Information and Knowledge Management
ISSN 2224-5758 (Paper) ISSN 2224-896X (Online)
Vol.3, No.4, 2013

Java Implementation and Performance Evaluation of Some Cryptographic Ciphers under WinXP and Linux Operating System Platforms

Najib A. kofahi

Department of Computer Science, Faculty of Information Technology and Computer Sciences

Yarmouk University - Irbid, Jordan, E-mail: nkofahi@yu.edu.jo

Abstract

In this article we present a performance comparison of four symmetric block ciphers namely DES, Triple-DES, AES, and Blowfish. Performance evaluation based on CPU execution time is conducted under WinXP and Ubuntu /Linux version 8.10 operating system platforms. The study is conducted using Java programming language, Java Cryptography Architecture (JCA) and Java Cryptography Extension (JCE). The evaluation of the performance of these algorithms is done for encryption, decryption and key generation operations.

Keywords: DES, Triple-DES, AES, Blowfish, Performance Evaluation, Cryptography, JCA & JCE, Operating Systems.

1. Introduction

Conducting commercial transactions (i.e., e-commerce) over the Internet has witnessed an important development and is widely spread these days. Among the factors that causes this includes developments in technology, networks and data security. This growth in e-commerce use lays an important role in the global development of economy and can be a key driver of increasing sales while using fewer production resources. The need to protect and secure e-commerce and other electronic transactions has placed a great demand on a strong and efficient internet security system.

These days, developments in security over the Internet is the key to confidence for conducting Internet services and for people wanting to protect their sensitive information, or doing business and all kinds of communications over the Internet.

Many encryption/decryption algorithms were developed and employed for this purpose with varying properties and advantages/disadvantages of each. The two general types of key-based encryption/decryption algorithms are symmetric (also called conventional) algorithms (Douglas 2005), (William Stallings 2009), (Eashwar and Madhuri 2003) and asymmetric (or public-key) algorithms (Douglas 2005), (William Stallings 2009), (Kofahi, 2006), (CGI Group 2013), (Jawahar and Nagesh 2011).

In the first approach, i.e., symmetric algorithms (En Wikipedia Cryptography 2013), the same (secret) key generated is used for both encryption and decryption operations. The two families of symmetric algorithms are block ciphers and stream ciphers. Data Encryption Standard (DES), Advanced Encryption Standard (AES), Triple-DES and Blowfish are considered the most popular symmetric block ciphers.

Asymmetric algorithms use public key for encryption and private key for decryption (Omar et al. 2012). Most popular asymmetric ciphers are RSA, Diffie-Hellman, and ECC.

In this work the implementation of four symmetric algorithms namely DES, Triple-DES, AES and Blowfish using Java Cryptography Architecture (JCA) and Java Cryptography Extension (JCE) is carried out. Performance evaluation based on their execution time will be presented under WinXP and Linux platforms.

Section 2 gives a brief description of the algorithms to be compared. The methodology and an overview of JCA and

the Java implementation of the ciphers under consideration are discussed in Section 3. Section 4 introduces the testing methodology and presents the results obtained from the implementation. Finally, Section 5 gives the conclusion of the study and points to the future work.

2. The Ciphers under consideration

Performance evaluation based on execution time of some encryption/decryption algorithms was reported (Kofahi 2006), (Jawahar and Nagesh 2011), (En Wikipedia Data_Encryption_Standard 2013), (Elminaam et al. 2010). In this paper we attempt to impleme, nt four algorithms in Java then compare their performance evaluation based on their execution time under WinXP and Linux platforms. In the next subsections a brief description of how each of these algorithms works is presented.

2.1. DES Algorithm

The Data Encryption Standard (DES) (Eashwar and Madhuri 2003), (Singh and Maini 2011), is a block ciphers which have been designated as cryptography standard by the US government. The DES has a pronounced effect on the development of modern cryptography in the academic society. DES (especially its still-approved and much more secure Triple-DES variant) remains quite popular; it is used across a wide range of applications, from ATM encryption to e-mail privacy and secure remote access. Many other block ciphers have been designed and released, with considerable variation in quality.

