Data Encryption and Decryption Using RSA Algorithm in a Network Environment

Nentawe Y. Goshwe.
Department of Electrical/Electronics Engineering
University of Agriculture, Makurdi

Abstract
One of the principal challenges of resource sharing on data communication network is its security. This is premised on the fact that once there is connectivity between computers sharing some resources, the issue of data security becomes critical. This paper presents a design of data encryption and decryption in a network environment using RSA algorithm with a specific message block size. The algorithm allows a message sender to generate a public keys to encrypt the message and the receiver is sent a generated private key using a secured database. An incorrect private key will still decrypt the encrypted message but to a form different from the original message.

Key words:
encryption, decryption, key, Java

1. Introduction

Cryptography is playing a major role in data protection in applications running in a network environment. It allows people to do business electronically without worries of deceit and deception in addition to ensuring the integrity of the message and authenticity of the sender. It has become more critical to our day-to-day life because thousands of people interact electronically every day; through e-mail, e-commerce, ATM machines, cellular phones, etc. This geometric increase of information transmitted electronically has made increased reliance on cryptography and authentication by users [1-4].

Despite the fact that secured communication has existed for centuries, the key management problem has prevented it from commonplace application. The development of public-key cryptography has enabled large-scale network of users that can communicate securely with one another even if they had never communicated before [6-8].

This paper considers a Public Key encryption method using RSA algorithm that will convert the information to a form not understandable by the intruder therefore protecting unauthorized users from having access to the information even if they are able to break into the system.

2. Methodology

There are many ways of classifying data cryptographic algorithms but for the purpose of this paper, they will be classified based on the number of keys that are employed for encryption and decryption. The three common types of algorithms are:

a. Secret Key Cryptography (SKC):
The SKC method uses only a single key for both encryption and decryption. The schemes are generally categorized as being either stream ciphers or block ciphers. Stream ciphers operate on a single bit (byte or computer word) at a time and implement some form of feedback mechanism so that the key is constantly changing while block cipher scheme encrypts one block of data at a time using the same key on each block.
The main drawback of this method is propagation error because a distorted bit in transmission will result in n distorted bits at the receiving side. Though stream ciphers do not propagate transmission errors, they are periodic therefore the key-stream will eventually repeat. This normally results in the use of digital signature mechanisms with either large keys for the public verification function or the use of a TTP.

b. Public Key Cryptography (PKC):
PKC scheme uses one key for encryption and a different key for decryption. Modern PKC was first described using a two-key crypto system in which two parties could engage in a secure communication over a non-secure communications channel without having to share a secret key [5]. In PKC, one of the keys is designated the public key and may be advertised as widely as the owner wants. The other key is designated the private key and is never revealed to another party. RSA is one of the first and still most common PKC implementation that is in use today for key exchange or digital signatures.
The cardinal advantage of this method is that administration of keys on a network requires the presence of only a functionally trusted TTP, as opposed to an unconditionally trusted TTP. Depending on the mode of usage, the TTP might only be required in an “off-line” manner, as opposed to in real time. Many public-key schemes yield relatively efficient signature mechanisms. The key used to describe the public verification function is

Manuscript received July 5, 2013
Manuscript revised July 20, 2013

typically much smaller than for the symmetric-key counterpart [6-9]

c. Hash Functions (HF):

The HF uses a mathematical transformation to irreversibly "encrypt" information. This algorithm does not use keys for encryption and decryption of data. It rather uses a fixed-length hash value which computed based on a plaintext that makes it impossible for either the contents or length of the plaintext to be recovered. These algorithms are typically used to provide a digital fingerprint of a file's content, often used to ensure that the file has not been altered by an intruder or virus. Hash functions are also commonly employed by many operating systems to encrypt passwords to provide some measure of the integrity of a file.

