 26 settings. If you move one more after that, you'll be back at 0.

That operation is called addition modulo 26. At each step, we added one within the modulus of 26. The only numbers in our universe are 0-25. All of our mathematical operations have to map back to that very limited set of numbers.

We can do not just addition in a modulus, but also subtraction, multiplication, and exponentiation. But we can't do division. That's because more than one number could lead to any given result.

Suppose we wanted multiply 2 times 24 modulo 26. 2 times 24 is usually 48, but in modulo 26, we can't count that high. So we subtract 26 to bring it back in range. That gives us 48-26, or 22.

So what about 2 times 11 modulo 26? Shouldn't that also be 22? Yes, in fact, it is.

So that's why we can't divide in modulo math. What would 22 divided by 2 give us? 11? 24? Both are equally valid.

14. Prime Numbers

The RSA and Elliptic Curve asymmetric algorithms are based on prime numbers. These numbers have interesting properties that make them well suited to cryptography.

15. Factorization

A prime number is a number that has no factors other than one and itself. That means that you can't multiply two smaller whole numbers to get a prime. If you think of multiplication as the process of building a number, primes are the atoms.

The number 1 (contrary to some early literature) is not a prime. Negative numbers are not primes. We're just talking about whole numbers 2 and up.

If you wanted to build the number 24, you could put together a 4 and a 6. You could say the number 24 is made up of these two parts. But the number 6 is also made up of two smaller numbers: 2 and 3. So if you broke a 24 into pieces, you would have a 2, a 3, and a 4.

The number 4 can also be broken into smaller pieces. It's made up of two 2's. That's right. Just because both of the 2's are the same number, there are still two of them. So the 4 is composed of these two pieces.

Now we've broken the number 24 down into the pieces 2, 2, 2, and 3. Three 2's and one 3. Can we break it down any further? No. There are no smaller numbers that you can multiply to get a 2 or a 3. So these are the prime factors of 24.

It turns out that factorization is a hard problem. We can't *prove* that it's hard, but we think that it is. In all our searching, we haven't found a nice fast way to factor large numbers. We pretty much just have to try all the possible primes to see if we can divide.

16. Fermat's Little Theorem

You may recall the name Fermat from a theorem that made the news. It was called Fermat's Last Theorem. The French mathematician wrote this therem in the margins of his notes, but did not provide a proof. According to his notation, it seems he believed that it had a short elegant proof. But as it turns out it was much harder to prove than he probably thought.

Well, this is not that theorem.

Fermat's Little Theorem is one that he proved during his career, and is much easier to understand.

The theorem states that the following equation is true for any prime number p, and any whole number g that is not a multiple of p.

$g^{p-1} = 1 \pmod{p}$

That is, if you were to multiply a small number with itself over and over again, and constrain the result to a prime modulus, then you would get back to 1 just before you had multiplied p-1 times.

For example, take the prime number 23. We're going to multiply the number 5 by itself within that modulus. 5 times 5 is 25, but 25 is greater than 23. So we

subtract 23 to bring it back into range, and we get 2. Now multiply 2 by 5 to get 10. Then 10 times 5 is 50. Subtract to get it back into range, and 50 - 23 is 27. Still not in range, so 27 - 23 is 4.

If we keep going, we get the numbers 20, 8, 17, 16, 11, 9, 22, 18, 21, 13, 19, 3, 15, 6, 7, 12, and 14. Multiply 14 by 5 and you get 70, which is 23 * 3 + 1. In other words, 70 is 1, in modulus 23.

So if we multiply 22 times, we get back to 1. Then the sequence starts over again. That means we've had enough time to hit all of the numbers between 1 and 22. We didn't hit 0, because if we had we would have stayed there. So this gives us a mapping. I can map any number 1 through 22 to any other number 1 through 22. This mathematical function works like a substitution cipher, but only if I know that I hit all the numbers.