In DES, 16-cycle Feistel system is used for encryption, with overall 56-bits key bit key permuted into 16 48-bits sub keys, one for each cycle. For decryption, an identical algorithm is used, but the order of sub keys is reversed. The left (L) and right (R) blocks are 32-bits each yielding 64-bits bloc size. The hash function specified by the standard using the "S-boxes", which takes 32-bits data block and one of the 48-bits sub keys as input and produce 32-bits output. In the block diagram the 64-bits key is used, but 8-bits are used only for parity.

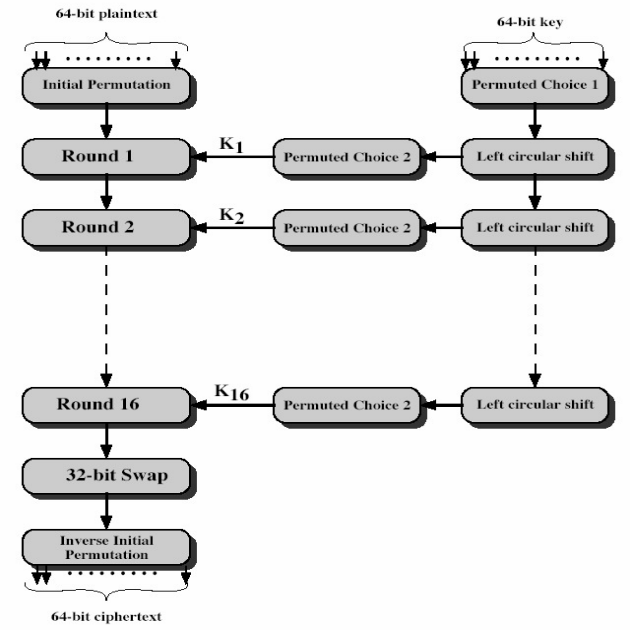

Figure 1. DES encryption process block diagram (Douglas 2005)

2.2. Triple-DES algorithm

T-DES (En Wikipedia Triple_DES 2013), (Hamdan et al. 2010), is a modified version of the DES algorithm that

improves the security strength of the DES by applying the algorithm three times in succession with three different keys each of 56-bits. T-DES thus simply extends the key size of DES by applying the algorithm three times as shown in Figure 2 below.

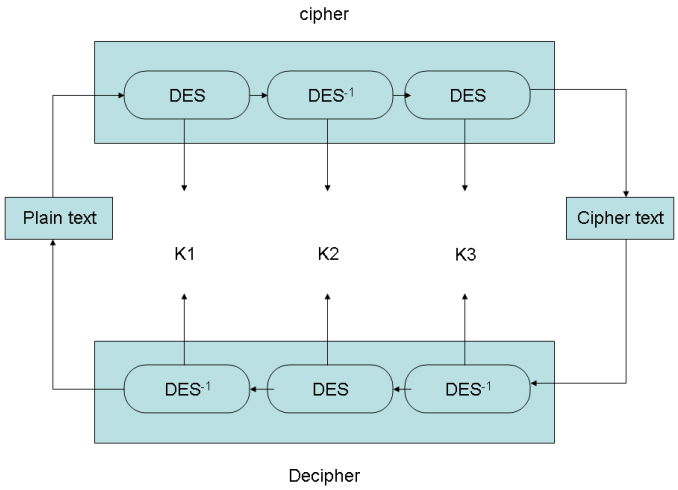

Figure 2. T-DES block diagram

In the T-DES algorithm, the combined key size is thus 168 bits (3 times 56), which is beyond the reach of brute-force techniques. TDES was the answer to the security flaws of the DES without the need to designing a whole new cryptosystem.

2.3. AES algorithm

NIST, The National Institute of Standards and Technology issued a call in 1997 for proposals for an encryption standard as a new official standard. The new standard is called the advanced encryption standard (AES). NIST required that AES should have security strength equal to or better than the TDES and significantly improved efficiency (Guido Bertoni1 et al. 2002).

NIST required that AES should have security strength equal to or better than the TDES and significantly improved efficiency (Hamdan et al. 2010), (En Wikipedia Advanced_Encryption_Standard 2013). AES consists of four invertible different stages that make up a standard round (William Stallings 2009). The stages are iterated 10 times for 128-bit key, 12 times for 192-bit key, and 14 times for 256-bit key. The four stages that consists the standard round are:

- Substitute bytes: nonlinear procedure that uses the S-box to perform byte by byte of the data block.
- Shift rows: a simple transformation that uses permutation that shifts the bytes within the data block in cyclic fashion.
- Mix columns: a simple transformation that uses arithmetic over $GF(2^8)$ to group 4-bytes together forming 4-term polynomial then multiplies the polynomials with a fixed polynomial mod (x^4+1).
- Add round key: bitwise XOR of the current block with a portion of the expanded key.

