A Hybrid Self-diagnosis Mechanism with Defective Nodes Locating and Attack Detection for Parallel Computing Systems[1]

Lake Bu, *Student Member, IEEE*, Mark Karpovsky, *Life Fellow, IEEE*
Reliable Computing Laboratory, Electrical and Computer Engineering
Boston University, Boston, USA

***Abstract*—In recent years parallel computing has been widely employed for both science research and commercial applications. For parallel systems such as many-core or computer clusters, it is inevitable to have one or more computing node failures due to random errors or injected attacks. Usually a diagnosis mechanism is able to locate several defective nodes through a number of tests and the analysis of those test signatures (syndromes). Although this covers the cases caused by random errors, sophisticated attacks are still able to manipulate the outputs of each node, so that they will be masked and pass the diagnosis. Therefore in this paper we propose a hybrid self-diagnosis mechanism. We adopt a new type of analysis with the linear syndromes, which are able to locate up to a certain number of defective nodes caused by random errors. In addition to this, we introduce a new type of robust analysis of the non-linear syndromes, which is capable of detecting the attacks undetectable by the linear syndromes at a probability close to one. Moreover, since this hybrid self-diagnosis mechanism is on the data level which makes little distinction among different operating systems and programming languages, it can be migrated onto any other platforms conveniently.**

Index terms — ***self-diagnosis, error locating, error detection, attack detection, OLSC, robust codes, coding theory, parallel computing, linearity, nonlinearity.***

I. Introduction

The speed of microprocessors has increased rapidly from the mid 1980s to the early 2000s at a growth of 50% per year. However, from then on the growth has decreased to less than 20% per year due to the limitation of clock frequency and the transistor integration [1]. Therefore, instead of building faster single processors, assembling multi-processor parallel systems has become a mainstream solution. It also yields a better performance on power consumption and heat generation. This solution is ubiquitous in the modern computing realm from smart phones, to multi-core CPUs and GPUs, and to large computing clusters such as the recent headline maker Google AlphaGo which uses parallel computing in the implementation and training of its deep neural network.

Parallel computing breaks down a task to many independent threads for its processors which operate concurrently. Since parallel computing involves many cores or processors (nodes), it is unavoidable to have one or more erroneous nodes due to random errors or even injected attacks. Thus it is crucial to have a self-diagnosis system to identify the errors.

The "straightforward" method for self-diagnosis consists of a test pattern generator and stored reference values of all correct test results. During a test session it applies a test to all N computing nodes and compare their outputs with the reference values one by one to locate the defective nodes.

However for the most common cases, the number of defective nodes are much smaller than the total number of nodes for any given test session. Therefore advanced self-diagnosis would compress the reference values of all N correct outputs into reference signatures (syndromes) of a much smaller size. The reference signatures are computed by multiplying the N reference values by an $(A \times N)$ check matrix M of an m-error correcting code (ECC) with Hamming distance $2m+1$. During a test session, the self-diagnosis system takes the test outputs and re-computes the syndromes to be compared with the reference syndromes. The inequality of the two syndromes indicates the existence of errors. Then the error locating algorithm of the ECC can locate up to m defective nodes. If $A \ll N$, the compression will result in a noticeable saving in storage space for reference values.

However, the self-diagnosis system based on this type of linear data compression has several limitations. Firstly if there are more than $2m+1$ defective nodes, it is possible, though not highly probable, that they will not even be detected. More dangerously, if an attacker knows the matrix M, then a set of carefully selected errors can be injected to the nodes, so that they will always pass the syndrome analysis. This type of attack can succeed with probability of 100%.

Therefore in this paper we propose a new hybrid self-diagnosis mechanism. It consists of analyses of both linear and non-linear syndromes. The major contributions of the diagnosis mechanism are:

- It proposes a new type of syndromes linearly compressed from the test results. Its analysis algorithm costs much less time and storage space then other methods such as the straightforward method and Reed-Solomon codes based method;
- It combines with the linear syndrome a non-linear syndrome, by which it is able to detect the errors invisible to the linear syndromes.

The rest of the paper is organized as follows. Sections II establishes a model of parallel computing system to help demonstrate the proposed mechanism. Section III introduces the analysis algorithm on the linearly compressed syndromes. Section IV explains the vulnerability of this analysis. Section V is on the analysis of the non-linearly compressed syndrome to detect attacks. Section VI describes the work flow of the entire hybrid mechanism and summarizes its advantages.

[1]This work was sponsored by the NSF grant CNS 1012910.

II. A Model of Parallel Computing System

Parallel computing systems can scale variously from two computing nodes to thousands of nodes. The world's fastest supercomputer Tianhe-2 has as many as 16,000 nodes.

For the sake of illustration and analysis, in this paper we take a 25-node parallel computing system named *PCS*-25 shown in Fig. 1 as the study example.