And that's why we will use prime numbers for cryptography. They help us build a large substitution table by simply raising a number to an exponent. And you can multiply two large primes to get an even larger composite, but other people will have a hard time factoring that composite back into the original two primes.

2. Fundamental set of cryptograph Algorithms

The fundamental set of cryptograph algotithms can be divided into three groups:

- Symmetric
- Asymmetric
- Hash functions

21. Symmetric Algorithms

Symmetric algorithms encrypt and decrypt a message using the same key. If you hold a key, you can exchange messages with anybody else holding the same key. It is a shared secret. But be careful who you give the key to. Once it gets in the wrong hands, there is no getting it back. That person can read all of your past messages, and create new messages that are indistinguishable from valid data.

Several symmetric algorithms have been used in the past. These include:

- Blowfish
- DES
- 3DES (Triple DES)
- AES

22. Asymmetric Algorithms

Asymmetric algorithms use a different key to encrypt than they do to decrypt. The encrypting key is called the *public key* and the decrypting key is the *private key*. If you hold the private key, I can send you a message that only you can read.

These keys will also work in the opposite direction. That is, anything you encrypt with your private key, I can decrypt with your public key. You can use this to digitally sign a document. Encrypt it with your private key, and I'll be

able to verify your signature by decrypting with your public key. I have confidence that the message came from you, because only someone who holds your private key could have produced a working signature.

There are three asymmetric algorithms in use today:

- Diffie-Hellman
- RSA
- Elliptic Curve

Diffie-Hellman is not quite suitable for establishing identity as describe above, but the other two are. RSA is the most common today, but Elliptic Curve appears to be on its way to becoming the next standard.

23. Hash Functions

An asymmetric algorithm is limited in the size of message that it can encrypt and decrypt. It can't be run over a large message the way that a symmetric algorithm can. So if I wan to use an asymmetric algorithm to sign a message, I have to first compute a digest, a smaller number based on the larger message. The way I do that is to run a hash function.

Some hash functions were invented for error detection during transmission. These hash functions are not suitable for digital signitures because they are easily reversible. Instead, we have devised cryptographically secure hash functions, which produce hashes that are hard to reverse. In other words, given a

hash, it's hard to make up a document that computes that hash. These hash functions include:

- MD5
- SHA 1
- The SHA 2 family (SHA-128, SHA-192, and SHA-256)
- The SHA 3 family

MD5 has been found to contain weaknesses, and is therefore no longer recommended for use. SHA 1 is somewhat stronger, but should still be phased out at this time. SHA 2 is secure, but was invented by the NSA. SHA 3 is secure, and was invented using an open selection process.

24. AES

AES can be run with a 128, 192, or 256 bit key. Each variation uses a slightly different key schedule and set of rounds to encrypt blocks of a message.

25. Blocks

AES, like Blowfish and DES before it, is a block cypher. It operates not on the entire message, but instead on 16-byte blocks of the message. If a message is not evenly divisible by 16 bytes, then it is padded with random numbers until it is. Each block is run through the cipher on its own.

To diffuse the message throughout the ciphertext, the blocks are not simply processed independently. A bit of information is carried forward from one block

to the next in a mode of operation called Cipher Block Chaining (CBC). The ciphertext of one block is XORed with the plaintext of the next block before it is encrypted. That way, a small change in the message toward the beginning is diffused throughout the rest of the message.

To prevent messages that begin with the same plaintext from having the same ciphertext at the beginning, the first block is XORed with a random set of bytes. These are called the Initialization Vector (IV). The recipient needs the IV to decrypt the message, but it is not part of the key. It can be transmitted in the clear with no loss of confidentiality.

26. Rounds

The key is not simply applied to the block in one operation. Instead, it is applied iteratively in a set of rounds. The number of rounds depends upon the key length, and is between 10 and 14.

Each round include the following four steps:

- Key expansion
- XORing the key with the block
- Substitution
- Shifting rows
- Mixing columns

Shifting rows and mixing columns are designed to confuse the effects of the key on the message. As opposed to simply XORing, these operations make sure that a small change in the key produces a large change in the ciphertext.

