IMAGE ENCRYPTION AND DECRYPTION USING AES ALGORITHM

CHEKKA SATISH KUMAR [#1], V.SARALA [#2]

[#1] MCA Student, Master of Computer Applications,

D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.

[#2] Assistant Professor, Master of Computer Applications,

D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.

Abstract

Cryptography is art of converting one form of data into another form and in turn make the data unreadable for the end users. This cryptography has two methods like encryption and decryption. The process of converting plain text into cipher text is known as encryption and the process of converting cipher text into plain text is known as decryption. In this application we try to use AES algorithm for encryption or decryption of either text or image data

1. INTRODUCTION

The main intention of this to implement file security using the latest and strongest algorithm, named after the founders, after Vincent Rijmen and Joan Daemon, who first published the algorithm in the year 1997. The project is divided into two modules. The first module deals with encryption, the second part deals with decryption.

ENCRYPTION

Encryption is the process of transforming information from an unsecured form ("clear" or "plaintext") into coded information ("cipher text"), that cannot be easily read by outside parties. An algorithm and a key control the transformation process. The process must be reversible so that the intended recipient can return the information to its original, readable form, but reversing the process without the appropriate encryption information should be impossible. This means that details of the key must also be kept secret.

Encryption is generally regarded as the safest method of guarding against accidental or purposeful security breaches. The strength of the encryption method is often measured in terms of work factor - the amount of force that is required to 'break' the encryption. A strong system will take longer to break, although applying greater force can reduce this (the more effort that is put into the attack, the less time required to break the code).

The main characteristics of private key cryptosystem are as follows:

1) In private key encryption, the same key is used for both encryption and decryption. The key must be kept secret so that unauthorized parties cannot, even with knowledge of the algorithm, complete the decryption process.
2) Once the encryption part is carried out, the main next part is the decryption, where the cipher text has to be converted back into the original text, so that whole process of file transfer is implemented. The receiver is interested in receiving only the original text, therefore decryption plays a vital role in this project.

The main problems that are dealt in the APTS, which mainly works on projects that deal with communication, are given below in detail. The need of the hour was to implement algorithms like Rijndeal so that security over the data transmitted could be assured. Yet another factor was the efficiency that this algorithm supported.

Types and Sources of File Threats

1) Unauthorized Access

"Unauthorized access" is a very high-level term that can refer to a number of different sorts of attacks. The goal of these attacks is to access some resource that your machine should not provide the attacker.

2) Executing Commands Illicitly

It's obviously undesirable for an unknown and untrusted person to be able to execute commands on your server machines. There are two main classifications of the severity of this problem: normal user access, and administrator access. A normal user can do a number of things on a system (such as read files, mail them to other people, etc.) that an attacker should not be able to do.

On the other hand, an attacker might wish to make configuration changes to a host (perhaps changing its IP address, putting a start-up script in place to cause the machine to shut down ever time it's started or something similar). In this case, the attacker will need to gain administrator privileges on the host.

3) Confidentiality Breaches

There is certain information that could be quite damaging if it fell into the hands of a competitor, an enemy, or the public. In these cases, it’s possible that compromise of a normal user’s account on the machine can be enough to cause damage (perhaps in the form of PR, or obtaining information that can be used against the company, etc.)

While many of the perpetrators of these sorts of break-ins are merely thrill-seekers interested in nothing more than to see a shell prompt for your computer on their screen, there are those who are more malicious.

4) Destructive Behavior

Among the destructive sorts of break-ins and attacks, one of the two major categories is.

2. LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, ten next steps are to determine which operating system and language used for developing the tool. Once the programmers start building the tool, the programmers need lot of external support. This support obtained from senior programmers, from book or from websites. Before building the system the above consideration r taken into for developing the proposed system.

RELATED WORK

The input, the output and the cipher key for Rijndael are each bit sequences containing 128, 192 or 256 bits with the constraint that the input and output sequences have the same length. A bit is a binary digit, 0 or 1, while the term ‘length’ refers to the number of bits in a sequence. In general the length of the input and output sequences can be any of the three allowed values but for the Advanced Encryption Standard (AES) the only length allowed is 128. However, both Rijndael and AES allow cipher keys of all three lengths. The individual bits within sequences will be enumerated starting at zero and increasing to one less than the length of the sequence. The number *i* associated with a bit, called its index, is hence in one of the three ranges 0 £ *i* < 128, 0 £ *i* < 192 or 0 £ *i* < 256 depending on the length of the particular sequence in question.

