Performance Analysis of Encryption Algorithms' Text Length Size on Web Browsers

Syed Zulkarnain Syed Idrus[1], Syed Alwee Aljunid[2], Salina Mohd Asi[3], Suhizaz Sudin[4], and R. Badlishah Ahmad[5]

[1-5]School of Computer and Communication Engineering, Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia

Summary

Information can easily be acquired over the Internet, as we know it today. Some are meant to be visible, while others are considered to be sensitive and meant to be secretive. This however, brings about the basic problem of susceptibility to abuse if mistreated. Thus, the proprietor would want sensitive data to remain sensitive and confidential. In order to do this, the data has to be secured.

In this study, the security measure that has been selected is encryption. We have proposed a Web programming language to be analyzed with four Web browsers in term of their performances to process the encryption of the programming language's script with the Web browsers.

Firstly, we introduce one of the encryption techniques that can be applied, which is the eXclusive OR (XOR) operation, and how the text data are converted into hexadecimals. This is followed by conducting tests simulation in order to obtain the best encryption algorithm versus Web browser. The results of the analysis are presented in the form of graphs. We finally conclude on the findings that different algorithms perform differently to different Web browsers. Hence, we now determine which algorithm works best and most compatible with which Web browser.

Key words:
Data Security, Encryption Method, Encryption Algorithms, Web Browsers, ASP

1. Introduction

There are various security measures that can be imposed in order to secure the information stored. As more and more technologies evolve, an irresponsible person may try to find a way to excavate any loopholes within the system in order to penetrate into the heart of its weaknesses. This is due to the fact that human-made designs can also be broken by another human. Thus, over time security measures must constantly be reviewed and strengthened in order to combat hackers or culprits hot on the heels of system developers who are also using high technologies.

One of the means to secure the data is to apply a secret code of encryption. By having it encrypted, the sender can pass the data to the receiver and only the receiver or authorized personnel can have access to the data provided they have been given a key by the sender to decrypt it in order for them to view the information. On the other hand, without having the right key, nobody is able to read the data received or stored. Even if hackers or unauthorized person managed to intercept or steal the data, it would be futile because the text looks ridiculous to them.

Encryption consists of various types known as algorithms and they have been developed or written by different people. Since many people developed them, there are pros and cons that we need to consider. Further more, the language of the algorithms can also be developed or written in many forms i.e. in different programming languages.

2. Conceptual Framework

In this study, we have proposed only one Web programming language script to be analyzed with four Web browsers in order to determine which type of algorithm is suitable to which type of Web browser in terms of their performance and compatibility.

Active Server Pages (ASP), has been selected and five different types of encryption algorithms have been chosen to be analyzed to observe their performance. The encryption algorithms selected are Blowfish, International Data Encryption Algorithm (IDEA), Advanced Encryption Standard (AES), Tiny Encryption Algorithm (TEA) and Twofish. These encryption algorithms are known to be able to support 128-bit key size [1]. Furthermore, the five types will be co-analyzed with the four selected Web

Manuscript received January 5, 2008.
Manuscript revised January 20, 2008.

browsers that are able to process its scripts effectively and in an efficient manner.

There are quite a number of Web browsers that are available in the market, but these four are known to be among the top and most popular. They are Internet Explorer, Mozilla Firefox, Opera and Netscape Navigator [2]. From analysis, we hope to find out the most impeccable Web browsers that can match in the best possible way with the encryption algorithms for ASP scripts.

3. Methodology

Before implementing an encryption algorithm, we need to understand the principle behind the encryption i.e. to secure data held within a message or file and to ensure that the data is unreadable to others. The unencrypted message or file is often referred to as *Plaintext*, and the encrypted message or file is referred as *Ciphertext*. In encryption, it consists of key length in number of bits. A key is a long sequence of bits used by encryption algorithms. Thus, the length of the key determines the probabilities if one ought to figure it out all its possible key values.