Key expansion is the process of running the key schedule, as described next.

27. Key schedule

The key schedule is an algorithm that computes a round key from the symmetric key. Since the round key is XORed with the 16-byte block, it must also be 16-bytes (or 128 bits). But the symmetric key can be as much as twice that length. So the key schedule produces only 16 bytes for each round, even when the whole key is bigger than that.

The AES key schedule is called "key expansion" because it produces more bits in the combined round keys than there are in the symetric key. But it's important to recognize that this process does not add any information, or entropy, to the system. There is no more key material in the set of round keys than there was in the original symmetric key. There are just more bits. The whole point of key expansion is to further confuse the key within the ciphertext.

The AES key schedule includes the following operations:

- XOR
- Shifting bits
- Multiplication in a Galois field

These operations make sure that a small change in the key results in a large change in the ciphertext, thus making AES resilient to differential cryptanalysis.

3. Cryptosystems

By combining algorithms from these three groups, you can create a cryptosystem. This is a protocol for communicating with both confidentiality and authenticity.

To achieve confidentiality, make up a random symmetric key. Encrypt your message with that key. Then encrypt the key using the recipient's public key. The recipient can then decrypt first the symmetric key, and then the message. Only they will be able to do so, provided that their private key is kept secret.

To achieve authenticity, compute a digest of your message using a cryptographically secure hash function. Encrypt this digest using your private key to produce the signature. When the recipient receives the message, they will be able to compute the digest themselves, and then decrypt your signature with your public key. If the answers are the same, then they have confidence that the message came from you and was not altered, provided that you've kept your private key secret.

This is the basis of all digital encryption today. Understanding the algorithms will help you choose the right ones, and to construct a secure Cryptosystem.

3.4 PYTHAGOREAN TRIPLE

3.41 Definition

The longest side of the triangle is called the "hypotenuse", so the formal definition is:

In a right angled triangle: the square of the hypotenuse is equal to

the sum of the squares of the other two sides.

If we know the lengths of **two sides** of a right angled triangle, we can find the length of the **third side**.

A "Pythagorean Triple" is a set of positive integers, **a**, **b** and **c** that fits the rule:

$a^2 + b^2 = c^2$

Example: The smallest Pythagorean Triple is 3, 4 and 5.

Let's check it:

$3^2 + 4^2 = 5^2$

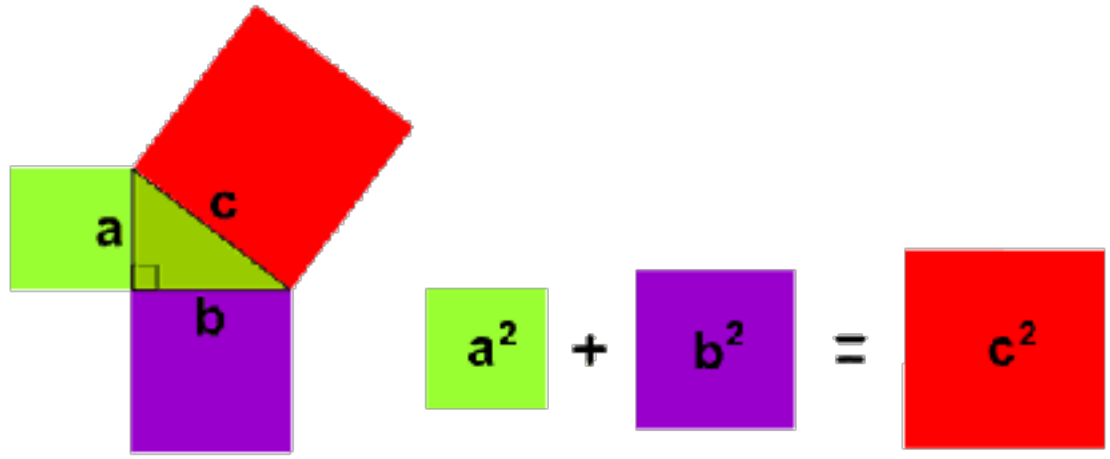
Calculating this becomes:

