An Enhanced Code Encryption Approach with HNT Transformations for Software Security

Sasirekha N
Department of Computer Science
Karpagam University
Coimbatore
Tamilnadu, India

Hemalatha M
Department of Computer Science
Karpagam University
Coimbatore
Tamilnadu, India

ABSTRACT

Security threats such as viruses, worms, trojans and spyware affects the security and authentication of software codes, forcing software developers to build security schemes for better software protection. These software threats exploit the authenticated data of the software and confidentiality, integrity and accessibility is greatly affected by these software threats. A number of code security techniques like tamper resistant packaging, code obfuscation, register encoding etc have been developed which mainly concentrates on providing solutions for a particular type of threats and are vulnerable to code tampering and code injection by complicated attackers. Hence, code encryption technique has become an active area of research. This paper proposes a novel software protection code encryption scheme based on the index table. This approach uses a novel and efficient encryption technique called quasigroup encryption for encryption the indexed table. It provides least resemblance of the original data when encrypted. But, quasi group encryption is not efficient in diffusing the statistics of the plain text. This drawback can be overcome by using transforms. Hence, this approach uses chained Hadamard transforms and Number Theoretic Transforms to introduce diffusion along with the quasigroup transformation. The proposed approach is compared with the other encryption approaches and is observed to provide better results.

Keywords

Software Piracy, Cryptography, Encryption, Quasigroup, Indexed Table, Number Theoretic Transforms and Hadamard Transforms.

1. INTRODUCTION

In recent years, Software protection has significant importance, in the field of software engineering. The importance about software protection has been an attractive area of research as software itself is vulnerable to theft and misuse [1].

Because of various software threats and attacks, a number of software protection schemes have be developed by various researches in the literature [2]. Previous protection approaches were largely limited to direct media-based protection and serial numbers. Various approaches such as processor dependent code, encryption, and obfuscation [3, 4] have been developed for the software protection.

More importantly, securing the software codes from attacks such as reverse engineering [5], analysis and tampering attacks is one of the main concerns in software industries. When security is concerned, cryptographic approaches have received the greatest academic attention, because of its classic mathematical data-manipulation algorithms involving secret keys, encryption algorithms for confidentiality and Message Authentication Codes (MACs) and digital signature algorithms for real-time authentication, data origin authentication, integrity or non-repudiation [6]. Therefore, Cryptography is observed to be the technique that can be incorporated in the software protection technique for improved protection [7].

In cryptography, encryption is the technique of hiding and securing information to make it unreadable for the intruders without required information. Encryption is very useful for secrecy, and typically for confidential communications [8, 9]. Encryption has been one of the efficient techniques to hide and secure information [10].

This research mainly concentrates on the protection of software based on the efficient cryptographic encryption technique. The concept of efficient code encryption techniques is used in this paper which offers confidentiality and a method to create code dependencies that implicitly protect integrity need to be established [11].

Efficient symmetric encryption approaches with randomization and hashing approaches provide effective confusion and diffusion [12, 13]. Quasigroup encryption approach is observed to be an efficient approach to generate an astronomical number of keys and thus it provide significant results at confusion [14]. But, a major drawback of the quasi group encryption is that it is ineffective in diffusing the statistics of the plain text. For quasigroup mappings in encryption, it is essential, to employ the quasi group mapping integrated with other statistics-diffusing mappings.

Therefore, in order to introduce diffusion, transformation approaches are incorporated with the quasi group approach. Thus, Hadamard and Number theoretic transformations are utilized in this approach to introduce diffusion the statistics of the plain text [15].

Number Theoretic Transforms are also a certain kind of discrete Fourier transforms. Number Theoretic Transform is based on generalizing the nth primitive root of unity to a quotient ring rather than using complex numbers. Figure 3 represents the general architecture of the proposed encryption and hash system scheme.

The encryption technique used in this approach is the quasigroup approach, Hadamard transformation and Number theoretic transformation for encrypting the indexed table data to make it tough for the intruder to hack the data.

