INTRODUCTION

Steganography comes from the Greek and literally means, *"Covered writing"*. It is one of various data hiding techniques, which aims at transmitting a message on a channel where some other kind of information is already being transmitted. This distinguishes steganography from covert channel techniques, which instead of trying to transmit data between two entities that were unconnected before.

The goal of steganography is to hide messages inside other “harmless” messages in a way that does not allow any “enemy” to even detect that there is a second secret message present. The only missing information for the “enemy” is the short easily exchangeable random number sequence, the secret key, without the secret key, the “enemy” should not have the slightest chance of even becoming suspicious that on an observed communication channel, hidden communication might take place.

Steganography is closely related to the problem of “hidden channels” n secure operating system design, a term which refers to all communication paths that cannot easily be restricted by access control mechanisms. In an ideal world we would all be able to sent openly encrypted mail or files to each other with no fear of reprisals. However there are often cases when this is possible, either because the working company does not allow encrypted email or the local government does not approve of encrypt communication (a reality in some parts of the world). This is where steganography can come into play.

Data hiding techniques can also be classified with respect to the extraction process:

- *Cover Escrow* methods need both the original piece of information and the encoded one in order to extract the embedded data.
- *Blind or Oblivious* schemes can recover the hidden message by means only of the encoded data.

Steganography has developed a lot in recent years, because digital techniques allow new ways of hiding informations inside other informations, and this can be valuable in a lot of situations. The first to employ hidden communications techniques -with radio transmissions- were the armies, because of the strategic importance of secure communication and the need to conceal the source as much as possible.

Nowadays, new constraints in using strong encryption for messages are added by international laws, so if two peers want to use it, they can resort in hiding the communication into casual looking data. This problem has become more and more important just in these days, after the international Wassenaar agreement, with which around thirty of the major - with respect to technology - countries in the world decided to apply restrictions in cryptography export similar to the US's ones.

Another application of steganography is the protection of sensitive data. A file system can be hidden in random looking files in a hard disk, needing a key to extract the original files. This can protect from physical attacks to people in order to get their passwords, because maybe the attacker can't even know that some files are in that disk.

The major concern of steganography is stealth, because if an attacker, either passive or active, can detect the presence of the

message, from that point he can try to extract it and, if encrypted, to decrypt it. The resistance to attempt at destruction or noise is not required, since we consider the sender and the receiver equally interested in exchanging messages, so that they will try to transmit the stego-medium in the best way they can. If the stego-data can be transmitted over the selected channel, and this is usually the case with all the media that are used, like images or sounds, then the embedded data will be preserved along with them. Thus, data hiding techniques for steganography must focus on the maximum strength against detection and extraction.

As a second request, we would prefer a high data rate, because we will usually want to be able to exchange any amount of data, from simple messages to top secret images.

STEGANOGRAPHY AND CRYPTOGRAPHY

Steganography and cryptography are cousins in spy-craft family. Cryptography scrambles a message so it cannot be understood. Steganography hides the message so it cannot be seen. A message in cipher text for instance might arouse suspicion on the part of the recipient while an “invisible” message created with steganographic methods will not.

In this way, we can say that steganography completes cryptography, and actually there are usually two ciphers to break when trying to extract the embedded message: one is the one with which the message was embedded, and the other is the one with which the message was enciphered.

Some history

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 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.

Invisible ink offered a common form of invisible writing. Early in WWII, steganographic technology consisted almost exclusively of these inks. With invisible ink, a seemingly innocent letter could contain a very different message written between the lines.

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

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

SOME DEFINITIONS

We give some definitions common to the steganography field:

Cover medium: This is the medium in which we want to hide data, it can be an innocent looking piece of information for steganography, or some important medium that must be protected for copyright or integrity reasons.

Embedded message: This is the hidden message we want to put in the cover. It can be some data for steganography and some copyright informations or added content for digital watermarking.

Stegokey: This is represented by some secret information, which is needed in order to extract the embedded message from the stego-medium

Stego-medium: This is the final piece of information that the casual observer can see.

