A Hybrid of RSA Cryptography with Huffman Compression Algorithms as a Data Security Measure for Internet Users

[1]Oladeji, Florence. A, [2]Ajetunmobi, Rukayat A. [3]Ajayi Olasupo O., [4]Adeniyi Adebayo and [5]Olatunji Michael.

[1] Department of Computer and Information Sciences, Covenant University, Ota
[2,3,4, 5] Department of Computer Science, University of Lagos, Akoka, Lagos, Nigeria
[1]foladeji@cu.edu. ng ,[2]rajetunmobi@unilag.edu.ng, [3]olaajayi@unilag.edu.ng [4]bayuyus4real@yahoo.com and [5]researchmichael01@gmail.com

Abstract

Security of transmitted data calls for serious concern especially in this information-centric era where data resources of businesses and organization are held on remote servers. Too much dependence on the network to protect data had exposed the data to danger of cybercrimes and middlemen attacks. It is then expedient that the stakeholders concerned with the data are involved in the security process that is, users are involved in placing some security measures on their data before giving it to the cloud or remote storages offering data storage as a service. Data encryption and compression algorithms play dependable roles on security of transmitted data (Forouzan and Fegan, 2006; Comer, 2009). Encryption disguises the content of a document such that only the sender and the receiver can get it back. There are many data encryption algorithms ranging from data encryption standard (DES), advanced encryption standard (AES) to Rivest-Shamir-Adleman (RSA) algorithms (Comer, 2009). Data compression or encoding on its own reduces the size of the data before it is transmitted. Data compression has the advantages of reducing network traffic by reducing the size of data to be transmitted. Data encoding algorithms also exist with differences in the technique and type of data to be coded. Among them are Huffman, Lempel-Ziv and arithmetic encoding mechanisms to mention a few. The two steps are not compelled to be adopted on all data or across networks. Firewall sometimes prompts the alert that this data is not encrypted. This work suggests a hybrid of encryption and data compression algorithms as a combined effort from the user level. The showcased algorithms are the use of RSA cryptosystem and Huffman encoding scheme. Data meant for transmission is first encrypted by the human (source of the data) and then subjected to encoding before submission. The implementation was carried out using Java language and sample GUI-based screenshots were documented.

Keywords: Cybercrimes, Cryptography, Huffman encoding, RSA cryptosystem, object-oriented programming

1.0 Introduction

Issue on security has cut across different computing paradigms. Early systems witnessed theft of computing systems and data corruptions due to viruses and power outages. The world today is full of indoor and outdoor data hijackers who move with technology to perpetrate evils leading many businesses running at loss. This makes Internet users more concerned on confidentiality and integrity of transmitted data. For instance, despite the tremendous business and technical advantages of the cloud, the security and privacy concerns have been one of the major hurdles preventing its widespread adoption (Shahzad, 2014). This might be justifiable as recent reports show that between January, 2008 and February, 2012, security issues accounted for a combined 54% of all cloud computing related threats. (Ryan, et al., 2013). IDC reports between 2008 and 2010 also showed that of all the challenges facing cloud computing, security ranked as the highest. (Gens, 2009).

Data storage is one of the major services provided in the cloud. Common examples include Amazon Simple Storage Service (Amazon S3) (Amazon Web Services, 2015), Google Drive (Google, 2012), Apple iCloud (Apple, 2011), Microsoft OneDrive (Microsoft, 2014) where storage facilities are opened to clients to keep their data as captured under the Infrastructure as a Service model of Cloud Computing. The security of data stored on these third party storage platforms is a major source of concerns for most cloud customers. Numerous security bridges have been recorded of recent, among which the Sony PlayStation data breach (Baker and Finkle, 2011), where Sony’s PlayStation Network was hacked and personal data of about 77 million users were stolen and Dropbox privacy leakage (Yin, 2011), where 25 million user accounts were left vulnerable to attack for close to four hours were one of the most major incidents.