2. LITERATURE SURVEY

Cappaert et al. (2008) [16] presented a partial encryption approach depending on a code encryption approach. In order to utilize the partial encryption approach, binary codes are partitioned into small segments and encrypted. The encrypted binary codes are decrypted at runtime by users. Thus, the partial encryption overcomes the faults of illuminating all of the binary code at once as only the essential segments of the code are decrypted at runtime.

Jung et al. (2008) [17] presented a code block encryption approach to protect software using a key chain. Jung's approach uses a unit block, that is, a fixed-size block, rather than a basic block, which is a variable-size block. Basic blocks refer to the segments of codes that are partitioned by control transformation operations, such as "jump" and "branch" commands, in assembly code. Jung's approach is very similar to Cappaert's scheme. Jung's approach tries to solve the issue of Cappaert's approach. If a block is invoked by more than two preceding blocks, the invoked block is duplicated.

(Gutmann, 2000) [18] put forth an apparent conversation of the security concerns facing cryptographic usage in software under general-purpose operating systems, and analyzes the design difficulties in nullifying these concerns faced by using secure cryptographic co-processors.

Various code encryption schemes have been adopted to protect the software against attacks like reverse engineering. A code encryption technique encrypts the binary executable code. Key management is the essential segment of the code encryption approach. Thus, (Sungkyu Cho et al. 2011) [19,33,34].proposed an approach which analyzed the previous code encryption approaches and then presented a code encryption scheme based on an indexed table.

However, the above discussed schemes did not meet the security requirements and moreover had an efficiency problem. Moreover, time cost and space cost should also be taken into consideration. Thus, a novel cryptographic technique is proposed in this approach which is the extension of the (Sungkyu Cho et al. 2011) approach [19,33,34].

3. METHODOLOGY

A code encryption scheme is proposed based on an indexed table to protect software. The indexed table can solve the problem of multiple paths. Moreover, it solves such problems as loops, recursions, and multiple calls.

Step 1: Source code compiling process. After this step, the source code is compiled and outputs a binary image.
Step 2: Construction of the indexed table. It is the most important procedure of our scheme.

A. Construction of Index Table

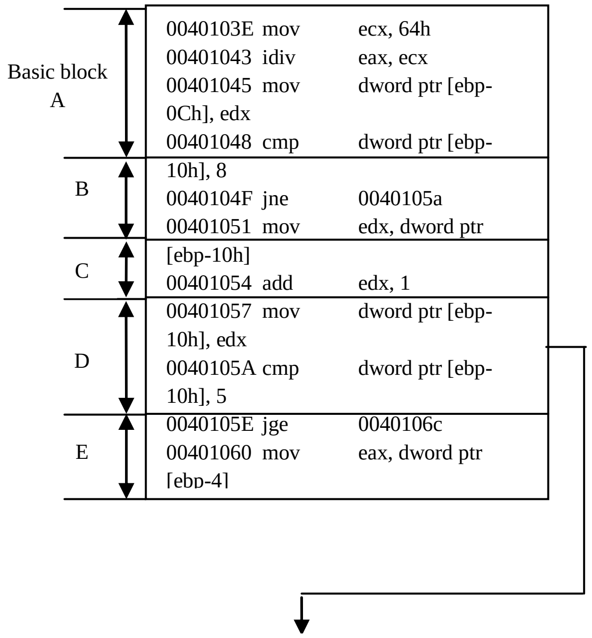
The correct key chain is obtained by means of the indexed table. The construction of the index table follows the set of procedure [19].

Initially, the present address of the basic block is stored, and the "*jump*" or "*branch*" command in the basic block is examined by moving the pointer. The commands consist of a block's address, which will be executed in the subsequent step. If the next address refers to the present address of the basic block, this shows a loop or recursion. When a loop or a recursion takes place because of the "*cmp*" command with the number of calls, the number of calls is marked in the table. Similarly, if a current address of a block is already stored in the table, this shows multiple calls. The *PK* is created at this time and stored in the binary image in a data section [19].
A fundamental block *D* is called upon by multiple blocks *B*, *C*, and *F*. The secret key of *B* is the hash value of block A, and the secret key of block *C* is the same. *F* is not invoked by block *A* directly, but it invokes the basic block *D*.

