

SECLUDER
A SECURITY SUITE FOR STEGANIZATION
IN
IMAGE AND AUDIO FILES



Estd-1984



ISO 9001:2000
Certified

P-1039

PROJECT WORK

Submitted by

PRAKASH BABU P. (0028Q0147)
RAJESH KUMAR D. (0028Q0152)
SENTHAMIL KANNAN S. (0028Q0158)
SURESH N.N. (0028Q0164)



Under the guidance of

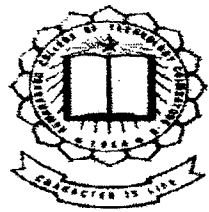
Ms. C.UBELLAH MARIA M.Sc CT

APRIL 2003

In partial fulfillment of the requirements for the award of the degree of
Bachelor of Science in Applied Science -
Computer Technology
of the Bharathiyar University, Coimbatore-641 046 .

DEPARTMENT OF COMPUTER TECHNOLOGY
KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE - 641 006.

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE: 641 006



Estd-1984

Department of Computer Technology

CERTIFICATE



ISO 9001:2000
Certified

This is to certify that this project entitled

**SECLUDER A SECURITY SUITE FOR STEGANIZATION
IN IMAGE AND AUDIO FILES**

has been submitted by

**PRAKASH BABU P., RAJESH KUMAR D., SENTHAMAN
KANNAN S.,
SURESH N.N.**

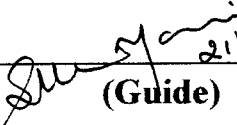
Mr. _____

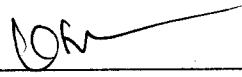
In partial fulfillment of the requirements for the award of degree of

Bachelor of Science Applied Science Computer Technology

of Bharathiyar University, Coimbatore:641 046

during the academic year 2002-2003.

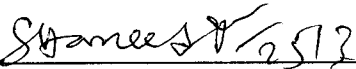

21/3/2003
(Guide)



(Head of the Department)

Certified that the Candidate was Examined by us in the Project Work
Viva-Voice Examination held on 25.03.2003

University Register Number _____

0028Q0147, 0028Q0152, 0028Q0155,
0028Q0164


(Internal Examiner)


(External Examiner)

ACKNOWLEDGEMENT

“Great things are not performed with strength but with prayer and perseverance”. We thank the Almighty for His abundant showers of blessings.

Any project big or small cannot be successful without the support of many people. We would like to place our humble accolades to all the respondents who were directly or indirectly involved in the completion of the project at this juncture.

We would like to extend our heartfelt thanks to our Principal, **Dr.K.K.Padmanabhan** for providing us with good education facilities and to **Dr.V.Sundaram**, Head of the Department, for being a constant source of encouragement throughout the tenure of our course.

We would also like to express our deep sense of gratitude to **Mrs.V.Geetha M.C.A.**, Course Co-ordinator, for the suggestions and encouragement provided. We greatly appreciate the enthusiasm, support and guidance provided by **Mr.S.Hameed Ibrahim M.C.A.**, Class Advisor, at every stage of our project.

Our project guide Miss **C. Ubellah Maria**, M.Sc. C.T., Lecturer rendered genuine support, involvement, appreciation and helped our team to give our very best. We thank her a lot.

We are greatly indebted and thankful to **Mr.V.Sivakumar B.E.**, Lecturer, who provided motivation, timely technical help and invaluable suggestions for better improvisation.

A special note of thanks to all the staff members and non-technical staff of our department who have been indirectly instrumental for accomplishing a task of this magnitude.

Our acknowledgement would be made complete with a word to our beloved parents for their moral support, our family members for their love and care. We owe a lot to all our friends who have been very supportive and have encouraged us to bring out the project to its current state of excellence.

With a profound sense of gratitude and heart filled with ecstasy, we take this opportunity to thank all our well wishers, who have stood by us in times of trouble and without whom successful completion of the project would not have been a dream come true.

SYNOPSIS

Dealing with sensitive information requires a huge rapport in most applications and has paved a way for a new field named steganography. In steganized data transfer, the idea is to hide the message within another large file, through the modification of the pixel code or chunk values in an image or audio file respectively.

The existing tools use only a selected type of file mostly of bitmap files for steganization. An attempt has been made in this project to extend this concept of data hiding for most of the available file types. Through steganized data transfer, one can tell the difference between the original and the loaded file only by comparing them. To view the steganized data, the user should enter the secret code and this will reveal the original file and hidden file separately.

"SECLUDER", which is a freeware steganographic tool with both GUI and command line version is developed. This project combines the explanatory information with the image and audio files, posting secret communication on the web, copyright protection, anonymity and hiding data. The striking feature of this product is that neither the quality of the source file nor the size of the resultant file is altered.

CONTENTS

Acknowledgement

Synopsis

List of Tables

List of Figures

Contents

1. Introduction

1.1 Motivation for the Project	1
1.2 Choice of Steganography	2
1.3 Objectives	2

2. Investigation and Analysis

2.1 Problem Definition	5
2.2 Background investigation	6
2.3 Drawbacks of existing approach	6
2.4 Justification of chosen approach	9
2.5 Choice of appropriate methodologies	10
2.6 Processing Environment	15
2.7 Product Features	18
2.8 Project Plan	19

3. System Design	
3.1 Overall Architecture	22
3.2 Design rules for Steganography	23
3.3 Module Design	25
3.4 User Interface Design	28
3.5 Supporting Diagrams	30
4. System Implementation	
4.1 Project Module	34
5. Testing	
5.1 Test Plans for a bug free product	39
5.2 Evaluation	40
6. Conclusion	44
7. Further Enhancements	48
8. Bibliography	50
9. Appendices	
Appendix A - BMP File Format	53
Appendix B - WAVE File Format	58
Appendix C - Software Requirements Specification	63
Appendix D - Dynamic Linked Libraries (DLL)	73
Appendix E - Secluder - Screen Shots	77

LIST OF TABLES

S.No	Particulars	Chapter	Page No
2.1	Comparison of Existing Techniques	2	12
2.2	Hardware Specification	2	15
2.3	Software Specification	2	15
2.4	Project Plan	2	20
5.2	Test Cases	5	41

LIST OF FIGURES

S. No	Particulars	Chapter	Page No
2.1	Illustration of Hide and Seek	2	7
2.2	Illustration of Stego Dos	2	8
2.3.	Process Overview	2	18
3.1	Stego Image	3	23
3.2.1	User interface flowchart	3	29
3.2.2	Data Hiding flowchart	3	30
3.2.3	Data retrieval flowchart	3	32
A.1	Wave format diagram	App - B	58
S.1	System flow representation	SRS	65
S.2	Supporting diagrams	SRS	68
S.3	Supporting diagram	SRS	69

In God we Trust, all others we monitor –

Bruce Scheiner

CHAPTER 1

1. Introduction

The digital information revolution has brought about insightful changes in our culture and lives. The enormous advantages of digital information have also produced new challenges and opportunities for innovation. Images are used for merely representing photographs and designs. Similarly everyone has a taste for a certain kind of music on the storage device of his /her computer. Also, it is quite common for people to share and transfer different music files among themselves. Digital imaging being one of the most developed branches of Computer Science can serve better purposes other than the above citations. This open source project aims at data hiding in digital images and audio files.

1.1 Motivation for the Project

Security is of prime concern in areas where sensitive data is dealt with. The field of data hiding is definitely a major section in the area of data security and is nowadays most talked about. The project that was taken over has to do with Steganography and its modern applications in data security and data hiding. Steganography belongs to the wide area of data hiding. This is an area of Computer Science that is still in its infancy. Only recently has much work been directed in the area of Steganography.

Other motivations to study steganographic methods are due to governmental restrictions that have been placed on encryption services and people desire a way to send private messages. Moreover it was recently reported that the notorious terrorist, Osama Bin Laden has been using steganographic technology to keep his contacts informed. Hence it would be beneficial to learn about this technology in order to be able to detect and decipher terrorist's hidden messages. These aspects of steganography are what sparked the research into this vast field.

1.2 Choice of Steganography

“**Steganography** is the art and science of communicating in a way which hides the existence of communication. In contrast to **Cryptography**, where the enemy is allowed to detect, intercept and modify messages without being able to violate certain security measures. The goal of **Steganography** is to hide messages inside other harmless messages in a way that does not allow an enemy to even detect that there is a second secret message present”[Markus Kuhn 1995-07-03].

One reason for the surge of interest is that publishing and broadcasting industries have become interested in techniques for hiding encrypted copyright marks and serial number in digital films, audio recordings, books and multimedia products. The new market opportunities due to digital distribution coupled with the fear that digital products could be too easy to copy will lead to the increase in popularity of Steganography. Hence **Steganography** was preferred instead of conventional **Cryptography**.

1.3 Objectives

The field of data hiding is rather broad and complicated. Hence the objectives of the project are limited.

The **core objectives** include:

- **Transparency:** There should be no perceivable changes in the image or audio file after data insertion.
- **Robustness:** The embedded data must survive malicious attacks.
- **Capacity:** The volume of information that is embedded must be high.

The ***secondary objective*** is to bring this technology to the masses, so that anyone can use Steganography to protect their data from intruders.

*Securing a Computer System has traditionally been a
battle of wits, the penetrator tries to find the holes and the
designer tries to close them - M.Gosser*

CHAPTER 2

Investigation and Analysis

Requirement analysis is the first step in software engineering process. It is essential to get a complete understanding of software requirements, which is essential for the success of a software development effort. No matter how well designed or implemented, a poorly analyzed and specified program will disappoint the user and bring frustration to the developer. This section focuses on identification of the problem context and provides a solution strategy to solve the problem.

1 Problem Definition

We have occasionally heard the very popular phrase that the “World is becoming smaller with each passing day”. Looking at it from a geographical viewpoint, it seems to appear nonsensical. However, the underlying fact is that with improving technology, it is becoming increasingly easier and faster to communicate. It is a widespread notion that telecommunications hold the key to the future. However, along with the good, there is always the bad part (hackers). The bad in this case is the misuse of the technology at the expense of others. Communications holds the key to business, personal life, etc. As people tend to rely more on these new means of communication, more and more important information is being conveyed along these new lines.

In order to ensure the privacy of the communication between two parties, data security is booming on the horizon as a potentially massive problem. Various new methods are being developed for this purpose. Cryptographic techniques provide secure channels for communicating entities. However, due to the lack of covertness, it is possible that an eavesdropper can identify encrypted stream through cryptanalysis. Thus unlike cryptography, where the goal is to secure communication from an eavesdropper, a novel technique named Steganography, which strives to hide the very presence of the message itself from an observer is proposed in this project.

2 Background Investigation

The idea of hiding information in an inconspicuous medium is called steganography and comes from the Greek, *stegano* grajein, which literally means "Covered Writing". It is by no means a new idea; there are many examples that can be taken from history when secret messages were sent.

In ancient times, the Greeks used to hide messages underneath the wax of a writing tablet. In Tudor England, when Mary Queen of Scots was imprisoned at Chartley castle, she sent secret messages to the Catholics including the French ambassador, by hiding the letters inside the empty beer barrels that left the castle. Modern legends explain the practice of tattooing secret information such as a map on the forehead of someone, so that the hair would conceal it.

During the Second World War, spies used the grille method or some variants. At the same period the Germans developed Microdot technology, which prints a large, good quality photograph shrinking it to the size of a dot.

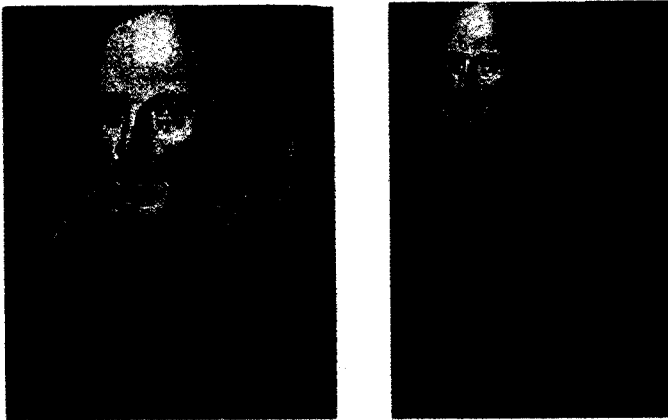
Now, with the electronic age and with the Internet, information can be sent all around the world. However, it is easy to detect. Instead, the only way to hide the fact that you are sending a secret message is to pass it off as something boring and ordinary so it doesn't arouse suspicion.

3 Drawbacks of Existing Approaches

Several Steganography software were downloaded and examined to see how they operate and the results were analyzed. A summary of the advantages and disadvantages of the various software are as follows:

2.3.1 Hide and Seek v4.1

Hide and Seek is a series of DOS programs written by Colin Maroney for hiding data inside GIF images. It works by randomly hiding the data throughout the image thus for a small data file the alterations are sparsely spread over the image. This means that bigger the data file, the more obvious is the change. It also has the limitation that only 19,000 bytes can be stored because the maximum display used is 320 x 480 pixels. This means that if an image is used that is smaller than this, rather than stretching the image, the rest of the area is padded in with black regions.



2.1 A greyscale image before and after having data inserted with "Hide and Seek"

2.3.2 StegoDos

StegoDos is again a series of DOS programs for hiding data. It requires that the image is 320 x 200 pixels and has 256 colours. It also requires a very complicated series of commands to be issued in order to insert and extract the data. It was very difficult to use and trying to get this program to run successfully. This led to the realization of how essential it is to have a user-friendly front-end.

