Advanced-Data Security using multiple Cryptographic Algorithms over multiple layers and secured Key sharing

Project ID:21J606593

Review - II

Group Members
RA1711003030606 Tanmoy Sen Gupta
RA1711003030593 Ritik Srivastava

Supervised By:
Dr. K. Marimuthu
Professor

Department of Computer Science & Engineering
Faculty of Engineering & Technology
SRM Institute of Science & Technology

April 7, 2021

Table of Contents I

1. Abstract
2. Objective
3. Literature Survey
4. Design of Proposed System
5. Algorithms Used
6. Partial Implementation
7. Expected Outcome
8. References

Abstract

Security of Data is one of the most important aspects of one's digital presence. With the advent of new and sophisticated technologies, existing data security systems are becoming less efficient in protecting data. Advanced data security protocols used in various applications like WhatsApp and Telegram are becoming inadequate to protect the privacy of an individual. With advancements in Hardware, the time required to break a cryptographic system is becoming even lesser than earlier.

Thus the requirement of a multi-layered encryption protocol that utilizes multiple advanced cryptographic algorithms implemented in a cascading manner is ever more necessary. In this project, data will be encrypted using multiple algorithms like AES, RSA, and Blowfish.The keys that will be used for encryption will also be secured similarly. Multi-layered encryption protocols are not widely used in current systems, hence this project will address the problems faced by standalone systems used for data security.

Objective

- The main objective of this project is to build a Hybrid Crypto-system that secures data on multiple layers and also ensures security of keys.
- The Hybrid crypto-system also securely stores the keys so that they don't lead to any vulnerabilities.
- To create a crypto-system that provides excellent security without compromising on performance and speed.
- To overcome the performance-security tradeoffs of cryptographic algorithms when used separately.

Literature Survey

TITLE AND AUTHOR	SUMMARY
Application of AES and RSA Hybrid Algorithm in Email - *Ye Liu, Wei Gong, Wenqing Fan - ICIS, 2018* [6]	Combining asymmetric encryption with symmetric encryption algorithms makes the system significantly secured and faster. The experimental system also shows that Hybrid Crypto-systems are a great alternative to traditional crypto-systems that rely on higher keys sizes and rounds.
Performance Comparison Between AES256-Blowfish and Blowfish-AES256 Combinations - *Muhammad Abdul Muin, Muhammad Abdul Muin, Arief Setyanto, Sudarmawan, Kartika Imam Santoso - ICITACEE, 2018* [7]	Result shows that a composite cryptosystem on AES256-Blowfish required longer decryption time compared to the composite cryptosystem in reverse order (Blowfish-AES256) therefore, is considered the most secure algorithm compare to Blowfish-AES256, Blowfish or AES256.
Enhancement the Security of Cloud Computing using Hybrid Cryptography Algorithms - *Ali Abdulridha Taha, Dr. Diaa Salama AbdElminaam, Prof. Khalid M Hosny - IJACT, 2017* [10]	The proposed system demonstrates that the use of hybrid algorithms increases the level of encryption of encrypted mobile data and also reduces the time required for encryption and decryption.
An Efficient Algorithm for Confidentiality, Integrity and Authentication Using Hybrid Cryptography and Steganography - *Chitra Biswas, Udayan Das Gupta, Md. Mokammel Haque - ICECCE, 2019* [2]	The resistivity of the proposed system, consisting of AES-RSA Data and Key security and LSB Steganography for storing encrypted key, against attack has been ensured. Thus this algorithm provides confidentiality, integrity and authentication together.
A Research Paper on New Hybrid Cryptography Algorithm - *Prof. Swapnil Chaudhari, Mangesh Pahade, Sahil Bhat, Chetan Jadhav, Tejaswini Sawant - IJRDT, 2019*[3]	The paper studies the implementation of a hybrid Crypto-system of symmetric encryption and asymmetric algorithms. It explores the flows in both standalone systems and uses a hybrid approach to enhance security and address the drawbacks of the standalone systems.