2.1 The RSA Algorithm for Creating RSA Public and Private Key Pair

The RSA algorithm can be used for both key exchange and digital signatures. Although employed with numbers using hundreds of digits, the mathematics behind RSA is relatively straight-forward. To create an RSA public and private key pair, the following steps can be used:

i. Choose two prime numbers, p and q. From these numbers you can calculate the modulus, $n = pq$

ii. Select a third number, *e*, that is relatively prime to (i.e. it does not divide evenly into) the product $(p-1)(q-1)$, the number *e* is the public exponent.

iii. Calculate an integer *d* from the quotient $\frac{(ed-1)}{(p-1)(q-1)}$. The number d is the private exponent.

iv. The public key is the number pair (n, e). Although these values are publicly known, it is computationally infeasible to determine d from n and e if p and q are large enough.

v. To encrypt a message, M, with the public key, creates the cipher-text, C, using the equation: $C = M^e Mod\ n$

vi. The receiver then decrypts the cipher-text with the private key using the equation: $M = C^d Mod\ n$

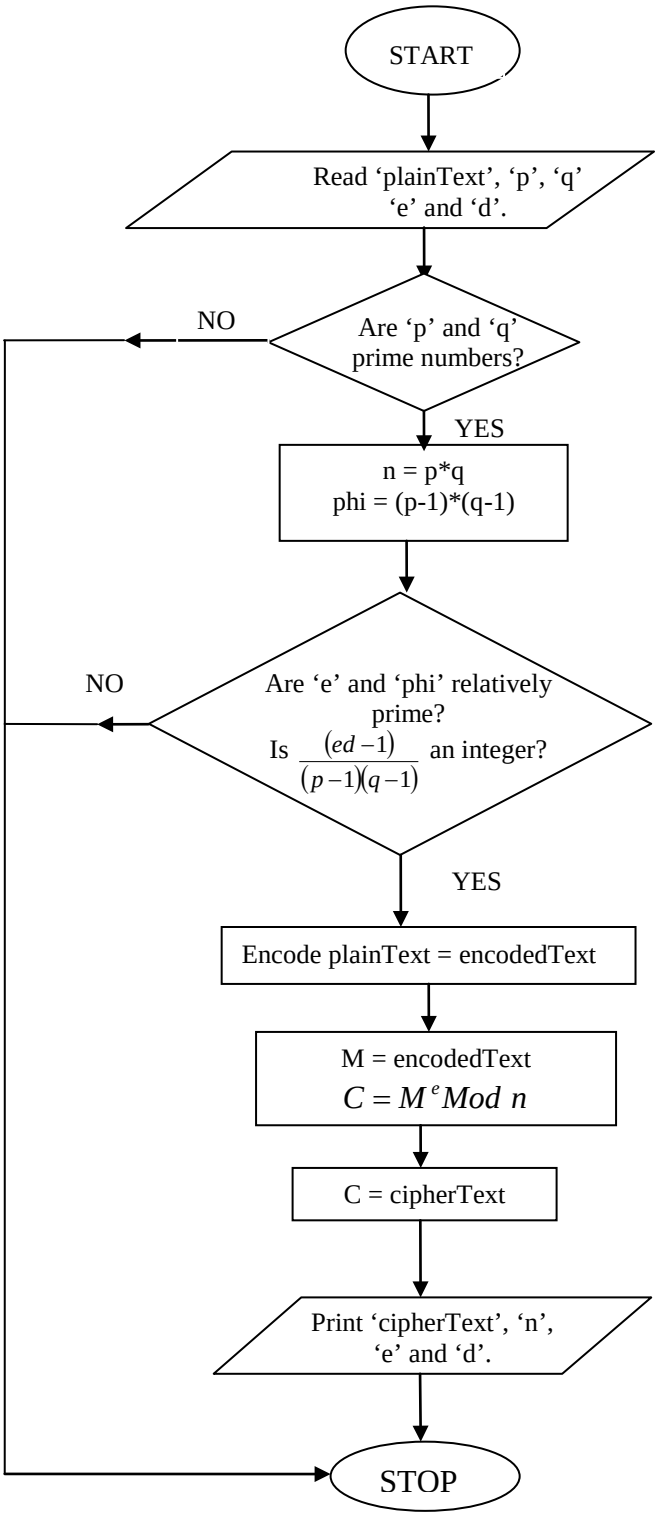

Figure 1.0: A flow chart illustrating the RSA decryption Algorithm

2.1 How to Use the Keys for Encryption

Assuming a sender "A" that wants to send a message to a receiver "B", the sender will take the following steps:-

i. Obtains the recipient B's public key (e, n)
ii. Represents the plaintext message as a positive integer M .
iii. Computes the cipher-text $C = M^e Mod\ n$.
iv. Send the cipher-text C to B.