2.3.3 White Noise Storm

Written by Ray (Arsen) Arachelian, this is much easier to use and was able to insert more data into an image than the previous programs. It was not hampered by having a fixed size of image and produced images with relatively little visible degradation.

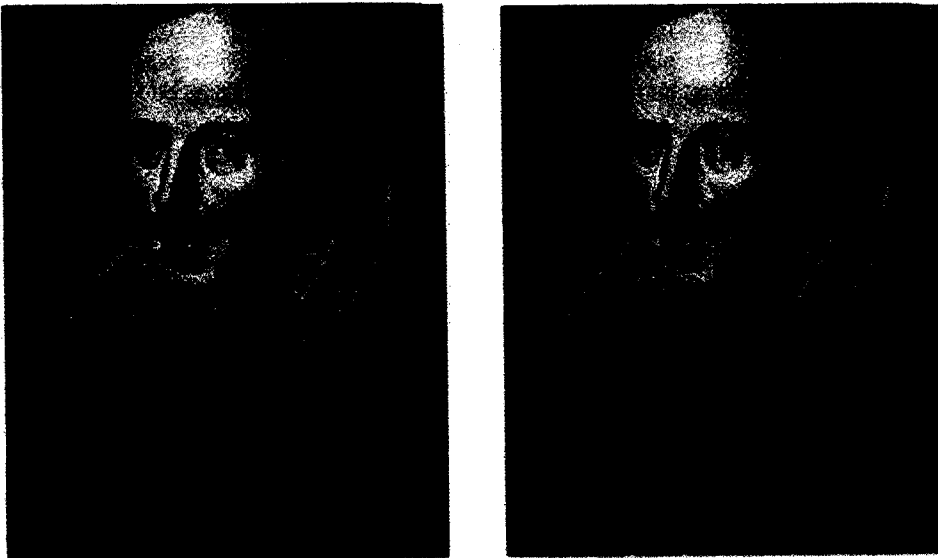


Fig: 2.2 Before and after having data embedded by White Noise Storm.

2.3.4 S-Tools for Windows

This is the best program that was tested. It can store data in numerous ways: inside BMP images, GIF images, WAV audio files and unused parts of a floppy disk. The tested version was 4.0 and found that though the GUI was unhelpful initially, it eventually became quite easy to use and was able to offer a range of possible options. It used a drag-and-drop approach; file icons were dropped onto the "work surface" and then the data file icon dropped onto the image. It used a passphrase for inserting the data and offered several different encryption methods: DES, triple DES, IDEA and MDC. The GUI approach taken by the program was unusual in that it relied totally on mouse input and there was none of the "to be expected features" such as File Open, File Save, etc. and thus it had a strange feel to it. The result was that it took a long time to learn how to use the GUI.

4 Justification of Chosen Approach

Digital Steganography, which is a novel technique to hide sensitive data within an image or sound file, is the proposed approach. In this digital era, steganographic techniques are used in a wide range of applications some of which are listed below:

- Used to combine explanatory information, within an image (like doctor's note accompanying X-rays), so that there are no problems matching records and images.
- Copyright protection

- Embedding corrective audio or image data in case corrosion occurs from a poor connection or transmission.
- Posting secret information on the web to avoid transmission
- War telecommunication, which used spread spectrum or meteor scatter radio in order to conceal both the image and the source.

5 Choice of Appropriate Methodologies

Digital Steganography is based on the following simple formula:

Given an original piece of data D , there is a threshold T below which the person will not spot any changes to the data. T is dependent on the experience of the observer or listener, but there is a minimum tolerance that is beyond the capability of the human senses.

Steganography can be applied to different media such as:

- Still Images
- Moving Images
- Audio Files
- Text Files
- File Systems

While any of the above said types could serve as a cover object, this project focuses on steganographic techniques involving digital still images and audio files.

2.5.1 Data Hiding in Still Images

Steganography, also commonly termed as “data hiding” in images has truly come of age with the invention of fast, powerful computers. These are pieces of data that are most frequently exchanged on the Internet and the methods of data hiding in them are quite mature.

To remain invisible to the eye, the data hiding technique must exploit the “holes” in the human visual system like:

- The masking effect of edges
- The varying sensitivity to contrast as a function of spatial frequency.
- The low sensitivity to small changes in luminance for random patterns
- The low sensitivity to very low spatial frequencies such as continuous change in brightness through an image.

An advantage in using still images for data hiding is that they represent a non-causal medium, since it is possible to access any pixel of the image at random.

2.5.1.1 Analysis of Techniques for Data Hiding in Still Images

The various techniques for data hiding in still images include:

- Least Significant Bit (LSB) insertion
- Spread spectrum image Steganography (SSIS)
- Texture block coding
- Patch work
- Orthogonal Projection Coefficients Manipulation
- Dithering Manipulation
- Perceptual Masking
- DCT coefficient Manipulation

S. No	Name of the Technique	Data Rate	Robustness	Suitable for Steganography
1.	LSB insertion	1/8 Hidden bits/ Data bits	Vulnerable	Suitable
2.	SSIS	1 hidden byte/ 50 Cover bytes to 10 hidden bytes/ 50 cover bytes	Highly Robust	Suitable
3.	Texture Block coding	Very Low	Fairly Robust	Not Suitable
4.	Patch work	Extremely low	Fairly Robust	Not suitable

Table: 2.1 Comparison of Existing Techniques

2.5.1.2 LSB Insertion

This project uses LSB Insertion technique for data hiding in still images. This is due to the fact that its data hide rate is comparatively high. It is a good choice even though it is vulnerable, since robustness is not such an important constraint for Steganography.

2.2 Data Hiding in Audio files

There are various techniques for data hiding in a digital audio. These techniques cannot rely on many of the methods described for images, because of the better performances of the Human Auditory System (**HAS**) with respect to the Human Visual System (**HVS**).

2.2.1 Analysis of Techniques for Data Hiding in Audio files

- Low bit Encoding
- Phase Encoding
- Spread Spectrum
- Echo data hiding

The first analyzed method is the easy and high bit rate LSB insertion technique.

The second technique is based on the IAS sensitivity, only for differential phase variation, but relative insensitivity to initial phase. Thus, the sound file is divided into blocks and each block's initial phase is modified using the embedded message, preserving the subsequent relative phase shifts.

Third, spread spectrum schemes can be used, even if they usually add perceivable noise to the sound. However, the embedded signal can be filtered through a perceptual mask, so that the most audible components of the added noise are reduced in power.

Echo data hiding involves hiding data by adding echo to the audio signal using two different delays to encode bits. Both these delays should be small enough to be heard by the naked ear as enrichments in sound and not as distortions.

Low bit encoding has been chosen as the method of data hiding in audio files.

5.3 Authentication

Having stated that LSB insertion is suitable for Steganography, an attempt has been made to improve one of its major drawbacks: the ease of extraction, by means of authentication. The stego image is protected by means of a password.

5.4 Data Retrieval

Generally data retrieval can be classified into 2 types with respect to the extraction process:

2.5.4.1 Cover escrow method

This method needs both the original piece of information and the encoded one in order to extract the embedded data.

2.5.4.2 Blind oblivious scheme

These schemes can recover the hidden message by means of the encoded data alone. Hence this method is adopted for data retrieval since it is usually impractical to distribute certified copies of the original medium.

6 Processing Environment

2.6.1 Hardware Specification

Particulars	Minimum Requirements
Processor	PIII/ Celeron 533 MHZ
Memory	64 MB RAM
Display Unit	Colour Monitor
Audio Device	Compatible Sound card and Speakers
Hardware	4 GB HDD

Fig: 2.2 Hardware Specifications

2.6.2 Software Specification

Particulars	Minimum Requirements
Operating System	Microsoft Windows 9x
Picture Viewer	MS Paint / ACD see Viewer/ Any picture Viewer
Audio S/w	Win amp

Fig: 2.3 Software Specifications

2.6.3 Developed in

- Turbo C 3.0
- Microsoft Visual C++ 6.0
- Microsoft Visual Basic 6.0
- Help Studio 2000

2.6.3.1 Reasons for choice of Software

2.6.3.1.1 Turbo C 3.0

Turbo C was developed at AT & T's Bell laboratories of USA in 1972. C became popular due to its reliability, simplicity and easiness. C is often regarded as problem oriented language and a machine oriented language. It is called as a middle level language since it possesses a good programming efficiency (as compared to machine oriented languages) and a good machine efficiency (as compared to problem oriented languages).

C is the best choice for programmers who which to find a development solution for projects which especially deals with involving access of low level resources of the system like hard disk and involves traversing of interrupts.

2.6.3.1.2 Microsoft Visual Basic

Microsoft Visual Basic offers a 32 bit application development platform which can be used by developers for Rapid Application Development (RAD). It has built in features for embedding third party software by means of distribution in Active-X and Dynamic Linking Libraries (DLLs).

The distribution of applications made in Visual Basic can be turned easy by supplying it by means of an Executable file (.EXE), which just requires visual Basic Run Time files for execution. The interface with Databases in Visual Basic is just easy to incorporate by means ActiveX Data objects.

2.6.3.1.3 Microsoft Visual C++ 6.0

Microsoft Visual C++ 6.0 offers a great platform for developers who wish to exchange their applications and distribute it. The unique feature of VC++ is the inclusion of Microsoft's Foundation Class Library (MFC) which actually offers the potential to the developers to make use of the objects present in that class library. Hence, the applications developed in Visual C++ can be made to reside as a standalone application.

Besides providing the above features, VC++ also offers the option to the developers to design their own template libraries for use in various developing environments by use of ATL. Also, 16 bit redistributable DLLs can be created in VC++, which can be referenced in a variety of DOS applications.

2.6.3.1.4. Help Studio 2000

Help Studio offers the feasibility for developers to construct interactive help modules that can be called from the application. The unique feature of Help Studio is that it provides publishing of the help files in a wide variety of formats like MS Help ver 3.0, MS Help ver 4.0, compiled HTML file (.chm). The help created can be incorporated into software like Robohelp for creating help for web applications. The help existing in MS Word format can also be imported into Help Studio.

2.7 Product Features

The overall function of the product is presented pictorially below:

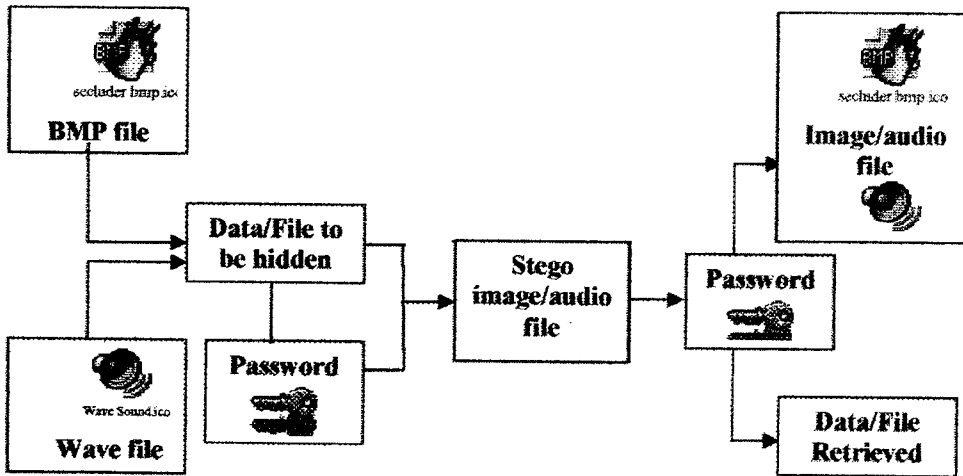


Fig: 2.3 Process Overview

The product has been aptly christened as "**SECLUDER**" which literally means Hiding. There are two versions, one meant for DOS and the other one is WINDOWS based.

2.7.1 Applications

Positive uses of this product are:

➤ **Copyright protection**

It prevents intruders from altering or duplicating a file.

➤ **Extended file Information**

Information can be saved within a file to describe the attributes of the file. For example, an image could have information about people, location and time saved within it.

➤ **Confidential Communication**

The product would aid in communication of confidential information such as credit card numbers, passwords etc.,

2.7.2 Security Aspects

The security provided by SECLUDER is very high due to the following reasons:

- The adversary does not know that there is sensitive data on your PC.
- Even if someone knows that you are hiding data, there is no way to tell where it resides (i.e.,) which image or audio file is infected and which is not.
- Even if someone knows that some image or audio file contains hidden data, it is very difficult to extract it without the password.

2.8 Project Plan

After a thorough analysis of steganography and the various available techniques, a plan was framed with respect to the duration and the activities to be performed. The detailed chart of the project plan has been tabulated below:

Duration	Activity
Week 1	Organize terms and project proposals due
Week 2	Analysis of various file formats, product definition (scope) processing environment, development tools to be decided.
Week 3	Software Requirements Specification (SRS) Preparation
Week 4	Algorithm (for hiding, retrieval, and authentication) designed. Design document to be prepared
Week 5	Implementation of prototype begins.
Week 6	Prototype implementation due (Text Hiding in BMP)
Week 7	Modest version of SECLUDER to be developed for BMP
Week 8	Modest version of SECLUDER to be developed for WAV
Week 9	DLL conversion and user interface design
Week 10	Testing
Week 11	Final version of product due
Week 12	Project document due

Table: 2.4 Project Plan

Thus at the end of requirement analysis, a general statement of the software scope is refined into a concrete specification that becomes the foundation for all software activities to follow.

The result of this phase are documented and provided as Software requirement Specification (SRS) in Appendix C.