Shahzad, (2014) reported some steps being taken to curb this menace. For example Amazon with its Amazon Web Services (Amazon Web Services, 2015) implements multiple degree of security measures to safe guard not just user data but also physical access to its data centres. Some of these measures include: restricted access to information about data centres (even their exact locations are kept confidential), encryption of data, implementing security best practices and various other forms of security to protect both applications and virtual machines. It should be noted that all these measures are provided by the service providers and not by the users who expose their data to third party hosts.

In this vein, it is also imperative that the sender, the receiver and the sending system with the service provider be involved in asserting security mechanism to hide their data from curious hackers In fact, local data contents encrypted by the owner can also serve as a backup to the simple password authentication system inherent in most operating systems This report presents a portable security conscious platform for individual users on their confidential files. Techniques in data security are presented and a hybrid of these concepts is proposed for network users, which consists of the popular RSA encryption algorithm and an encoding mechanism based on

Huffman as a measure to protect text data. This hybridised concept serves two purposes in one - protection of data and reduction of data size on disk or on transit.

The organization of the paper is as follows: Section one introduces the efficacies of having a handy portable and user friendly crypto-compress systems for personal use while section two discusses related algorithms for cryptography and data compression. In section three, the RSA and Huffman algorithms were presented respectively. The object-oriented implementations were described in section four while modality for usage and conclusions were reported in sections five and six respectively.

2.0 Related Works

The rapid growth in Information-centric networks emanating from the ability to keep confidential information in remote places rather than on local directory has unfortunately contributed to loss of confidentiality in data transmission to hackers. This is because as new technology emerges, network looting hackers are also advancing with the aim to intrude into the information being transmitted using the technology. Their perpetration ranges from wiretapping to phishing, spoofing to scams all eventually resulting in loss of valuable information or controls over the data. An objection to extensive use of the cloud by many organizations today is the fear on data confidentiality. Among the techniques that exist in literature to protect transmitted data are the use of data encryption (Diffie and Hellman, 1976; Vijay K and Sharma, 2012), digital certificate (Rivest et al, 1977; Stallings, 2007), digital signature (Stallings, 2007), two-factor tokenization, firewall and intrusion detection systems. Among these measures, data encryption dominates in handling bulky data such as text and video (Forouzan and Fegan, 2006).

2.1 Data Encryption

According to (Stallings, 2007; Stallings 2011; Kurose and Ross, 2010), data encryption, simply called cryptography is one of the fundamental security techniques that can guarantee data confidentiality or privacy on the Internet. It is a process of enscrambling of data such that it becomes meaningless to intercepting intruders called the middlemen during transit. Simply, an encryption mechanism accepts a message *M*, enscrambles it using a specified key to obtain a new message *C* called ciphertext. The cipher text is transmitted as packets to the specified destination. Upon reception, the destination system uses a specified key (which may be the same as the one at the source system or different) to decrypt the cipher text *C* to obtain the message *M* which is the original message. Since, the middleman might not know the key, he cannot decrypt it. The strength of these algorithms is the length of the key and the secrecy of the key. An algorithm that can stand the test of the time must use long keys at multiple runs, have strong mathematical backing and tested by expert (cryptanalyst) to prove its ruggedness. For instance, it took about twenty years for the DES to be hacked by cryptanalyst while AES is yet to be hacked (Kaliski, 1989; Daemen and Vincent , 2002)

In (Kurose and Ross, 2010; Forouzan and Fegan, 2006), two types of keys are common with the encryption algorithms, the private and public keys systems. In the Private Key system, a single secret key known only to the communicating system is used to for both encryption and decryption. The Public Key encryption system on the other hand uses two keys. The first is a private key and the second a public key. The Public Key is made known but the Private Key is kept secret. Thus, the original message is encrypted with a public key but decrypted with the private key. Exposing the public key does not guarantee an intruder to penetrate because he requires the receiver private key to decrypt. This can be modelled as, M= decrypt(public key, encrypt(private key, M), where M is the message being transmitted.

