Securing Sensor Network Using Polynomial Based Route Filtering

Prajakta J. Patil[1], Sunil G.Tamhankar[2]
[1]Dept of Electronics
[2]Assistant Professor, Dept of Electronics
[1, 2] Walchand College of Engineering, Sangli, Maharashtra 416415, India

Abstract- *In wireless sensor networks, adversary can inject false measurement reports into the controller through compromised sensor nodes. This will not only threaten the security of the system, but also consume network resources. So, detection of compromised nodes is very important. To deal with this issue, a Polynomial-based Route Filtering scheme (PRF) is proposed, which can filter false injected data effectively and achieve a high resilience to the number of compromised nodes without completely relying on static routes and node localization. Each node stores two types of polynomials: authentication polynomial and check polynomial, and used for authenticating and verifying the measurement reports to achieve high resilience.*

Keywords- Polynomial based Route Filtering, Message Authentication Codes, Cyber Physical Networks etc.

I. INTRODUCTION

1.1 Introduction to CPNS

Cyber-physical networked systems (CPNS) consisting of sensor nodes, actuators, controller, and wireless sensors, have been widely used to monitor and affect local and remote physical entities in the physical world. Monitoring and controlling physical systems through distributed sensors and actuators have become important tasks in various applications. In CPNS, sensor nodes obtain the measurement from the physical components, process the measurements and send measured data to the controller through networks. A Wireless Sensor Network (WSN) is composed of a large number of sensor nodes having limited computation capacity, limited memory space, limited power resource and short-range of communication. WSN has one or more base-station which does the functions of calculation and decision-making. In military applications, sensor nodes may be deployed in hostile environments such as battlefields to monitor the activities of enemy forces. In these scenarios, sensor networks may suffer different types of malicious attacks. One type is called false report injection attacks, in which adversaries inject into sensor networks the false data reports that contains nonexistent events or false readings from compromised nodes.

1.2 Problems with CPNS

As sensor nodes in CPNS are tiny, all sophistication like tamper detection are not implemented with the existing schemes and increases the chance of being compromised by adversaries. For example, the adversary can use the wireless devices to connect to the CPNS and compromise or physically capture sensor nodes through code injection attacks or node replication attacks, in which a number of compromised nodes can be controlled by the adversary throughout the sensor network and CPNS. The adversary can inject false measurement reports into the controller through compromised nodes. This causes the controller to estimate wrong system states and it can cause dangerous threat to the system.

To filter the false data, a number of schemes have been designed in the past. And they have few limitations. In PCREF [1] static path is required. In SEF [5] Collaborative Filtering of false report is not possible. Dynamic En-Routing scheme [7] is more complicated since a lot of control messages can be introduced and reports get delayed.

II. BACKGROUND

2.1 Security Threats and Goals

As security is one of the main issues in CPNS, information and resources should be protected over the network.

Poor quality security of CPSN devices led to an increase in:

1. Compromised-Key Attack
2. Eavesdropping
3. Man-in-the-Middle Attack
4. Denial-of-Service Attack etc.

Security is an important factor as it helps user perceive control over information and not vice versa. Below are the main security goals:

- ***Confidentiality*:** Confidentiality means that the information is available or accessible to the authorized users only. It is the most important security goal. To achieve confidentiality Encryption with security key is used.
- ***Availability*:** Data should be available to the authorized user whenever needed despite of any internal or external attacks i.e. DoS attack.
- ***Integrity*:** Data should not be altered or manipulated by adversary as it travels from sender to the recipient.
- ***Authentication*:** Data originates from the identified sender with which the node is communicating in the network.
- ***Authorization*:** Network services or resources can only be accessed by authorized nodes. Any loss of security in the CPNS systems may have real and direct consequences on efficiency and safety.

2.2 The Basis of En-Route Filtering

In this project Polynomial-based Route Filtering scheme (PRF) for CPNS is proposed, which can filter false injected data effectively and achieve a high resilience to the number of compromised nodes without relying on static data dissemination routes and node localization. PRF adopts polynomials instead of MACs (Message Authentication Codes) to verify reports and can mitigate node impersonating attacks against legitimate nodes. In this scheme, two types of nodes are considered: sensing node and forwarding node. The sensing node can not only sense and endorse the measurement reports of the monitored components, but also forward the measurement reports along the route, whereas the forwarding node is used to forward the received measurement reports to the controller.

Each node stores two types of polynomials: authentication polynomial and check polynomial, which are derived by different primitive polynomials. Each sensing node stores the authentication polynomial of local cluster and the check polynomial of other clusters with a pre-defined probability. Each forwarding node stores the check polynomial of each cluster.

