THE EXPERIMENT
INTERNATIONAL JOURNAL OF SCIENCE AND TECHNOLOGY

ISSN-2319-2119

RESEARCH ARTICLE

S.Bhuvaneswari et al, The *Experiment, 2013 Vol. 15(1), 1050-1055*

RIJNDAEL SECURE SYSTEM

ABSTRACT

This application provides security at file content level. It uses Cryptosystems which are highly technical systems that provide privacy through secret encoding using Rijndael algorithm which has been an important part of the electronic information transfer. This cryptosystem protects data by using hardware and software renovation in a process that protects data by performing mathematical operations/algorithms on it. The result cipher text can be even then transmitted over insecure lines or through wider networks. If this cipher text is intercepted by hackers, it is indecipherable and meaningless to him on to the strong rounds of Rijndael cipher active algorithm. When the cipher text reaches its final destination, it can be decrypted into the original state of the data only by the authentic users.

1. INTRODUCTION

Security is a great necessity in data operations today. The authentication process or commerce exchanges need security and reliability. There are several ways to guarantee the operation of these systems with security. Cryptography is one option and is very used today in many applications.

Cryptographic services are required across variety of platforms in a wide range of applications such as secure access to private networks, electronic commerce and health care. Cryptography means hidden writing, the practice of using encryption to conceal text. The security of conventional encryptions depends on several factors.

First, the encryption algorithm must be powerful enough that is impractical to decrypt a message on the basis of cipher text alone. Beyond that, the security depends on the secrecy of the key, not the secrecy of the algorithm. That is, it is assumed that is also impractical to decrypt a message on the basis of the cipher text plus knowledge of the encryption or decryption algorithm.

Cryptography needs a standard to allow the communication of both sides. In 1997 a new symmetric algorithm was defined by National Institute of Standard and Technology (NIST) as the Advanced Encryption Standard (AES). The winner of contest was Rijndael.

The hardware implementation of Rijndael could provide either high performance or low cost for specific applications. At backbone communication channels, or at heavily loaded server, it is not possible to lose processing speed running cryptography algorithms in general software, which drops the efficiency of the overall system. On the other side, a low cost and small design can be used in smart card applications, allowing a wide range of equipment to operate securely.

2. Problem Definition

People are reluctant to admit it, but the world revolves around secrets. Without secrets, there would be no privacy—everybody's personal and business information would be open to public inspection. It would be impossible to safeguard a personal or business identity, keep a lid on future plans, conduct financial transactions, or even maintain a bank account. Especially now, during the e-commerce explosion, secure and reliable exchange systems are vital for the world’s economy.

This application should provide security at file content level. It uses Cryptosystems, highly technical systems that provide privacy through secret encoding, have been an important part of the electronic information world for many years.

3. Proposed System

- In this project we use cryptography techniques for encryption and decryption of message. While transmitting packet from source to destination we encrypt packet and transmit the packet at the receiving side decryption is done using a key that is only available to the user. When there is any data hacking at the middle of the network it is not possible to decrypt the packet.

- Rijndael Secure System is supposed to run with Java runtime environments version 1.4 and higher. This has the advantage of working under most operating systems in use today.

- Write the code to execute as Command mode and Graphical Interface

- Provide interactive interface through which user can interact with different types of File Size.

4. Implementation

This project provides the security by using Rijndael Algorithm. Rijndael Algorithm is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds. The key size, which can be 128, 192, or 256 bits, depends on the number of rounds.
The Rijndael algorithm has five functions to encrypt the data. To perform the decrypt, it uses another set of functions that executes the inverse operation. The analog functions of both operations are executed in inverse sequence. The functions used are, in execution order: Byte Sub, Shift Row, Mix Column and Add Key, in the decryption the order is Add Key, IMix Column, IShift Row and IByte Sub. Note that Add Round Key is its own inverse function.

In this project the Key Schedule uses 10 rounds. Each round executes the same functions in the same order. Only one round, the first one in decryption and the last one in encryption, does not execute the Mix Column.

