American Journal of Information Science and Computer Engineering
Vol. 2, No. 6, 2016, pp. 79-90
http://www.aiscience.org/journal/ajisce
ISSN: 2381-7488 (Print); ISSN: 2381-7496 (Online)

Securing Software Programs by Applying Security Services with Microsoft VB Net Programming

Kamel Hussein Rahouma*

Department of Electrical Engineering, Faculty of Engineering, Minia University, Minia, Egypt

Abstract

Software programs can be secured by applying security services. Security service are many and it is important to choose, from them, what suits the software or application under study. In this paper, we discuss a simple software program (the shopper program) and how we can secure it by applying some already existing security algorithms. The program follows up the movement of goods in the store as well as in the shopping place. Visual Basic Dot NET is used to produce the software application. The program uses five files to store its data. The first file is used to save the passwords of the users. The second and third files include the information of goods in the store and the shopping place. The fourth and fifth files save the daily movement in the store and the sales places. For security purposes, the users can use the program without accessing the used files as well as their folders. This is done by changing the files' attributes. The files' attributes are set to: 1) Hiding the file and folders, 2) Read-Only. The files' contents are encrypted to make it hard for the user to recognize these contents. A fast Stream Cipher Encryption/decryption is done using a random key generator. The file's contents are digitally hashed and signed by the user. This guarantees the integrity and authenticity of the files' contents.

Keywords

Software Security, Stream Ciphers, Cryptography, Hashing, Digital Signature

Received: September 9, 2016 / Accepted: September 25, 2016 / Published online: January 9, 2017

1. Introduction

Software piracy and tampering is a well known threat the world is facing. A lot of attempts are done to protect software from reverse engineering and tampering. It appears as if there is an ongoing war between software developers and crackers. Both parties want to get the upper hand over each other. Piracy of software is always one of the burning threats to the software industry, especially after the advent of the Internet and the availability of many software analysis tools [1].

Computer software is an important asset to any organization developing it as massive investment of time; money and intellectual capital are involved in its production. However, once produced, software is at risk to theft and misuse. It is estimated that tens of billion dollars of revenue is lost by the software industry due to software piracy alone. Piracy is no longer the only issue, but software tampering with the malicious intent of planting a Trojan horse in the end user’s system is a worrisome possibility [2].

Software protection can be done to the code and/or to the executable programs and/or the output results of running software programs. Attacking software may include modifications to a program to omit critical checks, such as license file checks, or reverse engineering of a key piece of a program’s functionality.

Many approaches are employed by software developers to provide ample software protection. From these approaches, block and multi-block hashing schemes, the widely used hardware based approaches, obfuscation, Guards, cryptographic techniques, watermarking techniques.

* Corresponding author
E-mail address: kamelrahouma@yahoo.com

Information security is the process of protecting information. It protects its availability, privacy and integrity. Access to stored information on computer databases has increased greatly. More companies store business and individual information on computer than ever before. Much of the information stored is highly confidential and not for public [3, 4].

In this paper, we are interested in protecting the executable software programs and their output information. This is done by applying cryptographic and hashing techniques. The cryptography techniques are classified on the basis of their key selection into [5]:

a) Symmetric (Private) Cryptography:

Symmetric or private-key or secret-key encryption involves using the same key for encryption and decryption. An algorithm is applied to the data to be encrypted using the private key to make them unintelligible. The slightest algorithm is the exclusive OR and it can make the system nearly tamper proof. Users have the provision to update the keys and use them to derive the sub keys. It is much effective and fast approach as compared to asymmetrical key cryptography. In symmetrical key cryptography; key has been generated by the encryption algorithm and then send it to the receiver section and decryption takes place.

b) Asymmetric (Public) Cryptography

Asymmetric cryptography uses a pair of keys one to encrypt and one to decrypt a message. One of the keys is public and the other is private. The two keys are generated simultaneously and they are prime numbers. The method is more secure as compared to private key cryptography but it consumes more power and takes more processing time therefore extra hardware is required.

c) Modern Cryptography

A combination of both public key and private key cryptography is known as modern cryptography. A pair of public and private keys has been used to encrypt and decrypt the data. The technique has the salient features of private key; fast speed, easy to process and features of public key such as secured, avoid key transportation, provide the power to the users to generate their own keys of variable length. Users also have the flexibility to upgrade the key at any interval of time. In this technique; certification authority has been used to keep the track of the entire system and keys.

The generation, modification and transportation of keys have been done by the cryptographic algorithm. There are many cryptographic algorithms available in the market and their strengths depend upon certain criteria. Some important cryptographic algorithms include: Data Encryption Standard (DES), International Data Encryption Algorithm (IDEA), Blowfish, Triple DES (TDES), Advanced Encryption Standard (AES), Twofish, RSA, Diffie-Hellman, Elliptic Curve Cryptography (ECC), Pretty Good Privacy (PGP), Public key infrastructure (PKI) [5].

Mostly, owners of software applications need some security that helps them control and follow up the events running by their software. Owners of software need to guarantee the protection of their applications against insider and outsider attacks. For instance, a shopper's owner may like to guarantee that the shopper employees are not cheating on. In this paper, some security services[1] are applied that fulfill these needs such as:

1) Hiding the files and folders of transactions (by using the attribute "hide"),
2) Denying the access of these files and folders (by setting the attributes of some files and folders to Read-Only),
3) Using encryption,
4) Using hash functions (to guarantee the integrity of the files' contents) and
5) Using digital signature (to record the signature of the users)..

Each of these services helps the software owner to control and follow up the use of the software. However, the security services can be applied to any similar software. Visual Basic DOT NET programming language is used for the implementation.

This paper includes five sections. Section one is an introduction and section (2) introduces the security services to be applied in this paper. In section (3), we introduce the software application which is a simple shopper program designed by the author and we also explain the operation of the program. A discussion of the security of the software is given in section (4). Some conclusions are highlighted in section (5) and a list of the used references is given at the end of the paper.