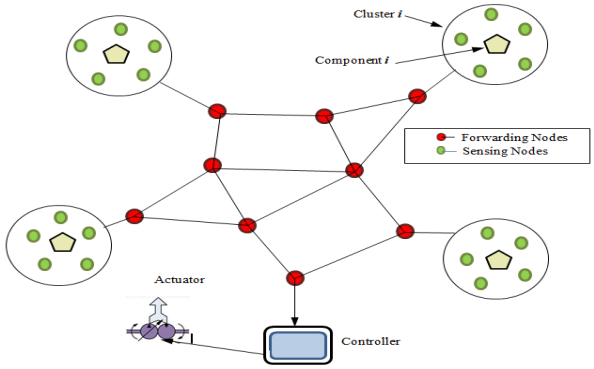


Fig. 2.1 Network System Model

In fig. 2.1 the green colored nodes are the sensing nodes monitoring a physical component. There are five nodes in a cluster. Red nodes are the forwarding nodes i.e. they are intermediate nodes. These nodes forward the data generated by the sensing nodes to the controller. The controller then sends feedback and control commands to the actuators.

In PRF instead of using the node association to share the authentication information between source nodes and forwarding nodes, our scheme uses the statistical pre-assignment to share the authentication information between nodes that makes our scheme independent from static routes.

In addition, PRF assigns the authentication polynomial and check polynomial for each node based on the cluster-based polynomial assignment, i.e., nodes in different clusters are assigned different primitive polynomials and generate different authentication polynomial and check polynomial. In this way, the compromised nodes in one cluster will not affect the security of nodes in other clusters, i.e., suppressing the effect of compromised nodes within the local area. Hence, PRF achieves high resilience on the increased number of compromised nodes.

III. PROPOSED SCHEME

2.3 Authentication Information Generation

A master key and a global primitive polynomial pool has to be generated before deploying the sensor nodes. The generated master key stored in the memory of nodes before node deployment and used to generate the cluster key for each cluster. The global primitive polynomial pool consists of several ternary polynomials, which are randomly created before nodes are deployed. This pool is used to assign the primitive polynomial as a node ID to each cluster and its size is N, where n is the number of sensing nodes monitoring a component and i represents the number of clusters in the

network. *N* is the total sensing nodes in the system. One more such pool is created to assign IDs to the forwarding nodes *fn*.

Step 1: Cluster organization

After the deployment, the nodes form a cluster. Clustering can be done on the basis of:

1. **Location**
2. **Parameter to be sensed**
3. **Type of Data etc.**

In our scheme we are doing location based clustering. Each cluster has a component to be monitored by the sensing nodes in the cluster. We can deploy sensing nodes close to the monitored component. Those nodes communicate with each other and each node only stores the node IDs of other sensing nodes in its cluster regardless of the number of hops any other node is away from it.

Step 2: Cluster Key generation

In this stage, each sensing node generates its cluster key by using the master key. The master key is erased once the network deployment is completed. Eq. (1) represents the cluster key.

$$f\,i(N) = \text{Masterkey}*CI*NI \qquad (1)$$

fi(N) is the cluster key generation function. The * is denoted as the multiplication operation and its function is to combine multiple strings to one new string, and then uses the new string to generate the cluster key. It is assumed that adversary cannot compromise the node during the initialization phase, the adversary cannot obtain from the unknown even in the filtering phase, because cluster key is generated by master key. the Hence, adversary cannot decrypt the measurement from the report.

Step 3: Local ID assignment

In this stage, each sensing node is assigned a local ID by its cluster-head. Note that there are *n* sensing nodes monitoring the physical component in each cluster. This process is only used to ensure that each cluster node is assigned one local ID.

By using the local ID assignment, the cluster-head assigns the local ID to all nodes in the cluster and ensures that for any *i*, there is a LocalID stored in one and only one cluster node. By screening the local ID attached in the sensing report of monitored components, our scheme can detect the false measurement reports sent by the compromised cluster-head and increase the resilience to false data injection attacks.

Step 4: Authentication and check polynomial assignment

In this stage, the network designer initializes all nodes and the network with the following parameters: node ID, master key, local ID, cluster ID. The sets of polynomial pools represent all sensing node IDs, all forwarding nodes IDs, authentication polynomials, check polynomials and all measurement reports of monitored components, respectively. For each sensing node, the designer stores the master key in it. He also reads the cluster ID and the master key stored in the node to compute the authentication polynomial of cluster for every node in it.

$$\text{AP} = \alpha\, f\, i(N); \qquad (2)$$

Where *NI* is the sensing node ID in cluster, AP is the authentication polynomial of i^{th} cluster for node *N* and α is a parameter where α is a positive integer. Note that the system designer can randomly choose the value of α while computing the authentication polynomial. Thus, no other party knows the value of except the system designer. Hence no other party can decrypt the authentication polynomial of the cluster. After the computation, the AP is stored in each sensing node. The designer then computes the check polynomials for node. For each sensing node, the designer computes the check polynomial and stores these check polynomials in node.

$$\text{CP} = \beta\, fi(N); \qquad (3)$$

Where CP is the check polynomial of cluster *i* stored in node *N* and β plays the same role as α. The system designer can randomly choose the value of β while computing the check polynomial. In fact, it can be any positive integral value. In our scheme, setting α and β is to allow the route filtering to be more efficient. Larger values of α and β in the set will increase memory storage cost rapidly because sensor nodes need more memory to store the authentication information based on Equations (2) and (3). Hence, to balance the storage cost of nodes and resilience of our scheme, α and β are set to small values. For each forwarding node, the designer computes the check polynomials of all the sensing nodes and stores the check polynomials in node.