At this moment, random number *r* is created for the secret key of *D*, and then it is encrypted with *IK*. The result of the encryption is *PK*. The *PK* is stored in executable images. Generic operating systems, such as Windows or Linux, store variables in the data section of an executable image. Thus, the *PK* is stored in the data section of an executable image. The indexed table consists of the number of iterations and recursions. If this is not taken into account, a basic block which has loops or/and recursions will be decrypted several times. Thus, if the number of loops and recursions in the table is marked, this problem can be prevented. When a basic block has been called, the number of calls is minimized by one, and then if the number of calls became zero, the block should be re-encrypted from memory to prevent against a memory dump.

The second operand of the "*cmp*" command is 0Ah. It shows the block "loc_401006" will be executed 10 (=0x0A) times, and that is the number of loops or recursions. Moreover, an example of constructing the indexed table is shown in Figure 1. The example code consists of five basic blocks. The basic blocks are partitioned by "*jump*" or "*branch*" commands. In the beginning, initialization is carried out to construct the indexed table [19]. 0x0040103E is set as the starting point of the program. Then, the commands are examined to discover the "*jump*" or "*branch*." If the command is "*jump*" or "*branch*," store the operand of the command in the table as it becomes the first address of another block. In this example, 0x0040105A is stored in the table due to the command "*jne* 0x0040105A," which is at0x0040104F. The next address of the command becomes the first address of another block. So, 0x00401051 is stored in the indexed table. Thus, 0x0040106C and 0x00401060 are stored in order. At 0x0040106A, the command "*jmp* 0x00401051" is discovered. 0x00401051 has been stored already, which shows that there are multiple paths taking into account the address 0x00401051. Thus, the block's data should be updated, and the random number should also be created. Thus, all the blocks can be identified [19].

Step 3: The above constructed indexed table is given as input to the Quasi group encryption technique.



```
0040103E  mov      ecx, 64h
00401043  idiv     eax, ecx
00401045  mov      dword ptr [ebp-
0Ch], edx
00401048  cmp      dword ptr [ebp-
10h], 8
0040104F  jne      0040105a
00401051  mov      edx, dword ptr
[ebp-10h]
00401054  add      edx, 1
00401057  mov      dword ptr [ebp-
10h], edx
0040105A  cmp      dword ptr [ebp-
10h], 5
0040105E  jge      0040106c
00401060  mov      eax, dword ptr
[ebp-4]
```

Address (offset)	Block size	Number of Calls	Flag
0x0040103E	19	1	0
0x00401051	9	2	1
0x0040105A	6	2	1
0X00401060	12	1	0
0X0040106C	8	2	1

Figure 1: Example of Constructing Indexed Table

B. Quasigroup Encryption for the Indexed Table

The encryption technique used in this approach is the quasi group encryption technique [20]. The quasigroup encryptor has very good data-scrambling properties and thus, it has effectively used in symmetric cryptography. The purpose of the scrambler is to maximize the entropy at the output, even in cases where the input is constant. The great complexity connected with the task of identifying the scrambling transformation assures the effectiveness of the encryption process. Quasigroup encryption is a development that has permutation based scrambling [21] at its basis. Figure 2 shows the block diagram of the quasi group encryptor.

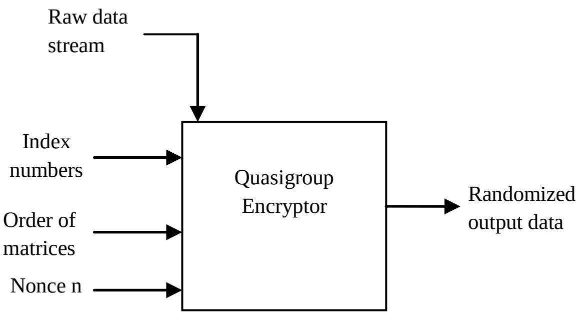


Figure 2: Quasi Group Encryptor

Input data*:* d_1, d_2, d_3,… d_n
Output data: e_1, e_2, e_3,…e_n

The two matrices*: R, S*
Multiplier Elements: q_1, q_2, q_3,… q_n
The indices: I_1, I_2, I_3, … I_n
The encryptor is defined by QE (stands for Quasi-Encryptor), and the decryptor is defined as QD (stands for Quasi-Decryptor).

