# Improving the Secure Socket Layer Protocol by modifying its Authentication function

H. Otrok, PhD student, ECE Department, Concordia University,
Montreal, QC, Canada. E-mail: h_otrok@ece.concordia.ca
R. Haraty, Assistant Dean, School of Arts and Sciences, Lebanese American University,
Beirut, Lebanon. E-mail: rharaty@lau.edu.lb
A. N. El-Kassar, Full Professor, Mathematics Department, Beirut Arab University,
Beirut, Lebanon. E-mail: ak1@bau.edu.lb

**Abstract**

Secure Socket Layer (SSL) is a cryptographic protocol widely used to make a secure connection to a web server. SSL uses three interdependent cryptographic functions to perform a secure connection. The first function is authentication. It is used to allow the client to identify the server and optionally allow the server to identify the client. The most common cryptographic algorithm used for this function is RSA. If we double the key length in RSA to have more secure communication, then it is known that the time needed for the encryption and decryption will be increased approximately eight times. In this paper, we propose a modification of RSA from the domain of integers to the domain of Gaussian arithmetic to be applied to the first function of SSL that would give more secure communication. This modification would use only double the time needed for the usual implementation of RSA with key size of 1024 bits.

## I. Introduction

$Secure\ Socket\ Layer$ (SSL) is a protocol used to make secure communication between a client and a server [5]. Both the Netscape and Internet Explorer support versions of SSL and the Internet Engineering Task Force (IETF) has approved SSL as a standard. SSL is located between the TCP and HTTP protocols [4]. In this case, HTTP is modified to be HTTP(Secure) abbreviated by HTTPS, which is the standard encrypted communication mechanism on the World Wide Web. The SSL protocol is designed using three interdependent cryptographic functions [5]. Authentication is the first function found in SSL. Its goal is to perform identification and authentication of the parties involved in the communication. Authentication is achieved using public key encryption and a digital certificate issued by a trusted Certificate Authority [8]. There are many public key cryptographic algorithms that could be used to achieve authentication such as RSA, Diffie Hellman and ElGamal. RSA is the most common cryptographic algorithm used to achieve authentication. RSA keys are classified into three categories [1]:

1) Short Key whose range is less than 900 bits.
2) Medium key whose range is between 900 and 1250 bits.
3) Long Key which is greater than 1250 bits.

At present the use of 512 bits keys are considered as unsecured keys after the successful attack by 200 PCs networked together working 16 hours a day and implementing the $General\ Number\ Field\ Sieve$ ($GNFS$) algorithm [1]. The $GNFS$ algorithm is used to factorize $n$, where $n$ is the multiplication of two large prime numbers $p$ and $q$ [1]. It is computed by distributing the result over large number of computers. To factor a 1024 bits number will be between $1,000$ and 10,000 times harder than factoring a 512 bits number, just for the distributed part of the algorithm, and then the collected results will probably run to between $10,000$ and $100,000$ times as much as before [1].

The idea behind this paper is to modify the RSA key from 1024 bits to 2048 bits by applying Gaussian integers instead of ordinary integers [2] using the same prime numbers used by the 1024 bits. In this way we are making SSL more secure by using 2048 bits [7] and 512 bits for prime numbers. Confidentiality is the second function used in SSL. Its goal is to keep the communications confidential. This is performed using symmetric encryption scheme. Symmetric encryption means that both parties use the same key. Usually, symmetric encryption is used to exchange the messages between the client and the server and not asymmetric encryption (Public Key) because symmetric scheme is 1000 times faster than asymmetric scheme. Asymmetric scheme is used only to exchange the symmetric keys between both parties. For this reason, using secure public key is a necessity in order to secure our symmetric keys that are later used for encrypting the messages. The $RC4$ is the symmetric cipher used to encrypt the exchanged messages. Integrity is the last function used in SSL. Its job is to ensure the integrity of the data against interfering. This is performed using message digests. The $MD5$ and $SHA-1$ algorithms are used to compute a complex function based on two things first the message that was sent and second the secret keys known to both parties. The receiver computes the same function on the data that arrives. If the values computed at both parties matches then message integrity was checked indicating positive result.

If an attacker knows the RSA key then the attacker can do any of the following [6]:

1) Can destroy the trust between the client and server.
2) Offer invalid information.
3) Pretend the role of the server and in this case he can receive either personal or financial information from the user.

Also, besides authentication, the use of RSA in SSL ensures the confidentiality of the data. So, in this case the exchanged messages will be known. In order to prevent such types of attacks, it is usually recommended to use a larger size of RSA keys [7]. Currently, the key size that is considered to be secure is of length 2048 bits.