In literature, a number of public key systems exist among which include the use of RSA (Rivest *et al*, 1977; Evgeny, 2009; Kurose and Ross, 2010), Data Encryption Standard, DES (Biham, 1990), Advanced Encryption Standard, AES (Daemen and Vincent, 2002). It amazes a network user at times when a prompt of insecurity message comes from a firewall such as "your data is not encrypted or "your network is not secured", do you want to allow this program?" As at now, it is the network protocol and/or network provider that play the role of data encryption or compression. Even though we have operating system facility like winrar, password for reading or modify in operating systems, most users are negligence to their use, instead they enforce physical restriction to offices hosting valuable data.

2.2 Data Compression

Data compression is another network approach to curtail full text exposure as it seeks to reduce the size of the conveying data and reduce network loads (Khalid, 2006, Blelloch, 2013). A compression algorithm accepts as input the message content *M* of size *S*, applies a scheme to produce another compressed message *C* of size *L*, $L \leq S$. The new message **C** is transmitted through transmission channel to a designated destination with decompressing information as codebook. At the destination, another algorithm called decompressing algorithm, takes the received message *C* with the codebook attached and produces a new message *D*, the decompressed message. The compression algorithm is classified as lossless if the original message *M* is exactly the same as message *D* at the destination, otherwise it is called lossful (Khalid, 2006, Blelloch, 2013).

An added advantage of data compression to data disguise in (Forouzan and Fegan, 2006) is that, it reduces the size of the disguised data before transmission, hence reduction in traffic and subsequently improve network performance. A number of algorithms also exist in literature for data compression. Some are the Huffman encoding (Khalid,2006), Lempel-Ziv encoding (Arthur et al, 2003) and arithmetic encoding.

3.0 Crypto-Compress Hybrid Model

The new model this paper is proposing at the user level in a network is the combination of encryption and compression algorithms to aid effective security on confidential data. As introduced, a security conscious network user implement a first level security measure on data before handing over to the network or to the cloud system. The objective therefore is to ensure an end-to-end data security in a bid to mitigate the effect of middleman attackers as users shift to the cloud

In order to achieve this, RSA encryption algorithm in (Rivest et al, 1977; Kurose and Ross, 2010; Evgeny, 2009) is used to showcase the proposal while Huffman encoding (Robert,2003) is then applied to the encrypted data before storage or communication.

3.1 RSA Encryption Algorithm

This is the first and most common public key encryption implementation; named after 3 MIT mathematicians who developed it in 1977, namely: Ronald Rivest, Adi Shamir, and Leonard Adleman (Rivest et al, 1977, Evgeny, 2009). RSA today is used in hundreds of software products and can be used for key exchange, digital signatures, or encryption of small blocks of data. RSA uses a variable size encryption block and a variable size key. The key-pair is derived from a very large number, say ***n*** which is the product of two prime numbers chosen according to special rules; these primes may be 100 or more digits in length each, yielding an ***n*** with roughly twice as many digits as the prime factors. The public key information includes ***n*** and a derivative of one of the factors of ***n***; an attacker cannot determine the prime factors of n (and, therefore, the private key) from this information alone and that is what makes the RSA algorithm so secure.

The RSA cryptosystem contains the following algorithms (Kurose and Ross, 2010):

1. Key Generation Algorithm

The algorithm in (Kurose and Ross, 2010) follows these procedures:

1. Generate two large random primes, ***p*** and ***q,*** of approximately equal size such that their product **n = pq** is of the required bit length, e.g. 1024 bits.
2. Compute **n = pq** and **φ = (p-1)(q-1)**.
3. Choose an integer ***e,*** **1 < e < φ,** such that **gcd(e, φ) = 1.**
4. Compute the secret exponent ***d,*** **1 < d < phi**, such that **ed ≡ 1 (mod phi).**
5. The public key is **(n, e)** and the private key **(d, p, q).** Keep all the values **d, p**, **q** and φ secret. It is preferred sometimes to write the private key as **(n, d)** because you need the value of **n** when using **d**.

where:

n is the *modulus*.
e is the *public exponent* or *encryption exponent* .

d is the *secret exponent* or *decryption exponent*.