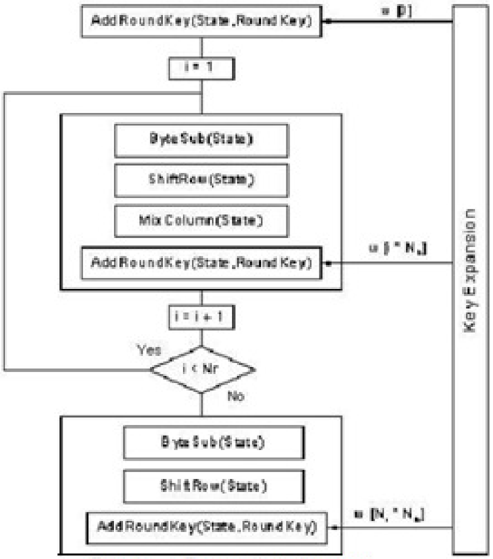


Figure : Encryption diagram.

To each round the key schedule generates a specific key, called Round Key. The fifth function, called Round Key Function, executes this operation. The round keys are generated based on the original key. Once this is done, the round keys of each round are the same until a new key are set. The round schedule works with xors, shifts and table look-ups. All keys are assumed as a sequence of one byte cells. The function executes a xor of some previous cells to determine the present cell.

Sub Byte

The first transformation, SubBytes, is used at the encryption site. To substitute a byte, we interpret the byte as two hexadecimal digits**.**

In the usual round operation, the first function in the encryption process is a table look up, also executed byte to byte. The function takes the data variable and assumes it as an address of a specific memory defined by the algorithm. The data stored in this address is taken as the new work variable (figure 4). The function is called Byte Sub.

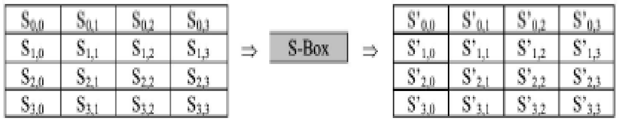
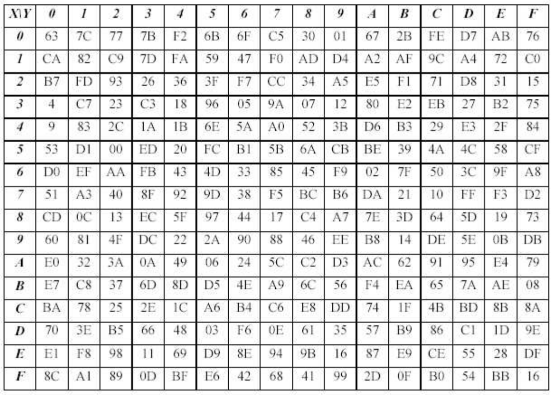

$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,0}$ $S_{1,1}$ $S_{1,2}$ $S_{1,3}$
$S_{2,0}$ $S_{2,1}$ $S_{2,2}$ $S_{2,3}$
$S_{3,0}$ $S_{3,1}$ $S_{3,2}$ $S_{3,3}$
⇒
S-Box
⇒
$S'_{0,0}$ $S'_{0,1}$ $S'_{0,2}$ $S'_{0,3}$
$S'_{1,0}$ $S'_{1,1}$ $S'_{1,2}$ $S'_{1,3}$
$S'_{2,0}$ $S'_{2,1}$ $S'_{2,2}$ $S'_{2,3}$
$S'_{3,0}$ $S'_{3,1}$ $S'_{3,2}$ $S'_{3,3}$

Figure : Byte Sub transformation.

In order to secure the cipher against the attacks related to this cryptanalysis, the non linearity of the S-Box should satisfy few properties. They are, the maximum input output correlation and the di_erence propagation probability should be minimum to the extent possible. The substitution component of Rijndael is having these properties at the maximum optimum level. Also the SubBytes component can be implemented as a table look-up operation to gain speed and to preclude timing and di_erential power attacks. Decryption can be achieved by reversing the steps of the algorithm with few exceptions and with inverted components.