The block that depicts the encryption/decryption process stages is shown in Figure 3.

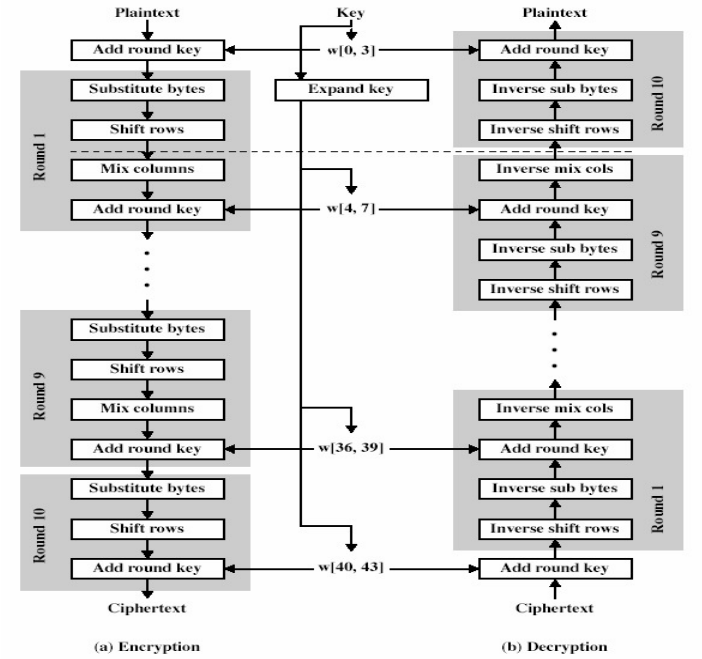

Figure 3. Block diagram of main steps of AES (Eashwar and Madhuri 2003)

2.4. Blowfish algorithm

Blowfish uses 64-bits block size, and a variable key size ranges from 32-bits to 448-bits. Blowfish algorithm consists two parts: key expansion part and data encryption part. Key expansion converts the key into several sub key sub key arrays of total 4168 bytes. Data encryption part is done via 16 round Feistel network. Each round consists of the key permutation, and the key- and data- dependent substitution.

All operations are XORs and addition on 32-bits words. The details descriptions can be found in (Douglas 2005), (Anand and Karthikeyan 2012). Figure 4 shows the block diagram.

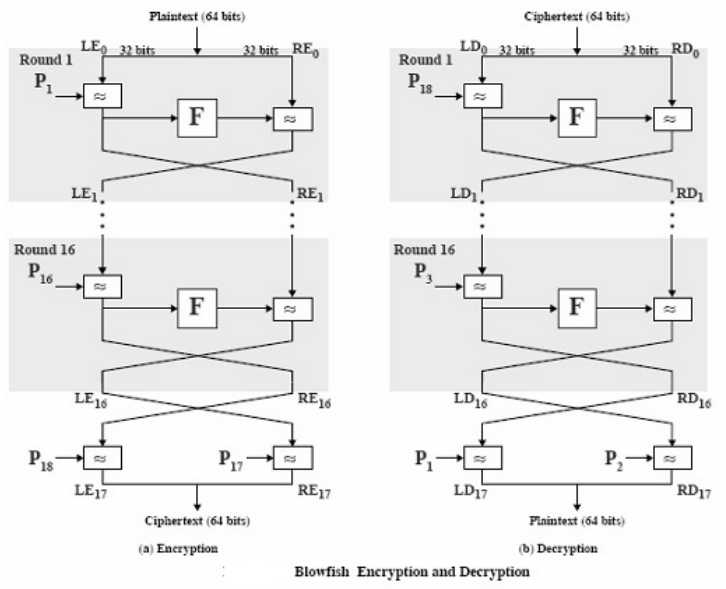

Figure 4: block diagram of blowfish (Douglas 2005)

3. Methodology and Java Cryptography Architecture

This section briefly introduces the implementation diagrams and the Java (JCA, JCE) tools used to implement the above mentioned algorithms under WinXP and Linux. The algorithms are implemented and tested in the same environment. The execution time is measured for the three phases: key generation, encryption process, and decryption.