The commencement of the encryption process begins after the authorization to use the system has been obtained, only then that the information inputted be submitted. In order not to be intercepted by culprits along the way, the text must first be encrypted prior to storage using the encryption secret codes along with its key known only to the sender and the receiver. For the receiver to be able to read it, the data has to be decrypted simply by reversing the process using the given key.

4. Encryption Technique

The simplest method of encryption is by considering a text contain in a single line of text. It is shown as follows: -

UniMAP

Based from the ASCII Table and Description [3], the above text consists of six characters i.e. UniMAP would be stored as the following series of bytes that have been converted into hexadecimals: -

U	n	i	M	A	P	Character
55	6E	69	4D	41	50	Hexadecimal

Let us say, we would want to encrypt the text *"UniMAP"* using the following key (or password): -

Perlis

This key (or password) also consists of six characters i.e. Perlis would actually be stored in the following series of bytes, which also have been converted into hexadecimals: -

P	e	r	l	i	s	Character
50	65	72	6C	69	73	Hexadecimal

We could encrypt the text *"UniMAP"* by applying say, an *XOR* function. If both data (Input A) and key (Input B) individual bits are the same, then the output (Output X) of the function is a zero and vice versa. The *truth* table adopted from [4] is shown in Table 1.

Table 1: XOR Function in Truth Table

Input A	Input B	Output X
0	0	0
0	1	1
1	0	1
1	1	0

The *truth* table can also be written as *x XOR y* (also written as $x + y$, $x \oplus y$, or $x \neq y$) with the following formula [5]: -

True (T)	≠	True (T)	=	False (F)
True (T)	≠	False (F)	=	True (T)
False (F)	≠	True (T)	=	True (T)
False (F)	≠	False (F)	=	False (F)

Hence, $x \neq y$ using the formula above is as follows [5]: -

x	*y*	≠
T	T	F
T	F	T
F	T	T
F	F	F

Now, we can use the *XOR* function to encrypt the text *"UniMAP"*, thus using the key (or password) *"Perlis"*, where *U* encrypt using *P*; *n* encrypt using *e*; *i* encrypt using *r*; *M* encrypt using *l*; *A* encrypt using *I*; and *P* encrypt using *s*. It is shown below: -

U	n	i	M	A	P
🔒	🔒	🔒	🔒	🔒	🔒
P	e	r	l	i	s

The output of the encryption and its key for characters *U* and *P* respectively is shown in Table 2.

Table 2: Encrypt *U* Using *P*-Key

Char	Hx	Dec		128	64	32	16	8	4	2	1		Dec	Hx
U	55	85	≡	0	1	0	1	0	1	0	1			
P	50	80	≡	0	1	0	1	0	0	0	0			
				0	0	0	0	0	1	0	1	≡	5	**5**

The output of the encryption and its key for characters *n* and *e* respectively is shown in Table 3.

Table 3: Encrypt *n* Using *e*-Key

Char	Hx	Dec		128	64	32	16	8	4	2	1		Dec	Hx
N	6E	110	≡	0	1	1	0	1	1	1	0			
e	65	101	≡	0	1	1	0	0	1	0	1			
				0	0	0	0	1	0	1	1	≡	11	**B**

The output of the encryption and its key for characters *i* and *r* respectively is shown in Table 4.

Table 4: Encrypt *i* Using *r*-Key

Char	Hx	Dec		128	64	32	16	8	4	2	1		Dec	Hx
i	69	105	≡	0	1	1	0	1	0	0	1			
r	72	114	≡	0	1	1	1	0	0	1	0			
				0	0	0	1	1	0	1	1	≡	27	**1B**

The output of the encryption and its key for characters *M* and *l* respectively is shown in Table 5.

Table 5: Encrypt *M* Using *l*-Key

Char	Hx	Dec		128	64	32	16	8	4	2	1		Dec	Hx
M	4D	77	≡	0	1	0	0	1	1	0	1			
l	6C	108	≡	0	1	1	0	1	1	0	0			
				0	0	1	0	0	0	0	1	≡	33	**21**

The output of the encryption and its key for characters *A* and *i* respectively is shown in Table 6.