2. The Software Program: The Shopper Program

2.1. Introducing the Program

The shopper program is designed for the purposes of storing goods, selling goods, following up the movement of goods in a daily basis. VB NET is used to implement the program. The program has the following operations:

a) Preparing the program for the first time

[1] These services are published by the author in previous papers.

1) The passwords

2) Data and database files

b) Movement of goods in and out of the stores.

1) Adding new goods.

2) Changing information of goods

3) Making a report

c) Movement of the goods in and out of the shopping center.

1) A new customer service.

2) Adding new goods.

3) Changing information of goods

4) Making a report

Some of the last operations and processes need to be protected and some of them do not. The program uses four text files to save the running information aswell as the goods' information in the stores and in the shopping center. The files' names and uses are given in table (1).

Table 1. The files used by the shopping program.

File name	File use
File1.txt	Input/output information including encrypted passwords, hash blocks, date of accessing files.
File2.txt	Movement of good in/out of the store. Contents of the file include records for every good, such as: the number, the name, the price, the quantity.
File3.txt	Movement of goods in/out of shopping center. Contents of the file include records for every good, such as: the number, the name, the price, the quantity.
File4.txt	Daily movement of good in the shopping center. Contents of the file include records for every good, such as: the number, the name, the price, the quantity.

2.2. Program Operation

Two versions from the program were produced. These are the English and Arabic versions. Same operations are programmed and explained here:

2.2.1. Starting and Setting up the Program

The program icon is twice clicked to start its operation. Then the screen shot in figure (1) appears. This screen has two options: 1) End, 2) Continue

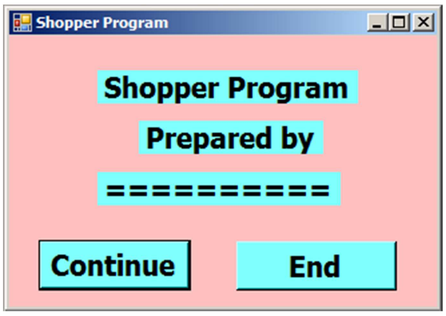


Figure 1. The first screen of program operation.

Pressing the button "End" in the last screen would end the program and pressing the button "Continue" starts operating the program and the screen in figure (2) appears. This screen has three options:

a) Program setup

b) Running the program

c) Ending the program.

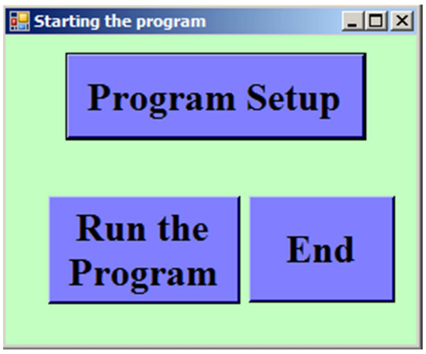


Figure 2. Running the program.

a) Pressing the button "Program Setup" in figure (2), the screen in figure (3) appears. This screen includes six options:

1) Passwords: This button is used to assign/modify passwords of the different parts of the program such as starting operation and accessing the files.

2) File/Folder access: This button is used to change the "hide" and "read-only" attributes of accessing files and folder.

3) Keys of Encryption: This button is used to generate/modify the seed, chaos bifurcation factor, and initial value of the chaotic equation.

4) Digital Signature: This button is used to change the signature/verification keys.

5) Previous Page: This button is used to go back to the previous page.

6) End: This button is used to end running the program.

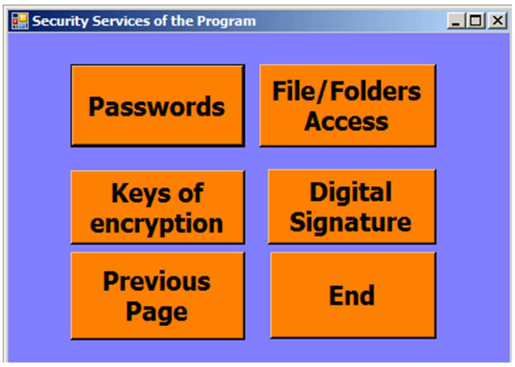


Figure 3. Security services of the program.

b) Pressing the "Passwords" button, the screen in figure (4) appears. This screen has 7 options:

1) Program Passwords: This button is used to set/modify/delete the password used to run the program.
2) Store Passwords: This button is used to set/modify/delete the password used to access the file recording the movement of goods in the store.
3) Daily Store Passwords: This button is used to set/modify/delete the password used to access the file recording the daily movement goods in the store.
4) Sales Center Passwords: This button is used to set/modify/delete the password used to access the file recording the movement of goods in the sales center.
5) Daily Sales Center Passwords: This button is used to set/modify/delete the password used to access the file recording the daily movement goods in the sales center..
6) Previous Page: This button is used to go back to the previous page.
7) End: This button is used to end running the program.

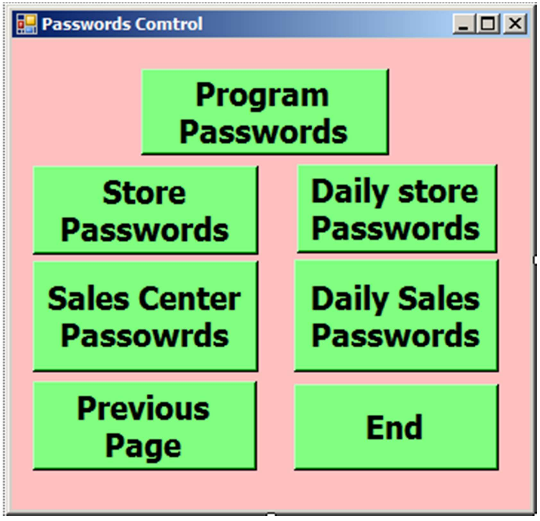


Figure 4. Passwords of the program.