As the node IDs of the sensing nodes are different, the primitive polynomial assigned for one cluster is different from others. The use of the ID-based polynomial generation ensures that the authentication polynomial and the check polynomial stored in one node are different from other nodes.

Table 1.1 : Notations

Sr. No.	Notation Used	Significance
1.	N	Sensing nodes in the Network
2.	n	Sensing nodes in a cluster
3.	fn	Forwarding nodes in the cluster
4.	α, β	Design parameters selected by designer
5.	i	Cluster number
6.	CI	Cluster ID
7.	NI	Node ID

Our scheme leads to a high resilience to node impersonation attacks because the authentication information of one cluster has no impact on another cluster. The formation of authentication information in our scheme does not require node localization, as required by existing schemes [9]–[12] and [13].

Overall, the node initialization of PRF consists of four steps. In particular, Step 1 is conducted during the pre-deployment by the network designer. Step 2 is conducted by the network designer after the CPNS is deployed. Step 3 and Step 4 are carried out after the CPNS is deployed and step 4 requires α *and* β inputs from the network designer. After completing the four steps described above, the authentication information assignment is complete and the corresponding authentication information is stored in all sensor nodes. The authentication information plays a critical role in the data security management to detect and filter the false measurement reports, which will be discussed next.

3.2. Data Security Management

Step 1: Sensing report generation

In this stage, each sensing node measures the data of the monitored component and generates the sensing report. This report consists of the encrypted measurement and MAP. Sensing nodes generate different sensing reports for the different measurements taken from the same component using their node ID and locally stored authentication polynomial and check polynomial. For example, node first computes the report by applying the encryption function to the measurement, by using the master key, represented by Equation (4).

EncData = Data* Masterkey; *(4)*

Where 'Data' is the measurements of its monitored component, master key is stored in node of the cluster to which it belongs. This step is carried out immediately after the network is deployed as the master key is erased after the deployment. Then node generates MAP for the measurement.

Step 2: Measurement report generation and transmission

After generating sensing reports, all the sensing nodes generate their MAPs. MAP is formed by merging authentication polynomial and check polynomial together. The sensing nodes then club the sensing report along with their MAPs. This clubbing together of three things is called as Measurement report. Then all the sensing nodes store the Measurement reports generated by the sensing nodes in their Report field.

All the sensing nodes then send their Measurement reports to their cluster head in the respective time slots. Cluster head then combines these measurement reports and forward to the first forwarding node which is nearby to it and this node is fixed in the structure.

The first intermediate node then verifies the node ID of the cluster head with the polynomial pool. If the ID matches, then it verifies the measurement report and then it forwards the data otherwise it drops the report and send a message to the sensing nodes that belong to that cluster to elect a new head as the current head node is compromised.

In this way, PRF can drop the false measurement report forged by the compromised cluster-header effectively at the first intermediate node along the forwarding route. Note that in our scheme, we do not focus on any specific communication schemes or routing protocols and the sensors need not always be on. The communication scheme and routing protocols developed in the past can be leveraged to establish routes to forward the measurement reports through forwarding nodes. The cluster-head and ordinary sensing node can also serve as the forwarding nodes. If the adversary compromises the ordinary sensing node or cluster-head, he can forge and send the false measurement reports of other components to the controller via the compromised nodes. Then the above approach cannot detect and filter this false report.

To deal with this issue, PRF adopts the route filtering mechanism described in the next steps.

Step 3: Route filtering

In PRF, the measurement report is transmitted to the controller hop-by-hop. The intermediate node, which has the corresponding check polynomial, determines whether the received measurement report is false through validating the following conditions:

(i) Condition 1: The forwarding node ID of previous node belongs to the polynomial pool.
(ii) Condition 2: MAPs can be verified by the corresponding check polynomial stored in the intermediate node.
If the above two conditions are not satisfied, the intermediate node will drop the measurement report. Otherwise, the measurement report will be forwarded.

To verify the Condition 2 intermediate node first calculates the values of the check polynomials from the MAPs of all the sensing nodes and then verifies the check polynomials in the report with the calculated check polynomials. If condition 1 and 2 meets, the measurement report forwarded by cluster head can be determined as valid one. The report is then forwarded to the first forwarding node in the shortest path. Otherwise, the measurement report will be filtered.