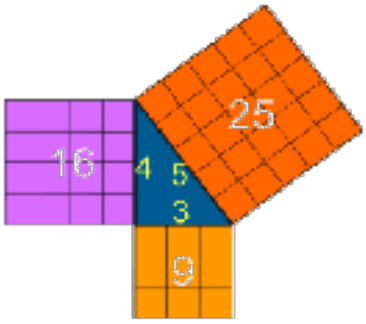
$9 + 16 = 25$

And that is true

3.42 Triangles

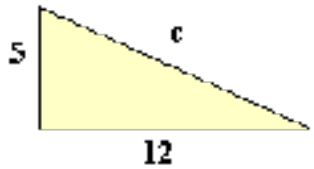
And when you make a triangle with sides **a**, **b** and **c** it will be a right angled triangle:





Example above: The Pythagorean Triple of 3, 4 and 5 makes a Right Angled Triangle and each triangle has a right angle!

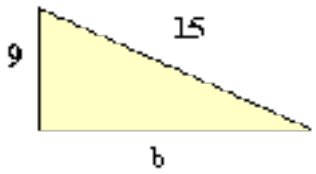
Example 2:



$$a^2 + b^2 = c^2$$
$$5^2 + 12^2 = c^2$$
$$25 + 144 = c^2$$
$$169 = c^2$$
$$c^2 = 169$$
$$c = \sqrt{169}$$
$$\mathbf{c = 13}$$

example 3:

$a^2 + b^2 = c^2$
$9^2 + b^2 = 15^2$
$81 + b^2 = 225$
Take 81 from both sides:
$b^2 = 144$
$b = \sqrt{144}$
b = 12



4. 3 List of the First Few

Here is a list of the first few Pythagorean Triples (**not** including "scaled up" versions mentioned below):

(3,4,5)	(5,12,13)	(7,24,25)	(8,15,17)	(9,40,41)
(11,60,61)	(12,35,37)	(13,84,85)	(15,112,113)	(16,63,65)
(17,144,145)	(19,180,181)	(20,21,29)	(20,99,101)	(21,220,221)
(23,264,265)	(24,143,145)	(25,312,313)	(27,364,365)	(28,45,53)
(28,195,197)	(29,420,421)	(31,480,481)	(32,255,257)	(33,56,65)
(33,544,545)	(35,612,613)	(36,77,85)	(36,323,325)	(37,684,685)

44. Scale Them Up

The simplest way to create further Pythagorean Triples is to scale up a set of triples.

Example: scale 3,4,5 by 2 gives 6,8,10

Which also fits the formula $a^2 + b^2 = c^2$:

$6^2 + 8^2 = 10^2$

$36 + 64 = 100$

3.5 Data Encryption and Decryption Using New Pythagorean Triple Algorithm

A Pythagorean triple represent an ordered triple of the type $(x,y,z) \in Z^3$ such that:

$x^2 + y^2 = z^2$

The conventional way of interpretation of the above mentioned equation is that there is one solution (x_1,y_1,z_1) to the aforementioned equation.

There are many ways of generating Pythagorean triples. One of the most known methods is the Euclids formula which is a fundamental formula for Pythagorean triples for given arbitrary pair of positive integers p and q where $p > q$. The formula states that the integers derived from Euclids formula as given below:

$x = p^2 - q^2$

$y = 2pq$

$z = p^2 + q^2$

represent a Pythagorean triple.

Another approach for generating Pythagorean triples lies in Newtons method which is based on the identity:

$(p^2 - q^2)^2 + (2pq)^2 \equiv (p^2 + q^2)^2$

From the identity it is clearly visible that integer solutions to the equation $x^2 + y^2 = z^2$ are of the form:

$x = d(p^2 - q^2)$, $y = 2dxy$, $z = d(p^2 + q^2)$ with $p > q > 0$.