Bytes

A **byte** in Rijndael is a group of 8 bits and is the basic data unit for all cipher operations. Such bytes are interpreted as finite field elements using polynomial representation, where a byte *b* with bits *b*0 *b*1 … *b*7 represents the finite field element:

$$b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0=\sum_{i=0}^{7}b_ix^i$$

The values of bytes will be presented in binary as a concatenation of their its (0 or 1) between braces. Hence {011000011} identifies a specific finite field element. Unless specifically indicated, bit patterns will be presented with higher numbered bits to the left. It is also convenient to denote byte values using hexadecimal notation, with each of two groups of four bits being denoted by a character as Follows.

bit pattern	character
0000	0
0001	1
0010	2
0011	3

bit pattern	character
0100	4
0101	5
0110	6
0111	7

bit pattern	character
1000	8
1001	9
1010	a
1011	b

bit pattern	character
1100	c
1101	d
1110	e
1111	f

Hence the value {011000011} can also be written as {63}, where the character denoting the 4-bit group containing the higher numbered bits is again to the left.

Some finite field operations utilize a single additional bit (*b*8) to the left of an 8-bit byte. Where this bit is present it will appear immediately to the left of the left brace, for example, as in 1{1b}.

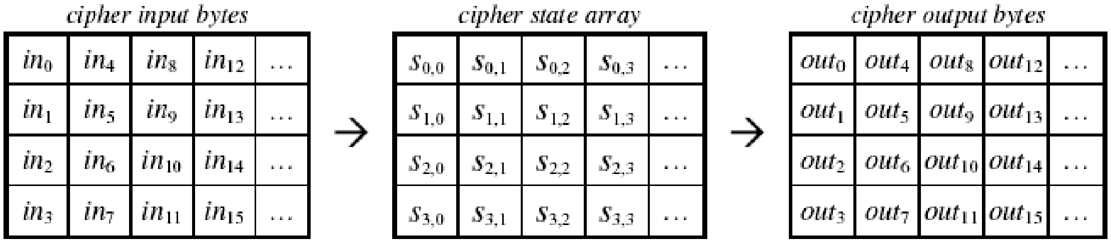
Arrays of Bytes

All input, output and cipher key bit sequences are represented as one-dimensional arrays of bytes where byte *n* consists of bits 8*n* to 8*n*+7 from the sequence with bit 8*n*+*i* in the sequence mapped to bit 7-*i* in the byte for 0 <= □*i* < 8. For a sequence denoted by the symbol *a*, the *n*'th byte will be referred to using either of the two notations *an* or *a*[*n*], with *n* in one of the ranges 0 <=□*n* < 16, 0 <=□*n* < 24 or 0 <=□*n* < 32.

The Rijndael State

Internally Rijndael operates on a two dimensional array of bytes called the **state** that contains 4 rows and *Nc* columns, where *Nc* is the input sequence length divided by 32. In this state array, denoted by the symbol *s*, each individual byte has two indexes: its row number *r*, in the range 0 <=□*r* < 4, and its column number *c*, in the range 0 <=□*c* < *Nc*, hence allowing it to be referred to either as *c r s* , or *s*[*r*, *c*]. For AES the range for *c* is 0 <=□*c* < 4 since *Nc* has a fixed value of 4.

At the start (end) of an encryption or decryption operation the bytes of the cipher input (output) are copied to (from) this state array in the order shown in Figure 1.



cipher input bytes

in_0	in_4	in_8	in_{12}	…
in_1	in_5	in_9	in_{13}	…
in_2	in_6	in_{10}	in_{14}	…
in_3	in_7	in_{11}	in_{15}	…

cipher state array

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$	…
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	…
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$	…
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$	…

cipher output bytes

out_0	out_4	out_8	out_{12}	…
out_1	out_5	out_9	out_{13}	…
out_2	out_6	out_{10}	out_{14}	…
out_3	out_7	out_{11}	out_{15}	…

Figure 1 – Input to, and output from, the cipher state array

Hence at the start of encryption or decryption the input array *in* is copied to the state array according to the scheme:

s[*r*, *c*] = *in*[*r* + 4*c*] for 0 £ *r* < 4 and 0 £ *c* < *Nc*

and when the cipher is complete the state is copied to the output array *out* according to:

out[*r* + 4*c*] = *s*[*r*, *c*] for 0 £ *r* < 4 and 0 £ *c* < *Nc*

Arrays of 32-bit Words

The four bytes in each column of the state can be thought of as an array of four bytes indexed by the row number *r* or as a single 32-bit **word** (bytes within all 32-bit words will always be enumerated using the index *r*). The state can hence be considered as a one dimensional array of words for which the column number *c* provides the array index. The key schedule for Rijndael, described below, is an array of 32-bit words, denoted by the symbol *k*, with the lower elements initialized from the cipher key input so that byte 4*i*+*r* of the key is copied into byte *r* of key schedule word *k*[*i*]. The cipher iterates through a number of cycles, called **rounds**, each of which uses *Nc* words from this key schedule. Hence the key schedule can also be viewed as an array of **round keys**, each of which consists of an *Nc* word sub-array. Hence word *c* of round key *n*, which is *k*[*Nc* * *n* + *c*], will also be referred to using two dimensional array notation as either *k*[*n*,*c*] or *kn*,*c* . Here the round key for round *n* as a whole, an *Nc* word sub-array, will sometimes be referred to by replacing the second index with '-' as in *k*[*n*,-] and - , *n k* .

