# Merging Secret Sharing within Arabic Text Steganography for Practical Retrieval

**Khaled Aedh Alaseri and Adnan Abdul-Aziz Gutub**

*Computer Engineering Department, Umm Al-Qura University, Makkah, Saudi Arabia*

*Khaled_al3seri@hotmail.com , aagutub@uqu.edu.sa*

## Abstract

*Secret sharing is becoming famous technique in cases where access to important resources has to be saved by many authorized persons. Steganography is another method used to hide information for personal usage whenever required.*

*This paper improves modeling a steganography tool to hide shares generated from target access within texts, helping in the retrieval process. The work has built its model on an Arabic data set of Prophet Haddiths utilizing known Kashida Arabic text steganography, which is considered acceptable methods for hiding information. The suggested new approach merged the secret shares via steganography showing increase in capacity promising for research interesting direction of higher security.*

***Keywords:*** *Information security, Steganography, Kashida Steganography, secret bit sharing.*

## 1. Introduction

Among the increase demand on information technology and multimedia communication, information security is very important concept. Many techniques have been improved and proposed to increase the security of information [1]. Some focused on blending the multimedia information not to be useful, such as cryptography based on mathematical concepts and special arithmetic operations [2] based on different Galois Fields (GF) arithmetic [3]. Other security techniques based on hiding the information itself such as steganography, which also showed different change based on the hiding schemes [4]. Some implemented the security features on general software platforms while others specialized hardware modules for it, showing interesting security achievements [5]. These security techniques are focusing on the idea that one person is controlling the secrecy of information. Nowadays, some applications are requiring security to be performed by several persons. This idea is known as the secret-sharing [6].

Secret sharing schemes are becoming more important [7] for storing high sensitive, of great effect information [8]. Examples include encryption keys, missile launch control, and numbered bank accounts; where access to each of these, must be kept collectively top confidential, as their exposure could be disastrous. Secret sharing schemes are becoming important in cloud computing media. Thus shares of a key can be distributed over many servers by a threshold secret sharing mechanism. The key, i.e. target key, is then reconstructed as needed. In other words, secret sharing can be studied as the method of

distributing the ownership of a secret target key amongst a group of participants, each of whom is allocated a share of this secret. The secret can be reconstructed only when a sufficient number of shares are combined together; whereas individual shares cannot be useful on their own [9]. The secret sharing scheme is a tool can be used in many cryptographic protocols. It is dedicated for assisting key management security and authentication [10].

In secret-sharing, a secret target-key (TK) is split into **n** useful shares, which are distributed by a dealer to a number of participants [8]. In theory, the target key TK can be reformed in A collection A of subsets, however, not all A are useful as appropriate secret shares for our TK combination function. The useful share from set (A) is labeled as (n), where any threshold subset, i.e. **k** out of **n** can reconstruct the target-key TK from its shares. In order to reconstruct the target-key, these k ∈ n shares must be merged together in a specific combination. The reconstruction main concern is that any group of (k) shares or more (k) is the threshold , can together be merged to get the secret TK, but no group of less than (k) participants can. Also, repetition of a share is not allowed in the reconstruction process resulting in false TK output. Such a system is called a (k, n) - or (k) out of (n) - secret sharing threshold scheme. The process of reconstructing the target-key TK from an access structure is called a combiner or target-key TK reconstruction.

We used counting-based sharing because it is flexible to implement in hardware and software. Its security can be improved gradually and its process time is considered practically fast. To be concise with the literature, the main two properties that any secret sharing scheme has to fulfill are:

• Recoverability: where the target-key TK can be reconstructed given any k shares.
• Secrecy: where no information can be known about TK given any number of shares < k.

The idea was first introduced in 1979 by George Blakley [7] and by Adi Shamir [6]. Since then, different efficient schemes were proposed as classified in the next section. We propose a new secret-sharing scheme that works based on recovering the target-key TK via counting the ones of the (k) shares in parallel. The applicable (k) secret shares are placed with their bits in parallel allowing their ones to be counted, i.e. in parallel, making the resulting secret output one if the threshold is passed. The work details the method model and simulates it, adopting two different secret shares generation techniques, i.e. focusing on 1-bit one or 2-bits ones, where both are studied showing promising results.

Three major aspects affecting steganography are security, capacity and robustness. **Capacity** refers to the amount of data bits that can be hidden in the cover medium. **Security** relates to the ability of intruders to figure the hidden information easily. **Robustness** is concerned about the resist possibility to modify or destroy the unseen data. The current steganography science uses the opportunity of hiding into digital multimedia files such as audio, video, image, text and IP datagram.

As we mentioned before, secret sharing phenomena is very important in cases where there is access from many authorized persons and needed together. The system indicates two ways to generate the share numbers from the target key, by changing zeros to one, i.e. each time, as we pass scanning through all bits in the TK number, this is in (1-bit) method; while in (2-bit) technique we change two numbers by applying (1-bit) in the usual way, then produce shares numbers depending on share numbers generated from (1-bit) operation. There

are many applications for this technique, namely Bank Sensitivity Accounts, Error Tracking, Voting Systems Trust, Medical Agreement, Wills and Inheritance. The secret shares are generated automatically requesting a specific remembering way for users as a typical variation than normal password selection, i.e. password choices are normally given to users to decide. Therefore, this research is proposing to help users remember secret shares electronically via storing them via steganography. In fact, literature show many types of steganography, i.e. for hiding secret shares within a text, picture, sound, or a video [4]. This paper, however, we will focus to hide the share numbers in texts, where all our testing is performed on many standard Prophetic Hadith (narration) used as our Arabic-text steganography bench mark, similar in principle to previous work in [11].