3.1 Encryption/Decryption Processes

The following steps are used to generate the key pair for ECC:

- Initialize the required pair of keys and their sizes and specify the algorithm and provider that support them.
- Specify the suitable curve by specifying the parameters of elliptic curve.
- Generate pair of Keys.
- Determine public and private keys.

The block diagram of the encryption phase for the algorithms under consideration is shown in Figures 5.

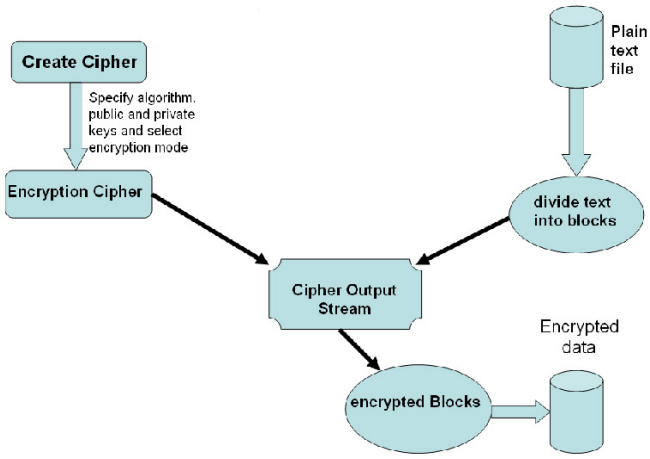

Figure 5. Block diagram of encryption stage (Kofahi 2006)

The block diagram of the decryption phase for the algorithms under consideration is shown in Figures 6.

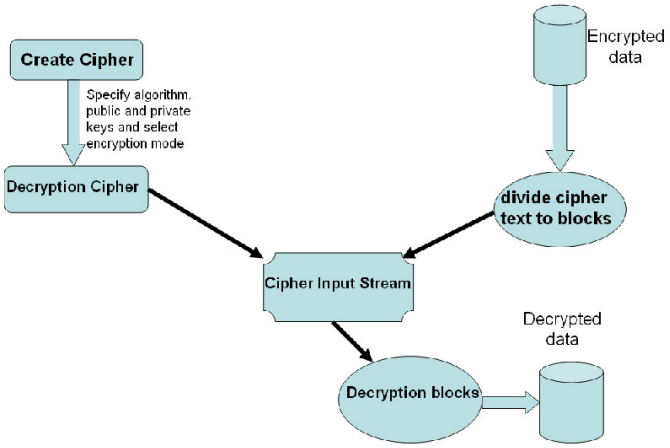

Figure 6. Block diagram of decryption stage (Kofahi 2006)

3.2. Java and Java Cryptography Architecture

Java, Java Cryptography Architecture (JCA), and Java Cryptography Extension (JCE) were used as implementation tools. JCA is designed to allow developers to incorporate both low-level and high-level security functionality into their programs (En Wikipedia Java_cryptography 2013), (Marco *et al.* 1999), (Jamie *et al.* 2000), (Java Sun j2se 2013), (*Java Sun* 2013), (Marco Pistoia *et al.* 2013). It includes the parts of the Java 2 SDK Security API related to cryptography, as well as a set of conventions and specifications. It also includes a "provider" architecture that allows for multiple and interoperable cryptography implementations.

Two Java APIs JCA and JCE both part of J2SE SDK v1.6 (Larry *et al.* 2001), (Flanagan 1999), define the general architecture and specific services for cryptographic operations. J2SE v1.6 comes with nine bundled providers: “SUN1.6”, “SunJSSE1.6”, “SunRsaSign1.5”, “SunJCE1.6”, "SunSASL1.5", "XMLDSig1.0",

"SunPCSC1.6", "SunMSCAPI1.6" and “SunJGSS1.0”. Other third party providers like “BC1.6” (Bouncycastle 2013), “Flexi1.6” (CDC Informatic 2013) were also used in the implementation.

