Volume 4, Issue 11, November 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Mod-Encoder Based Compression Algorithm for Secure Data Communications

[1]G. V. R. Pavan Kumar, [2]N. S. L. Kumar. Kurumeti
[1]Post Graduate Student, [2]Assistant Professor
Department of CSE, Aditya Engineering College,
Surampalem, A.P., India

Abstract: ***Now a days, data communications over internet is increased, as the internet usage is increased, the traffic and adversary attacks on the communication path also increased. To avoid attacks, secure data communications are needed to provide privacy and confidentiality of data. Cryptography is the most secure mechanism to provide privacy and confidentiality. The standard encryption techniques do not consider compression techniques and standard compression techniques do not consider security issues. A Mod-Encoder mechanism over a language L, which provides encryption followed by compression. In this mechanism the server will acts as a group controller. It generates a DH-style (diffie helmen) shared group key. The key will be distributed among the group members. To communicate with the server and to generate the encryption key all the members will use this group key.***

Keywords— ***Cryptography, Data Compression, Decryption, Encryption, Number representation, Modulo Arithmetic.***

I. INTRODUCTION

The rapid growth of Internet in the recent days and the wide spread availability of networks have lead to the development of powerful and creative applications are becoming online, not to mention the Google Docs and Microsoft Office Live. So, the networks have become more open and accessible. The volume of data transmitted over the Internet is also increasing. Presently, we have eBooks, multimedia, e-business, open source applications, etc. online. Therefore, the information on the Internet is becoming more sensitive and vulnerable. Many applications demand confidential data communication between the sender and the receiver. Moreover, such overwhelming internet traffic demands the efficient use of bandwidth available. So, we require secure communication with low bandwidth usage. In this regard, the role played by data compression becomes significant as it offers an attractive approach for reducing the communication costs by using the available bandwidth effectively. Furthermore, textual data, where every single character matters, cannot be afforded to be compressed with Loss Compression techniques.

Hence, 'Lossless Data Compression Algorithms' are applied to ensure the reconstruction of the original data from the compressed data. Many data compression techniques are used to reduce the size of the data to be communicated. Some of the recently developed ones are depending upon the application. The compression may be lossless or lossy.

On the other hand, science of Cryptology that dates back to Caesars time provides various heuristics for secured communication. Cryptography encompasses techniques that protect the secrecy of messages transmitted over public communication lines. Encryption is done at the sender side which uses some secret information (a key) and produces a cipher in such a way that an eavesdropper cannot interpret. The legitimate receiver does the Decryption with the corresponding key to get back the message.

Although, as mentioned above, a wide variety of techniques have been employed for encryption, research, to the best of our knowledge, which focuses on Data Compression to reduce encryption overheads is less prevalent. Motivated by this, in this paper, we present a secure data communication mechanism which uses an encoding technique for better bandwidth utilization as well as encryption overhead reduction. The encoding mechanism provides a lossless compression, so that, the original data can be reconstructed at the receiver side. The original message $M = m0m1 \ldots mn$ is a string over a language L with finite alphabet set $\sum$. Let $mk \in M$, a letter of the string, corresponds to the *ith* letter of $\sum$.

The message M is encoded as a bi-tuple $< R,Q >$. R represents a vector of remainders with respect to a modulus Δ, $\Delta < |\sum|$. That is, the *kth* element of R is $i\%\Delta$. Q represents the corresponding quotients in a *Base B* number, where B is the measure of *base* to represent the individual quotients. That is, B equals to $\lceil |\sum| / \Delta \rceil + 1$. All the quotients in Q are within the range $[0, B-1]$. The encoded message M can be securely communicated when the bi-tuple $< R,Q >$is transmitted. While R may be transmitted through open channel, Q should be transmitted through a secure channel.

The rest of the paper is organized as follows. In Section II we present the MOD-ENCODER algorithm with detailed examples. Compression Mechanism is discussed in Section III followed by Examples and Discussions in Section IV. Finally, Conclusions of our work are mentioned in Section V.