3. EXISTING SYSTEM

All the existing systems failed to provide security for the data in terms of encryption and decryption. In general the data which is send from sender to receiver will be send in the form of plain text and hence there is a lack of security in the primitive methods.

LIMITATION OF EXISTING SYSTEM

The following are the limitations that take place in the existing system. They are as follows:

1) No Security

2) All the files are open access

3) Data can be viewed by any one

4. PROPOSED SYSTEM

Hence in this current project, we try to add a new level of security for the data which is transferred from one location to another by using cryptography algorithm AES. The process of converting plain text into cipher text is known as encryption and the process of converting cipher text into plain text is known as decryption. In this application we try to use AES algorithm for encryption or decryption of either text or image data.

ADVANTAGES OF THE PROPOSED SYSTEM

The following are the advantages of the proposed system, they are as follows:

1. The data is secure in this proposed method.

2. All the users cannot able to access the information directly because it is encrypted.

3. Only the valid users can only decrypt the data by using the decryption key.

5. SOFTWARE PROJECT MODULES

Implementation is the stage where the theoretical design is converted into programmatically manner. In this stage we will divide the application into a number of modules and then coded for deployment. We have implemented the proposed concept on Java programming language with J2EE as the chosen language in order to show the performance this proposed protocol. The proposed application is divided into mainly 3 modules such as

1) Data Encryption Module

2) Data Decryption Module

3) Choose Properties Module

5.1 Data Encryption Module

In this module, the data user can choose the input data what he want to encrypt.The data can be either of text message or image or any text related files.Once he choose input then he need to set the properities like key size for encryption and then press on encrypt button.In the process of emmcryption he is asked to choose the valid password which is of min 4 to max 8 characters length.

5.2 Data Decryption Module

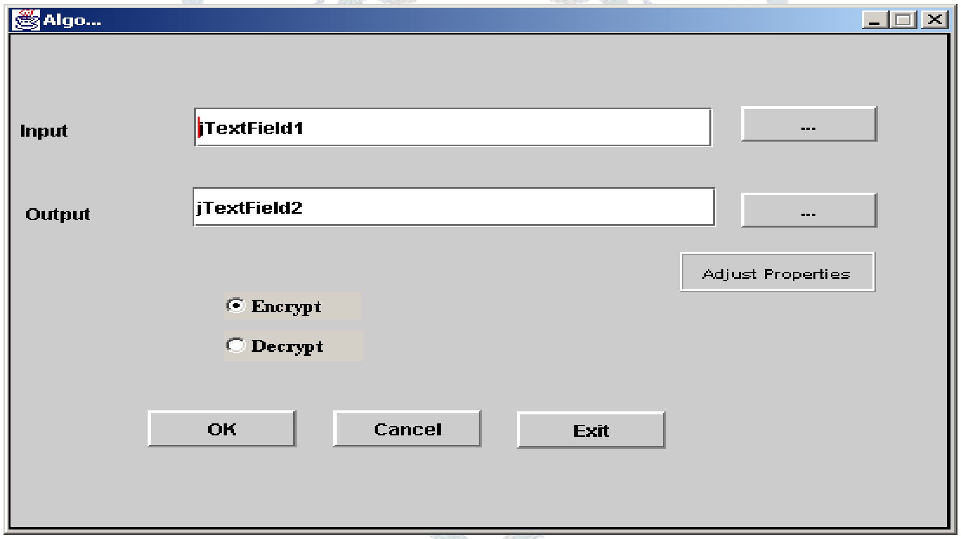
In this module the user will try to upload the encrypted file into application as input and he will now asked to enter the resultant password for decrypting the file.If the data user substitute the valid password then data can be decrypted and displayed as plain text manner.If the password is not substituted correctly the data cannot be decrypted into plain text manner.

5.3 Choose Parameters Module

During the process of encrypting or decrypting the data,the user is asked to choose parameters such as Key Size and Length.In general each and every cryptography algorithm has individual key size and length.So in this module we try to set three key sizes such as 128 bits,192 and 256 bits.So depend on the user requirement the key size is selected.