The flow chart of the encryption algorithm is as given in Figure 1.0.

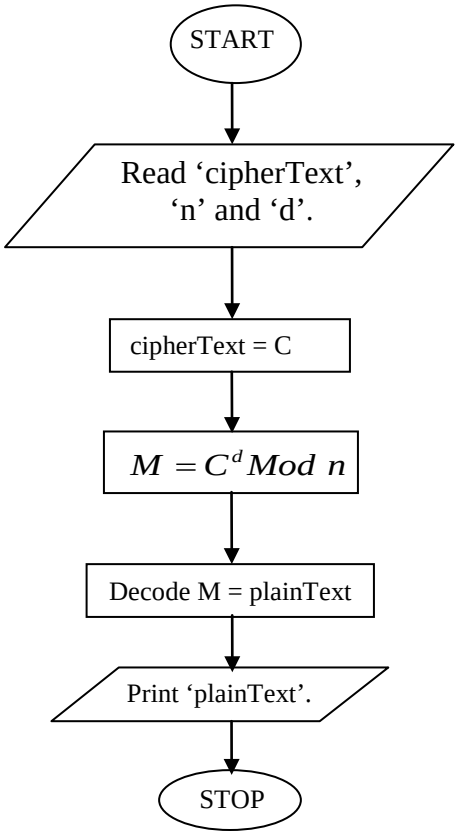

Figure 2.0: A Flow Chart to Illustrate the Decryption Algorithm

2.2 How to Use the Keys for Decryption

For the recipient "B" to receive the message sent by the sender "A", the recipient will take the following steps:-

i. Uses the private key (n, d) to compute $M = C^e Mod\ n$.
ii. Extracts the plaintext from the integer representative M .

This is actually the smallest possible value for the modulus n for which the RSA algorithm works. Figure 2.0 illustrates the decryption procedures.

2.3 The Design of the Unified Modeling Language (UML)

An object programming paradigm (of which java is one of them) uses a unified form of describing each programming steps called Unified Modeling Language (UML). It is a standard notation that originated in the mid-1990s from the work of James Rumbaugh, Ivar Jacobson and Grady Boch. UML is a graphical way of representing and designing an object oriented language for proper description of each step involved and the flow layout of the program itself.

This work chooses to use UML because it has the advantage of clearly showing the relationship that exists between the classes that form this work. There are three packages that exist in this work, they are:

i. The *applicationGUI* Package
ii. The *dbinterface* Package
iii. The *encodinganddecoding* package.

The *applicationGUI* package contains four classes; *MainApp.java*, *ReceiverInterface.java*, *SenderInterface.java* and *TableDisplay.java*. The *dbInterface* package only contains the *RetrieveMessage.java* and the *SendMessage.java* class. The *encodinganddecoding* package contains the *EncodingAndDecoding.java* class. All these packages are embedded in the project named *DataEncryptionAndDecryption*.

In this paper the Top-down approach is used for the design of the program therefore all the small objects are put together to form the main object. The individual classes of these smaller objects are specified with names and are then linked together to form the major object.

The class names for the individual objects are;

i. TableDisplay.java
ii. SendMessage.java
iii. RetrieveMessage.java
iv. EncodingAndDecoding

The necessary java packages were imported while the database was created in *mySQL* with three fields namely: *"Cipher ID"*, *"Cipher Text"* and *"n"* with security administered on it. Frames were created with menus and call the Action*Listener*, *SendMessage* Interface is created with *labels*, *Buttons* and *TextAreas*. This is followed by creation of another *RetrieveMessage* interface with *Label*, *Buttons* and *TextAreas*. And in addition, the Encoding and Decoding class (which is a public class that encodes, decodes, encrypt and decrypt by making use of the *BigInteger*) were also created.