II. PROPOSED ALGORITHM

The proposed MOD-ENCODER algorithm uses standard encryption technique which incorporates lossless compression in order to satisfy the needs of low bandwidth and data security. Let *L* be a defined language over the alphabet set ∑. For example, *L* is English language and ∑ be *{A, B, Z, a, b, z}*. Let the letters of ∑ be indexed by a bisection function *I* that maps a letter to an integer *i*, where1 ≤ *i* ≤ | ∑ |. Δ is a constant, called modulus constant.

Let the data string *M* be *{m*1*, m2, m3, mn} ε* ∑. Performing modulus operation on every *I*(*mi*) by Δ, sequentially yields remainder set *R* as *{r*1*, r*2*, r*3*, . . . , rn}* and quotient set *Q* as *{q*1*, q*2*, q*3*, . . . , qn}*. The elements in *R* having the values between [0,Δ − 1]. So, we consider the elements in *R* to be a vector of numbers in base *R*. Each *ri* takes _log2 *R*_ bits for binary representation. If the message is of *n* characters, the number of bits required to represent the vector *R* is *n *[*$\log_2$ *R]*.The quotient set is represented in a different way.

Let $B = \lceil |\sum| \rceil /\Delta+1$ be another parameter called *Base-valuer*. The elements of *Q* will have values within [0,*B*−1]. Consider *Q* as a number in *base B*, i.e. (*q*1*, q*2*, . . . , qn*)*B*. Convert the number to a higher base. It is obvious that a higher base representation would reduce the digits in the number. If *B* is less then 10,the we convert *QB* to a *Q*10 number.

Mod Encoder Algorithm

1) The proposed MOD-ENCODER algorithm provides both encryption and also lossless data compression.
2) L be a language, ∑ be an alphabet set, Data String M be {m1, m2, m3…mn} ∈ ∑.
3) Δ is a constant value and is used for calculates quotients and remainders, and this Δ is generated using any key generation algorithm.
4) The quotient vector is also represented by using B= $\lceil|\sum|/\Delta\rceil$ +1.

And further this B value is used for the compression of the encoded message.

Key-Exchange algorithm

Diffie-hellman is one of the key exchange algorithms and is used for Delta value generation.

- **Global public elements:**

This algorithm considers the two public keys:

p(prime number)
q(primitive root)
q<p.

- **User 'A' key generation:**

User A selects a private key and calculates a public key.
Select private key X_A X_A<p
Generate public key Y_A
$Y_A=q^{XA} \bmod p$

- **User 'B' key generation:**

User B selects a private key and calculates a public key.
Select private key X_B X_B<p
Generate public key Y_B
$Y_B=q^{XB} \bmod p$

- **Generation of secret key by User A:**

User A generates a secret key using his private key and User B public key.
$K=(Y_B)^{XA} \bmod p$.

- **Generation of secret key by User B:**

User B generates a secret key using his private key and User a public key.
$K=(Y_A)^{XB} \bmod p$.

- **Generation of Delta Value**

If the key value < 29
Add 29 to the key
If the key value >=29 and <256
Assume the key value as Delta
If the key value >256
Calculate key mod 256
Generation of Base Value
Base= $\lceil|\sum|/\Delta\rceil$ +1.

The **MOD-ENCODER** Encoding Algorithm can be stated as :

1) Input: *M ε* ∑
2) *n* = |*M*| , i.e. length of *M*
3) *Z* = *n* × *bit size*
   i.e. *bit size* is the number of bits required to represent each character and n is the length of the message.
4) for *i* = 1 to *n*

4.1) Read *mi* the *ith* character from *M*.
4.2) **Find** *R*
$R[i] = I(mi)\%\Delta$
4.3) **Find** *Q*
$Q[i] = I(mi)/\Delta$
5) **Representation of** *R*
for $i = 1$ to n
5.1) Represent $R[i]$ in *Base* Δ.
6) **Representation of** *Q*

Interpret *Q* as *Base B* number and convert it to *Base 10* .The vector *R* is communicated through open channel, whereas *Q* is encrypted to a cipher *QC* using any standard cryptographic technique and communicated to the receiver to ensure the confidentiality of the message *M*. By doing so, the overhead of encryption is reduced as we encrypt only tuple *Q*, rather than the whole message *M*. The receiver on Receiving *R* and *QC*, decrypts *Qc* to *Q* and decodes the message from the bituple < *R*, *Q*>.

