Sudoku and Image Security

Project Final Report

Alice H. Becker

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213
May 1, 2013

Abstract

This project will investigate the different ways the Sudoku puzzle can be used to secure information stored in an image. The number of solutions to the Sudoku puzzle is very large, allowing it to be used as a security tool resistant to brute-force attacks. Additionally, Sudoku's rules of not have repeated digits in rows, columns, and squares allow it to be used to detect tampering attempts. The project will look into the following areas of use: verification codes and image scrambling. Sudoku can be used as a verification code resistant to malicious network tampering attacks that can detect alterations in the image it is securing. Also, Sudoku puzzles can be used to scramble images and maintain a map between original and scrambled image.

Introduction

The goal of this paper is to study the different ways the Sudoku puzzle can be used in image security. First, this paper will give an overview of the Sudoku puzzle and look into its special properties. Then, two applications of the Sudoku Puzzle in image security will be discussed. The two areas of application are: image authentication and image scrambling. The goals, use of Sudoku, implementation, and findings of each application will be examined. Finally, conclusions on the use of Sudoku in image security will be drawn from all these findings together.

About the Sudoku Puzzle

Sudoku is a logic puzzle traditionally played in a 9x9 grid subdivided into nine 3x3 blocks [1]. The objective of the puzzle is to fill each row, column, and block with exactly one of each of the digits from 1 to 9. A Sudoku puzzle is typically played with a grid, where some of the entries have already been filled [2]. One must then logically deduce the remaining entries to complete the puzzle in a valid manner. Below in figure 1 we provide a figure of a Sudoku puzzle solution:

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Figure 1: Example of Sudoku puzzle solution (pre-filled units in lighter color) [2]

History

The origins of the Sudoku puzzle begin with a game called Latin Square, where one aims to fill an NxN grid by filling each row and column with exactly one of each of N different symbols. In 1979, Garns added the block condition to the Latin Square and Dell Magazines published this new puzzle, calling it “Number Place”. Sudoku then became popular in Japan, where its name means, “the digits must be single.” Sudoku finally rose to worldwide popularity in 2005.

Special Properties of Sudoku

The Sudoku puzzle has many special properties that make it a useful tool for image security. The first property is quite simple; Sudoku is a 2-Dimensional grid, so making it suitable for images, since they can be represented as a 2-D matrix of pixel values. Additionally, the traditional Sudoku puzzle can be easily scaled to attain any dimension. For scaling, we can make the puzzle an N^2xN^2 grid, subdivided into NxN blocks, that is to be filled with digits 1 through N^2. We can also simply tile the Sudoku puzzle and pieces of it to attain any size of grid. Perhaps,

the most important property Sudoku provides for image security is its large number of solutions. In 2005, Felgenhauer and Jarvis found that the number of 9x9 Sudoku puzzle solutions is approximately 6.671 x 10^{21}. Another special property is what we refer to the Sudoku Rule: the requirement that each row, column, and block contain exactly one of each of the possible digits. [3]

Image Authentication

In 2009, Wu and Ren [2] proposed an image authentication scheme using Chaotic Map and Sudoku. This method accurately detects if an image has been tampered during delivery, using an embedded verification code.

Goals

The main goal of this proposed method is to verify an image against tampering that may happen while its being sent from sender to receiver. This helps secure an image against malicious attacks, such as the forging of data. The method should also accurately detect tampered areas. The embedding of the verification code should not be noticeable to the naked eye. The method as a whole should be efficient. Additionally, this method should not require the message receiver to hold the original image or verification codes.

How it uses Sudoku

The Wu and Ren method uses a valid Sudoku solution as an embedded verification code. The method uses a solution to a 4 by 4 Sudoku puzzle and, then tiles it to the size of the image. To check if the image is still valid, the receiver must simply check the validity of the Sudoku puzzle. If in any place, the Sudoku rule breaks, we know that that region of the image has been tampered with.

Implementation

The proposed method has two phases. The first is the verification code-embedding phase, which happens on the sender side before the image is delivered to the receiver. The second phase is the tampering detection, which happens on the receiver side after the image may have been exposed to tampering attacks. Before we go into each of the phases, we will discuss the chaotic system that both phases use.

