Design of Reliable and Secure Devices Realizing Shamir’s Secret Sharing

Zhen Wang, Mark Karpovsky and Lake Bu
Reliable Computing Laboratory, Boston University, Boston, USA
wang.zhen.mathworks@gmail.com, markkar@bu.edu, bulake@bu.edu

Abstract—Shamir’s secret sharing scheme is an effective way to distribute secret to a group of shareholders. The security of the unprotected sharing scheme, however, can be easily broken by cheaters or attackers who maliciously feed incorrect shares during the secret recovery stage or inject faults into hardware computing the secret. In this paper, we propose cheater detection and identification schemes based on robust and algebraic manipulation detection (AMD) codes and m-disjunct matrices (superimposed codes). We present the constructions of codes for cheater detection and identification and describe how the cheater identification problem can be related to the classic group testing algorithms based on m-disjunct matrices. Simulation and synthesis results show that the proposed architecture can improve the security level significantly even under strong cheating attack models with reasonable area and timing overheads.

Keywords—Secret Sharing, Cheater Detection, Cheater Identification, Error Control Codes, Superimposed Codes

I. INTRODUCTION

Secret sharing is widely used in cryptographic applications to share keys or other important information among multiple shareholders [1]. In a secret sharing scheme, a secret \( S \) is divided into \( L \) shares and distributed to \( L \) shareholders by a trusted dealer. The shared secret \( S \) can only be recovered when \( l \) (\( l \leq L \)) or more than \( l \) shares are available. Such a scheme is called an \((l, L)\) secret sharing.

One of the most popular secret sharing scheme is the Shamir’s secret sharing scheme proposed in [2]. The original Shamir’s secret sharing scheme is not resistant to dishonest shareholders (cheaters) and malicious attackers. A cheater can manipulate the secret by providing incorrect shares or inject errors into the secret reconstruction network. It has been shown in [3], [4] that by analyzing the fake secret and injecting errors, a cheater can completely break the security of the system and retrieve the real secret or generate a wrong secret.

Various methods have been proposed to protect secret sharing scheme in cryptographic applications against cheaters and attackers. In [3], the authors showed that the probability for the cheater to succeed can be dramatically reduced by slightly modifying the Shamir’s secret sharing scheme. In [5], the authors proposed to use the idea of sub-secrets and sub-shares distributed by the shareholders such that each shareholder is also a dealer. In [6], it was proposed to apply a one-way hashing function along with the use of arithmetic coding to detect cheating and identify the cheaters. In [7], the authors proposed a computationally secure scheme based on RSA assumption for secret sharing. A robust Shamir’s secret sharing using error detection codes to detect cheating and malicious attacks is proposed in [8].

In applications with high security requirement, it may demand to not only detect the presence of cheaters but also identify who these cheaters are. There are several methods for cheater identification described in the literature [4], [7], [9], [10]. The dealer can generate and distribute additional information, such as using check vectors and certificate vectors for each shareholder. Error-correcting codes can also be introduced for cheater detection and identification such that fake shares can be treated as errors to be detected and corrected [4], [11], [12]. In [4], the authors proposed to use more than \( l \) shareholders during the secret reconstruction thus the redundant shares can be used for cheater detection and identification in a \((l, L)\) Shamir’s secret sharing schemes.

In this paper, we propose cheater detection and identification methods based on robust and algebraic manipulation detection codes and non-adaptive group testing algorithms. Similar to [4], for cheater identification we require more than \( l \) shareholders to participate in the reconstruction of the secret for a \((l, L)\) Shamir’s secret sharing scheme. Instead of doing majority decoding as described in [4], we analyze the syndrome computed by the corresponding error detection codes and identify potential cheaters using non-adaptive group testing techniques.

Compared to other cheater detection and identification methods presented in the literature, the proposed methods in this paper can work under a much stronger error model and can deal with not only cheating but also fault injection attacks. Moreover, to our best knowledge, this is the first paper to establish the connection between the cheater identification for secret sharing and the classical group testing theory using superimposed codes.

The rest of this paper is organized as follows. In Section II, basic conceptions of Shamir’s secret sharing will be reviewed and several cheating models used in this paper will be introduced. In Section III, we discuss the architecture of the detection schemes based on robust and AMD codes. In Section IV-A, We formalize the problem of cheater identification and show how it is related to the classical research area of group testing. In Section IV-B, we briefly introduce the concept of \( m \)-separable matrix, \( m \)-disjunct matrix and superimposed codes and show how they can be used to identify cheaters in Shamir’s secret sharing scheme. In Section IV-C, we show that the presented cheater identification method using superimposed codes can tolerate undetected errors during the secret recon-
struction stage. In Section IV-D, we present and analyze the cheater identification scheme based on superimposed codes constructed from Maximum Distance Separable (MDS) codes and present the guidelines for the selection of parameters of the codes. In Section VI, we show the simulation and hardware synthesis results for the proposed cheater detection and identification methodologies.

II. BASIC CONCEPTIONS

A. Shamir’s (l, L) Secret Sharing

In a Shamir’s (l, L) secret sharing [2], [3], [4], there are L shareholders \( P = \{P_0, P_1, \cdots, P_{L-1}\} \) and a trusted dealer \( D \). We use the shareholders’ IDs \((z_0, z_1, \cdots, z_{L-1})\) to denote each participant. Secret is generated and distributed to shareholders by the dealer \( D \). Secret can be reconstructed based on Lagrange interpolation polynomial by taking any \( l \) participants’ shares \((\alpha_{i_0}, \cdots, \alpha_{i_{l-1}})\) of participants and their IDs \((z_{i_0}, \cdots, z_{i_{l-1}})\) where \( \{i_0, \cdots, i_{l-1}\} \subseteq \{0, 1, \cdots, L - 1\} \).

Shamir’s secret sharing scheme provides a method of hiding the secret such that any \( l \) or more participants would be able to reconstruct the original secret but any less than \( l \) participants would not be able to reconstruct the original secret. Assume all the computation are in \( GF(2^n) \) \((L \leq 2^n)\), in which \( n \) is the number of bits of the secret, Shamir’s secret sharing scheme consists of two algorithms [2], [4]:

1) **Share generation algorithm** Select a polynomial \( \alpha(z) = S_0 + S_1z + S_2z^2 + \cdots + S_{l-1}z^{l-1} \) where \( S_i (i \in \{0, 1, \cdots, l - 1\}) \) and \( z \) belong to \( GF(2^n) \). \( S_i (i \in \{0, 1, \cdots, l - 2\}) \) are randomly generated in \( GF(2^n) \) and \( S = S_{l-1} \) is the secret. Each shareholder with ID \( z_i \) receives a share \( \alpha(z_i) \) and we assume the IDs \( z_i \) are publicly known and unique for each shareholder. We denote \( \alpha(z_i) \) as \( \alpha_i \), and \( z_j \) as \( z_j \) for simplicity in this paper, \( z_i \neq z_j \).

2) **Secret reconstruction algorithm** For a polynomial of degree \( l - 1 \), with knowledge of at least \( l \) data points, we can construct the exact polynomial using Lagrange interpolation in \( GF(2^n) \) and thus reconstruct the secret \( S = S_{l-1} \).

In this paper, we choose \( S_{l-1} \) as the secret for simplicity, and the equation for secret reconstruction is as follows:

\[
S = S_{l-1} = \frac{\sum_{i=0}^{l-1} \alpha_i}{\prod_{j=0,j \neq i}^{l-1} (z_i + z_j)}, \tag{1}
\]

and equivalently

\[
S = c_0 \alpha_0 + c_1 \alpha_1 + \cdots + c_{l-1} \alpha_{l-1}, \tag{2}
\]

where \( c_i = \prod_{j=0,j \neq i}^{l-1} (z_i + z_j)^{-1} \). For a shareholder \( z_i \), we can represent the cheating by replacing \( \alpha_i \) by \( \tilde{\alpha}_i = \alpha_i + e_i \), where \( e_i \) is the error injected by \( z_i \). A cheater can inject errors into his share and generate a fake secret \( \tilde{S} = S + e \) and he can precisely control the error \( e \) when the IDs \( z_i \) are publicly known.