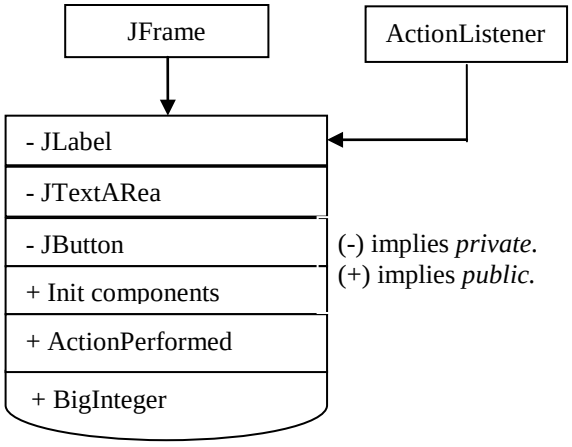

Figure 3.0: A UML to Illustrate the Program Design for the Private and Public Keys

3. Results and Discussion

The program was run and compiled on Windows XP and tested on University of Agriculture, Makurdi local area network which is structured on V-LAN topology. The Graphic User Interface (GUI) is designed to be user friendly and can be used without knowledge of programming in Java.

3.1 The "Sent Message" GUI Output.

Running the program gives a frame with menus that can send or retrieve a message from the database. The sender sent a test message *"my credit card number is 234M99934"*. The Plaintext (*"my credit card number is 234M99934"*) is entered in the plain text area before clicking on encode to convert the text to ASCII code and the result is placed in the Encoded Text as shown in Figure 4.0.

Figure 4.0: The GUI of the Plain Text, Encoded Text and Cipher Text Result.

The Encrypt Button is clicked, a Dialog box appear asking for the value of "q", which must be a prime number and another dialog box prompting for the value of "p" which must also be prime number (the product of "q" and "p" must not have more than 4 digits; which is the specified block-size for this program).

To send the Cipher text to the database, the "Send Button" is clicked and the dialog box returns the value of *n, e* and *d* as 9557, 17 and 7973 respectively.

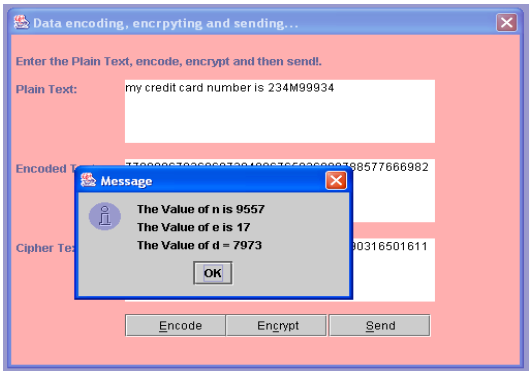

Figure 5.0: A Dialog Box Returning the Values of n, e and d.

At the receiver end, the receiver uses the "Retrieve Message" GUI to request for the value of "cipher ID" from the database. A correct entry of the Cipher ID will return the Cipher text dialog box and requesting for value of "d" for decryption to take place as shown in Figure 5.0. The database identifies the message and the corresponding Cipher ID. This converts the Cipher text to ASCII codes and returns it in the "Encoded Text" box.

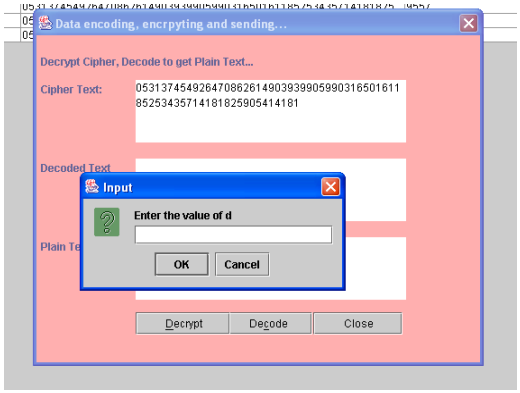

Figure 6.0: A Correct Entry of the Cipher ID with the Requested Value of "d".

For a correct Cipher ID, the dialog box will return the correct Cipher text as shown in Figure 6.0. Correct entry of the value of "d" at the Receiver Interface will return the Correct Retrieved "Plain Text" Message as shown in Figure 7.0.