*The product of human creativity grows only
arithmetically whereas the capacity to share and distribute
them increases Geometrically*

– Brain Hayes

CHAPTER 3

System Design

Design Phase is the stage, which charts out the system vision and overall architecture. It involves developing a conceptual view of the system, identifying data decomposing high level functions to sub functions, developing concrete data representations and specifying algorithmic details.

Overall Architecture

The proposed system deals with the hiding and retrieval of data in any form (text, text file, image file or audio file) within or from. In case of data hiding, the container and the hidden data together form the stego medium, which is protected by means of a password. The sensitive data as well as the password are embedded in the cover medium by making use of the LSB insertion technique.

The data retrieval process is employed to retrieve the embedded data from the cover medium. The hidden data is retrieved after proper authentication by employing the reverse process of data hiding.

The notable feature of this product is that the quality of the image as well as the size of the image or audio file remains the same even after data hiding. Data hiding in still images and audio files is accomplished by using .bmp files and .wav files respectively as the cover media.

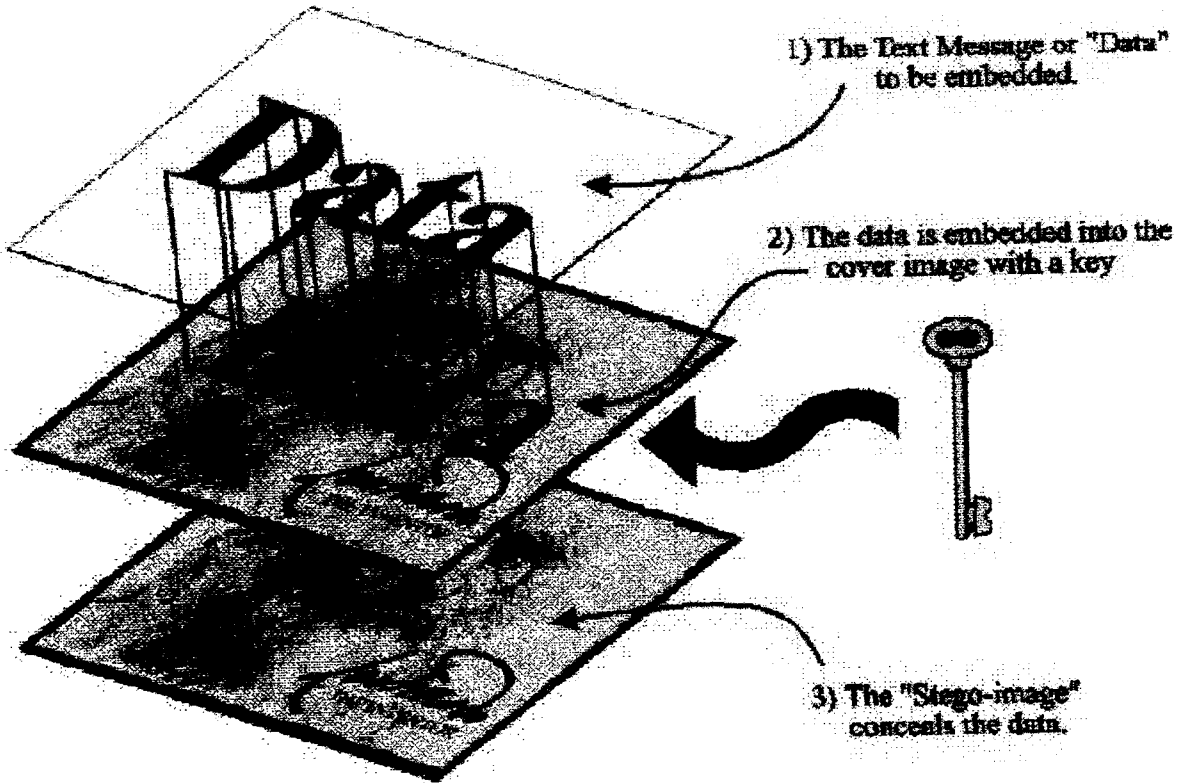


Fig.3.1 Stego Image

3.2 Design Rules for Digital Steganography

A Steganographic system is perfectly secure only if it does not arise suspicion to the malicious observer. The following issues must be considered while choosing the cover medium:

- The choice of the images
- The choice of the format

.2.1 Choice of images

- The cover images must seem casual; hence it must be chosen between a set of subjects that can have a reason to be exchanged between the source and the receiver.

For example: A big exchange of naturalistic pictures between two governmental agencies would be spotted as strange by even a dumb attacker.

- It must have quite varying colours and it must be noisy so that the added noise is going to be covered by the already present one. Wide solid color areas might magnify very much any little amount of noise added to them.
- Unusually big files exchanged between two persons are likely to arise suspicion.
- Commonly known images (Such as Mona Lisa) must be avoided.

3.2.2 Choice of the Format

The image can be either 24-bit color or 8-bit gray scale image with .bmp extension. It is recommended to use 8 bit gray scale images, since the palette is much less varying than the color images and so LSB insertion will be hard to detect by the human eye.

3 MODULE DESIGN

The product consists of two flavours:

One primarily meant for DOS and the other with an elegantly designed graphical User Interface meant for WINDOWS platform.

The various modules that are common in both the versions are:

- Data hiding Module
- Data Retrieval Module
- Authentication Module
- Help Module

In addition to the above mentioned modules, the GUI version also provides a security module.

3.3.1 Module Description

The following section provides a functional description of the modules along with their input and output specifications. The algorithmic details of these modules will be explained in detail in the implementation section of this project thesis.

3.3.1.1 Data Hiding Module

Input

➤ **Source File (Cover Medium)**

It can be a .bmp or a .wav file. The bmp file can either be a 24-bit color image or an 8-bit gray scale image.

➤ **Data to be Hidden**

It can be just text alone or a text file, an image file, an audio file or any other file type.

➤ **The encrypted password from authentication Module**

Function

This module accepts the user input (i.e., cover medium, data and password) and hides it in the LSB of the container. A bit by bit insertion of the data is embedded at a particular frequency within the cover medium.

Output

The stego medium, which is the combination of cover medium and data, is obtained. The size and quality of the resultant remains the same as that of the original medium.

3.3.1.2 Data Retrieval Module

Input

- The Stego file

This is the file into which the retrieved cover medium is stored.

Function

This module first retrieves the password stored and sends it to the authentication module. After proper authentication, the stored bits are retrieved from the specified frequency position from the LSB position and are provided as output to the user.

Output

The data and the cover medium are provided as distinct outputs.

3.3.1.3 Authentication Module

Input

- Password Text

Function

The password is encrypted by means of XOR and NOT function along with the data and sends to the data-hiding module.

Output

- Encrypted text

3.3.1.4 Help Module

This module is a descriptive nature. A novice user can get a clear cut idea about the product features as well as its functional details.

3.3.1.5 Utility Module

It consists of utilities like notepad, calculator, picture viewer and system information.

4 USER INTERFACE DESIGN

Initially the product was designed to work for the DOS platform. The investigation and analysis phase brought to the limelight the disadvantages of DOS based application when utilized by a common user. This led to the design of a user-friendly GUI based product with a full-fledged Help facility.

The design outline of the GUI based (product) version is deposited as follows:

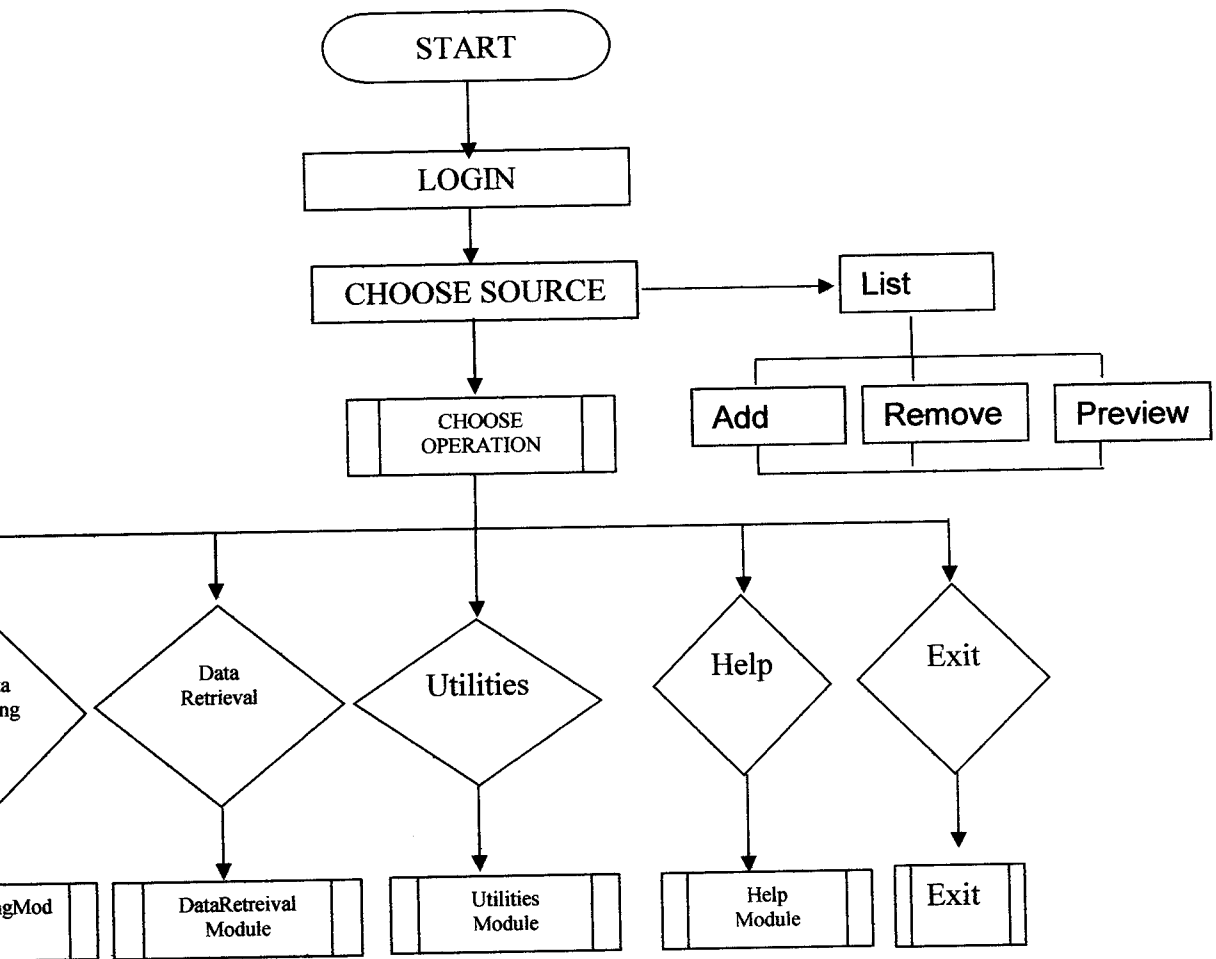
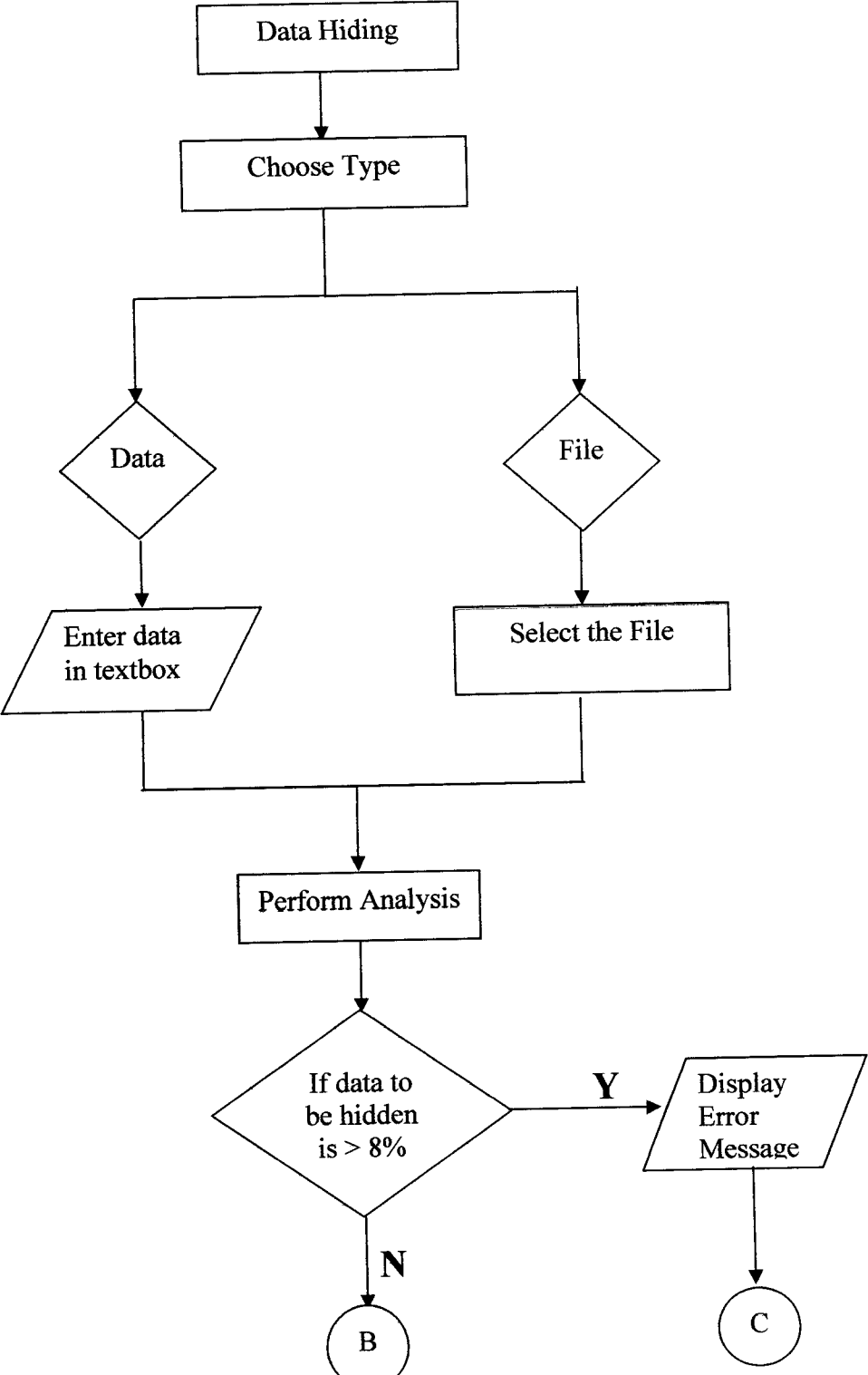