Chaotic system

First, Wu and Ren define Chaos to be "a similar random phenomenon on dynamical behavior." Although it's hard to predict its variation, most of the randomness is regular and generally non-chaotic. We can describe a chaotic system with the following mathematical expression:

$$x_{n+1} = \mu\, x_n\, (1 - x_n), where\ \mu\ \in [1, 4]\ and\ x_n \in [0, 1]$$

These systems are highly susceptible to changes in initial conditions μ and x_1. Due to this, these systems can be used to produce very different, uncorrelated sequences of numbers. Wu and Ren use such a chaotic system to generate a secure 2-D random map, M, given initial parameters μ and x_1. Using the equation above, an entry is created in a sequence for each pixel in the image we are to secure. The values are then normalized and rounded to be an integer within the range of our pixel values: 0 to 255. Lastly, the sequence is broken up into the rows of the 2-D map.

Embedding phase

This phase takes the chaotic map M described above and the image I. To embed the verification code, each pixel value of I is slightly modified. Below we provide detailed steps of the embedding algorithm:

1. Generate an arbitrary 4x4 Sudoku grid that is subdivided into four 2x2 blocks and is fully filled with the digits 1 through 4. Tile the grid to the size of the cover image I so that we generate a Sudoku map S.
2. Starting with the first pixel in I, repeat the following two steps until all of image I has been processed.
3. Take the i-th pixels of I and M, which we label p_i and m_i, respectively. Calculate: $A = 1 + (p_i \oplus m_i) mod\ 4$. We x-or p_i and m_i and, then, perform a modulus 4 operation and add 1, so that the resulting value is a digit between 1 and 4.
4. Now, we compare A to s_i, the i-th entry in the Sudoku map. If they are equal, we do nothing to p_i. Otherwise, we find p_j: a neighbor of p_i that satisfies the following property: $s_i = 1 + (p_j \oplus m_i) mod\ 4$. Set the original pixel p_i to p_j. By choosing a nearby pixel, which generally will have a similar pixel value, we reduce the level of modification we make to the image.

After we perform these steps, we obtain a new image I', that looks just like I to the naked eye. Now, this image can be safely transmitted from sender to receiver without the risk of undergoing tampering undetected.

Tampering detection phase

The tampering detection phase happens after the transmission of image I' on the receiver's end. This phase uses the same chaotic map M to derive the Sudoku map S from I using the same equation as before.

1. Generate the chaotic map M, using the same parameters μ and x_1, as before.
2. Starting with the first pixel in I', repeat the following until all of image I' has been processed. Calculate $s_i = 1 + (p_i \oplus m_i) mod\ 4$ to find each entry in the Sudoku map S.
3. Divide S into 4x4 blocks that do not overlap.
4. Check if each of these blocks follows the Sudoku rule. If any do not, mark the block as a fault. Because the blocks are quite small this allows us to identify with great accuracy the areas that have been tampered.
5. If all blocks pass verification, then the image has not been tampered.

Throughout this process, the only shared state between sender and receiver are the parameters μ and x_1. This allows the receiver to verify the image without needing to have access to the whole initial Sudoku grid from the sender.

Findings

The experimental results of the Wu and Ren method state that it can accurately identified tampering in any region of several images of different sizes (see figure 4). The visual quality of the embedded image of just under 44db is good enough so that the human eye cannot detect the embedding, as you can see in figure 3. The method is reasonably efficient in both embedding and detection time, depending on the size of the image, amongst other things. The method also succeeds at not needing to pass original images or verification codes to the receiver.

Figure 2: Original Cover Image (left) vs. Embedded Image (right) [2]

Figure 3: Tampered Image (left) and. Tamper Detection (right) [2]

A suggestion for improving security on this method would be to create a Sudoku map by tiling many different 4x4 Sudoku puzzles, instead of repeating the same one. This could prevent any patterns in the puzzle from being discovered. However, it is worth to note that the chaotic map already obscures these patterns to some extent. Additionally, this change in implementation could however come at an increase time-complexity cost in the process of building many puzzles.

Image Scrambling

In 2011, Zou, Tian, Xia, and Song [3] introduced an image scrambling method that uses the Sudoku puzzle. This method securely scrambles images well, making them appear to contain no information.