B. Cheating and Fault Injection Models

For Shamir’s secret sharing, we assume that the dealer is honest and the shares distributed to the shareholders are not distorted. According to the definition of Shamir’s secret sharing and the description above, any of these \( l \) shareholders can distort the secret by submitting a false share.

In order to emulate the real cheating situations and estimate the security level of the proposed schemes, we consider four types of cheating models in this paper.

- **Type 1** Less than \( l \) shareholders are cheating and the output of the secret reconstruction module is hidden from the participants;
- **Type 2** Less than \( l \) shareholders are cheating but there is a feedback from the output such that the cheaters can get information of the secret to help them to generate a fake secret at the next round;
- **Type 3** All of the \( l \) shareholders are cheating and the output of the secret reconstruction module is hidden from the participants;
- **Type 4** All of the \( l \) shareholders are cheating and there is a feedback from the output such that the cheaters can gain some knowledge of the secret to help them to cheat in the next round.

In general, the more information the cheaters can gain, the stronger the resulting cheating model is. Among the above four models, Type 1 model is the weakest Type 4 model is the strongest. As shown in the Section III, cheatings of types 2 to 4 can only be detected using AMD codes.

For fault injection attacks, we assume that the attacker has knowledge about the implementation of the hardware platform and can inject arbitrary stuck-at or bit-flip faults into the hardware that performs secret reconstructions. The attackers can use any available fault injection mechanisms [13], [14], [15] and can conduct fault injection attacks targeted for a specific portion of the hardware, e.g. the gates close to the outputs. However, the attacker cannot adjust the injected faults after observing the outputs of the secret reconstruction circuit. We also assume that the rate of fault injection is much lower than the clock rate of hardware for secret retrieval. The simulation results for protecting the secure Shamir’s secret sharing against fault injection attacks using the proposed architecture are shown in Section VI-B.

III. SECRET SHARING WITH BUILT-IN SELF-ERROR DETECTION

The proposed architecture for cheater detection and identification is shown in Figure 1. The cheater detection block is composed of two sub-blocks. The secret reconstruction block will compute the secret according to the shareholder inputs. The recovered secret will be verified by the secret checking block. We require the secret \( S \) to be a codeword of an error detecting code \( C \), which can be written as \( S = (y, R), y \in GF(2^n), R \in GF(2^n), S \in GF(2^n), n = k + r \). At the presence of cheaters, let the distorted secret be \( \tilde{S} \). The error detection is conducted by checking whether \( \tilde{S} \) is a codeword of \( C \). An error is detected if \( \tilde{S} \notin C \). To identify the cheater, multiple secret reconstructions and error detections are performed. The error detection results are stored in a temporary buffer and passed into the cheater identification block. The cheater identification block will identify the cheaters according to the errors detected during the secret reconstruction stage.
A. Why Linear codes are not sufficient

For secure Shamir’s secret sharing based on linear codes, the secret $S$ generated by the dealer is a codeword of a $(n, k)$ linear code $C (S \in C)$, where $n$ is the length of the code and $k$ is the number of information bits. In secret reconstruction stage, errors can be injected into the shares by the shareholders to generate a fake secret $\tilde{S}$, $\tilde{S} = S + e$. For linear codes, the error is masked if $\tilde{S}$ is also a codeword of $C (\tilde{S} \in C)$.

If the cheater has no knowledge of the code used in this architecture, he will successfully cheat the system with a probability of $2^{-r}$ assuming that he injects error $e$ in $S_{l-1}$ with the same probability. However, one major weakness of linear codes used for security is that the sum of any two codewords is also a codeword. Therefore if the cheater has knowledge of the code used in this architecture, he can bypass the detection with a probability 1 by injecting an error $e$ into $S$ such that $e \in C$. Thus the secret sharing scheme based on a linear code is not secure against strong cheaters.

For example, if $z_0$ is cheating and other participants are not cheating, then he can inject an error $e_0$ into his share such that $e = e_0z_0 \in C$ (see (2)). Thereby with the knowledge of the linear code $C$, any single cheater out of $l$ participants can cheat the system into reconstructing a wrong secret and bypass the detection with a probability of 1.

Example 3.1: Use a (5,2,3) Hamming code $C = \{00000, 110001, 10111, 01110\}$ and $l = 3$ as a simple example. For the secret generation stage, denote the secret distribution polynomial as $\alpha(z) = S_0 + S_1 z + S_2 z^2$, where $S_2$ is the secret. If we assume the cheater knows the code that is being used, he also knows the equation for reconstructing the secret is

$$\frac{1}{(z_0 + z_1)(z_0 + z_2)} + \frac{1}{(z_0 + z_1)(z_1 + z_2)} + \frac{1}{(z_0 + z_2)(z_1 + z_2)} = S_2.$$  (3)

Assume the cheater $z_0$ can inject error $e_0$ into his share, the resulting output becomes

$$\frac{1}{(z_0 + z_1)(z_0 + z_2)} + \frac{1}{(z_0 + z_1)(z_1 + z_2)} + \frac{1}{(z_0 + z_2)(z_1 + z_2)} = S_2 + e,$$  (4)

where $e = e_0(\frac{1}{(z_0 + z_1)(z_0 + z_2)} + \frac{1}{(z_0 + z_1)(z_1 + z_2)} + \frac{1}{(z_0 + z_2)(z_1 + z_2)})$. The probability of missing an error is $\frac{1}{3}$ if the cheater has no information about the system. However, since $z_0, z_1$ and $z_2$ are publicly known, in order to bypass the protection by linear codes, the cheater only has to find an $e_0$ such that $e \in C$. By injecting the corresponding $e_0$, the cheater can guarantee that his cheating will be successful with probability 1.

B. Detection of cheating with Robust codes

According to the discussion in Section III-A, it’s obvious that linear codes are insufficient for the protection of a secure secret sharing module under strong cheating models. In order to improve the security level of the system, we will use robust codes [16], [17], [18] to protect the secret sharing module in this section.

For robust codes, the codewords have the form of $(y, f(y)) y \in GF(2^k), f(y) \in GF(2^r)$. Error $e = (e_y, e_f), (e_y \in GF(2^k), e_f \in GF(2^r))$ is masked for a given $y$ if

$$f(y + e_y) = f(y) + e_f.$$  (5)

For good robust codes, the fraction of $y$ satisfying (5) is very small for any error $(e_y, e_f)$ [16], [17], [18].

For instance, when $k = 96$, we can divide $y$ into 3 parts $(y_0, y_1, y_2), y_i \in GF(2^{32}), i = 1, 2, 3$. The redundant bits for robust codes can be calculated as

$$R = f(y) = y_0 y_1 + y_2^3.$$  (6)

The number of redundant bits for the code is 32.

If an error $(e_0, e_1, e_2, e_f) (e_i, e_f \in GF(2^{32}))$ is injected, the error masking equation is

$$(y_0 + e_0)(y_1 + e_1) + (y_2 + e_2)^3 = f(y) + e_f,$$  (7)

then

$$y_0e_1 + y_1e_0 + y_2^2 e_2 + y_2 e_2^2 + e_2^3 + e_0e_1 + e_f = 0.$$  (8)

Thus the probability of a nonzero error being masked assuming equiprobable $y$ is $Q_{eq}(e) = Q_{eq}(e_0, e_1, e_2, e_f) = 2^{-r+1} = 2^{-31}$ [16], [17], [18].

Suppose the secret is selected to be a codeword of the above robust code for the Shamir’s Secret Sharing scheme. After the codeword is chosen, it can be easily verified in a similar way that randomly injected errors which are uniformly distributed will be detected with a probability of $2^{-r}$.

Example 3.2: In this example, we will show why Robust code is sufficient for Type 1 cheating model, but fails for Type 2-4 models. Under Type 1 cheating model, the cheater does not know $S = S_2 = (y, f(y))$ and thus cannot predict an error that can be injected without being detected with certainty.
In this example, we will use a system with 9 bits in the original secret and 3 bits redundancy to explain the robust code based system. The generating polynomial for $GF(2^3)$ is $x^3 + x + 1$ and $x^{12} + x^6 + x^3 + x + 1$ is the primitive polynomial for $GF(2^{12})$. If we take $y = (010111101)1$, in which $y_0 = (010), y_1 = (111)$ and $y_2 = (101)$, then $f(y) = y_0 + y_1 + y_2 = (011)$. Thus the extended secret with redundancy is $S = (010111101011)$. For this robust code based system, the chance for the cheater to select an undetectable error $e$ once $y$ is fixed is $2^{-3}$.