This paper is organized as follows. Section 2 covers the related counting-based secret sharing literature survey. It discusses the different classifications of many secret sharing schemes. Then, Section 3 presents Arabic-text steganography introducing the simulation tool built for the initialization data base and design user interface. Section 4 proposed the method of comparisons elaborating on the benefits and drawbacks and effect on the security level. Section 5, concludes the paper with interesting future recommendations.

## 2. Literature Review

There are many studies in steganography using text or pictures as in [11], which suggested an improved approach to embed the secrets into Arabic text covering using Kashida, the Arabic symbol for extension. The suggested approach is improved to hide more information in two digits' binary bits, and a stego system was developed on this approach, and after many tests and assessments a good system was developed that is capable of hiding text.

In [12] the text was hidden in sensitive text so as to prevent its detection. The study concentrated on Arabic texts improved hiding secrets in the text using Kashida to cover the text in the multimedia files. It was suggested to modify Kashida cover technique by imbedding sensitive data within whitespaces in its new method. The ability of this modification made hiding secret data in Arabic text tested within the last 30 Surah (Chapters) of the Holy Quran (Sura Al-Buruj #85 to Sura An-Nas #114), and was compared to the usual method with improving the suggested method; the results demonstrated clear increase in capacities, as expected, with keeping secrecy and security in practical level.

In [13] a new approach was suggested for Arabic-text steganography. The main idea that each Arabic word could have some letters that could be extended by Kashida, the ranks 'locations' such characters and the inserted Kashida, and construct coding method to represent block of secret bits, different scenarios were suggested based on the maximum number of Kashidas possible to be inserted per word; and was compared to existed Arabic texts steganography approaches in capacity and security. *This paper improves modeling a steganography tool to hide shares generated from target access within texts, helping in the retrieval process. The suggested new approach merged the secret shares via steganography showing increase in capacity promising for research interesting direction of higher security.* In shares generation stage we generate set of shares out of the target key. In this paper we assume that the target key is 32 bit; so that the target key is not composed of (zeros) or (ones) only, (extraction process is called continuous based secret shares), then we test these share numbers by

choosing the allowed numbers to reconstruct the target key (K). It is not allowed to repeat the shares; and the number of extracted shares is equal to number of (zeros) in the target key, as the total of numbers which we extracted out of the target key is A, and the extracted numbers are called N. And K out of N and N belongs to A; we have two methods to extract the shares through (1-bits) or (bits-2) we have to choose all the shares which we will use before we introduce the participants so all possible probabilities for extracting the shares according to K value are successful, so that they are in pairs if the result is equal to the target key the operation is added to list of successful numbers. Total of number of shares is N=14.

**Example 1 , 32 bits**

| | | | |
|---|---|---|---|
| Target key : | 01011101110011010110110000110101 | >> | 5DCD6C35 |
| Shares 0 | 00000000000000000000000000000000 | >> | 00000000 |
| Shares 1 | 11011101110011010110110000110101 | >> | DDCD6C35 |
| Shares 2 | 01111101110011010110110000110101 | >> | 7DCD6C35 |
| Shares 3 | 01011111110011010110110000110101 | >> | 5FCD6C35 |
| Shares 4 | 01011101111011010110110000110101 | >> | 5DED6C35 |
| Shares 5 | 01011101110111010110110000110101 | >> | 5DDD6C35 |
| Shares 6 | 01011101110011110110110000110101 | >> | 5DCF6C35 |
| Shares 7 | 01011101110011011110110000110101 | >> | 5DCF6C35 |
| Shares 8 | 01011101110011010111110000110101 | >> | 5DCDEC35 |
| Shares 9 | 01011101110011010110111000110101 | >> | 5DCD6E35 |
| Shares 10 | 01011101110011010110110100110101 | >> | 5DCD6D35 |
| Shares 11 | 01011101110011010110110010110101 | >> | 5DCD6CB5 |
| Shares 12 | 01011101110011010110110001110101 | >> | 5DCD6C75 |
| Shares 13 | 01011101110011010110110000111101 | >> | 5DCD6C3D |
| Shares 14 | 01011101110011010110110000110111 | >> | 5DCD6C37 |

We have many cases for K for adding condition

- **first case : ( number of shares = K)**

We have n=14 K=4

| | | | |
|---|---|---|---|
| Shares 1 | 11011101110011010110110000110101 | >> | DDCD6C35 |
| Shares 4 | 01011101111011010110110000110101 | >> | 5DED6C35 |
| Shares 8 | 01011101110011010111110000110101 | >> | 5DCDEC35 |
| Shares 13 | 01011101110011010110110000111101 | >> | 5DCD6C3D |
| Counting result: | 14044404441044040441440000441404 | | |
| Output: | 01011101110011010110110000110101 | | 5DCD6C35 |

After adding the (shares) to each other, we put each bit equal to value of (K), in this case to one, else it is zero, and then we compare the output with the target key

- **The second case: (number of shares > K)**

This is valid in case the result of adding process is greater than or equal to K

| | | | |
|---|---|---|---|
| Shares 1 | 11011101110011010110110000110101 | >> | DDCD6C35 |
| Shares 3 | 01011111110011010110110000110101 | >> | 5FCD6C35 |
| Shares 5 | 01011101110111010110110000110101 | >> | 5DDD6C35 |
| Shares 7 | 01011101110011011110110000110101 | >> | 5DCF6C35 |
| Shares 9 | 01011101110011010110111000110101 | >> | 5DCD6E35 |
| Shares 11 | 01011101110011010110110010110101 | >> | 5DCD6CB5 |
| Counting result: | 16066616660166061660661010660606 | | |
| Output: | 01011101110011010110110000110101 | | 5DCD6C35 |

- **The third case: (number of shares < K)**

| | | | |
|---|---|---|---|
| Shares 1 | 11011101110011010110110000110101 | >> | DDCD6C35 |
| Shares 3 | 01011111110011010110110000110101 | >> | 5FCD6C35 |
| Shares 5 | 01011101110111010110110000110101 | >> | 5DDD6C35 |
| Counting result: | 13033313330133030330330000330303 | | |
| Output: | 01011101110011010110110000110101 | | 5DCD6C35 |