Goals

The method should scramble the image making it appear to contain no information. The method should provide good security. The image should not be restored without the correct keys.

How it uses Sudoku

The proposed method uses pairs of Sudoku puzzles to map original and scrambled images. The method takes a pair of Sudoku puzzles and modifies it so there is a 1-1 relationship between the digits of the puzzles. It adds the digits corresponding to column number in front each of the digits for the puzzle corresponding to the original image. It does the same with row numbers to the puzzle for the scrambled image. It then scrambles the image by taking a pixel in the original image, locating the digit entry in the Sudoku puzzle in the same place as the pixel, and moving it to the corresponding digit in the other puzzle. The proposed method takes advantage of the Sudoku rule to create this 1-to-1 correspondence between puzzles. The method also benefits from the large number of Sudoku solutions to provide security against unscrambling attempts.

Implementation

The scrambling algorithm for this method is divided into four parts: Sudoku pair selection, Sudoku pair preparation, image marking and mapping, and bit scrambling. We will additionally discuss how to unscramble the image.

Sudoku Pair Selection

The first is the Sudoku puzzle pair selection. In this step, one must simply make pairs of Sudoku puzzles. The pairs can be of any size and there can be any number of pairs. Having many different pairs can be beneficial to improve the security of the method.

Sudoku Pair Preparation

In the second part, we need to establish 1-to-1 relations between the puzzles in each pair. We do this by adding a prefix to each of the digit entries in order to make them all unique. This way there is exactly one of each entry in the first puzzle for each entry in the second. We modify the entries in the first puzzle with the formula NewValue = OldValue + Column x 10^{Digits}, where digits is the number of digits in the puzzle. Note that this simply adds a column prefix to each entry. We modify the second puzzle with the formula NewValue = OldValue + Row x 10^{Digits}, where digits is the number of digits in the puzzle. Note that this formula simply adds a row prefix to each entry. Below we provide and image exemplifying this process:

4	2	1	3
3	1	2	4
2	3	4	1
1	4	3	2

Original Sudoku Puzzle 1

1	2	3	4
4	3	1	2
2	1	4	3
3	4	2	1

Original Sudoku Puzzle 2

14	22	31	43
13	21	32	44
12	23	34	41
11	24	33	42

Changed Sudoku Puzzle 1

11	12	13	14
24	23	21	22
32	31	34	33
43	44	42	41

Changed Sudoku Puzzle 2

Figure 4: Sudoku Pair Preparation [3]

Image Marking and Mapping

The third part uses these prepared pairs of Sudoku puzzles to establish a relation between the original image and the scrambled one. This part is sub-divided into two sub-parts: block scrambling and sub-block scrambling.

Block Scrambling

In block scrambling, we use the Sudoku pairs to scramble blocks of the same size in the original image. We use the first Sudoku puzzle in a pair to mark the pixel positions of the original image. We place that pixel in the equivalent entry for the second Sudoku puzzle. The following steps and figure explain the process in more detail:

1. For the i-th pixel in the original image block p_i, we take the i-th entry in the first Sudoku puzzle a_i.
2. We then locate the entry in the second Sudoku puzzle such that $b_j = a_i$.
3. We set the j-th pixel in the scrambled image to be $s_j = p_i$.
4. We repeat these steps until all pixels in the block have been processed.

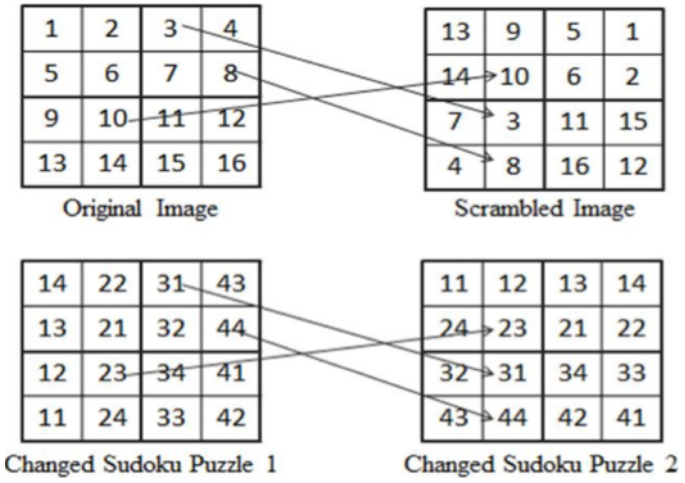

Figure 5: Block scrambling using Sudoku pairs [3]

Sub-block Scrambling

In sub-block scrambling, we take each scrambled block and break it up into smaller sub-blocks. We then repeat the same process from block scrambling with these smaller blocks.