Encryption: It should be that if Q is a quasigroup such that a_1,a_2, a_3,... a_n belong to it then the encryption operation QE, which is defined over the defined elements, maps those elements to another vector $\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \dots \mathrm{b}_\mathrm{n}$ such that the elements of the resultant vector also belong to the same quasigroup.
The mathematical equation used for encryption (basic level) is defined by:

$$E_a(a_1, a_2, a_3, \dots, a_n) \;=\; b_1, b_2, b_3, \dots b_n \qquad (1)$$

where the output sequence is defined by:
b_1= a∗ a_1

$$b_i = b_{i-1} * a_i$$

where *i* increments from 2 to the number of elements that have to be encrypted, and *a* is the *hidden key* (*leader* in Markovski and Dimitrova terminology [22]. Equation (1) describes a typical single level quasigroup encryptor.
It is assumed that the initial input data given by the vector $\mathrm{a}_1, \mathrm{a}_2, \mathrm{a}_3, \mathrm{a}_4, \mathrm{a}_5, \mathrm{a}_6$. It is mapped to the vector $\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4, \mathrm{b}_5, \mathrm{b}_6$ by equation (1). The following steps are used during the process of encryption:

$$b_1 = a * a_1 = 2 * 2 = 1$$
$$b_2 = b_1 * a_2 = 1 * 4 = 1$$
$$b_3 = b_2 * a_3 = 4 * 1 = 4$$
$$b_4 = b_3 * a_4 = 4 * 2 = 5$$
$$b_5 = b_4 * a_5 = 5 * 3 = 1$$
$$b_6 = b_5 * a_6 = 1 * 3 = 2$$

Quasigroups are very competent in generating an astronomical number of keys and are significant at confusion [23] but are not very efficient at diffusing the statistics of the plaintext. In particular, the quasigroup transformation can be easily identified by the known plaintext attack. For quasigroup mappings in encryption, it is essential, to employ this mapping together with other statistics-diffusing mappings [24].

C. Proposed Quasi group Encryption Approach with Efficient Transformation Approaches

The usage of transforms [25] would effectively diffuse statistics where the security is improved through a variety of them and by transforming them [26]. The employment of chained Hadamard transforms and NTTs (number theoretic transforms) are investigated in this appraoch to introduce diffusion together with the quasigroup transformation.

In this approach, the input sequence will be subjected to different transformations sequentially like quasigroup transformation, Hadamard transformation and Number theoretic transformation. For Hadamard and Number theoretic transforms, the input data is divided into definite group of bits in a manner that each group bit count is the order of the equivalent matrix.

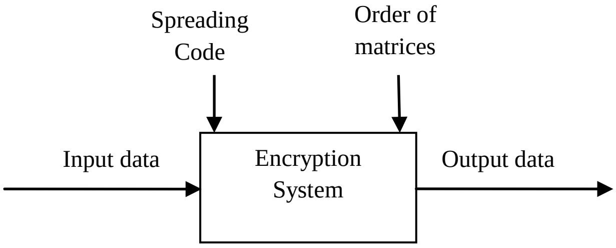


Figure 3: General Architecture of the Proposed Encryption and Hashing System

i. Hadamard Transforms

The Hadamard transform is a generalized class of discrete Fourier transforms [27], [28]. It is created either recursively, or through binary representation. All the values in the matrix are non-negative. Each negative number is replaced with equivalent modulo number. For instance in modulo 7 Hadamard matrixes -1 is replaced with 6 to make the matrix non-binary. Because of its symmetric form, it can be used in applications such as data encryption and randomness measures [29]-[30]. Only prime modulo operations are carried out since non-prime numbers can be divisible with numbers other than 1 and itself [23]. Recursively, 1 × 1 Hadamard transform H_0 is defined by the identity $H_0 = 1$, and then define H_m for $m > 0$ by,

$$H_m = \frac{1}{\sqrt{2}}\begin{pmatrix} H_{m-1} & H_{m-1} \\ H_{m-1} & -H_{m-1} \end{pmatrix}$$