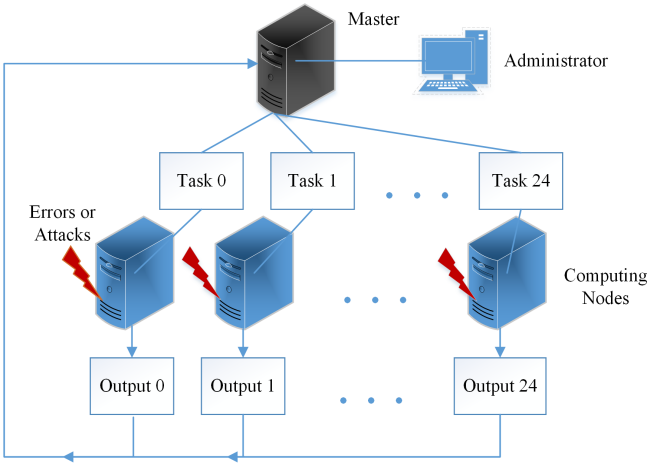


Fig. 1: A 25-node parallel computing system processing up to 25 tasks simultaneously. All the outputs of either computing projects or tests will be sent back to the Master for futher processing.

Our proposed hybrid self-diagnosis procedure is as follows:

- The Master's diagnosis system will generate a test for all the nodes to compute. The returned results are possibly affected by random errors or injected attacks;
- The test results of all the nodes are linearly compressed into A syndromes (linear syndromes) to be analyzed for locating up to m defective nodes;
- The test results are also non-linearly compressed into a one digit syndrome (non-linear syndrome) to verify if there are more faulty nodes, or even worse - attacks that the linear analysis fails to discover.

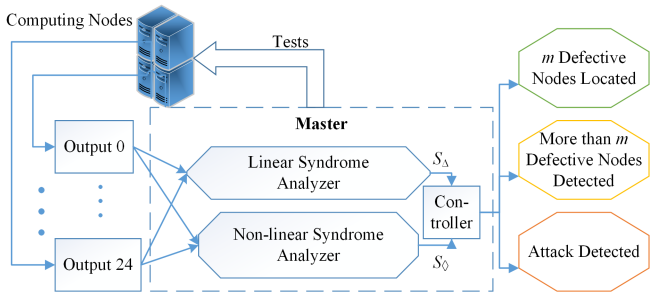


Fig. 2: The proposed hybrid self-diagnosis can be launched during any idle periods. It is able to locate up to m defective nodes caused by random errors with a probability of 100%, and detect hidden attacks with a probability close to 100%.

To help describe and evaluate this hybrid self-diagnosis mechanism, we introduce the following notations:

- N: the total number of computing nodes;
- m: the number of nodes distorted by random errors;
- e: the random errors or injected attacks to all the N nodes: $e = (e_0, e_1, \cdots, e_{N-1})$;
- $\tilde{v}$: the N-digit test outputs probably distorted by errors or attacks such that: $\tilde{v} = (\tilde{v}_0, \cdots, \tilde{v}_{N-1}) = v \oplus e$, where v is the reference outputs and $\oplus$ is the bitwise XOR;
- A: the number of components in a linear syndrome;
- S_A: the A-digit linear reference syndrome vector;
- S_ω: the one digit non-linear reference syndrome;
- b: the number of bits of each node's output.

In order to study the most general cases in parallel systems, we base our research on two assumptions:

1) For random errors, the most probable cases are $m \ll N$. The cases when m is large is much less probable;
2) For attacks, an attacker is able to manipulate the outputs of any number of nodes but not the timing of tests.

III. Diagnosis with Linear Syndromes for Defective Nodes Locating

In this section we will propose our diagnosis for the most common case that the outputs of m nodes ($m \ll N$) in the parallel computing system are distorted by random errors.

This diagnosis is based on the syndromes compressed by Orthogonal Latin Squares (OLS). It is modified from Hsiao's Orthogonal Latin Square Codes (OLSCs) [2].

A. Hsiao's Binary OLSCs

An OLSC code has $N = q^2$ information bits and $A = 2mq$ redundant (parity) bits, where q is an integer and m is the number of errors to be corrected. This makes the overall length of an OLSC codeword $(N + A)$ bits. The OLSCs use majority voting to correct the m errors in the N information bits only, and leave the A redundant bits uncorrected. It uses more redundant bits than other codes such as BCH or Reed-Solomon, but achieves a much lower decoding complexity [3].

The $(2mq \times (q^2 + 2mq))$ binary check matrix $H = \{H_{j,i}\}$ for an m-error correcting OLSC code is as follows.

$$H = \left| \begin{array}{c|c} \begin{matrix} M_0 \\ M_1 \\ \vdots \\ M_{2m-1} \end{matrix} & I_{2mq} \end{array} \right|$$

The $q \times q^2$ sub-matrices $M_0, \cdots, M_{2m-1}$ are derived from Orthogonal Latin Squares of size $q \times q$ and I_{2mq} is a $(2mq \times 2mq)$ identity matrix. Thus the first N columns in M have $2m$ ones each and the last $2mq$ columns have only 1 one each.

