Error Correction at Malicious Nodes using Reed-Muller Codes

Ratnakumari Challa
Dept. of CSE
APIIIT (RGUKT)
AndhraPradesh, India

Devaraju Isuru
Dept. of CSE
JNTUK, Kakinada
AndhraPradesh, India

Kanusu Srinivasa Rao
Dept. of Computer Science
Yogi Vemana University
AndhraPradesh, India.

ABSTRACT
In the network several problems are caused due to the presence of malicious nodes. It is very important to find the malicious nodes in the network in order to eliminate the problems caused by those nodes. This paper proposes a model where Reed-Muller codes are used to find the locations of the malicious nodes and calculate the probability that a node is malicious. Based on the probability of each malicious node, the system localizes or discards the nodes which have higher error probability. Sometimes removal of a malicious node causes breakage of network into parts i.e. if it is an articulation point. It leads to the reconstruction of the network. This reconstruction process is very complex and expensive. In this case, such nodes cannot be discarded. To avoid the reconstruction of the network an algorithm is proposed to handle the malicious activity caused by an articulation point. Message tampering is the frequently occurred malicious activity in most of the networks when the communication takes place between source and destination. To handle message tampering at articulation points, this system performs error correction using Reed-Muller decoding algorithm.

Keywords
Coding Theory, Error Control Codes, Malicious Behaviour, Reed-Muller Codes.

1. INTRODUCTION
The proposed network consists of 'N' number of internal nodes. Two more nodes should be included in the network to act like source and destination. The network can be represented by G= (V, E), Where 'V' represents set of nodes including source node and destination node as $V = u_s, u_1, u_2 \dots u_N, u_d$ and 'E' represent all the edges in the network. The entire network can be represented by directed paths, So that every message is directed from source to destination. The source node takes the initiation in the network by transmitting a message to destination node. On the other hand, the destination node takes the responsibility of finding malicious nodes presented in the network. The internal nodes are vulnerable to infection, i.e. chances to become malicious while the source and destination are not.

Before sending a message, the source node must perform encoding operation on the given message. To do this a standard encryption technique was introduced in the proposed approach. This suggests that the destination node will be able to approve whether it receives an original message or a tampered message after performing decryption. So, the destination node can take an uncorrupted packet by a '0' and a corrupted one by '1'. The malicious node can tampers the original message and transmit to next node. If an internal node receiving a message from a malicious node, then it will transmit tempered message across remaining path. Finally, the destination node will determine, the message received on the path was legitimate or not. Every path is comprised of several nodes. Let us represent a malicious node by '1' and an uncorrupted node by '0', So that this information can be stored in a variable. A variable 'S' is declared as a vector containing the information about the status of each node in the network. This information can be expressed as

$$S= \{s_1, s_2, s_3, \dots, s_N\} \quad (1)$$

$$\text{Where } s_i = \begin{cases} 1 & \text{If } u_i \text{ is malicious} \\ 0 & \text{Otherwise} \end{cases}$$

The terminal node will receive different messages from different paths. The output of a particular path is OR operation of malicious status of each node. It can be denoted by a vector $\hat{c}$. The output of path i can be mathematically expressed as

$$\hat{c_i} = V_{j:u_j \epsilon P_i} s_j \quad (2)$$

The same can be represented as an addition in a Galois Field of order 2, GF (2), as

$$\hat{c_i} = \sum_{j:u_j \epsilon P_i} s_j + \sum_{j:u_j \epsilon P_i} \sum_{k>j:u_k \epsilon P_i} s_j s_k + \dots + \prod_{j:u_j \epsilon P_i} s_j \quad (3)$$

Once output vectors are known for all the paths in the network, those can be grouped and a decoding algorithm is performed on that group. This operation is carried out by the destination node and reveals the location of the byzantine nodes. In order to determine the byzantine nodes paths are coordinated carefully by the terminal node. In this paper Reed-Muller codes are used as error detection codes to reveal the locations of artificial malicious nodes. The probability of a node to become malicious can be updated after identifying the corrupted nodes. A message received by the malicious node must be tampered and forwarded to next node. If any node among those byzantine nodes is an articulation point [1] in the network, it performs error correction on the tampered message instead of removing it from the network. In order to perform the error correction at a particular articulation point, this approach must apply the reed decoding algorithm to correct the errors in the tampered message vector.