Bit Scrambling

After the third part, the image is not sufficiently scrambled and still appears to show some information in the scrambled image and in the histogram. For this reason, we need the fourth part: bit scrambling. In this part, we take the bits of the image and modify them so we can treat them like a 2-D grid. To do this we first flatten the scrambled image into a 1-D grid by connecting rows to each other. The grid's length is P, where P is the number of pixels. For each pixel in the grid, we create a column containing its binary representation, giving us a 2-D grid of size 8xP. There are at most 8 rows because pixel values range between 0 and 255. We reshape the grid into a square of size MxM, where M is the floor of $\sqrt{8\times P}$. We do this reshaping by going through entries row-by-row and adding them to the square grid. We then perform the same puzzle pair scrambling process to this grid and obtain new pixel values in the image. Please refer to the image below for clarification:

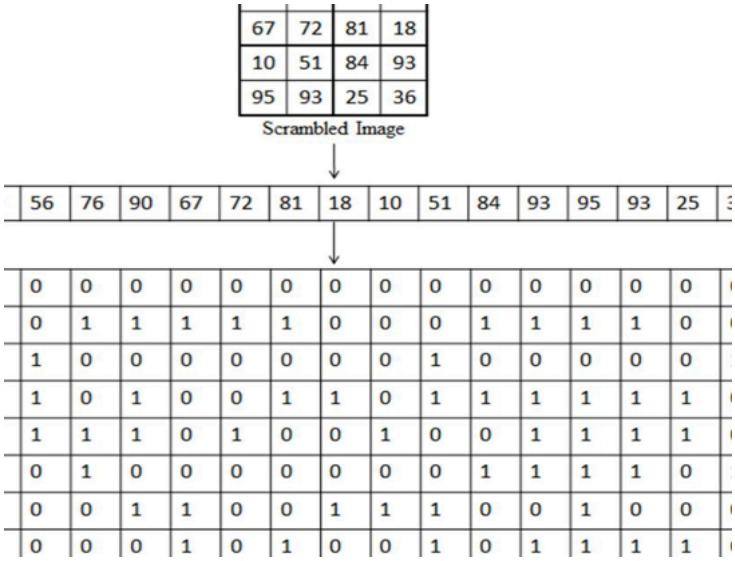

Figure 6: Bit scrambling matrix generation [3]

Unscrambling

To restore the image to its pre-scrambled form, one must simply exchange roles of prepared Sudoku puzzles and repeat the scrambling steps with the same iteration numbers. This will successfully restore the image as long as the correct sets of puzzles and iteration numbers are used.

Findings

Both the scrambled image and its histogram do not reveal any information about the original image, as seen in figure 7. The proposed method provides good image security. Attempts to restore the images without all the correct keys do not reveal the images information, as seen in figure 8.

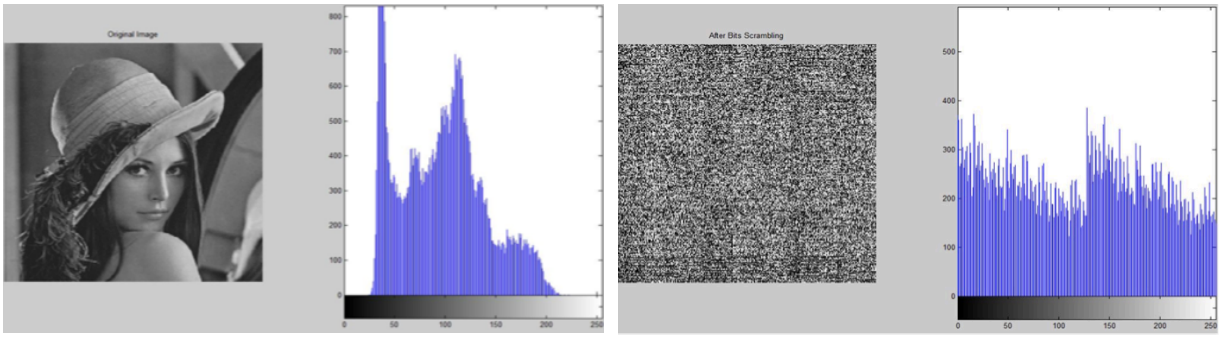

Figure 7: Original image and its histogram (left) vs. Scrambled image and its histogram (right) [3]