The **MOD-ENCODER** Decoding Algorithm can be stated as follows:
1) Input: Bi-tuple < *R*, *Q*>
2) **Convert** *Q* **from** ***Base 10*** **to** ***Base B***
Let $QB = (q1, q2. . . qn)$ be the representation in
Base B
3) **Interpret** *R* **as a vector of Base** Δ number
4) For$1 \leq i \leq n$
4.1) $i = qi \times\Delta + ri$
Where *qi* the *Ith* digit of *QB ri* the *Ith* element of *R*.
4.2) $mi = I - I(i)$
5) $M = (m1, m2, mn)$

III. COMPRESSION MECHANISM

As mentioned above, the encrypted message *M* is a bi-tuple < *Q*, *R*>of quotient and remainder. So, the size of the encrypted message can be obtained by calculating the number of bits required to represent *Q* and *R*. Let *X* be the total number of bits required to represent *R* and is given by $X = n*\log2\ \Delta$, where *n* is the length of the message and _log2 Δ_ is the number of bits required to represent each remainder. The quotient *Q* is looked upon as a *Base B* number. Each qi *needs*log2 *B* bits for its representation.

As we know, a number in *Base 10* requires lesser number of bits than its equivalent in another *Base B* for representation, considering $B<10$.Therefore, to lessen the number of bits required to represent *Q*, we first convert it into a *Base 10* number, say *T*. So, the number of bits required to represent *Q* is given as $Y = {}_{log2T}$. Hence, the total number of bits needed for representation of the encrypted message *M* can be given by $X + Y$. Considering, a 7-bit representation for every character as in ASCII (or 8-bit in Unicode), $Z = n*7$ is the total number of bits required for plain text message.

We can observe that Z >(X+Y). So this reduction in bits provides us with the desired compression and the Compression Ratio *C.R.* is given by *C.R. = X+Y/2.*

IV. EXAMPLES AND DISCUSSION

Let us consider ∑ to be the alphabet set in English with 26 Characters.

MOD-ENCODER Encryption:
Mod-Encoder algorithm is used to encrypt the data, which uses: Message M = "WELCOME TO SRI ", Δ = Constant value=, used to calculate remainders and quotients.

The following table gives the ASCII values, remainders, and quotients for the Message M.

Table quotient and remainders for different values of δ

MESSAGE	ASCII ALUE	QUOTIENT	REMAINDER
W	87	2	27
E	69	2	9
L	76	2	16
C	67	2	7
O	79	2	19
M	77	2	17
E	69	2	9

	32	1	2
T	84	2	24
O	79	2	19
	32	1	2
A	83	2	23
D	65	2	10
I	73	2	13
T	84	2	24
Y	89	2	29
A	65	2	5

Compression of the message:

Compression mechanism is used to reduce the size of the encoded message. For this which converts the Base B to Base 10 is as follows:

- Q_B=Quotient vector with base9
  $= (22222221221222)_9$
- Q_{10}= T = Quotient vector with base 10
  $= (27354744737524945631)_{10}$
- X = Total Number of bits needed for Remainder Vector:
  $= n*\log_2 \Delta$
  $=23*_{\log 2}{}^{30}$
  =153.465782866
- Y = Total Number of bits needed for Quotient Vector without compression:
  = 20.025145647
- X+Y=Total Number of bits needed for message without compression
  =153.490928513
- Y' = Total Number of bits needed for Quotient Vector with compression:
  $= \log_2 T$, T is the low base value to high base value.
  $= \log_2{}^{27354744737524945631}$
  = 17.900855321
- X+Y'=Total Number of bits needed for message with compression =120.366638187
  Q'=27354744737524945631
  User A sends tuple(Q',R) to User B.