A Hadamard matrix, H_n, is a square matrix of order n = 1, 2 or
4k where k represents a positive integer. The elements of H are either +1 or –1 and $H_n . H_n{}^T = n\boldsymbol{I}_n$, where $H_n{}^T$is the transpose of H_n, and I_n is the identity matrix of order n. A Hadamard matrix is said to be normalized if all of the elements of the first row and first column are +1. Some examples of the Hadamard matrices are given below

$$H_0 = +1$$

$$H_m = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Hadamard matrix of modulo 31 of size 8*8

1	1	1	1	1	1	1	1
1	30	1	30	1	30	1	30
1	1	30	30	1	1	30	30
1	30	30	1	1	30	30	1
1	1	1	1	30	30	30	30
1	30	1	30	30	1	30	1
1	1	30	30	30	30	1	1
1	30	30	1	30	1	1	30

Hadamard matrix of modulo 7 of size 4*4

1	1	1	1
1	6	1	6
1	1	6	6
1	6	6	1

The concept of encryption is to multiply the decimated input sequence with the non-binary Hadamard matrix in a chained manner block by block. The block size is based upon the size of the selected Hadamard matrix. Input sequence is taken in the form of column matrix. Figure 4 shows the block diagram of Hadamard Encryption.

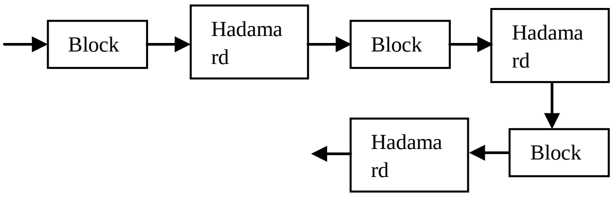


Figure 4: Hadamard Encryption

ii. Number Theoretic Transforms

Number Theoretic Transform depends on generalizing the nth primitive root of unity to a quotient ring rather than through complex numbers [31].

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & w & w^2 & w^3 \\ 1 & w^2 & w^4 & w^6 \\ 1 & w^3 & w^6 & w^9 \end{pmatrix}$$

The unit w is exp (2π/ n). Number Theoretic Transform is now all about is that $w^n = 1$
NTT matrix of order 6*6

1	1	1	1	1	1
1	3	2	6	4	5
1	2	4	1	2	4
1	6	1	6	1	6
1	4	2	1	4	2
1	5	4	6	2	3

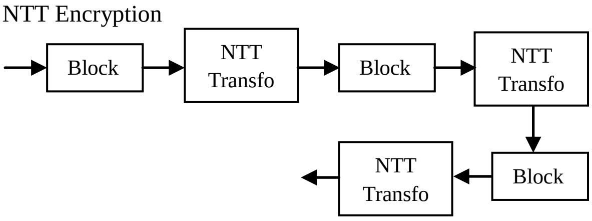


Figure 5: Number Theoretic Encryption

Figure 5 shows the block diagram for the Number Theoretic Encryption. The notion of encryption is to multiply the decimated input sequence which is the output attained after encryption by means of Hadamard transform with the non-binary Number theoretical matrix in a chained manner block by block. The block size is based upon the size of the selected Number theoretical matrix. The Input sequence is taken in the form of column matrix.

4. PROPOSED ENCRYPTION SYSTEM

A. *Encryption*

- Phase1: Encryption of input data using quasigroup based encryption system.
- Phase2**:** Output of Phase1 is given as input to the Phase2. In phase2 Hadamard transformation of data is carried out.
- Phase3: Output of Phase2 is given as input to the Phase 3. In phase 3 Number Theoretic transform is performed.
- Phase4: Phase2 is repeated with a different order of Hadamard matrix.

These four phases are clearly depicted in figure 9.

B. *Decryption*

As the Hadamard matrix operations are invertible, decryption of the data can be performed by generating inverse Hadamard matrix.