2.2.2. Running the Program

Pressing the "Run The Program" button, the screen in figure (5) appears. This screen has 3 options:

a) Stores: This button is used to record the movement of good in the store.

b) Sales Center: This button is used to record the movement of goods in the sales center.

c) End: This button is used to end running the program.

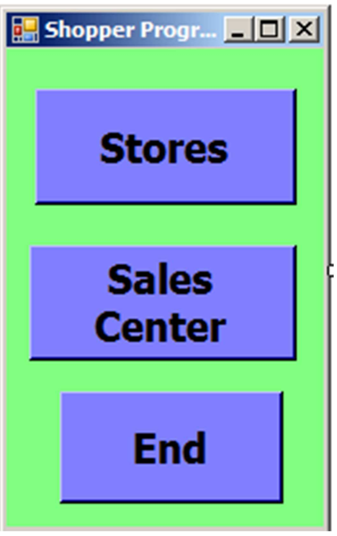


Figure 5. Movement of goods in the store and sales center.

2.2.3. Processes in the Store

Pressing the "Stores" button in figure (5), the screen in figure (6) appears. This screen includes 5 options:

a) New Goods Addition: This button is used to add a new item to the store. The "Name", "Number", "Price", and "Quantity" of the new item are entered as a record. The VB DataGridView tool is used to enter these information. DataGridView tool is shown in figure (7).

b) Modifying Goods Infos: This button is used to modigy the information about an item in the store. Mostly, the "Quantity" is modified when new quantities are brought to the store. The VB DataGridView tool is used to do that modification as in figure (7).

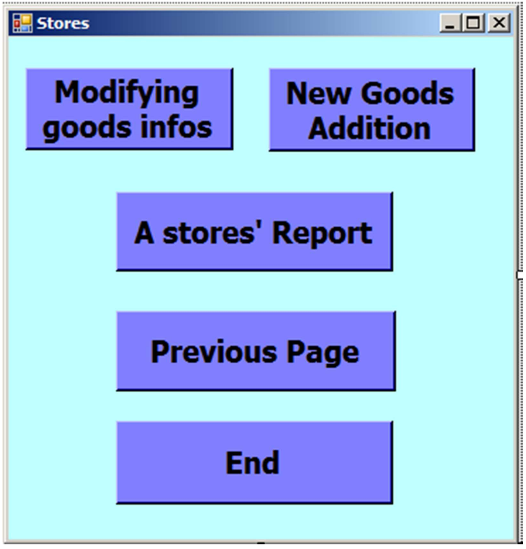


Figure 6. Adding/Modifying goods information in the store.

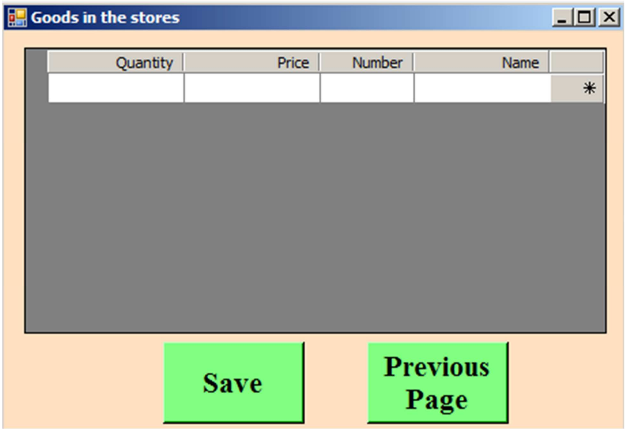


Figure 7. Goods in the store.

c) A Stores Report: This button is used to produce a report about the goods in the store. This report can be saved with any file name chosen by the user for later on printing. Figure (8) shows the screen of the report.

d) Previous Page: This button is used to go back to the previous page.

e) End: This button is used to end running the program.

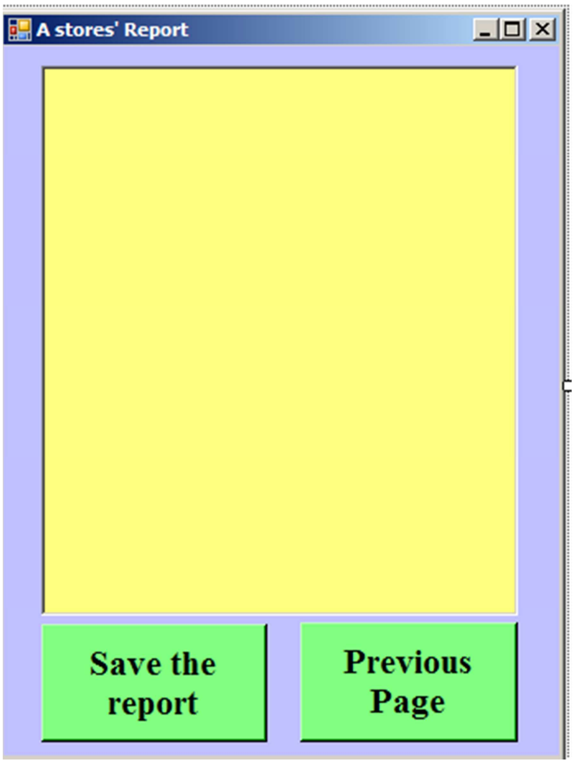


Figure 8. A stores Report.

2.2.4. Processes in the Sale Center

By pressing the button "Sales Center" in figure (5), the screen in figure (9) will appear. This screen has 6 options:

a) A New Customer: This button is used to serve a new customer. The "Name", "Number", Price", and "Quantity" are recorded and the VB DataGridView tool is used for that. Figure (10) shows the screen of serving the customer.

b) A Selling Report: This button is used to make a report about the daily sales in the sales center. The report can be save by any name chosen by the user for a later on printing. Figure (8) shows the report screen.

c) Goods Addition: This button is used to add a new item to the sales center. The "Name", "Number", "Price", and "Quantity" of the new item are entered as a record. The VB DataGridView tool is used to enter the information. DataGridView tool is shown in figure (11).

d) Modifying Goods Infos: This button is used to modify the information about an item in the store. Mostly, the "Quantity" is modified when new quantities are brought to the store. The VB DataGridView tool is used to do that modification as in figure (10).