Where $(p, q) = 1$, p and q are of opposite parity (one even and one odd) and $(x, y, z) = d$. It can be also proved that every Pythagorean Triple can be written in this way so it is essentially useful to observe these x, y and z values. If $d = 1$ the triples are considered to be Primitive. The above mentioned equations cab be extended by at least one (in special cases by two) other solutions to Pythagorean Triples.

The rest of this project section is structured as follows: In section 2 elaboration in detail the derivation of two new solutions to Pythagorean Triples, section 3 illustrates a symmetric cryptosystem based on the newly generated Pythagorean Triples.

3.51 New Pythagorean Triple Algorithm

Let us have $x^2 + y^2 = z^2$ and $gcd(x, y) = 1$. There is a number z so that:

$$z = x + u$$

$$z = y + v \tag{1}$$

where $gcd(x, u) = 1$ and $gcd(y, v) = 1$. As a consequence, from the last system of equations, we have:

$$x + u = y + v$$

$$x - v = y - u$$

Let us mark $y - u = x - v = \lambda$, then:

$$x = v + \lambda$$

$$y = u + \lambda \tag{2}$$

If we replace x in equation 1 from 2 we get:

$$z = u + v + \lambda \tag{3}$$

Equations 2 and 3 given as:

$$x = v + \lambda$$

$y = u + \lambda$

$z = u + v + \lambda$ (4)

represent the new fundamental solutions to the Pythagorean theorem. If we replace these expressions in $x^2 + y^2 = z^2$ we will get:

$(u + \lambda^2) + (v + \lambda^2) = (u + v + \lambda)^2$

from which, after further extension, we have:

$\lambda^2 = 2vu$ (5)

Values of v and u will be selected that way so that they determine λ, out of which we derive the Pythagorean

fundamental solutions:

$v = 2p^2$

$u = q^2, \qquad v > u, \; gcd(p, q) = 1$ (6)

If u and v are replaced in 5 we get:

$\lambda^2 = 4p^2q^2$

and then:

$\lambda = \pm 2pq$ (7)

If now 6 and 7 are replaced in 4 we have:

$x = 2p2 \pm 2pq$

$y = q2 \pm 2pq$

$z = 2p2 + q2 \pm 2pq$ (8)

From the conventional definition of Pythagorean triple, it results that only one fundamental solution (*x, y, z*) exists for *p* and *q* (one of which is odd and the other even).

Based on 8, the previous definition is re-defined to: for any numbers *p* and *q* (one of which is odd and the other even) there are at least two fundamental solutions (*x*1, *y*1, *z*1) and (*x*2, *y*2, *z*2) which can be expressed in the form of New Pythagorean Triple formulas:

There is a special case for numbers *p* and *q*, when we get three fundamental solutions: A short list of solutions for the Pythagorean triples for values (3, 1), (5, 3), (7, 2) and (7, 4) is illustrated as in Table 1. Table 2 illustrates the number encodings of the English alphabet which will be used further on for encryption / decryption purposes.

This will be elaborated in the following section

$x_1 = 2p^2 + 2pq$ $x_2 = 2p2 - 2pq$

$y_1 = q^2 + 2pq$ $y_2 = q2 - 2pq$

$z_1 = 2p^2 + q^2 + 2pq$ $z_2 = 2p^2 + q^2 - 2pq$

$$x_3 = 2pq$$

$$y_3 = p^2 - q^2$$

$$z_3 = p^2 + q^2$$

Table 1: New Pythagorean Triple Algorithm

p	q	x_1	y_1	z_1
3	1	24	7	25
5	3	80	39	89
7	2	126	32	130
7	4	154	72	170
p	q	x_2	y_2	z_2
3	1	12	-5	13
5	3	20	-21	29
7	2	70	-24	74
7	4	42	-40	58
p	q	x_3	y_3	z_3

