IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 12, December 2019

Detection of Misbehaving Nodes Due to Outsider Attack in Wireless Sensor Network

Ashlesha Vivek Patil

Graduate Student, Bapurao Deshmukh College of Engineering, Sevagram, University of Nagpur, India

Abstract: Wireless Sensor Networks (WSNs) are engaging technology for monitoring large regions at high spatial and temporal resolution. Hence the security of these sensor nodes must have the most top priority. The wireless sensor network can carry some secret information to the sink node in some sensitive applications. Packet dropping and modifying are the two most common attacks on wireless sensor networks. Till now, various algorithms are proposed to identify packet dropping, but no algorithm is proposed to catch those packet droppers and modifiers, which may again cause packet dropping and modifying. In this paper, we recommend a simple yet powerful scheme to catch packet droppers and modifiers.

Keywords: Packet dropping, packet modification, intrusion detection, wireless sensor networks.

I. INTRODUCTION

A Wireless Sensor Network (WSN) is consisting of distributed devices using different sensors to cooperatively monitor both physical as well as environmental circumstances, such as temperature, sound, vibration, pressure, and motion at various unfriendly locations. This data is forwarding toward a sink, which could be a gateway, base station, storage node, as shown in fig.1.

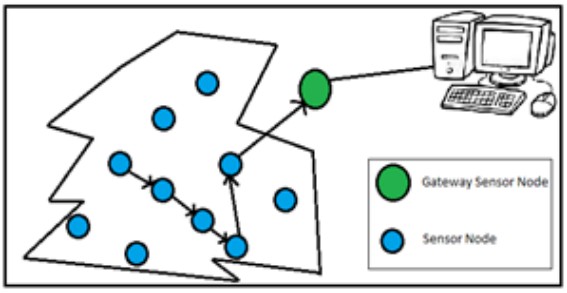

Fig.1 Nodes in WSN

The nodes in the network are attached via wireless transmission channels like on radio waves on different frequencies. A sensor network consists of multiple nodes with small, lightweight, and portable sensors. Each sensor node is equipped with a transducer, microcomputer, transceiver, and power source like a low battery. The transducer is used to generate electrical signals based on sensed physical effects and appearances. They create packets, which is one unit of binary data collected by the nodes capable of being routed through a computer network. A sensor node is often placed in an unfriendly remote condition to perform the monitoring and data acquisition tasks. When it is deployed in such a condition, it lacks physical safeguards and is subject to node yielded by an adversary. After compromising one or multiple sensor nodes, an adversary may launch various attacks to disrupt the in-network communication. Packet dropping and modifying are the two most common attacks launched by them after compromising the nodes. Packet dropping is a compromised node that drops all or some of the packets that are supposed to forward. Packet modification is a compromised node that modifies all or some of the packets that are supposed to forward. In order to deal with the packet dropping problem, a multipath forwarding is a widely used countermeasure to forward each packet along multiple different paths; therefore, the packet dropping is not tolerated in all of these paths. To deal with packet modifiers, most of the existing solution aims to filter modified messages at the time of propagation in the network within a certain number of hops. These countermeasures can tolerate the packet dropping and attacks, but the intruders can still continue attacking the network without being caught. In this, the scheme-routing tree is established with a sink node at the root position. When data packets are transmitted along with the established tree structure toward the sink node, each packet sender or forwarder node adds a small number of extra bits to the packet. These extra bits are called packet marks whose format is purposely designed such that the sink can obtain useful information from those marks. After adding the packet marks, the packet is encrypted at each node. Based on those marks, the sink can figure out the dropping ratio associated with every sensor node and then runs our proposed algorithms.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 12, December 2019

II. PROPOSED WORK

The proposed scheme is consisting of a system initialization phase and several equal-duration rounds of intruder identification phases. These equal-duration rounds of intruder identification phases are described below:

- The initialization phase is consisting of sensor nodes forming a topology which is directed acyclic graph (DAG) and a routing tree is extracted from it. The routing tree structure is followed by Data reports.
- In every round, data is transferred to the sink node through the routing tree. A small number of extra bits are added to the packet, followed by encryption by each sender or forwarder. After finishing one round the extra bits carried in the received packets, the sink(root) runs a node categorization algorithm to identify bad nodes (i.e., either droppers or modifiers) and nodes that are suspiciously bad (i.e., doubted to be either packet droppers or modifiers).
- The shape of the routing tree changes after every round. As a certain number of round passes, the sink node collects information about the behaviors of each node in different routing topologies. This information consists of nodes that are bad, which nodes are doubted to be bad, and the nodes' topological relationship. For further identification of bad nodes from the potentially large number of suspicious nodes, the sink node runs a heuristic ranking algorithm.