We can define this simple formula:

Cover-medium + embedded-message = stego-message

BASIC METHOD BEHIND STEGANOGRAPHY

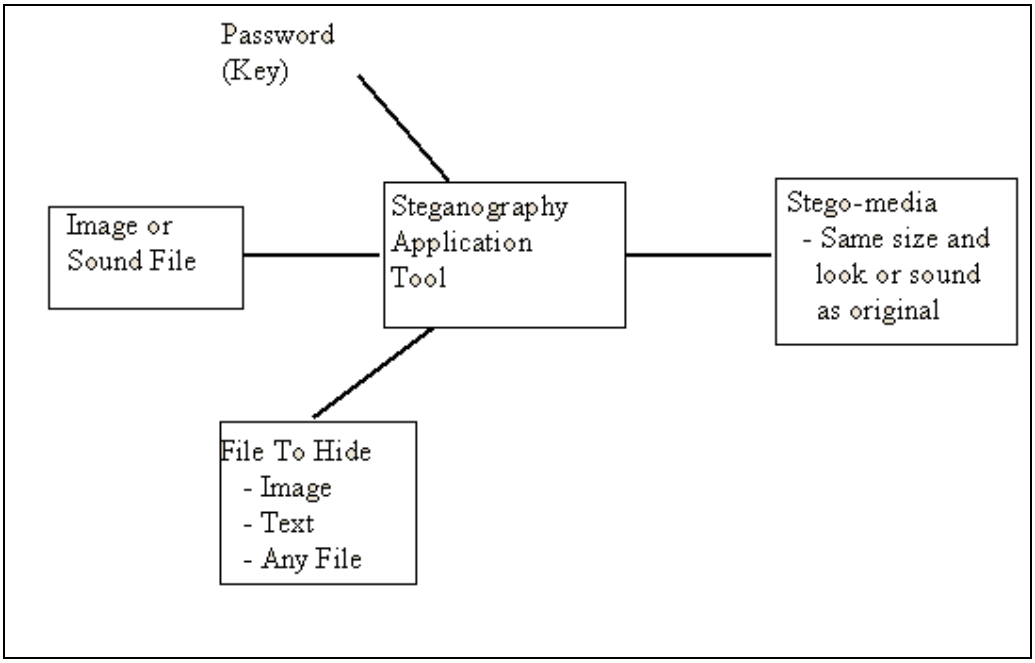

IMAGE FILES

To a computer, an image is an array of numbers that represent an array of numbers that represent light intensities at various points or pixels. These pixels make up the image's raster data. A common image size is 640 * 480 and 256 colors (or 8 bits per pixel). Such an image could contain about 300 kb of data.

Digital images ate typically stored as either 24-bit or 8-bit files. A 24-bit image provides the most space for hiding information, however, it can be quite large except for the JPEG images. A 24-bit image of 1,024 pixels width and 768 pixels height has more than two million pixels, each having 24-bits, which would produce a file exceeding 2 Mega bytes. Such a file would attract attention during transmission. File compression would thus be beneficial, if not necessary, to transmit such a file.

File compression

There are two types of file compression methods- lossless and lossy. Both methods save storage space but have different results, interfer9ing with the hidden information, when information is uncompressed. Lossless compression lets us reconstruct the original message exactly; therefore it is preferred when the original information must remain intact (as with steganographic images). Lossless compression us typical of mages saved as GIF and 8-bit BMP.

Lossless compression, on the other hand, saves space but may not maintain the original image's integrity. This method typifies

images saved as JPEG. Due to the lossy compression algorithm, which we discuss later, the JPEG formats provide close approximations to high-quality digital photographs but not an exact duplicate. Hence the term “lossy compression”.

Embedding data

Embedding data, which is to be hidden, into an image requires two files. The first is the innocent looking image that will hold the hidden information, called the *cover image*. The second file is the message- the information to be hidden. A message may be plain text, cipher text, other images, or anything that can be embedded in a bit stream, when combined, the cover image and the embedded message make a *stego- image*. A *stego-key* (a type of password) may also be used to hide, and then later decode, the message.

Most steganographic software neither supports not recommends using JPEG mages. But recommends instead the use of lossless 24-bit images such as BMP. The next best alternative to 24-bit images is 256- color or gray scale images. The most common of these found on the Internet are GIF files.