Fig: 3.4 User Interface Flowchart

Supporting Diagrams



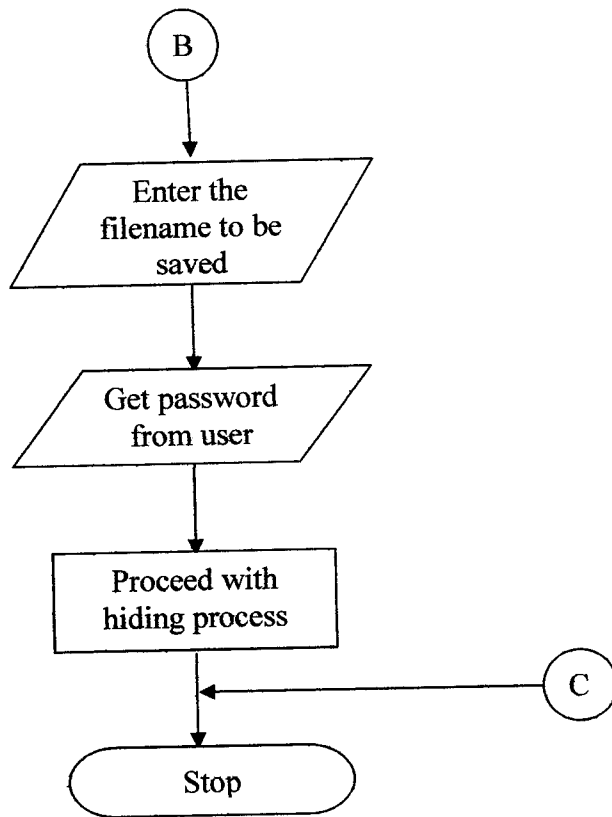


Fig: 3.5.1 Data Hiding Flowchart

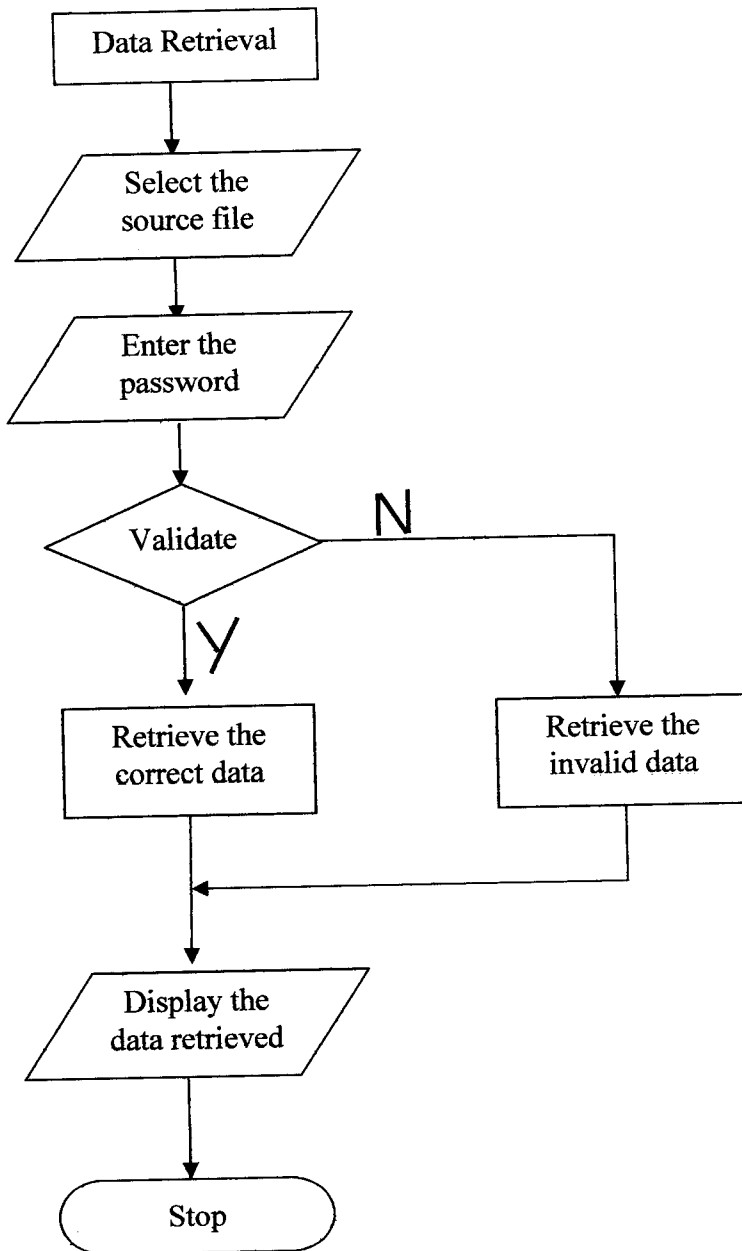


Fig: 3.5.2 Data Retrieval flowchart

I hear and I forget

I see and I remember

I do and I understand

- Confucius

CHAPTER 4

Implementation

The design phase helped to identify the sub problems of the original problem main. Human beings tend to apply a divide and a conquer strategy when they are confronted with a complex problem. Hence the complex problem is partitioned into smaller ones that are more manageable. The entire project is accomplished by the integration of various tasks are stated below:

- Study of file formats
- Data Hiding
- Data Retrieval
- Authentication
- Conversion of C files into DLLs
- User interface creation

2.1 Project Modules

- **Study of File formats**

The file formats which are under consideration for the accomplishment of the product are .bmp and .wav files. They are uses as containers within which any form

data can be hidden. Hence a thorough study of the 2 file formats is an essential. The detailed file formats are presented in the Appendix.

➤ **Data hiding**

A complex understanding of the structure of .bmp and .wav files paved way further process. It was identified that the LSB's can be used to store the sensitive data without drawing any suspicion.

To a computer, an image is an array of numbers that represent high intensities at various points or pixels. These pixels mark up the images raster data. Digital images are stored in 24 bits or 8 bits per pixels.

24 bit colors

They are known as true color images. Every pixel can have one in 2^{24} colors and these are represented in quantities of 3 basic colors: Red (R), Green (G) and Blue (B). They provide more space for hiding images but are generally large.

8 bit colors

Every pixel can have one in 256 (2^8) colors, chosen from a palette or a table of colors.

8 bit Grey scale

Every pixel can have one in 256 (2^8) shades of gray.

Algorithm

- Choose the cover image

- Enter the data to be hidden
- Convert the data to binary
- Compute length of data
- Compare the length of data with the size of the image. If it is less than 8% of cover image, store length in the file header, else print appropriate message
- Replace the LSBs present in the RGB quad of the cover image with the bits of the data to be hidden at a particular frequency.

Illustration

Consider the example for storing the letter 'A' in the 3 pixel of a 24-bit image:

Pixels

```
(00100111 11101001 11001000)
(00100111 11001000 11101001)
(01100100 00100111 11101001)
```

The letter 'A' has the ASCII code of 65, which is 10000001 in binary.

Result

The values of pixels after insertion of A:

```
(00100111 11101000 11001000)
(001001110 11001000 11101000)
(01001000 00100111 11101001)
```

Thus we did not have to change all of the LSBs but only those that didn't match to be modified. So on an average 50% of the pixels of an image will not be affected by embedding.

The above example uses adjacent bytes to embed data, whereas the same process is done at a specified frequency in the product.

Note: Data is embedded in the chunks using LSB insertion in a similar manner for .wav files.

➤ **Data Retrieval**

Algorithm

- Enter the password
- Retrieve the length of the password and bits one by one
- Password verification is done by authentication module

➤ **Creation of C DLLs**

The modules written using C have to be converted into DLL (Dynamic Link Libraries) because these routines are of 16 bit in nature. Since the Graphical user interface (GUI) is developed in VB, which is a 32-bit application, this conversion must be employed in order to access the C programs from the VB Application. The details of how a DLL is created and their need are explained in the Appendix D.

➤ **User Interface Creation**

This is the last module of the project. Once the base application was completed and refined, the user interface was created for both the versions. The DOS version has a simple interface designed by using a wide variety of DOS functions and interrupts. On the other hand, the windows version has a comfortable user-friendly GUI developed using Visual Basic.

*Undetectable errors are infinite in variety, in
contrasting to detectable errors which by definition are
limited*

- Ben Lach Mann

CHAPTER 5

Testing

The time taken to produce a product should be reflected significantly in the quality of the product produced. Hence the developed product was put under an exhaustive test procedure.

Testing was performed for two factors:

- To provide an error free product.
- To evaluate the quality of the product.

1 Test Plans for a Bug Free Product

Some test cases include:

- Nominal inputs were provided to test if expected results were achieved.
- Boundary conditions were tested with minimum and maximum values.
- Logically related inputs were provided to check their correctness of relations.
- Special values namely empty files were also tested.

“Bugs lurk in corners and congregate at boundaries” – Beizer. This product, “Secluder” was completely tested and almost all the bugs and errors were covered. Suitable remedial actions were taken to make the product an error free

2 Evaluation

The evaluation of the product was performed with regard to:

- The amount of data that can be stored.
- Quality of original source media.
- Size of the resultant file.

5.2.1 Data Capacity

An algorithm’s ability to store the maximum amount of data in image is a measure on a scale, from not being able to store any information to the maximum capacity that an image can store.

5.2.2 Quality of Data hiding

Steganography basically exploits the weakness of Human Visual System (HVS) and Human Auditory System (HAS). Since quality testing of data hiding is a subjective assessment, the results were presented to the users. They were asked to detect any change in the images or sound. The users were also allowed to zoom in and out to look closely at the pixels and the image as a whole respectively.

Test case for Text Hiding

File Name	File Size(kb)	Percentage of Data Hidden(%)
Acdc.bmp	269 kb	1% of source file
Ganesh.bmp	380 kb	2% of source file
Cut.wav	5.44 mb	1% of source file

Table: 5.2 Test Cases

➤ Data hidden in Acdc.bmp and Cut.wav

Steganography is the process of hiding data in any of the digital files. Sparks are in good form to complete the project. Definitely INDIA will win the World Cup 03. Australia defeated the Lankan in the semifinal and entered the finals of the world cup with ease. America is confident of starting second Gulf war with Iraq to destroy the weapons possessed by the Iraq.

When you love someone Show it!

Test case for File hiding

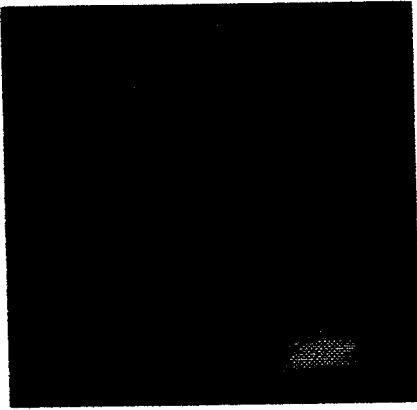
➤ File Hidden in Ganesh.bmp

Beck.bmp is the file hidden inside the Ganesh.bmp file and was retrieved without any damage.

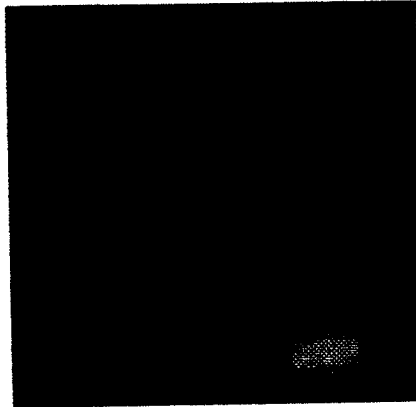
The following images give an illustration about the proceedings :

Case 1

BEFORE HIDING



AFTER HIDING



24 bit Color image

Case 2

BEFORE HIDING



AFTER HIDING



8 bit Gray Scale

Case 3

Hidden in Ganesh.bmp



Retrieved from Ganesh.bmp



In a similar method wave files were also tested and it was found that the quality was highly acceptable.

5.2.3 Size of the resultant file

Another striking feature of the product is that the size of the image/audio file remains identical before and after hiding.

A few of the screenshots which support this fact are exhibited at the Appendix.

Conclusion isn't the important part, it's the journey

CHAPTER 6

CONCLUSION

“Security is a continuous evolving process” says Bruce Schneier. Total security is a Herculean task.