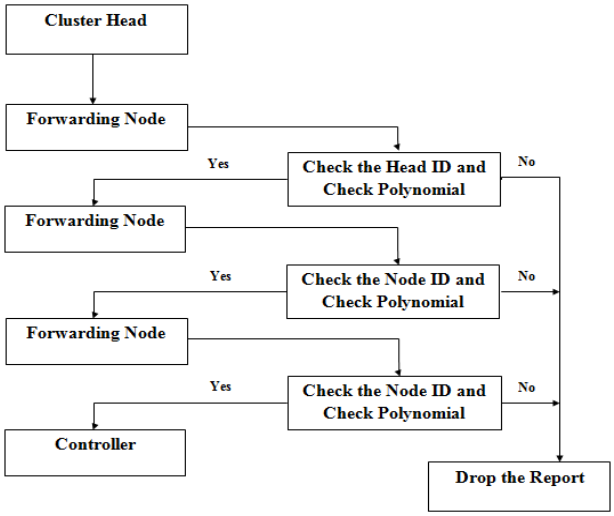


Fig. 3.1 Route Filtering Scheme

Then the first forwarding node forwards the report to the next forwarding node. The next forwarding node checks the authenticity of the report in a same way as the previous forwarding node does. This process goes on until the report reaches to the controller.

Step 4: Controller authentication

It is assumed that the controller is well protected and the adversary cannot compromise it, and thus the cluster IDs of all clusters, the masker keys of all clusters, the cluster key generation function, authentication polynomial generation function, check polynomial generation function can be stored in the controller without losing confidentiality.

After receiving the measurement report, the controller validates it in the same way as the intermediate node. Controller can validate all received measurement reports and filter the false measurement reports, which bypass the detection of intermediate nodes. If the report is confirmed as legitimate, the controller decrypts the measurements from the report, and estimates the state of monitored component and sends the feedback control commands to the actuators to control the operation of physical systems.

If the controller finds the report compromised, it then simply filters out the route through which the report travelled and the route will not be considered for the next transmissions.

Because it contains the complete authentication information, the controller is the last defense in the system and can detect and filter all the false measurement reports forged by the adversary.

IV. SIMULATION RESULTS AND ANALYSIS

4.1 Result of Network Formation and Data Forwarding

4.1.1 Our approach to create a network scenario using MATLAB

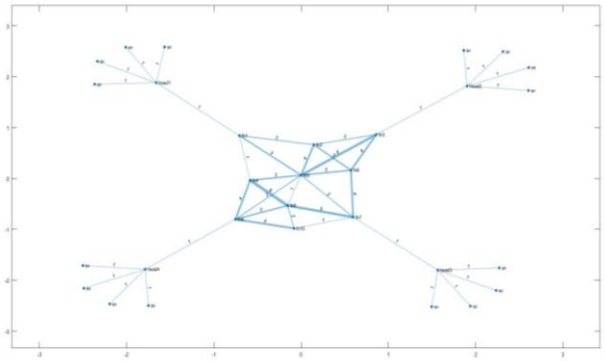
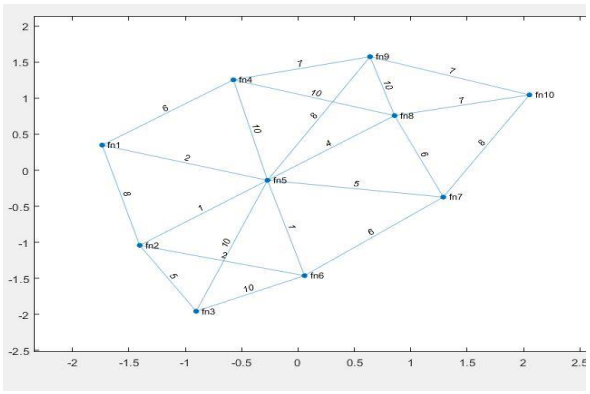


Fig. 4.1. Wireless network scenario simulation using MATLAB.

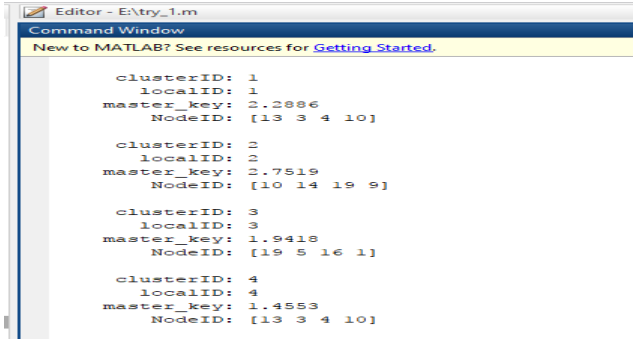
In fig. 4.1., four clusters are there with heads head1, head2, head3, head4 resp. The head1 is sending data to a predefined forwarding node fn1. The head2 is sending data to a predefined forwarding node fn3. The head3 is sending data to a predefined forwarding node fn7. The head4 is sending data to a predefined forwarding node fn9. All these 4 forwarding nodes are then forwarding the data to the controller which is fn10 via 4 different shortest paths to the controller.