Following are the names of the algorithms that are being used for each equal-duration rounds of intruder identification phases:

A. DAG Establishment and Packet Transmission: It consists of system initialization, which is used to set up secret pairwise keys between the sink and every regular sensor node, and to establish the DAG and the routing tree to facilitate packet forwarding from every sensor node to the sink.

B. Node Categorization Algorithm: This algorithm is used to identify nodes that must be bad (Packet dropper or modifier) and nodes that are suspiciously bad. (Packet droppers or modifiers).

C. Tree Reshaping and heuristic Ranking Algorithm: The tree used to forward packets is dynamically changed from each round. For this Tree reshaping algorithm is used. After few rounds have passed, the sink node will have all the information about nodes behaviors. The information includes which nodes are bad for sure, which nodes are suspiciously bad, and the nodes' topological relationship. To further identify bad nodes from the potentially large number of suspiciously bad nodes, the sink runs heuristic ranking algorithms. For this Heuristic Ranking Algorithm is used.

This system is capable of identifying both packet droppers and modifiers. It has low communication and energy requirements. It is also compatible with existing false packet filtering schemes.

III. IMPLEMENTATTION

Ad hoc On-Demand Distance Vector (AODV) is a routing protocol for Mobile Ad hoc Networks (MANETs) and other wireless ad hoc networks[1]. In AODV, the system is silent until a connection is needed. At that point, the network node that requires a link broadcasts a request for contact.
Other AODV nodes forward broadcast messages and record the node that they gathered it from, producing an eruption of temporary routes back to the needy node. A node already has the path to the desired node while receiving such a message. The needy node then established the route that has the minimum number of hops through other nodes, and the unused entries in the routing table were recycled after a time. The process repeats after the failure of the link. To avoid repeat route requests that node already passed, the nodes used a particular sequence number of each node. If a route request fails, another route request is not made until the double timeout of the previous route request is passed.
AODV does not require any much memory as it uses distance vector routing , and also it creates no extra traffic to establish interaction along surviving links.

A. Platform: Most UNIX and UNIX-like system, for example, it runs on GNU/Linux, FreeBSD, Solaris, Mac OS X, and Windows versions that support Cygwin. This project can also be implemented on Microsoft Windows popular Versions like XP, seven or the latest OS 8 and 8.1, but for this to work, the Cygwin tool is required. Cygwin is an Open Source tool that provide functionality similar to a Linux distribution on Windows. It's a DLL (cygwin1.dll) which provides substantial POSIX API functionality. But we must rebuild the application we are using for the project from source if we wanted it to run on Windows. Hence, we find Fedora 7 Operating System a better, easy, and familiar environment to implement the project than Windows.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 12, December 2019

B. System Design:

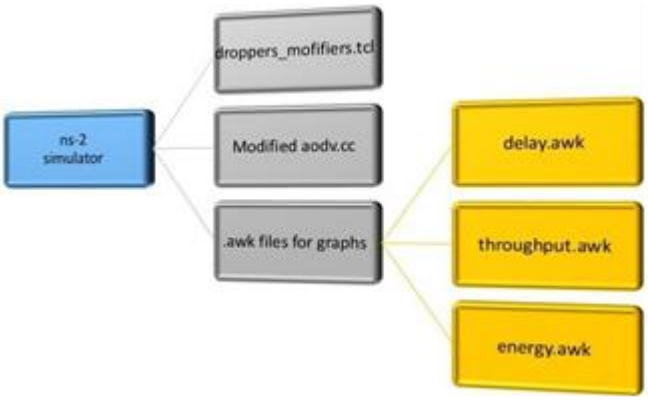

Fig.2 Basic modules in the project

ns-2 simulator is the tool for making simulation and Code files required:

1. Dropper_modifiers.tcl - It's the file needed for simulation which defines the network attributes like the number of nodes, a protocol to use, topology, topography i.e., the grid size of the network (dimensions), etc. and the code to create nodes with these properties.

