Implementing secure data accumulation by ant agents in wireless sensor network using randomized dispersive routes

Sharon C DSouza [1], Mrs. Rajapuspha [2]

[1] Student, Dept. of CSE, MTech,
M.V.J College of Engineering, Bangalore, India
shar.dsz@gmail.com

[2] Associate Professor, Dept. of CSE,
M.V.J College of Engineering, Bangalore, India
puspha.rambabu@gmail.com

Abstract— **In this paper we discuss the implementation of data accumulation in wireless sensor networks and also the methods to increase the security of the accumulated data using dispersive routing techniques like NRRP(Non Repetitive Random Propagation). The data accumulation is accomplished by the concept of Honey Ants and ant based routing and the implementation of the Dijkstra's algorithm to reach the data using the shortest path. Further the use of NRRP assures that the data is securely transported from the source to the sink by the ant agents.**

Keywords- Accumulation, Dispersive routing.

I. INTRODUCTION

As WSN is mostly used for gathering application specific information from the surrounding environment, it is highly essential to protect the sensitive data from unauthorized access. WSNs are vulnerable to security attacks due to the broadcast nature of radio transmission. Sensor nodes may also be physically captured or destroyed by the enemies. The uses of sensor network in various applications emphasis on secure routing. Various protocols are proposed for routing and data gathering but none of them are designed with security as a goal.

Sensor nodes are restricted with limited amount of resources. One such resource is storage. The storage problem can be resolved by the introduction of Ant Agents. These agents traverse the network accumulating data and transmitting the data from the source to the sink. The mobile agents traverse the network for data accumulation. Once the data is found the other mobile agents are intimated of this available data. The mobile agents hence collectively reach the data source using any shortest path algorithm; these agents carry some data and traverse the network to deposit this data into a sink that acts as spare storage. spare storage implementation is done by the administrator.

Generally a well known and efficient routing algorithm is used to deliver the data from source to the sink. The security of the data thus travelling in such routes is compromised as the adversary can easily compute the route to be taken by the data by just getting the routing algorithm information. Two major attacks are that are likely to occur are the Denial –of –Service (DoS) and the compromised node attack. In order to avoid this, the path taken by these agents from the source to the sink can be randomized to avoid the adversary from gaining information on the route taken by these agents. Hence the need for random propagation.

II. THE AIM AND OVERVIEW OF THE PROJECT

The intent of this project is to design a co-operative data accumulation technique called Honey Ant data Accumulation [1] and avoid the possible DoS and CN attacks over the nodes in the network by using Dispersive Randomized Routing Algorithm like NRRP [2] to enhance the security of data transmitted over the network.

The design involves putting up migratory data transporting agents called Ant Agents, which are similar to Mobile Agents in some aspects. These Ant Agents originate at the sensing device and migrate the network, foraging for 'repletes' that have spare storage capacity. Once found, the agent deposits the data on the host 'replete' and returns back to its originating node to fetch the next piece of information. The ant agents use shortest path algorithm like the Dijkstra's algorithm to reach the source node that generates the data, after collecting the data the ant agent follow a different route to take the data towards the spare storage node called as

" replete" to deposit the data. This can be accomplished by generating randomized routes. One such way to generate the path is using the NRRP protocol.

III. ARCHITECTURE

In this section we discuss the architecture of our system as shown in Figure 1. Considering the resource constraints of the nodes involved, the system architecture [3] [4] aims to offload as much intelligence as possible on to the Ant Agents while keeping its device support requirements to a bare minimum. The Security Manager first verifies the credentials of all incoming Ant Agents. AAs with valid security credentials are then queued up for execution at the Virtual Machine.

The Virtual Machine acts as a hardware abstraction layer for loading, scheduling and executing tasks generated by incoming AAs. It interfaces with the Resource Manager, which estimates the amount of storage space left in Tag Space. The Tag Space represents the name-based memory region that stores data objects persistent across AA executions.

When an AA is executed at the Virtual Machine, it first queries the Resource Manager to check if the hosting device is can be its 'replete '. If yes, the sensor data is now deposited in the Tag Space of the hosting device. If not, the AA continues its foraging phase by executing its ant routing algorithm. Post execution, the AAs is injected back into the network to allow them to migrate to their next destination.