Also, if λ denotes the maximum number of ones in common between any two columns of H, then $\lambda = 1$ in H.

Denote an A-bit binary syndrome vector $S = H \cdot \tilde{v}$ where $S = (S_0, S_1, \cdots, S_{A-1})$, $\tilde{v}$ as the distorted $(N + A)$-bit codeword and $\tilde{v} = v \oplus e = (\tilde{v}_0, \tilde{v}_1, \cdots, \tilde{v}_i, \cdots, \tilde{v}_{N+A-1})$, $\tilde{v}_i \in GF(2)$, where v is the legal codeword, and e is the error vector. By the structure of H, any information bit $\tilde{v}_i \in \{\tilde{v}_0, \cdots, \tilde{v}_{N-1}\}$ participates in the computation of $2m$ corresponding syndromes $\{S_{k_0}, \cdots, S_{k_{2m-1}}\}$ indexed by:

$$H_{k_0,i} = H_{k_1,i} = \cdots = H_{k_{2m-1},i} = 1.$$

If $E = \{S_j | S_j = 1\}$, $S_j \in \{0, S_{k_0}, \cdots, S_{k_{2m-1}}\}$. Then:

$$e_i = majority\{0, S_{k_0}, S_{k_1}, \cdots, S_{k_{2m-1}}\}$$

where the majority voting function is defined as:

$$majority\{0, S_{k_0}, \cdots, S_{k_{2m-1}}\} = \begin{cases} 1, & \text{if } |E| \geq m+1; \\ 0, & \text{else.} \end{cases}$$

The principle of majority voting can be formulated as the following: for m errors (or less), when one error distorts a given bit, because $\lambda = 1$, the remaining $(m-1)$ errors can at most affect $(m-1)$ out of $(2m)$ syndrome bits corresponding to the given bit. Therefore, still a majority of $(m+1)$ syndrome bits will indicate the error's magnitude in the given bit.

B. Analysis of Non-binary OLS-based Linear Syndromes

Inspired by binary OLSCs, we propose a new type of syndromes for defective nodes locating. These syndromes are linearly compressed from test results by Orthogonal Latin Square (OLS) matrices. However, we have two major modifications from the OLSCs: 1) error locating with reference syndromes and without redundant digits; 2) majority voting for non-binary codewords.

For our proposed diagnosis, since reference syndromes are stored to verify the correctness of the test outputs, then there is no need of the redundant digits which are designed for the same reason. Therefore our proposed linear compression will only use the $(A \times N)$ sub-matrix M from H:

$$M = \left| \begin{array}{c} M_0 \\ M_1 \\ \vdots \\ M_{2m-1} \end{array} \right|. \tag{1}$$

OLSCs' majority voting algorithm is commonly used for binary codewords. It has been applied to non-binary codewords in the way of interleaved decoding [4], whose essence is still concurrent binary majority voting. However here we will generalize majority voting to non-binary fields.

Definition 3.1 For a set X with t non-binary elements and $X = \{x_0, x_1, \cdots, x_{t-1}\}$, if $E_k = \{x_j | x_j = x_k\}, x_j, x_k \in X$, then the majority voting function is defined as follows:

$$majority\{x_0, \cdots, x_{t-1}\} = \begin{cases} x_k, & \text{if } |E_k| \geq \lceil t/2 \rceil; \\ 0, & \text{else.} \end{cases}$$

$Note$: an efficient algorithm searching for E_k can be found in [5] with the time complexity $O(t)$.

By this definition, for a non-binary N-digit codeword $v = (v_0, v_1, \cdots, v_{N-1})$ whose check matrix is an $(A \times N)$ OLS matrix M, if v is distorted by $e = (e_0, e_1, \cdots, e_{N-1})$ into $\tilde{v}$, such that $\tilde{v} = v \oplus e$, where $v_i, e_i, \tilde{v}_i \in GF(2^b), b \geq 1$, and if the syndrome vector is $S = M \cdot \tilde{v}$, where $S = (S_0, S_1, \cdots, S_{A-1})$, and $S_i \in GF(2^b)$, then base on S:

$$e_i = majority(0, S_{k_0}, S_{k_1}, \cdots, S_{k_{2m-1}}).$$

Similarly $(S_{k_0}, \cdots, S_{k_{2m-1}})$ are indexed by the condition:

$$M_{k_0,i} = M_{k_1,i} = \cdots = M_{k_{2m-1},i} = 1.$$

With these two modifications, we have the following theorem to locate defective nodes.