“**SECLUDER**” is implemented in two forms – one for DOS and one for WINDOWS and works perfectly without any bugs. Transparency, Robustness and Capacity, which are the core objectives of the project, has been achieved. Since it has a professionally and user friendly designed GUI, it is feasible for the common mass to utilize this software for securing their private information.

The analysis phase provided a better understanding of the various file formats and their implementation. The existing steganographic software was tested and the drawbacks were found. After a detailed study, the proposed product’s features were designed.

Initially the product was developed only for DOS. To increase the functionality and for better acceptance by the common mass, another version with an elegantly designed user interface was implemented for Windows.

During the course of the project various risks like computer failure leading to delay and data loss were identified. Timely measures were taken to minimize them. The risk that computer failure would cause delay in the project, was managed by ensuring more than one system at a time. Similarly, risk of data loss was minimized by taking regular backups. Thus, by eliminating the risks of delay and data loss, the schedule was kept on track. By keeping to schedule, the risks of missing deadlines were reduced.

Thus, SECLUDER proves to be an easy to use security suite that incorporates high-end security by using LSB steganographic technique to hide data in BMP and WAVE files.

So, if one wants to hide information and easily transmit to someone else without being intercepted, one can try out SECLUDER, fascinating stuff!

If a program is useful, it will have to be changed.

If a program is useless it will have to be documented

CHAPTER 7

FURTHER ENHANCEMENTS

7.1 Future of SECLUDER

- The authentication process can be improved to a great extent by employing standardized encryption algorithms like Blowfish, to ensure high-end security.
- The volume of data that can be employed within an image / audio file can be considerably increased without affecting the quality of the original file.
- Data Compression, if added to the existing tool, would be an added advantage leading to the increase in the percent of data that can be steganized within an image / audio file.

7.2 Recommendations for Future Projects

“Steganography and digital Water marking” – a Global view of a research work done by Johnson and Jajodia states that LSB insertion technique is a recommended technique for Steganography. So, one can improve the data rate by using 2 or 3 bits of the LSB in case of 24 bit images for embedding data. Moreover by using more sophisticated cryptographic techniques that would take powerful computers a very long time to break could enhance security and integrity of the embedded message.

*The World is a book. Every step we take opens a new
page for us*

- Lamartine

CHAPTER 8

BIBLIOGRAPHY

- [1]. N.F Johnson, "Steganography and Digital Watermarking for Information Hiding" at <http://www.jjtc.com/stegdoc/stegdoc.html>
- [2]. Duncan Sercans, "An Introduction to Steganography", at <http://www.cs.uct.ac.za/cs400w/nis/papers99/dsellars/stego.html>
- [3]. N.F. Johnson and S. Tajodiam, "Exploring Steganography – seeing the unseen" at <http://www.jjtc.com/pub/r2026.pdf>
- [4]. Jumi Kassim, "Hiding in Plain Sight – Steganography in today's Digital Environment" – done as S/W Engineering Research Project for March 2002
- [5]. Ronald Mendell, "Steganography—An Electronic Spycraft" at http://www.earthweb.com/article/0,1045fm_624101_00.html
- [6]. R.J. Anderson and F.A.P. Peticolas, "On the limits of Steganography", IEEE Journal of selected areas in Communication 16(4):474-481, May 1998
- [7]. S-Tools
<ftp://ftp.funnet.fi/pub/crypt/mirrors/idea.sec.dsi.unini.it/code/s-tools4.zip>
- [8] Steg Tools
<http://members.tripod.com/steganography/stego/software.html>

Peter Norton, "Peter Norton's Guide to Visual Basic 6 "

]. Yashvant Kanetkar, "Let Us C"

]. Charles Petzold, "Programming Windows Edition 5 for Microsoft Visual C++
)"

2]. Roger S. Pressman, "Software Engineering , A practitioners approach",
McGraw Hill, 1997

If at all you first succeed, call it version 1.0

CHAPTER 9

APPENDIX - A

BMP FILE FORMAT

The BMP file has been created by Microsoft and IBM and is therefore very tightly bound to the architecture of the main hardware platform that both companies support: the IBM compatible PC. This means that all values stored in the BMP file are in the Intel format, sometimes also called the Little Indian format because of the byte order that an Intel processor uses internally to store values.

The BMP files are the way, Windows stores bit mapped images. The BMP image data is bit packed but every line must end on a dword boundary - if that's not the case, it must be padded with zeroes. BMP files are stored bottom-up, that means that the first scan line is the bottom line.

The BMP format has four incarnations, two under Windows (new and old) and two under OS/2, all are described here.

BMP Contents

1.1 Field Details

The following will try to provide more information in this regard:

1.1.1 Height Field

The *Height* field identifies the height of the bitmap in pixels. In other words, it describes the number of scan lines of the bitmap. If this field is

negative, indicating a top-down DIB, the *Compression* field must be either BI_RGB or BI_BITFIELDS. Top-down DIBs cannot be compressed.

1.1.2 Bits Per Pixel Field

The *Bits Per Pixel* (BPP) field of the bitmap file determines the number of bits that define each pixel and the maximum number of colors in the bitmap.

- **When this field is equal to 1.**

The bitmap is monochrome, and the palette contains two entries. Each bit in the bitmap array represents a pixel. If the bit is clear, the pixel is displayed with the color of the first entry in the palette; if the bit is set, the pixel has the color of the second entry in the table.

- **When this field is equal to 4.**

The bitmap has a maximum of 16 colors, and the palette contains up to 16 entries. Each pixel in the bitmap is represented by a 4-bit index into the palette. For example, if the first byte in the bitmap is 1Fh, the byte represents two pixels. The first pixel contains the color in the second palette entry, and the second pixel contains the color in the sixteenth palette entry.

- **When this field is equal to 8.**

The bitmap has a maximum of 256 colors, and the palette contains up to 256 entries. In this case, each byte in the array represents a single pixel.

- **When this field is equal to 16.**

The bitmap has a maximum of 2^{16} colors. If the *Compression* field of the bitmap file is set to BI_RGB, the *Palette* field does not contain any entry.

Each word in the bitmap array represents a single pixel. The relative intensities of red, green, and blue are represented with 5 bits for each color component. The value for blue is in the least significant 5 bits, followed by 5 bits each for green and red, respectively. The most significant bit is not used.

If the *Compression* field of the bitmap file is set to BI_BITFIELDS, the *Palette* field contains three dword color masks that specify the red, green, and blue components, respectively, of each pixel. Each word in the bitmap array represents a single pixel.

1.1.2.1 Windows NT specific

When the *Compression* field is set to BI_BITFIELDS, bits set in each dword mask must be contiguous and should not overlap the bits of another mask. All the bits in the pixel do not have to be used.

1.1.2.2 Windows 95 specific

When the *Compression* field is set to BI_BITFIELDS, Windows 95 supports only the following 16bpp color masks: A 5-5-5 16-bit image, where the blue mask is 0x001F, the green mask is 0x03E0, and the red mask is 0x7C00; and a 5-6-5 16-bit image, where the blue mask is 0x001F, the green mask is 0x07E0, and the red mask is 0xF800.

- **When this field is equal to 24**

The bitmap has a maximum of 2^{24} colors, and the *Palette* field does not contain any entries. Each 3-byte triplet in the bitmap array represents the relative intensities of blue, green, and red, respectively, for a pixel.

- **When this field is equal to 32**

The bitmap has a maximum of 2^{32} colors. If the *Compression* field of the bitmap is set to BI_RGB, the *Palette* field does not contain any entries. Each dword in the bitmap array represents the relative intensities of blue, green, and red, respectively, for a pixel. The high byte in each dword is not used.

If the *Compression* field of the bitmap is set to BI_BITFIELDS, the *Palette* field contains three dword color masks that specify the red, green, and blue components, respectively, of each pixel. Each dword in the bitmap array represents a single pixel.

1.2.3 Compression Field

The *Compression* field specifies the way the bitmap data is stored in the file. This information together with the *Bits per Pixel (BPP)* field identifies the compression algorithm to follow.

4 Colors Field

The *Colors* field specifies the number of color indices in the color table that are actually used by the bitmap. If this value is zero, the bitmap uses the maximum number of colors corresponding to the value of the *BBP* field for the compression mode specified by the *Compression* field.

If the *Colors* field is nonzero and the *BBP* field less than 16, the *Colors* field specifies the actual number of colors the graphics engine or device driver accesses.

If the *BBP* field is 16 or greater, then *Colors* field specifies the size of the color table used to optimize performance of Windows color palettes.

If *BBP* equals 16 or 32, the optimal color palette starts immediately following the three double word masks.

If the bitmap is a packed bitmap (a bitmap in which the bitmap array immediately follows the bitmap header and which is referenced by a single pointer), the *Colors* field must be either 0 or the actual size of the color table.

5 Important Colors Field

The *Important Colors* field specifies the number of color indices that are considered important for displaying the bitmap. If this value is zero, all colors are important.

APPENDIX B

WAVE File Format

The WAVE file format is a subset of Microsoft's RIFF specification for the storage of multimedia files. RIFF is a Windows file format for storing chunks of multi-media data, associated descriptions, and formats, play lists, etc. A RIFF file starts out with a file header followed by a sequence of data chunks. A WAVE file is then just a RIFF file with a single "WAVE" chunk which consists of two sub-chunks - a "fmt" chunk specifying the data format and a "data" chunk containing the actual sample data. This form is called the "Canonical form".

The Canonical WAVE file format

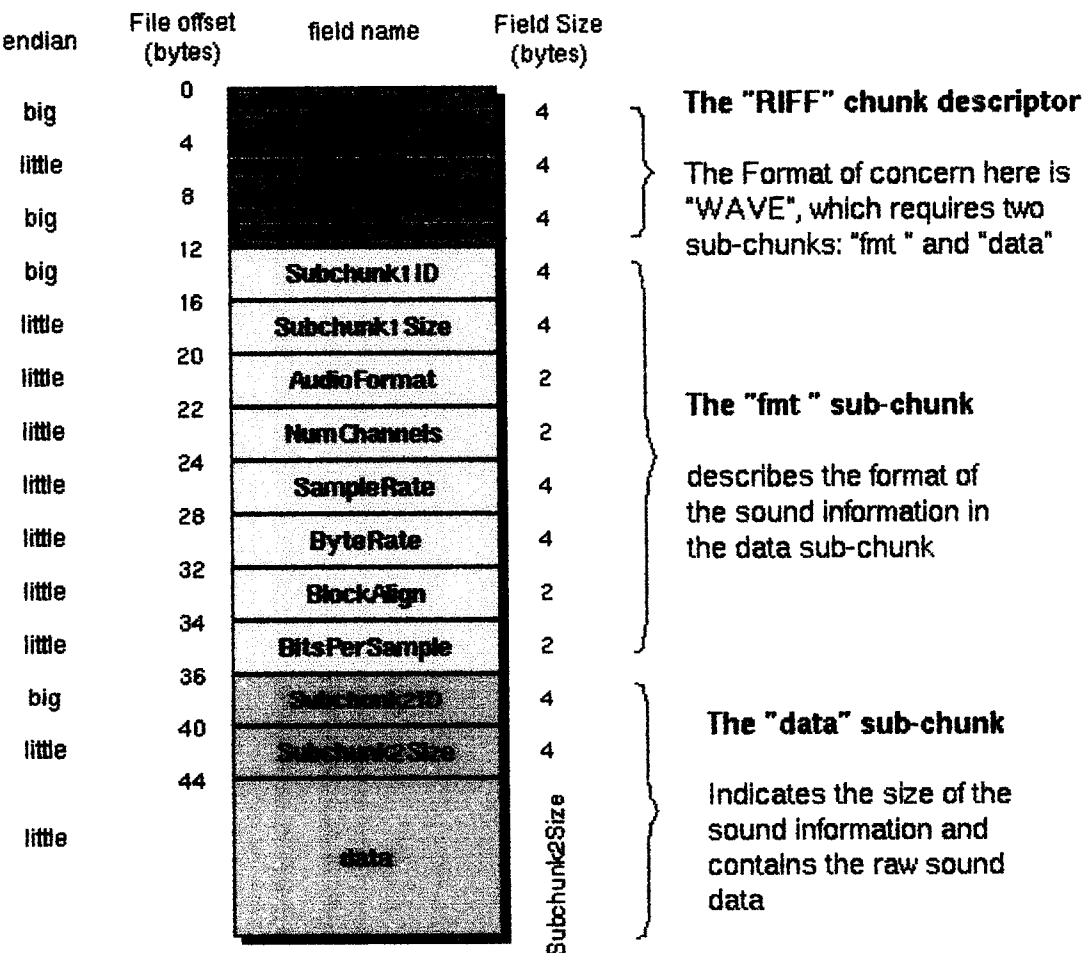


Fig: A.B.1 Wave Format Diagram

RIFF Header

The canonical WAVE format starts with the RIFF header:

A RIFF file has an 8-byte RIFF header, identifying the file, and giving the residual length after the header. The riff header is immediately followed by a 4-byte data type identifier. For ".WAV" files this is "WAVE".

ChunkID

Contains the letters "RIFF" in ASCII form.
(0x52494646 big-endian form).

ChunkSize

36 + SubChunk2Size, or more precisely:
 $4 + (8 + \text{SubChunk1Size}) + (8 + \text{SubChunk2Size})$

This is the size of the rest of the chunk. This is the size of the entire file in bytes minus 8 bytes for the two fields not included in this count namely:

ChunkID and ChunkSize.

Format

Contains the letters "WAVE".
(0x57415645 big-endian form).