3	1	6	8	10
5	3	30	16	34
7	2	28	45	53
7	4	56	33	65

Table 2: The English Alphabet

a	b	c	d	e	f	g
0	1	2	3	4	5	6
h	i	j	k	l	m	n
7	8	9	10	11	12	13
o	p	q	R	s	t	u
14	15	16	17	18	19	20
v	w	x	y	z		
21	22	23	24	25		

3.52 Data Encryption and Decryption

We will now see how we can encrypt and decrypt a text by using the New Pythagorean Triple Algorithm formulas for creating the key. Let us mark with m the plaintext, whereas with $\boldsymbol{k}$ the key and with $\boldsymbol{c}$ encrypted message (cipher text). If we want to encrypt a message, we will use the formula:

$$\boldsymbol{c = m + k(mod 26)}$$

If we want to decrypt a message, we use:

$$\boldsymbol{m = c - k(mod 26)}$$

Now showing how the key is going to be created.

Numbers p and q are put within the New Pythagorean Triple Algorithm formulas given below to create the key.

After we have found the values:

$x_1 = 2p^2 + 2pq$ $x_2 = 2p2 - 2pq$

$y_1 = q^2 + 2pq$ $y_2 = q2 - 2pq$

$z_1 = 2p^2 + q^2 + 2pq$ $z_2 = 2p^2 + q^2 - 2pq$

$$x_3 = 2pq$$

$$y_3 = p^2 - q^2$$

$$z_3 = p^2 + q^2$$

$(x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)(mod 26)$

We can freely create the encryption key in the form:

$x_1, y_1, z_1, x_2, y_2, z_2, x_3, y_3, z_3$

For example: If we have a plaintext, *South East European University* (as given in Table 3)

Table 3

S	O	u	t	h	e	a	s	t	e	u
18	14	20	19	7	4	0	18	19	4	20
R	O	p	e	a	n	u	n	i	v	e
17	14	15	4	0	13	20	13	8	21	4
R	S	i	t	y						
17	18	8	19	24						

Which we want to encrypt, we will have to use numbers $p = 5$ and $q = 3$, and use them in the New Pythagorean

Triple algorithm formulas:

$$x_1 = 2 \cdot 5^2 + 2 \cdot 5 \cdot 3 = 80$$

$$y_1 = 3^2 + 2 \cdot 5 \cdot 3 = 39$$

$$z_1 = 2 \cdot 5^2 + 3^2 + 2 \cdot 5 \cdot 3 = 89$$

$$x_2 = 2 \cdot 5^2 - 2 \cdot 5 \cdot 3 = 20$$

$$y_2 = 3^2 - 2 \cdot 5 \cdot 3 = -21$$

$$z_2 = 2 \cdot 5^2 + 3^2 - 2 \cdot 5 \cdot 3 = 29$$

$$x_3 = 2 \cdot 5 \cdot 3 = 30$$

$$y_3 = 5^2 - 3^2 = 16$$

$$z_3 = 5^2 + 3^2 = 34$$

After we have found these values:

$$(80,39,89,20,-21,29,30,16,34)(mod 26)$$

we get the key:

(2,13,11,20,5,3,4,16,8)

The message is now being converted into numbers. In order to convert each letter of the text into numbers, we use Table II.As a result, we get values as in table 4.

Table 4: Message Encryption

S	O	U	t	h	e	a	s	t	e	u
18	14	20	19	7	4	0	18	19	4	20
2	13	11	20	5	3	4	16	8	2	13
20	1	5	13	12	7	4	8	1	6	7
U	B	F	N	M	H	E	I	B	G	H
r	O	P	e	a	n	u	n	i	v	e

17	14	15	4	0	13	20	13	8	21	4
11	20	5	3	4	16	8	2	13	11	20
2	8	20	7	4	3	2	15	21	6	24
C	I	U	H	E	D	C	P	V	G	Y
r	S	I	t	y						
17	18	8	19	24						
5	3	4	16	8						
22	21	12	9	6						
W	V	M	J	G						

The person whom we want to send the encrypted message to, needs to have the pair of number $(p,q) = (5,3)$. The received message can now be decrypted, by finding the key.