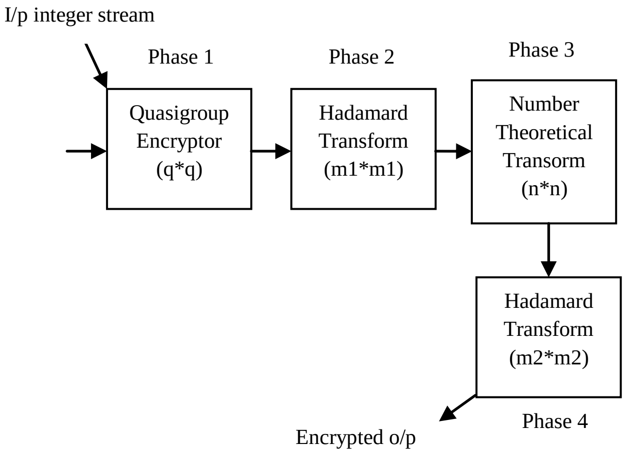

I/p integer stream
Phase 1
Phase 2
Phase 3
Quasigroup Encryptor (q*q)
Hadamard Transform (m1*m1)
Number Theoretical Transorm (n*n)
Hadamard Transform (m2*m2)
Phase 4
Encrypted o/p

Figure 6: Proposed Encryption Systems

All the matrices such as the quasigroup, Hadamard Matrix and Number Theoretic transform matrix should have different orders in such a way that at each encryption level, the size of the input block size varies which eventually raises the randomness of the input sequence. Input size should be multiple of orders of all three matrices i.e. quasigroup, Hadamard and Number theoretic matrix to attain the appropriate block size.

Hadamard transforms and Number Theoretic transforms perform as hash functions which produce diverse hash values for different input values [23]. There is a huge difference in the generated random sequence if there is a one bit change in the input sequence.

5. EXPERIMENTAL RESULTS

This experimental result section mainly focuses on the security analysis and performance analysis of the proposed approach. The performance of the proposed approach is compared with standard software schemes.

A. *Security Analysis*

In order to enhance the security, the indexed table approach is adopted based on a quasi group encryption scheme. The main focus of the software protection is to secure the original binary code from various attacks by remaining confidential. The proposed approach uses the quasi group encryption technique that has very significant data-scrambling properties and thus it has significant uses in symmetric cryptography. The main aim of the scrambler is to increase the entropy at the output, even if the input is constant. The enormous complexity connected with the task of identifying the scrambling transformation assures the effectiveness of the encryption process. Cappaert's scheme did not satisfy the correct key chain requirement [11].

B. *Experimental Set up and Result*

The implementation of the proposed approach is based on certain set up. The operating system used for the proposed approach is Windows XP and is implemented using Microsoft Visual Basic.Net. The cryptographic library and CPU used is Win32 OpenSSL version 0.9.8 and Intel Core2Duo CPU E7200 respectively.

PEDasm version 0.33 is referred for this experiment which is an open source disassembler. In order to evaluate the performance, three small default executable files in Windows XP are chosen. A stream cipher, quasi group is used as a cryptographic algorithm to encrypt and decrypt the code. Initially, the executable file is entered, disassembled, and partitioned into basic blocks. Then, the program executes table indexing and code encryption through the partitioned basic block.

C. *Performance Evaluation*

Two sets of performance evaluation metrics are used in this experimental validation.

i. *Set 1*
- time cost C_t
- space cost C_s

ii. *Set 2*
- the number of basic blocks that need protection
- the execution penalty (overhead)

The performance of the proposed approach is compared with the (Sungkyu Cho et al. 2011) code encryption scheme [19] and the code encryption using quasi group encryption. For instance, if a program P and its modified version P' is available. Then, the time cost C_t and the space cost C_s is defined as

$$C_t(P,P^{'}) = \frac{T(P^{'})}{T(P)}$$

$$C_s(P,P^{'}) = \frac{S(P^{'})}{S(P)}$$

where $T(X)$ is the execution time of program X, and $S(X)$ is its size. The encryption time and decryption time of three programs are evaluated. At the moment, external libraries such as ".dll" files are eliminated as they are implemented externally to the executable file. The results are shown in Table 1.