3.3. Packages and Classes used in implementation

The following packages and classes are used in the program:

```
java.io.*;                    // to initialize files using (File, FileInputStream, FileOutputStram)  classes.
 java.security.    KeyPairGenerator;  // to initialize pair of Keys and which algorithm  and provider //support theses
                        key and to initialize the size of the key  .
java.security.KeyPair;        // to generate pair of Keys.
                        // to determine the public and the private keys.
java.security.PrivateKey;
java.security.PublicKey;
java.security.SecureRandom;        // this class are used when the pair of keys are
                                   //generated to choose parameter randomly.
java.security.Security;                    //to add the two items of this provider and  to //use there code
                                           in program.

javax.crypto.Cipher;                       //to initialize cipher for the algorithm and  specify the mode
                                           which we want ENCRYPT-MODE or DECRYPT-MODE
javax.crypto.CipherInputStream;    // to decrypt the file.
javax.crypto.CipherOutputStream;   // to encrypt the file.
```

4. Experimental setup

We give here the experimental set up, and graphically present the execution time of each algorithm for the sake of performance evaluation. We experiment with different plaintext file sizes and different file contents WinXp and Linux platforms.

4.1 Experiment

The four algorithms were run on two different tests; the first test executed under windows XP professional version 2002 service pack 2 on Intel®, Pentium® with Dual-Core CPU speed of 1.60 GHz and a total of 1GB of RAM.

The second test was conducted in the same environment and was executed under Ubuntu /Linux version 8.10 released in October 2008. The four ciphers are performed on the same files of sizes (100KB, 1MB and 10 MB). The algorithms were tested using key sizes shown in Table 1.

Table 1. Algorithms key sizes

Algorithm name	**Key size**
DES	56
T-DES	112
AES	128
Blowfish	56

The experiment was repeated 100 times for each algorithm and for key generation, the encryption operation and the decryption operation.

The average of the 100 runs for each operation was computed for each algorithm.

4.2. Results

In the experiment, the CPU execution time is computed for the four algorithms under WinXP and linux. CPU time includes: system (kernel) time and user time. The system time is the execution time in kernel mode, and the user time is execution time in user mode.

4.2.1. First Test Results

The CPU execution time in seconds for the four algorithms under WinXP with file size of 100 KB is shown in Figure 7.

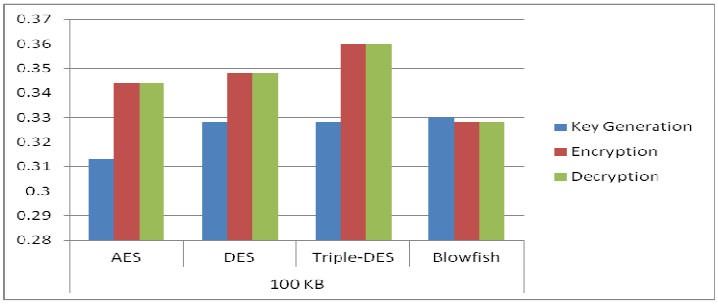

Figure 7. CPU execution time in seconds (WinXP, 100KB)

The CPU execution time in seconds for the four algorithms under WinXP with file size of 1MB is shown in Figure 8.

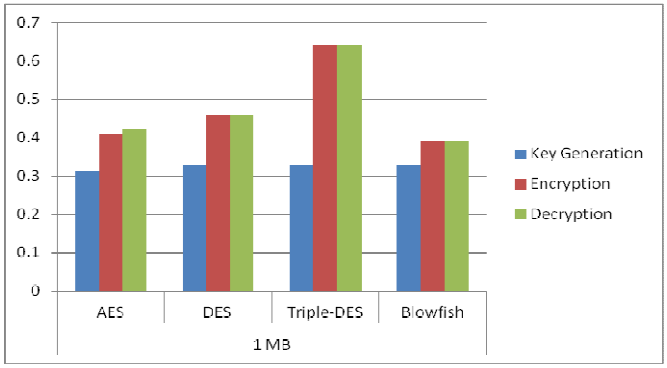

Figure 8. CPU execution time in seconds (WinXP, 1MB)

Also, the CPU execution time in seconds for the four algorithms under WinXP with file size of 10MB is shown in Figure 9.