Based on the New Pythagorean Triple algorithm formulas, we find the key values:

$$x_1 = 2\cdot 5^2 + 2\cdot 5\cdot 3 = 80$$

$$y_1 = 3^2 + 2\cdot 5\cdot 3 = 39$$

$$z_1 = 2\cdot 5^2 + 3^2 + 2\cdot 5\cdot 3 = 89$$

$$x_2 = 2\cdot 5^2 - 2\cdot 5\cdot 3 = 20$$

$$y_2 = 3^2 - 2\cdot 5\cdot 3 = -21$$

$$z_2 = 2\cdot 5^2 + 3^2 - 2\cdot 5\cdot 3 = 29$$

$$x_3 = 2\cdot 5\cdot 3 = 30$$

$$y_3 = 5^2 - 3^2 = 16$$

$$z_3 = 5^2 + 3^2 = 34$$

After we have found these values:

$$(80,39,89,20,-21,29,30,16,34)(mod 26)$$

we get the key:

$$(2,13,11,20,5,3,4,16,8)$$

Once the key has been created, it is quite easy to decrypt the encrypted message, by using the formula:

$$m = c - k(mod 26)$$

The decrypted message is given as in table 5.

Table 5: Message Decryption

U	B	F	N	M	H	E	I	B	G	H
20	1	5	13	12	7	4	8	1	6	7

2	13	11	20	5	3	4	16	8	2	13
18	14	20	19	7	4	0	18	19	4	20
s	O	U	t	h	e	a	s	t	e	u
C	I	U	H	E	D	C	P	V	G	Y
2	8	20	7	4	3	2	15	21	6	24
11	20	5	3	4	16	8	2	13	11	20
17	14	15	4	0	13	20	13	8	21	4
r	O	P	e	a	n	u	n	i	v	e
W	V	M	J	G						
22	21	12	9	6						
5	3	4	16	8						
17	18	8	19	24						
r	S	I	t	y						

Based on the New Pythagorean Triple algorithm formulas, for any numbers p and q (one of which is odd and the other even) there are at least two fundamental solutions (x_1,y_1,z_1) and (x_2,y_2,z_2), but there are also some special cases when we can even get three fundamental solutions (x_1,y_1,z_1),(x_2,y_2,z_2) and (x_3,y_3,z_3). This algorithm can also be used for creating the key for data encryption and decryption.

3.53 ADVANTAGE OF ALGORITHM-

a. This algorithm use random size of key.

b. Because of this random size the middle person can't predict the size of key and data**.**

c. The number of times execution of loop is not fixed so that more secure algorithm.

d. This is more secure and easy to implement.