MOD-ENCODER Decryption:

Mod-Encoder Decryption is used to decrypt the message using compressed Quotient and Remainder vector to get the original message.

- User B receives tuple(Q',R) from User A

Q'=27354744737524945631

R=27,9,16,7,19,17,9,2,24,19,2,23,5,13,

Q = Converting from base 10 to base 9

=2,2,2,2,2,2,2,1,2,2,1,2,2,2,

For i=1 to 21

I(i) = Qi * Δ + Ri

I(1) = 2 * 30 +27

=87

m1 = W , m2 = E, ……m21 = A.

- M={m1,m2…m23}= " WELCOME TO ADITYA "

V. CONCLUSION

The proposed MOD-ENCODER technique represents the message as a bi-tuple < *R,Q* >.The individual tuples can be communicated independently to the receiver. The secrecy of the message is ensured by encoding one half, generally *Q* ,of the bi-tuple. The error probability of decoding is considerably high when either *Q* or *R* is unknown. The proposed technique also provides a lossless compression that facilitates better bandwidth utilization. Moreover, as the encryption is applied it reduces the computational complexity.

REFERENCES

[1] G. Praveen Kumar*, Biswas Parajuli, Arjun Kumar Murmu, Prasenjit Choudhurand Jay deep Howlader "A Lossless Mod-Encoder Towards A Secure communication" (2010 International Conference on Recent Trends in Information).

[2] Huffman's original article: D.A. Huffman, *"A Method for the Construction of Minimum-Redundancy Codes"*, Proceedings of the I.R.E., Sep.1952, pp.10981102.

[3] A G.R. Blakley, "Safeguarding Cryptographic Keys," Proc. Am. Federation of Information Processing Soc. (AFIPS '79) Nat'l 317, 1979.

[4] S. Berkovits, "How to Broadcast a Secret," Proc. Eurocrypt'91 Workshop Advances in Cryptology, pp. 536-541, 1991.

[5] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," Proc. Euro crypt '84 Workshop Advances in Cryptology, pp. 335-338, 1984.

[6] Debra A. Lelewer, Daniel S. Hirschberg *"Data Compression"*, ACM Computing Surveys (CSUR), vol 19, Issue 3, pp. 261 - 296, Sep. 1987.

[7] Elliptic Curve Cryptography, Certicom Research, 2000.

[8] *"Announcing the ADVANCED ENCRYPTION STANDARD (AES)"*,Federal Information Processing Standards Publication 197,Nov. 2001.

[9] William C. Barker, *"Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher"*, National Institute of Standards and Technology, NIST Special Publication 800-67, 2008.

[10] Farina, A.; Navarro, G.; Parama, J.R., *"Word-Based Statistical Compressors as Natural Language Compression Boosters",* Data Compression Conference 2008, pp. 162 - 171, Mar. 2008-Behrouz.

[11] Amit Jain, Ravindra Patel, *"An Efficient Compression Algorithm (ECA) for Text Data"*, icsps, pp.762-765, 2009 International Conference on Signal Processing Systems, 2009.

AUTHOR BIOGRAPHIES

G.V.R. PAVAN KUMAR is an M.Tech student at Aditya Engineering college Surampalem, He has received his B.Tech (C.S.E) degree in 2010., He Completed his diploma in (D-CM-E) from Andhra Polytechnic college, Kakinada in 2006., His areas of interests are information security and networks, Cloud Computing.

N.S.L. Kumar. Kurumeti received M.C.A from Acharya Nagarjuna University in 1999, M.Tech (CSE) form Jawaharlal Nehru Technological University, Kakinada in 2010, currently a research scholar at Acharya Nagarjuna University, Guntur. He has 13 years of teaching experience, currently working as Asst.Professor at Aditya Engineering College Surampalem, India. He guided Graduate and Post Graduate students project works. His area of interests includes Data Mining & Knowledge discovery, Pattern recognition, Cloud Computing, Analysis of algorithms, Grid computing.