4.1.2 Placement of forwarding nodes



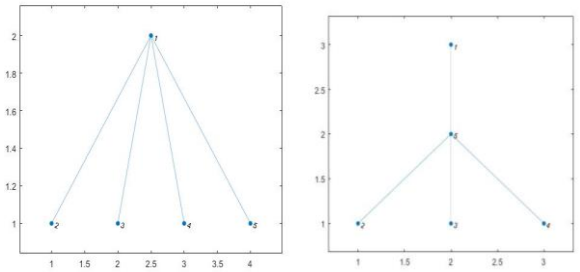
The forwarding nodes have fixed structure because, even if a malicious node is introduced and if it tries to access the network nodes it will fail. As the structure is fixed, there are no chances that a fake node will reply to cluster head to create a black hole.

4.1.3 Node Parameters Allocation Result



A random node from each cluster is selected to show the node parameter allocation Four clusters have 4 IDs, Each of them have 5 nodes each. Local IDs from 1 to 5. Master key is randomly generated for every sensing node.

4.1.4 Clustering

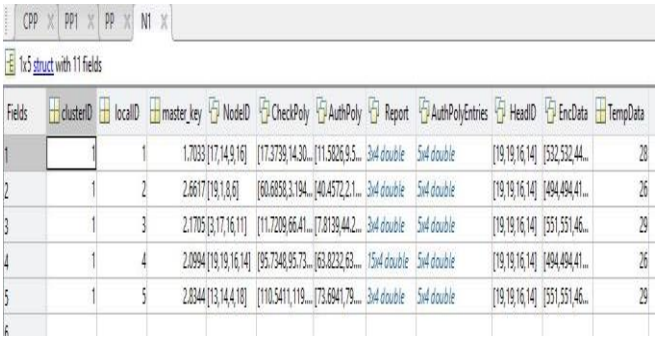


Cluster formed could look like any of above structures with 5 nodes.

4.1.5 Sensing Node Parameter

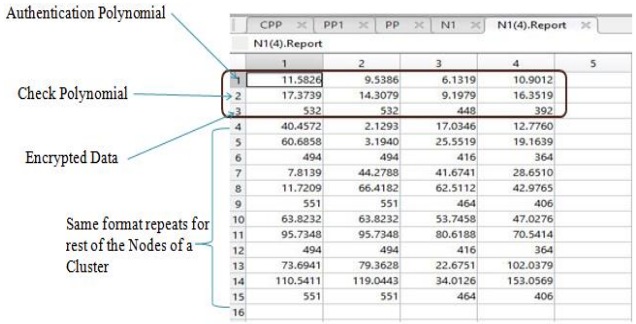
The following are the fields that belong to a sensing node. Each node has a HeadID field which carries the node ID of a current head. Every node has its report field but the head of te cluster has a report field which carries the reports of all the nodes that belong to that cluster.

TempData is the data sensed by the sensing nodes. EncData is the data encrypted using master key. NodeID, HeadID, AuthPoly, CheckPoly are 1*4 matrices. Ssensing node fields are shown in screenshot below:



4.1.6 Data Report Format

The following is a report format. It consist of three fields namely Authentication polynomial, check polynomial and Encrypted data i.e. three 1*4 matrices. The total 15 fields shows the report of 5 nodes of a cluster. So a report that a cluster head forwards has 15 fields i.e. fifteen 1*4 matrices.



4.1.7 Controller Data Base

a. Sensor node IDs
b. Forwarding node IDs
c. Master keys
d. Authentication polynomial pool
e. Check polynomial pool
f. Cluster keys

Following are some pictures of the data that is available with the controller for the authentication purpose.

a. Authentication polynomial pool

CPP | PP1 | PP | N1 | N1(4).Report

20x4 double

	1	2	3	4
1	11.5826	9.5386	6.1319	10.9012
2	40.4572	2.1293	17.0346	12.7760
3	7.8139	44.2788	41.6741	28.6510
4	63.8232	63.8232	53.7458	47.0276
5	73.6941	79.3628	22.6751	102.0379
6	2.5147	20.1175	12.5734	25.1468
7	24.1382	60.3456	36.2074	44.2534
8	66.1969	48.1432	78.2327	18.0537

The authentication polynomials and check polynomials are basically 1*4 matrices. The coefficients of these matrices are generated using the eq. 2 and 3 in MATLAB. And both the polynomials are different for different sensing nodes.

b. Check Polynomial Pool

CPP

20x4 double

	1	2	3	4
1	17.3739	14.3079	9.1979	16.3519
2	60.6858	3.1940	25.5519	19.1639
3	11.7209	66.4182	62.5112	42.9765
4	95.7348	95.7348	80.6188	70.5414
5	110.5411	119.0443	34.0126	153.0569
6	3.7720	30.1762	18.8601	37.7203
7	36.2074	90.5184	54.3110	66.3802
8	99.2953	72.2148	117.3490	27.0805