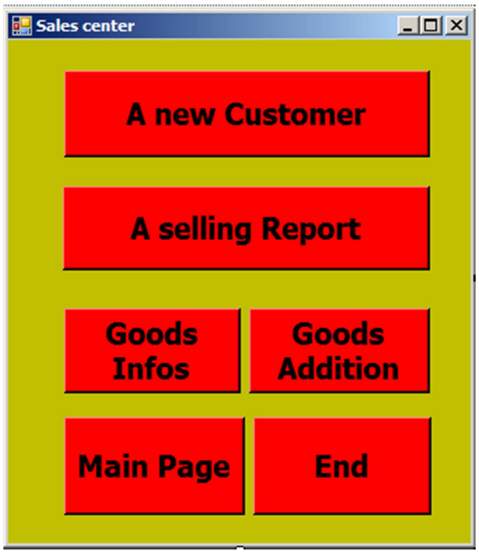


Figure 9. Processes in the sales center.

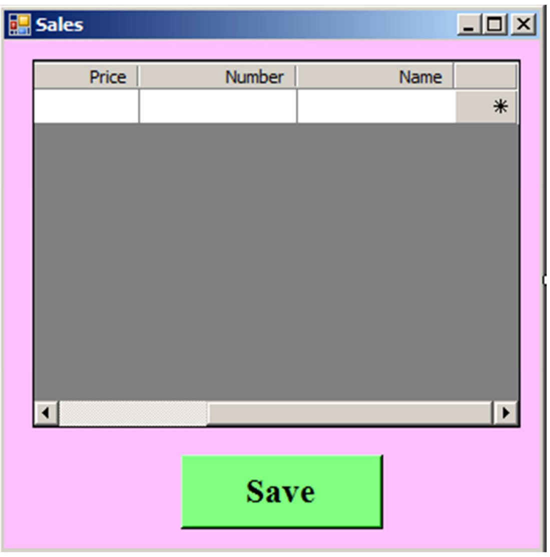
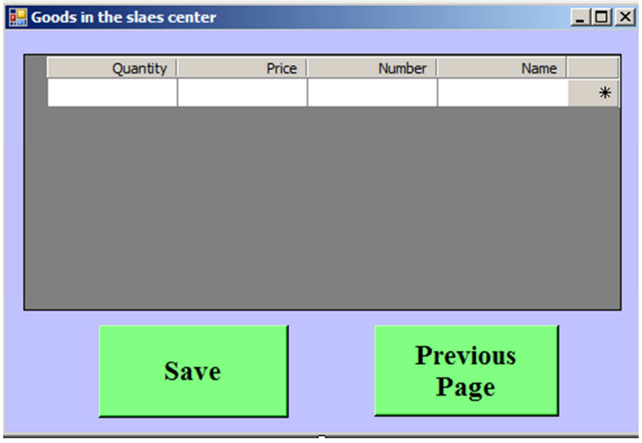


Figure 10. Customer service screen.



Quantity	Price	Number	Name

Figure 11. Goods in the sales center.

3. Security Services for Software Protection

In this section, we discuss a group of the protection tools and techniques that can be used with any software. These tools and techniques include:

a) Password of running software and accessing the different files.

b) Controlling the files and folder attributes such as: 1) hiding, 2) Read only.

c) Encryption of the files' contents.

d) Hashing the files' contents.

e) Signing certain parts in the used files.

In the following subsections, we explain using these services.

3.1. Passwords of Running Software and Accessing Files

Passwords are a tool that protects the access of applications and/or software. Choosing passwords is critical and crucial. Experts advise users of computerized access to select their passwords in clever ways. The following tips are important for password uses [6]:

a) Capital letters and small ones, as well as numbers and symbols should be included in the passwords.

b) Length of the password can as long as it can be and it is advisable to change it from time to time.

c) Passwords must be changed from time to time with short periods of gabs between them.

d) Passwords are not saved in plain text but they are encrypted.

It should be noted that passwords may include symbols and non English letters. This means that a suitable coding is needed. The coding with utf8 is found to be suitable.

3.2. Controlling the Files and Folders Access

Attributes of files and folders control their access. From these attributes: Hiding and ReadOnly attributes. These interesting attributes can be implemented according to the used programming language. In the following, we explain how to control these attributes:

3.2.1. Hide Attribute

When the file attribute is Hidden, the user can not see the file or the folder. This means that no one can access the folders or files contents.

In VB NET, the following code is used to hide the file or folder.

```
Dim attribute As IO.FileAttributes =
IO.FileAttributes.Hidden

System.IO.File.SetAttributes("c:\shopdat\", attribute)
```

3.2.2. Read Only Attribute

When the file attribute is Read Only, the user can open the file and read it. No changes are allowed to be done to the contents of the file. This means that no deletion or addition of any information can happen to the contents.

In VB NET, the following code is used to change the attribute of the file and folder to Read Only.

```
Dim attribute As IO.FileAttributes =
IO.FileAttributes.ReadOnly

System.IO.File.SetAttributes("c:\shopdat\file1.txt",
attribute)
```

3.3. Encryption of Passwords and Contents of Files

Encryption is a security method that protects the information by changing them into un-understood forms. It is known, to the experts, that the most secure cryptographic systems can be rendered completely insecure by a single specification or programming error. No amount of unit testing will uncover a security vulnerability in a cryptosystem. keys can be discovered and consequently the cipher texts can be deciphered. Thus, the difference between a security system and another lies in how long it is needed to break them. Breaking a security system depends on the used keys, as well as the used algorithm. Length of the used key is an important standard that affects the speed of running the software. The speed of software may be also affected by the encryption algorithm. Thus, we compromise between the used keys and algorithms and speed [7].