6. EXPERIMENTAL RESULTS

MAIN WINDOW



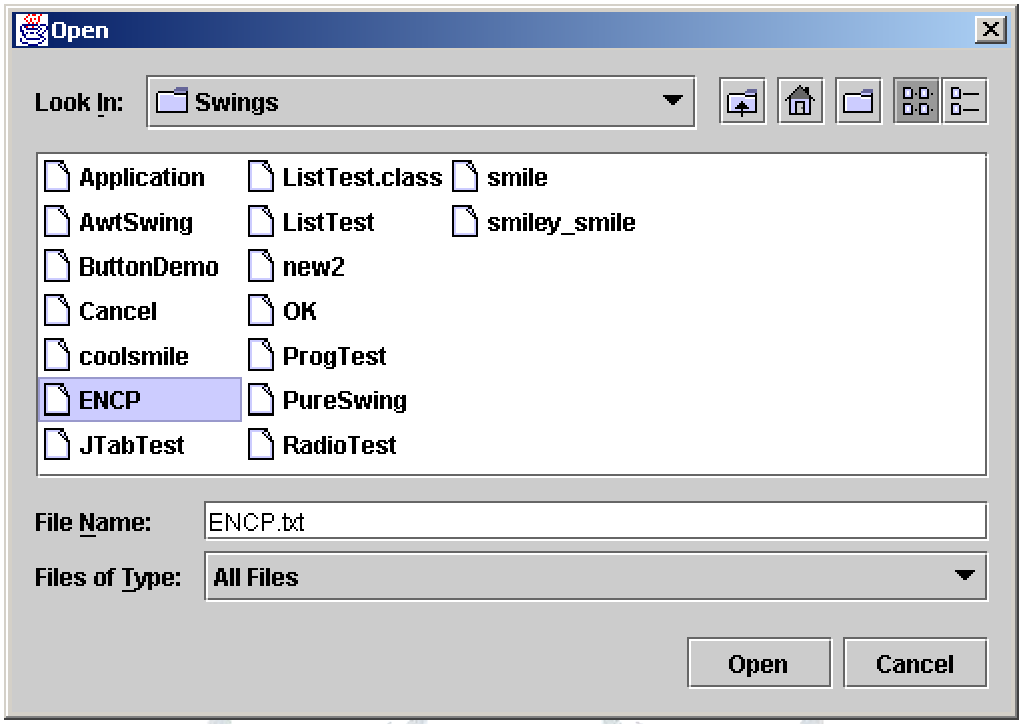
Input jTextField1

Output jTextField2

● Encrypt

○ Decrypt

USER CHOOSE FILE TO ENCRYPT

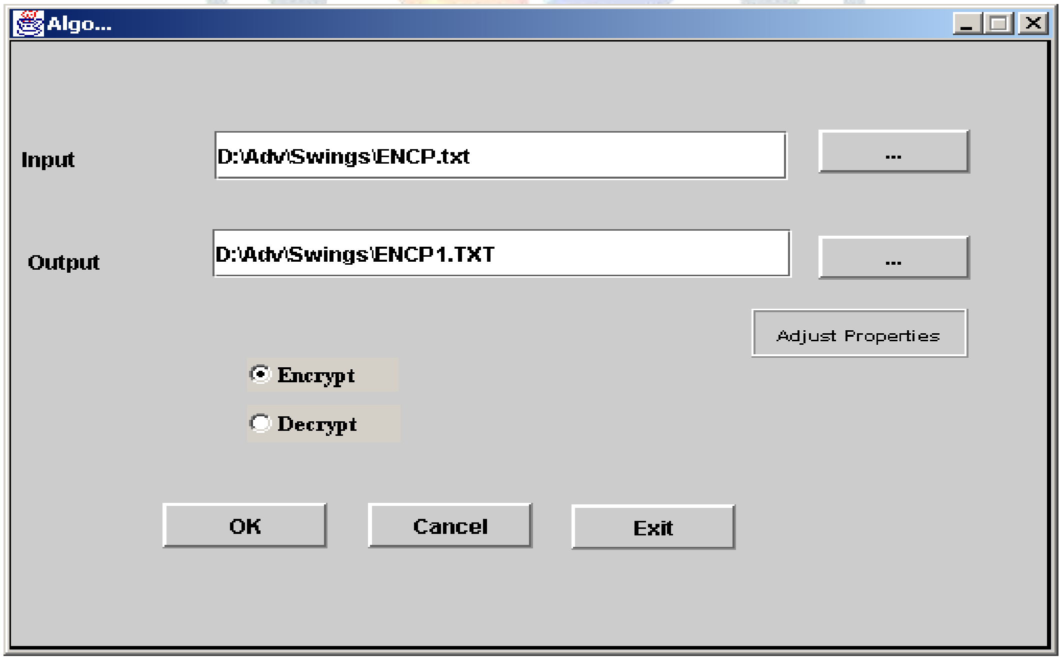


Look In: Swings

File Name: ENCP.txt

Files of Type: All Files

USER CHOOSE OUTPUT FILE NAME AFTER ENCRYPTION