For Type 2 cheating model, if the cheater can obtain information about $S$ from the feedback, they can calculate the error to inject thus to produce an undetectable error. For example, if $z_0 = (1000111111110), z_1 = (0010000111110), z_2 = (0110111101011)$ are the participants for the secret reconstruction, and $z_0$ is the cheater with knowledge of $S$, he can inject an error $e_0$ such that $\frac{1}{x_0 + z_2 (x_0 + z_0)} e_0 + S \in C$ to bypass the protection of the error checking module based on robust code, where $C$ is the set of codewords of the robust code in equation (6). Since $z_0, z_1, z_2, S, C$ are all known, it is easy to work out the error $e_0$ which will cause an undetectable error.

For instance, to distort the secret into $\tilde{S}_2 = (0111001110110)$, the cheater needs to introduce an error such that $e = (0001111111110) = e_0 (\sum_{i=0}^{3} x_i + z_i)$. It is not difficult for $z_0$ to find out that an error $e_0 = (110111110001)$ can be introduced into this share to achieve this.

Similarly, under Type 3 cheating model, the $l$ shareholders can collaborate to work out the secret $S$ and thus inject undetectable errors into the system. Once $S$ is known, it will be easy for the cheaters to inject errors which can bypass the error checking module for the following rounds of secret retrieval. Under Type 4 cheating model, the cheaters can get information of $S$ either from the feedback or collaboration. Thereby the robust code based scheme is not secure under Type 2-4 cheating models.

C. Detection of cheating with AMD codes

In this section, we will present an architecture based on algebraic manipulation detection (AMD) codes [19], [20], [21], [22], [23] and discuss the proposed architecture under Type 2-4 cheating models.

For AMD code, a random vector $x \in GF(2^m), m = tr, t \in \{1, 2, 3, \ldots \}$ will be generated and stored securely in the device such that it cannot be read by any cheater without destroying the device itself.

We note that it is generally not a good idea to store the information bits $y$ of the secret in the secure memory since if a cheater reads the secure memory by destroying it, he can obtain knowledge of the secret $y$. This would allow outside cheaters to gain access to the secret without knowledge of the secret sharing scheme itself. However, if we only store the random bits $x$ in the secure memory, even if this memory is read and the cheaters gain some knowledge of $x$ by destroying the secret sharing module, our secret still remains secure.

The codewords $(y, x, f(x, y))$ of an AMD code consist of three parts, the message $y \in GF(2^r)$, the random vector $x \in GF(2^m)$, and redundant bits $f(x, y) \in GF(2^r)$. Since $x$ is stored in the system and thus the secret $S_{l-1}$ is in the form of $(y, f(x, y))$. For this architecture, when the secret is reconstructed, the device will retrieve $x$ from the secure memory and check if

$$f(\tilde{x}, \tilde{y}) = \tilde{f}(x, y),$$

where $\tilde{y}$ and $\tilde{f}(x, y)$ are from the distorted secret, $\tilde{x}$ is the distorted random number $x$.

With knowledge of $y$ and $f(x, y)$, as well as the polynomial used to construct $f(x, y)$, the cheaters could work out a set of possible values for $x$. If $x \in GF(2^r), m = r$, and $f(x, y) = \sum_{i=1}^{b} y_i x_i^{2^i} + y_{b+2}^1 + y_{b+3}^1$, where $y_i, x_i \in GF(2^r), b$ is odd and $y = (y_1, \ldots, y_0)$, then the size of the set of possible $x$ is at most $b + 2$ [19], [20], [21]. To increase the security level of our system, the size of the random number in this system can be $m = tr$ where $t \geq 2$ [19], [20], [21], [24]. Here we use AMD code with $t = 2$ and thus $x = (x_0, x_1), x_0, x_1 \in GF(2^{32})$. Then the size of the set of possible $x = (x_0, x_1)$ will be much larger and thus improve the security level of the system. The same as robust code based scheme, $y$ is also divided into three parts $(y_0, y_1, y_2) \in GF(2^{32})$ for the AMD code based system. The encoding polynomial used for AMD code in this system is

$$f(x_0, x_1, y) = y_0 x_0 + y_1 x_1 + y_2 x_0 x_1 + x_0^3 + x_1^3,$$

(10)

where $y = (y_0, y_1, y_2)$. Assume the cheaters can inject errors into their shares and inject faults into the hardware system as well to distort $x_0, x_1$. Denote the error for a set of $(y_0, y_1, y_2, x_0, x_1, f)$ as $(e_0, e_1, e_2, e_{x_0}, e_{x_1}, e_f)$. Then the error masking equation is:

$$(y_0 + e_0)(x_0 + e_{x_0}) + (y_1 + e_1)(x_1 + e_{x_1}) + (y_2 + e_2)(x_0 + e_{x_0})(x_1 + e_{x_1}) + (x_0 + e_{x_0})^3 + (x_1 + e_{x_1})^3 = f(x_0, x_1, y) + e_f.$$

(11)

All the computation are in $GF(2^r)$. Equivalently

$$y_0 e_{x_0} + x_0 e_{x_0} + e_0 e_{x_0} + y_1 e_{x_1} + x_1 e_{x_1} + e_1 e_{x_1} + y_2 x_0 e_{x_1} + y_2 x_1 e_{x_0} + y_2 e_{x_0} e_{x_1} + e_2 e_{x_0} x_0 + e_{x_1} e_{x_0} + x_{0}^2 e_{x_0} + e_{x_0} x_0 + e_{x_0}^3 + x_{0}^2 e_{x_1} + e_{x_0} e_{x_1} + e_{x_1} e_{x_1} + e_f = 0.$$

(12)

Since $y$ is known and the errors are injected and controlled by the cheaters, (12) can be rewritten as

$$x_0^2 e_{x_0} + x_1^2 e_{x_1} + e_{x_0} x_0 x_1 + x_{0}^2 e_{x_1} + e_{x_0} x_0 + B x_1 + C = 0,$$

(13)

where $A, B, C$ are functions of $y$ and the error vector according to (12). For any error introduced by the cheater, it is easy to see that the error masking probability for the above AMD code is no larger than $2^{-r+1}$ assuming $x$ is uniformly distributed.

Example 3.3: Let $S = (y_0, y_1, y_2, R)$ and random bits $x_0, x_1$ are stored securely. Choose $(y_0, y_1, y_2) = (001001100)$ and $x_0 = (010), x_1 = (110)$, then $R = f(x_0, x_1, y) = (001), S = (001001110001)$. The cheaters wish to choose an error $e$ such that $S = S + e \in C$. Under Type 2 to 4 cheating models, the cheater knows the secret $S$. However, without the knowledge of $x_0, x_1$, the cheater still has no way to inject
errors that are always undetectable by AMD codes. Given \( y = (y_0, y_1, y_2) \) and \( R = f(x_0, x_1, y) \), there is only a limited number of \( x = (x_0, x_1) \) that satisfies (10). The cheaters can generate a set of all \((x_0, x_1)\) satisfying (10) and inject errors that are undetectable by one of the \( x'\)'s in the set. In this example, there are 5 possible \((x_0, x_1): (001, 101), (010, 110), (100, 101), (011, 111), (101, 101)\). Thus, the probability of successfully cheating is \(\frac{1}{5}\), which is less than the upper bound \(2^{-r+1} = \frac{1}{3}\) [19], [20], [21].

When \( r \) is large, finding all possible \(x_0\) and \(x_1\) satisfying a given secret \((y_0, y_1, y_2, R)\) becomes impractical. If the cheater randomly injects uniformly distributed errors into the share, it is easy to verify that the error masking probability can be as small as \(2^{-r}\).

IV. CHEATER IDENTIFICATION AND SUPER IMPOSED CODES

In this Section, we will show that the cheater identification problem for a \((l, L)\) Shamir’s secret sharing scheme is tightly related to classical coding theory topics such as \(m\)-separable matrices and superimposed codes.