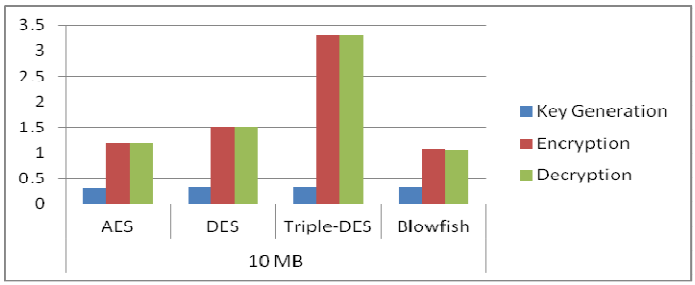

Figure 9. CPU execution time in seconds (WinXP, 10MB)

4.2.2. Second Test Results

The CPU execution time in seconds, for the four algorithms under Linux with file size of 100KB, is shown in Figure 10.

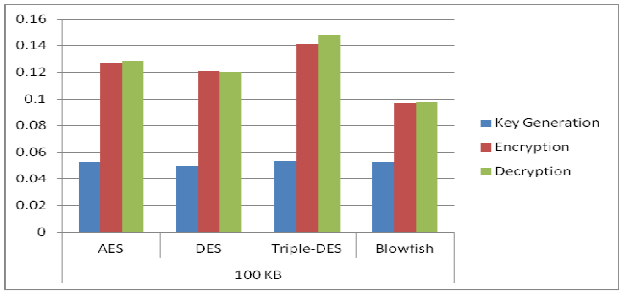

Figure 10. CPU execution time in seconds (Linux, 100KB)

The CPU execution time in seconds, for the four algorithms under Linux with file size of 1MB, is shown in Figure 11.

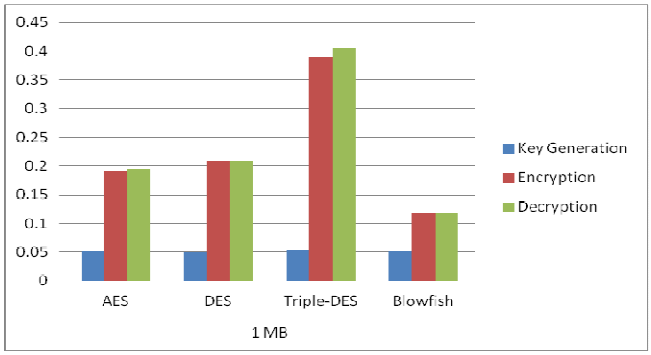

Figure 11. CPU execution time in seconds (Linux, 1MB)

The CPU execution time in seconds, for the four algorithms under Linux with file size of 10MB, is shown in Figure 12.

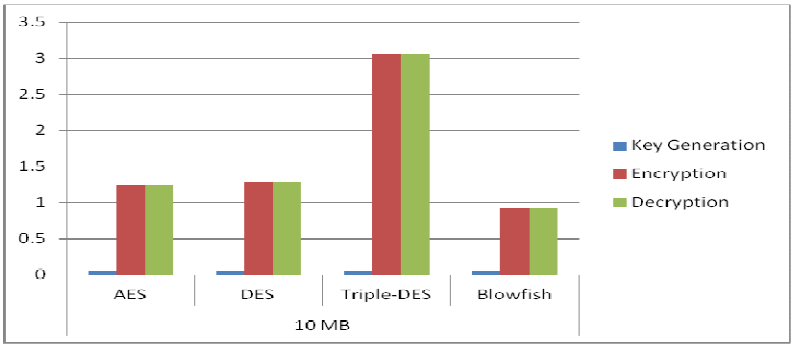

Figure 12. CPU execution time in seconds (Linux, 10MB)

From the first test results for WinXP (Figures 7, 8, and 9), it is clear that the Blowfish algorithm is the fastest followed by AES then DES and then the TDES algorithm. It is obviously clear that because of the increased complexity incorporated within the TDES, which is needed to address the security issues that exist in the DES algorithm, is degrading the performance in terms of the CPU time. Also, it is clear from Figures above that the blowfish algorithm outperforms the other algorithms in all tests. Also, the key generation time is almost negligible compared to encryption/decryption time. From Figures for Linux, it is clear that the execution time is shorter for all algorithms than that for WinXP.

5. Conclusions and Future work

In this research we presented performance evaluation using Java implementation of four symmetric block ciphers: DES, Triple-DES, AES, and Blowfish.