- **The forth case: (Case of adding shares to shares is wrong (intruder) or more )**

| | | | |
|---|---|---|---|
| Shares 1 | 11011101110011010110110000110101 | >> | DDCD6C35 |
| Shares 2 | 01111101110011010110110000110101 | >> | 7DCD6C35 |
| Shares 3 | 01011111110011010110110000110101 | >> | 5FCD6C35 |
| Shares (false) | 00010101001001010100101101111011 | >> | 5DED6C35 |
| Counting result: | 13143414331034040430431101441314 | | |
| Output: | 00010101000001010100100000110001 | >> | 315054831 |

We notice that the result is not equal to target key

## 3. Modelling Steganography Simulation Tool

In this section we will define the applicable steganography system. Then, we will start building the steganography tool for our testing in addition to studying cases of letters of Arabic language. The section is building the experiments database and design user interfaces of the program for this study.

### 3.1 Steganography and Arabic Texts

Steganography is considered the science of hiding important information and data (secret share, in our case) in multimedia files (text, image, video, sound) in a way that make it difficult for anyone, except for the intended person, to discover the information contained in the file. As message coding (or encryption) make the message secret for anyone except for the receiver and sender, and nobody can decode this message except through the general and/or private key it protects message contents only. Steganography, protect the idea of

hiding as well as the secret sharing sensitive data itself, i.e. suitable for our secret sharing applications [1].

Steganography requires less memory to store data making it easy to transfer and more efficient. Methods of hiding texts differ according to languages and their properties. We propose to hide secret-sharing in Prophet's Hadiths via steganography assuming it is simple to save and fast to store or memorize. The participants select the Hadith as cover-text and the system hides the secret-shares in this Hadith.

In 1996 workshop, organized in Cambridge about hiding information, they introduce clear developments in theory and practice using steganography [9]. They declared the most important conditions to be fulfilled in any steganography system, as to be difficult for any third party to risk, other than the receiver and sender, to distinguish the text, video or sound file containing secret information from ordinary files [12]. The most important features of data hidden with steganography to distinguish from encryption are ability to hide (security), robustness and capacity [16]. Our research focused on Arabic-text steganography considering Arabic text language features. For example, we benefitted from letters of Arabic, Persian, Urdu and other similar languages, special properties to help in hiding information and data in multimedia [14] as described next.

### 3.2 Conditions of Arabic Letters

Arabic is one-way language, and it is read and written from right to left. Most of its letters are joined to previous and next letters; and hence take many shapes according to its position of the word. The letter "Haa" is showing different shapes as in Figure 1.

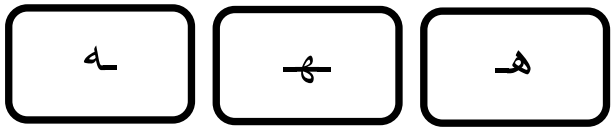

**Figure** 1 This show the letter "Haa" at the beginning, at the middle and at the end of the word.

Another feature of Arabic text is found in dots. More than half of the 26-Arabic letters are having dots, different than English only having dots in two letters, "i" and "j". Actually, some Arabic letters have single dot, two dots, or three dots, as listed in Table 1, which can be interesting feature for steganography [18].

| No. of dots | Letters |
|---|---|
| Not dotted | أ , ح , د, ر, س , ص , ط , , ع , ك , ل , م , ه , و |
| Single dot | ب , ج , خ , ذ , ز , ض , ظ , غ , ف , ن |
| Two dots | ت , ق , ي |
| 3 dots | ث , ش |

**Table** 1 shows the Arabic language letters which contains dots and those with two dots or three dots

### 3.3 Kashida

Kashida (-) connects letters as extension [18]. It is one of Arabic language special writing properties, which helps in letter extension, without affecting the meaning or letter shape [19]. We can add two Kashidas in middle letters of the word, adding 2 bits inside the word. If the number of letters of word, for example, is N, so we can add n-1. If we have (J) number of words, we can add (N-1)*J.

### 3.4 Initialize Database

Database was built of group of Prophet Hadith (sayings and traditions), which we tested them through steganography, our source is from website: (https://goo.gl/TdtEHi). And we removed all comas and full stops, in addition to the excess spaces between words and Kashida present in plain text. We used Excel program for building database to link it to visual basic, assuming that database contains 40 Hadith see Figure 2.