There are two main types of algorithms for encryption. These are the symmetric key encryption, and the public key encryption. In the following, we describe how to generate the keys in both of the types [6].

3.3.1. Symmetric Encryption Algorithms

For this type of algorithms, one secret key is used for encryption and decryption. The secret key can be generated by using any good random number generator. The output of the random number generator is used as a key to encrypt the text. When we need to decrypt the cipher, we simply run the random number generator and use its output to decrypt the message and obtain the plain text. Thus, if the random number generator is really good, it will give good results. Good results here mean that no way, an attacker can use output random numbers to obtain the original information the generator uses. In this case, we can simply use the XOR operation to encrypt the plain text at the sender and decrypt the cipher at the receiver to obtain the plain text.

However, with computers, we can not pretend that the generated keys are really random, but to be fair, we can say they are pseudo-random keys. Thus, depending on the cycle of the pseudo-random generator, we can assure that the used keys are strong or not.

A good pseudo-random generator can be designed using the following chaotic equation [7, 8]:

$$X_{n+1} = r * x_n * (1 - x_n) \qquad (1)$$

With $3.57 \leq r \leq 4$ and $0 \leq x_0 \leq 1$, the obtained sequence of numbers is really pseudo-random. The equation uses an initial seed x_0 and a bifurcation factor r to compute recursively values x_{n+1}. Knowing the numbers of the sequence does not help any attacker gets the values of r and x_0. Thus, the obtained numbers are used as encryption/decryption keys. With such good keys, we use a block cipher algorithm of the simple XOR operation for encryption/decryption. The files' contents that are used by the program are divided into fixed length blocks. Each block is encrypted using a different key that is generated from the random number generator. Figure (12) gives the block disgram of the encryption process.

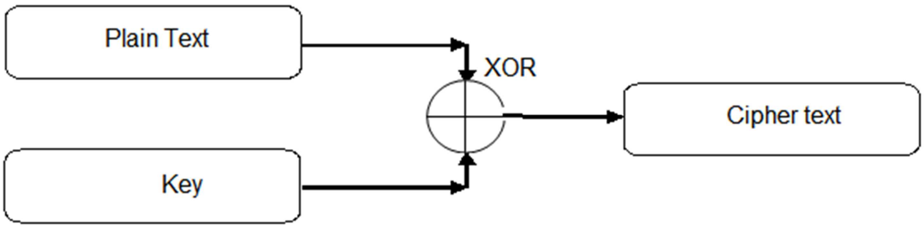


Figure 12. Encryption process.

3.3.2. Public Key Cryptography

For this second type, two keys are used: one of them is kept secret, and the other is published. One key is used for encryption and the other key is used for decryption. The two keys can be gene

Using the pair of public and private keys plays an important role in asymmetric cryptography. The public key is published and used by people while the private key is kept secret and used by the owner. Public key cryptography depends always on the cryptographic algorithm. Thus, a hard mathematical problem lies behind the used algorithm. Examples of such hard problems: the discrete logarithm problem, the elliptic curve problem, … etc. However, the two keys have a very important characteristic. Encrypting a message with the public key is reversed (i.e., decrypted to the same message) by using the private key. Thus, the effect of the public key in the encryption algorithm is cancelled by the effect of the private key in the decryption algorithm. Because of the computational complexity of asymmetric encryption, it is usually used only for small blocks of data, typically exchanging the symmetric cryptography keys, digital signature, message authentication codes [9-11], …etc.

3.3.3. The RSA Public Key Algorithm

The RSA is referred to its inventors; Ron Rivest, Adi Shamir, and Leonard Adleman [Rivest; Shamir; and Adleman, 1978]. It gets its security from the difficulty of factoring large numbers. The public key e and private key d are functions of a pair of large prime numbers p and q (100 to 200 digits or even larger). Recovering the plain-text from its cipher-text and the public key is equivalent to factoring n to its factors p and q [9-11]. The following algorithm is used in RSA [12],

a. Choose p and q

b. Compute n = p * q

c. Compute $\varphi(n) = (p - 1) * (q - 1)$

d. Choose e such that $1 < e < \varphi(n)$ and e and n are co-prime.

e. Compute a value for d such that $(d * e) \% \varphi(n) = 1$.

f. Public key is (e, n)

g. Private key is (d, n)

h. For encryption $C=m^e(\text{mod } n)$ and decryption $m =c^d(\text{mod } n)$

Hence, by following above algorithm the plain text in encrypted form or cipher text and then decrypted from cipher text to plain text.

To generate the two keys, e and d, choose two random prime numbers p and q of equal lengths (for maximum security). Then compute:

$$n=p * q \text{ and } \psi=(p-1) * (q-1) \qquad (2)$$

Randomly select a value e as the encryption public key such that greatest common divisor gcd(e, ψ)=1 and e * d=1 mod ψ where d is the decryption private key. This means that:

$$d=e^{-1} \bmod \psi \qquad (3)$$

Thus e, n are the public keys and d is the private one. The two numbers p and q are no longer needed. The keys e and d can be interchanged.

To encrypt a MAC (M) where $M \leq n$, and M, n are of the same size. The block M is processed according to the following equation:

$$C=M^{e} \bmod n. \qquad (4)$$

The idea here is to transform e into a binary form and then

Example

Assume that M = the symbol (') which has the ascii representation of 96

$$\text{Assume } p=17 \; q=19 \; N = p*q = 17*19 = 323$$

$$\psi=(p-1) * (q-1) = 16*18 = 288 \; e = 13 \; d=(13)^{-1} \bmod 288 = 133$$

In RSA cryptographic algorithm the main disadvantage is its encryption speed. It consumes lot of time to encrypt data. Actually this is disadvantage of asymmetric key algorithms because the use of two asymmetric keys. It provides good level of security but it is slow for encrypting files. Another threat in this algorithm is fake key insertion at decryption level so the secret key should be private and correct to achieve the encryption in successful manner.