This paper is organized as follows: Section 2 describes a background of SSL. Section 3 describes the RSA used in the classical SSL, which depends on integer arithmetic. Section 4 describes the Modified RSA, which depends on Gaussian integer that will be used in the modified SSL. Section 5 describes the experimental results. A conclusion is drawn in Section 6.

## II. Background

SSL was designed by $Netscape$ to perform a secure communication between a client and a server [2]. SSL uses three interdependent cryptographic algorithms. The first function is $authentication$. It is used to allow the client to identify the server and optionally allow the server to identify the client. SSL uses digital certificates to authenticate servers. The most common cryptographic algorithm used in this phase is the RSA algorithm. The second function is $confidentiality$ that is used to keep the communication confidential. It uses symmetric cryptography to exchange messages confidentially. $Integrity$ is the last function used to ensure the integrity of the data against interfering. This is performed using message digests. Checksum of the message is used for message digest. SSL uses three protocols to implement the above three algorithms $Handshake$ *protocol*, *Record protocol*, and the *Alert protocol*. Handshake protocol is used to let a client authenticates a server. The Handshake protocol is shown in $Table$ 1 [1]. After the handshake is complete the Record protocol takes place. The exchange of the encrypted messages is handled in the Record protocol. If one of both parties finds an error, it sends an alert containing the error. This is handled in the Alert protocol. Change-Cipher-Spec message used in step 8 in Table 1 is used to specify the following [10]:

- The bulk data encryption algorithm such as DES.
- The hash algorithm used for MAC calculation such as MD5 or SHA-1.
- It also defines cryptographic attributes such as the hash size.

Table. 1

1) Client sends "$Hello - Server$" message.
2) Server acknowledges with "$Hello - Client$" message.
3) Server sends its certificate.
4) Optional: Server requests Client's certificate.
5) Optional: Client sends its certificate.
6) Client sends "$Client - key - Exchange$" message.
7) Client sends Certificate Verify message.
8) Both Client and Server "$Change - Cipher - Spec$" message.
9) Both Client and Server send "$Finished$" messages.

## III. Classical RSA used in SSL

As we mentioned in Section 1, the authentication in SSL is done using RSA. The classical value used for RSA Key was 512 bits [3]. Then, a modified version of SSL was published using 1024 bits which is considered to be more secure but it is currently recommended to use 2048 bits key for better secure communication [7]. The RSA used in SSL depends on the integer arithmetic. In order to generate a key with size 1024 bits we need two distinct large primes each with 512 bits size. 512 bits is equivalent to 155 decimal digit and 1024 bits is equivalent to 309 decimal digit. While the 2048 bits key is equivalent to 617 decimal digits.

The classical RSA Algorithm used for authentication is as follows:

1) Find two large primes $p$ and $q$ and compute their product $n = p \times q$.
2) Find an integer d that is co-prime to $\Phi\ (n) = (p-1)(q-1)$.($\Phi$ (n) denotes the Euler Phi function).
3) Compute $e$ from $e\ d \equiv 1 mod (p-1).(q-1)$.
4) Broadcast the public key, that is, the pair of numbers $(e, n)$.
5) Represent the message to be transmitted, $m$, say as a sequence of integers $\{m\}$ each in the range 1 to $n$.
6) Encrypt each message, $m$, using the public key by applying the rule $c = m^e (mod\ n)$.

7) The receiver decrypts the message using the rule $m = c^d (mod\ n)$.

In order to generate the public-key, entity $A$ selects two large primes
$p$= 44127800978823464012065719098179126374973898914209562046988441981248518969110303221293358949658122748493278114403281377231038484764264028329274512672488483
and
$q$ = 87991403152545640995703032622624305655160180091639203111472417516472670337466780894540980230220202377853499430986227955778392051812841853877814940672190319
Then, $A$ computes the product $n = p \times q$=
3882867126162953573787108573972341271025993245208934559324686536509560395815432643068226155108318571993854693394662687125829047856240140535160011825507903034749121688262228113716949782394273137622516743022648238234787587194209102473390361327930555694003279809158940828442493529613837230888788067225595911596077
Then, $A$ chooses the encryption exponent $e$ =
1695892032992212889389428894665136705415067004871073511310367457154757740054615570393003058578343159088694625693579162510012706730266884826779468652475263976374197897744469565648425761741651955357068152987782221142877140446191676380884354604523964065046357978522084931277589484871473785367218263536852321714271
also, $A$ uses the extended Euclidean algorithm for integers to find the decryption exponent
$d$ =
1436485276569741791302590744728293848722784377591127102199982283543280354888206914392709316981206231819157390317670939754939744441315445353099582094355844290387278964755948071689877914487327501017869803247346959135569241605715606963773192226802473089538483993962657378720959576480719947139738985688458818414279