A. Cheater Identification

We increase the number of participants for secret reconstruction for a \((l, L)\) Shamir’s secret sharing from \(l\) to \(N\), \(l < N \leq L\). All the \(N\) participants will first submit their shares. Then the secret will be computed for \(A\) times using different subsets of \(l\) shareholders out of the \(N\) participants. During the whole period of secret reconstruction, no participants are allowed to re-submit their shares. Let \(M\) be a \((A \times N)\) cheater identification matrix where \(M_{i,j} = 1\) if the \(j\)th shareholder participates in the \(i\)th secret reconstruction and \(M_{i,j} = 0\) otherwise, \(1 \leq i \leq A\), \(1 \leq j \leq N\).

We assume that the cheating will cause the secret reconstruction to fail and will be detected by the corresponding AMD codes as described in the previous Section. Let \(u \in GF(2^N)\) be an \(N\)-bit binary vector, where \(u_{i} = 1\) indicates that the \(i\)th participant is a cheater. Let \(m\) be the maximum number of potential cheaters. Then the weight of \(u\) (number of 1's in the vector) is upper bounded by \(m\). We define the syndrome \(T \in GF(2^4)\) of the performed \(A\) secret reconstructions to be

\[
T = M \cdot u,
\]

or equivalently

\[
T_i = M_{i,1}u_1 | M_{i,2}u_2 | \cdots | M_{i,N}u_N,
\]

where \(|\) is the bitwise OR operation.

The following three statements follow from (14) and (15).

- When any of the shareholders participating in the \(i\)th secret reconstruction cheats, the secret reconstruction will fail and \(T_i = 1\).
- If the \(i\)th participant is a cheater \((u_i = 1)\), then all the secret reconstructions he participates will fail.
- If more than 1 shareholder among the \(N\) participants cheats, the syndrome \(T\) will be the bitwise OR of the column vectors of matrix \(M\) that correspond to the cheaters.

The following theorem follows directly from the above statements and describes the necessary and sufficient condition for identifying up to \(m\) cheaters based on \(M\).

**Theorem 4.1:** Up to \(m\) cheaters can be identified using \(M\) if and only if the bitwise OR of any two sets of columns with size at most \(m\) are different.

**Example 4.1:** Let \(l = 4\) and \(N = 5\) and

\[
M = \begin{bmatrix}
0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 1
\end{bmatrix}.
\]

Suppose the second shareholder is the only cheater, then \(u = (01000)^T\) and \(T = M \cdot u = (1011)^T\). If the second and the forth shareholders are both cheaters, then \(u = (01010)^T\) and \(T = M \cdot u = (1111)^T\), which is the bitwise OR of the second and the forth columns of \(M\).

All of the columns in \(M\) are unique, thus all single cheaters can be identified based on \(M\). However, the 5th column of \(M\) is equal to the bitwise OR of the 2nd and the 4th columns. Thereby \(M\) cannot guarantee the identification of 2 cheaters.

Following the above analysis, the problem of identifying up to \(m\) cheaters for a \((l, L)\) Shamir’s secret sharing scheme can be converted to the problem of constructing a \((A \times N)\) binary matrix \(M\) satisfying conditions in Theorem 4.1, where \(A\) is the number of secret reconstructions that need to be performed and \(N \geq l\) is the total number of participants in all secret reconstructions. In the next Section, we will show that the desired matrix \(M\) is actually an \(m\)-separable matrix which is tightly related to other research topics such as \(m\)-disjunct matrices and superimposed codes.

**Remark 4.1:** In practice, the efficiency of the above cheater identification scheme is affected by the error detection mechanism. When errors are not detected, an erroneous secret will be re-constructed and the corresponding \(T_i\) will be 0’s, which may result in incorrect identification of cheaters. However, we note that the probability of missing errors can be made arbitrarily small by selecting a strong enough error detection mechanism (see Section III). Moreover, the presented cheater identification methodology can actually tolerate some undetected errors. The cheaters can still be correctly identified even if not all erroneous secrets are detected during the secret reconstruction stage (see Section IV-C).

B. \(m\)-Separable Matrices, \(m\)-Disjunct Matrices and Superimposed Codes

\(m\)-separable matrices, \(m\)-disjunct matrices and superimposed codes [25] are classical coding theory topics that are well studied in the literature and widely adopted for different applications such as group testing [26].

**Definition 4.1:** [25] A binary matrix is \(m\)-separable if the logical sum of up to \(m\) columns are mutually different.

We note that the necessary and sufficient conditions for \(M\) to identify up to \(m\) cheaters described in Theorem 4.1 is coherent to the definition of \(m\)-separable matrix. As a result, the following Corollary holds.
Corollary 4.1: Up to \(m\) cheaters can be identified with \(N\) participants and \(A\) secret reconstructions for a \((l, L)\) Shamir’s secret sharing scheme if there exists a \((A \times N)\) \(m\)-separable matrix \(M, N \leq L\).

Definition 4.2: For any two \(A\)-bit binary vectors \(u\) and \(v\), we say that \(u\) covers \(v\) if \(u|v = u\), where \(\lceil\cdot\rceil\) denotes the bitwise OR operation. A \((A \times N)\) matrix \(M\) is \(m\)-disjunct if the bitwise OR of any set of no more than \(m\) columns does not cover any other single column that is not in the set. The columns of a \(m\)-disjunct matrix compose a \(m\)-superimposed code.

In [25], it has been shown that an \(m\)-disjunct matrix, which also defines a superimposed code, is always an \(m\)-separable matrix.

Superimposed codes can be constructed from conventional error correcting codes such as Reed-Solomon codes and Hamming codes [25].

Construction 4.1: [25] Let \(C\) be a \((n_q, k_q, d_q)\) \(q\)-ary linear error correcting code. Suppose we represent each element of \(C\) as \((q^n, d_q)\) binary vector with Hamming weight 1. Construct \(C'\) by substituting every \(q\)-ary element in the codewords of \(C\) by its corresponding binary vector. The binary code \(C'\) has length \(qn_q\), distance \(2d_q\) and \(q^k\) codewords, it is a \(m\)-superimposed code, where \(m = \left\lfloor \frac{n_q - 1}{q^n - d_q} \right\rfloor\).

Example 4.2: The codewords of a ternary Reed-Solomon code \((3, 2, 2)_3\) are listed below.

\[
C = \{(0, 0, 0), (1, 1, 1), (2, 2, 2), (0, 1, 2), (1, 2, 0),
(2, 0, 1), (0, 2, 1), (1, 0, 2), (2, 1, 0)\}. \tag{16}
\]

Suppose 0, 1, 2 in \(GF(3)\) are represented by binary vectors \((100), (010)\) and \((001)\) respectively, then the binary code \(C'\) consists of 9 codewords shown as the columns in the following 2-disjunct matrix.

\[
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
\delta_1 \cdot 0 & \delta_2 & \delta_2 & \delta_3 & \delta_4 & \delta_4 & \delta_5 & \delta_6 & \delta_7 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
\delta_1 \cdot 0 & \delta_2 & \delta_2 & \delta_3 & \delta_4 & \delta_4 & \delta_5 & \delta_6 & \delta_7 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
\end{bmatrix}
\]

The binary code \(C'\) has length 9, Hamming distance 4 and contains the same number of codewords as the original ternary code \(C\). The code \(C'\) is a 2-superimposed code and it is easy to verify that the bitwise OR of any two sets of up to 2 columns are mutually different. Each row of the above matrix contains exactly three 1’s. Thereby, this matrix can be directly used as the \(M\) matrix to identify up to 2 cheaters of any \((3, L)\) Shamir’s secret sharing scheme where \(N = 9 \leq L\).

To identify cheaters for a \((l, L)\) Shamir’s secret sharing scheme, each row of the matrix \(M\) has to contain exactly \(l\) 1’s. We note that for the \(m\)-superimposed codes constructed from \(q\)-ary linear codes based on Construction 4.1, this property is always satisfied. To prove this statement, we first start with the following lemma.

Lemma 4.1: For any \(q\)-ary linear error correcting codes, every single non-constant redundant digit takes values from \(GF(q)\) with a probability of \(q^{-1}\).