In 8-bit color images such as GIF files, each pixel is represented by a single byte, and each pixel nearly points to a color index table (a palette) with 256 possible colors. The pixels value is between 0 and 255. The software simply paints the indicated color on the screen at the selected pixel position.

Many steganography experts recommend the use of images featuring 256 shades of grapy. Gray scale images are preferred because the shades change very gradually from byte to byte, and the less the value changes between palette entries, the better they can hide information.

When considering an image in which to hide information, you must consider the image as well as the palette. Obviously, an image with large areas of solid colors is a poor choice, as variances created from the embedded massage will be noticeable in the solid areas.

CONCEALMENT IN DIGITAL IMAGES

Information can be hidden many different ways in images. To hide information, straight message insertion may encode every bit of information in the message or selectively embed the message in "noisy" areas that draw less attention- those areas where there is a great deal of natural color variation. The message may also be scattered randomly throughout the image. Redundant pattern encoding "wallpapers" the cover image with the message.

A number of ways exist to hide information in digital images. Common approaches include:

- Least significant bit (LSB) insertion.
- Masking and filtering.
- Algorithms and transformations.

Least significant bit insertion

Least significant Bit insertion is a common, simple approach to embedding information in a cover file. To hide an image in the LSBs of each byte of a 24-bit image, you can store 3 bits in each pixel. A 1,024 * 768 image has the potential to hide a total of 2,359,296 bits of information. Ti you compress the message to be hidden before you embed it, you can hide a large amount of information. To the human eye, the resulting stego-image will look identical to the cover image.

Masking and Filtering

Masking and filtering techniques, usually restricted to 24-bit and tray-scale images, hide information by marking an image, in a manner similar to paper watermarks. Watermarking techniques may be applied without fear of image destruction due to lossy compression because they are more integrated into the image.

Visible watermarks are not steganography by definition. The difference is primarily one of intent. Traditional steganography conceals information; watermarks extend information and become an attribute of the cover image. Digital watermarks may include such information as copyright, ownership, or license. In steganography, the object of communication is the hidden message. In digital watermarking, the object of communication is the cover.

To create a watermarked image, we increase the luminance of the masked area by 15 percent. If we were to change the luminance by a smaller percentage, the mask would be undetected by the human eye. Now we can use the watermarked image to hide plaintext or encoded information.

Masking is more robust than LSB insertion with respect to compression, cropping and some image processing. Masking techniques embed information in more significant areas so that the hidden message is more integral to the cover image than just hiding it in the “noise” level. This makes it more suitable than LSB with lossy JPEG images.

Algorithms and Transformations

LSB manipulation is a quick and easy way to hide information but is vulnerable to small changes resulting from image processing or lossy compression. Such compression is a key advantage that JPEG images have over other formats. High quality images can be stored in relatively small files using JPEG compression method.

One steganographic method that integrates the compression algorithm for hiding the information is Jpeg-Jsteg.

Jpeg-Jsteg creates a JPEG stego image from the input of a message to be hidden and a lossless cover mage.

Another method used in *Patchwork* and similar techniques is the *redundant pattern encoding.* Here the hidden information is scattered throughout the cover mage. These approaches may help protect against image processing such as cropping and rotations and they hide information more thoroughly than by simply masking. They also support image manipulation more readily than tools that rely on LSB. In using *redundant pattern encoding*, you must trade off message size against robustness. A large message may be embedded only once because it would occupy a much greater portion of the image area.

Other techniques *encrypt and scatter* the hidden data throughout an image. Scattering the message makes it appear more like noise. Proponents of this approach assume that even if the message bits are extracted, they will be useless without the algorithm and stego-key to decode them.

Scattering and encryption helps protect against hidden message extraction but not against message destruction through image processing. A scattered message in the image’s LSBs is still as

vulnerable to destruction from lossy compression and mage processing, as is a clear text message inserted n the LSBs.

LEAST SIGNIFICANT BIT (LSB) INSERTION

Technique basics

Today, when converting an analog image to digital format, we usually choose between three different ways of representing colors:

- 24-bit color: every pixel can have one in 2^24 colors, and these are represented as different quantities of three basic colors: red (R), green (G), blue (B), given by 8 bits (256 values) each.

