Comparative Study On Encryption Algorithms And Proposing A Data Management Structure

Ali Makhmali, Hajar Mat Jani

Abstract: In implementing a web-based application, security is one of the most important issues to be addressed. Generally, two factors should be addressed prior to implementing the application to ensure security: the structure of data management, and data security strategy. In this research, we try to find a suitable implementation and solution to handle these two problems. These issues first led us to perform a comparative study on several encryption algorithms, and consequently, to find the most suitable one; and second, to find the best management structure of data to ensure a reasonable level of security for the clients of the web-based application. We study and compare the concepts of five encryption algorithms that are most widely used: DES, Triple DES, RSA, Blowfish, and AES. The focus is on the general strategy the encryption algorithms are using, and their implementation applicability on websites or web-based applications. A survey was also conducted in finding the level of awareness and concern regarding online systems' security. Based on the findings of this study, a data management structure for storing confidential data on a server is proposed.

Index Terms: AES, Blowfish, DES, Encryption Algorithms, RSA, Data Management Structure, Triple DES, Website Security

1 Introduction

Acommon purpose of using encryption methods or algorithms in web-based applications is to ensure security when the application needs to have communication with its clients or visitors. This basically means that sending messages including texts, images, and etc. need to be readable at the senders' and receivers' machines, but not the machines and devices it passes through before reaching its final destination. This paper focuses on studying websites that contain confidential information about their clients. These confidential information or details are generated once in a while by managers from the website and need to be sent to the respective clients. The confidentiality of the transmitted messages (i.e. text messages in this case) should ensure that it is secure from the point it is being sent to the client, until it is read by the client. Even though security is the main concern here, the web-based system or application should still be efficient and convenient for the clients. For example, it should not ask the client several passwords or CAPTCHAs along the process of retrieving the confidential information.

2 Research Objectives

The objectives of this paper are as the following:

- To choose a suitable encryption algorithm to apply on clients' information being stored on a server.
- To define a data management structure for the clients' information being stored on the server.

The first objective will cover studying five chosen encryption algorithms out of several available ones. The selection was based on the popularity and security of available encryption algorithms, and their applicability on such a system being studied in this paper. The latter objective covers how the information should be stored on the server to make sure that they cannot be compromised from the server, or if it happens, to ensure they are not readable.

- Ali Makhmali – Masters in Information Technology – Universiti Tenaga Nasional, Malaysia.
  E-mail: ali.mkhm@gmail.com.
- Dr. Hajar Mat Jani is currently a Senior Lecturer at Universiti Tenaga Nasional, Malaysia.
  E-mails: hajar@uniten.edu.my, hajarmj@gmail.com.

3 Literature Review

Encryption is the process of encoding data (i.e. text, images, etc.) in such a way that hackers or eavesdroppers cannot read the data, but only the authorized parties can [1]. It is essential for any organization to encrypt their important data, not only to prevent data from being hacked, but to respect customers' privacy. Generally, it can be seen from different aspects. For example, in a business organization, the data are confidential because compromising the data may cause loss of benefit. In a university, it is important because unauthorized people should not be able (even if they have physical access) to read confidential data. Another example is medical organizations like hospitals and clinics where patients' (or customers') data have to be kept private to respect privacy. In higher level organizations, encryption becomes even more important. For example, militaries and governments have long been using encryptions to facilitate secret communication, which is extremely important. Encryption is now commonly used in protecting information within many kinds of civilian systems. The Computer Security Institute reported that 71% of companies surveyed utilized encryptions for their data in transit, while 53% utilized encryptions of their data in storage [2]. The functionality of encryption can be seen from different aspects; for example, their compatibility on hardware, on different platforms, various programming languages implementations, their licenses and etc. Each of these aspects has their own factors to consider. A study performed by Penchalaiah and Seshadri [3] compares the evaluation of AES and DES encryption algorithms in Hardware Description Language, Verlog. In this paper, the structure and design of the two have been analyzed by considering three criteria: resistance against all known attacks, speed, and code compactness on wide range platforms and design simplicity. Over their study, their finding briefly shows that the simple transformations of AES can quite comfortably be implemented in any high level or low level languages and software tools [3]. One of the commonly used encryption algorithms was DES, which was invented on 1974. Since then many attacks and methods were being recorded that exploited weaknesses of DES [4]. One of the factors that was assumed to make DES not so secure was the limitation of its block size to 64-bit long. It was then replaced by 3DES to increase its security, which was successful in terms of protection, but it was theoretically three times slower than the DES. Encryption algorithms are

not limited to the five being discussed in this paper. The reason these five were selected over the variety was that these are commonly used in websites and web-based applications, which we had to consider in the first place.