Proof: Let \(C\) be a \(q\)-ary linear error correcting code. Denote by \(c_1, c_2, \ldots, c_k\) the information digits and \(c_{k+1}, c_{k+2}, \ldots, c_n\) the redundant digits in the codewords. For linear codes, all the redundant digits can be represented as linear functions of the information digits. Let

\[
c_{k+1} = \delta_1 c_1 + \delta_2 c_2 + \cdots + \delta_k c_k, \tag{17}
\]

where \(\delta_i, 1 \leq i \leq k\) are coefficients in \(GF(q)\). Since \(c_{k+1}\) is not a constant, \(\delta_i\) cannot be all zero’s. Without loss of generality, assume \(\delta_1 \neq 0\). Then for any given \(c_{k+1}\) and \(c_2, c_3, \ldots, c_k\), there is exactly one solution for \(c_1\) that satisfies the above equation. As a result, for any element \(v\) in \(GF(q)\), the probability that \(c_{k+1} = v\) is \(q^{-1}\).

According to Lemma 4.1, for any element \(v\) in \(GF(q)\), any non-constant redundant digit of a \(q\)-ary \((n_q, k_q, d_q)\) code \(C\) is equal to \(v\) for exactly \(q^{k_v-1}\) codewords. After converting the \(q\)-ary code \(C\) to the \(m\)-superimposed binary code \(C'\) as described in Construction 4.1, any single redundant digit corresponds to a \(q \times N\) submatrix. Since different binary vectors of Hamming weight 1 are used to represent the elements in \(GF(q)\), it is easy to verify that each of this \(q\) rows contains exactly \(q^{k_v-1}\) 1’s. The locations of these ones correspond to codewords in the original code \(C\) that have the same value for this particular redundant digit.

Similarly, we can show that all the rows in \(M\) corresponding to the information digits in the original code \(q\)-ary code \(C\) also contains exactly \(q^{k_v-1}\) 1’s. Thereby we have the following corollary.

Corollary 4.2: Any linear \((n_q, k_q, d_q)\) code can be used to construct a \((A \times N)\) matrix to identify up to \(m\) cheaters for a \((l, L)\) Shamir’s secret sharing scheme using Construction 4.1, where \(A =qn_q, N = q^{k_v} l = q^{k_v-1}, A \leq L\) and \(m = \left\lfloor \frac{n_q - 1}{q^n - d_q} \right\rfloor\).

The matrix \(M\) constructed from superimpose codes is not only \(m\)-separable but also \(m\)-disjunct. No single column \(h\) will be covered by the bitwise OR of up to \(m\)-columns other than \(h\). Moreover, the number of 1’s in each column of \(M\) is exactly \(n_q\) assuming we represent each element in \(GF(q)\) by a \(q\)-bit binary vector with Hamming weight 1. This property results in a simple error locating algorithm as shown in the next Theorem.

Theorem 4.2: Let \(v\) be the \(A\)-bit binary vector representing the error detection results. Then \(u = v \cdot A\) is a vector of length \(N\), where \(\cdot\) is the arithmetic multiplication. Suppose all errors in the secret reconstruction processes are successfully detected. We have \(u_i \leq n_q, 1 \leq i \leq N\). Moreover, there are at most \(m\) indexes \(i\) such that \(u_i = n_q\). These indexes correspond to the cheaters.

Example 4.3: In Example 4.2, when the second and the third participants cheat, the error detection result is \(v = (011011011)\). Then \(u = v \cdot M = (033222222)\). Thereby \(u_2 = u_3 = 3\), which successfully identified the participants that cheat. All the other elements of \(u\) are less than 3.
C. Tolerance of Undetected Errors in Cheater Detection Stage

As mentioned in Section III, the input to the error identification network is an $A$-bit binary vector representing the error detection results. A '1' in the binary vector stands for a detected error during the secret reconstruction stage. We note that for cheater detection method based on error detection codes, there are small chances that errors (cheaters) will not be detected in one or more secret reconstructions. However, since the binary code composed of the columns of $M$ has a Hamming distance of $2d_q$ (see Construction 4.1 and Example 4.2), it can correct up to $d_q - 1$ bit errors. Thereby, even if some erroneous secrets are not detected during the secret reconstruction stage, we may still be able to identify the cheater as stated in the following Theorem.

Theorem 4.3: The cheater identification method based on superimposed codes constructed from a $(n_q, k_q, d_q)$ linear error correcting codes as described in Construction 4.1 can identify a single cheater when up to $d_q - 1$ errors in the secret reconstruction stage are missed for any $m$.

Proof: When there is a single cheater, the $A$-bit syndrome is a column of the cheater identification matrix $M$ assuming all errors are detected. The weight (the number of 1’s) of the syndrome is exactly $n_q$. When there are more than one cheater, the syndrome is the bitwise OR of multiple columns in the matrix $M$. It is easy to verify that the weight of the un-distorted syndrome at the presence of multiple cheaters is at least $n_q + d_q$. Undetected secret reconstruction failures will result in $1 \rightarrow 0$ undirectional errors in the syndrome. The weight of the distorted syndrome will be strictly smaller than the weight of the correct syndrome. Suppose up to $d_q - 1$ errors are missed during the secret reconstruction stage, the weight of the syndrome will be in the range of $[n_q - d_q + 1, n_q]$ if and only if there is a single cheater. The correct syndrome can be recovered after the undetected errors are corrected by the code composed of all columns of $M$, which has a Hamming distance of $2d_q$. After this the single cheater can be identified by comparing the syndrome to the columns of matrix $M$.

Example 4.4: In Example 4.2, the binary code $C'$ has a Hamming distance of 4. It can correct any 1-bit error. Suppose $m = 1$ and the second participant cheats. Then the second, the fifth and the eighth secret reconstructions will be affected. If all errors are detected, the syndrome is (0100100010). Without loss of generality, we assume the error in the second secret reconstruction is not detected, then the distorted syndrome becomes (0000100010). Since $C'$ has a Hamming distance of 4, (0000100010) can be easily corrected to (0100100010) using the classical maximum likelihood decoding algorithm. After correcting the 1-bit error in the binary vector, the cheater can be successfully identified.

Let $P_M$ be the probability that errors in a single secret reconstruction is not detected. Since up to $d_q - 1$ bit errors in the syndrome can be tolerated for detecting single cheaters, the probability $P_I$ of successfully identifying a single cheater for any $m$ can be calculated as follows:

$$P_I = (1 - P_M)^{n_q} + \sum_{i=1}^{d_q-1} (1 - P_M)^{n_q-i} P_M^i$$  \(18\)

As discussed in the previous Sections, for error detection using robust codes or AMD codes, $P_M$ will decrease exponentially as $r$ increases. Thereby for large $r$, e.g. $r = 32$, $P_I$ will be extremely close to 1. In this case single cheaters are almost always successfully identified by the proposed cheater identification algorithm. (See Section VI.)

Remark 4.2: Theorem 4.3 can be further extended for identifying multiple cheaters at the presence of undetected errors during the secret reconstruction stage. In general, let $C_i$ be a code that consists of all the vectors that are the bitwise OR of $t$ columns in the matrix $M$. Let $d_{C_i}$ be the Hamming distance of $C_i$. Then $t$ cheaters can be potentially identified at the presence of up to $\lceil \frac{d_{C_i} - 1}{q} \rceil$ undetected errors. For instance, when $t = 2$ and $n_q = 4$, it is easy to show that $d_{C_2} = 2n_q - 4 = 4$. Double cheaters can still be successfully identified when there is one undetected error.

From Corollary 4.2, it is easy to show that when $q$, $n_q$, $k_q$ are fixed, linear error correcting code with larger Hamming distance $d_q$ will result in stronger superimposed codes with bigger $m$. When $C$ is a maximum distance separable (MDS) code that meets the Singleton bound [27], we have $d_q = n_q + 1 = n_q - k_q + 1$ and $m = \lceil \frac{n_q - 1}{q} \rceil$. The parameters of the cheater identification scheme constructed from various MDS codes will be studied in the next Section.

D. Cheater Identification Based on Superimposed Codes Constructed from MDS Codes

The most well known non-trivial MDS codes are Reed-Solomon codes, extended Reed-Solomon Codes and the shortened version of these codes [27]. For any $q = p^v$, where $p$ is a prime and $v$ is a positive integer, there exist $(n_q = q - 1, k_q = n_q - k_q + 1)$ Reed-Solomon codes for $1 \leq k_q \leq n_q - 1$. When $k_q = n_q - 1$, the Reed-Solomon codes coincide with the trivial MDS codes - the 1-d parity codes. The constructions of extended Reed-Solomon codes with parameters $(n = q + 1, k_q, d_q = n_q - k_q + 1)$ can be found in [27]. The shortened versions of both the original and the extended Reed-Solomon codes can be constructed by deleting any number of information digits from the codes. The resulting code has parameters $(n_q' = n_q - \delta, k_q' = k_q - \delta, d = n_q - k_q + 1)$, where $1 \leq \delta \leq k_q - 1$. These shortened codes still meet the Singleton bound and are also MDS codes.