RIFF Chunks

The entire remainder of the RIFF file is "chunks". Each chunk has an 8-byte chunk header identifying the type of chunk, and giving the length in bytes. The "WAVE" format consists of two sub chunks:

"fmt " and "data" chunks

This concludes the general RIFF file description. The types of chunks to expect for .WAV files and the format of the content data of each chunk type are described in the sections that follow.

WAVE Format Chunk

This section describes the Waveform format, which is used to represent digitized sound. The WAVE format chunk specifies the format of the data. However, the format chunk must always occur before data chunk and both of these chunks are mandatory in a WAVE file.

The format chunk is defined as follows:

Subchunk1ID

wFormatTag -Contains the letters "fmt "
(0x666d7420 big-endian form)

Subchunk1Size

This is the size of the rest of the Sub chunk.

Audio Format

PCM = 1 (i.e. Linear quantization)

Values other than 1 indicate some form of compression.

NumChannels

wChannels -The number of channels represented in the waveform data, such as 1 for mono or 2 for stereo. Mono = 1, Stereo = 2.

SampleRate

dwSamplesPerSec -The sampling rate (in samples per second) at which each channel should be played (8000, 44100, etc).

ByteRate

dwAvgBytesPerSec -The average number of bytes per second at which the waveform data should be transferred. Playback software can estimate the buffer size using this value.

$$(\text{== SampleRate} * \text{NumChannels} * \text{BitsPerSample}/8)$$

BlockAlign

wBlockAlign -The block alignment (in bytes) of the waveform data.

Playback software needs to process a multiple of wBlockAlign bytes of data at a time, so the value of wBlockAlign can be used for buffer alignment.

BitsPerSample

8 bits = 8, 16 bits = 16, etc.

DATA CHUNK

The "data" subchunk contains the size of the data and the actual sound:

subchunk2ID

Contains the letters "data".
(0x64617461 big-endian form).

subchunk2Size

This is the number of bytes in the data.
($= \text{NumSamples} * \text{NumChannels} * \text{BitsPerSample} / 8$)

Data

The actual sound data

APPENDIX – C

Real programmers don't document if it was hard to write, it should be hard to understand" – John Carmach

Software Requirements Specification

This SRS was produced at the culmination of initial investigation and analysis of problem domain.

Introduction

A. Purpose of the project

Security is of prime concern in areas where sensitive and confidential data transactions are involved. The physical evidence of a message being communicated leaves way for a security threat. Hence a means to provide secrecy is accomplished through Steganography

B. Scope

The main objective of this project is to keep information out of the reach of unauthorized intruders and this is achieved by means of steganizing the data. This would serve to provide high-end security for a wide variety of real time applications like banking, copyright protection etc.,

C. Overall Description

The project aims to hide explanatory information, the secret communications within an image (or) an audio file and maintain anonymity.

D. Software Project Constraints

➤ General Constraints

- The source file specified should be in any one of the file formats.

Image file formats => *.bmp.

Audio file formats => *.wav.

- The volume of data that can be steganized solely depends on the size of the source file chosen.

➤ User Constraints

- The user should choose only the specified file format.
- The user should be well aware of the various general constraints posed by the tool.

Information Description

Problem Analysis

For the few decades of the existence, computers were primarily used for search and communication. Under these conditions security did not get a lot of attention. But now millions of ordinary citizens use computers for banking, shopping, paying tax and so on., So security is looming on the horizon as a potentially massive problem. Security is the broad topic and covers multitude of sins.

Most security problems are intentionally caused by malicious people trying to gain some benefit (or) harm someone. Few of the most common perpetrators are hackers, business men, fired exemployers, spy's, terrorists.

Security problems are majorly divided into 5 parts

- Authorization
- Secrecy
- Authentication
- Non Repudiation
- Integrity

Secrecy has to do a lot with keeping information out of the hands of unauthorized intruders.

Conventionally cryptography is used for secrecy. Though many new algorithms are introduced cryptanalysts are smart enough to break all those algorithms. This enforces to accomplish secrecy with a new technology.

In Cryptography, there is physical evidence that a message is being communicated. It creates an enthusiasm to the passive user to intrude the physical message present. Steganography has the advantage that the physical evidence of a message is not exposed in any form.

System Flow Representation

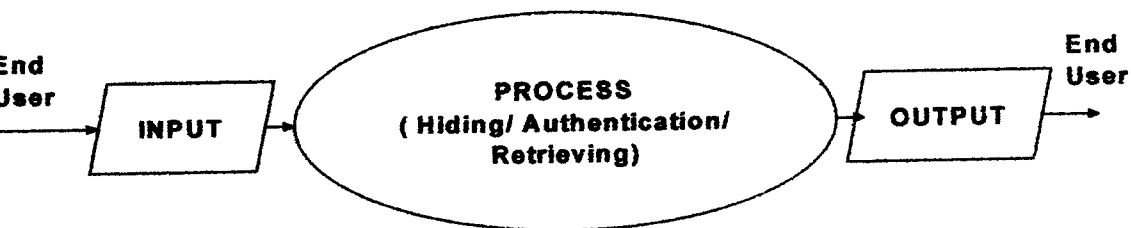


Fig: S.1 System Flow Representation

- The confidential data is obtained as input
- The data is then hidden inside the specific file format

- On request, after proper authentication the data is retrieved and presented as output

Functional Specification

➤ Functional Partitioning

- Hiding process
- Authentication
- Retrieval of hidden data

➤ Functional Description

- Processing Narrative
- Hiding Process

The human inability to distinguish minor changes in image color or sound quality is to make use of in objects that contain redundant information. Speaking of images, changing the value of LSB (Least Significant Bits) of the pixel color won't result in perceivable change of that color.

One can tell the differences between the clean and the loaded file only by comparing them, so if one looks at the resulting file only, it looks totally innocent. This idea is utilized for hiding data within an image or sound file.

For better security it is recommended that one uses images with many halftones and preferable unknown to the public because minor changes in them will not be noticed.

Authentication

To enhance the security of the steganized file, authentication by means of passwords/pass phrase will be implemented. The input to this module randomly mixed with the image/audio file and is made sure to defeat any Steganalysis attack.

Retrieval of hidden data

The Hidden data can be retrieved after proper authentication without any damage to the steganized file. The reverse process of hiding is employed for retrieval of hidden data.

Restrictions/Limitations

- The data to be steganized should be less than 8% of the source file.
- End user should choose only the specified file format.

Performance Requirements

➤ H/w Specification

Particulars	Minimum Requirements
Processor	PIII / Celeron 533 MHz
Memory	64 MB RAM
Display Unit	Color Monitor
Audio device	Speakers
Hard disk	4 GB HDD

➤ S/w Specification

Particulars	Minimum Requirements
Operating System	Microsoft Windows 98
Picture viewer	MS Paint /ACD See viewer any picture viewer.
Audio Software	Qcd Player

➤ **Developed in**

- Turbo C 3.0
- Microsoft Visual Basic 6.0
- Microsoft Visual C++

Design Constraints:

Statement

A generalized procedure for hiding process is not followed for all the file formats.

Justification

Since the file formats are not uniform, a generalized algorithm cannot be proposed.

Statement

The amount of data that can be steganized is comparatively less when compared to the size of the image/audio file.

Justification

More the number of bits steganized, the more it will affect the original picture and hence the volume of data is limited.

Supporting Diagrams

➤ Hiding Process

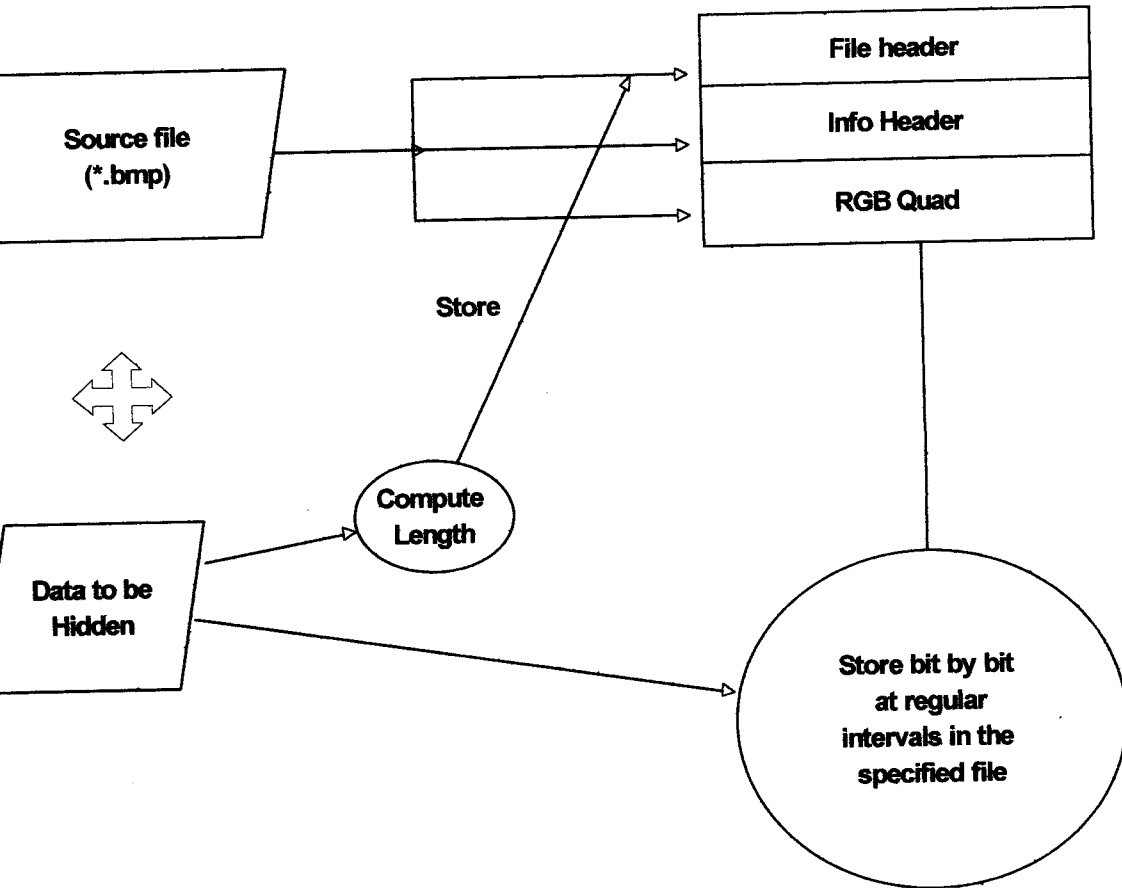


Fig: S.2 Hiding Process

➤ Retrieval Process

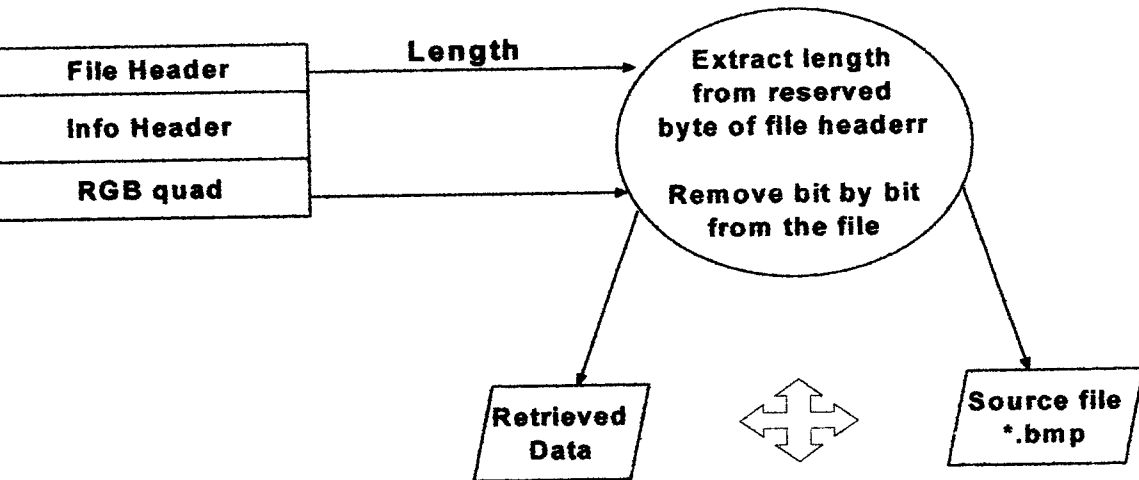


Fig: S.3 Retrieval Process

Foreseeable Modifications / Enhancements

- The Authentication process may be improved to a greater extent to ensure high-end security.
- User interfaces play a major role in determining the response of the project. So the user interface screen should be customized to provide much more sophistication aiming at better design and end-user's comfort.
- A wide range of file formats can be augmented to the existing list of file formats.
- The volume of data that can be steganized within an image/ audio file should be considerably increased without any loss to the original file.

- Data compression should precede the steganization process because it will be an added advantage leading to the increase in the percentage of data that can be steganized within an image/audio file.

Essary

Authentication

Establishing the identity of the person one is dealing with.

Authorization

Set and manage access privileges for users and groups.

Cryptography

An art of converting plain text to cipher text.

Half-Tones

A reproduction printed from a block in which the various tones of grey are produced from small and large black dots.

Integrity

Proof that, the message contents have not been altered deliberately or accidentally, during transmission.

Non-Repudiation

The certainty of knowing that, the sender of the message cannot later deny having sent it.

Pixels

An array of tiny dots of color, of which, an image is composed of.

Secrecy

Evidence that the contents of the message have not been disclosed to third parties.