4 Comparative Study of Encryption Algorithms

The five encryption algorithms are discussed here in terms of their structure.

4.1 DES

DES stands for Data Encryption Standard. It was previously named "Lucifer" first, and was developed by IBM in 1974. The U.S. government judged it to be difficult to break and the government adopted it as a standard (US patent 3,962,539) that everyone could use it for secure communication. The US National Security Agency (NSA) made several modifications, after which it was adopted as Federal Information Processing Standard (FIPS) standard 46-3 and ANSI standard X3.92 [5]. There are 72,000,000,000,000,000 (72 quadrillion) or more possible encryption keys to be used in DES. It operates on blocks of 64 bits using a secret key that is 56-bit long. A key is usually chosen randomly within the available possible keys. DES is symmetric, which means that the same key used to encrypt the message must be used to decrypt it [6]. DES uses eight predefined S-boxes which have been determined by the U.S. National Security Agency (NSA). Using the S-boxes, groups of six bits are mapped to groups of four bits. These S-boxes are resistant against an attack called differential cryptanalysis, which was first known in the 1990s. Generally, the encryption process of DES is done in 16 rounds. The following steps explain the brief process of DES encryption a message [5]:

1. The input key is used to derive sixteen 48-bit keys (known as subkeys). Each of these keys is then used in each round.
2. The right half is expanded from 32 bits to 48 bits using another fixed table.
3. The result is combined with the subkey for that round using the XOR operation.
4. Using the S-boxes the 48 resulting bits are then transformed again to 32 bits, which are subsequently permutated again using yet another fixed table. In the next round, this combination is used as the new left half.

Fig. 1 illustrates the process of DES encryption visually; the left and right halves are denoted as L0 and R0, and in subsequent rounds as L1, R1, L2, R2, and so on. The function f is responsible for all the mappings described above [5].

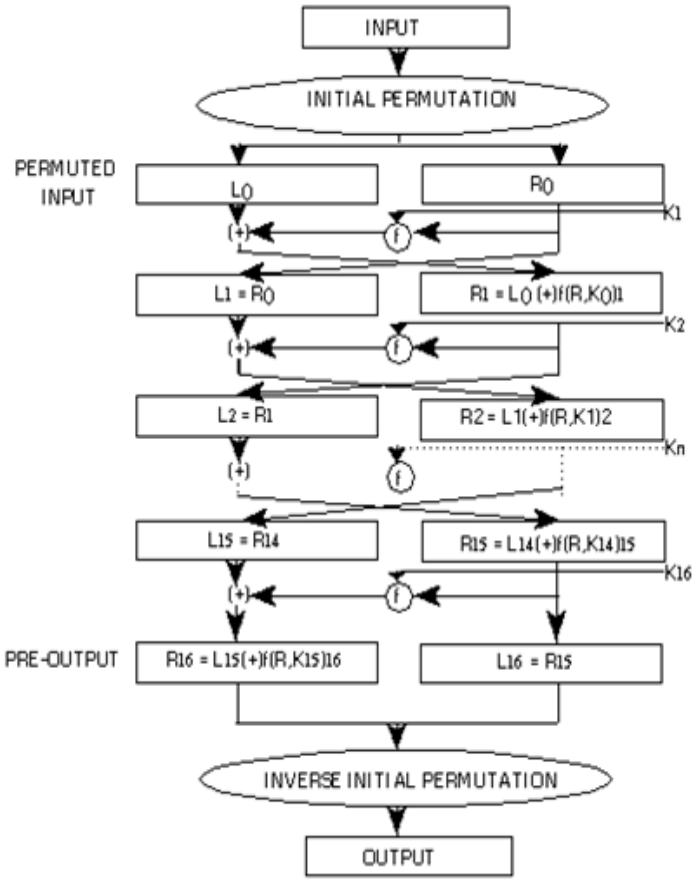

Fig 1. DES encryption process [5]

The number of possible keys that could be used to encrypt DES was very huge for the computers of that time and it was believed that computers could not possibly ever become fast enough to try all of the possible keys. But eventually, in 1998 the Electronic Frontier Foundation (EFF) created a special-purpose machine, costing less than $250,000 which could decrypt a DES encrypted message by trying all of the possible keys in less than three days. This machine could search over 88 billion keys per second [5].