4.1.8 Data Verification and Forwarding

C1, C2, C3, C4 are the four shortest paths from the respective clusters to the controller. Data forwarded via C1 path to the controller is shown step by step here. All the data is verified by the forwarding nodes and then it is being forwarded. Node 1 collects the data from cluster head 1. Then node 1 forwards the data to node 4 then node 4 to node 8 and then node 8 to the controller.

```
Command Window
>> DataFWDNew
C1 path
     1     4     8    10

C2 path
     3     5     8    10

C3 path
     7    10

C4 path
     9    10

Flag is here
     1

cpp is verified
Flag is here
     4

cpp is verified
Flag is here
     8

cpp is verified
Flag is here
    10

All the data reached to controller
forword next clusture data clusture number is
     2

Flag is here
     3
```

4.1.9 Forwarding Node Parameter

Forwarding node parameters means the fields that are available with the forwarding nodes to accept the data, carry out the necessary verification and forward the data to the next forwarding node. They are shown in screenshot below:

CPP | PP1 | PP | N1 | N1(4).Report | APP | FN

1x10 struct with 7 fields

Fields	name	Edges	Report	ID	Flag	TxID	HTxID
1	'fn1'	1x3 cell	15x4 double	[9,2,5,11]	0	[0,0,0,0]	[4,1,10,15]
2	'fn2'	1x3 cell	15x4 double	[23,1,30,39]	0	[0,0,0,0]	[0,0,0,0]
3	'fn3'	1x2 cell	15x4 double	[36,24,33,8]	0	[0,0,0,0]	[16,12,16,1]
4	'fn4'	1x3 cell	15x4 double	[23,25,30,12]	0	[0,0,0,0]	[0,0,0,0]
5	'fn5'	1x4 cell	15x4 double	[31,37,14,17]	0	[36,24,33,8]	[0,0,0,0]
6	'fn6'	1x1 cell	15x4 double	[36,30,24,10]	0	[0,0,0,0]	[0,0,0,0]
7	'fn7'	1x2 cell	15x4 double	[34,13,19,8]	0	[0,0,0,0]	[2,8,6,2]
8	'fn8'	1x2 cell	15x4 double	[6,20,4,29]	0	[31,37,14,17]	[0,0,0,0]
9	'fn9'	1x1 cell	15x4 double	[8,11,34,8]	0	[0,0,0,0]	[10,12,20,15]
10	'fn10'	1x3 cell	15x4 double	[7,30,28,21]	0	[8,11,34,8]	[0,0,0,0]

4.2 Attack Scenarios

4.2.2 Malicious Node Creation

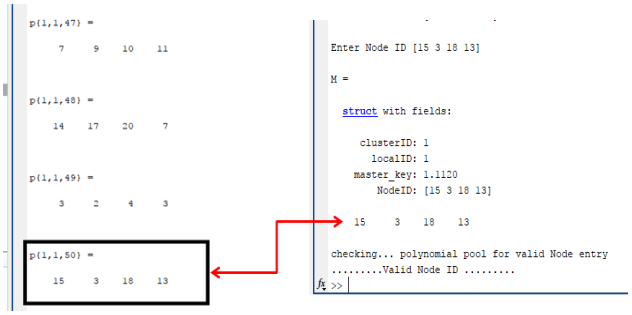
A malicious node is created in the network to show that when there is an invalid node entry found, the network can detect it as a malicious node.
Here a node is created with ID [20 20 10 15]. This does not belong to the polynomial pool for Node IDs.

a**. Following screenshot represents a user defined malicious node**

```
New to MATLAB? See resources for Getting Starte
Enter Cluster ID 1

M =

  struct with fields:

    clusterID: 1

Enter Local ID2

M =

  struct with fields:

    clusterID: 1
      localID: 2

Enter master key1.4

M =

  struct with fields:

     clusterID: 1
       localID: 2
    master_key: 1.4000

Enter Node ID [20 20 10 15]
Enter Node ID [20 20 10 15]

M =

  struct with fields:

     clusterID: 1
       localID: 2
    master_key: 1.4000
        NodeID: [20 20 10 15]

    20    20    10    15

checking... polynomial pool for
.............Malicious node fou
>>
```

It shows that a malicious node with an invalid node ID is identified as a malicious node.

b. When a Malicious Node is Created with a Valid ID



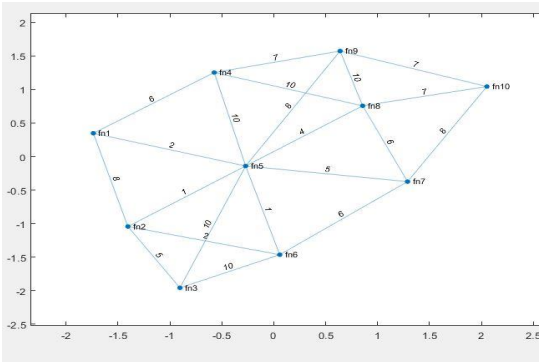
When a malicious node with a valid ID is created, it means a node is compromised. So, the forwarding nodes will not be able to identify it as its ID belongs to the polynomial pool. But if the node tries to manipulate the data and forwards it, controller will be able to identify it.