- 8-bit color: every pixel can have one in 256 (2^8) colors, chosen from a palette, or a table of colors.

- 8-bit gray-scale: every pixel can have one in 256 (2^8) shades of gray.

LSB insertion modifies the LSBs of each color in 24-bit images, or the LSBs of the 8-bit value for 8-bit images.

Data Rate

The most basic of LSBs insertion for 24-bit pictures inserts 3 bits/pixel. Since every pixel is 24 bits, we can hide

3 hidden-bits/pixel / 24 data-bits/pixel = 1/8 hidden-bits/data-bits.

So for this case we hide 1 bit of the embedded message for every 8 bits of the cover image.

If we pushed the insertion to include the second LSBs, the formula would change to:

6 hidden-bits/pixel / 24 data-bits/pixel = 2/8 hidden-bits/data-bits

And we would hide 2 bits of the embedded message for every 8 bits of the cover image.

Adding a third-bit insertion, we would get:

9 hidden-bits/pixel / 24 data-bits/pixel = 3/8 hidden-bits/data-bits

Acquiring a data rate of 3 embedded bits every 8 bits of the image. The data rate for insertion in 8-bit images is analogous to the 1 LSB insertion in 24-bit images, or 1 embedded bit every 8 cover bits. We can see the problem in another light, and ask how many cover bytes are needed to send an embedded byte.

For 1-LSB insertion in 24-bit images or in 8-bit images this value would be 8/1*8 = 8 Bytes. For 2-LSBs insertion in 24-bit pictures it would be 8/2*8 = 4 Bytes, for 3-LSBs insertion it would be 8/3*8 = 21.33 Bytes.

Robustness

LSB insertion is very vulnerable to a lot of transformations, even the most harmless and usual ones. Lossy compression, e.g. JPEG, is very likely to destroy it completely. The problem is that the "holes" in the Human Visual System that LSB insertion tries to exploit - little

sensitivity to added noise - are the same that lossy compression algorithms rely on to be able to reduce the data rate of images. Geometrical transformations, moving the pixels around and especially displacing them from the original grid, are likely to destroy the embedded message, and the only one that could allow recovery is a simple translation. Any other kind of picture transformation, like blurring or other effects, usually will destroy the hidden data. All in all, LSB insertion is a very little robust technique for data hiding.

Ease of detection/extraction

There is no theoretical outstanding mark of LSB insertion, if not a little increase of background noise.

It's very easy, instead, to extract LSBs even with simple programs, and to check them later to find if they mean something or not.

Suitability for steganography or watermarking

First of all, since it is a so vulnerable technique even for simple processing, LSB insertion is almost useless for digital watermarking, where it must face malicious attempts at its destruction, plus normal transformations like compression/decompression or conversion to analog (printing or visualization)/conversion to digital (scanning).

Its comparatively high data rate can point it as a good technique for steganography, where robustness is not such an important constraint.

Problems and possible solutions

Having stated that LSB insertion is good for steganography, we can try to improve one of its major drawbacks: the ease of extraction. We don't want that a malicious attacker be able to read everything we are sending.

This is usually accomplished with two complementary techniques:

- Encryption of the message, so that who extracts it must also decrypt it before it makes sense.
- Randomizing the placement of the bits using a cryptographical random function (scattering), so that it's almost impossible to rebuild the message without knowing the seed for the random function.

In this way, the message is protected by two different keys, acquiring much more confidentiality than before. This approach protects also the integrity of the message, being much more difficult (we could say at least computationally infeasible) to counterfeit the message.

Anyway, since we don't want our message to be only an encrypted and scrambled message, we must go back to the purpose of making the communication hidden.

The two most important issues in this problem are:

- The choice of images
- The choice of the format (24-bit or 8-bit, compressed or not)

The cover image first of all must seem casual, so it must be chosen between a set of subjects that can have a reason to be

exchanged between the source and the receiver. Then it must have quite varying colors, it must be "noisy", so that the added noise is going to be covered by the already present one. Wide solid-color areas magnify very much any little amount of noise added to them. Second, there is a problem with the file size, which involves the choice of the format. Unusually big files exchanged between two peers, in fact, are likely to arise suspicion.