Then encrypting M is done as follows:

$$C = (96)^{13} \bmod 323 = (96)1101 \bmod 323 = 96 * 96\text{^}4 * 96\text{^}8 \bmod 323$$

We can notice that the process is simply squaring the first multipliers to get the second multiplier and squaring the second multiplier to get the third multiplier, and so on. Then, for k binary bits the total multiplication value x^y is computed as follows:

$$\text{Assume that } y)_{10} = (b_n \, b_{n-1} \,\ldots\ldots\ldots\, b_2 \, b_1)_2$$

$$x^y \bmod = \text{MULT } (a\text{^}b_i*(2\text{^}(i-1))), = 1, 2, 3, \ldots.., k$$

Thus:

$$96^{13} = 96 * (96\text{^}2 * 96\text{^}2) * (96\text{^}2 * 96\text{^}2)\text{^}2 \bmod 323 = 96 * (9216 * 9216) * (9216 * 9216 * 9216 * 9216) \bmod 323 = 39 * 9216 * 9216 * 9216 * 9216 * 9216 \bmod 323 = 248 * 9216 * 9216 * 9216 * 9216 \bmod 323 = 20 * 9216 * 9216 * 9216 \bmod 323 = 210 * 9216 * 9216 \bmod 323 = 267 * 9216 \bmod 323 = 58$$

Thus, the cipher of (') = (:)

To decrypt a received cipher C compute:

$$M' = C^{d} \bmod n = 58^{133} \bmod 323 = 96$$

The obtained M' is the same as M because

$$C^{d} = (M^{e})^{d} = (M)^{e.d} = (M)^{k(p-1)(q-1)+1} = M. (M)^{k(p-1)(q-1)} = M.1= M.$$

This technique is the easiest one to understand and implement. Although the crypt-analysis neither proved nor disproved RSA security, it does suggest a confidence level in the algorithm. The technique is also applicable to encryption and digital signature. In this case the following points are considered to make the use of RSA algorithm more secure [6]:

a Do not sign a random message received from a stranger and use a one-way hash function first.

b Do not share a common modulus n among a group of users.

c Make sure before encryption that the processed block is smaller than n and they are of the same size. If the last message block size is less than that of n then pad it with random values.

d The decryption exponent should be large.

Note:

a The sender and receiver use their special keys (es, ds, ns) and (er, dr, nr) respectively.

b Care should be taken here such that the large modulo number (of the sender and receiver) is used first.

c Some people have mixed RSA with conventional crypto-systems (e.g. block cipher ones) and digital signatures.

3.4. Hashing of Plain Texts and Message Authentication Code (MAC)

MAC algorithms are symmetric key cryptographic techniques to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key K.

Essentially, a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication.

Figure (13) presents the generation of the MAC and figure

(14) explains how to use that MAC to authenticate the plain text message.

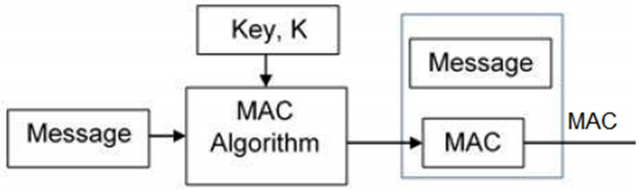


Figure 13. Generating the message authentication code (MAC) at the sender.

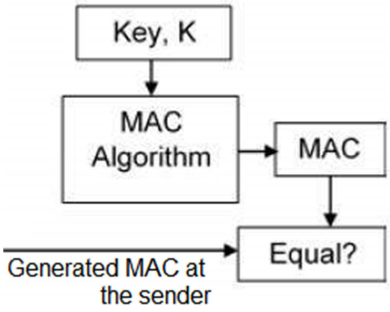


Figure 14. Regenerating MAC at the receiver and authenticating the received message.

3.4.1. The MAC Algorithm

The steps of MAC algorithm are [6]:

a) The sender uses some publicly known MAC algorithm, inputs the message and the secret key K and produces a MAC value.

b) Similar to hash, MAC function also compresses an arbitrary long input into a fixed length output. The major difference between hash and MAC is that MAC uses secret key during the compression.

c) The sender forwards the message plain text along with the MAC for authentication. If confidentiality is required then the message needs encryption.

d) On receipt of the message and the MAC, the receiver feeds the received message and the shared secret key K into the MAC algorithm and re-computes the MAC value.

e) The receiver now checks equality of freshly computed MAC with the MAC received from the sender. If they match, then the receiver accepts the message and assures that the message has been sent by the intended sender.

f) If the computed MAC and the MAC sent by the sender do not match, the receiver cannot determine if the message has been falsified at the origin or altered during transmission.

3.4.2. Limitations of MAC

There are two major limitations of MAC, both are due to its symmetric nature of operation [6]:

A). Establishment of Shared Secret.

1) It can provide message authentication among pre-decided legitimate users who have shared keys.

2) This requires establishment of shared secret channels prior to use of MAC.

B). Inability to Provide Non-Repudiation

1) Non-repudiation is the assurance that a message originator cannot deny any previously sent messages and commitments or actions.

2) MAC technique does not provide a non-repudiation service. If the sender and receiver get involved in a dispute over message origination, MACs cannot provide a proof that a message was indeed sent by the sender.

3) Though no third party can compute the MAC, still sender could deny having sent the message and claim that the receiver forged it, as it is impossible to determine which of the two parties computed the MAC.

Both of these limitations can be overcome by using the public key based digital signatures discussed in following section.

3.5. Signing the MAC with RSA Cryptosystem

Digital signature is a security service that allows people to sign documents electronically. This process needs two keys for the signature, a secret key and a public key. The secret key is used to encrypt the plain text document. Because no one else knows this secret key, it means that the sender has signed the document using his secret key. The public key is used to decrypt the signed document and obtain the plain text one. No other public key can be used to decrypt the signed document except the public key of the sender. This confirms the signature of the sender. Generation of the secret and public keys has its rules and algorithms [9-11]. Figure (15) explains the signature-verification process.