Theorem 3.1: For a b-bit parallel computing system with N nodes, of which m nodes are distorted by random errors, these the m nodes can be located by the following procedure:

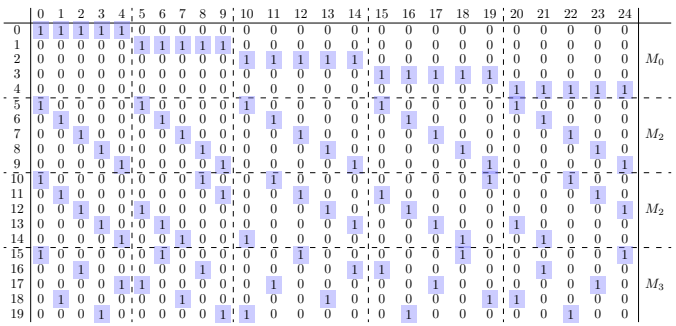
1) The self-diagnosis system has in storage a test and a reference syndrome $S_A = M \cdot v$, where M is an $(A \times N)$ binary OLS matrix and $v = (v_0, v_1, \cdots, v_{N-1}), v_i \in GF(2^b)$ is the the correct N-digit output;
2) During any idle or diagnosis period, the test is applied to all the N nodes and the testing results are denoted by an N-digit vector $\tilde{v} = (\tilde{v}_0, \tilde{v}_1, \cdots, \tilde{v}_{N-1})$;
3) $\tilde{v}$ is then compressed to an A-digit syndrome $\tilde{S}_A = M \cdot \tilde{v}$;
4) The non-binary majority voting based on the syndrome difference $S_\triangle = \tilde{S}_A \oplus S_A$ locates all m defective nodes.

C. An Example with the PCS-25 System

Example 3.1: In the *PCS*-25 system, nodes 7 and 11 are distorted by random errors. The size of each node's output is $b = 3$ bits. The self-diagnosis system is able to locate up to 2 defective nodes ($m = 2$) based on $S_\triangle$. It has in storage a test and its reference syndrome:

$$S_A = (110, 100, 110, 100, 001, 111, 011, 110, 100, 111, 110, 111, 010, 100, 110, 111, 001, 101, 000, 010)$$

$S_A = M \cdot v$ is computed based on the correct result v of this test and an $(A \times N)$ OLS matrix M, where $N = q^2 = 25, A = 2mq = 20$:



During a test session, the self-diagnosis system applies the test to *PCS*-25, and has the returned 25-digit output:

$$\tilde{v} = (011, 100, 101, 100, 000, 001, 011, 110, 001, 011, 110, 101, 101, 100, 001, 010, 010, 011, 010, 101, 001, 110, 001, 111, 000)$$

To locate the defective nodes, firstly the self-diagnosis system compresses the outputs into syndromes:

$$\tilde{S}_A = M \cdot \tilde{v} = (110, 110, 011, 100, 001, 111, 110, 100, 100, 111, 011, 111, 010, 100, 100, 111, 001, 000, 010, 010)$$

Then it is compared with the stored reference syndrome:

$$S_\triangle = (S_{\triangle 0}, \cdots, S_{\triangle 19}) = \tilde{S}_A \oplus S_A = (000, 010, 101, 000, 000, 000, 101, 010, 000, 000, 101, 000, 000, 000, 010, 000, 000, 101, 010, 000)$$

Then the linear test analyzer majority votes for each output to locate the errors. For example, for the error e_0 on $\tilde{v}_0$, since $M_{0,0} = M_{5,0} = M_{10,0} = M_{15,0} = 1$, by 4) in *Theorem 3.1*:

$$\begin{aligned} e_0 &= majority\{000, S_{\triangle 0}, S_{\triangle 5}, S_{\triangle 10}, S_{\triangle 15}\} \\ &= majority\{000, 000, 000, 101, 000\} = (000). \end{aligned}$$

For e_7 on $\tilde{v}_7$, since $M_{1,7} = M_{6,7} = M_{13,7} = M_{15,7} = 1$:

$$\begin{aligned} e_7 &= majority\{000, S_{\triangle 1}, S_{\triangle 7}, S_{\triangle 14}, S_{\triangle 18}\} \\ &= majority\{000, 010, 010, 010, 010\} = (010) \neq (000). \end{aligned}$$

In the same way we have $e_{11} = (101)$, and $e_0 = \cdots = e_{24} = (000)$. Thus the defective nodes are node 7 and 11. ■

D. An Alternative Test Scheme for $m = 1$

For an arbitrary $m \ll N$, a corresponding OLS matrix can be generated. For example, to locate single defective nodes ($m = 1$) in the *PCS*-25 system, M's dimensions would be $N = q^2 = 25, A = 2mq = 10$. In this case it is the sub-matrix consisting of the first 10 rows of the M in *Example 3.1*.

For $m = 1$, besides using OLS matrix for M, we can also use binary Hamming check matrices [6] as an alternative. The advantage of doing so is that it can result in a shorter syndrome vector. For example, if $N = 25$, $A = \lceil \log_2(N + 1) \rceil = 5$:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1
2	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
3	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
4	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Example 3.2: In the *PCS*-25 system, node 24 is distorted by an error of (101). The stored reference syndrome for a given test is $S_A = (000, 000, 000, 000, 000)$. After the test, the returned outputs are: $\tilde{v} = (000, 000, \cdots, 000, 101)$. And so $\tilde{S}_A = M \cdot \tilde{v} = (101, 101, 000, 000, 101) = S_\triangle$. Since $S_\triangle = (101) \cdot column24$, it is easy to locate the error in node 24.