2. Encryption

A sender A on a network does the following:

1. Obtains the recipient B's public key **(n, e).**
2. Represents the plaintext message as a positive integer ***m*, 1 < m < n**
3. Computes the cipher text

$\mathbf{c = m^e \bmod n}$ Sends the cipher text ***c*** to B

3. Decryption

Recipient B does the following:-

1. Uses his private key **(n, d)** to compute

$m = c^d \bmod n$.

2. Extracts the plaintext from the message representative **m**.

3.2. Huffman Encoding

According to (Robert, 2003, Khalid, 2006) , this encoding scheme accepts a message, computes a frequency count of available letters in the document, sort them into a priority queue list. It then draws a binary tree structure of the priority list having all the letters at the leaves of the tree. The trees has both the left and right subtrees. Each edge in any of these subtrees that falls to the left side of a root node is assigned a binary value 0 and each edge at the right of any root node is assigned a 1. Thus, codewords are generated for each letter by traversing from the root to the particular letter writing out the binary values of the edges to the letter starting from the main root of the binary tree.

Letters that occur most often has a lesser code word than those that less frequent. A new message called codewords is then produced and send to the destination with a codebook containing the letter and its code word. An agreed codeword demarcator is used to separate the codewords of the message characters.

At the receiving end, the decompression algorithm receives the coded message and the codebook and follows an agreed standard character code (ASCII, UNICODE, EBCDIC) to obtain the original message back.

4.0 Object Oriented Implementation of RSA

The architecture of the implemented RSA encryption algorithm shown in Figure 1 with each stage involved as a high level view. The architecture shows the process and the sub-processes involved from the launch of the application to the termination of the application process.

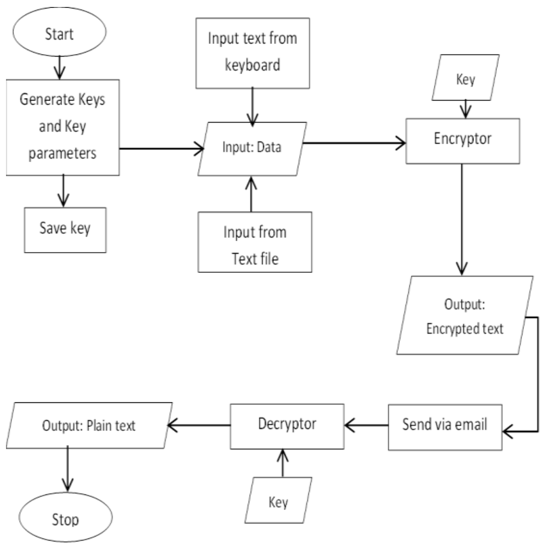


Figure 1: Architecture of the RSA-Huffman implementation

For Application development of the RSA using Java programming language (Deitel and Deitel, 2012), the following classes were built:

The Mainframe class: This class represents the main entry into the system; it is a frame which is displayed first when the application is started. The class contains the main controls found in the system; these controls are in form of menu items, buttons and a text area; the class also contains the various methods which handle user's actions through the controls (Encrpt, Decrypt, Compress and decompress buttons).

The keygenerationform class: This class is also a Jframe found in the application; it is used during the key generation process. The class contains various controls and methods for responding to the various actions carried out by a user. This class is responsible for the representation of the generation of the needed keys and key parameters in visual form.

The output class: This is yet another Jframe that can be found in the implementation of this RSA cryptosystem, it contains buttons and a text area. In the class, there are methods which handle the action of the user during decryption and mail sending.

RSAkeygeneration class: This class contains the methods for the key parameter generation and the saving of keys to file. It contains the following methods;

- **RSAkeygenerator:** This method is used to initialize the private key size class member and construct an object of the class.
- **Setkeyfile:** This is used to set the string name of the keyfile class member.
- **Getkeyfile:** This is used to return the name of the file. i.e. the key file class member
- **Getparam:** This method is used to calculate private and public key, once the key size is set during construction of the class object. The method performs the major arithmetic involved in key generation and stores the values for each parameter in a class variable.
- **Getkeysize:** This method is used to return the value of the key size class member to a method or caller.
- **PrintTofile:** This method is used to save the value generated during a call to the Getparam method. It contains a print writer; which is used to print the values of private and public key plus the other parameters that might be needed to file.
- **Setkeysize:** This method is used to set the value of the keysize class member to a specified value.