In the existing technique [2], if there is any node having more probability to become malicious then the node can be localized and discarded. Removal of such malicious node gives trouble to the network when it is an articulation point. Here the most important issue is what happened to the network if it discards that kind of malicious node. If it discards the articulation point then the network behaves like previous or it loses some connections between the nodes is another question. If it deletes that node then reconstruction of entire network is required and this operation is expensive and time consuming. Moreover, there is no guarantee to get the same network with all supportive paths after reconstruction.

This is the major problem to be solved by this approach. In the next section, the information about Reed-Muller codes is discussed. And also give brief comparisons about how this work is different from earlier attempts and followed by the Section 3 shows different approaches that are used for correcting errors.

2. BACKGROUND AND RELATED WORK

2.1 Reed-Muller Codes:

Reed Muller codes [3,4] are a family of linear error-correcting codes used in communications. Let m, r be any two positive integers with $r \leq m$, and let code length $n = 2^m$. The Reed Muller code RM(r, m) is defined [1] by the set of code words

$$RM(r, m) = \{(f(a_0), \dots, f(a_{n-1})) : f \in \mathbb{P}(r, m)\} \quad (4)$$

Where $\mathbb{P}(r, m)$ is the set of m-variate polynomials of degree at most r on $\mathbb{F}_2$, $a_0, \dots, a_{n-1}$ are all the elements of $\mathbb{F}_2^m$.

Length of the code word was defined by the parameter 'm' but the number of code words varied each time when the order was changed. The length of the code word can also be called as the dimension [3] of the code and it is denoted as k. The dimension k for RM(r, m) is defined as follows:

$$k = 1 + \binom{m}{1} + \binom{m}{2} + \binom{m}{3} + \cdots + \binom{m}{r} \quad (5)$$

Reed Muller codes are also used for detection and correction of the errors that are mainly occurred in network communications. The number of errors corrected by an RM code was depends on the parameter called as the minimum distance and is denoted as d_{min}. The minimum distance of RM (r, m) is $d_{min} = 2^{m-r}$.

The message transmitted across the network must be encoded before sending it. The length of the message need to be equal to the dimension of the code as discussed above. The message vector which is suitable for RM(r, m) code is denoted as X and defined as

$$X = (x_0, x_1, \dots, x_{k-1}) \quad (6)$$

In order to encode the message X, it defines a generator matrix for RM (r, m) with k rows and 2^m columns and it is denoted as $G_{r,m}$. All rows in the generator matrix are different with each other. Consider m binary linearly independent vectors $v_1, v_2, \dots, v_m$.

Let M be a set of all possible Boolean functions of these m vectors that are defined [1] by a monomial term,

$M = \{1, v_1, \dots, v_m, v_1v_2, \dots, v_{m-1}v_m, \dots, v_1v_2 \dots v_m\}$.

The generator matrix for RM (r, m) is defined as

$$G_{r,m} = \begin{bmatrix} 1 \\ \hline v_1 \\ \vdots \\ v_m \\ \hline v_1v_2 \\ \vdots \\ v_{m-1}v_m \\ \hline \vdots \\ v_{m-r+1} \cdots v_{m-1}v_m \end{bmatrix}$$

The encoding process gives a different code word as an outcome for each transmitted message vector respectively. The code word is denoted as C and it is formed by using generator matrix and message vector as follows:

$$C = (c_0, c_1, \dots, c_{2^m-1}) \quad (7)$$
$$= X.G_{r,m}$$

$= \sum_{i=0}^{k-1} x_i R_i$ where $R_i = i^{th}$ row of generator matrix

$$= x_0 + x_1v_1 + \cdots + x_m v_m + x_{12}v_1v_2 + \dots + x_{m-r+1\, m-r \dots m} v_{m-r+1 \dots} v_m$$

Where x_{m+1} has been treated as x_{12}, x_{m+2} has been treated as x_{13}, and so on until x_N has been treated as $x_{m-r+1 \dots m-1\, m}$.