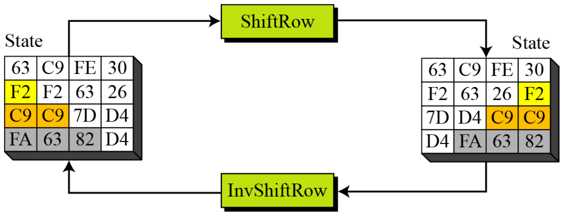


X\Y	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	4	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	9	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	16	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Figure S-Box Table

Shift Rows

This function is a simple permutation or it is sometimes called transposition. The main function of this layer is to spread the changes made at the Sub Bytes layer. The di_usion should be well optimized to give resistance against the linear and di_erential cryptanalysis. The Shift Rows layer operates on row level of the state. The permutation is achieved here by shifting the rows cyclically over di_erent o_sets. The four rows should have di_erent o_sets. And the shift o_sets for various block lengths. The choice of the shift o_sets matters in case of the algorithm's strength against di_erential and saturation attacks. So in Rijndael, the simplest and the strong options are been chosen for the o_sets. For the target block length of 128 bytes, the first row is not shifted, but the second by one left shift, third by two and fourth row by three left shifts.

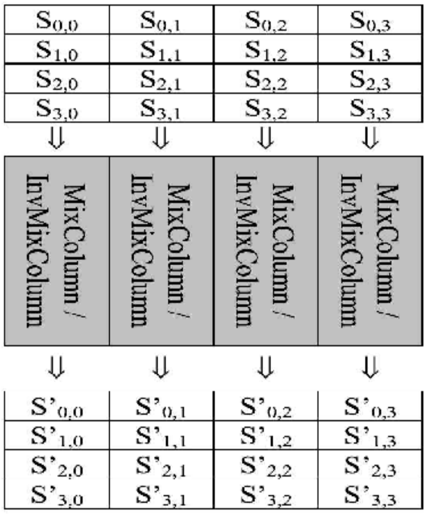


63	C9	FE	30
F2	F2	63	26
C9	C9	7D	D4
FA	63	82	D4

63	C9	FE	30
F2	63	26	F2
7D	D4	C9	C9
D4	FA	63	82

Mix Columns

This is an important component worth attention. The sub-bytes component, we discussed before, is actually changing the bytes and this local byte change is di_used (spread) further with the help of the mix-columns operation.



$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

$S'_{0,0}$	$S'_{0,1}$	$S'_{0,2}$	$S'_{0,3}$
$S'_{1,0}$	$S'_{1,1}$	$S'_{1,2}$	$S'_{1,3}$
$S'_{2,0}$	$S'_{2,1}$	$S'_{2,2}$	$S'_{2,3}$
$S'_{3,0}$	$S'_{3,1}$	$S'_{3,2}$	$S'_{3,3}$

Figure: Mix Column transformation

The last operation made in the Rijndael algorithm is a xor operation between the data and the key variable. In the first round, data and key are read as input. In the next rounds, the data is the work variable and the key is the round key of the specific round. This xor procedure is called Add Key and it operates over each byte.

Round Key

Add Round Key proceeds one column at a time. Add Round Key adds a round key word with each state column matrix; the operation in Add Round Key is matrix addition

Every round has a di_erent key, where each round key is derived from the cipher's original key. The key schedule algorithm has two Sub components. First one is the key expansion which is used to derive the expanded key from the cipher key. The expanded key is the concatenation of the individual round keys. The second subcomponent is the round key selection, where in simple to sophisticated ways to select the round keys can be accomplished. But Rijndael has the simple key selection procedure. Nonlinearity, di_usion and symmetry elimination are achieved to avoid some speci_c kind of attacks. Nonlinearity is achieved by using the same sub bytes component, and di_usion is used to e_ciently spread the cipher key di_erences into the expanded key and the symmetry elimination is achieved by using a di_erent constant in generating each round key.

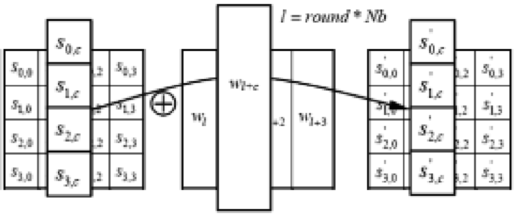


Figure: Add Round Key Transformation

Architecture

Besides Rijndael was designed to work with data blocks of 128, 192 and 256 bits using 128, 192 and 256 bits key, the AES was defined as three versions – AES128, AES168 and AES256 – corresponding to the usage of 128, 192 and 256 bit cipher keys. In this work, all the implementation was focused in the AES-128. Here we present the implementation of Rijndael in three ways: the first one just encrypt the data, the second one just decrypt and the third one does the both executions. This options can provide a choice to implement the Rijndael, as the area increases with the both devices together. If either decrypt or encrypt function are not needed, just one device could be implemented. Although, the use of the third implementation is better as it is easiest to operate. All versions use a very similar structure.