So that $e.d \equiv 1.(mod \Phi(n))$.
Now, the public-key is the pair: $(n, e)$.
While A's private-key is the pair: $(n, d)$
For example, to encrypt the message $m$ = "Hello Server" needs an average time of $0.5953$ seconds, using Pentium IV 2.4 GHZ with 256 Memory. Now, entity $B$ computes the cipher-text $c$ and sends it to $A$.
To decrypt the sent cipher-text $c$, entity $A$ should compute $m$ which is the original message. This needs an average time of $0.8674$ seconds in order to recover the message.

## IV. Modifying the Authentication function using the Modified RSA

In this section, we will briefly present the background needed by the modified RSA and the modified version of RSA in the domain on Gaussian integers.

### A. Background

We first give a brief review of Gaussian integers. Arithmetic in the domain of Gaussian integers [2], $\mathbb{Z}_n$ [i], is the set of all elements in the form of $\{a + ib$ where $a, b \in \mathbb{Z}n\}$ where n is a prime integer in the form of $4k + 3$ where $k$ is any integer. In this case the order of the set is $\Phi(n) = n^2$. So, $\mathbb{Z}_n^*$[i] = $\{\mathbb{Z}_n$ $[i] - \{0\}\}$ in this case the order of $\mathbb{Z}_n[i]$ is $\Phi(n) = n^2 - 1$.
Take a prime integer $n = 3$ in the form $4k + 3$ where $k = 0$. Here,
$\mathbb{Z}_3[i] = \{a + ib$ where $a, b \in \mathbb{Z}_3\} = \{0, 1, 2, i, 1 + i, 2 + i, 2i, 2i + 1, 2i + 2\}$
The order of the set $\mathbb{Z}_3[i]$ is $\Phi(3) = 3^2 = 9$. Here,
$\mathbb{Z}_3^*[i] = \{\mathbb{Z}_3[i] - \{0\}\} = \{1, 2, i, 1 + i, 2 + i, 2i, 2i + 1, 2i + 2\}$
The order of the set $\mathbb{Z}_3^*[i]$ is $\Phi(3) = 3^2 - 1 = 9 - 1 = 8$

### B. Modified RSA

The RSA algorithm has been modified from the domain of integers to the domain of Gaussian integers. It was proved that this modification is reliable and more secure than the classical RSA [3]. The modified algorithm is as follows:

1) Find two large primes $p$ and $q$ and compute their product $n = p \times q$.
2) Find an integer $d$ that is co-prime to $\Phi(n) = (p^2 - 1).(q^2 - 1)$.
3) Compute $e$ from $e.d = 1. \mod (p^2 - 1).(q^2 - 1)$.
4) Broadcast the public key, that is, the pair of numbers $(e, n)$.
5) Represent the message to be transmitted, $m$, say as a sequence of integers $\{m\}$ each in the range $1\ to\ n$.
6) Encrypt each message, $m$, using the public key by applying the rule $c = m^e (mod\ n)$.

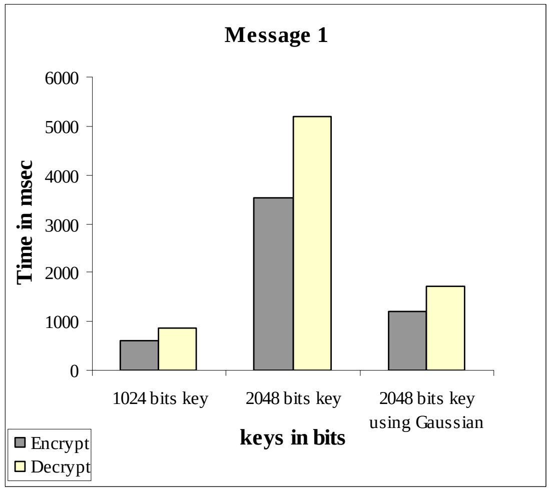


Fig. 1. Message 1

7) The receiver decrypts the message using the rule $m = c^d (mod\ n)$.

In order to generate the public-key, entity $A$ selects two large primes $p$ and $q$ with the same values as shown in the example 2.1.

Then computes the product $n = p.q$ which is the same value as in example 1.