Table 6: Encrypt *A* Using *i*-Key

Char	Hx	Dec		128	64	32	16	8	4	2	1		Dec	Hx
A	41	65	≡	0	1	0	0	0	0	0	1			
i	69	105	≡	0	1	1	0	1	0	0	1			
				0	0	1	0	1	0	0	0	≡	40	28

The output of the encryption and its key for characters *P* and *s* respectively is shown in Table 7.

Table 7: Encrypt *P* Using *s*-Key

Char	Hx	Dec		128	64	32	16	8	4	2	1		Dec	Hx
P	50	80	≡	0	1	0	1	0	0	0	0			
s	73	115	≡	0	1	1	1	0	0	1	1			
				0	0	1	0	0	0	1	1	≡	35	23

Therefore, our encrypted text would result with the following set of numbers compose of Table 2 – Table 7 in boxes under the column Hx (Hexadecimal): -

5	B	1B	21	28	23

5. Performance Analysis

In order to verify which of the five encryption algorithms perform better to the four Web browsers mentioned earlier, a test have been conducted using two computers that have been setup and dedicated as Client and Server via a router. Encryption testing is to test the performance of five encryption algorithms in encrypting a set of text and key via Web browsers for ASP scripts. Thus, the text length starting at 10 will be increase four times its initial characters, whereas the key length for each text length remains unchanged.

6. Results

The outcome of the testing will project the response time i.e. the encryption process and the time taken of the four Web browsers namely Internet Explorer, Mozilla Firefox, Opera and Netscape Navigator after performing the encrypting scripts timed in *millisecond* onto the computer screen. Fig. 1 to Fig. 4 were the test results after having increased the text length for each encryption algorithms for the four Web browsers by 10 characters, where it had been observed and noted of their performance results.

Fig. 1 illustrates the result of Internet Explorer and its Text Length versus Response Time. From the analysis, Twofish performs better compared to others and sustain almost lower response time.

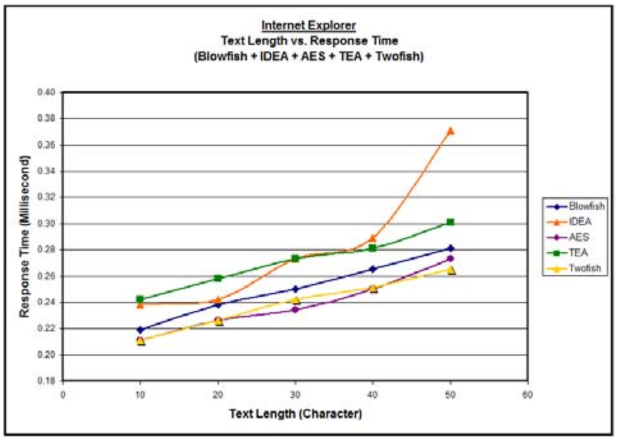

Fig. 1 Internet Explorer's Text Length vs. Response Time

Fig. 2 illustrates the result of Mozilla Firefox and its Text Length versus Response Time. From the analysis, Twofish yet again performs better compared to others and just about sustaining lower response time. It does however perform less at 20 and 40 Text Length with a couple of algorithm namely Blowfish and AES.

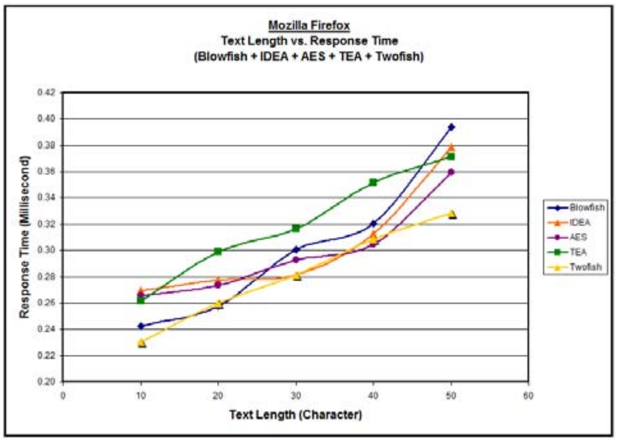

Fig. 2 Mozilla Firefox's Text Length vs. Response Time

Fig. 3 illustrates the result of Opera and its Text Length versus Response Time. From the analysis, IDEA performs slightly less than Blowfish at the start. But nonetheless, it performs better for the remaining text lengths compared to others in its response time.