1) Maximizing $m$ for Fixed $q$: From Corollary 4.2, the number of shareholders $l$ needed for secret recovery for the Shamir’s scheme and the cheater identification capability $m$ can be written as functions of the parameters of the MDS codes, i.e. $l = q^{n_q} - 1, m = \lceil \frac{n_q - 1}{k_q} \rceil$.

Table I shows $m$ and $l$ resulting from Reed-Solomon codes, extended Reed-Solomon codes and shortened Reed-Solomon codes constructed over $GF(5)$ ($q = 5$). We note that $m = \lfloor \frac{n_q - 1}{k_q} \rfloor$ is an increasing function of the ratio of $n_q$ over $k_q$. To achieve a stronger cheater identification capability, we should select $n_q$ to be as large as possible and $k_q$ to be as small as possible. The selection of $k_q$ is independent of the selection of $n_q$. (Once $n_q$ is fixed, $k_q$ only affects the distance of the MDS codes.) The extended version of the Reed-Solomon codes with $n_q = q + 1$ and $k_q = 2$ will maximize $n_q$ and minimize $k_q$. 

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Corollary 4.3: A cheater identification scheme constructed by Construction 4.1 based on Reed-Solomon codes over $GF(q)$ can identify at most $q$ cheaters. The upper bound of the identification capability is achieved when an extended Reed-Solomon code with $k_q = 2$ is used.

Example 4.5: When $q = 5$, at most 5 cheaters can be identified using schemes based on Reed-Solomon codes. From Table I, if we start from an extended Reed-Solomon codes with $n_q = 6, k_q = 2$ and $d_q = 5$, we have $m = 5$ and $l = 5$. Five shareholders are required each round of the secret recovery. In total $N = 25$ participants are needed and $A = qn_q = 30$ rounds of the secret recoveries have to be conducted in order to identify all the cheaters.

Remark 4.3: Given $q$ and $n_q$, minimizing $k_q$ will maximize the Hamming distance of the MDS codes. It is not surprising that increasing the cheater identification capability $m$ is coherent with increasing the error correcting capability of the MDS codes, which is determined by the Hamming distance of the codes.

2) Minimizing the Hardware Overhead given $m$: The hardware overhead of the cheater identification scheme constructed by Construction 4.1 increases as either $A$ or $l$ increases. The former affects the number of secret recoveries that need to be conducted for cheater identifications. The latter determines the amount of computations required for each round of secret recovery. When $q$ is fixed, $A$ is a strictly increasing function of $n_q$ and $l$ is a strictly increasing function on $k_q$. To minimize the hardware overhead, $k_q$ and $n_q$ should be as small as possible. Since $m = \left\lfloor \frac{n_q - 1}{k_q - 1} \right\rfloor$, for any given $m$ minimizing $k_q$ and $n_q$ are coherent with each other. Thereby, the smallest $k_q$ such that there exists $n_q$ satisfying $m = \left\lfloor \frac{n_q - 1}{k_q - 1} \right\rfloor$ should be selected to minimize the hardware overhead for any given $m$.

Example 4.6: In Table I, there are three sets of $n_q, k_q, d_q$ that achieve $m = 2$, which are

- $n_q = 3, k_q = 2, d_q = 2$;
- $n_q = 5, k_q = 3, d_q = 3$;
- $n_q = 6, k_q = 3, d_q = 4$.

Among these three alternatives, $n_q = 3, k_q = 2$ and $d_q = 2$ results in the smallest $l$ and $A = qn_q$ (see Table I). It requires the least hardware overhead to identify up to 2 cheaters.

The former analysis can be used to minimize the hardware overhead for a specific $m$ once $q$ is fixed. The hardware for the cheater identification scheme based on superimposed codes constructed from MDS codes, however, not only depends on $n_q, k_q, d_q$ but also depends on $q$ itself.

For a specific $m$, the question of how $q$ should be selected is not yet answered. To answer this question, we start with the following two observations.

- $m$ is independent of $q$.
- The minimum $k_q$ and $n_q$ for any $q$ are always 2 and 3 respectively, which results in $m = 2$.

Corollary 4.4: For any given $q$ and any $m, 1 \leq m \leq q$, a Shamir’s secret sharing scheme that is able to identify up to $m$ cheaters can be constructed from RS codes over $GF(q)$ (The RS codes may be shortened or extended.) Let $U_{q,m}$ be the set of parameters $n_q, k_q, d_q$ for RS codes over $GF(q)$ that can be used to identify up to $m$ cheaters. Then $U_{q_1,m} \subset U_{q_2,m}$ if $q_1 \leq q_2, m \leq q_2$.

TABLE I

<table>
<thead>
<tr>
<th>$m$</th>
<th>$l$</th>
<th>$N$</th>
<th>$n_q$</th>
<th>$A$</th>
<th>$k_q$</th>
<th>$d_q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>25</td>
<td>3</td>
<td>15</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>25</td>
<td>4</td>
<td>20</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>125</td>
<td>4</td>
<td>20</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>125</td>
<td>5</td>
<td>25</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>125</td>
<td>625</td>
<td>5</td>
<td>25</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>125</td>
<td>6</td>
<td>30</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>125</td>
<td>625</td>
<td>6</td>
<td>30</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>625</td>
<td>3125</td>
<td>6</td>
<td>30</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

TABLE II

<table>
<thead>
<tr>
<th>$m$</th>
<th>$l$</th>
<th>$N$</th>
<th>$n_q$</th>
<th>$A$</th>
<th>$k_q$</th>
<th>$d_q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>16</td>
<td>3</td>
<td>12</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>16</td>
<td>4</td>
<td>16</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>16</td>
<td>64</td>
<td>4</td>
<td>16</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>64</td>
<td>5</td>
<td>20</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>64</td>
<td>625</td>
<td>5</td>
<td>20</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>
requires less participants \( N \) at the cost of larger number of tests \( A \).

In Example 4.6, we have seen that 5 participants are enough for identify a single cheater for any \((l, L)\) Shamir’s secret sharing scheme, where \( l = 4 \) and \( L = 5 \). This method can be generalized for any \( l \) and \( L \geq l + m \) as shown in the next Construction.

Construction 5.1: For any \((l, L)\) Shamir’s secret sharing scheme, let \( \delta = L - l \). The minimum number of participants \( N \) that is required to identify \( m \) cheaters based on superimposed codes is \( N = l + m, m \leq \delta \). The corresponding \( M \) can be constructed by using all \( N \)-bit vectors with \( l \) ones as rows of the matrix, except for the vector with \( l \) consecutive ones at the beginning. The number of tests required to identify \( m \) cheaters is \((l + m) - 1\).

Proof: If the number of participants \( N \) is less than \( m + l \), any round of secret reconstruction will contain at least one cheater. When \( N = l + m \), then only one subset of \( l \) shareholders can retrieve the correct secret. The other \( m \) shareholders participating in the secret reconstruction will be cheaters. Thereby, the minimum number of participants needed to identify \( m \) cheaters is \( N = l + m \). The corresponding number of tests is \( N = (l + m) - 1 \).

Generally speaking, the cheater identification scheme based on Construction 5.1 has smaller \( N \) but larger \( A \) compared to the scheme based on Construction 4.1, which is shown in the next Example.

Example 5.1: For \( m = 2, l = 4, N = 6 \), the following 2-disjunct \((14 \times 6)\) matrix \( M \) will be used to identify two cheaters according to Construction 5.1. The total number of required secret reconstructions is 14. In Table II, it is shown that the cheater identification scheme based on Construction 4.1 only requires 12 tests. However, for Construction 4.1 the number of participants is 16 and is more than doubled compared to Construction 5.1.

\[
M = \begin{bmatrix}
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 1 & 1 & 0 & 1
\end{bmatrix}.
\]

VI. SIMULATION AND SYNTHESIS RESULTS FOR DETECTION OF CHEATING

In this section, we will present the simulation results to justify the security level of the proposed secure Shamir’s secret sharing scheme. We will also show the synthesis results for the proposed architecture to estimate the required timing and area overhead.