Then, $A$ chooses the encryption exponent $e$ =
13143511024003378934804029482945817342435107097711451422752828830067982562122427
37098072908144958099708496864308980670956420766044142694492978803652290804927001
06895421199568667394707462588680459304065700227144555715807316725164762349260264
49953943259668598815281724359511008403772603187945569240204442986741892140250483
34313753363412573445555801882439726051339142297456225362028346588228771609786746
63374349504427049786413674845802510544249985782136687791003362847679899078421651
73235590323367855711391114571271107442238264205162434342265345274972470721174686
600464232520568933883297217542999809829276760079500932622583

also, $A$ uses the extended Euclidean algorithm for integers to find the decryption exponent
$d$ = 40067003768300115941623495059951529038600437508541907346588688715083064670317
85657425607595446246967730709367964424416572378048478990722147495181583206149621
66626038798049750332796521824254994482976830022755174863351147986960000183445222
08471411119443643610813862120736017867140702285832157990757443987637961004156407
42182574783456513322856065555426867232069600964347518011928104863503273026367541
34376292250395574914458421952994373172392941080809210033823687172001494356375971
82613326500420326565484822182070053666372917298227049038673832421288070087678020
63037018255553586606209104758699906829135051770575260938331847

So that $e.d \equiv 1 (mod\ \Phi(n))$.

Now, the public-key is the pair: $(n, e)$

While $A$'s private-key is the pair: $(n, d)$

For example, to encrypt the message $m =$ "Hello Server" needs an average time of $1.1985$ seconds, using Pentium IV 2.4 GHZ with 256 Memory. Now, entity $B$ computes the cipher-text $c$ and sends it to $A$.

To decrypt the sent cipher-text $c$, entity $A$ should compute $m$ which is the original message. This needs an average time of $1.736$ seconds in order to recover the message.

## V. Experimental Results

In this section, we compare and evaluate the classical and the modified authentication functions of SSL by showing the run time results of three different examples:

1) $1024$ bits key generated using two prime numbers each with $512$ bits.
2) $2048$ bits key generated using two prime numbers each with $1024$ bits.
3) $2048$ bits key generated using two prime numbers each with $512$ bits.(Gaussian Integer)

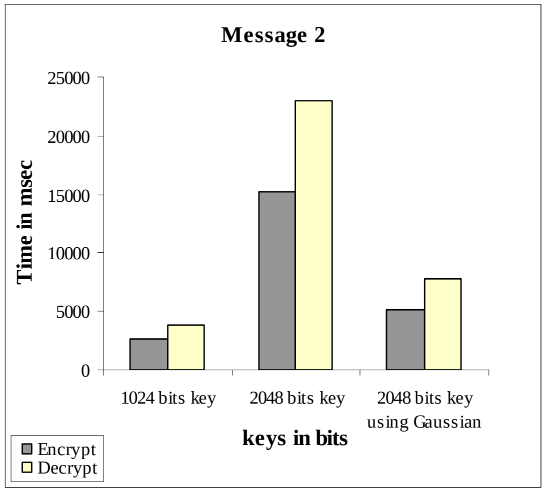


Fig. 2. Message 2

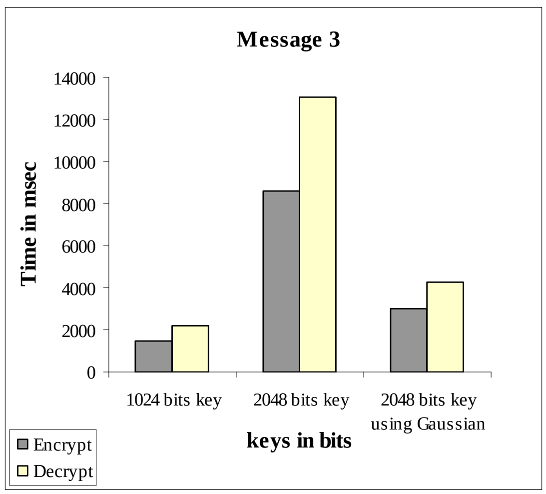


Fig. 3. Message 3

After executing the authentication function of SSL ten times, and implementing RSA in the two domains Integers and Gaussian integers we obtain the following results as shown in Figure 1:

1) The average time needed to encrypt the message "Hello Server" using $1024$ bits key is $0.5953$ seconds and the average time needed to decrypt the same message is $0.8674$ seconds.
2) The average time needed to encrypt the message "Hello Server" using $2048$ bits key generated using two primes each with $1024$ bits is $3.5236$ seconds while the average time needed to decrypt the message is $5.1954$ seconds.
3) The average time needed to encrypt the same message as above using $2048$ bits key generated using two primes each $512$ bits under the domain of Gaussian integers is $1.1985$ seconds while the average time needed to decrypt the message is $1.736$ seconds.