4.2 Triple DES

When it was clear that with powerful machines, DES can be cracked easily, the Triple-DES variant was developed. Triple DES, or 3DES, uses three 56-bit DES keys, which create a key with the total length of 168 bits [5]. The key in 3DES is usually considered one, which is then split into three same length keys during implementation. The process of encrypting messages using 3DES is exactly the same as DES, except that the three keys are used in the following order [7]:

1. First key is used for encryption.
2. Second key is used for decryption.
3. Third key is used for another encryption.

The process of decrypting 3DES is also the same as DES's decryption, except that it is performed in the reversed order of the list above. Even though 3DES is significantly more secure than DES, one of the obvious drawbacks of it is that 3DES takes three times as long as standard DES for encryption and decryption [5].

4.3 RSA Algorithm

The name was derived from its developers: Ron Rivest, Adi Shamir, and Leonard Adleman [8]. It was developed in 1977 and it is widely used in various applications such as Internet Browsers (from Microsoft and Netscape), Lotus Notes, Intuit's Quicken and etc. RSA algorithm is owned, licensed and sold as development kits by RSA security [8]. RSA is also widely used in electronic commerce protocols (SSL). Unlike DES (and other encryptions discussed in this paper), RSA is the only one that is asymmetric, which means that the same key for encryption cannot be used (alone) for decryption. It uses a combination of public and private keys (discussed below). Generally, the process of creating the public key and private key in RSA is as follows [9]:

Two large prime numbers of the same size are generated; we call them *p* and *q*. The product of *p* and *q* gives the value *n*. In this case, *p* and *q* should be large enough (at least 100 digits), and are kept secret to the sender's side. Considering the fact that *n* is the product of two very large prime numbers, it is practically impossible to derive out the two (i.e. *p* and *q*) from a given n [10].

A random integer is then selected, named *e*; where *e* should be greater than 1, and the *gcd (e, (p-1), (q-1)) = 1* (the value of *e* is called *public exponent*).

Next, find the multiplicative inverse of *e* modulo *(p-1)(q-1)*, named *d* (the value of *d* is called *private exponent*). The public key is (*n*, *e*) and the private key is *d*. One of the major benefits of RSA algorithm is that the public key can be created and sent to someone (e.g. from administrative) to encrypt a message, but only the private key of the dedicated receiver can be used to decrypt it.

A brief visual presentation of how RSA works entirely is shown in Table 1 [8]:

TABLE 1
RSA ALGORITHM PROCESS [8]

To:	*User:*	*Kind of key*
Send an encrypted message	Use the receiver's	Public key
Send an encrypted signature	Use the sender's	Private key
Decrypt an encrypted message	Use the receiver's	Private key
Decrypt an encrypted signature (and authenticate the sender)	Use the sender's	Public key

Even though RSA seems very secure and has been widely used by many companies and applications, an uncommon way of manipulation has claimed to crack RSA. Three members of the University of Michigan have claimed they could break it simply by tweaking a device's power supply [11]. Their method was to wave the CPU's voltage in such a way that it generates a single hardware error per clock cycle, compromising the server to flip single bits of the private key at a time. Several iterations of the process will slowly piece together the password. On a Pentium 4 chips and 104 hours of processing time, they could hack a 1024-bit encryption in OpenSSL [11].

4.4 Blowfish Algorithm

Blowfish was designed in 1993 by Bruce Schneier, and it was mainly designed to be used in embedded systems [12, 13]. Blowfish key size can be of any length up to 448, and like the other encryption algorithms studied in this paper (except RSA), Blowfish is also a symmetric algorithm, which means that the same key used to encrypt a message, is used to decrypt it [12][13]. Blowfish is Block Cipher, meaning that it will divide the plain text into fixed-length blocks during encryption and decryption and the block size is 64 bits (if the size of the plain text is not a multiple of 64, it is first padded to match the block size) [12][13]. The process of Blowfish algorithm is as follows [13]:

1. The given key will be converted into several subkey arrays totaling 4168 bytes. These subkeys should be pre-computed before the actual encryption and decryption take place.
2. Blowfish has 16 rounds Feistel network. The input is a 64-bit data element, x. Divide x into two 32-bit halves: xL, xR. Then,
   a. for i = 1 to 16:
   b. xL = xL XOR Pi
   c. xR = F(xL) XOR xR
   d. Swap xL and xR
3. After all of the sixteen rounds have passed, the xL and xR are swapped again to undo the last swap.
4. Then perform, xR = xR XOR P17 and xL = xL XOR P18.
5. Finally, recombine xL and xR to get the cipher text.