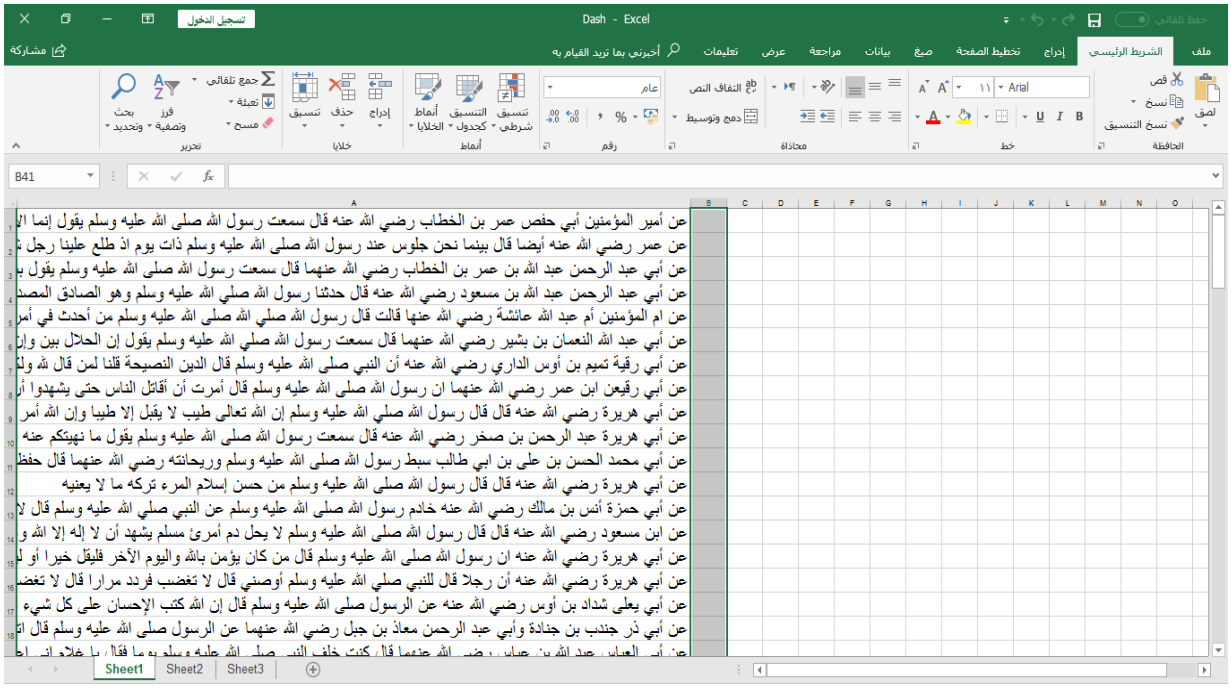


**Figure** 2 This is the main database which containing the texts

### 3.5 User Interface Design

We have used visual basic program to design user interface, then we linked it to database, as we assumed that target key is 32 bits; we randomly generate the target key, and after that it generates shares by counting zeros and ones in the target key. Number of zeros is

equal to shares. Table 2 shows method of generating the shares from the target key. We used MSflexGrid, which is one of visual basic tools, to link with databases.

For example, we have (TK: 0100), so we have group of numbers (shares) which are: (0101) , (0110) , (1100) , according to number of zeros in the target key.

| Target Key | | 0100 (N=6) | | | |
|---|---|---|---|---|---|
| 1-bits | Shares1 | 1 | 1 | 0 | 0 |
| | Shares2 | 0 | 1 | 1 | 0 |
| | Shares3 | 0 | 1 | 0 | 1 |
| | 1's | 1 | | | |
| | 0's | 3 | | | |
| | Shares | 3 | | | |

**Table** 2 shows an example of generation of shares using 1 bit method.

If, for example, we have the target key (01011101110011010110110000110101) # 5DCD6C35, total number of the shares will be 14, that is n=14. So we will have the following shares according to the next table:

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Shares (0) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5DCD6C35 |
| Shares (1) | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | DDCD6C35 |
| Shares (2) | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 7DCD6C35 |
| Shares (3) | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5FCD6C35 |
| Shares (4) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DED6C35 |
| Shares (5) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DDD6C35 |
| Shares (6) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCF6C35 |
| Shares (7) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCF6C35 |
| Shares (8) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCDEC35 |
| Shares (9) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCD6E35 |
| Shares (10) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCD6D35 |
| Shares (11) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCD6CB5 |
| Shares (12) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 5DCD6C75 |
| Shares (13) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 5DCD6C3D |
| Shares (14) | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 5DCD6C37 |

**Table** 3 shows an example of generation of shares using 1 bit method from Tk in 32 bits

See Figure 3 for user interface designed by visual basic V 6 and linked with the database (benchmark) in Excel

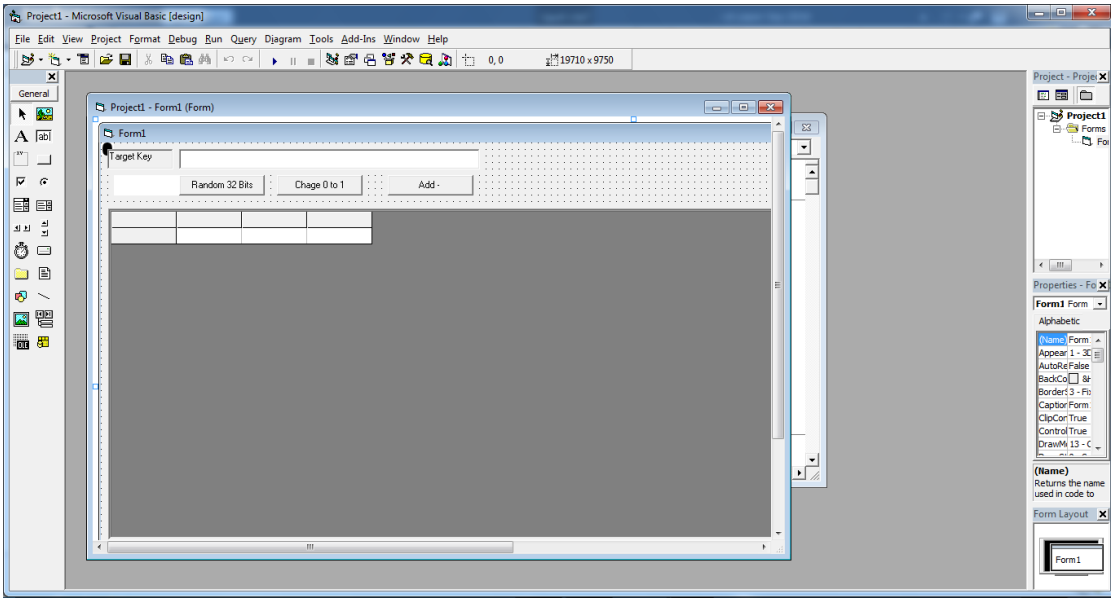


**Figure** 3 shows the elemental initial user interface of the program

## 4. Preserving Secret Shares via Text Steganography

In this section we will explain the aim of our proposal in addition to the algorithms of the program operations methods; and then the comparisons and results.