2. Modified aodc.cc – It's the modified version of the C++ file of AODV protocol, which already comes with the ns-2 package. The extra modification is the code for the algorithm we are using in the project.

3. .awk files for graphs - An AWK program is of a series of pattern-action statements. AWK reads a line at a time as a input. A line is looked for each pattern in the program, and for each pattern that matches, the allied action is performed.

4. delay.awk – It calculates the transmission delay time for the network.

5. throughput.awk – It calculates the ration of the packets successfully delivered to the destination nodes to the total packets sent by source nodes. It is represented in percentage.

6. energy.awk – It is the amount of energy that remained in the nodes after the end of the simulation.

B. System Block Diagram:

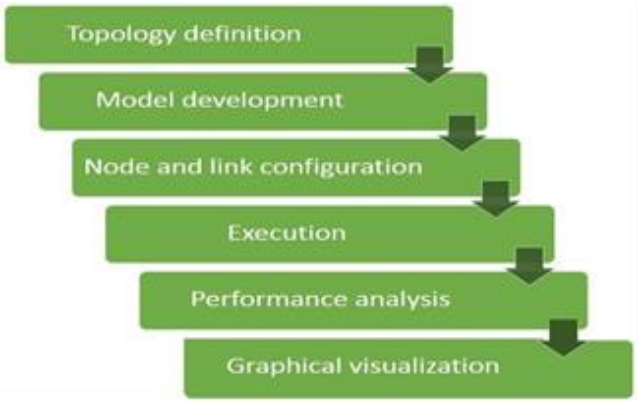

Fig.3 General process of creating a simulation

In our project following step has been performed in order to get the results of the simulation:

1. In the first step, the network simulator variables are initialized, this includes the initialization of the total number of nodes, the packetize and the time interval between the packets. Also, the wireless channel is selected as the communication network.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 12, December 2019

2. In the next step, the topology of the network is initialized, and the communication between those nodes is established.
3. Now the secure port is selected manually, and different security algorithm and firewall is installed on that secure port. We have used the default security algorithm, and the protection system resides in function SEC_PORT in AODV.cc file.
4. Now the secure port algorithm is applied in order to catch the packet dropper and modifier in WSN. This secure port algorithm is implemented in AODV.cc file.
5. Now the important data of simulation is saved in the trace file. This trace file is used for plotting the graph for the delay, throughput, and energy after the end of the simulation.
6. Now go to step 2 if there are some packets remaining for transmission. If there no packets for transmission, then go to step 7.
7. End the simulation and close the trace and the nam file.
8. Now plot the graph for energy, throughput, and delay on the basis of the information in the trace file.

The following is the simulation diagram:

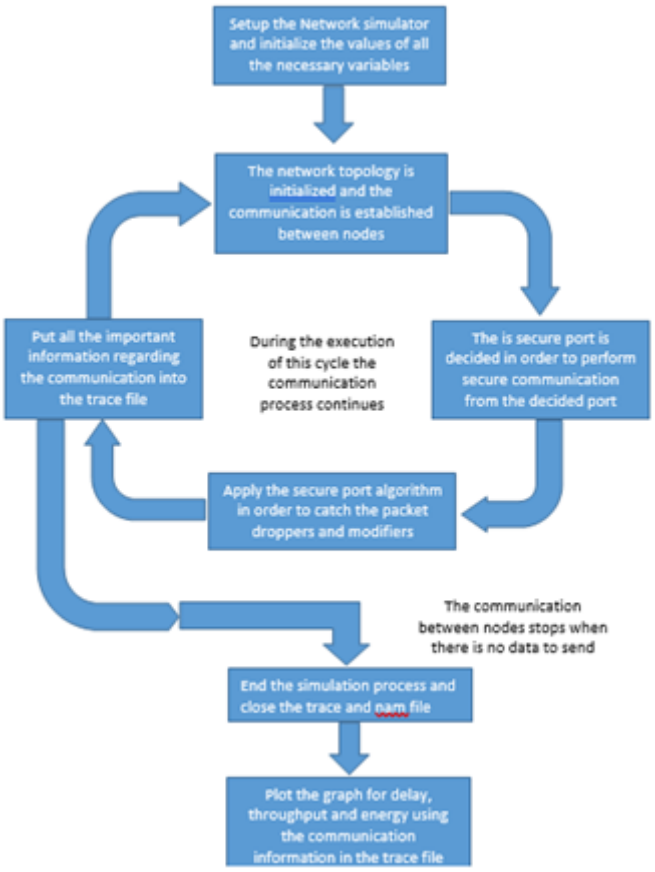

Fig. 4 Block Diagram for Simulation

IV. ALGORITHM AND DATA FLOW

The Secure port algorithm is as follows:
1. Initialize an array DROPPER_NODES whose length is 100, i.e.; there are going to be 100 nodes.
2. If the current algorithm is made available to the WSN, then initialize the source port and the destination port.
3. If a packet is coming to an untrusted node, which might be either a packet dropper or modifier, then forward all the packet is coming to this node directly towards their destination without any processing.
4. If a packet is coming from a secured port then, then the packet is passing from a secure path; hence it is not modified. Therefore, there is no need to drop the packets along this path.
5. If a packet is coming from an unsecured port, then drop the packets directly, also marks the destination node as an untrusted node; hence all the packets coming to this untrusted node will be forwarded directly towards their destination without processing.
6. End of the algorithm.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 12, December 2019

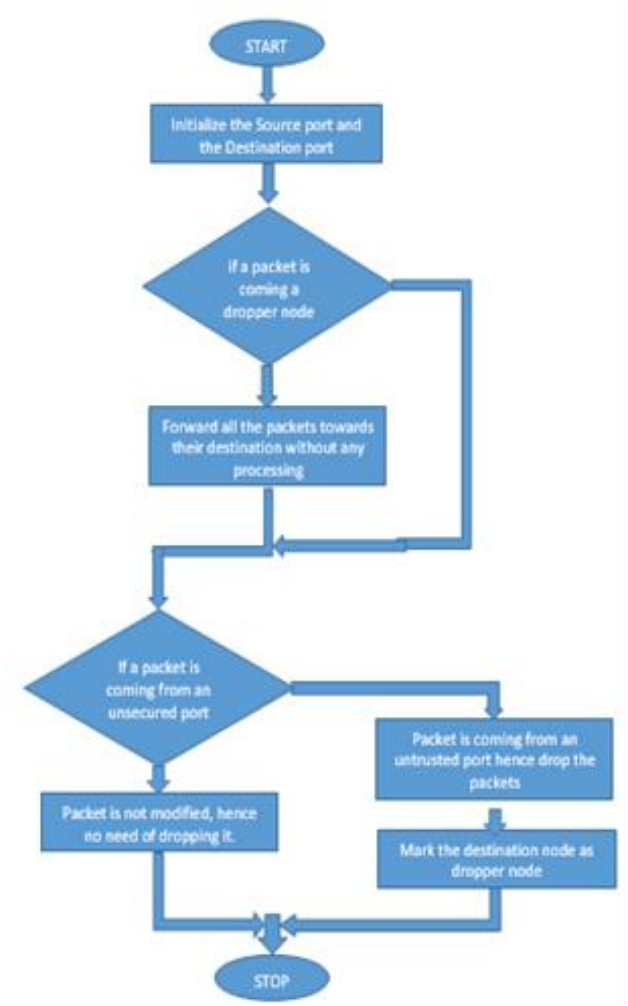

Fig. 5 Data Flow Diagram

V. ANALYSIS AND GRAPHS

A. Normal transmission delay time graph delay.xgvs optimized graph opt_delay.xg shows the packet transmission delay rate of the network.