1. **RsakeyEncryption Class**: This class performs a major function in the implementation. It contains private members both of the BigInteger type. And the methods:

- **RsakeyEncryption:** This is used to create an instance of the class and at the same time; pass the value of the file name of the file containing the encryption key.
- **encrypt:** This method performs the function of encryption; it takes two parameter; the first, a name of the file where the plaintext can be found and the second a name of the file where the encrypted text is to be stored.
- **PutBtesBlock:** This method is used by the encrypt method to put the array of Bytes of the cipher text generated into blocks and remove the extra byte added during conversion from Big integer to Byte (Array of Bytes). The method takes two parameter;
- **PadByteBlock**: This method is used to ensure the number Bytes in the clear text are large enough so that when broken into parts, it gives an equal division.

6. **RsaKeyDecryption Class:** This class represents another major part of the implementation; it contains the following class members.

- **d**: This is the private decryption key: it is of the type Big integer
- **n**: The key modulus; is also of the Big integer type
- **RsakeyDecryption:** This is used to create an instance of this class and at the same time pass the name of the file containing the private key and the key modulus.

- **decrypt**: This method is used to perform the decryption function, it takes two parameters; the first is the name of the encrypted file while the second is the name of a location where the decrypted file will be saved.
- **GetDataSize:** This method is used to get the data size from a padded block; it takes an array of byte as its only parameter and returns an integer value which represent the value of the data size.
- **PutBytesBlock**: This method is used by the decrypt method to put the array of bytes of the clear text generated or recovered into blocks and at the same time remove the extra-byte added during conversion from big integer into an array of Bytes. The method takes as input two parameter of the types array of Bytes.

1. **Domail Class:** This class contains the method that performs the sending of mails, the mail will contains an attached file which will represent the encrypted data that a sender desires to send to a receiver.
2. **MailForm.java class:** This class provides the user interface where a user can perform the mailing operation; it contain methods that checks if the user has supplied all the required fields with the needed parameters; it also contain methods that respond to the actions performed by users through the clicking of a button.

7. CompressFile class: this invokes the compression methods

- **GetFile** option which prompts for word to compress or data file to compress (browse option)
- **SetFrequencyFile**: a method that computes the frequency counts of the letters in the document
- **ProduceTree**: A method that take letter frequency file and produces a binary tree
- **GenerateCodeWord**: A method that generate the codewords and the codebook
- **DecompressCodeWord**: Accepts the codebook and code words and use ASCII character representation to obtain back the message.

5.0 Results and Usability of Simulated Crypto-compress Model

The intended aim of this paper is to come up with a portable user friendly crypto-compress application that can be used indoor, outdoor, online or offline to scramble and reduce confidential data kept on hard disk, databases, phones, cloud and even data that are displayed on any visual display unit. For analysis purpose, we intend to benchmark the algorithm with the original RSA algorithm to estimate the overhead associated with attaching a compression scheme to it. Since, we cannot obtain effect of encryption from a network provider for now, the algorithm complexities are considered in time and space required for the benchmark and the hybridized version on a single computer system. It is expected to be distributed for usage and the outcome influence gathered for analysis.

The Interface designs are shown in Figure 2 Figure 3, Figure 4 and Figure 5. Figure 2 represents the main entry into the system; it shows the various operations performed by the system in form of buttons and menu items. A user can make selection from the available options based on the intended operation he/she wants to perform. A user of the system begins by generating a key and saving it to file as shown in Figure 3.