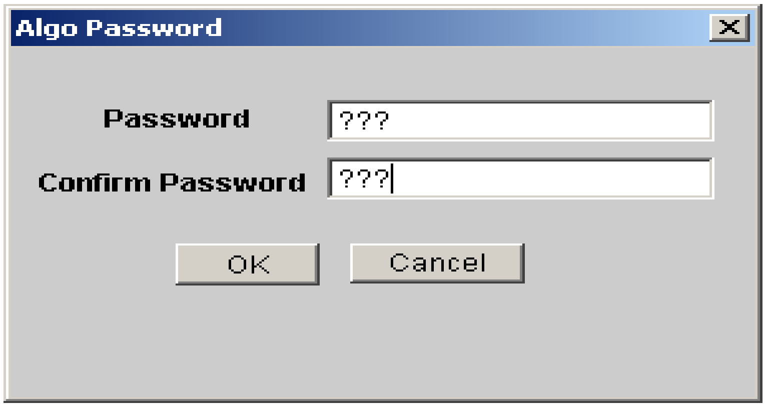
Input D:\Adv\Swings\ENCP.txt

Output D:\Adv\Swings\ENCP1.TXT

● Encrypt

○ Decrypt

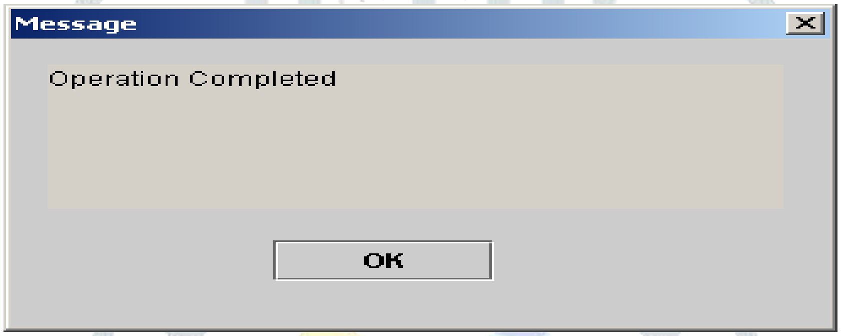
USER CHOOSE PASSWORD AND CONFORMATION PASSWORD FOR ENCRYPTION



Password ???

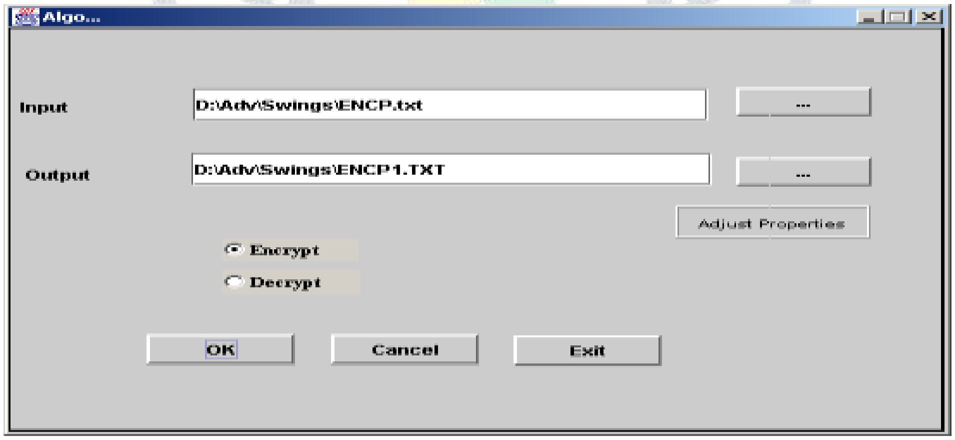
Confirm Password ???