This code word is transmitted by the source and the same code word may not be received by the destination. This may happen due to the noise and other issues affect the code word when it is in transmission. To conform whether the received code word is same as the sender or not, it needs to perform decoding operation on the receiving code word. For that consider the received code word as $R = (r_0, r_1, \dots, r_{2^m-1})$ and an algorithm was used [3] for decoding that was developed by Reed. The decoding algorithm was discussed in detail in the next section.

2.2 Review of Related work

Security and reliability are two major challenges for successful communication in wireless networks. A reliable network must be able to cope with the malicious behaviour [5] of these components. The authors in [6], developed a computer system by introducing the communication between loyal and disloyal generals through a messenger. This approach is also known as well known Byzantine Generals problem [6] and it only describes about when reliable communication take place in the network. To detect adversarial modifications in the network an information-theoretic approach was developed in [7], using random linear network coding. The method is to show how malicious packets can be encountered, but failed to find their origin in the network. The origin of malicious nodes was determined by developing [8] a secure trace route protocol. The protocol was used to detect poorly performing routers that appear doubtful in the network. The proposed approach, on the other hand, consider a standard encryption algorithm to reveal the specific locations of malicious nodes using this information at which level a node can be trusted is determined. If any node is having more probability to become malicious that should be neglected. In [8] the author introduces malicious routers misdirect the data. In this paper, it is assumed that malicious node temper the message and propagate to rest of the path in the same direction.

The malicious nodes can also be detected by using error control codes. Reed-Muller codes are used as error control codes to determine the locations [2] of the malicious nodes in the network. The probability that a node is malicious can be calculated for all the nodes that are presented in the network. The nodes with higher probability are localized or discarded from the network.

Sometimes the system may needed to discard a malicious node has very critical role in the network i.e. if it is an articulation point. Removal of such node needs additional reconstruction of the entire network. All the paths associated with this node will become illegal. To reach the same network structure it must do lot of work and it takes more time. This reconstruction work is expensive too. The proposed approach takes this problem into account and provides an appropriate solution. This method corrects the errors in the tampered

message at each articulation point in the network instead of removing those malicious nodes.

Table.1. Comparison of various approaches and their accomplishments

Method/Approach	Contributions	Failure Chances
Byzantine Generals	When Reliable communication will occur	No chance for finding malicious behaviour
Random linear network coding	How to detect malicious packets	To find the origin of malicious packets.
Secure Traceroute Protocol	Finds the origin of malicious packets.	Misdirect the packets. No Localization
Localization using reed-Muller codes	Localize and/or discard the malicious node.	No error correction for message tampering
Error correction using Reed-Muller codes (proposed approach)	Detection of malicious nodes Error correction of tampered message.	Less suitable for large networks.

3. LOCALIZATION OF MALICIOUS NODES AND ERROR CORRECTION

3.1 Combining-Paths

In this approach Reed-Muller parameters are calculated as discussed in section (2). Prior to sending a message from the source node it must map the network paths to Reed-Muller codes. To find the artificial malicious nodes from the network, it must choose the node status variable S= $\{s_1,s_2,\ldots,s_N\}$ as the message vector. For each path 'i', it gets an N-bit vector P_i which has value '1' in the place of a node if it belongs to P_i and '0' otherwise. Similarly, it gets a code vector for each path i.e. $C_i=s_0+s_1v_1(i)+\ldots+ s_mv_m(i)+s_{12}v_1(i)v_2(i)+\ldots+ s_{m-r+1\ldots m-1m}v_{m-r+1}(i)\ldots v_m(i)$ and the corresponding path vector as $P_i=(1, v_1(i),\ldots, v_m(i), v_1(i)v_2(i),\ldots, v_{m-r+1}(i)\ldots v_m(i))$. This code vector is same as in Eq. (2) and by comparing this with the code vector in Eq. (3) then it seems to be mismatch. Here the transmitted code word i.e. Eq. (2) having less terms when compared with the received code word i.e. Eq. (3). The additional terms in Eq. (3) for the path 'i' will be considered as the error in the i^{th} code word bit of the received vector. This is given by

$$\hat{c}_i=c_i+\varepsilon_i$$

where $\varepsilon_i=\sum_{j:u_j\epsilon P_i}\sum_{k>j:u_k\epsilon P_i} s_js_k+\ldots+\prod_{j:u_j\epsilon P_i} s_j$ (8)