A. Simulation Results for Detection and Identification of Cheating

We will present the simulation results under Type 1 cheater model for the proposed cheater detection and identification methods based on robust codes, AMD codes and superimposed codes. Secure Shamir’s Secret Sharing Scheme using robust codes can only provide sufficient protection against Type 1 cheater. A cheater of Type 2, 3 or 4 can easily introduce an error that is always not detectable by robust codes. Secure Shamir’s Secret Sharing Scheme using AMD codes can provide high protection against all types of cheaters. Without the knowledge of \( x \), knowing the secret or collaborating with other cheaters cannot increase the chance for the cheater to inject undetectable errors. Thereby, the simulation results presented in this Section can also be used to evaluate the effectiveness of the proposed architecture using AMD codes under other types of cheater models.

We implement a \((l = 4, L = 16)\) secure Shamir secret sharing scheme based on the proposed architecture in C++. The design is capable of identifying up to two cheaters. The share generation and the secret reconstruction equations can be easily derived from Section II-A for \( l = 4 \). We select the secret to be 128-bit. The generator polynomial of \( GF(2^{128}) \) is \( x^{128} + x^7 + x^2 + x + 1 \). The length of the codewords for the chosen error detection codes is determined by the length of the secret, which is 128-bit. The number of information bits and the Galois field in which the code is constructed, however, can be selected according to the required security level. To compare the performance of secure Shamir secret sharing schemes based on different error detection codes, we considered and implemented designs using different robust and AMD codes as shown below.

- Robust codes defined over \( GF(2^{32}) \) with \( k = 96, r = 32 \) and \( f(y) = y_0y_1 + y_2^3 \);
- AMD codes defined over \( GF(2^{32}) \) with \( k = 96, r = 32 \), 64-bit random data and \( f(y, x) = y_0x_0 + y_1x_1 + y_2x_0x_1 + x_0^3 + x_1^7 \);
- Robust codes defined over \( GF(2^{16}) \) with \( k = 112, r = 16 \) and \( f(y) = y_0y_1 + y_2y_3 + y_4y_5 + y_6^3 \);
- AMD codes defined over \( GF(2^{16}) \) with \( k = 112, r = 16 \), 16-bit random data and \( f(y, x) = y_0x + y_1x^2 + y_2x^3 + y_3x^4 + y_4x^5 + y_5x^6 + y_6x^7 + x^9 \).

To identify up to two cheaters, we use the \((12 \times 16)\) cheater identification matrix constructed from a \((3, 2, 2)\) Reed Solomon code shown in Figure 2. Twelve secret reconstructions need to be performed in order to identify the cheaters. Every secret reconstruction requires four shareholders. Each shareholder will participate in three reconstructions. A total number of 16 participants are required by the proposed secure Shamir’s secret sharing scheme to recover the secret.

For the secure Shamir’s Secret Sharing scheme based on each of the above four error detection codes, we perform the following simulation steps

1) Randomly generate 50 sets of share holder IDs and coefficients \( S_0, S_1, S_2 \) and \( S_3 \), where secret \( S = S_3 \) is a random non-zero codeword of the selected code;

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
2) Compute the shares and distribute them to the share holders;
3) For each set of IDs and secrets, perform 1 million rounds of random cheats. For each round of the cheat, randomly select 1 or 2 cheaters among the 16 participants. Randomly generate errors and inject the errors into shares of the cheaters;
4) Conduct error detection and cheater identification in the existence of cheaters. Summarize and categorize the undetected errors and unidentified cheaters.

When robust codes and AMD codes are constructed over $GF(2^{12})$, the error masking probabilities are extremely small (see Section II). During the simulation, all errors are successfully detected at the secret reconstruction stage. All the cheaters are correctly identified by the cheater identification algorithm presented in Theorem 4.2.

The simulation results for the proposed cheater detection and identification scheme using error detection codes constructed over $G(2^{16})$ are shown in Table III. For both robust and AMD codes constructed over $GF(2^{16})$, the probability of not detecting an error during the secret reconstruction stage is close to $2^{-16}$ assuming the cheater inject random errors into the share.

In Theorem 4.3, we have shown that the proposed cheater identification can still successfully identify single cheaters at the presence of up to $d_4 - 1$ undetected errors. This is verified by the simulation results. The chosen Reed-Solomon code has $d_4 = 2$. Thus the resulting cheater identification scheme can tolerate up to one undetected error. For robust codes and AMD codes constructed over $GF(2^{16})$, the probability of having two undetected errors is very small. (It is comparable to the error masking probability for codes constructed over $GF(2^{32})$.) All single cheaters are successfully identified during the simulation.

When there are two cheaters, in most situations we correctly identified one out of two cheaters. There are few cases where both cheaters are missed. The overall probability of unidentified cheaters has an order the magnitude of $10^{-5}$ when two shareholders cheat.

In general, the probability of not detecting cheaters is reduced when the error masking probability for the error detection codes decreases. To better understand the relationship between the two, we performed simulations to evaluate the probability of not detecting cheaters under different error masking probabilities. During the simulation, the error masking probability $P_M$ is changed from $2^{-1}$ to $2^{-32}$. For each error masking probability $P_M$, we intentionally miss the fake secrets caused by the cheaters with a probability of $P_M$. 10 millions cheater identification simulations are performed for each fixed $P_M$. The results are shown in Figure 3. The X-Axis is $-\log_2(P_M)$. The Y-Axis is the probability of not detecting cheaters. It is clear that when the error masking probability decreases, the probability of unidentified cheaters is drastically reduced. When $\log_2(P_M) \leq -29$, all the cheaters are successfully identified during the simulation. This is consistent with the simulation results presented earlier for codes constructed over $GF(2^{32})$.

![Cheater Identification Matrix Used for Simulation](image)

![The Probability of Unidentified Cheaters VS the Error Masking Probability](image)

B. Simulation Results for Detection of Fault injection Attacks

Besides cheating, the attackers may also directly inject faults into the hardware to distort the output of the secret reconstruction platform computing the secret [28], [29], [30]. The proposed architecture is also resistant to fault injection attacks. In this section, we will describe the fault injection simulation results to justify the effectiveness of the proposed architecture as a countermeasure against fault injection attacks on the secret reconstruction hardware.

We assume the attacker has knowledge of the hardware implementation of the Shamir’s secret sharing scheme and can conduct fault injection attack targeting for gates close to the output of the circuit. The attacker has no access to the reconstructed secret (similar to Type 1 cheaters). In the fault injection simulation, we added "control signals" to the gate models for gates close to the output of the circuit. The fault types can be changed by changing the values of the control signals. The secret is fixed during the simulation. We randomly inject stuck-at-0, stuck-at-1 and bit-flip faults into 1 to 5 gates close to the outputs of the secret reconstruction hardware. Fifteen groups of faults are injected into the hardware. For
Silent faults that do not alter the outputs of the hardware have manifested as non-zero errors at the output of the hardware. As the number of affected gates increases, the fault masking probability slightly increases but still has an order of magnitude of $10^{-5}$. Thereby, the proposed architecture based on robust, AMD and superimposed codes can effectively protect the Shamir’s secret sharing scheme against not only cheaters but also fault injection attacks.

We note that robust codes may not be sufficient to provide a satisfactory security level if the codewords are not uniformly distributed [31] or the attacker can conduct biased fault injection attacks [32]. For more advanced attackers who can gain knowledge of the outputs of the secret reconstruction device and can precisely control the injected faults according to the outputs, the security level of Shamir’s secret sharing scheme based on robust codes cannot be guaranteed. Under such strong attacker models, AMD codes should be used whose security level is ensured by the use of random data for encoding and decoding.

### C. Synthesis results for the proposed architectures

The proposed secure secret sharing architecture protected by robust and AMD codes constructed in both $GF(2^{16})$ and $GF(2^{32})$ are synthesized to estimate the timing and area of the proposed architecture. The synthesis results are summarized in Table IV.