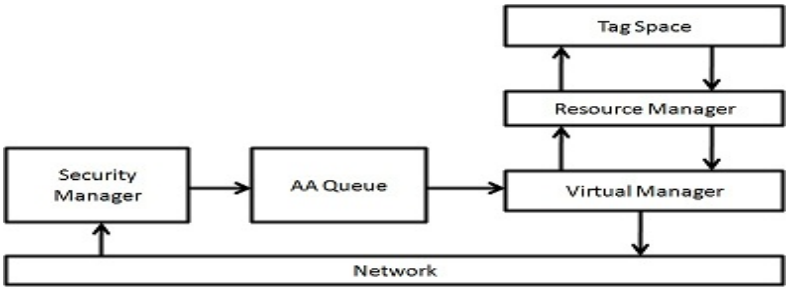

Figure 1: System Architecture

IV. TAG SPACE AND ANT AGENT STRUCTURE

A Tag Space [5] consists of a limited number of tags that are persistent across AA executions. Figure 2 illustrates the structure of a tag. It consists of an identifier, a digital signature, lifetime information and data. The identifier represents the name of the tag. The access of AAs to tags is restricted based on the digital signature. The tag lifetime specifies the time at which the tag will be reclaimed by the device from the Tag Space. AAs can store data (like pheromone) at the Tag Space by creating their own tags.

Figure 2: Tag Space

In addition to its identity and authentication information, an AA is comprised of a code section, a data section as well as a lightweight execution state. A digital signature together with the AA and sensor IDs identifies an AA. The digital signature is used by the host devices to protect the access to an AA tags. The code and data sections contain mobile code and sensor data that an AA carries from one device to another. The data section sets a bound on the expected storage needs of the AA at the host device. Figure 3 depicts the skeletal structure of the Ant Agent.

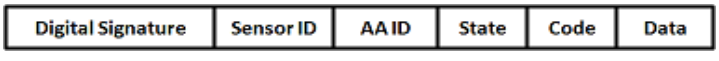

Figure 3: Ant Agent format

V. DIJKSTRA'S ALGORITHM

The algorithm maintains two sets of vertices S and Q. Set S contains all vertices for which we know that the value d[v] is already the cost of the shortest path and set Q contains all other vertices. Set S starts empty, and in each step one vertex is moved from Q to S. This vertex is chosen as the vertex with lowest value of d[u]. When a vertex u is moved to S, the algorithm relaxes every outgoing edge (u,v).

```
DIJKSTRA (G, w, s) pseudo code
1   Function Dijkstra(G, w, s)
2     for each vertex v in V[G]   // Initialization
3       do d[v] := infinity
4         previous[v] := undefined
5     d[s] := 0
6     S := empty set
7     Q := set of all vertices
8     while Q is not an empty set
9       do u := Extract-Min(Q)
10        S := S union {u}
11        for each edge (u,v) outgoing from u
12          do if d[v] > d[u] + w(u,v) //Relax( u,v)
13            then d[v] := d[u] + w(u,v)
14              previous[v] := u
```

The Dijkstra's algorithm has been used to find the shortest path from the source to destination. In this project we use the Dijkstra's algorithm to find the shortest path to find the data generating node.

The Ant Agents generated in each of the node of the network traverse the network to find the data generating node, once the data generating node is found then the Ant Agents intimate other ant agents of the source. These Ant Agents reach the data generating node through the shortest path of the network. The Algorithm used by the Ant Agents or this regard is the Dijkstra's single source shortest path Algorithm [6] [7]. Thus the Ant agents reach the data generating source to take the data to the Replete which is the destination node in this scenario.

VI. NON REPETITIVE RANDOM PROPAGATION

NRRP improves the propagation efficiency by recording all the nodes that the propagation has traversed so far. More specifically, NRRP adds a "node-in-route" (NIR) field to the header of each share. Initially, this field is empty. Starting from the source node, whenever a node propagates the share to the next hop, the id of the up-stream node is appended to the share's NIR field. Nodes included in NIR are excluded from the random pick of the next hop of propagation. This non repetitive propagation guarantees that the share will be relayed to a different node in each step of random propagation, leading to better propagation efficiency.