The above expression is suitable to realizable paths only. There are several non-realizable paths are also presented in RM-paths. In this method, multiple paths are combined to form non-realizable paths. Suppose a path P_k is formed from 'n' different paths $p'_{k_1}, p'_{k_2}, \ldots, p'_{k_n}$. This suggests that

$$P_k=\sum_{i=1}^{n} p'_{k_i} \text{ and } C\hat{}_k=\sum_{i=1}^{n} c\hat{}_{k_i} \quad (9)$$

An error occurs in the k^{th} code word bit only if

$$\epsilon'_k=\sum_{i=1}^{n}\left[\sum_{j:u_j\epsilon P_i} s_j+\sum_{j:u_j\epsilon P_i}\sum_{k>j:u_k\epsilon P_i} s_js_k+\ldots+\prod_{j:u_j\epsilon P_i} s_j\right]=1$$

The probability of error can be updated after detection of error bits for each path. Initially assumed that the probability as 'p' for each node in the network.

3.1.1 Correction of Tampered Message:

The malicious nodes have one or more malicious behaviour's [5] as already discussed. The locations of the malicious nodes are revealed by the destination node after performing the decoding operation on each received code word. This system compares each malicious node to the list of articulation points in the network. If no match was found then the malicious node is localized or discarded from the network. But it cannot do the same if any matching occurs. Instead of removing the malicious node from the network, this introduces an approach to correct the tampered message at malicious node. To do this the following algorithm can be applied on the received code word $\mathbf{r}=(r_0, r_1, \ldots, r_{2^m-1})$ as given below:

Reed-Decoding Algorithm for Localization of Malicious Nodes

- **Step1**: Stars with the message bits that have the highest order r and let $j=r$ and $\hat{c}=\mathbf{r}$.
- **Step 2**: For all the $\binom{m}{j}$ of the order j message bits perform:

1) Pick message bit $m_{i_1 i_2 \ldots i_j}$.
2) Consider associated incidence vector $v_{i_1}v_{i_2}\ldots v_{i_j}$.
3) Let the subspace of points $S=\{P_{i_1}P_{i_2}\ldots P_{i_j}\}$.
4) Form the set difference $\{k_1,k_2,\ldots,k_{m-j}\}=\{1,2,\ldots,m\}-\{i_1,i_2,\ldots,i_j\}$. Let T be the complementary subspace of S of points $T=\{P_{k_1}P_{k_2}\ldots P_{k_{m-j}}\}$.
5) Form k_{m-j} translations of T by translating $\{P_{i_1}P_{i_2}\ldots P_{i_j}\}$ by each $P_{k_l}, l\in\{1,2,\ldots,m-j\}$.
6) Take one of the translations $P_{t_1}P_{t_2}\ldots P_{t_j}$, and form first message bit measure $m'^{(1)}_{i_1,i_2,\ldots,i_j}=\hat{c}_{t_1}+\hat{c}_{t_2}+\cdots+\hat{c}_{t_j}$. Do this for all the translations.
7) Let $m'_{,i_2,\ldots,i_j}=\text{maj}\{m'^{(1)}_{i_1,i_2,\ldots,i_j},\ldots,m'^{(k_{m-j})}_{i_1,i_2,\ldots,i_j}\}$ (where maj represents most occurring value).

- Step 3: If all message bits of order j have been estimated then form m'_r from the measures of the message bits of order j and do the following step. Otherwise go to step 2.
- Let j=j-1 and if j>=0, then let $\hat{c}=\hat{c}-m'_rG_r$ and go to step 2. Otherwise decoding process is done.

Tampering of messages over a path is considered as the main problem. It also include changing of bits of a message vector across a path in the network, those can be corrected by using Reed Muller codes as explained below. The decoding algorithms [9] for RM coding rely upon an interesting idea in coding, the majority logic decoding.