Table 1: Results Comparison

Features	Sungkyu Cho et al., Code Encryption Scheme		Indexed Table based Quasigroup Approach		Proposed Quasigroup with HNT Transformation	
	Pgm1.exe	Pgm2 .exe	Pgm1 .exe	Pgm2 .exe	Pgm1 .exe	Pgm2 .exe
Original file size (B)	3584	4096	3584	4096	3584	4096
Number of blocks	12	26	12	26	12	26
Decryption and re-encryption time (s)	0.0016	0.0032	0.0012	0.0024	0.0009	0.0018
C_t	6.680	6.119	5.985	5.845	5.625	5.3
C_s	1.027	1.674	0.925	0.942	0.842	0.812

Table 2 shows the efficiency of the proposed quasi group encryption scheme with transformation approach in identifying the different threat regions. The number of vulnerable blocks is determined is less in case of the proposed approach. From the table, it is observed that for all the benchmarks, the number of vulnerable blocks after the execution of the proposed quasi group encryption with transformation approach is very less compared to the other encryptions techniques taken into considerations.

Table 2: Number of Unique Basic Blocks executed and Number of Vulnerable Blocks

Benchmark	Unique Basic Blocks Executed	Number of Vulnerable Blocks		
		Sungkyu Cho et al., Scheme	Indexed Table based Quasigroup Approach	Proposed Quasigroup with HNT Transformation
BITCOUNT	1451	1200	1050	870
CRC	1190	980	810	645
DIJKSTRA	1381	1200	975	725
FFT	1700	1450	1258	986
PATRICIA	2270	1975	1752	1500
SHA	1238	1000	880	650

6. CONCLUSION

Efficient encryption approach is used in this approach for handling the software threats and attacks which has become one of the major concern of the software industries. This paper presented and discussed code encryption schemes for protecting software against various attacks like reverse engineering, tampering etc. A new code encryption approach based on an indexed table to guarantee secure key management is proposed in this paper. Efficient Quasi group encryption technique is used in this paper. In order to improve the security and efficiency, along with the quasi group encryption technique, Hadamard transform and NTT are also used in this approach. These transforms are very effective in introducing the diffusion along with the quasi group encryption. The experimental results reveal that the proposed approach provides significant results in terms of the time cost, space cost and reducing the number of vulnerable blocks.

7. REFERENCES

[1] P.C. van Oorschot "Revisiting Software Protection", pp.1–13, Proc. of 6th International Information Security Conference (ISC 2003), Bristol, UK, October 2003, Springer-Verlag LNCS 2851 (2003)..

[2] N.Sasirekha and Dr.M.Hemalatha, "A Survey on Software Protection Techniques against Various Attacks", Global Journal of Computer Science and Technology Volume XII Issue I Version I, 2012.

[3] Collberg, C., C. Thomborson, and D. Low, "A taxonomy of obfuscating transformations," University of Auckland Technical Report, vol. 170, 1997.

[4] Collberg, C., C. Thomborson, and D. Low, "Breaking abstractions and unstructuring data structures," Proceedings from International Conference on Computer Languages, pp. 28-38,

[5] E. Eilam, Reversing: Secrets of Reverse Engineering, Wiley Publishing, Inc., 2005.

[6] D. Denning, “Cryptography and Data Security”, Addison Wesley, 1982.

[7] Nicol, D.M.; Okhravi, H.; “Performance analysis of binary code protection”, Proceedings of the Winter Simulation Conference, 2005.

[8] Pandiarajan, V.; Martin, T.L.; Joiner, L.L., “Recommendations on a new cellular encryption standard using elliptic curve cryptography”, Proceedings IEEE SoutheastCon 2001.

[9] Tieming Chen; Shilong Ma, “A Secure Email Encryption Proxy Based on Identity-Based Cryptography”, International Conference on MultiMedia and Information Technology, 2008. MMIT '08.

[10] Kostas Zotos, Andreas Litke, “Cryptography and Encryption”, arXiv:math/0510057v1 [math.CT], 2005

[11] Jan Cappaert, Nessim Kisserli, Dries Schellekens, and Bart Preneel, “Self-encrypting Code to Protect Against Analysis and Tampering”, 1st Benelux Workshop Inf. Syst. Security, 2006

[12] C.E. Shannon, Communication theory of secrecy systems. Bell System Technical Journal, 28:656-715, 1949

[13] Yevgeniy Dodis and Krzysztof Pietrzak, “Improving the Security of MACs Via Randomized Message Preprocessing”, Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007.