Steganalysis

Steganalysis is the process of detecting steganography by looking at variances between bit patterns and unusually large file sizes.

Steganography

Steganography is the hiding of information within a more obvious kind of communication. Digital steganography involves the hiding of data inside a sound or image file.

Bibliography

Johnson, Neil. "Steganography". URL: <http://www.jitc.com/stegdoc/>

Johnson, Neil. "Steganography and Digital Watermarking".
<http://www.jitc.com/Steganography>

File Formats. <http://www.wotsit.org>

File formats. <http://www.fastgraph.com>

Yashwant Kanetkar. <http://www.funducode.com/fileformats>

File formats. <http://www.myfileformats.com>

APPENDIX – D

The answer to the limitless problems of DLLs is obvious; don't use them” – Dr. Dobb's Journal

DLL CONVERSION

What is a DLL?

DLLs (Dynamic Link Libraries) are an important aspect of Windows. A DLL contains functions that your executable program can call during execution. In other words, a DLL is a library of functions that your program can link with dynamically.

Dynamic Links are created as needed. When your program needs a function that is not in the executable file, Windows loads the Dynamic Link Library (the DLL), making all of its functions available to your application. At that time, Windows resolves the address of each function and dynamically links it your application.

Why Use DLLs?

Here are four reasons why you might want to use a DLL:

Access to C Run-Time Functions

The C run-time library has many useful functions that would be available to Visual Basic programmers were it not for DLLs. For example, the `_dos_getdiskfree` function allows you to calculate the total amount of disk space and the free disk space available on a drive.

Access to Windows API (Application Programming Interface) Functions that Require Callback Routines

Some Windows API functions require a callback function. A callback function is a function that Windows will call while executing the API call. An example of this sort of function is EnumTask Windows, which will give the handle of all Windows that are owned by a particular task.

Speed

It is a fully compiled language that works at a level that is fairly close to native machine code. This means that the execution of the programs that are written in C will be fast.

Load on Use

Code and data from a DLL are loaded only when needed. A DLL can be organized such that only required parts are loaded as opposed to the entire DLL. This reduces the amount of memory required and the time taken to load.

Building a DLL Using Visual C++

Here are the steps necessary to build a DLL using Visual C++ :

1. Start Visual C++
2. Create a new project by choosing New from the Project menu. Select the following options:

- Set the Project Type to “Windows dynamic-link library (.DLL)”.
- Clear the “Use Microsoft Foundation Classes” check box.
- You can also set or view these options later by choosing Project from the Options menu.

Add your existing .C and .DEF files to the project by using the dialog box that comes up when you choose Edit from the Prohec menu. Or enter your code directly in the Visual C++ editing window. (See the .C and .DEF example code listed below.)

From the Project Menu, choose the Build .DLL option.

Sample Coding for DLL

c file coding

```
LPSTR __stdcall dispstr(LPSTR str)
```

```
    strcpy(str,"12345");
    strcat(str,"22");
    return(str);
```

def file coding

```
EXPORTS
```

```
    dispstr @1
```

B Coding

```
Private Declare Function dispstr Lib "c:\steg.dll" (ByVal x As String) As String
```

```
Private Sub Form_Load()
```

```
    = dispstr("asdasdas")
```

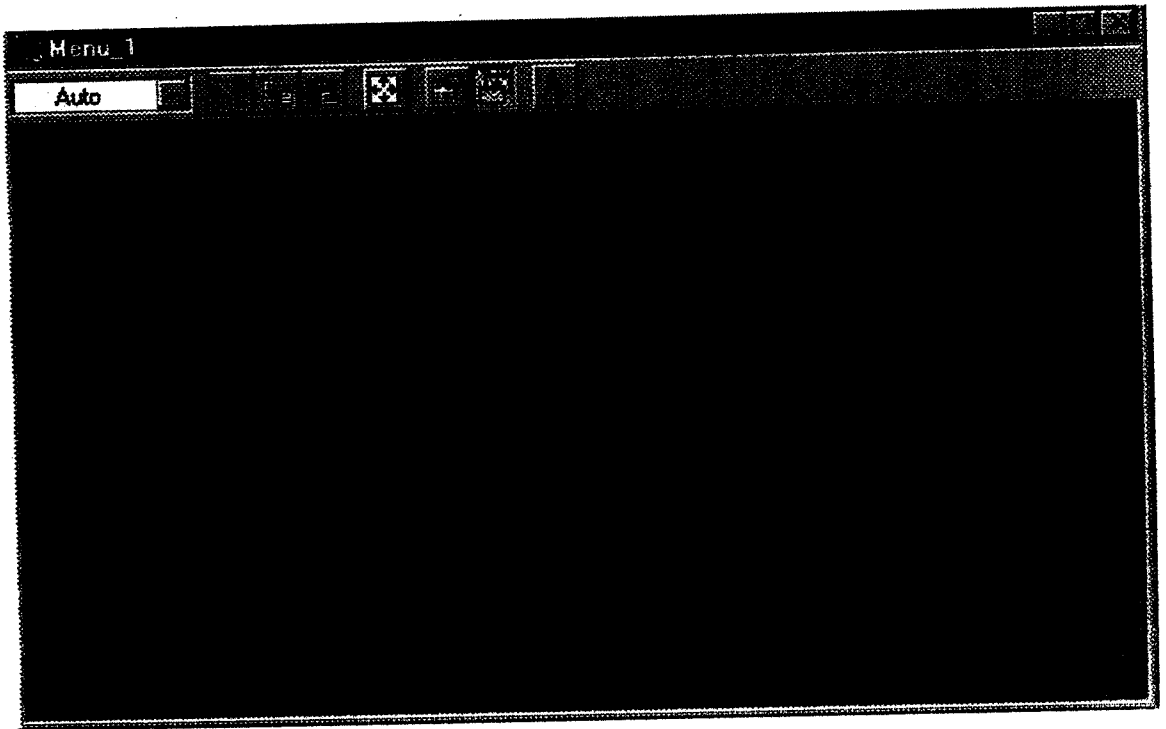
```
MsgBox x
```

```
End Sub
```

APPENDIX – E

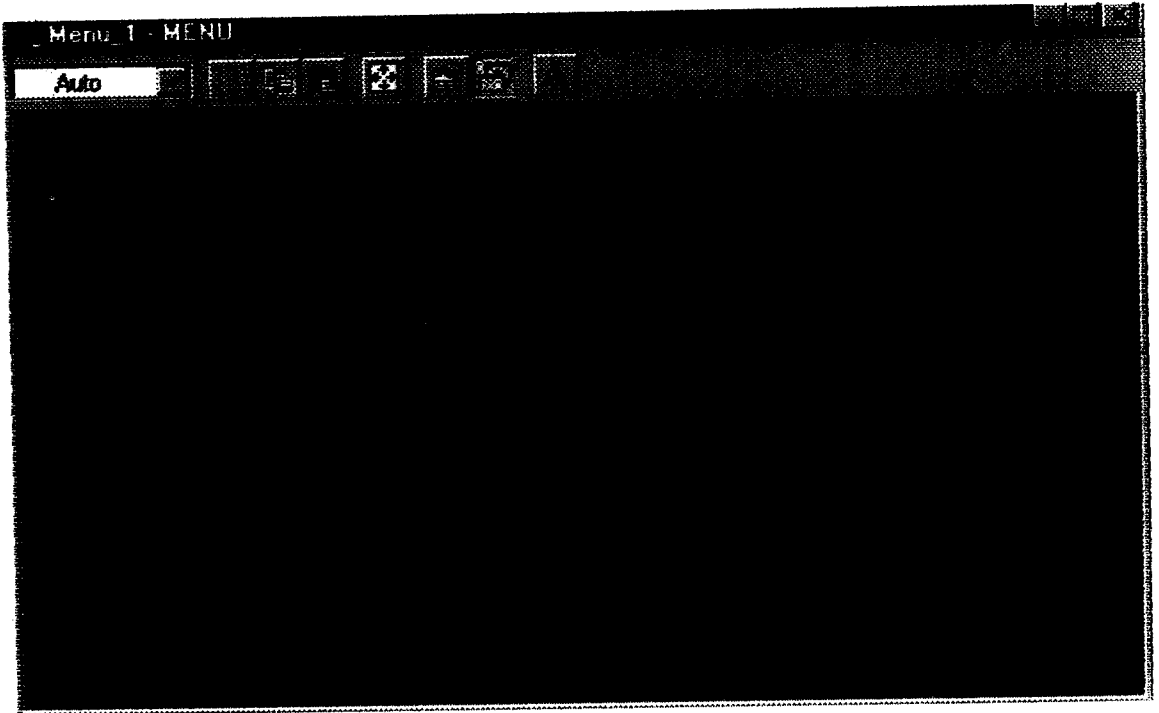
“A picture is worth thousand words”