Decryption is exactly the same as encryption, except that P1, P2,..., P18 are used in reversed order. Implementations of Blowfish that require the fastest speeds should unroll the loop and ensure that all subkeys are stored in the cache. One of the drawbacks of Blowfish is that it is only suitable for applications where the key does change frequently, like being used in communication link or automatic file encryptor. It could be a bit time consuming in some implementations regarding its complicated encryption functions [13].

4.5 AES Algorithm

The US national Institute of Standards and Technology (NIST) indicated that DES encryption algorithm should only be used for legacy systems, and instead, the 3DES encryption (discussed earlier) should be used [14]. With the modern technology and new encryption algorithms being developed, one problem restricting DES and 3DES to be used widely is the slow process of encryption and decryption. In fact, 3DES is even three times slower than DES. Another drawback of these was the small block size of 64 bits, which is not secure anymore. All these issues led NIST to call for proposals for a new Advanced Encryption Standard in 1997. After three subsequent evaluation rounds up to 2001, NIST declared one of the proposals "Rijndael" as the new AES and published it as a final standard (FIPS PUB 197) [14]. Rijndael was proposed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen [14]. The proposed encryption's key size varies between 128, 192 and 256 bits; but only the key size of 128

bits was approved as the AES standard [14]. AES is symmetric and uses 128 bits of block sizes. Generally, the working progress of AES can be divided into four main functions: SubBytes, ShiftRows, MixColumns and AddRoundKey. It contains 10 rounds of encryption iterations. The following is a brief explanation of AES encryption process: (1) Key Schedule and (2) Encryption. The key schedule is the process of generating several subkeys derived out of the given key. Then, the steps taken to encrypt are as follows [14]:

1. ***SubBytes*:** Each byte of the state is replaced with its coordination to the S-Box.
2. ***ShiftRows*:** Row 1 of the state is unchanged. Row 2 is shifted to the left by 1 byte.
3. ***MixColumns*:** At this step each column of the state is multiplied with a fixed matrix.
4. ***AddRoundKey*:** Each column of the current state is then XORed by the same column from the subkey.

These steps are repeated for 10 rounds (in 128 bits), and some minor functions will generate the encrypted text. The decryption algorithm is just the reversed steps of the above process.

5 Scaling Public Concern

One component of this study was to scale the public concern about online security. The following opinions were gathered from students and non-students at Universiti Tenaga Nasional (UNITEN). UNITEN was considered a suitable case because most of the people in academic organizations are students and staff. It is almost certain that everyone in a university would have at least one online account (e.g. email, forum, bank, etc.) that deals with users' information, and the assumption is to consider online security a concern. A total of 66 respondents have been asked to share their opinions. The following are the results and information derived from the survey. The first section of the survey asked respondents about their designation at UNITEN and their field of study or profession. 88% of the respondents have been students, 3% lecturers, 6% researchers, and 3% others. Fig. 2 illustrates the study or profession field of the respondents.

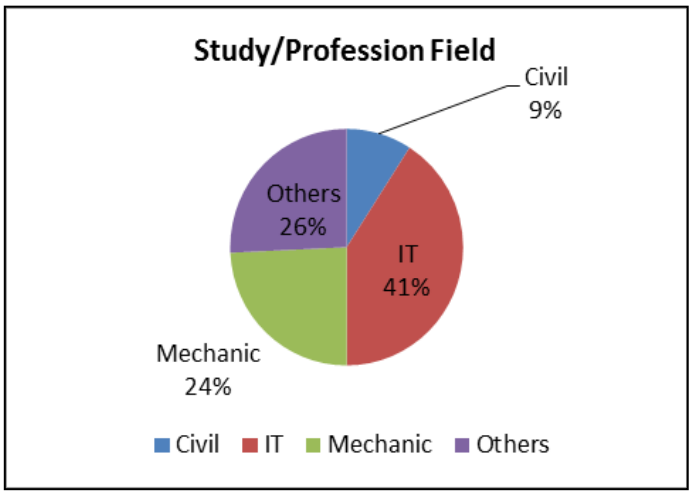

Fig 2. Study/Profession field

Fig. 3 illustrates answers to the following question: "Do you use any Web-based applications, Social Networks, or any similar online service that deals with your identifying personal information such as your name and location?"