We know that $m = 1$ is a special case that both OLS and Hamming matrices apply. For $m > 1$ only OLS matrices do.

E. Evaluation of the Proposed Linear Diagnosis

It has been stated in the Introduction that this type of diagnosis saves storage spaces by storing reference values only. It also has a higher error locating speed than the other methods based on Reed-Solomon codes or BCH codes. We will evaluate our proposed linear tests in these two aspects.

1) Saving of Storage: Based on the assumption 1) in Section II, in a parallel system $m \ll N$. Also since $A = 2mq = 2m\sqrt{N}$, storing the compressed A-digit syndrome could result in a notable saving over the straightforward method, which stores all the N-digit reference values of all outputs for one-by-one comparisons.

The figure below shows the storage space needed by the straightforward method and by our proposed diagnosis with compressed reference values:

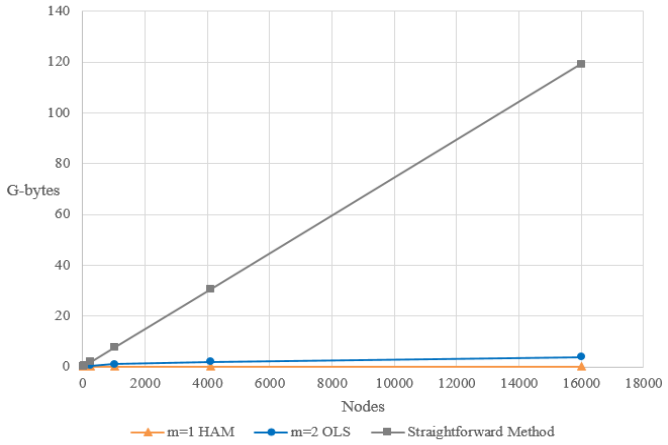


Fig. 3: Storage space needed for reference values for diagnosis on parallem systems to locate single and double defective nodes. The number of nodes ranges from 16 to 16000 and each node has a 64-bit output ($b = 64$). Totally 1 million tests are run. The comparison is made between the straightforward method of storing all N outputs as reference values for each test, and the proposed diagnosis storing only the linear syndromes by OLS or Hamming matrices.

As the number of nodes increases, the saving can be more prominent. For example, to run 1 million tests for a 64-bit 16,000-node parallel system with one or two defective nodes, the straightforward method requires 120 GigaBytes (GB) of space for the reference value. However, our proposed self-diagnosis system only needs 3.8GB for double defective nodes locating, and 104MB for single node locating.

2) Saving of Time: A self-diagnosis system with syndromes compressed by Reed-Solomon codes (RS) was proposed in [7]. In this way it only needs to store $A = 2m$ reference syndromes. However, RS' error locating algorithm has a much higher complexity than that of the majority voting. Even the efficient algorithm involves extensive finite field computations over $GF(2^b)$ [8]. For computer systems where b is 32 or 64, it will result in a much larger latency.

The figure below shows the time cost of detection of $m = 2$ defective nodes in the *PCS*-25 system by the OLS-based majority voting and the efficient RS decoding algorithm.

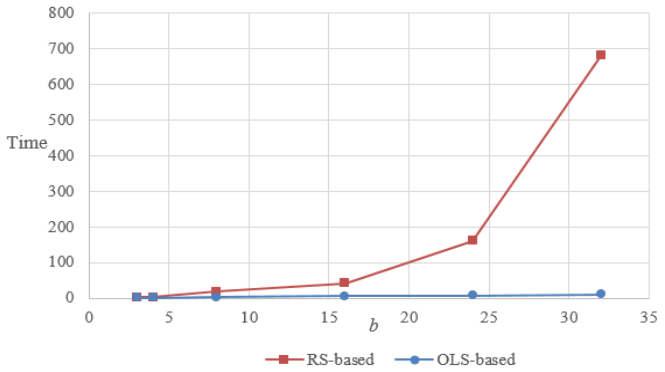


Fig. 4: Time cost on locating 2 defective nodes in the *PCS*-25 system by RS-based and the OLS-based syndrome compressions and analyses. The outputs' size b of a node ranges from 3 to 32-bit. The time cost of OLS-based syndrome compression and analysis under $b = 3$ is set to 1 as the baseline.

In Fig 4 when the nodes' output size b increases, the time cost difference between RS' decoding and majority voting increases drastically. Since during a short idle or test period, the self-diagnosis system needs to finish thousands or millions of tests, the considerable latency of RS decoding hinders it.

IV. Attacks That Can Never Be Detected by Linear Syndromes

In the previous section, *Theorem 3.1* has provided a convenient algorithm of locating up to m defective nodes in a parallel system. However by the assumption 2) in Section II, an attacker can inject errors to any number of nodes. In addition, if he or she has known M, the attacker can inject a type of errors that will never be detected by the proposed linear diagnosis.