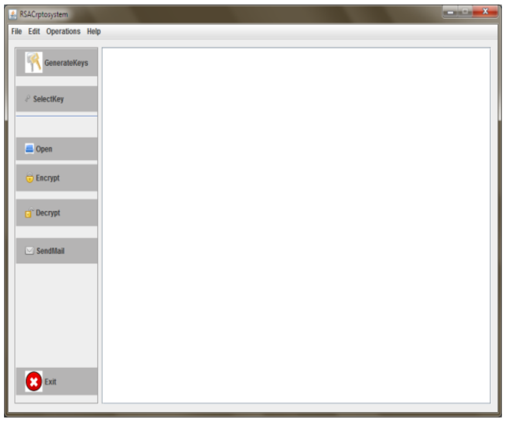
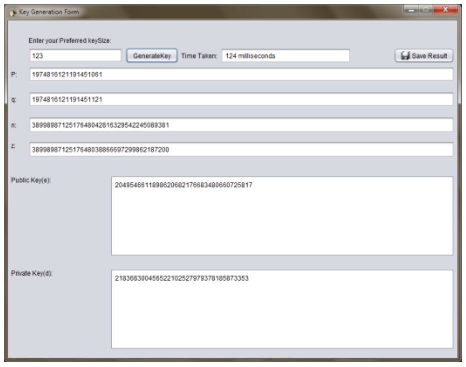


Figure 2: The Application interface

Figure 3 shows the process of key generation in a graphical form; the form contains text fields for representing various key parameters when they get generated, it also contain a button for performing the save operations, it launches the save dialog form when clicked.



Enter your Preferred keySize: 123

Time Taken: 124 milliseconds

P: 1974816121191451061

q: 1974816121191451121

n: 3899898712517648042816329542245089381

z: 3899898712517648038866697299862187200

Public Key(e): 2049546611898620682176683480660725817

Private Key(d): 2183683004565221025279793781858733353

Figure 3: The key generation interface

The user supplies the desired key length and then clicks on generate key button to generate the key. The key and key parameters generated must be saved to file with a name by clicking on the save result button for later reference.

The diagram below shows where the outputs of encryption and decryption will be displayed. This frame contains a text field where the result of encryption and decryption are shown. It also contains a button for the sending of mails; in case the user intends to send the encrypted output as mail. Also in the same frame a user can save the content of the text area by specifying a name with an extension by pressing the save button.

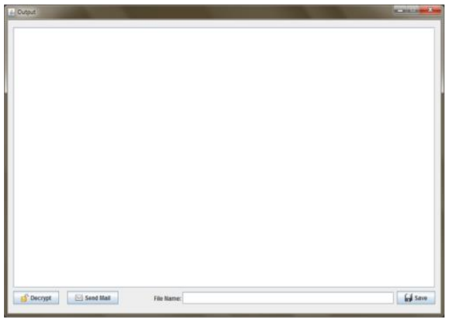


Figure 4: The output interface

A user can from the form above decrypt an encrypted data and save the content of the text area into a file by specifying the file name and clicking on the save button. The user can also launch the mailing interface from here by clicking on the send mail button.

Finally, Figure 5 presents the area where a user if connected to the Internet can send the encrypted file as a mail by selecting the file from the systems memory. Here the user will be asked to provide the necessary parameters needed for authentication on the internet before the file can then be sent to its destination address.

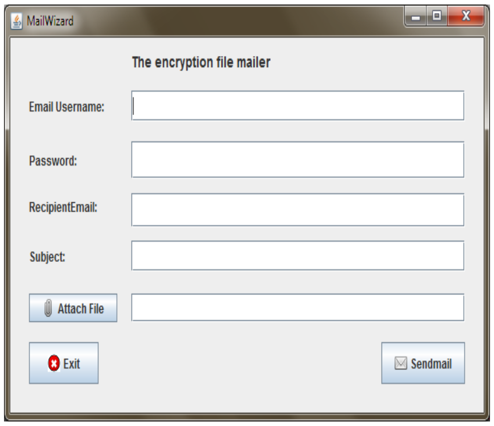


Figure 5: Emailing form interface

The compression and decompression interfaces are as shown in Figure 6 and Figure 7.