Literature Survey

TITLE AND AUTHOR	SUMMARY
Performance evaluation of Hybrid Cryptography Algorithm for Secure Sharing of Text and images - *Pooja Patil, Dr. Rajesh Bansode - IJRET, 2020* [8]	A combination of AES-ECC and SHA-256 is implemented and targeted towards securing medical sector data. It proves efficient in securing text and image based data.
Secure File Storage using Hybrid Cryptography - *S.Gokulraj , P.Ananthi , R.Baby , E.Janani - SSRN, 2021* [9]	A robust and highly secure cryptosystem is implemented using a combination of AES, DES, RC2 to secure data and LSB Steganography is used to ensure Key Security.
Efficient Hybrid Cryptography Algorithm - *Mayes M. Hoobi - Journal of Southwest Jiaotong University, 2020* [4]	Hybrid crypto-system with a combination of DES and ECC, based on test results, proved to increase complexity of block cipher, in addition to increasing the search space of DES Key.
Design And Implementation Of A Hybrid Cryptography Textual System - *Dr. Mahmood Zaki Abdullah, Zinah Jamal Khaleefah - Transactions on Image Processing (Journal), 2018* [1]	The experimental results for the proposed algorithm based on Lorenzo Chaotic and image steganography show that it is secure because it has a large key space, high sensitivity to plain text and secret key.
Novel Hybrid Cryptography for Confidentiality, Integrity, Authentication - *Avinash Jain, V. Kapoor - IJCA, 2017* [5]	A combination of AES and RSA cryptosystem proves to provide high data security and administer key distribution providing secure transmission of data and key

Design of Proposed System

Data Encryption

Data is encrypted using 3 layered hybrid crypto-system consisting of a combination of Blowfish, RSA and AES algorithms. The Plaintext is encrypted using Blowfish using a secret key, then the ciphertext from Blowfish is encrypted by RSA Algorithm using the RSA Public Key. The encrypted ciphertext from RSA is then finally encrypted using the AES Algorithm. The encrypted ciphertext from the AES Algorithm is the final required ciphertext.

Design of Proposed System

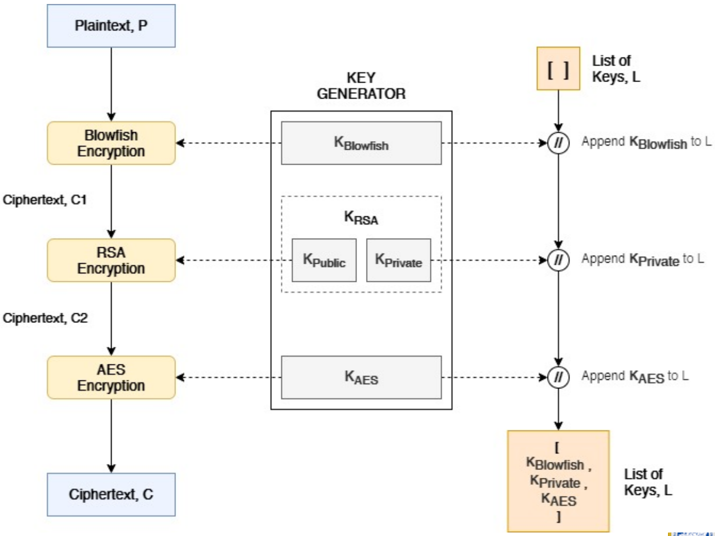


Figure: Data Encryption using Cascading Cryptographic Systems

Design of Proposed System

Secure Key storage using Keys encryption and LSB Steganography

Security of the Keys used is ensured by Encrypting the keys and embedding them in an image. The Keys used are appended to a list of keys after each usage. Then the list is encrypted using AES Algorithm, using the hash of the password set by the user as the key.

The encrypted keys are embedded in an image using LSB Steganography.