A. The Design of Invisible Attacks

According to step 4) in *Theorem 3.1*, the defective nodes locating is based on the syndrome difference:

$$S_\triangle = \tilde{S}_A \oplus S_A.$$

If $\tilde{S}_A = S_A$ then $S_\triangle = \tilde{S}_A \oplus S_A = 0$. It indicates that there is no error. However, since:

$$\tilde{S}_A = M \cdot \tilde{v}; \qquad \tilde{v} = v \oplus e; \qquad S_A = M \cdot v;$$

We have:

$$\tilde{S}_A = M \cdot (v \oplus e) = (M \cdot v) \oplus (M \cdot e) \\ = S_A \oplus (M \cdot e).$$

If an attacker injects a carefully selected error e such that:

$$M \cdot e = 0; \qquad (e \neq 0) \tag{2}$$

Then: $\tilde{S}_A = S_A \ \rightarrow \ S_\triangle = \tilde{S}_A \oplus S_A = 0$.

Due to the self-diagnosis system's linearity, it will report no error when $e \neq 0$. Actually any e satisfies (2) will never be detected by the algorithm described in *Theorem 3.1*.

In fact, this is the hazard for all self-diagnosis systems based on linear compression of syndromes.

B. An Example of an Invisible Attack to the Linear Diagnosis

We will take the *PCS*-25 as an example to demonstrate how a successful attack makes itself invisible to the linear tests.

Example 4.1: Similar to *Example 3.1*, in the *PCS*-25 system, the size of each node's output is $b = 3$ bits. The self-diagnosis system is able to locate up to 2 defective nodes ($m = 2$). It has in storage a test session and its reference syndrome S_A which is the same as in *Example 3.1*:

$$S_A = (110, 100, 110, 100, 001, 111, 011, 110, 100, \\ 111, 110, 111, 010, 100, 110, 111, 001, 101, 000, 010)$$

$S_A = M \cdot v$ is computed based on the same M and the correct result v of this test:

$$v = (011, 100, 101, 100, 000, 001, 011, \\ 100, 001, 011, 110, 000, 101, 100, 001, 010, 010, \\ 011, 010, 101, 001, 110, 001, 111, 000)$$

Now an attacker has acquired M and computed e by (2):

$$e = (111, 000, 000, 000, 111, 000, 000, \\ 111, 111, 000, 000, 000, 111, 000, 111, 000, 000, \\ 000, 000, 000, 111, 000, 000, 111, 000).$$

The distorted outputs of *PCS*-25 will then be:

$$\tilde{v} = v \oplus e = (100, 100, 101, 100, 111, 001, 011, \\ 011, 110, 011, 110, 000, 010, 100, 110, 010, 010, \\ 011, 010, 101, 110, 110, 001, 000, 000)$$

As $\tilde{S}_A$ is computed, it will be the same as S_A:

$$\tilde{S}_A = M \cdot \tilde{v} = (110, 100, 110, 100, 001, 111, \\ 011, 110, 100, 111, 110, 111, 010, 100, \\ 110, 111, 001, 101, 000, 010) \\ = S_A.$$

Since $S_\triangle = \tilde{S}_A \oplus S_A = 0$, this error will not be seen. ■

Remark 4.1: For a b-bit parallel computing system with N nodes which employs the self-diagnosis based on the linear compression of syndromes by an $(A \times N)$ matrix M, then there are $(2^b)^{N-R}$ errors of this type that can never be detected. $R \leq A$ is the rank of M.

Therefore, linear diagnosis is not sufficient anymore. We will need a hybrid self-diagnosis system with both linear and non-linear diagnosis in order to achieve both reliability and security.

V. Non-Linear Tests Detecting Invisible Attacks

In response to the invisible attacks, we will design a diagnosis based on non-linear syndromes.

Definition 5.1 $C \subseteq GF(2^N)$ is the set of N-bit codewords and M is an $(A \times N)$ matrix. C is defined by $C = \{c | M \cdot c = 0\}$. Set K_d is called the Kernel of C if:

$$K_d = \{e | e \oplus c \in C, \ \forall c \in C\}.$$

If C is linear, then $K_d = C$.

For the cases in this paper, Kernel K_d is the set of all attacks that satisfy (2) and mask themselves in all diagnosis by majority voting. As stated in *Remark 4.1*, the linear diagnosis have a large invisible error set K_d.

Therefore in addition to the linear syndromes, we will introduce a new type of syndromes non-linearly compressed by Robust codes whose Kernel $K_d = 0$ [9]. Thus no error is able to mask itself for all diagnoses. Robust codes are often used in cryptosystems for its high security attribution [10].