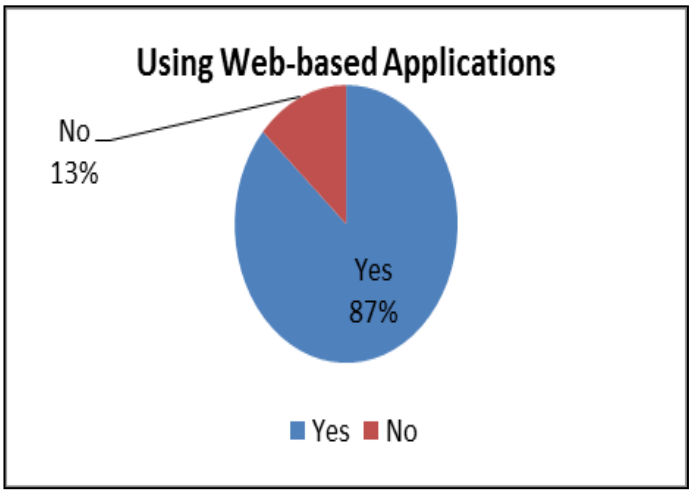

Fig 3. Using web-based applications

The next question asked the respondents if they send or share their confidential information online; information such as credit card monthly report, an account's password or a personal confidential email. Fig. 4 shows the responses.

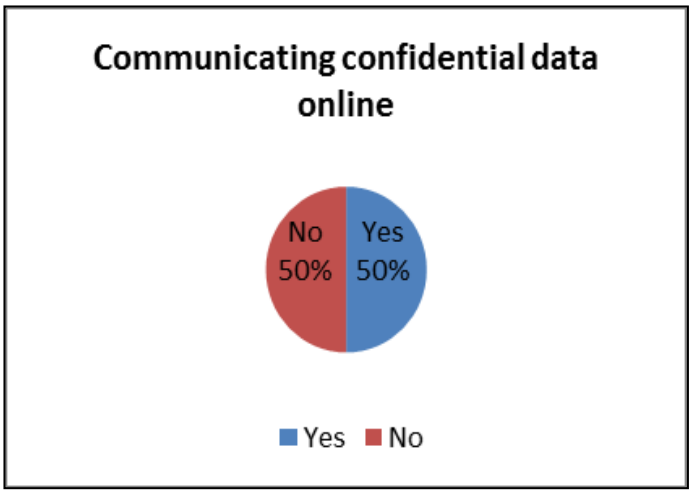

Fig 4. Communicating confidential data online

Fig. 5 illustrates respondents answers to a question asking if they have ever experienced a loss of password, unauthorized access to their email or similar issues that occurred due to using online systems.

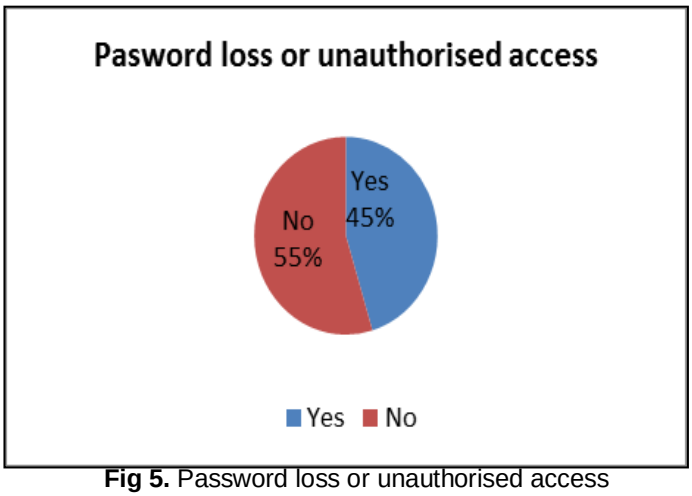

Fig 5. Password loss or unauthorised access

Respondents were asked if they have any information about terms like spams, junk emails, brute force attacks, crack, hack, spyware or similar issues. Fig. 6 presents the answers to this question.