### 4.1 Proposed Model

The aim is to utilize steganography in Arabic text to improve capacity of adding Kashida in the text preserving security. We hide secret shares bits from the target key using Kashidas. If the bit equals 1 we add Kashida otherwise not. Adding Kashida is performed according to letters condition, i.e. if accepting Kashida or not. Notes that keeping spaces between words don`t add any Kashida similar to the end or the beginning of the word. In fact, we look at all possibilities of adding Kashida and try utilizing them within all the Arabic-text. Figure 4 is an example showing adding Kashida possibility to text (بسم الله الرحمن الرحيم).


**Figure** 4 Shows probabilities of addition of Kashidas to a text, and show the text with added Kashidas.

**First algorithm**: (secret sharing)
01: generating the target key TK
02: choosing k value
03: changing all zeros to one, changing one (zero) each time
04: choosing suitable shares according to k value.
05: comparing shares with the target key TK
06: if the bits in shares equal to target key, this share is correct
07: if not, change the shares and go to 04
08: if all share finish
09: end

**The second algorithm:** (merge secret shearing and steganography algorithm)
01: input: text (shares) or message (text cover)
02: output: text steganography
03: word reading
04: letters examination
05: if letter suitable for Kashida, add Kashida
06: if not, go to 05
07: if space after letter
08: read the next word
09: end

Figure 5 is a graphic representing how program works. It generates secret numbers (shares) from target key randomly (random number generator RNG)], which is the generation of

group or a series of numbers or strings out of an unknown behavior function that cannot be predicted. There are many technically accredited and safe methods of generation of numbers through encrypted method as Yaro, Fortuna (PRNG) algorithms, and others. Random numbers are used in many fields as probabilities, game programming, reverse engineering and communication protocols as TCP.

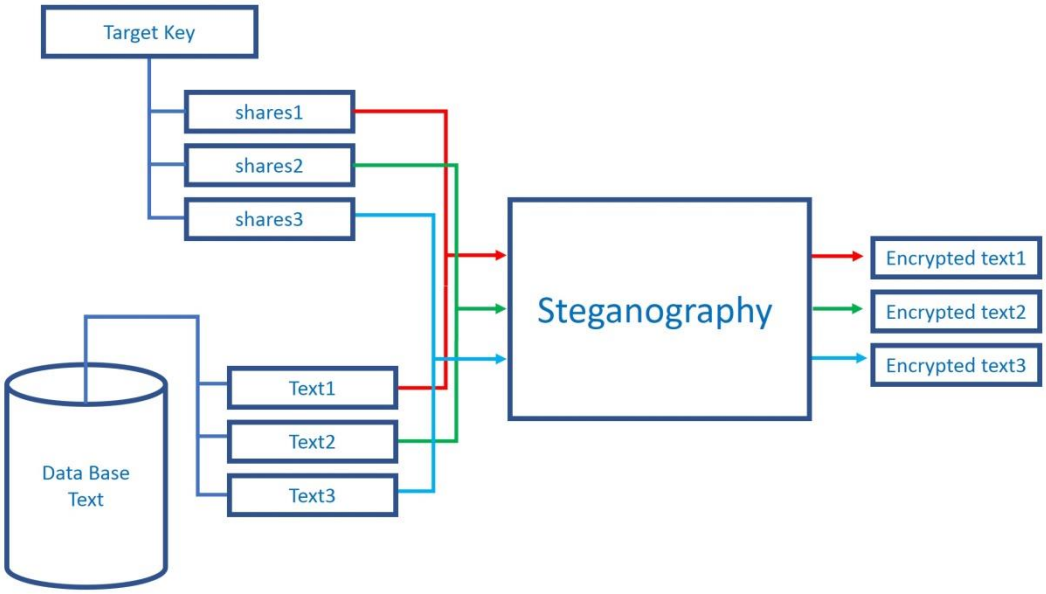


**Figure** 5 Proposed method to merge counting based secret shares with steganography

As we assumed database contains 40 Hadith, so that if the target key is zeros (TK:000000000000000) which is the worst case of having 32 shares. But assume that target key is zeros and ones, we will have group of shares, and each number has a Hadith so as to hide it inside the text using the Kashidas, so that if the bit equal to 1 and the letter suitable for adding Kashidas and not at the word end, Kashida will be added. But if bit equal 0, no Kashida will be added and will go to the next letter or next word. So we will have group of texts or new Hadith after adding the Kashida in it, as figure 5.

See figure 6 the total number of shares which we extracted is A, and the number of shares which is actually used is N; and as K belong to N; and N belong to A (K out of N).

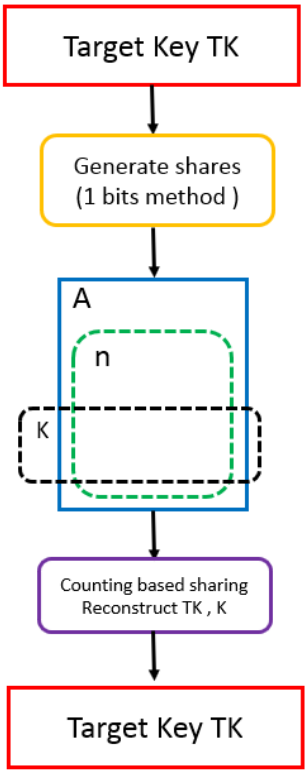


**Figure** 6 This scheme [1] show counting based secret shares sharing from stage of generating the shares

### 4.2 Benchmark Preparation

We must make sure that the text is ordinary without any formatting, and also there is no Kashida in it, and no extra spaces between words. We will choose Hadith from the book of 40 Nawawi Prophet Hadith, to do our study and it will be our benchmark. Next table will show text case in addition to number of letters and words in it, in addition to the spaces in each Hadith and also the maximum Kashida adding to the text. We choose 40 Steganography cover (Hadith) according to the target key and We will use a percentage equation by dividing total secret shares by total letters in a Hadith*100, as shown in table 4.