The Robust Code compresses the reference test results into a one-digit reference syndrome by its encoding equation [11]:

$$S_\omega = \begin{cases} \bigoplus_{i=0}^{(N-2)/2} (v_{2i} \cdot v_{2i+1}), & N \text{ is even;} \\ v_{N-1}^3 \oplus \left[\bigoplus_{i=0}^{(N-3)/2} (v_{2i} \cdot v_{2i+1}) \right], & N \text{ is odd.} \end{cases} \tag{3}$$

(*Note*: $\cdot$ is the multiplication in finite fields. $\oplus$ is the bitwise XOR. $\bigoplus$ is the XOR sum in the sub and super scripts' range.)

This precomputed S_ω will serve as the non-linear reference syndrome to verify if $\tilde{v} = v$ when $\tilde{S}_A = S_A$.

A. Attack Detection Probability by S_ω

We will firstly study the error masking equation for S_ω [12]. We denote the Robust non-linear syndrome computed from the test results as $\tilde{S}_\omega$, and the syndrome difference between $\tilde{S}_\omega$ and reference S_ω as $S_\diamond$. By assumption 2) in Section II, the attacker can manipulate the outputs of any number of nodes. Therefore taking *PCS*-25 as an example, for a given test, in order to have $S_\diamond = \tilde{S}_\omega \oplus S_\omega = 0$:

$$\tilde{S}_\omega = (\tilde{v}_{24})^3 \oplus \left[\bigoplus_{i=0}^{11} \tilde{v}_{2i} \cdot \tilde{v}_{2i+1} \right] = S_\omega.$$

Since $\tilde{v}_i = v_i \oplus e_i$,

$$(v_{24} \oplus e_{24})^3 \oplus \left[\bigoplus_{i=0}^{11} [(v_{2i} \oplus e_{2i}) \cdot (v_{2i+1} \oplus e_{2i+1})] \right] = S_\omega. \tag{4}$$

Substituting (3) into (4), the error masking equation is:

$$(v_{24}^2 \cdot e_{24}) \oplus (e_{24}^2 \cdot v_{24}) \oplus \left[\bigoplus_{i=0}^{11} [(v_{2i} \cdot e_{2i+1}) \oplus (v_{2i+1} \cdot e_{2i})] \right] \\ = \left[\bigoplus_{i=0}^{11} (e_{2i} \cdot e_{2i+1}) \right] \oplus e_{24}^3.$$

To make a most probable attack satisfying the error masking equation, it would be making $e_0 = e_1 = \cdots = e_{23} = 0$, and $e_{24} \neq 0$. Then the above equation becomes:

$$(v_{24}^2 \cdot e_{24}) \oplus (v_{24} \cdot e_{24}^2) \oplus e_{24}^3 = 0$$

For any given e_{24}, v_{24} has at most two solutions out of 2^b possible values. And sometimes no solution can be found. Also

by Assumption 2) in Section II, the attacker has no control of the timing of the tests, therefore he/she has to take a guess for v among all possibilities. If $v_i \in GF(2^b)$, assuming all the outputs of v_i are equiprobable and denoting Q_{ATK} as the probability of a successful attack that satisfies (4), then:

$$Q_{ATK} = \frac{2}{2^b} = 2^{-b+1}. \tag{5}$$

For a parallel computing system of $b = 32$, $Q_{ATK} = 4.65 \times 10^{-10}$. Meaning for a given test, the attacker has less than one in a billion chance to inject an invisible attack successfully. Moreover, since for the Robust codes $K_d = 0$, there is no error able to mask itself for all $\tilde{v}$ except $e = 0$. Hence even if an injected error passes one diagnosis session, the next few diagnoses will definitely detect it [13]. A 64-bit parallel computing system will be even much more secure.

B. Detecting the Invisible Error from Example 4.1

Taking *Example 4.1*'s parameters, that by the correct v, $S_\omega = (111)$. While the injected error which has made $S_\triangle = \tilde{S}_A \oplus S_A = 0$, the non-linear diagnosis computes:

$$S_\diamondsuit = \tilde{S}_\omega \oplus S_\omega = (110) \oplus (111) \neq 0$$

Thus the invisible attack is detected.

C. Evaluation of the Hybrid Diagnosis Mechanism

Based on the system chart Fig. 2, the hybrid self-diagnosis system consisting of diagnosis with both linear and non-linear syndromes can be evaluated at immensely severe situations:

An attacker knows M and even the error masking equation (4), and is able to inject any errors to the outputs of all nodes. However the attacker does not know the timing of the tests thus he/she is unable to predict the correct output for each test session. Millions of tests will be run during a diagnosis period.

Under this situation, denoting $P_{ATK} = 1 - Q_{ATK}$ as the attack detection probability, Fig. 5 shows the experimental data of P_{ATK} for 1 billion test sessions. Invisible attacks are injected during every session to the PSC-25 system. However in our experiments, when $b \geq 16$, P_{ATK} is already close to 1. When $b = 32$ and $b = 64$, there was not a single attack missed by the hybrid diagnosis system among the 1 billion tests. Besides, this self-diagnosis system is capable of locating up to m defective nodes at a probability of exactly 1, protecting the system from random errors too.