We have tested the above three examples on other messages such as:

1) Hello server, I need to check if you are operational.
2) Hello client, I am operational.

The corresponding results are shown in Figures 2 and 3 respectively.

Also, we have tested the three different key sizes on key exchange. Key exchange is usually used to exchange symmetric keys between parties and usually it uses $128$ bits key. The corresponding result is shown in Figure 4 for a key equal to: $949548400134250557732413815862324586391$.

From Figures 1-4, we can conclude that the time needed to encrypt or decrypt a message using Gaussian integer with key size $2048$ bits is double the time needed to encrypt or decrypt any message in the domain of integers with key size $1024$ bits. While encryption and decryption using $2048$ in the domain of integer is 6 times greater than the one uses $1024$ bits.

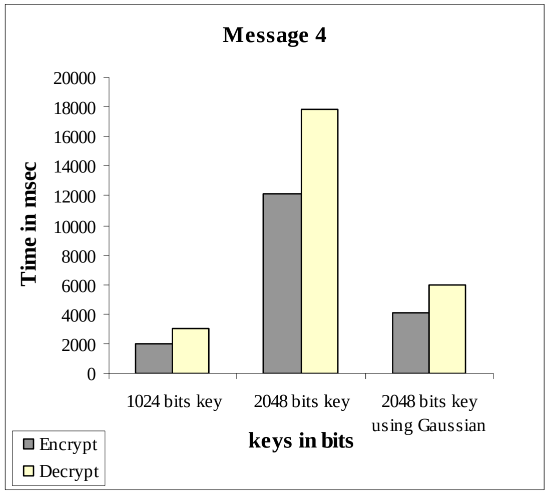


Fig. 4. Message 4

## VI. Conclusion

SSL is considered to be one of the most widely used protocols for securing the Internet. Authentication, confidentiality and integrity are the main functions used to design SSL protocol. Authentication is the function used to authenticate both parties and to exchange the symmetric keys that are used to exchange the messages in a confidential way. The third function is used to ensure the integrity of the data against interfering. The RSA is used for authentication, Computer speed comes much faster than before, and $512$ bits keys are considered as unsecured keys then SSL has been modified to work under $1024$ bits keys. The $1024$ bits keys are used to authenticate both parties but it is recommended to use $2048$ bits keys to ensure the authentication which usually needs 6 times slower than $1024$ bits keys for encryption and decryption. $2048$ bits keys are considered more secure than $1024$ bits and because the time of encryption and decryption is a main problem we modified the authentication function by modifying RSA from the domain of integers to the domain of Gaussian integers which takes double the time needed by $1024$ bits keys and less than the half time needed by $2048$ bits under the domain of integers.

## References


[1] Cisco Systems, *Introduction to Secure Sockets Layer*, http://www.cisco.com.
[2] A. O. Freier, P. Karlton and P. C. Kocher, *The SSL Protocol*, version 3.0, http://wp.netscape.com/eng/ssl3/draft302.txt.
[3] R. Haraty, A. N. El-Kassar and H. Otrok, *A Comparative Study of RSA based Cryptographic Algorithms*, Proc.13th International Conference on Intelligent & Adaptive Systems and Software Engineering (IASSE-2004), Nice, France, p.p. 183-188.
[4] IT security web site, *The risks of short RSA keys for secure communications using SSL*, www.itsecurity.com/papers/ncipher1.htm.
[5] IT security web site,*The Secure Sockets Layer Protocol Enabling Secure Web Transactions*, www.itsecurity.com/paper.
[6] A. J. Kenneth, P. C. Van Orshot and S. A. Vanstone,*Handbook of applied Cryptography*, CRC press, 1977.
[7] H. Otrok,*Security testing and evaluation of Cryptographic Algorithms*, M.S. Thesis, Lebanese American University, June 2003.
[8] RSA website,*5.1 Security on the Internet*, www.rsasecurity.com.
[9] RSA website,*3.1 RSA*, www.rsasecurity.com.
[10] W. Stallings,*Cryptography and Network Security*, $2^{nd}$ *ed.*, Prentice Hall, Upper Saddle River, NJ, 1999.
[11] Steve Mathews,*SSL – does it provide the security that users believe?*, 14 March, 2002, www.itsecurity.com.
[12] UPAQ Ltd,*An Introduction to Encryption and PKI*, Zurich, Switzerland, http://www.itsecurity.com/papers/upaq.htm.