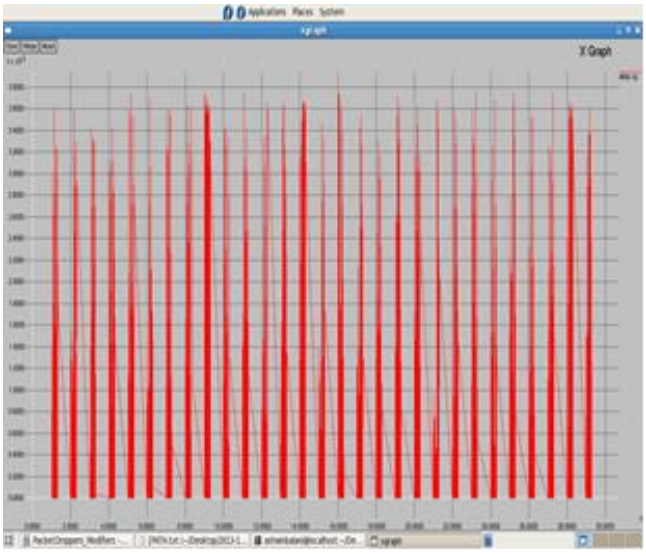

Fig. 6 Normal Delay Graph

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 12, December 2019

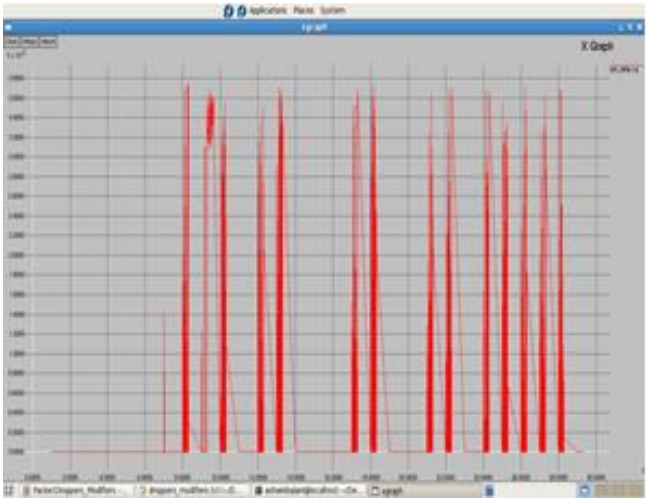

Fig. 7 Optimized delay graph under influence of applied algorithm

B. Normal throughput ratio graph throughput.xgvs optimized graph opt_throughput.xg shows that under normal conditions, throughput drops consistently as compared to under influence of the algorithm where it stand close to 100% mostly.

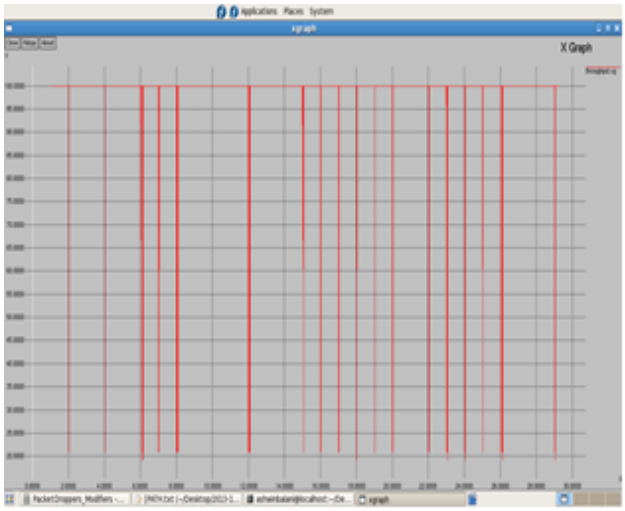

Fig. 8 Normal Throughput Graph

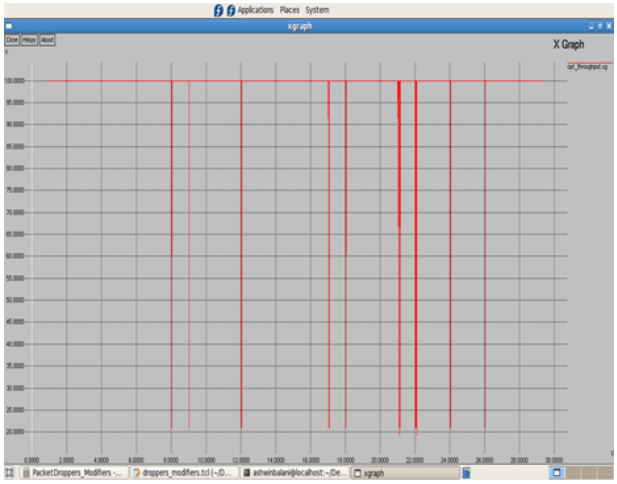

Fig. 9 Optimized throughput graph under influence of applied algorithm

C. Normal graph of energy remained in nodes energy.xgvs optimized graph opt_energy.xg is 105 units whereas its above 120 units under influence of algorithm.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 12, December 2019