| Stego # | Steganography cover (Hadith) | Number of words | Number of letters | No. of Maximum Kashidas | Percentage of secret shares to letters number |
|---|---|---|---|---|---|
| 1 | الأعمال بالنيات | 52 | 209 | 107 | 51.196 |
| 2 | حديث جبريل | 183 | 753 | 367 | 48.7383798 |
| 3 | بني الاسلام على خمس | 44 | 171 | 81 | 47.3684211 |
| 4 | عمل اهل الجنة وعمل اهل النار | 106 | 423 | 226 | 53.427896 |
| 5 | من عمل عملا ليس عليه أمرنا فهو رد | 41 | 142 | 72 | 50.7042254 |
| 6 | الحلال بين والحرام بين | 83 | 321 | 162 | 50.4672897 |

| | | | | | |
|---|---|---|---|---|---|
| 7 | الدين النصيحة | 28 | 119 | 62 | 52.1008403 |
| 8 | أمرت أن اقاتل الناس حتى يشهدوا ان لا اله الا الله | 49 | 201 | 92 | 45.7711443 |
| 9 | إن الله تعالى طيب لا يقبل إلا طيبا | 77 | 303 | 147 | 48.5148515 |
| 10 | ما نهيتكم عنه فاجتنبوه | 40 | 167 | 94 | 56.2874251 |
| 11 | دع مايريبك | 36 | 127 | 67 | 52.7559055 |
| 12 | من حسن اسلام المرء | 22 | 77 | 38 | 49.3506494 |
| 13 | لا يؤمن أحدكم حتى يحب لأخيه | 32 | 113 | 60 | 53.0973451 |
| 14 | لا يحل دم امرئ | 39 | 155 | 73 | 47.0967742 |
| 15 | من كان يؤمن بالله واليوم الآخر فليقل خيرا او ليصمت | 40 | 161 | 77 | 47.826087 |
| 16 | لاتغضب | 23 | 81 | 39 | 48.1481481 |
| 17 | ان الله كتب الإحسان على كل شيء | 36 | 148 | 74 | 50 |
| 18 | اتق الله حيثما كنت | 34 | 132 | 70 | 53.030303 |
| 19 | احفظ الله يحفظك | 116 | 451 | 228 | 50.5543237 |
| 20 | اذا لم تستح | 33 | 124 | 57 | 45.9677419 |
| 21 | آمنت بالله ثم استقم | 34 | 121 | 59 | 48.7603306 |
| 22 | أرأيت أذا صليت المكتوبات | 41 | 167 | 77 | 46.1077844 |
| 23 | الطهور شطر الإيمان | 55 | 235 | 116 | 49.3617021 |
| 24 | إني حرمت الظلم على نفسي | 178 | 722 | 376 | 52.0775623 |
| 25 | أهل الدثور بالأجور | 100 | 395 | 193 | 48.8607595 |
| 26 | كل سلامي من الناس عليه صدقة | 55 | 220 | 118 | 53.6363636 |
| 27 | البر حسن الخلق | 27 | 99 | 51 | 51.5151515 |
| 28 | جئت تسأل عن البر | 45 | 174 | 84 | 48.2758621 |
| 29 | اياكم ومحدثات الأمور | 65 | 297 | 155 | 52.1885522 |
| 30 | ألا أدلك على أبواب الخير | 142 | 569 | 285 | 50.0878735 |
| 31 | ان الله فرض فرائض فلا تضيعوها | 43 | 171 | 91 | 53.2163743 |
| 32 | ازهد في الدنيا يحبك الله | 44 | 170 | 85 | 50 |
| 33 | لاضرر ولا ضرار | 24 | 80 | 37 | 46.25 |
| 34 | لو يعطى الناس بدعواهم | 31 | 125 | 60 | 48 |
| 35 | من رأى منكم منكرا | 33 | 132 | 71 | 53.7878788 |
| 36 | كل المسلم على المسلم حرام | 69 | 178 | 139 | 78.0898876 |
| 37 | نفس عن مؤمن كربة | 103 | 397 | 213 | 53.6523929 |
| 38 | إن الله تعالى كتب الحسنات والسيئات | 74 | 295 | 168 | 56.9491525 |
| 39 | من عادى لي وليا فقد آذنته بالحرب | 69 | 270 | 136 | 50.3703704 |
| 40 | إن الله تجاوز لي عن أمتي الخطأ | 25 | 96 | 47 | 48.9583333 |

**Table** 4 Steganography cover (Hadith) capability: number of letters, words and spaces and studies the maximum number of Kashidas that could be added

Take the Steganography cover 33 as an example:

عن أبي سعيد سعد بن مالك بن سنان الخدري رضي الله عنه أن رسول الله صلى الله عليه وسلم قال لا ضرر ولا ضرار

Kashida will be added as shown in figure 7. Letters in Steganography cover is 80, maximum allowed Kashida to be added is 37, Kashida to letters ratio is 46.25.

possibility of chachida

Original text

عن أبي سعيد سعد بن مالك بن سنان الخدري رضي الله عنه أن رسول الله صلى الله عليه وسلم قال لا ضرر ولا ضرار

Text after add kashida

عن أبي سعيد سعد بن مالك بن سنان الخدري رضي الله عنه أن رسول الله صلى الله عليه وسلم قال لا ضرر ولا ضرار

**Figure** 7 This shows the possibilities of adding Kashidas in each Steganography cover and the resulting text after adding Kashidas.

Observe figure 8, the ratios hiding secret shares within all Steganography cover texts. Interestingly, the percentage of cover-text 2 and 24 are giving the highest, while cover-text numbers 12, 16 and 33 are showing the worst scenarios. This proofs the concept of data-dependency which can be utilized for more capacity or more security purposes.