To solve this problem, it has been studied a modification to the JPEG algorithm that inserts LSBs in some of the lossless stages or pilots the rounding of the coefficients of the DCT used to compress the image to encode the bits. Since we need to have small image file sizes, we should resort in using 8-bit images if we want to communicate using LSB insertion, because their size is more likely to be considered as normal.

The problem with 256 colors images is that they make use of an indexed palette, and changing a LSB means that we switch a pixel from a position to an adjacent one. If there are adjacent contrasting colors in the palette, it can happen that a pixel in the image changes its color abruptly and the hidden message becomes visible.

To solve this problem different methods have been studied, like rearranging the palette so that adjacent colors don’t contrast so much, or even reducing the palette to a smaller number of colors and replicating the same entry in the table in adjacent positions, so that the difference after the embedding of the message is not visible at all. Moreover for most images the reduction of colors from, for instance, 256 to 32 is hardly visible.

Most of the experts, anyway, advise to use 8-bit grayscale images, since their palette is much less varying than the color one, so LSB insertion is going to be very hard to detect by the human eye.

Steganography - Seeing the Unseen

SSIS - SPREAD SPECTRUM IMAGE STEGANOGRAPHY

We point out this technique as an example for spread spectrum data-hiding methods. Spread spectrum techniques are now widely used in military radio communications, due to their very high robustness to detection and extraction.

SSIS is a quite mature process, and its aim is to achieve low detectability, ease of extraction, high data rate and good robustness to removal. It is based on spread spectrum techniques, but it enhances them by adding other encoding steps, acquiring better performance.

Technique basics

The core of SSIS is a spread spectrum encoder. These devices work by modulating a narrow band signal over a carrier. The carrier's frequency is continually shifted using a pseudorandom noise generator fed with a secret key. In this way the spectral energy of the signal is spread over a wide band, thus decreasing its density, usually under the noise level.

To extract the embedded message, the receiver must use the same key and noise generator to tune on the right frequencies and demodulate the original signal. A casual observer won't be able even to detect the hidden communication, since it is under the noise level.

Data Rate

The data rate for this technique can be fairly high, but it depends on the choices made for the different parameters of the encoding. We can assume that the message will be compressed before embedding to allow for a higher capacity. The ECC encoder instead is going to insert redundant data into the stream to be able to correct the errors. The more errors we want to correct, the more bits will be added. Then, we have a tradeoff between good retrieval and capacity. If we can allow for small glitches in the recovered message, then we can use a weaker encoding.

Moreover, the more data we want to insert in the image, the more noise we are going to add to it. Then, if our cover is not noisy, we will be able to hide very little data, while if we choose a noisy one, its capacity will be higher. Experiments with 512x512 grey scale images (256 KB) could embed from 500 bytes to 5KB, depending on the cover. These experiments used a spread spectrum signal powerful enough to give almost total error-free retrieval, because the compression method adopted didn't allow for any errors. This means a data rate varying from 1 hidden-bytes/50 cover-bytes to 10 hidden-bytes/50 cover-bytes, a rate surpassed only by LSB insertion.

Robustness

Spread spectrum techniques are usually quite robust. Every transformation that adds noise to the image isn't able to destroy the message. Anyway, a determined attacker can quite easily compromise the embedded data using some digital processing, like for example

noise reduction filters, the same that are used in decoding to estimate the original cover.

Ease of detection/extraction

Spread spectrum encoding is widely used in military communications for its robustness against detection. An attacker can't usually even know if the message was embedded, and anyway it will be very hard for him to extract it without knowing the right *key2* and *key3*.

Suitability for steganography or watermarking

Due to its fairly high capacity and low ease of detection and extraction, SISS is very good for steganography.

Problems and possible solutions

The basic tradeoff in using SSIS is between the error rate we can afford and the amount of informations we want to embed, that varies in turn the power of the added noise. The ECC is used to allow for a lower power without increasing the Bit Error Rate as well.

Further improvements will deal with improving the original cover estimate stage, so that it'll lead to a lower Bit Error Rate in the recovered signal, allowing to use less redundant ECCs.

The first is redundant encoding by dividing the cover into blocks, and embedding the same message in each of them, so that the hidden data can be extracted even from a part of the image as big as one block, but the more of it we have, the more certain we can be about the result.