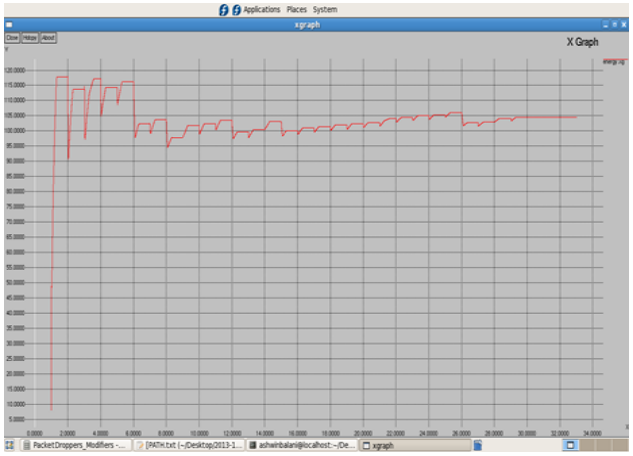

Fig. 10 Normal Energy Graph

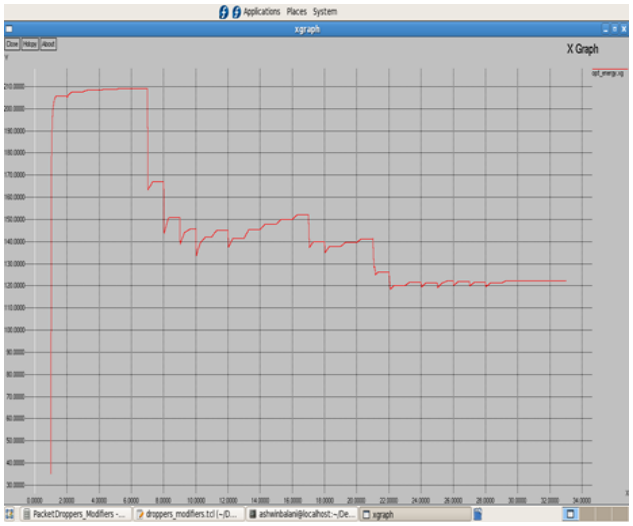

Fig. 11 Optimized Energy graph under influence of applied algorithm

VI. CCONCLSION

This system is a simple and effective scheme to identify misbehaving packet forwarders that drop or modify packets. In order to hide the source of the packets, the packets are encrypted and padded. The small numbers of extra bits are added in each packet such that the sink can recover the source of the packet and then figure out the dropping ratio associated with every sensor node. The shape of the routing tree is changed after each round, and the behavior of each sensor node is monitored so as to send the information to the sink node. Hence most of the bad nodes are identified using a heuristic Ranking Algorithm and hence can be repaired to prevent the packet dropping and modifying.

REFERENCES

[1]. https://en.wikipedia.org/wiki/Ad_hoc_On_Demand_Distance_Vector_Routing [online]..
[2]. C. Wang, T. Feng, J. Kim, G. Wang and W. Zhang, “Catching Packet Droppers and Modifiers in Wireless Sensor Networks,” IEEE Transaction on Parallel and Distributed Systems, Vol. 23, No. 5,pp. 835, May 2012.
[3]. V. Bhuse, A. Gupta, and L. Lilien, “DPDSN: Detection of Packet- Dropping Attacks for Wireless Sensor Networks,” Proc. FourthTrusted Internet Workshop, 2005.
[4]. M. Kefayati, H.R. Rabiee, S.G. Miremadi, and A. Khonsari,“Misbehavior Resilient Multi-Path Data Transmission in MobileAd-Hoc Networks,” Proc. Fourth ACM Workshop Security of Ad Hoc and Sensor Networks (SASN ’06), 2006.
[5]. R. Mavropodi, P. Kotzanikolaou, and C. Douligeris, “Secmr—ASecure Multipath Routing Protocol for Ad Hoc Networks,” Ad HocNetworks, vol. 5, no. 1, pp. 87-99, 2007.
[6]. H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh, “TowardResilient Security in Wireless Sensor Networks,” Proc. Sixth ACM Int’l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc ’05),2005.
[7]. N.Vanitha and G.Genifa, “Detection of Packet Dropper in Wireless Sensor Network using Node Categorization algorithm”, Int’l Journal of Advanced Research in Computer Science and Software Engineering (Vol 3, Issue 3, March 2013), 2013.