4.2.3 Forwarding Node is Compromised

If a forwarding node is compromised and if it manipulated the report that is being forwarded, then the node that receives the data from a compromised node will check its ID and the check polynomials of the report. If it finds them wrong, the node will simply drop the report and the compromised node will be discarded from being considered in the network.

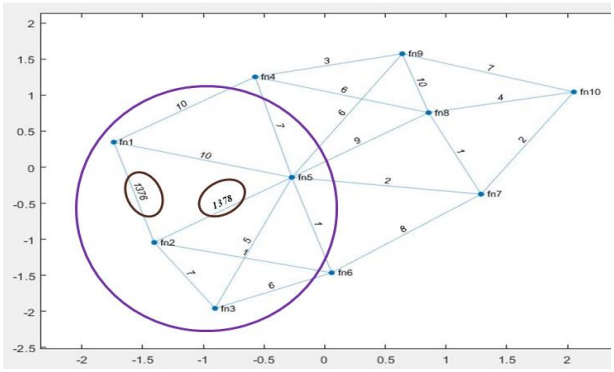
a. **Path before attack** : 1--2--3

Data is being forwarded from node 1 to 2 to 3.



b. After Attack Scenario

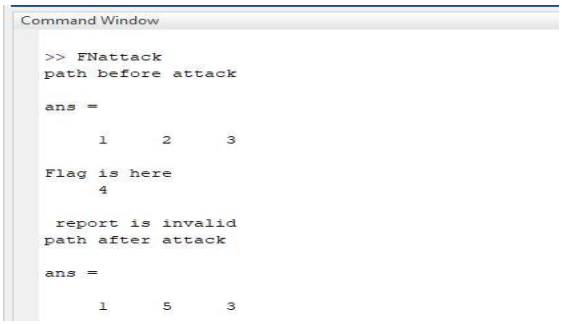
Since node 2 is compromised, it is not considered while selecting a shortest path. The weights of the incoming paths to the compromised node are increased to a very large value so that it will not be taken while calculating a shortest path.



c. Remedy

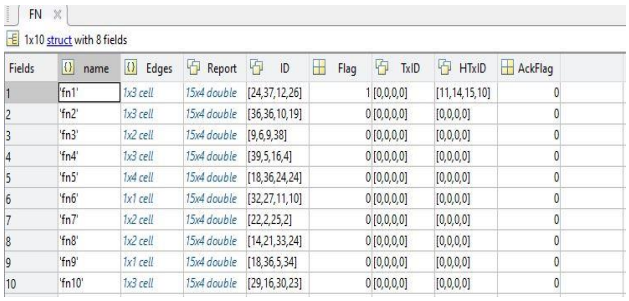
So, node 2 is discarded and new shortest path is found as node 1 to node 5 and node 5 to node3.

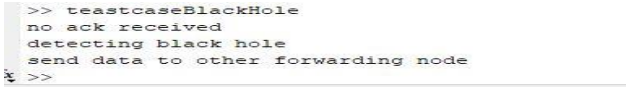
- Alternate Path



4.2.4 Forwarding Node Creates a Black Hole

This is the case when forwarding node only receives the data and doesn't forward it. It creates a black hole in the network.





Whenever a forwarding node forwards its data to the next node in the path, it sets the AckFlag of the previous forwarding node in the path .The AkFlag field is empty since

no acknowledgement is received as the first intermediate node has created a black hole here.

- Increasing the weights and discarding the node

Here again node 2 is compromised so it is discarded while selecting a new shortest path. In the result shown below, again the weights of the paths incoming to the compromised node 2 are set to higher values so that node 2 will not be considered in the network for next transmissions.

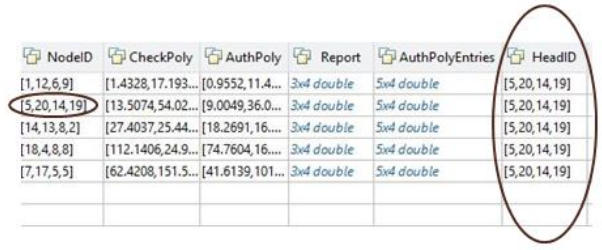
The result is same as that of section 4.2.2 b.

4.2.5 Cluster Head is Compromised

When a cluster head is compromised, all the nodes from that cluster stop sending the data to it. They re-elect the head without considering the previous head while electing.

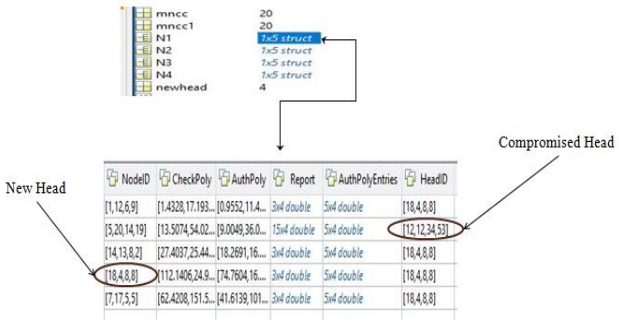
- **Cluster Values Before the attack**

The node ID of a cluster head is there with all the sensing nodes belong to that cluster.