DOS-Main Menu

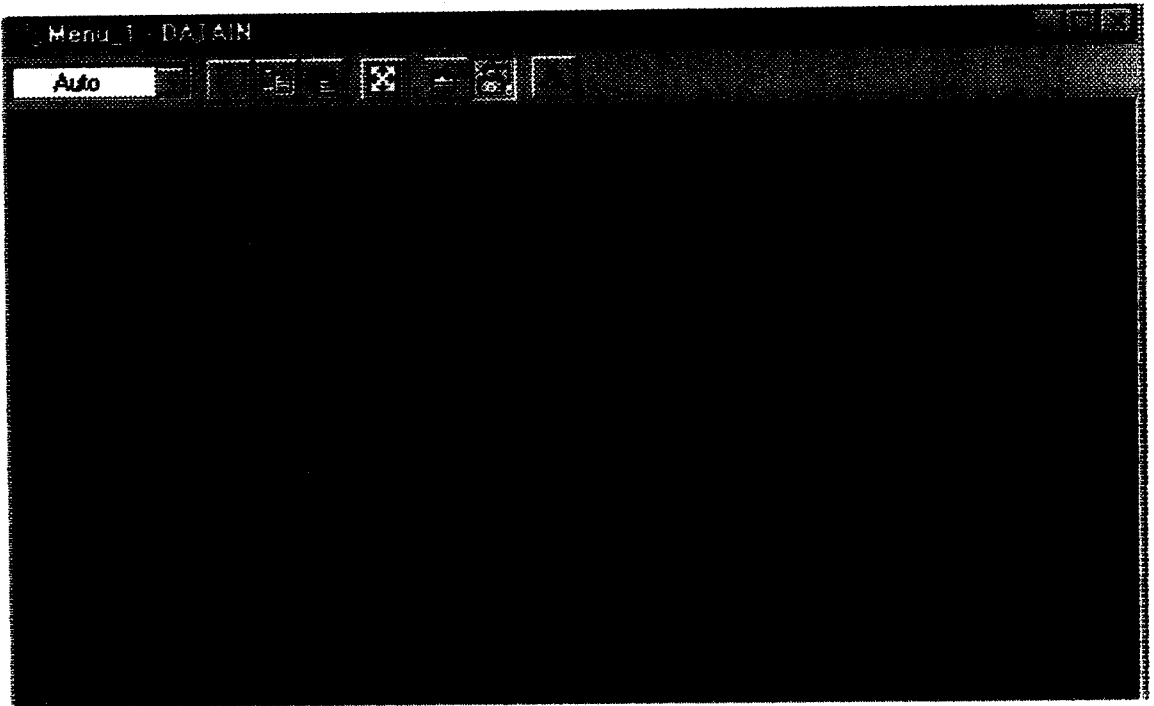


>

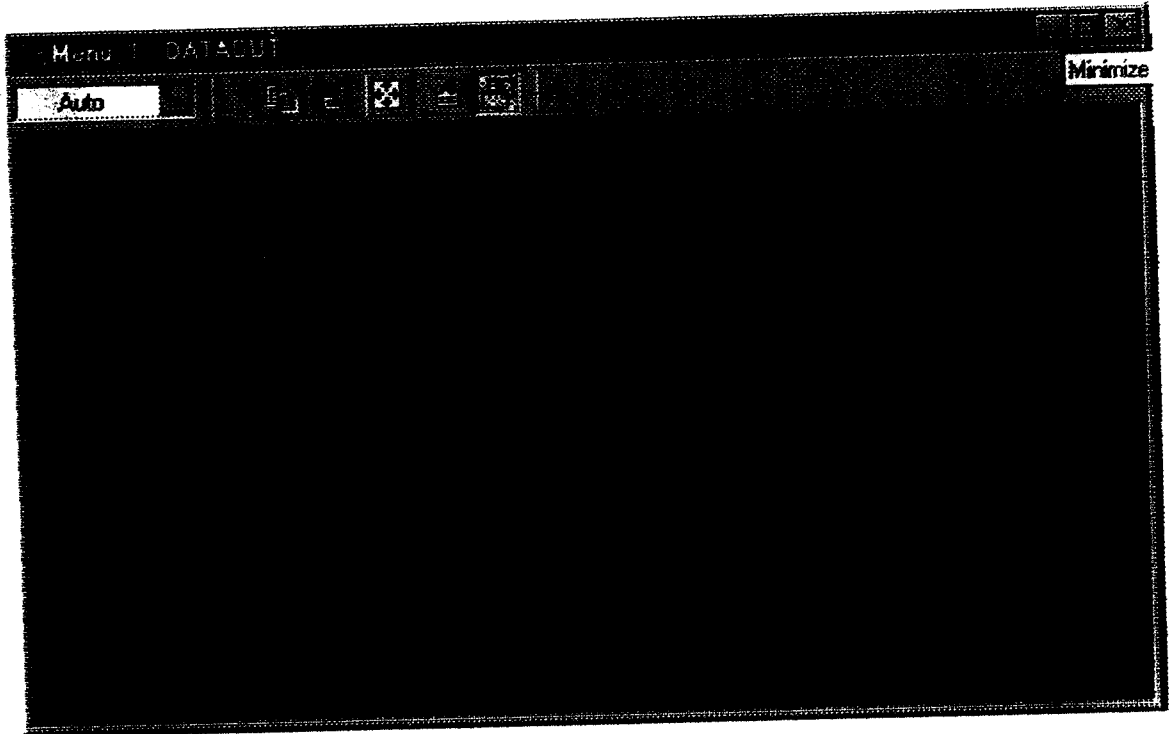
DOS-Menu for BMP



DOS-Data Hiding in BMP



DOS-Data Retrieval in BMP



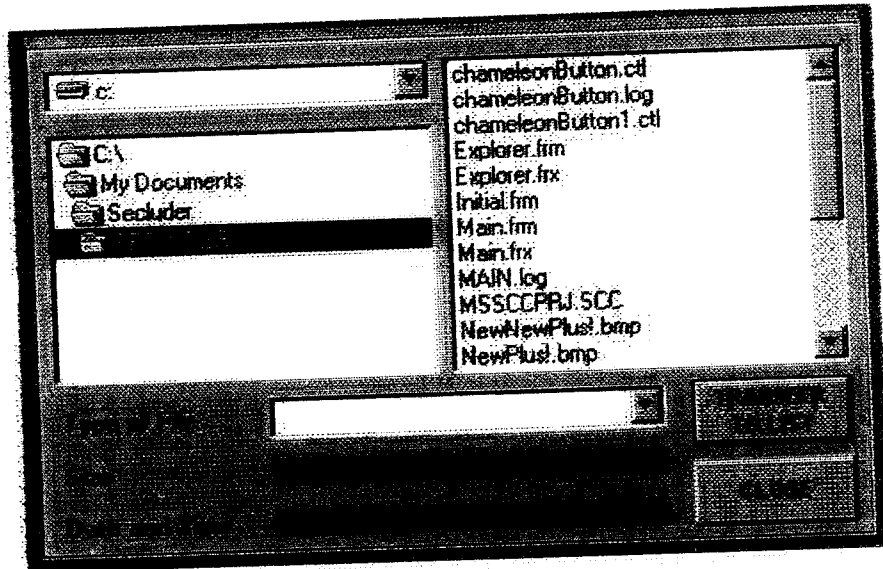
Snapshots for Windows Version

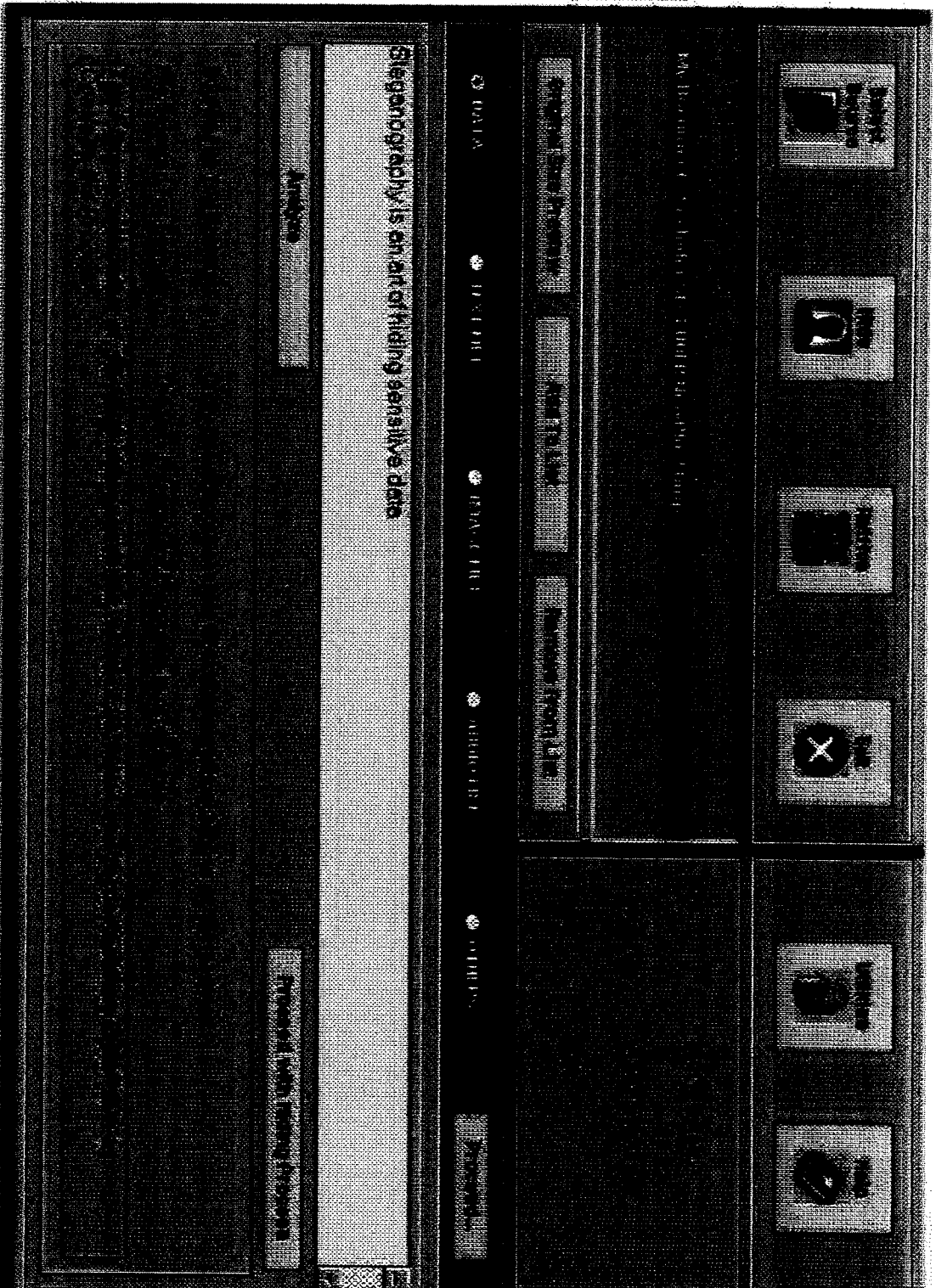


Select Source



Explorer Window





Sleepography is an art of hiding sensitive data

Andres

© 2011 © 2012

© 2011

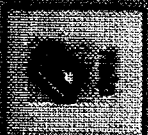
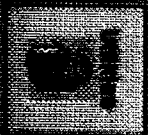
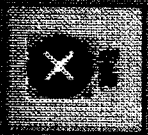
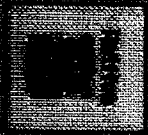
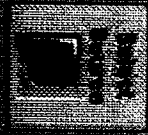
© 2012

© 2013

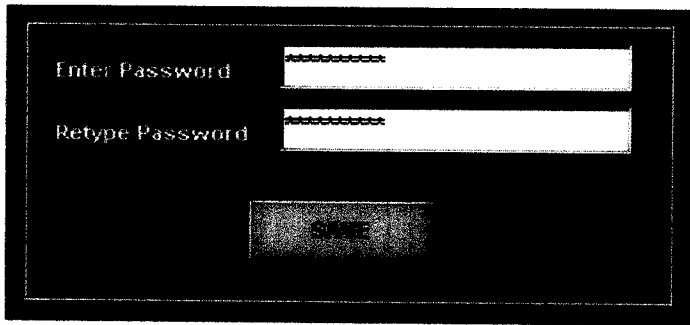
© 2014

© 2015

© 2016



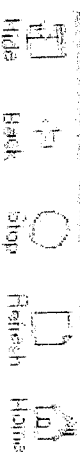
Password Screen



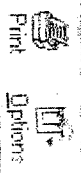
Enter Password

Retype Password

Windows-Help Screen 1



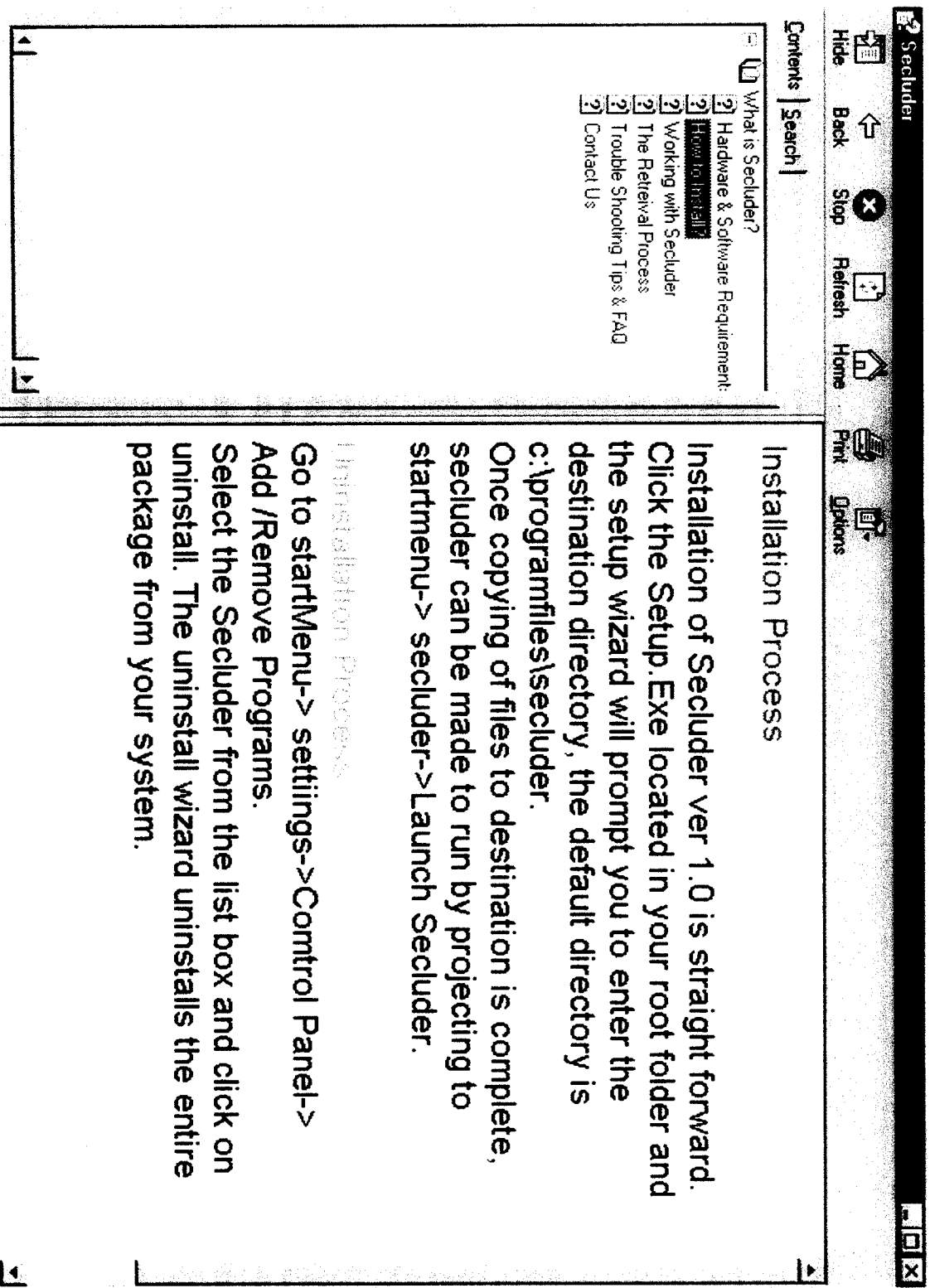
Contents | Search |



Secluder is a powerful package utility which brings in you the trick of hiding your sensitive data from prying eyes.

Using Secluder, you can embed your sensitive data like plain text, text file, Audio or even picture into another text file, picture file or Sound file. Secluder's unique speciality is that the file which contains the embedded data will not show any additional size and the size of the file will be equal to the original file size.

Windows-Help Screen 2



Windows-Help Screen 3

The screenshot shows a Windows Help window with a title bar containing 'Windows-Help' and standard window controls. The window is divided into several sections:

- Navigation Bar (Top):** Contains icons for Home, Refresh, Stop, Back, Hide, Search, and Print. Below these are labels for 'Home', 'Refresh', 'Stop', 'Back', 'Hide', 'Search', 'Print', and 'Options'.
- Search Section:** Includes a 'Contents Search' button, a text input field with the placeholder 'Type in the keyword to find:', and a search button. Below this is a list of search results, with 'seculder' selected.
- Topic List:** A box titled 'List Topics' containing the following links:
 - Contact Us
 - How to Install?
 - Trouble Shooting Tips and FAQ
 - What is Seculder?
 - Working with Seculder
- Main Content Area:** Displays the search results for 'Seculder'. The word 'Seculder' is highlighted in a box. The text below reads:

The Main requirement for **seculder** is the installation of VB RunTime files / Windows Core DLL's. The windows core DLL's would normally be installed alongwith a normal windows installation and there is no need to install it seperately.

The VB Run time files have to be installed manually. How ever, **Seculder** has made this process automatic if **Seculder** is installed properly via Setup.Exe created from Install Shield.

If you have copied **Seculder** from any other system, you would have faced this problem.
- Footer:** A 'Display' button is located at the bottom left of the window.