Design of Proposed System

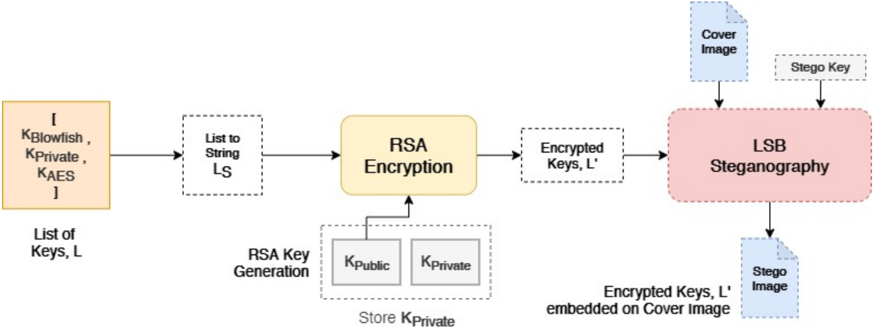


Figure: Secured Key Storage

Algorithms Used

Blowfish Algorithm

Blowfish is a Symmetric Block Cipher developed by B. Schneier in 1993. It has complicated key schedules and key-dependant s-boxes. It has a block size of 64 bits and a variable key length in the range of 32 to 448 bits. It being a feistel cipher has 16 rounds with each round having 4 steps. At each step the left part of the block is XORed with its corresponding Round Key and fed into a round function F, the output of which is XORed with the right half of the original block and then swapped.

Algorithms Used

RSA Algorithm

Rivest–Shamir–Adleman (RSA) Algorithm is an asymmetric cryptographic algorithm that uses a Public Key, available to everyone on network, to encrypt and a Private Key, available to only the Sender and Receiver, for decryption. The keys are large prime numbers of lengths 1024 / 2048 / 3072 / 4096 Bits.

Using RSA, encryption of plaintext, M is done using the Public Key, e as:

$$Ciphertext, C = M^{\mathrm{e}} \; modn$$

The Ciphertext, C is decrypted using the Private Key, d as:

$$Plaintext, M = C^{\mathrm{d}} \; mod \; n$$

Algorithms Used

AES Algorithm

Rijndael is a Block Cipher developed by Belgian cryptographers, Vincent Rijmen and Joan Daemen that has been established as the Advanced Encryption Standard by the U.S. National Institute of Standards and Technology (NIST) in 2001. The AES has key lengths of 128, 192, 256 bits and 10, 12, 14 number of rounds respectively.

The 128 Bit key is expanded using AES Key Schedule into a number of subkeys depending on the number of rounds. At the beginning, an Initial Round Key is XORed to the input block. Then, for the first N-1 rounds, where N is the number of rounds, Multiple Round Functions are applied on the block.

Algorithms Used

LSB Steganography

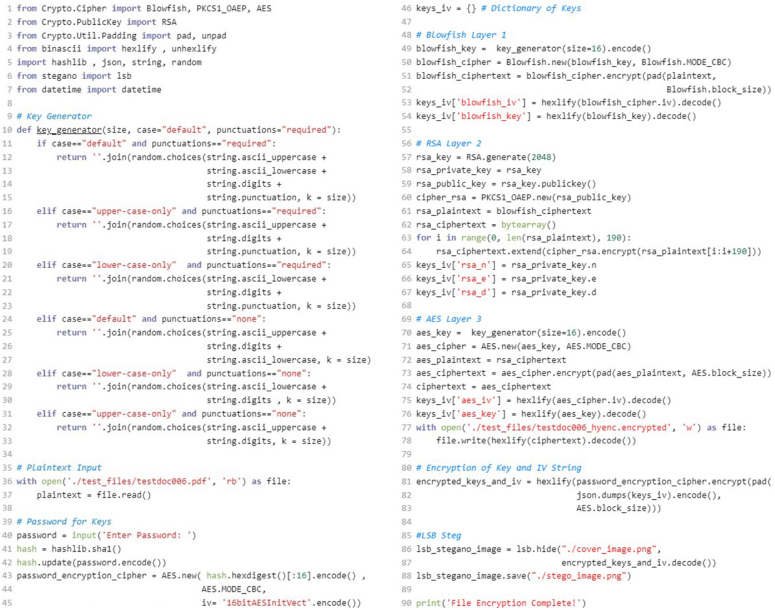
Least Significant Bit Steganography or LSB Steganography is the method of hiding secret data inside any form of digital media, here, Image.

Images are made up of pixels which usually refer to the color of that particular pixel. In a grayscale image, these pixel values range from 0-255, 0 being black and 255 being white. In LSB Image Steganography, changing the last bit value of a pixel, won't have much of a visible change in the color.

A cover image is used to embed the data in. The otput of the process is

called a Stego Image.