NodeID	CheckPoly	AuthPoly	Report	AuthPolyEntries	HeadID
[1,12,6,9]	[1.4328,17.193...	[0.9552,11.4...	3x4 double	5x4 double	[5,20,14,19]
[5,20,14,19]	[13.5074,54.02...	[9.0049,36.0...	3x4 double	5x4 double	[5,20,14,19]
[14,13,8,2]	[27.4037,25.44...	[18.2691,16....	3x4 double	5x4 double	[5,20,14,19]
[18,4,8,8]	[112.1406,24.9...	[74.7604,16....	3x4 double	5x4 double	[5,20,14,19]
[7,17,5,5]	[62.4208,151.5...	[41.6139,101...	3x4 double	5x4 double	[5,20,14,19]

- **Cluster Values After the attack**



Because the head compromise attack took place, all the sensing nodes except the compromised node elected a new cluster head. The new HeadID is [18,4,8,8]. The HeadID field of all the sensing nodes belong to that cluster is changed with new HeadID except for the previous head node.

V. CONCLUSIONS

- In MATLAB simulation of 10 nodes with 4 degree polynomials, efficiency to easily find out malicious node is 100%. As the number of nodes will increase from 10 to 500, efficiency will decrease to around 90%.
- In consideration with implementation on embedded platform, network doesn't become complex for 4 degree polynomials. As the degree increases to 8 or 12 or so, square complexity comes into the picture.
- As the number of nodes increases above 30, time to evaluate network increases at the cost of secured data transmission.

VI. FUTURE SCOPE

As we have worked with a small network, future researchers may try to provide secure communication in a large network with less complexity which can prevent node compromisation and data compromisation.

REFERENCES

[1] Xinyu Yang, Jie Lin, Wei Yu, Paul-Marie Moulema, Xinwen Fu, and Wei Zhao, ''A Novel En-Route Filtering Scheme Against False Data Injection Attacks in Cyber-Physical Networked Systems,'' IEEE Transactions On Computers, vol. 64, no. 1, January 2015.

[2] Chen, Xiangqian, Kia Makki, Kang Yen, and NikiPissinou. "Sensor network security: a survey." Communications Surveys & Tutorials, IEEE 11.2 (2009): 52-73.

[3] Y. Liu, M. K. Reiter, and P. Ning, "False data injection attacks against state estimation in electric power grids," in Proc. 16th ACM Conf. Comput. Commun. Security (CCS), 2009, pp. 21–32.

[4] Y. Mo and B. Sinopoli, "False data injection attacks in control systems," in Proc. Preprints 1st Workshop Secure Control Syst., CPS Week, 2010.

[5] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injection false data in sensor networks," IEEE J. Sel. AreasCommun., vol. 23, no. 4, pp. 839–850, Apr. 2005.

[6] L. Yu and J. Li, "Grouping-based resilient statistical en-route filtering for sensor networks," in Proc. 28th IEEE Int. Conf. Comput.Commun. (INFOCOM), 2009, pp. 1782–1790.

[7] Z. Yu and Y. Guan, "A dynamic en-route filtering scheme for datareporting in wireless sensor networks," IEEE/ACM Trans. Networking(ToN), vol. 18, pp. 150–163, 2010.

[8] J. Lin, X. Yang, W. Yu, and X. Fu, "Towards effective

en-route filtering against injected false data in wireless sensor networks," in Proc. IEEE Global Telecommun. Conf. (GLOBECOM), 2011, pp. 1–6.

[9] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh, "Toward resilient security in wireless sensor networks," in Proc. 6th ACM Int. Symp. Mobile Ad Hoc Netw. Comput. (MobiHoc'05), 2005, pp. 34–45.

[10] W. Zhang, N. Subramanian, and G. Wang, "Lightweight and compromise-resilient message authentication in sensor networks," in Proc. 27th IEEE Int. Conf. Comput. Commun. (INFOCOM), 2008, pp. 1418–1426.

[11] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An interleaved hop-by-hop authentication scheme for filtering of injection false data in sensor networks," ACM Trans. Sensor Netw., vol. 3, no. 4, pp. 259–271, 2007.

[12] K. Ren, W. Lou, and Y. Zhang, "Leds: Providing location-aware end-to-end data security in wireless sensor networks," IEEE Trans. Mobile Comput., vol. 7, no. 5, pp. 585–598, May 2008.

[13] H. Yang and S. Lu, "Commutative cipher based en-route filtering in wireless sensor networks," in Proc. 60th IEEE Veh. Technol. Conf. (VTC), 2004, pp. 12–23.