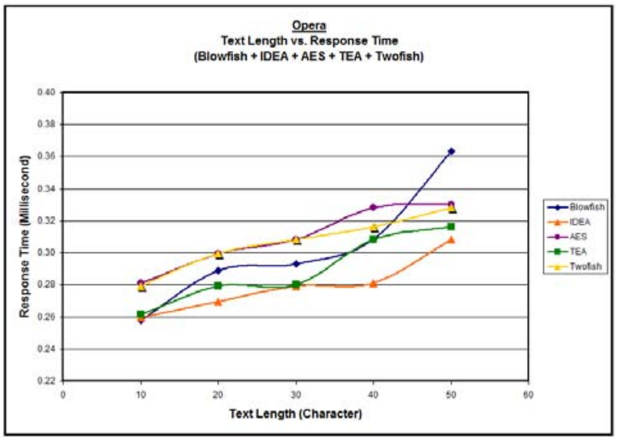

Fig. 3 Opera's Text Length vs. Response Time

Fig. 4 illustrates the result of Netscape Navigator and its Text Length versus Response Time. From the analysis, TEA had a good start and performs better compared to others up until 30 Text Length. Unfortunately, it failed to sustain its lower response time, whereby AES and IDEA had outperform TEA in the last two text lengths.

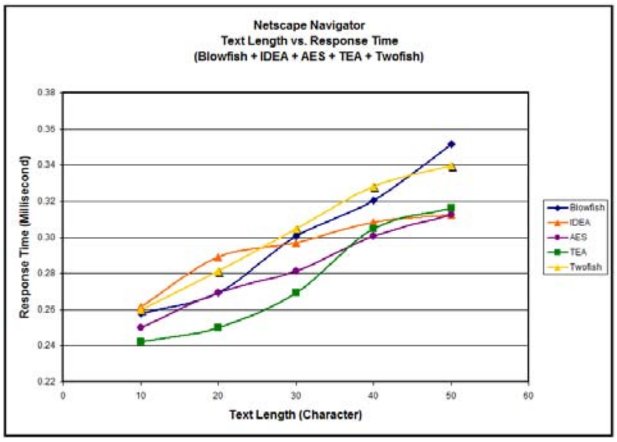

Fig. 4 Netscape Navigator's Text Length vs. Response Time

7. Conclusions

In an actual observation, the response time sometimes fluctuates when we ought to run the test twice with an encryption algorithm on the same Web browser using the same text length. This could be due to the network traffic or even the heavy usage of the Server. But in this case, there is only one Client and a Server, hence there should not be any traffic at all as only one Client accessing the Server. Thus, we can safely conclude that it must been caused by the time it takes for the Server to process the ASP script of an algorithm on the Web browser, along with many other processes running at the same time within the Server. This can cause the Central Processing Unit (CPU) usage amounting high, hence slows down the encryption process.

Therefore, apart from the network conditions that we are aware of from using Local Area Network (LAN), Wide Area Network (WAN) and Internet, Server also plays an important role for better performance.

From our findings, we came to the conclusion that for a one-time run simulation test of an algorithm that performs best on Web browser are as follows: -

(i) Internet Explorer Web browser suited for Twofish encryption algorithms.
(ii) Mozilla Firefox Web browser suited for Twofish encryption algorithms.
(iii) Opera Web browser suited for IDEA encryption algorithms.
(iv) Netscape Navigator Web browser suited for TEA encryption algorithms.