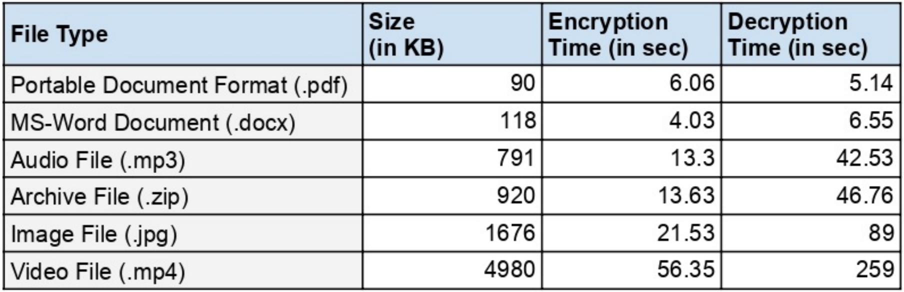
Partial Implementation using Python



```python
from Crypto.Cipher import Blowfish, PKCS1_OAEP, AES
from Crypto.PublicKey import RSA
from Crypto.Util.Padding import pad, unpad
from binascii import hexlify , unhexlify
import hashlib , json, string, random
from stegano import lsb
from datetime import datetime

# Key Generator
def key_generator(size, case="default", punctuations="required"):
    if case=="default" and punctuations=="required":
        return ''.join(random.choices(string.ascii_uppercase +
                                      string.ascii_lowercase +
                                      string.digits +
                                      string.punctuation, k = size))
    elif case=="upper-case-only" and punctuations=="required":
        return ''.join(random.choices(string.ascii_uppercase +
                                      string.digits +
                                      string.punctuation, k = size))
    elif case=="lower-case-only"  and punctuations=="required":
        return ''.join(random.choices(string.ascii_lowercase +
                                      string.digits +
                                      string.punctuation, k = size))
    elif case=="default" and punctuations=="none":
        return ''.join(random.choices(string.ascii_uppercase +
                                      string.digits +
                                      string.ascii_lowercase, k = size)
    elif case=="lower-case-only"  and punctuations=="none":
        return ''.join(random.choices(string.ascii_lowercase +
                                      string.digits , k = size))
    elif case=="upper-case-only" and punctuations=="none":
        return ''.join(random.choices(string.ascii_uppercase +
                                      string.digits, k = size))

# Plaintext Input
with open('./test_files/testdoc006.pdf', 'rb') as file:
    plaintext = file.read()

# Password for Keys
password = input('Enter Password: ')
hash = hashlib.sha1()
hash.update(password.encode())
password_encryption_cipher = AES.new( hash.hexdigest()[:16].encode() ,
                                     AES.MODE_CBC,
                                     iv= '16bitAESInitVect'.encode())
keys_iv = {} # Dictionary of Keys

# Blowfish Layer 1
blowfish_key =  key_generator(size=16).encode()
blowfish_cipher = Blowfish.new(blowfish_key, Blowfish.MODE_CBC)
blowfish_ciphertext = blowfish_cipher.encrypt(pad(plaintext,
                                                  Blowfish.block_size))
keys_iv['blowfish_iv'] = hexlify(blowfish_cipher.iv).decode()
keys_iv['blowfish_key'] = hexlify(blowfish_key).decode()

# RSA Layer 2
rsa_key = RSA.generate(2048)
rsa_private_key = rsa_key
rsa_public_key = rsa_key.publickey()
cipher_rsa = PKCS1_OAEP.new(rsa_public_key)
rsa_plaintext = blowfish_ciphertext
rsa_ciphertext = bytearray()
for i in range(0, len(rsa_plaintext), 190):
    rsa_ciphertext.extend(cipher_rsa.encrypt(rsa_plaintext[i:i+190]))
keys_iv['rsa_n'] = rsa_private_key.n
keys_iv['rsa_e'] = rsa_private_key.e
keys_iv['rsa_d'] = rsa_private_key.d

# AES Layer 3
aes_key =  key_generator(size=16).encode()
aes_cipher = AES.new(aes_key, AES.MODE_CBC)
aes_plaintext = rsa_ciphertext
aes_ciphertext = aes_cipher.encrypt(pad(aes_plaintext, AES.block_size))
ciphertext = aes_ciphertext
keys_iv['aes_iv'] = hexlify(aes_cipher.iv).decode()
keys_iv['aes_key'] = hexlify(aes_key).decode()
with open('./test_files/testdoc006_hyenc.encrypted', 'w') as file:
    file.write(hexlify(ciphertext).decode())

# Encryption of Key and IV String
encrypted_keys_and_iv = hexlify(password_encryption_cipher.encrypt(pad(
                                json.dumps(keys_iv).encode(),
                                AES.block_size)))

#LSB Steg
lsb_stegano_image = lsb.hide("./cover_image.png",
                             encrypted_keys_and_iv.decode())
lsb_stegano_image.save("./stego_image.png")

print('File Encryption Complete!')
```