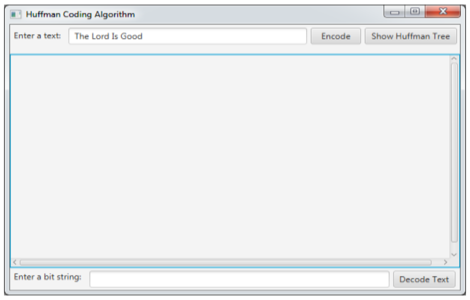
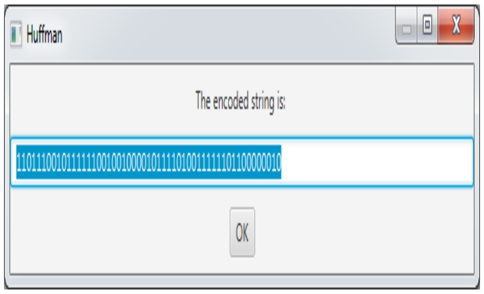


Figure 6: Huffman Sample result



1101110010111111001001000010111101001111110110000010

Figure 7: Huffman Output

6.0 Conclusion

This paper carried out a review on RSA encryption and Huffman compression algorithms and then moves ahead to design a usable encryption-compression algorithm that is portable, user friendly and can be personalized to be used on phones, laptops, and the even for data meant to be transmitted through the Internet. It was implemented using the Java programming language (i.e. object oriented programming language). RSA encryption algorithm has been tested, cryptanalysed but still stand as a rugged algorithm.

In conclusion, computer users need not relent on individual vigilance on important data in this era of indoor and outdoor intruders. Cryptography is the first step to protect confidential data and a further attempt by data owner to compress the encrypted data before sending would be of higher advantage considering storage devices and network load.

References

Apple (2011) “iCloud” Available online at https://www.apple.com/icloud/ accessed on 11 May, 2015

Amazon Web Services Inc. Products and Services (2015), Available online at http://aws.amazon.com/products/?nc2=h_ql_sf_keynote/ Accessed on 9 May, 2015

Arthur J. F., Arlindo L.O. and Mario A.T. Figueiredo (2003) “ On Suitability of suffix Arrays for Lempel-Ziv Data Compression” Insituto Superior de Engeniharia de Lisboa, Portugal

Baker L. and Finkle J. (2011) “Sony PlayStation suffers massive data breach”. Available online at http://www.reuters.com/article/2011/04/26/us-sony-stoldendata-idUSTRE73P6WB20110426 Accessed on 22 April, 2015

Biham E. (1990) “Differential Cryptanalysis of DES-like Cryptosystems”, Journal of Cryptology, Vol. 4 pp. 42-73.

Blelloch G.E. (2013) “Introduction to Data Compression”, Department of Computer Science, Carnegie Mellon University

Comer D. (2009) “Computer Networks and Internets”, 5th edition, Pearson Prentice Hall, NJ pp. 509-520.

Ryan K. L, Stephen G., Rajan V. (2013) “Cloud Computing Vulnerability Incidents: A Statistical Overview ”, White paper

Daemen J. and Vincent R. (2002) “ The Design of Rijindael”, Springer-Verlag, New York

Deitel P. and Deitel H.(2012;. Java How to Program. 9th edition, Prentice Hall, Upper saddle New Jersey

Diffie W. and Hellman E (1976) “New Directions in cryptography” IEEE Transactions of Information Theory, Vol. 22 (5) pp. 644-654,

Evgeny M. (2009) “ The RSA Algorithm”, white paper

Forouzan B.A. and Fegan S.C. (2006) “Data Communications and networking”, 4th edition McGrawHill, New York, pp. 927-994

Gens F. (2009) “New IDC IT Cloud Services Survey: Top Benefits and Challenges” IDC eXchange, Available online at http://blogs.idc.com/ie/?p=730 Accessed on 23 April, 2015

Google (2012) “Google Apps: Energy Efficiency in the Cloud” Available Online at www.google.com/green/pdf/google-apps.pdf Accessed 8th April, 2015

Kaliski B. (1989) “The Mathematics of RSA Public Key Cryptosystem, RSA Laboratory

Khalid S. (2006) introduction to data Compression. Elsevier Incorporation , San Francisco pp 43-45

Khalid S. (2006) Introduction to Data Compression, 3rd edition, Morgan Kaufmann Publisher, Elsevier, San Francisco