3.54 DISADVANTAGE OF ALGORITHM-

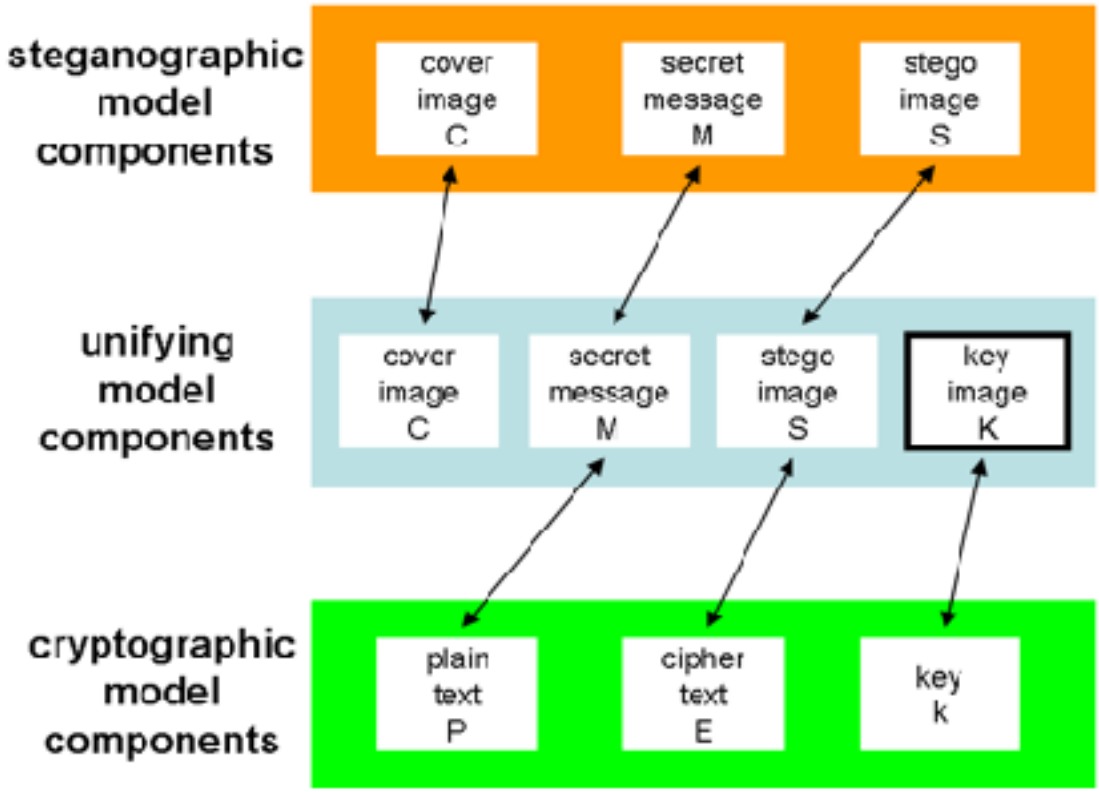
a. Key distribution.

3.55 COMPLEXITY OF ALGORITHM-

Complexity of algorithm is depend on size of key and text it is approximately equal to O(mn) where m and n is size of key and text respectively.

A STEGO-CRYPTOGRAPHIC MODEL

Discussing how we could unify two models, in order to devise a new model holding the features that are peculiar both to the steganographic and to the cryptographic model.



Mapping between model components.

The mapping between *P* and *M*, *E* and *S*, and *k* and *K* is possible because we can consider all the components as bit sequences and then realize a relation between the co-respective bit sets.

The unifying model results as a steganographic one with the addition of a new element: the *key image K*. It gives the unifying model the cryptographic functionality we are searching for, preserving its steganographic nature.

The unifying model embedding process yields *S* exploiting not only *C*'s bits but also *K*'s ones this way of proceeding gives the chance to embed the secret message *M* (that is, the recover *M* through *S* and *K*. In addition, it will be

difficult to detect that *M* is embedded in *S* nor be able to access the content of the secret message

PIXEL PROCESSING

After the converting our information in secret code or encrypted form we need to patch that data in the image. We use least significant bit for the patching of data because of following reason.

a. Because the intensity of image is only change by 1 or 0 after hiding the information.

b. Change in intensity is either 0 or 1 because the change at last bit .e.g.

1111100**0** 11111001

The change is only one bit so that the intensity of image is not effected too much and we can easily transfer the data

Steps To Insert Data In Image *:-*

a. Take an input image.

b. Find out the pixel values.

c. Select the pixel on which we want to insert data.

This process of selection of pixel is done as user's choice he may choose pixel continuous or alternate or at a fixed distance.

i. Insert the data values in pixels eg.

For example a grid for 3 pixels of a 24-bit image can be as follows: 00101101 00011100 11011100 10100110 11000100 00001100 11010010 10101101 01100011 When the number **200**, which binary representation is **11001000**, is embedded into the least significant bits of this part of the image, the resulting grid is as follows:

00101101 00011101 11011100

10100110 11000101 00001101

11010010 10101100 01100011