Java programming language, JCA and JCE were used as an implementation tools. Performance evaluation based on their CPU execution time is presented under WinXP and Linux operating system platforms.

From the experimental results shown in figures one can derive the following:

- Blowfish algorithm is the fastest among the four algorithms for both encryption and decryption and outperform the other three algorithms in all tests and under WinXP and Linux.
- The key generation time in all algorithms is almost negligible compared to encryption/decryption time.
- The CPU execution time under Linux platform is shorter for the four algorithms than that under WinXP.
- Triple-DES algorithm is the best in terms of strength, but its encryption/decryption times are the longest among all algorithms for all cases.

In future, we would like to experiment with these and new asymmetric algorithms under different parameters and setups for better characterization and performance evaluation.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Yarmouk University, Irbid-Jordan for supporting the publication of this research work. This research was completed while I was on sabbatical leave from Yarmouk University. Also, I would like to express my gratitude to the Hashemite University – Jordan, where I spent my sabbatical leave.

References

Anand Kumar M., and Dr. S. Karthikeyan. (2012). Investigating the Efficiency of Blowfish and Rejindael (AES) Algorithms. I. J. Computer Network and Information Security, V4, N2, pp. 22-28.

Bouncycastle. http://bouncycastle.org/ (Accessed on 20 January 2013).

CDC Informatic. http://www.cdc.informatik.tu-darmstadt.de/ (Accessed 31 January 2013).

CGI Group Inc. White paper, Public Key Encryption and Digital Signature: How do they work? http://www.cgi.com/files/white-papers/cgi_whpr_35_pki_e.pdf. (Accessed 2 January 2013).

Douglas, R. Stinson. (2005). Cryptography Theory and Practice. (3rd Edition). Chapman & Hall/CRC.

Eashwar Thiagarajan and Madhuri Gourishetty. (2003). Study of AES and its Efficient Software Implementation. http://www.cs.ucsb.edu/~koc/cs290g/project/2003/thiagarajan-gourishetty.pdf. (Accessed 10 April 2013).

Elminaam D S Abd; Kader H M Abdual and Hadhoud, M Mohamed. (2010). Evaluating the Performance of Sysmmetric Encryption Algorithms. International Journal of Network Security, Vol. 10, No. 3, pp. 216-222, May (2010).

En Wikipedia Advanced_Encryption_Standard. http://en.wikipedia.org/wiki/Advanced_Encryption_Standard. (Accessed 8 January 2013)

En Wikipedia Cryptography. http://en.wikipedia.org/wiki/Cryptography/ (Accessed 31 January 2013).

En Wikipedia Data_Encryption_Standard. http://en.wikipedia.org/wiki/Data_Encryption_Standard (Accessed 15 January 2013).

En Wikipedia Java_cryptography. http://en.wikipedia.org/wiki/Java_cryptography/ (Accessed 20 January 2013).

En Wikipedia Triple_DES. http://en.wikipedia.org/wiki/Triple_DES. (Accessed 8 January 2013).

Flanagan D. (1999). *Java in Nutshell*. (3rd Edition). O'REILY publishing.

Guido Bertoni, Luca Breveglieri, Pasqualina Fragneto, Marco Macchetti, and Stefano Marchesin uido Bertoni1. (2002). Efficient Software Implementation of AES on 32-Bit Platforms. in *Cryptographic Hardware and Embedded Systems* - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, pp. 159-171, August 13-15, (2002).

Hamdan.O.Alanazi, B.B.Zaidan, A.A.Zaidan, Hamid A.Jalab, M.Shabbir and Y. Al-Nabhan. (2010). New Comparative Study Between DES, 3DES and AES within Nine Factors. *JOURNAL OF COMPUTING*, Vol. 2, ISSUE 3, pp. 152-157.

Jamie Jaworski, Paul J. Perrone, and Venkata S. R. R. Chaganti (2000). *Java Security Handbook*. Sams Publishing.

Java Sun. http://java.sun.com/products/jdk/1.2/docs/guide/security/CryptoSpec.html/ (Accessed 28 January 2013).

Java Sun j2se. http://java.sun.com/j2se/ (Accessed 28 January 2013).