Acknowledgment

The gratitude goes to The Ministry of Science, Technology and Innovation (MOSTI), Malaysia whom had supported this work for the duration of this study. The authors would also like to express their cordial thanks to the School of Computer and Communication Engineering, Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia for the support and had made this paper possible for publication.

References

[1] Wikipedia Contributors. (2007). … (cipher). Wikimedia Foundation, Inc. Retrieved May 19, 2007, from http://en.wikipedia.org/wiki/…_(cipher).

[2] Wikipedia Contributors. (2007). Web browser. Wikimedia Foundation, Inc. Retrieved May 23, 2007, from http://en.wikipedia.org/wiki/Web_browser.

[3] LookupTables Team. ASCII Table and Description. www.LookupTables.com. Retrieved May 4, 2007, from http://www.asciitable.com/.

[4] Gandalf Team. (1996-2005). Encryption: Basic Concepts. Gandalf. Retrieved April 26, 2007, from http://www.exegesis.uklinux.net/gandalf/encrypt/basic.htm.

[5] Wikipedia Contributors. (2007). Truth table. Wikimedia Foundation, Inc. Retrieved May 28, 2007, from http://en.wikipedia.org/wiki/Truth_table.

Syed Zulkarnain Syed Idrus received the B.Sc. degree in Information Systems Engineering, from University of Manchester Institute of Science and Technology (UMIST), Manchester, United Kingdom in 2001. He started his career as an IT Support Executive cum Trainer (from 2002) in Cosmopoint College of Technology Penang, Malaysia and Information Systems Officer (from 2005) at the Universiti Sains Malaysia (USM), Penang, Malaysia. He is currently pursuing a M.Sc. degree in Computer Engineering at University Malaysia Perlis (UniMAP), Perlis, Malaysia. His research interest includes information systems, systems development, systems and network security, information security (text, sound and image encryption) and their applications to quantum cryptography. He was also a member of The British Computer Society (BCS) in 2001.

Syed Alwee Aljunid Syed Junid received the B.Eng. in Computer and Communication System (First Class Honors) and PhD in Communication and Network Engineering from University Putra Malaysia (UPM), Malaysia in 2001 and 2005, respectively. He is currently an Associate Professor and Deputy Dean (Academic and Research) of the School of Computer and Communication Engineering, Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia.

Salina Mohd Asi completed her B.CompSc. degree program, at Universiti Teknologi Malaysia (UTM), Kuala Lumpur, Malaysia in 1995. After working for about a year, she pursued her study in Master of Science (Real Time Software Engineering) at UTM. After she had completed her Master's degree, she worked in industry for 6 years before joining the Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia as a lecturer. Her research interest includes artificial intelligence in embedded system, artificial neural network, image processing, bioinformatic (protein analysis) and parallel computing.

Suhizaz Sudin received the B.IT. (Hons.) degree, from University Utara Malaysia (UUM), Malaysia in 1998. After his graduation, he had been working with several organizations before he pursued a Master of Science in Computer Science from University Putra Malaysia (UPM), Malaysia. He then joined Legenda Group of Colleges as a lecturer until 2004. Later, he became a lecturer and also had been appointed as a Deputy Director of Centre for Industrial Collaboration in Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia. His research interest includes computer networks (network security, network modeling, network performance study), ubiquitous computing, distributed, parallel and GRID computing and information systems. Currently, he is pursuing a PhD, researching on Network Performance at Massey University, New Zealand.

R. Badlishah Ahmad obtained a B.Eng in Electrical & Electronic Engineering from Glasgow University, United Kingdom in 1994. He continued his M.Sc study in Optical Electronic Engineering and a PhD at University of Strathclyde, United Kingdom and graduated in 1995 and 2000 respectively. He is currently the Dean of the School of Computer and Communication Engineering and also the Head of Embedded Computing Research Cluster in Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia. His research interest includes computer network modelling and embedded system based on GNU/Linux.