Expected Outcomes

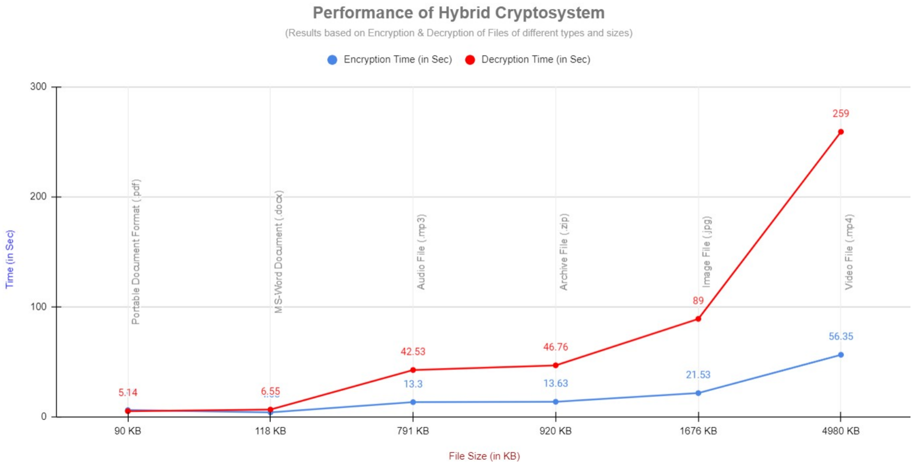
The Hybrid Crypto-system implemented in Python was tested against different file types of varying sizes and 'srm2017' as the password. The following results were obtained.



File Type	Size (in KB)	Encryption Time (in sec)	Decryption Time (in sec)
Portable Document Format (.pdf)	90	6.06	5.14
MS-Word Document (.docx)	118	4.03	6.55
Audio File (.mp3)	791	13.3	42.53
Archive File (.zip)	920	13.63	46.76
Image File (.jpg)	1676	21.53	89
Video File (.mp4)	4980	56.35	259

The tests were conducted on a Windows 10 based laptop with Intel i3 processor and 4 GB of RAM.

Expected Outcomes



References I

M. Z. Abdullah and Z. J. Khaleefah.
Design and implement of a hybrid cryptography textual system.
In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6, 2017.

C. Biswas, U. D. Gupta, and M. M. Haque.
An efficient algorithm for confidentiality, integrity and authentication using hybrid cryptography and steganography.
In *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pages 1–5, 2019.

S. B. C. J. T. S. Chaudhari Swapnil, Mangesh Pahade.
A research paper on new hybrid cryptography algorithm.
05 2018.

References II

M. M. Hoobi.
Efficient Hybrid Cryptography Algorithm.
Journal of Southwest Jiaotong University, 2020.

A. Jain and V. Kapoor.
Novel hybrid cryptography for confidentiality, integrity, authentication.
International Journal of Computer Applications, 171:35–40, 08 2017.

Y. Liu, W. Gong, and W. Fan.
Application of aes and rsa hybrid algorithm in e-mail.
In *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, pages 701–703, 2018.

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

References III

M. A. Muin, M. A. Muin, A. Setyanto, Sudarmawan, and K. I. Santoso.
Performance comparison between aes256-blowfish and blowfish-aes256 combinations.
In *2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pages 137–141, 2018.

D. R. B. Pooja Patil.
Performance Evaluation of Hybrid Cryptography Algorithm for Secure Sharing of Text and Images.
International Research Journal of Engineering and Technology (IRJET), 07(09), 2020.

References IV

- R. B. E. J. S. Gokulraj, P. Ananthi.
  Secure File Storage Using Hybrid Cryptography.
  SSRN, 2021.

- A. Taha, D.-D. Salama, S. Abdelminaam, M. Khalid, and K. Hosny.
  Enhancement the security of cloud computing using hybrid cryptography algorithms.
  International Journal of Advancements in Computing Technology, 9, 12 2017.