USER GET CONFORMATION MESSAGE AFTER ENCRYPTION



Operation Completed

RECEIVER USER CHOOSE DECRYPTION PROCESS

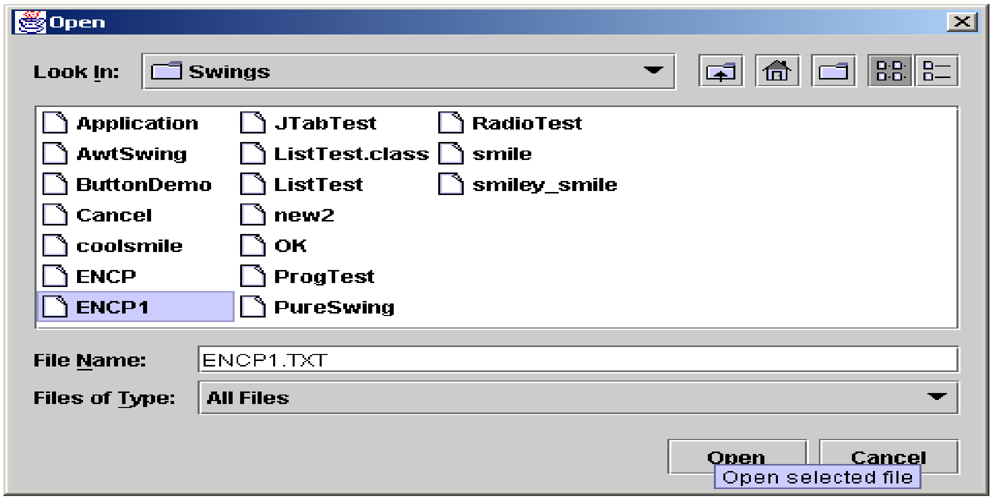


Input D:\Adv\Swings\ENCP.txt

Output D:\Adv\Swings\ENCP1.TXT

● Encrypt

○ Decrypt

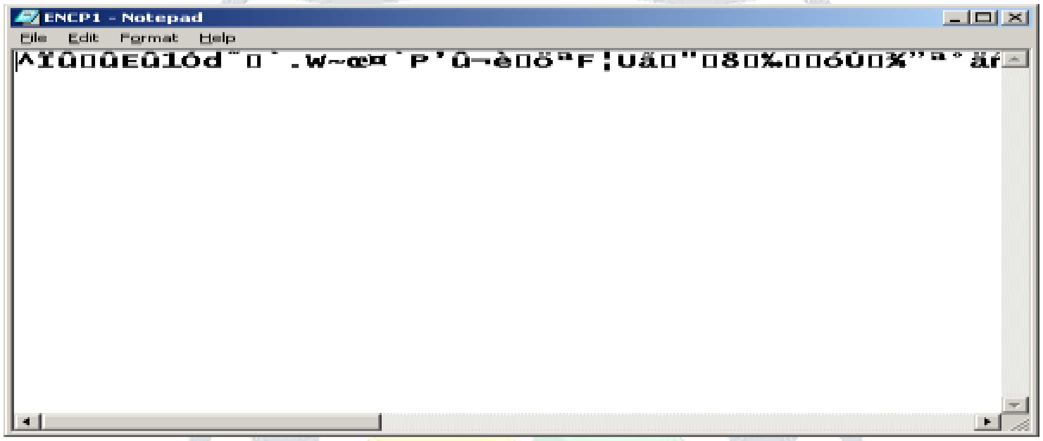


Look In: Swings

File Name: ENCP1.TXT

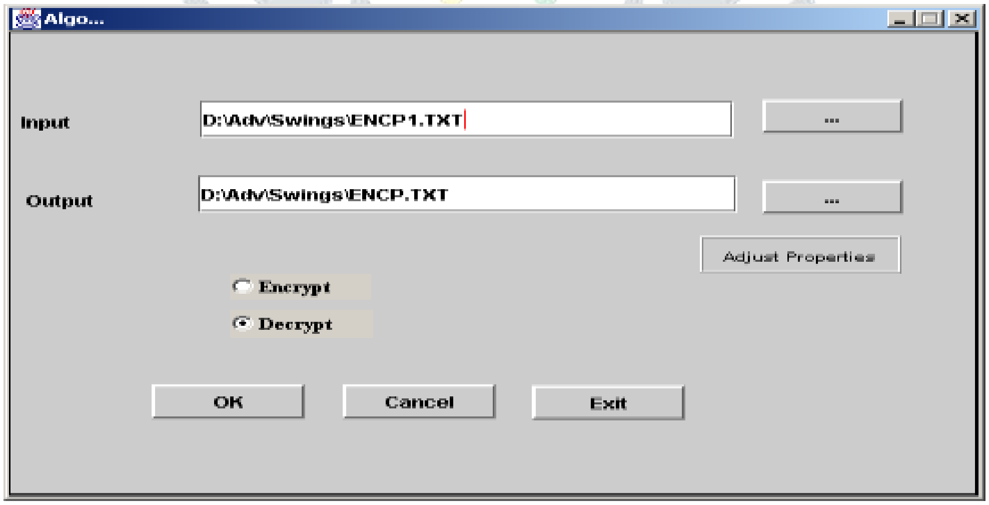
Files of Type: All Files

USER TRY TO OPEN THAT ENCRYPTED FILE CONTENT



```
^ÏÛ□ÛEÛ1ód¯□`.W~œ¤`P'Û¬è□ö ªF¦Uä□"□8□‰□□óÙ□%''ª°äŕ
```

USER CHOOSE CORRECT PASSWORD FOR DECRYPTION

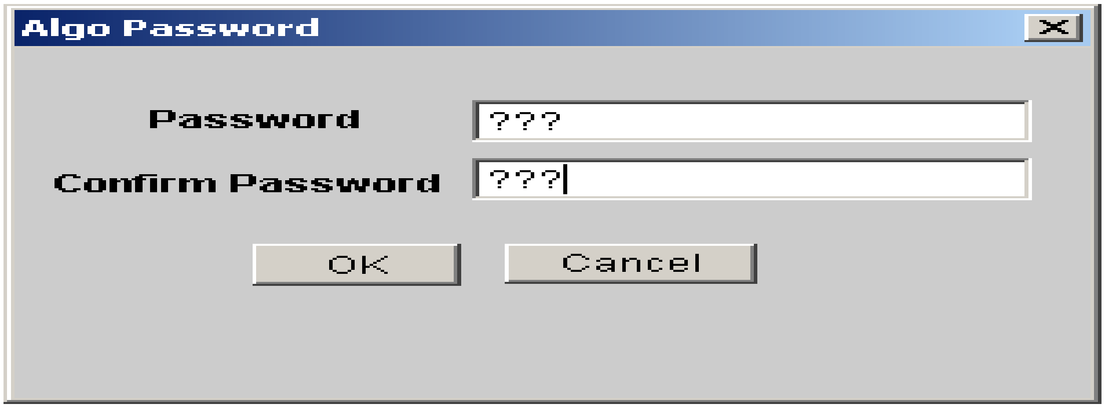


Input D:\Adv\Swings\ENCP1.TXT

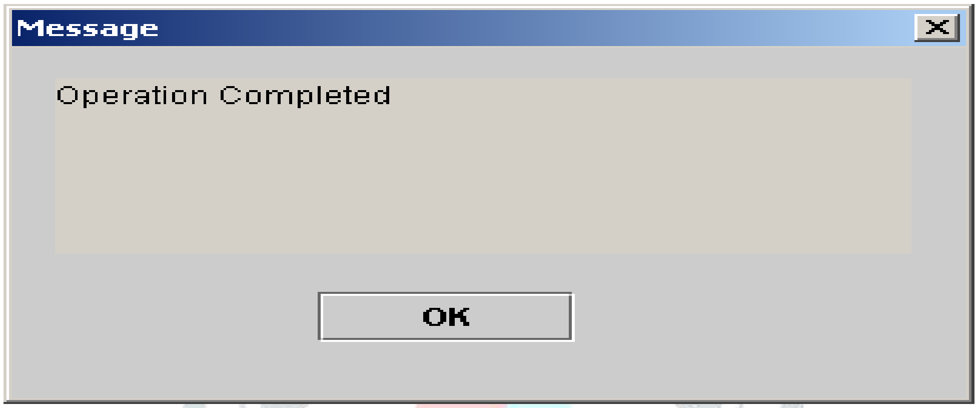
Output D:\Adv\Swings\ENCP.TXT

○ Encrypt

● Decrypt

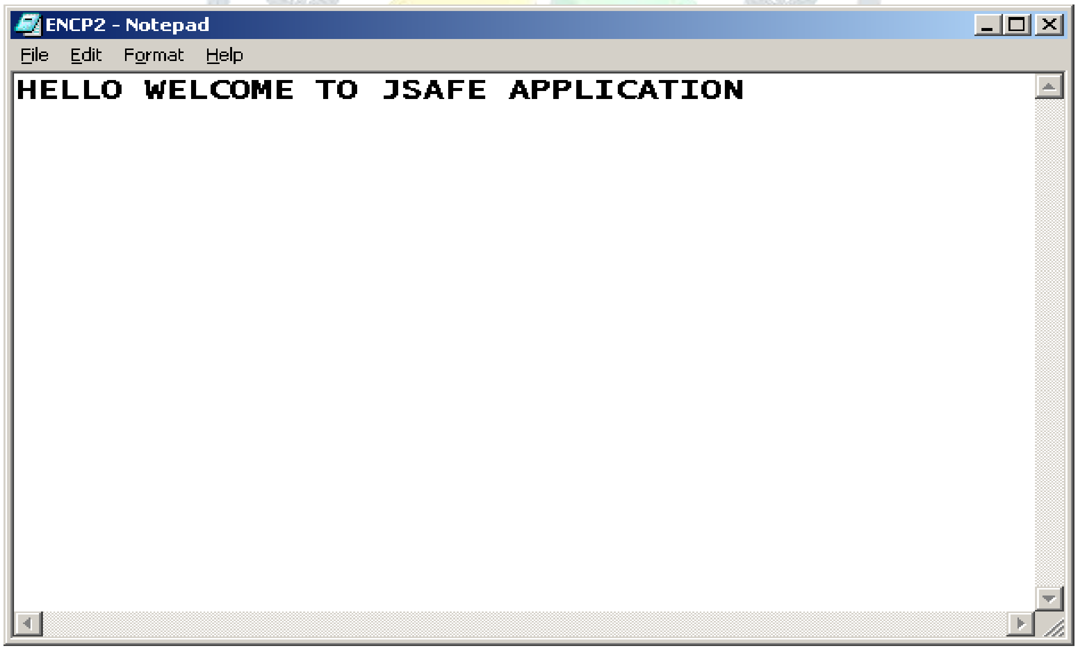


USER GET CONFORMATION



Operation Completed

USER GETS THE FILE DECRYPTED PROCESS



```
HELLO WELCOME TO JSAFE APPLICATION
```

7 . CONCLUSION

The cipher has a variable block length and key length. We currently specified how to use keys with a length of 128, 192, or 256 bits to encrypt blocks with length of 128, 192 or 256 bits (all nine combinations of key length and block length are possible). Both Block length and key length can be extended very easily to multiples of 32 bits. Rijndael can be implemented very efficiently on a wide range of processors and in hardware

8. REFERENCES

[1] Z. J. Haas, J. Deng, B. Liang, P. Papadimitratos, and S. Sajama, "Wireless ad hoc networks," *Encycl. Telecommun.*, 2002.