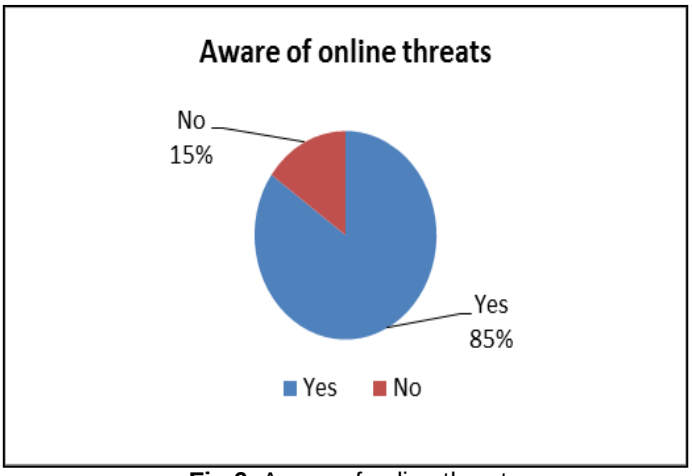

Fig 6. Aware of online threats

To become a bit more specific, respondents were asked if they know what encryption algorithms are and what they are used for. Fig. 7 illustrates their answers.

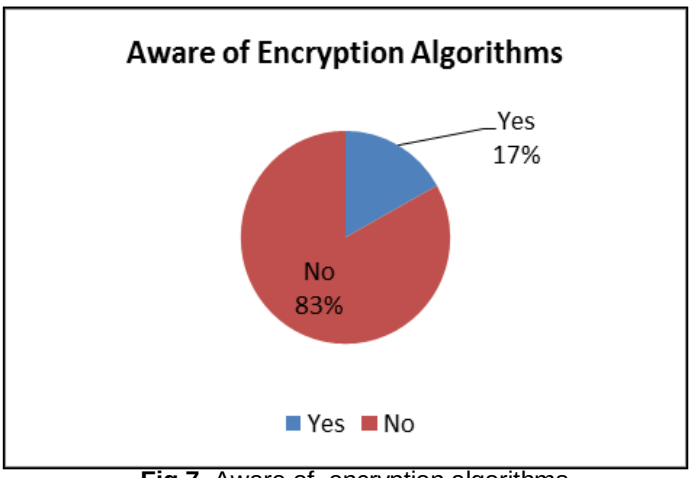

Fig 7. Aware of encryption algorithms

Out of the small portion of respondents who knew what encryption algorithms are, they were asked if they know, or have information about a given list of encryption algorithms including AES, DES, 3DES, Serpent, RSA, Blowfish, Twofish or others. Fig. 8 illustrates the answers.

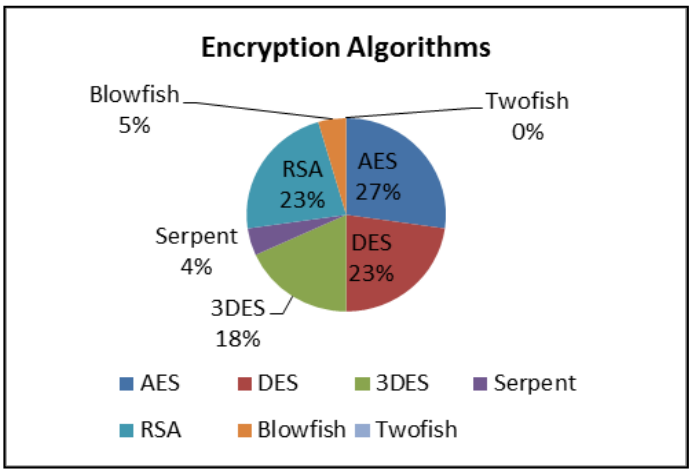

Fig 8. Encryption algorithms

Next, the respondents were asked to rate their own knowledge about online security. They could select an option out of four provided: *very low*, *low*, *medium*, and *high*. Fig. 9 shows the results.

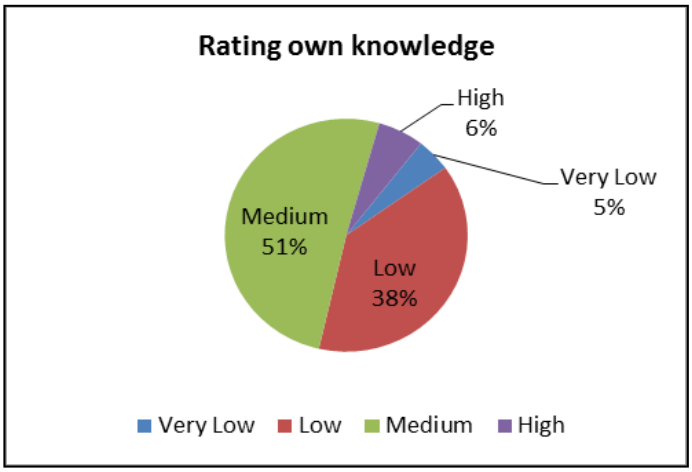

Fig 9. Rating own knowledge about online security

Eventually, the respondents were asked to prioritize their preference between three features: *speed*, *convenience*, and *security*. Fig. 10 illustrates the respondents' preferences.