<table>
<thead>
<tr>
<th>Undetected Errors</th>
<th>Architecture Based on Robust Codes</th>
<th>Architecture Based on AMD Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>3259</td>
<td>3165</td>
<td></td>
</tr>
<tr>
<td>Total Number Of Errors During Simulation</td>
<td>209998762</td>
<td>210001409</td>
</tr>
<tr>
<td>Error Masking Probability</td>
<td>$1.56e^{-5}$</td>
<td>$1.51e^{-5}$</td>
</tr>
<tr>
<td>Unidentified Single Cheaters</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Single Cheater Identification Rate</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Total Number of Single Cheaters</td>
<td>25001103</td>
<td>24999519</td>
</tr>
<tr>
<td>Identified One Out of Two Cheaters</td>
<td>1938</td>
<td>1869</td>
</tr>
<tr>
<td>Unidentified Double Cheaters</td>
<td>241</td>
<td>209</td>
</tr>
<tr>
<td>Total Number of Double Cheaters</td>
<td>24998897</td>
<td>25000481</td>
</tr>
<tr>
<td>Double Cheater Identification Rate</td>
<td>$1 - 4.84e^{-5}$</td>
<td>$1 - 4.57e^{-5}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Undetected Single Faults</th>
<th>Architecture Based on Robust Codes</th>
<th>Architecture Based on AMD Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Total Number Of Single Faults</td>
<td>3001257</td>
<td>3005445</td>
</tr>
<tr>
<td>Fault Masking Probability</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Undetected Double Faults</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>Total Number of Double Faults</td>
<td>3010132</td>
<td>3001089</td>
</tr>
<tr>
<td>Fault Masking Probability</td>
<td>$3.65e^{-6}$</td>
<td>$2.67e^{-6}$</td>
</tr>
<tr>
<td>Undetected Triple Faults</td>
<td>16</td>
<td>13</td>
</tr>
<tr>
<td>Total Number of Triple Faults</td>
<td>3000177</td>
<td>3001404</td>
</tr>
<tr>
<td>Fault Masking Probability</td>
<td>$5.34e^{-6}$</td>
<td>$4.33e^{-6}$</td>
</tr>
<tr>
<td>Undetected Quadruple Faults</td>
<td>44</td>
<td>39</td>
</tr>
<tr>
<td>Total Number of Quadruple Faults</td>
<td>3023004</td>
<td>3020145</td>
</tr>
<tr>
<td>Fault Masking Probability</td>
<td>$1.45e^{-5}$</td>
<td>$1.29e^{-5}$</td>
</tr>
<tr>
<td>Undetected Quintuple Faults</td>
<td>50</td>
<td>47</td>
</tr>
<tr>
<td>Total Number of Quadruple Faults</td>
<td>3030656</td>
<td>3024223</td>
</tr>
<tr>
<td>Fault Masking Probability</td>
<td>$1.65e^{-5}$</td>
<td>$1.55e^{-5}$</td>
</tr>
</tbody>
</table>

| Fault Injection Simulation Results for Secure Shamir’s Secret Sharing Scheme Using Robust and AMD Codes Constructed over $GF(2^{16})$ |
|--------------------------|-----------------------------------|---------------------------------|
| Undetected Single Faults | Architecture Based on Robust Codes | Architecture Based on AMD Codes |
| 0                        | 0                                 |
| Total Number Of Single Faults | 3001257 | 3005445 |
| Fault Masking Probability | 0 | 0 |
| Undetected Double Faults | 11 | 8 |
| Total Number of Double Faults | 3010132 | 3001089 |
| Fault Masking Probability | $3.65e^{-6}$ | $2.67e^{-6}$ |
| Undetected Triple Faults | 16 | 13 |
| Total Number of Triple Faults | 3000177 | 3001404 |
| Fault Masking Probability | $5.34e^{-6}$ | $4.33e^{-6}$ |
| Undetected Quadruple Faults | 44 | 39 |
| Total Number of Quadruple Faults | 3023004 | 3020145 |
| Fault Masking Probability | $1.45e^{-5}$ | $1.29e^{-5}$ |
| Undetected Quintuple Faults | 50 | 47 |
| Total Number of Quadruple Faults | 3030656 | 3024223 |
| Fault Masking Probability | $1.65e^{-5}$ | $1.55e^{-5}$ |
overhead. The architecture is mainly composed of three parts, the secret reconstruction module, the error checking module, and the cheater identification module (see Figure 1). These three modules have been modeled in Verilog and synthesized in Cadence Encounter(R) RTL Compiler with the Nangate OpenCell Library revision 1.0. The designs were placed and routed using Cadence Encounter Compiler. The power and the areas of the proposed schemes were estimated under a supply voltage of 1.25V.

To identify cheaters, multiple rounds of secret reconstructions are required. For secure Shamir’s secret sharing scheme protected Shamir’s secret sharing scheme is completed in one clock cycle. For robust codes, algebraic manipulation detection (AMD) codes and m-disjunct matrices (superimposed codes). The constructions and the hardware implementations of the proposed codes are presented. The connections between the cheater identification for Shamir’s secret sharing and the classic group testing theory based on superimposed codes are established. A method for identifying cheaters based on m-separable matrices and superimposed codes is presented. We analyzed the security level of the proposed schemes under different cheating models and also ran fault injection simulation to test the security level of the proposed schemes. Results show that the proposed method can increase the security level of the system and protect the system against strong cheaters and fault injection attacks at the cost of reasonable hardware overhead.

VII. CONCLUSION

In this paper, we described a cheater detection and identification methodology for Shamir’s secret sharing scheme based on robust codes, algebraic manipulation detection (AMD) codes and m-disjunct matrices (superimposed codes). The constructions and the hardware implementations of the proposed codes are presented. The connections between the cheater identification for Shamir’s secret sharing and the classic group testing theory based on superimposed codes are established. A method for identifying cheaters based on m-separable matrices and superimposed codes is presented. We analyzed the security level of the proposed schemes under different cheating models and also ran fault injection simulation to test the security level of the proposed schemes. Results show that the proposed method can increase the security level of the system and protect the system against strong cheaters and fault injection attacks at the cost of a reasonable increase of the hardware complexity.

ACKNOWLEDGMENT

We would like to thank Pei Luo and A. Yu-Lun Lin for their valuable contributions to the sections on the detection of cheating.

REFERENCES


Zhen Wang Zhen Wang received his B.S and M.S degree in Information and Communication Engineering from Zhejiang University, China in 2006. Since the summer of 2006, he started working towards his PhD degree at Boston University under the supervision of Prof. Mark G. Karpovsky and Prof. Ajay Joshi. His research is focused on the design of robust codes and their variations for building reliable and secure devices, e.g. secure cryptographic devices and reliable memories. He also conducts research to investigate the influence of nano-scale technologies on the reliability of modern digital systems, etc. He is now working for Mathworks.

Mark Karpovsky Mark Karpovsky (M’80–SM’84–F’91) has been a professor of computer engineering and the director of the Reliable Computing Laboratory, Department of Electrical and Computer Engineering, Boston University, Massachusetts, since 1983. Before joining Boston University, he taught at the State University of New York, Binghamton, and Tel-Aviv university, Israel. He was a visiting professor at the University of Dortmund, Germany, the Ecole National Superieure des Telecommunications, Paris, France, and the New Jersey Institute of Technology, Newark. He has been a consultant for IBM, Digital Corporation, Honeywell Corporation, AT&T, Raytheon, and several companies in Europe. He has published more than 200 papers and several books in the areas of logical design, testing and diagnosis of computer systems, fault-tolerant computing, error-correcting codes, and computer communication networks. He conducts research in the areas of design, testing and diagnosis of computer networks, message routing for multiprocessors and computer communication networks, and design of cryptographic devices resistant to side-channel attacks. He recently published together with R.S. Stankovic and J.T. Astola the book “Spectral Logic and its Applications for the Design of Digital Devices” by Wiley & Sons. He is a life Fellow of the IEEE.

Lake Bu Lake Bu received his B.S degree in Electrical Engineering from Wuhan University, China in 2006 and his M.S degree in Computer Engineering from Boston University in 2009. Since 2013 he has been working as a PhD candidate under Prof. Mark G. Karpovsky in Boston University. His research is focused on developing new error correcting codes for the reliable systems requiring strong fault tolerance and low overhead. He also does research on using non-linear codes to enhance the security of various devices by detecting artificial fault injections from attackers.