Kurose, F. J., & Ross, K. W. (2010). *Computer Networking:A top down Approach.* New york: Addison wesley

Larry P, and Bruce S.D. (2007) Computer Networks: A system Approach, 4th edition pp. 557-625

Microsoft (2014) "OneDrive" Available online at https://onedrive.live.com/ accessed on 11 May, 2015

Rivest R., Shamir L and Addleman L. (1977) "A Mathematical Method for obtaining Digital Signatures and Public key Cryptosystems", Communications of the ACM, Vol. 21(2), pp. 120-126

Robert L (2003) Data Structures and Algorithms in Java, 2nd edition, Sams Publishing, Indianapolis , Indiana pp. 415 - 421

Shahzad F. (2014) "State-of-the-art Survey on Cloud Computing Security Challenges, Approaches and Solution" Proceeding of Computer Science Vol. 37 pp. 357 – 362.

Stallings W. (2007) Data and Computer Communications, 8th edition Pearson Edu. Inc. Upper-Saddle, River, NJ, pp. 701-743

Stallings W. (2011). Cryptography and Network Security Principles and Practice. New york: Prentice Hall.

Vijay K and Sharma V. S (2012) "A Steganography Algorithm for Hidden Image in Image by improved LSB substitution by minimize detection". Journal of Theoretical and Applied Information Technology

Yin, S. (2011) "Dropbox accounts were accessible by anyone for four hours". Available online at http://www.pcmag.com/article2/0,2817,2387343,00.asp Accessed on 22 April, 2015

Biography of Authors

Oladeji, Florence Alaba	*B.Sc., M.Sc., PhD (Computer Science) Data Communications. Goal: to reach to others positively.*
Ajetunmobi, Rukayat A.	*Born on 4th August 1975, B.Sc., M.Sc. (Digital Forensics)*
Ajayi, Olasupo O.	*B.Sc. and M.Sc. (Computer Science), Cloud.*
Adeniyi Adebayo	*A B.Sc. (Computer Science), Cryptography*

Olatunji Michael	*A B.Sc. (Computer Science), Computer Networks and cryptography*

CYBERCRIME ACT OF 2015 AND E-COMMERCE IN NIGERIA: IMPACTS, CHALLENGES AND PROSPECTS

Augustine Nduka Eneanya, Ph.D.
Department Of Political Science
University Of Lagos,
Akoka, Yaba, Lagos.
E-MAIL: austineneanya_2010@yahoo.com

Abstract
The information revolution coupled with the strategic leveraging of the internet has exposed a number of relatively open societies to the dangers of cybercriminals, hackers and malicious cyber activities in commercial business transactions. A new digital revolution is now experienced worldwide, about which 30 percent of global population is actively living in the cyberspace, in the real terms. Virtually all business transactions and other daily human endeavours take place in online platforms. Cybercrimes and the threat it creates are growing in its reach, in accordance with similar growth in information technology. The purpose of this paper, therefore, is to investigate the impact of Cybercrimes Act of 2015 on e-commerce in Nigeria, its challenges and prospects. Qualitative research technique was adopted. Data were collected through secondary sources, such as textbooks, journal articles, internet, archival records, constitution and cybercrime Act. Content, thematic and secondary analytical methods were adopted, using research questions as drivers. Applying the Cybercrimes Act, 2015 as framework of analysis, the paper argues that most of the lacunae which had hitherto rendered the Nigerian cyberspace unsafe for transacting business have now been addressed. Moreover, the coming into force of the cybercrimes Act changed the Nigeria legal landscape significantly, with the overall effects of better securing and further expanding the scope of e-business transactions in Nigeria. The paper further argues that, though the Act has generated fresh risk management issues, it has created institutional framework for enforcement of cybercrimes. There are prospects that the new legal regime will boost the confidence of individuals, firms and companies to transact more businesses and render services online without the fear of falling victims to identity theft, plagiarism or copyright violation.

Key Words: E-commerce, cyberspace, cybcybercrimes, cyber security and malicious cyber activities.