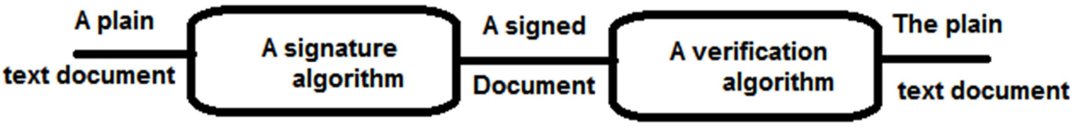


Figure 15. The signature-verification process.

4. Applying the Security Services to the Shopping Software Program

4.1. Preparing the Program for the First Time

a) A test is done to check if the folder "c:\shpdat\" is found or not, if not, then it is created.

b) The password of running the program is set by entering it twice to verify its correctness.

c) A file is opened by the name "file1.txt" and used to save the date and password of running the program, in their plain text and encrypted forms and their hash block.

d) The file (file2.txt – files3.txt – file4.txt) are created to be used like in table (1).

4.2. Encoding Arabic Alphabetics

When the password contains arabic letters, it needs to use the utf-8 coding not the ascii coding. Then, every letter in the password will be transformed into two hexadecimal digits. The followuing code is implemented for coding and decoding Arabic alphabetical strings.

```
Dim utf8Encoding As Encoding = Encoding.UTF8

Dim stringvalue, stringspw, stringsd As String

Dim bytes(300) As Byte

stringvalue = strings

MsgBox("Strings to encode= " & stringvalue)

Dim written As Integer =
utf8Encoding.GetBytes(stringvalue, 0, stringvalue.Length,
bytes, ind)

ind = ind + written

npw = ind

stringspw = ShowByteValues(bytes, ind)

MsgBox("Encoded bytes= " & stringspw)

Dim newStringd As String = utf8Encoding.GetString(bytes,
0, ind)

MsgBox("Decoded= " & newStringd)
```

Example of Using utf-8

Assume the password is: "تأمين برنامج المتجر" meaning in English "Securing the shopper program".

Number of Arabic letters = 17 letters + 2 spaces

Number of bytes = 36

Notice that the space is treated as one byte because it is already included in the ascii code. Thus we got

Encoded bytes = D8 AA D8 A3 D9 8A D9 86 20 D8 A8 D8 B1 D9 86 D8 A7 D9

85 D8 AC 20 D8 A7 D9 84 D9 85 D8 AA D8 AC D8 B1

4.3. The Pseudo-Random Number Generator

The pseudo-random number generator is used to obtain the different keys of operation. This generator uses the values r and x0. The time of logging into the program is used to generate these values. Seconds are divided by 60 to give x0. Minutes and hours are concatenated (as strings) and then taken as a number and divided by 6024. If the result is greater than 0.4 then it is multiplied by 0.4 and added to 3.57 and if it is less than 0.4 it is directly added to 3.57. This gives the value of r. The following recursive equation is computed where x_n is set to x_0. $X_{n+1} = r * x_n * (1 - x_n)$

X_{n+1} is set to a double format and it is treated to obtain the encryption key as follows:

a) The information are divided into fixed length blocks (e.g., L=16 bytes).

b) The right L bytes from xn+1 are taken as the key which we XOR with the plain text to obtain the encrypted cipher.

c) The encrypted block is XORed with the same key to obtain the original plain text.

4.4. Hashing the File Contents

The contents of a file can be hashed as follows:

a) The file contents are divided into fixed length blocks (e.g., L=16 bytes).

b) An initial hash block is generated, from the pseudo-random number generator, and used to hash the blocks with the XOR function.

c) A final block is obtained which is saved as the hash of the file.

To check the file contents for any change, we simply obtain the contents hash and compare it to the one saved in step (3). In case they are equal then the file contents are not changed and if not, then the contents are surely changed.

4.5. Signing the Hash Block

The hash block is signed using the RSA system described in section (2.5). This means that every user of the program will have a private key and a public key. These keys are stored in the file "file1.txt" and the public keys of the users are known to everyone while the private keys are kept secret to their owners. The program checks the signature of the hash by applying the RSA algorithm.

5. Analysis and Discussion of the Software Security

The security of a system is analyzed based on the used encryption algorithm and the key management. In this section we discuss how the current paper took them into consideration for software protection.

5.1. Criteria of a Cryptographic Algorithm

Cryptographic algorithms differ based on certain criteria. One algorithm provides security at the cost of hardware, while another one is reliable but uses more number of keys, and a third one takes more processing time. However, cryptographic algorithms are compared based on [13, 14]

a) Level of Protection: The cryptographic techniques are compared on the basis of computational time and memory, number of users, recovery time from key failure, anti-attack procedures.

b) Complexity: Key generation and encryption techniques are usually based upon the mathematical properties of numbers and functions. Higher order polynomials cause more complexity due to increasing the error probability.

c) Availability: Few encryption algorithms are patented and are not freely publicly available. This puts legal liabilities on choosing the encryption algorithm.

d) Overheads Cryptography requires efforts to fulfill security conditions in: the generation of keys, encryption/decryption, transmission of the messages or saving them. These issues are always translated into overheads including financial overheads, less communication channel bandwidth, heat dissipation affecting processors, power consumption and time delay of processing.

5.2. Key Management

In modern, users have their own keys and it is required to keep these keys and information about their data must be kept secret from each other. In practice, the keys used by the individual user are different from each other. Number of keys, length of keys and their generation and transportation are important factors for the security system [14, 15].

An attacker may succeed to attack the system but fail to expose the information because they are protected. Attacker may then try to crash the information by damaging them. This means, some techniques are needed to protect the information from being crashed. This can be done by protecting the documents and files from being accessed and/or changed. Multiple keys for individual users maybe then a good solution to increase information protection but this increases the calculations and consequently slowdown the speed of information processing.