Consider RM (2, 4) code then it gets following observations:

Where r=2 and m=4

Let the message vector 'M' is having message bits corresponding to nodes that are presented in the network. And this vector can be represented by

$M = (m_0, m_1, m_2, m_3, m_4, m_{12}, m_{13}, m_{14}, m_{23}, m_{24}, m_{34})$

$= (M_0, M_1, M_2)$

M_k corresponds to the message bits of order 'k'

And the Generator Matrix:

$$G = \begin{bmatrix} 1 \\ \hline v_1 \\ \vdots \\ v_4 \\ \hline v_1 v_2 \\ \vdots \\ v_3 v_4 \end{bmatrix} = \begin{bmatrix} G_0 \\ \hline G_1 \\ \hline G_2 \end{bmatrix} \quad (10)$$

From the above information the code word can be formed as follows:

$$c = (c_0, c_1, \dots, c_{15}) = MG = [M_0 | M_1 | M_2] \begin{bmatrix} G_0 \\ \hline G_1 \\ \hline G_2 \end{bmatrix} \quad (11)$$

The source node sends the code word across the paths in the network and the destination node must receive another code word correspondingly. But the received code word may not be same as the original code word; meaning that it may has some errors due to the message bits are tampered by the malicious nodes. To get the original code word from the received code word the following steps has to follow:

$\mathbf{r} = (r_0, r_1, \dots, r_{15})$ ($\mathbf{r}$ is the received code word)

Determine the measures for the highest-order block of message bits, M_2. Then M_2G_2 is subtracted off from $\mathbf{r}$, leaving only lower-order code words. Then the message bits for M_1 are obtained, which are subtracted, and so forth. Measures can be obtained for each message bit by using the Reed decoding algorithm and then apply the majority logic by considering all derived measures so that it gets one message bit respectively. The key to finding the message bits comes from writing multiple equations for the same quantity, and taking a majority vote. Observe the following:

$c_0 = m_0$; $c_1 = m_0 + m_1$;

$c_2 = m_0 + m_2$; $c_3 = m_0 + m_1 + m_2 + m_{12}$

Adding these code bits together to obtain:

$$c_0 + c_1 + c_2 + c_3 = m_{12}$$

$$c_4 + c_5 + c_6 + c_7 = m_{12}$$

$$c_8 + c_9 + c_{10} + c_{11} = m_{12}$$

$$c_{12} + c_{13} + c_{14} + c_{15} = m_{12}$$

Now what it actually has to deal with is the received sequence $\mathbf{r} = (r_0, r_1, \dots, r_{15})$.

By using this in conjunction with the equations above to obtain four measures of m_{12} :

$$m_{12} = \begin{cases} m'_{12} = r_0 + r_1 + r_2 + r_3 \\ m'_{12} = r_4 + r_5 + r_6 + r_7 \\ m'_{12} = r_8 + r_9 + r_{10} + r_{11} \\ m'_{12} = r_{12} + r_{13} + r_{14} + r_{15} \end{cases}$$

From these four measures, it determines the value of m'_{12} by majority logic: only one is incorrect to get it right; two are incorrect to detect that an error has occurred. Do the same for all the message bits of order 2 in the message vector. It is similarly possible to set up multiple equations for the other second-order bits $m_{13}, m_{14}, \dots, m_{34}$. Based upon these equations it determines a measure of the entire second-order block as $M_2 = (m_{34}, m_{24}, m_{14}, m_{23}, m_{13}, m_{12})$. The system then subtracts the entire second-order block to get $\boldsymbol{r}' = \mathbf{r} - M_2G_2$.Now repeat the same for the first-order bits. Finally have eight orthogonal check sums on each of the first-order message bits. For example,

$$m_1 = \begin{cases} c_0 + c_1 \\ c_2 + c_3 \\ c_4 + c_5 \\ c_6 + c_7 \end{cases} \text{And} \qquad m_1 = \begin{cases} c_8 + c_9 \\ c_{10} + c_{11} \\ c_{12} + c_{13} \\ c_{14} + c_{15} \end{cases}$$

From eight measures obtained from the received code word, it estimates M_1 by voting. Having obtained all the bits in M_1, it removes M_1, $\boldsymbol{r}'' = \boldsymbol{r^1} - M_1G_1$ and looks for M_0. But

$$\boldsymbol{r}'' = M_0 1 + \epsilon$$

So the parity sums in this case are just the bits r_0 through r_{15}. Then repeat the same procedure for all the message bits of order 1 to estimate those bits.