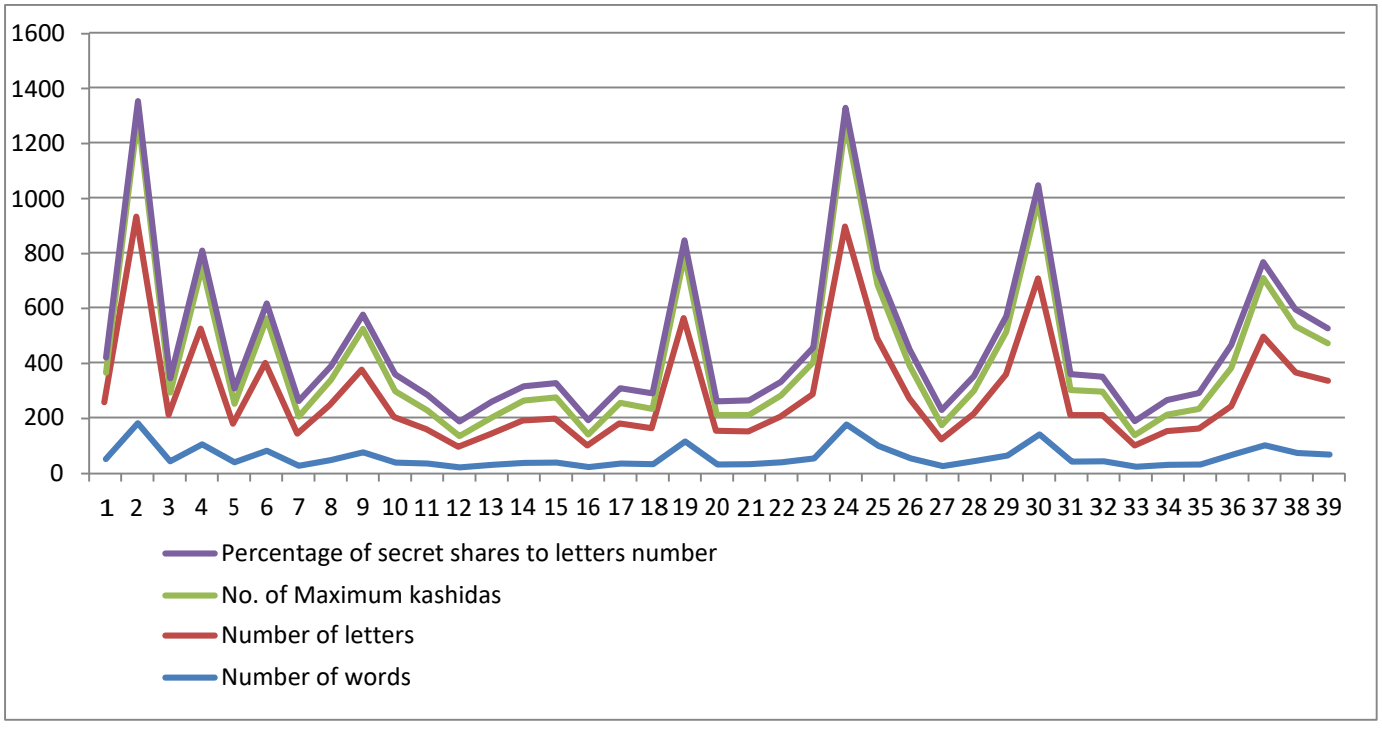


**Figure** 8 Show number of letters and words for each Steganography cover (Hadith) , and the maximum number of Kashida could be added to it and ratios for all Steganography cover.

## 5. Conclusion

We introduce in this paper a new merged method of hiding secret shares within the Arabic text serving counting based secret sharing systems. The aim is that no eavesdropper can know the secret shares numbers. Security of the method is enhanced in order to exchange information between users. The study compared hiding within deferent bench mark stego-covers as standardized within Hadiths. The results have been analyzed showing the number of letters and words for each Steganography cover (Hadith) as well as expressing the maximum number of Kashida that could be added to it and ratios for all Steganography covers.

In the future, we plan to show this method and its requiring affect increasing the target key from 32 bit to 64 bit, or even 128 bit, as practical security needs. It was found that most Stego-Cover Hadiths can contain more than 32 letters allowing for more capacity of secret shares. We can also improve the margining strategy to add Kashidas in some locations and leave others, intending to have more ambiguity as higher security. This idea can assume Kashida locations as one location on and the other off, i.e. consider one location and leave one, which may be giving attractive results. Interestingly, most of the Stego-cover Hadiths will accommodate this strategy too since many are having empty locations available for any additions. These ideas can be the ignition of increasing the size of the target key to enable making counting based secret sharing more practical and complex for any eavesdropper to break and know the hidden information.

**References:**

*[1] Adnan Gutub, Nouf Al-Juaid, and Esam Khan. "Counting-based secret sharing technique for multimedia applications." Multimedia Tools and Applications (2017): 1-29.*

*[2] Adnan Gutub, Alexandre Ferreira Tenca. "Efficient scalable VLSI architecture for Montgomery inversion in GF (p)." Integration, the VLSI journal, 37.2 (2004): 103-120.*

*[3] Adnan Gutub. "Area flexible GF (2k) elliptic curve cryptography coprocessor." International Arab Journal of Information Technology (IAJIT) 4.1 (2007): 1-10.*

*[4] Esraa Ahmadoh, and Adnan Gutub. "Utilization of two diacritics for Arabic text steganography to enhance performance." Lecture Notes on Information Theory Vol 3.1 (2015).*

*[5] Adnan Gutub. "High speed hardware architecture to compute galois fields GF (p) montgomery inversion with scalability features." IET Computers & Digital Techniques 1.4 (2007): 389-396.*

*[6] Adi Shamir. "How to share a secret." Communications of the ACM 22.11 (1979): 612-613.*