Jawahar Thakur, Nagesh Kumar. (2011). DES, AES and Blowfish: Symmetric Key Cryptography Algorithms Simulation Based Performance Analysis. *International Journal of Emerging Technology and Advanced Engineering*. Volume 1, Issue 2, pp. 6-12. http://www.ijetae.com/files/Issue2/IJETAE_1211_02.pdf.

Kofahi, N. A. (2006). Performance Evaluation of AES/Triple-DES/Blowfish Ciphers under W2K and Linux operating system platforms. *ABHATH ALYARMOK, Basic Sci. & Eng*, vol. 15, no. 2, pp. 265-285.

Larry Koved,Marco Pistoia and Aaron Kershenbaum. (2001). Understanding Java™ 2 Platform Security Permissions Practical Approach.. *Sun's 2001 world wide java developer conference*.

Marco Pistoia, Duane F. Reller Deepak Gupta, Milind Nagnur, Ashok K. Ramani Java 2 Network Security. http://www.redbooks.ibm.com/redbooks/SG242109.html. (Accessed 8/1/2013).

Marco Pistoia, Pistoia, Deepak Gupta and Ashok Ramani. (1999). *Java 2 Network Security*. Prentice Hall.

Omar M.Barukab, Asif Irshad Khan, Mahaboob Sharief Shaik , MV Ramana Murthy. (2012). Secure Communication using Symmetric and Asymmetric Cryptographic Techniques. *I. J. Information Engineering and Electronic Business*. Vol. 4, No. 2, pp. 36-42. http://www.mecs-press.org/ijieeb/ijieeb-v4-n2/IJIEEB-V4-N2-6.pdf

Singh, S Preet and Maini, Raman. (2011). Comparison of Data Encryption Algorithms. *International Journal of Computer Science and Communication*, vol. 2, No. 1, January-June, 2011, pp. 125-127.

William Stallings, (2009). Cryptography and Network Security Principles and Practice. (8th Edition, Prentice Hall, 8th Edition.

Najib A. Kofahi is a professor of computer science at Yarmouk University (YU). He received his Ph.D. in computer science from the University of Missouri-Rolla (USA) in 1987. He was granted scholarship for MSc. and Ph.D. from YU in 1981. Currently he is on sabbatical leave from Yarmouk University and he is a faculty member in the department of computer sciences at YU – Jordan since 1987.
During his stay at YU he was the dean of the Faculty of Information Technology and Computer Sciences. He also worked as the chairman of the computer science department during the years 1990-1992. During his stay at yarmouk he worked on the computer science curriculum development since his appointment. Also, he worked extensively in the curriculum development at Philadelphia University-Jordan in the academic year 1993-1994 during which he worked as the chairman of science department, while on sabbatical leave from YU. In September 2000 he went on leave from YU for four years during which he worked as a visiting associate professor at King Fahd University of Petroleum and Minerals (FUPM). While at KFUPM, he led an online course development team to develop online course material for algorithms course. He has several journal and conference research publications in a number of research areas. His research interest focuses on E-Learning, operating systems, distributed systems, performance evaluation and computer and data security.

This academic article was published by The International Institute for Science, Technology and Education (IISTE). The IISTE is a pioneer in the Open Access Publishing service based in the U.S. and Europe. The aim of the institute is Accelerating Global Knowledge Sharing.

More information about the publisher can be found in the IISTE's homepage: http://www.iiste.org

CALL FOR PAPERS

The IISTE is currently hosting more than 30 peer-reviewed academic journals and collaborating with academic institutions around the world. There's no deadline for submission. **Prospective authors of IISTE journals can find the submission instruction on the following page:** http://www.iiste.org/Journals/

The IISTE editorial team promises to the review and publish all the qualified submissions in a **fast** manner. All the journals articles are available online to the readers all over the world without financial, legal, or technical barriers other than those inseparable from gaining access to the internet itself. Printed version of the journals is also available upon request of readers and authors.

IISTE Knowledge Sharing Partners

EBSCO, Index Copernicus, Ulrich's Periodicals Directory, JournalTOCS, PKP Open Archives Harvester, Bielefeld Academic Search Engine, Elektronische Zeitschriftenbibliothek EZB, Open J-Gate, OCLC WorldCat, Universe Digtial Library , NewJour, Google Scholar