3.2 RD-erasures Algorithm

In this approach non-realizable paths are treated as erasures. Here it assumes an erasure with value 'e' for the code word bit c_i^ and initialize an empty set, W=ϕ. The set W represents the message bits which are known with complete certainty. The algorithm that is proposed under this method is the combination of Reed-decoding algorithm (discussed in section 3.1.1) with erasures.

Algorithm:

1. $\forall \hat{c}_i \; i \in \{1, \dots, 2^m\}$ such that $\hat{c}_i = 0 \& \forall s_{i_1 i_2 \dots i_j} \in P_i$ such that $\hat{s}_{i_1 i_2 \dots i_j} = 0$ and add $\{i_1 i_2 \dots i_j\}$ as a set into W. It means that if $\{i_1 i_2 \dots i_j\}$ is in W that does not necessarily imply that i_1 is in W.
2. $\forall \hat{c}_i \; i \in \{1, \dots, 2^m\}$ such that $\hat{c}_i = e$ check if $\{i_1 i_2 \dots i_j\} \in W$, $\forall s_{i_1 i_2 \dots i_j} \in P_i$. If so then set $\hat{c}_i = 0$.
3. Let j=r (highest order)
4. Start with step 2 of Reed-decoding algorithm and proceed down till the completion of all the code word bits.

4. RESULTS AND DISCUSSIONS

There are several paths are presented between source and destination in the network. But the proposed method takes care about only the paths derived from the generator matrix of the [10] Reed-Muller code. All RM-paths are categorized into realizable paths and non-realizable paths. By considering these paths two different methods are proposed in this work. One method is called "combining-paths" uses to reed decoding algorithm to find malicious nodes and to correct error bits at malicious nodes. In this approach non-realizable paths are formed by combining multiple paths between source

and destination. In the next method non-realizable paths are treated as erasures and uses RD-Erasures algorithm to form a message estimate. The majority logic decoding is followed by both approaches while estimating a message bit.

Consider that the number of nodes/message bits is 11. This number can also be used as dimension for the Reed-Muller code R (m, r). The Reed-Muller parameters can be calculated from the dimension of the code and they are m=4, r=2. Similarly total codeword bits are given by 2^m i.e. 16. Each codeword bit has must be decoded at the destination node. But there are the chances of getting erroneous results like non malicious nodes are mistakenly treated as malicious and vice versa. Similarly, the destination node will assume correct bits as errors and vice versa. The probability of getting errors at destination node can be calculated based on the minimum distance of Reed-Muller codes, i.e. $d_{min} = 2^{m-r}$. The code with minimum distance d_{min} can correct up to $\left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ errors. The probability of getting this wrong decoding is also called as bit error probability and it can be calculated as:

Bit Error Probability =1-Probability of correctly decoding each bit

$$= 1 - \sum_{i=0}^{\left\lfloor \frac{d_{min}-1}{2} \right\rfloor} \binom{N}{i} p^i (1-p)^{N-i}$$

Where $N=2^m$ (N will be reduced if any erasures are used)

Next, it has to specify a node connectivity matrix consisting of possible paths.

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

All of these paths may not be used in RM-paths. In particular there are 16 RM-paths are classified as 12 realizable paths and 4 non-realizable paths. In the proposed setup always the first node has a connection with the source node. Similarly the last node in each row vector having the value '1' is directly connected to the destination node. By taking only realizable paths a matrix is proposed as shown below:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

For the "combining-paths" technique all the simple paths [11] required to form each non-realizable path must be found. Similarly, RD-erasures algorithm can also be used to reveal the original message bits from a tampered message vector. In this method it assumes that non-realizable paths as erasures. The number of erasures need to assume is changed for each different message. After completion of each decoding operation will give certain number of bit errors. Then the bit error probability is calculated over different malicious node probabilities for each approach and listed as a table below:

Table.2. Comparison of bit error probabilities for different number of erasures

Malicious probability of single node (p)	Combining paths	RD Erasures algorithm (proposed)	
	No erasures	2- erasures	4-erasures
0.01	0.01093	0.00840	0.00617
0.02	0.03986	0.03103	0.02310
0.04	0.13266	0.10593	0.08093
0.06	0.248945	0.20369	0.15954
0.08	0.370145	0.30996	0.24868
0.10	0.48527	0.41537	0.39465
0.12	0.58847	0.51413	0.43141
0.14	0.67727	0.60305	0.51658

From the above table, it states that the probability of a node being malicious is increases leads to the more bit error probability. The same information can also be represented as a graph having different plots, i.e. one for combining-paths technique and other for RD-erasures technique. Fig.1 shows the bit error probability over different malicious node probabilities for the methods that are discussed in section 3. As it already describes the major difference occurred between these two approaches while considering non-realizable paths of the network. These paths may changed depends on the node connectivity matrix of the network. It is more suitable for RM codes with degree 2.