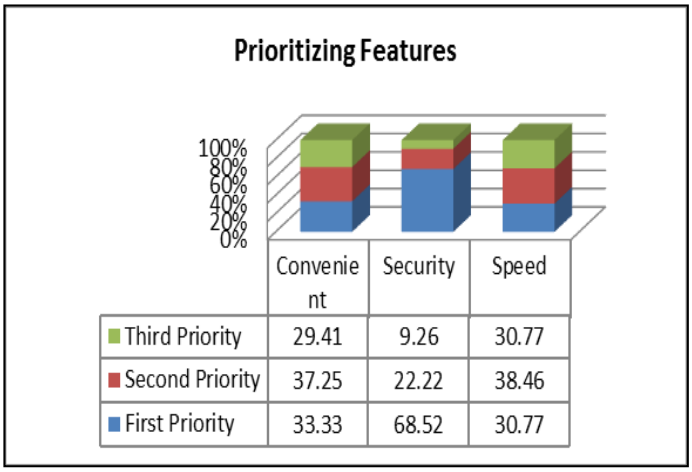

Fig 10. Prioritizing features to select an online service

6 Results Discussion

Based on the reviews and research performed in this paper over the five encryption algorithms, the following is the brief outcome of this study. It should be noted that some extra information presented here may have not been covered previously.

Table 2 reviews the basic information of the five encryption algorithms.

TABLE 2
BASIC INFORMATION COMPARISON

	Abbreviation of	*Invented by*	*Invented in*
DES	Data Encryption Standard	IBM	1977
3DES	Triple DES	IBM	-
RSA	Rivest-Shamir-Adleman	Ron Rivest, Adi Shamir and Leonard Adleman	1977
Blowfish	N/A	Bruce Schneier	1993
AES	Advanced Encryption Standard	Joan Daemen and Vincent Rijmen	2001

Table 3 presents a summary of the key types and block features of the five encryption algorithms that are discussed in this paper.

TABLE 3
KEY AND BLOCK COMPARISON

	Key Type	*Key Size*	*Block Size*
DES	Symmetric	64 bits (56 bits are actually used)	64 bits
3DES	Symmetric	192 bits (combination of three 64bit keys)	64 bits
RSA	Asymmetric	Not specified	Not Specified
Blowfish	Symmetric	64 bits	From 32 bits to 448 bits
AES	Symmetric	128 bits	128 bits, 192 bits and 256 bits

7 Conclusion and Future Work

After reviewing the specifications of the encryption algorithms listed in Section 4 and their applicability to our website, AES is chosen to be used in our study. This decision is made based on the following reasons:

1. ***The block size*:** The data being encrypted in our system are long and could contain up to several pages of data. Taking this fact into consideration, AES is the best choice among the five encryption algorithms. Even though Blowfish could have a larger block size, but it has been proven to be crack-able against brute force attacks.
2. ***The key size*:** Comparatively, AES has the larger key size along the five discussed. Even though 3DES has literally a larger key size (i.e. 192), but that is actually a combination of three 64 bit keys; considering the slow process speed of 3DES, AES is the preferred choice.
3. ***Applicability*:** AES is open source, and can be easily implemented in web-based applications using a variety of programming languages such as PHP. It is even possible to modify the actual process of AES since you have access and can clearly trace the process; though it is not advised because its process is notably complicated and a small false manipulation in the process may cause inconsistencies along the rest of it.