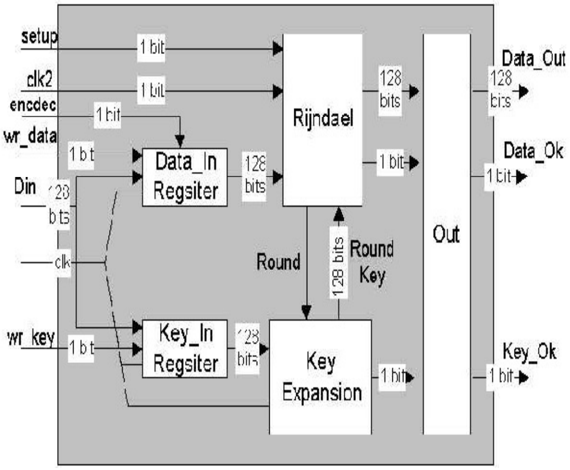


Figure: Encrypt and decrypt architecture.

To allow the free execution of the Rijndael algorithm, the structure was divided in some processes. The Rijndael process itself executes only the encrypt or decrypt algorithm, according to the case. The others processes only provide support to read and write bus operation and to round keys generation.

5. CONCLUSION

Historically, encryption's main use has been to protect files stored on a hard drive or floppy disk. Encryption ensures that if a protected computer or disk should fall into the wrong hands, the contained information will be indecipherable. This type of encryption is particularly useful to portable computer users, whose systems are particularly vulnerable to theft.

With most computers now connected to the Internet, there's a critical need to protect both e-mail and attached files. Encryption allows users to send and receive information over public networks without worrying about whether their communications will be intercepted and used by unauthorized parties.

Encryption technology has also allowed the creation of virtual private networks (VPNs). Unlike an office LAN, which uses dedicated wiring to connect workstations, a VPN relies on the Internet to link users located within an office or across continents. By locking out unauthorized parties, encryption technology allows a VPN to operate as securely as a LAN.

Encryption is also playing an increasingly important role in the emerging e-commerce economy. The technology is widely used to encrypt credit card information, bank account numbers, and other types of financial records so they can be sent safely and securely across the Internet. Encryption is also being used to protect much of the intellectual content that's marketed on the Web, such as music, videos, articles, and software, restricting its availability to paying customers.

REFERENCES

1. The Design of Rijndael: AES - The Advanced Encryption Standard (Information Security and Cryptography) – By Joan Daemen (Author), Vincent Rijmen (Author).
2. Elisabeth Oswald, Joan Daemen and Vincent Rijmen, AES - The State of the Art of Rijndael's Security, October 30, 2002
3. Joan Daemen, Vincent Rijmen, Answers to new observations on Rijndael, August 11, 2000 Daemen, J., Rijmen, V. (1999). The Rijndael Block Cipher. Document Version 2.
4. DAEMEN, Joan e RIJMEN, Pawel. The block cipher Rijndael. Rijndael official homepage, available at http://www.esat.kuleuven.ac.be/~rijmen/rijndael/
5. Gladman, B. (2002). A Specification for the AES Algorithm. Berkeley.
6. Lin, T. F., Huang, C. T., and Wu, C. W. (2002). A High-Throughput Low-Cost AES Cipher Chip. Taiwan: Laboratory for Reliable Computing Department of Electrical Engineering National Tsing Hua University Hsinchu.
7. McLoone, M. and McCanny, J.V. (2001). Rinjdael FPGA Implementation Utilizing Look-Up Tables. IEEE 0-7803-7145-3/01.
8. Satoh, A. and Morioka, S.(2003). Unified Hardware Architecture for 128-Bit Block Ciphers AES and Camellia. Berlin
9. Heidelberg: Springer-Verlag. Sanchez, C., Avila, K. and Reillo, S. (2001). The Rijndael Block Chiper.
10. Altera, “High Speed Rijndael Encryption/Decryption Processors.”, Hammercores whitepaper v. 1.0.

S.Bhuvaneswari [1] Radha Madhavi Vutukuri[2]

[1]Reader & Head, Department of Computer Science, Pondicherry University, Karaikal Campus, India

[2]Department of Computer Science, Pondicherry University, Karaikal Campus, India