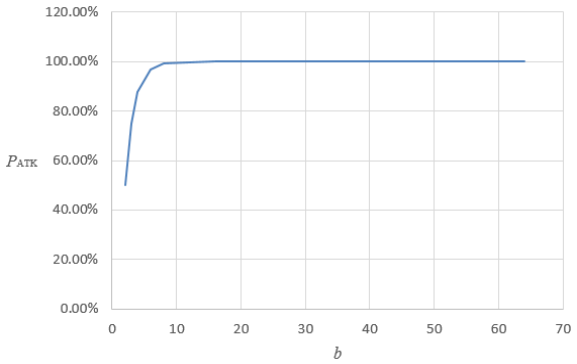


Fig. 5: Under 1 billion tests on PCS-25, the probability P_{ATK} of successfully detecting invisible attacks. The attacks are injected in every test session. When the computing nodes' output $b \geq 32$ bits, this hybrid system detected all the invisible attacks.

VI. Conclusion

In this paper we propose for parallel computing systems a hybrid self-diagnosis mechanism providing both reliability and security against random errors and injected attacks.

The diagnoses are based on analyses with linear and non-linear syndromes. The linear syndromes are compressed from test results by a selected OLS matrix M, while the non-linear syndrome is compressed by Robust Code's encoding equation. They are to be compared with the stored reference linear and non-linear syndromes. $S_\triangle$ and $S_\diamondsuit$ stand for the differences respectively. The self-diagnosis uses majority voting on $S_\triangle$ to locate up to m defective nodes, where m is determined by M. And if an attacker injects invisible errors making $S_\triangle = 0$ by (2), the diagnosis with $S_\diamondsuit$ will still detect it.

In conclusion, if $S_\triangle \neq 0$ and the majority voting is able to locate the errors, then there are up to m defective nodes; if not able then more than m defective nodes. If $S_\triangle = 0, S_\diamondsuit \neq 0$, then a probable attack is reported. If $S_\triangle = S_\diamondsuit = 0$, then there is no error. For a parallel system with b-bit outputs, the probability Q_{ATK} of an attack making $S_\triangle = S_\diamondsuit = 0$ is at most 2^{-b+1}. When $b \geq 32$, which is the case for modern processors, $Q_{ATK} \to 0$. And even if an attack happens to be so, since for Robust codes $K_d = 0$, it will be detected by another diagnosis session without exception.

Besides serving parallel computing systems with both reliability and security, since this mechanism is on the data level, it can be migrated to any platforms conveniently, such as distributed computing and cloud storage systems.

References

[1] A. Danowitz, K. Kelley, J. Mao, and J. P. Stevenson, "Cpu db: Recording microprocessor history," *Communications of the ACM*, 2012.

[2] M. Y. Hsiao, D. C. Bossen, and R. T. Chien, "Orthogonal latin square codes," *IBM Journal of Research and Development*, 1970.

[3] C. Wilkerson, A. Alameldeen, and Z. Chishti, "Scaling the memory reliability wall," *Intel Technology Journal*, vol. 17, 2013.

[4] S. Venkataraman, "A bit-interleaved embedded hamming scheme to correct single-bit and multi-bit upsets for sram-based fpgas," *24th FPL International Conference*, pp. 1–4, 2013.

[5] R. S. Boyer, "Automated reasoning: Essays in honor of woody bledsoe," *Springer Science & Business Media*, Vol. 1, 2012.

[6] L. Bu, M. G. Karpovsky, and Z. Wang, "New byte error correcting codes with simple decoding for reliable cache design," *2015 IEEE 21st International On-Line Testing Symposium (IOLTS)*, 2015.

[7] M. G. Karpovsky, L. Levitin, and F. Vainstein, "Diagnosis by signature analysis of test responses," *IEEE Transactions on Computers*, 1994.

[8] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of reed-solomon codes," *IEEE Transactions on Information Theory*, 2003.

[9] W. Zhen, M. Karpovsky, and K. J. Kulikowski, "Replacing linear hamming codes by robust nonlinear codes results in a reliability improvement of memories," *IEEE/IFIP International Conference on Dependable Systems and Networks*, 2009.

[10] G. Gaubatz, B. Sunar, and M. G. Karpovsky, "Non-linear residue codes for robust public-key arithme," *Fault Diagnosis and Tolerance in Cryptography*, 2006.

[11] K. Kulikowski, Z. Wang, and M. G. Karpovsky, "Comparative analysis of robust fault attack resistant architectures for public and private cryptosystems," *IEEE 5th Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2008.

[12] K. J. Kulikowski, M. G. Karpovsky, and A. Taubin, "Robust codes and robust, fault-tolerant architectures of the advanced encryption standard," *Journal of systems Architecture*, 53.2, 2007.

[13] M. G. Karpovsky, K. J. Kulikowski, and A. Taubin, "Robust protection against fault-injection attacks on smart cards implementing the advanced encryption standard," *IEEE International Conference on Dependable Systems and Networks*, 2004.