7.1 Data Management Structure: A Proposal

It is unsecure to store both the encrypted message and its encryption key on the same server. It is also unsafe to email the encryption key to the client, while if their email address is compromised, so is the key. To overcome this issue, the following is the proposed data management structure to ensure security of the system:

1. Based on the essence of our service, physical existence of the client is necessary (at least in the first visit). To avoid communicating a password or a key online, a *passphrase* is generated by a computer program (following a certain pattern), and it is then given physically to the client. The *passphrase* is also hashed and stored in a database of the server.
2. Once a manager of our service attempts to create a message for the client, the system will automatically generate a random encryption key for that message. The encryption key is stored in the server's database,

along with an ID of the message.
3. The created message is encrypted using the encryption key, but instead of storing the encrypted message in the database, it is actually stored in a file on the server. This is to ensure that if the database has been compromised (which contains the encryption key), the message is not in the same place. No plain message has been stored either in the database or files.
4. The path and encrypted message ID are both stored in the database.
5. An email containing a link is sent to the client. The link is to a specific webpage of the website, and only contains the message ID.
6. Once the client opens the link using an Internet browser, a page opens and asks for the client's *passphrase* given in step one.
7. If the correct *passphrase* is entered, the system automatically finds the message (based on the given ID in the link), and also the path to its actual file. It then retrieves the encryption key related to the message from the database, and eventually decrypts the message using the encryption key. The message is displayed to the client.

Other aspects of the process, such as the client losing the *passphrase*, etc. are detailed instructions that depend on the system's structure, and therefore, will not be discussed here.

7.2 Future Work

Further improvements can be performed over the proposed data management structure. For example, the path to the file and the encryption key are both currently stored as plain text in the database. There could be alternatives, which ensure that they are not stored at the same place, or at least not as plain text. Hashing could not be performed in this case because they need to be automatically retrieved by the system (once a client needs to read them). Applying another level of encryption will not resolve this issue as well, because a standard encryption has the potential of being compromised using patter detections. Implementing another encryption method or algorithm to encrypt them (using a key) may seem redundant, and may eventually lead to the same problem.

ACKNOWLEDGMENT

This research study was funded by the Fundamental Research Grant Scheme (FRGS), Ministry of Higher Education (MOHE), Malaysia.

REFERENCES

[1] G. Oded, *Foundations of Cryptography: Volume 2, Basic Applications*, Vol. 2. Cambridge University Press, 2004.

[2] R. Richardson, "CSI Computer Crime and Security Survey," *Computer Security Institute*, 2008.

[3] N. Penchalaiah and R. Seshadri, "Effective Comparison and Evaluation of DES and Rijndael Algorithm (AES)," *International Journal on Computer Science and Engineering* Vol. 02, No. 05, 2010.

[4] S.P. Singh and R. Maini, "Comparison of Data Encryption Algorithms," *International Journal of Computer Science and Communication*, Vol. 2, No. 1, January-June 2011, pp. 125-127, 2011.

[5] A. Engelfriet, "The DES encryption algorithm," Available at www.iusmentis.com/technology/encryption/des/, Accessed on 18 Oct. 2012.

[6] M. Rouse, "Data Enryption Standards (DES)," 2006, Available at http://searchsecurity.techtarget.com/definition/Data-Encryption-Standard, Accessed on 18 Oct. 2012.

[7] Tropical Software, "Triple DES Encryption," Available at http://www.tropsoft.com/strongenc/des3.htm, Accessed on 12 Oct. 2012.

[8] M. Rouse, "RSA Algorithm," 2005, Available at http://searchsecurity.techtarget.com/definition/RSA, Accessed on 16 Oct. 2012.

[9] J.S. Coron, *How to Implement RSA in Practice,* Universite du Luxembourg, 2009.

[10] D. Joyner, "Application: RSA Encryption," 2002, Available at http://www.usna.edu/Users/math/wdj/_files/documents/book/node44.html, Accessed on 22 Oct. 2012.

[11] S. Hollister, "1024-bit RSA encryption cracked by carefully starving CPU of electricity," 2010, Available at http://www.engadget.com/2010/03/09/1024-bit-rsa-encryption-cracked-by-carefully-starving-cpu-of-ele/, Accessed on 15 Oct. 2012.

[12] B. Gatliff, "Encrypting data with the Blowfish algorithm," 2003, Available at http://www.design-reuse.com/articles/5922/encrypting-data-with-the-blowfish-algorithm.html, Accessed on 8 Nov. 2012.

[13] B. Schneier, "Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)," *Fast Software Encryption, Cambridge Security Workshop Proceedings, Springer-Verlag,* 1994, Available at http://www.schneier.com/paper-blowfish-fse.html, Accessed on 28 Oct. 2012.

[14] C. Paar and J. Pelzl, *Understanding Cryptography: A Textbook for Students and Practitioners*, ISBN 978-3-642-04101-3, 2010.