[2] S. Yousefi, M. S. Mousavi, and M. Fathy, "Vehicular ad hoc networks (VANETs): challenges and perspectives," in *ITS Telecommunications Proceedings, 2006 6th International Conference on*, 2006, pp. 761–766.

[3] H. Deng, W. Li, and D. P. Agrawal, "Routing security in wireless ad hoc networks," *IEEE Commun. Mag.*, vol. 40, no. 10, pp. 70–75, 2002.

[4] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Comput. networks*, vol. 47, no. 4, pp. 445–487, 2005.

[5] V. Balakrishnan and V. Varadharajan, "Packet drop attack: A serious threat to operational mobile ad hoc networks," in *Proceedings of the International Conference on Networks and Communication Systems (NCS 2005), Krabi*, 2005, pp. 89–95.

[6] M. Peng, W. Shi, J.-P. Corriveau, R. Pazzi, and Y. Wang, "Black hole search in computer networks: State-of-the-art, challenges and future directions," *J. Parallel Distrib. Comput.*, vol. 88, pp. 1–15, 2016.
[7] J.-M. Chang, P.-C. Tsou, I. Woungang, H.-C. Chao, and C.-F. Lai, "Defending against collaborative attacks by malicious nodes in MANETs: A cooperative bait detection approach," *IEEE Syst. J.*, vol. 9, no. 1, pp. 65–75, 2015.

[8] A. Aijaz and A. H. Aghvami, "Cognitive Machine-to-Machine Communications for Internet-of-Things: A Protocol Stack Perspective," *IEEE Internet Things J.*, vol. 2, no. 2, pp. 103–112, 2015.

[9] P. Chen, S. Cheng, and K. Chen, "Information Fusion to Defend Intentional Attack in Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 337–348, 2014.

[10] X. Meng and T. Chen, “Event-driven communication for sampled-data control systems,” *Am. Control Conf. (ACC), 2013*, no. 1, pp. 3002–3007, 2013.

[11] F. Razzak, “Spamming the Internet of Things: A possibility and its probable solution,” *Procedia Comput. Sci.*, vol. 10, pp. 658–665, 2012.

[12] J.-H. Cho, R. Chen, and K. S. Chan, “Trust threshold based public key management in mobile ad hoc networks,” *Ad Hoc Networks*, vol. 44, pp. 58–75, 2016.

[13] J. Friginal, D. de Andrés, J.-C. Ruiz, and M. Martínez, “REFRAHN: a resilience evaluation framework for ad hoc routing protocols,” *Comput. Networks*, vol. 82, pp. 114–134, 2015.

[14] L. H. G. Ferraz, P. B. Velloso, and O. C. M. B. Duarte, “An accurate and precise malicious node exclusion mechanism for ad hoc networks,” *Ad hoc networks*, vol. 19, pp. 142–155, 2014.

[15] H. Xia, Z. Jia, X. Li, L. Ju, and E. H.-M. Sha, “Trust prediction and trust-based source routing in mobile ad hoc networks,” *Ad Hoc Networks*, vol. 11, no. 7, pp. 2096–2114, 2013.

[16] D. Cheelu, M. R. Babu, and P. Venkatakrishna, “A fuzzy-based intelligent vertical handoff decision strategy with maximised user satisfaction for next generation communication networks,” *Int. J. Process Manag. Benchmarking*, Dec. 2013.

[17] R. V Boppana and X. Su, “On the effectiveness of monitoring forintrusion detection in mobile ad hoc networks,” *IEEE Trans. Mob. Comput.*, vol. 10, no. 8, pp. 1162–1174, 2011.

[18] Y. Yu, L. Guo, X. Wang, and C. Liu, “Routing security scheme based on reputation evaluation in hierarchical ad hoc networks,” *Comput. Networks*, vol. 54, no. 9, pp. 1460–1469, 2010.

[19] A. Khan, T. Suzuki, M. Kobayashi, W. Takita, and K. Yamazaki, “Packet size based routing for stable data delivery in mobile ad-hoc networks,” *IEICE Trans. Commun.*, vol. 91, no. 7, pp. 2244–2254, 2008.

[20] N. Komninos, D. Vergados, and C. Douligeris, “Detecting unauthorized and compromised nodes in mobile ad hoc networks,” *Ad Hoc Networks*, vol. 5, no. 3, pp. 289–298, 2007.