Moreover, they added to the spectrum a template that can, through a log-polar transform applied to the spectrum of the stego-image, determine the original scale factor and orientation of the image, rendering the stego-message virtually immune to scaling and rotation.

Finally, spread spectrum techniques can add an adaptive perceptual masking filter before the insertion of the signal, so that the added noise is quite sure to be under perceptual limits. This, however, will increase the error rate in the retrieval, because it reduces the power of the embedded signal.

CONCLUSION

In this paper we tried to give an all-round view of steganography, both used to exchange messages and watermarking. First we gave an outline of the problem, telling also some of the history of this quickly developing field.

Then we showed the different techniques invented, from the simplest to the more complex ones, trying to evaluate them under many points of view. Major emphasis was put on data hiding in images, for the techniques involved are usually more mature than the corresponding ones for other kinds of informations. Image encoding algorithms can also be representative for manipulation of other types of media like voice, text, binary files, binary files, communication channels etc.

Then we gave an outline of the problems involved with watermarking, a field that has come into light after the development of broadband worldwide digital networks.

Steganography and digital watermarking are undergoing a development process similar to that of encryption. Steganography's niche in security is to supplement cryptography and not to replace it. There is a continuous invention of new techniques for steganography followed by successful breakings and new improvements of them.

REFERENCES

- Neil F. Johnson, Sushil Jajodia, George Mason University, *"Exploring Steganography: Seeing the Unseen"*, IEEE Computers, February 1998, pp. 26-34

- W. Bender, D. Gruhl, N. Morimoto, A. Lu, *"Techniques for Data Hiding"* IBM Systems Journal, Vol. 35

- Ross Anderson, Roger Needham, Adi Shamir, *"The Steganographic File System"*, 2nd Information Hiding Workshop, 1998

- Ross J. Anderson, Fabien A.P. Petitcolas, *"On the limits of steganography"*

ABSTRACT

Steganography comes from the Greek and literally means, *"Covered writing"*. It is one of various data hiding techniques, which aims at transmitting a message on a channel where some other kind of information is already being transmitted. This distinguishes steganography from covert channel techniques, which instead of trying to transmit data between two entities that were unconnected before.

The goal of steganography is to hide messages inside other " harmless" messages in a way that does not allow any "enemy" to even detect that there is a second secret message present. The only missing information for the "enemy" is the short easily exchangeable random number sequence, the secret key, without the secret key, the "enemy" should not have the slightest chance of even becoming suspicious that on an observed communication channel, hidden communication might take place.

Steganography is closely related to the problem of "hidden channels" in secure operating system design, a term which refers to all communication paths that cannot easily be restricted by access control mechanisms. In an ideal world we would all be able to sent openly encrypted mail or files to each other with no fear of reprisals. However there are often cases when this is possible, either because the working company does not allow encrypted email or the local government does not approve of encrypt communication. This is where steganography can come into play.

CONTENTS

1. Introduction
2. Steganography and Cryptography
3. A brief history of steganography
4. Some definitions
5. Basic methods behind steganography
6. Image Files
6.1. File compression
6.2. Embedding data
7. Concealment in digital images
7.1. LSB insertion
7.2. Masking and filtering
7.3. Algorithms and transformations
8. Least Significant Bit insertion
8.1. Technique basics
8.2. Data rate
8.3. Robustness
8.4. Ease of detection/extraction
8.5. Suitability for steganography
9. Spread Spectrum Image Steganography
9.1. Technique basics
9.2. Data rate
9.3. Robustness
9.4. Ease of detection/extraction
9.5. Suitability for steganography
10. Conclusion
11. Reference

ACKNOWLEDGEMENT

I extend my sincere thanks to **Prof. P.V.Abdul Hameed,** Head of the Department for providing me with the guidance and facilities for the Seminar.

I express my sincere gratitude to Seminar coordinator **Mr. Berly C.J**, Staff in charge, for their cooperation and guidance for preparing and presenting this seminar.

I also extend my sincere thanks to all other faculty members of Electronics and Communication Department and my friends for their support and encouragement.

ABID PARAPARAMBIL