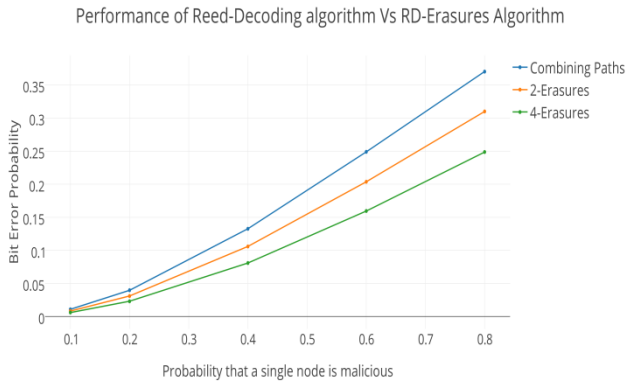

Fig.1: Comparing the "combine-paths" technique with RD-erasures algorithm

For the "combining-paths" technique, paths are combined to form non-realizable paths. Similarly RD-erasures algorithm handles the missing paths as erasures. From the above figure it is better to assume non-realizable paths as erasures rather than forming those non-realizable paths from realizable paths.

5. CONCLUSION

In this paper, it has taken the problem of correcting the error bits of the message vector, and localizing the malicious nodes in a wireless network. It has shown the application of Reed-Muller codes for correcting the errors that are frequently occurred. The parameters of Reed-Muller code are derived from the network elements and those are used to estimate the number of errors to be corrected. Based on these parameters it forms a generator matrix and it shows how the network paths are founded from the generator matrix. For the case of non-realizable paths, it derives an algorithm to correct errors in an efficient manner. It compares the performance over different approaches and shown that the proposed algorithm achieves the lowest bit error probability. Message tampering was solved by using the Reed-Muller codes. By using these codes, similar kinds of malicious acts such as packet drop, denial of sending etc can be solved or not is leaving as the future scope.

6. REFERENCES

[1] Aho AV, Hopcroft JE. The design and analysis of computer algorithms. Pearson Education India; 1974 Sep 1.

[2] Kacewicz A, Wicker S. Application of Reed-Muller codes for localization of malicious nodes. In Communications (ICC), 2010 IEEE International Conference on 2010 May 23 (pp. 1-7). IEEE.

[3] Cooke B. Reed-Muller error correcting codes. MII Undergraduate J. Math. 1999; 1:21-7.

[4] Wicker SB. Error control systems for digital communication and storage. Englewood Cliffs: Prentice hall; 1995 Jan 15.

[5] Saini R, Khari M. Defining malicious behavior of a node and its defensive methods in Ad Hoc network. International Journal of Computer Applications. 2011 Apr; 20(4):18-21.

[6] Lamport L, Shostak R, Pease M. The Byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS). 1982 Jul 1; 4(3):382-401.

[7] Ho T, Leong B, Koetter R, Médard M, Effros M, Karger DR. Byzantine modification detection in multicast networks using randomized network coding.

[8] Padmanabhan VN, Simon DR. Secure traceroute to detect faulty or malicious routing. ACM SIGCOMM Computer Communication Review. 2003 Jan 1; 33(1):77-82.

[9] Todd K. Moon, Lecture 9 Reed Muller Codes, ECE 7670", 1 April 2006 (2006-04-01), pages1-8, XP55004016.

[10] Reed IS. A CLASS OF MIJLTIPLE-ERROR-CORRECTING CODES AND THE DECODING SCHEFE.

[11] Rubin FR. enumerating all simple paths in a graph. IEEE Transactions on Circuits and Systems. 1978 Aug; 25(8):641-2.