[14] M.Satti and S.Kak, Multilevel indexed quasigroup encryption for data and speech.IEEE Trans on Broadcasting 55: 270-281, 2009.

[15] J. Hoffstein, J. Pipher, J.H. Silverman, An Introduction to Mathematical Cryptography. Springer,2010.

[16] Jan Cappaert, Nessim Kisserli, Dries Schellekens, and Bart Preneel, “Toward Tamper Resistant Code Encryption: Practice and Experience,” LNCS, vol. 4991, 2008, pp. 86-100.

[17] D.W Jung, H.S Kim, and J.G. Park, “A Code Block Cipher Method to Protect Application Programs From Reverse Engineering,”J. Korea Inst. Inf. Security Cryptology, vol. 18, no. 2, 2008, pp. 85-96 (in Korean)

[18] P. Gutmann, “An Open-source Cryptographic Co-processor”, Proc. 2000 USENIX Security Symposium.

[19] Sungkyu Cho, Donghwi Shin, Heasuk Jo, Donghyun Choi, Dongho Won, and Seungjoo Kim, “Secure and Efficient Code Encryption Scheme based on Indexed Table”, ETRI Journal, Volume 33, Number 1, February 2011.

[20] Maruti Venkat Kartik Satti, “Quasi Group based Crypto-System”, A Thesis, 2007.

[21] Koscienly, C. 2002. Generating quasi groups for cryptographic applications. Int. J. Appl. Math. Comput. Sci., vol.12, No.4, 559–569.

[22] V. Dimitrova, J. Markovski, On Quasigroup Sequence Random Generator. Proceedings of the 1st Balkan Conference in Informatics, Y. Manolopoulos and E. Spirakis, Eds., 21-23 November, 2004, Thessaloniki, Greece, pp. 393 – 401.

[23] M.Satti and S.Kak, Multilevel indexed quasigroup encryption for data and speech.IEEE Trans on Broadcasting 55: 270-281, 2009.

[24] Vaignana Spoorthy Ella, “Sequence Randomization Using Quasigroups and Number Theoretic Transforms”, Third Conference on Theoretical and Applied Computer Science (TACS 2012).

[25] J. Hoffstein, J. Pipher, J.H. Silverman, An Introduction to Mathematical Cryptography. Springer, 2010.

[26] R.S.Reddy, “Encryption of binary and non-binary data using chained Hadamard transforms”.arXiv:1012.4452.

[27] Ulman, L.J. “Computation of the Hadamard Transform and the R-Transform in Ordered Form”, IEEE Transactions on Computers, Volume: C-19, Issue: 4, Page(s): 359 – 360, 1970.

[28] Ce Zhu and Bing Xiong, “Transform-Exempted Calculation of Sum of Absolute Hadamard Transformed Differences”, IEEE Transactions on Circuits and Systems for Video Technology, Volume: 19 , Issue: 8, Page(s): 1183 – 1188, 2009.

[29] V.Godavarty,Using Quasigroups for Generating Pseudorandom Numbers.arXiv:1112.1048.

[30] B.Goldburg, S.Sridharan, E.Dawson, “Design and cryptanalysis of transform-based analog speech scramblers”, Journal of Selected Areas in Communications 11:735-744, 1993.

[31] S. Kak, Classification of random binary sequences using Walsh-Fourier analysis.IEEE Trans on Electromagnetic Compatibility EMC-13, pp. 74-77, 1971.

[32] N.Sasirekha, A.Edwin Robert and Dr.M.Hemalatha, “Novel Obfuscation Obfuscation Algorithms for Software Security”, Proceedings of an International Conference on Networks, Intelligence and Computing Technologies[ICNICT 2011], December 2011**.** ISBN: 978-81-8424-742-8

[33] N.Sasirekha, M.Hemalatha, ”A Survey on Software Protection Techniques against Various Attacks”, Global Journal of Computer Science and Technology, Vol 12, Issue 1, January 2012.

[34] N.Sasirekha, M.Hemalatha,”An Improved Secure Code Encryption Approach Based on Indexed Table”, ACM ICPS, August 2012. ISBN: 978-1-4503-1196-0.