5.3. For the Current Paper

Securing software lies in a critical area of research. Research has been done to include or embed security services in software programs. Increasing software security reduces its running speed. Practically, people compromise between increasing security and speeding up the program execution speed. However, in the following, we give some of the advantages of using the security services in the present work.

a) The attributes of "Hide" and "Read_only" are used to protect the files and folders from being freely accessed by illegal users.

b) Using the chaotic equation for key generation, pseudo-randomly, makes it hard to guess the passwords and keys. Also, it is hard to collect the generated numbers and reversely find the factors of the chaotic equation.

c) Encoding Arabic alphabets helps the administrator to use Arabic letters in passwords and keys.

d) Encrypting the files' contents makes it hard for to get the plain information of these contents without using the same keys XORed with the cipher. However, using the XOR keeps the encryption/decryption processes fast.

e) Hashing the file contents yields an possibility of checking up the integrity of these contents.

f) Signing the hash block using the RSA algorithm allows us to follow up the users who used the program. It is well known that breaking the RSA system is hard because it depends on reversing the modulo-exponent.

6. Conclusions

Software programs can be secured by applying security services. In this paper, we used a simple software program (the shopper program) and applied some security service to protect it. The software uses 5 files to store its data to follow up the movement of goods in the store and in the shopping place. The first file is used to save the information of the users. The second and third files are used to record the movement of goods in the store and in the shopping place. The fourth and fifth files are used to record the daily movement of goods in the store and the shopping place. Visual Basic Dot NET is used to produce the application.

From the security service applied in this paper: 1) Hiding the file and folders, 2) Read-Only attribution for the files' contents, 3) Fast Stream Cipher Encryption using a random key generator, 4) Hashing the contents of files, 5) Signing the hash of the file.

We mention here that using security services to protect the different operations and processes of the software is a good step towards applying such security services in other applications. A compromise must be discussed when we apply security services to protect the software programs. Speed of the program running is affected by applying the calculations of the security services. Thus, the type and method of the used service would be a challenge for high security applications. That is why we used the XOR encryption/decryption of stream cipher in this software and the used keys were generated using a chaotic pseudo-random number generator which guarantees a good repletion cycle. The RSA is known to be of heavy calculations, and this is the reason why we applied it only to sign the hash block of the file contents.

References

[1] Wang, C.; J. Davidson; J. Hill and J. Knight (2001): "Protection of software-based survivability mechanisms", *The International Conference on Dependable Systems and Networks,* Goteborg, Sweden*, IEEE Press*, pp. 193–205, 2001.

[2] Jan M. Memon Asma Khan and Amber Baig Asadullah Shah (2007): "A Study of Software Protection Techniques", in T. Sobh (ed.), Innovations and Advanced Techniques in Computer and Information Sciences and Engineering, 249–253, 2007, Springer

[3] WANG, P (2005): Tamper resistance for software protection." M. S. thesis, School of engineering Information and Communications University, Daejeon, 2005. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.8556&rep=rep1&type=pdf. Accessed in 1st of July 2016.

[4] Koko, Soheila O. and Amin B. Mustafa (2015): "Comparison of Various Encryption Algorithms and Techniques for improving secured data Communication", Journal of Computer Engineering (IOSR-JCE), Volume 17, Issue 1, Ver. III (Jan – Feb. 2015), PP 62-69.

[5] Vishwa gupta, Gajendra Singh,. Ravindra Gupta (2012): "Advance cryptography algorithm for improving data security", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 1, January 2012.

[6] Stallings, W. (2011): "Cryptography and Network Security Principles and Practice", Prentice Hall, London, Fifth Edition, 2011.

[7] Boneh, D. and Shoup, V. (2015): "A Graduate Course in Applied Cryptography", Available in: https://crypto.stanford.edu/~dabo/cryptobook/draft_0_2.pdf, downloaded at: 1/5/2016.

[8] Rahouma, Kamel (2000), A chaos-based stream cipher algorithm for high speed networks and real time applications, Presented and published in the Applied telecommunication symposium, as a part of the 2000 advanced simulation technologies conference (ASTC2000), Washington, D. C. USA, April 16-20 2000

[9] TutorialsPoint (2015): "Cryptography just for beginners", TutorialsPoint SimplyEasyLearning, ltd. Available at: http://www.tutorialspoint.com/cryptography/cryptography_tutorial.pdf, Downloaded at 1/6/2016.

[10] Smart, N. (2013): "Cryptography: An Introduction", Mcgraw-Hill, 3rd Edition, Ebook (2013). Available at: https://www.cs.umd.edu/~waa/414-F11/IntroToCrypto.pdf, downloaded at 1/6/2015.

[11] Ruohonen, K. (2014): "Mathematical Cryptography: A Translation of Lecture Notes in Finnish Language", Translated by Jussi Kangas and Paul Coughlan. Available at: http://math.tut.fi/~ruohonen/MC.pdf, downloaded at 1/6/2016.

[12] Rajdeep Bhanot and Rahul Hans (2015): "A Review and Comparative Analysis of Various Encryption Algorithms", International Journal of Security and Its Applications Vol. 9, No. 4 (2015), pp. 289-306.

[13] Ajay Kakkar, M. L. Singh, P. K. Bansal (2012): "Comparison of Various Encryption Algorithms and Techniques for Secured Data Communication in Multinode Network", International Journal of Engineering and Technology Volume 2 No. 1, January, 2012.

[14] Ajay Kakkar, Dr. M. L. Singh, Dr. P. K. Bansal (2010): “Efficient Key Mechanisms in Multinode Network for Secured Data Transmission”, International Journal of 92 Engineering Science and Technology, Vol. 2, Issue 5, 2010, pp. 787-795.

[15] Suyash Verma, Rajnish Choubey, Roopali soni (2012): "An Efficient Developed New Symmetric Key Cryptography Algorithm for Information Security", International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 7, July 2012) 18.