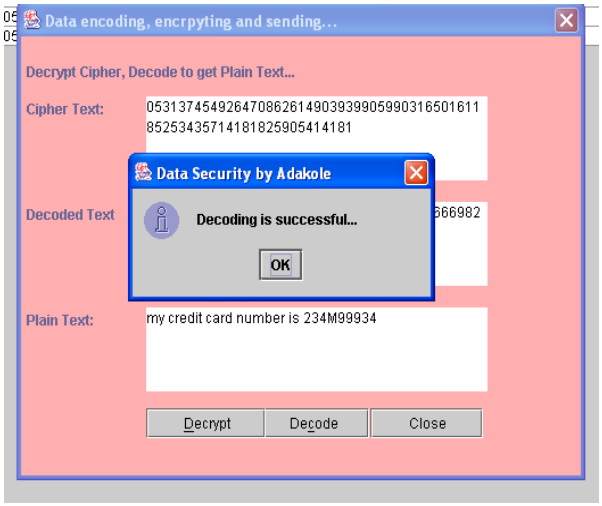

Figure 7.0: Receiver Interface with the Correct Retrieved "Plain Text" Message.

To ensure that the data is secured, for any entry of wrong value of "d", the Receiver Interface will return a meaningless "Plain Text" Message as shown in Figure 8.0. This ensures the data is secured against hackers within the network environment.

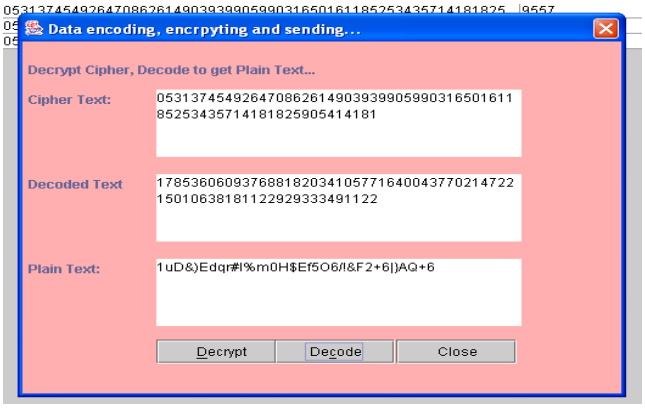

Figure 8.0 Interface Shows a Meaningless Message with Wrong Value of "d".

4. Conclusion

The paper has presented data encryption and decryption in a network environment that was successfully implemented. With this software, data can be transferred from one computer terminal to another via an unsecured network environment. An eavesdropper that breaks into the message will return a meaningless message. Obviously encryption and decryption is one of the best ways of hiding the meanings of a message from intruders in a network environment.

References

[1] Afolabi, A.O and E.R. Adagunodo, 2012. Implementation of an Improved data encryption algorithm in a web based learning system. International Journal of research and reviews in Computer Science. Vol. 3, No. 1.

[2] Bhoopendra, S.R., Prashanna, G. and S. Yadav, 2010. An Integrated encryption scheme used in Bluetooth communication mechanism. International Journal of Computer Tech. and Electronics Engineering (IJCTEE), vol. 1, issue 2.

[3] DI management (2005) "RSA algorithm", available at: http://www.di-mgt.com.au/rsa_alg.html.

[4] Gaurav, S., 2012. Secure file transmission scheme based On hybrid encryption technique. International Journal of management, IT and Engineering. Vol. 2, issue 1.

[5] Hellman, M. and J. Diffie, 1976. New Directions in Cryptography. IEEE transactions on Information theory, vol. IT-22, pp:644-654.

[6] Shinde, G.N. and H.S. Fade War, 2008. Faster RSA algorithm for decryption using Chinese remainder theorem. ICCES, Vol. 5, No. 4, pp. 255-261.

[7] Yang L. and S.H. Yang. 2007. A frame work of security and safety checking for internet-based control systems. International Journal of Information and Computer security. Vol.1, No. 2.

[8] Washington, L.C. 2006. Introduction to Cryptography: with coding theory by Wade Trappe. Upper Saddle River, New Jersey, Pearson Prentice Hall.

[9] Wuling Ren College of Computer and Information Engineering Zhejiang Gongshang University. 2010. A hybrid encryption algorithm based on DES and RSA in Bluetooth communication. Second International Conference on Modeling, Simulation and Visualization methods.

Nentawe Y. Goshwe received B. Engineering in Electrical/ Electronic from University of Agriculture, Makurdi and M. Engineering degrees in Electronics from ATBU Bauchi in 1992 and 2000, respectively and a PhD from University of Nigeria, Nsukka. He Joined the University in 1996 and doubles as both a Lecturer in the Department of Electrical/Electronic Engineering, and Director of Information and Communication Technology, University of Agriculture, Makurdi, Nigeria.