*[7] Blakley, George Robert. "Safeguarding cryptographic keys." Proceedings of the national computer conference. Vol. 48. 1979.*

*[8] Kai Wang, Xukai Zou, and Yan Sui. "A multiple secret sharing scheme based on matrix projection." Computer Software and Applications Conference, 2009. COMPSAC'09. 33rd Annual IEEE International. Vol. 1. IEEE, 2009.*

*[9] Jessica Fridrich, Steganography in digital media: principles, algorithms, and applications. Cambridge University Press, 2009.*

*[10] Adnan Gutub. "Efficient utilization of scalable multipliers in parallel to compute GF (p) elliptic curve cryptographic operations." Kuwait Journal of Science & Engineering (KJSE), December 2007 34.2 (2007): 165-182.*

*[11] Ahmed Al-Nazer and Adnan Gutub, "Exploit Kashida Adding to Arabic e-Text for High Capacity Steganography", International Workshop on Frontiers of Information Assurance & Security (FIAS 2009) in conjunction with the IEEE 3rd International Conference on Network & System Security (NSS 2009), Gold Coast, Queensland, AUSTRALIA, 19-21 October 2009.*

*[12] Safia Al-Nofaie, Manal Fattani, Adnan Gutub, "Capacity Improved Arabic Text Steganography Technique Utilizing 'Kashida' with Whitespaces", The 3rd International Conference on Mathematical Sciences and Computer Engineering (ICMSCE2016), pp. 38-44, Langkawi, Malaysia, February 2016.*

*[13] Fahd Al-Haidari, Adnan Gutub, Khalid Al-Kahsah, and Jameel Hamodi, "Improving Security and Capacity for Arabic Text Steganography Using 'Kashida' Extensions", AICCSA-2009 - The 7th ACS/IEEE International Conference on Computer Systems and Applications, Pages: 396-399, Rabat, Morocco, 10-13 May 2009.*

*[14] Ammar Odeh , Khaled Elleithy, and Miad Faezipour. "Steganography in Arabic text using Kashida variation algorithm (KVA)." Systems, Applications and Technology Conference (LISAT), 2013 IEEE Long Island. IEEE, 2013.*

*[15] Tayana Morkel , Jan HP Eloff, and Martin S. Olivier. "An overview of image steganography." ISSA. 2005.*

*[16] Abbas Cheddad, et al. "Digital image steganography: Survey and analysis of current methods." Signal processing 90.3 (2010): 727-752.*

*[17] https://informatic-ar.com/random_numbers , 25/5/2018 20:30 P.M.*

*[18] Adnan Gutub and Manal Fattani. "A Novel Arabic Text Steganography Method Using Letter Points and Extensions", WASET International Conference on Computer, Information and Systems Science and Engineering (ICCISSE), Pages: 28-31, Vienna, Austria, May 25-27, 2007.*

*[19] Mohammed Aabed, Sameh Awaideh, Abdul-Rahman Elshafei, and Adnan Gutub, "Arabic Diacritics Based Steganography", IEEE International Conference on Signal Processing and Communications (ICSPC 2007), Pages: 756-759, Dubai, UAE, 24-27 November 2007.*

**ABOUT AUTHORS:**

**Khaled Aydh Alasiri** is currently a graduate student, pursuing Master of Sciences (MS) degree in Computer Engineering, offered by College of Computers and Information Systems, at Umm Al Qura University (UQU) under the umbrella of Ministry of Higher Education. His MS program at UQU is specialized in Information Security, Saudi Arabia.

**Prof. Adnan Abdul-Aziz Gutub** is ranked as Full Professor in Computer Engineering specialized in Information and Computer Security within College of Computers and Information Systems at Umm Al-Qura University (UQU). He has been working as the Vice Dean of the Custodian of the Two Holy Mosques Institute of the Hajj & OmrahResearch,

Known publicly as Hajj Research Institute (HRI), within (UQU), Makkah Al-Mukarramah, all Muslims religious Holy City located within the Kingdom of Saudi Arabia.

Adnan's academic experience in Computer Engineering was gained from his previous long-time work as Associate Professor, Assistant Professor, Lecturer, and Graduate Assistant, all in Computer Engineering at King Fahd University of Petroleum and Minerals (KFUPM) in Dhahran, Saudi Arabia. He received his Ph.D. degree (2002) in Electrical and Computer Engineering from Oregon State University, USA. He had his BS in Electrical Engineering and MS in Computer Engineering both from KFUPM, Saudi Arabia.

Adnan's research work can be observed through his 70+ publications (international journals and conferences) as well as his 5 US patents registered officially by USPTO. His main research interests involved optimizing, modeling, simulating, and synthesizing VLSI hardware for crypto and security computer arithmetic operations. He worked on designing efficient integrated circuits for the Montgomery inverse computation in different finite fields. He has some work in modeling architectures for RSA and elliptic curve crypto operations. His interest in computer security also involved steganography focusing on image based steganography and Arabic text steganography. Adnan's research interest in computing and information technology have been broaden to also relate to smart crowd management and intelligent transportation engineering systems because of the involvement in Hajj and Omrah Research at UQU - Makkah.

Administratively, Adnan Gutub filled many executive and managerial academic positions at KFUPM as well as UQU. At KFUPM - Dhahran, he had the experience of chairing the Computer Engineering department (COE) for five years until moving to Makkah in 2010. Then, at UQU - Makkah, Adnan Chaired the Information Systems Department at the College of Computers & Information Systems followed by his leadership of the Center of Research Excellence in Hajj and Omrah (HajjCoRE) serving as HajjCoRE director for around 3-years until the end of 2013. Then, he was assigned his last position (until March 2016) as the Vice Dean of HRI, i.e. the Custodian of the Two Holy Mosques Institute of the Hajj & Omrah Research.