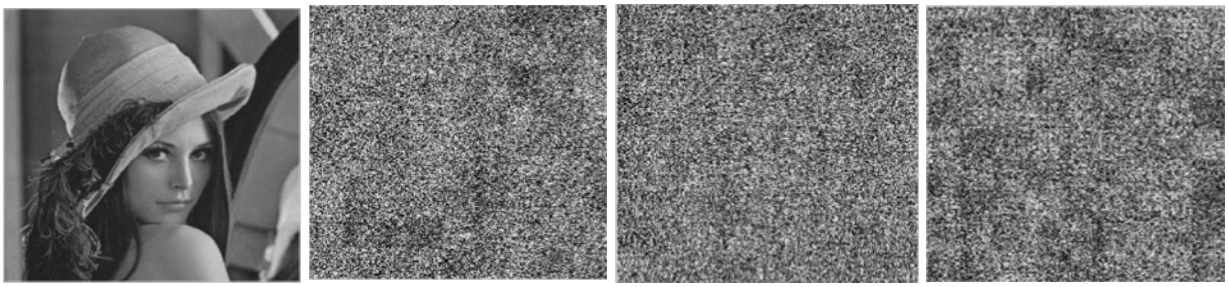

Figure 8: From left to right: image restore with iteration times and pairs, right times and wrong pairs, wrong times and right pairs, wrong times and wrong pairs [3]

Conclusions

Due to the Sudoku puzzle's many special properties, it can be a powerful tool in image security. The puzzle's simplicity makes it very flexible, leading to many potential applications of it as a security tool. Its inherent 2-dimensionality and scalability make it appropriate to be used on images of any size. Additionally, Sudoku's large number of solutions makes it resistant against random attempts to crack the security systems that use it.

More specifically, in Wu and Ren's image authentication method, a Sudoku puzzle is used as a verification code, taking advantage of the special Sudoku rule to detect tampering attempts. The Sudoku rule benefits this method by both helping accurately detect tampered regions and, also, allowing the receiver to not need the original verification codes in order to verify the image. Additionally, the method produces a decent visual quality of embedded images and can is reasonably efficient both in embedding and detection time.

The Zou, Tian, Xia, and Song image scrambling method also benefits from the Sudoku rule, as it makes it possible to derive a 1-to-1 relation between any two Sudoku puzzles. It uses Sudoku puzzles as maps between original and scrambled images, so that they may be easily converted to and from each other. Additionally, this method provides good security, as no information is revealed in the scrambled image and histogram, and random attempts to unscramble the image without the correct keys lead to no more information.

In general, the Sudoku puzzle can be used as a tool in image security that can provide good security without sacrificing performance in basic metrics. The use of the puzzle can additionally introduce new properties to the security algorithms that distinguish them from other similar methods.

References

1. Chin-Chen Chang, Yung-Chen Chou, and The Duc Kieu. "An Information Hiding Scheme Using Sudoku," *Proceedings of the Third International Conference on Innovative Computing Information and Control,* Dalian China, Jun. 2008, pp. 17.1-17.5.

2. Wen-Chuan Wu, and Guang-Ruei Ren. "A New Approach to Image Authentication Using Chaotic Map and Sudoku Puzzle" *Proceedings of the Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing,* Kyoto Japan, Sep. 2009, pp. 628-631.

3. Yang Zou, Xiaolin Tian, Shaowei Xia, and Yali Song. "A Novel Image Scrambling Algorithm Based on Sudoku Puzzle" *Proceedings of the Fourth International Congress on Image and Signal Processing, Vol. 2,* Shanghai China, Oct. 2011, pp. 737-740.