NRRP pseudo code

```
1. Function Rand()
2. NIR=0
3. do
4. Select the next hop
5. Compare(NIR)
6. If (NIR== true)
                Skip node(id)
7. Else
                Append(NIR)
8. While(dest_not _reached)
```

VII. AD-HOC COOPERATIVE STORAGE

In order to realize the local, in-network storage of the generated sensor data, we introduce a distributed storage model called the "Ad-hoc Cooperative Storage" (ACS) model, that is based on the behaviour of Honey Ants in the real world. The basic idea in implementing storage in the network is through the cooperation among the individual nodes in the network. Just like the 'majors' among Honey Ants, nodes with spare storage capacity in wireless sensor networks, volunteer to become 'replete' in the proposed model. The data generating sensors then create and inject "Ant Agents" (AA) into the network which essentially functions as Honey Ants food fetching medium sized workers. These AAs not only carry sensor data but also forage the network for 'repletes' to gorge them with information. Eventually, when the sink node becomes available, it also injects AAs into the network albeit for a different purpose. The sink injected AAs are akin to the hungry members of the Honey Ant colony, which deplete the 'repletes' of their food reserves. Like any mobile agent, an AA too carries along with it its mobile code as well as its lightweight execution state. AAs are also self routing, namely, they are responsible for determining their own paths through the network. In the proposed ACS model, AAs forage the network for 'repletes' using shortest path Dijkstra's algorithm, once the replete is found they take on a new strategy and travel using non repetitive path protocol (NRRP).

Such routing algorithms based on the behavior of social insects are known to result in optimal routes between the source and the destination. The execution of AAs in the ACS model can be described in two phases: forage and migrate phase followed by a information deposit phase. The AA execution performed at each step may differ based on the storage properties of its hosting device. On devices that meet the targeted storage properties ('repletes'), an AA may deposit its information while on other devices it only executes its ant routing algorithm. From the ANs point of view, devices with the same storage properties are interchangeable. The forage and migrate phases are followed by the information deposit page. The forage and migrate phase may use the shortest path algorithm and the information deposit phase uses NRRP.

VIII. IMPLEMENTATION OF THE PROJECT

The system is implemented comprising of 3 modules

- Ant agent module
- Replete module
- Monitor module

A. Ant Agent and Replete Module

The nodes are off 2 types namely the data generating normal nodes and the replete node. The replete nodes cannot introduce ant agents in the network. The Process begins where the administrator registers as either ant agent on a node or as a replete.

- If it is registered as ant agent then these entities go in search of data generated so as to collect it and store in the replete node
- If the node is registered as replete then it does not produce any data but stores the data that is brought in by the ant agents.

B. Monitor Module

The process begins from the Monitor operations which monitors and keeps log of the entire system. The Monitor will receive information regarding the data available in the network from the Repletes, based on this 2 steps are taken

- Send all ant agents to collect the data from the source
- Carry the data from the source to the spare node/ replete

After this the process stops, we can Refresh the process to start from beginning.

IX. TOPOLOGOGY IMPLEMENTATION

The network topology can be designed by the network administrator. Various topologies that give a well connected network can be implemented. This project uses the following topology comprising of 5 nodes. In this 5 node topology the administrator decides which the data generating node is and which happens to be the "replete" data storing node.

This projects works on the topology as shown in Figure 4, its 2 algorithms namely the Dijkstra's algorithm to find the shortest path to the data generating node and the NRRP algorithm towards depositing data on the data storing node or the replete node. Weights can be randomly assigned to the links.

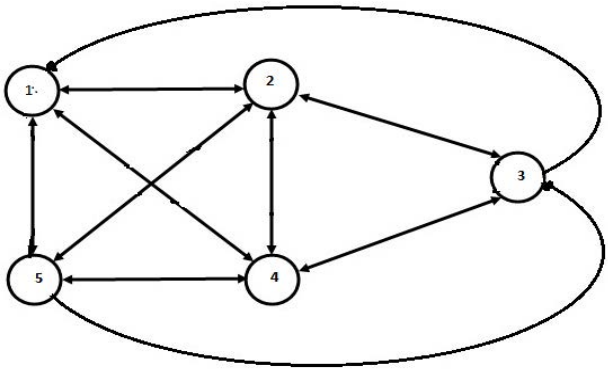

Figure 4: Network Topology

X. CONCLUSION AND FUTURE WORK

The data carried by the mobile agents deploying the NRRP method are more efficient and the routes more secure than those carried by mobile agents using the PRP method. NRRP assures that the data is not transmitted back and forth between the neighboring hops hence is more efficient in transmitting the data from source to sink

using the mobile agents. Moreover the storage issue is also been addressed by the use of Adhoc Cooperative Storage model.

The randomness of this method as compared to the traditional deterministic algorithm deployed usually enhances the security of the data and prevents the occurrences of DoS a CN attacks. More efficient propagation algorithms like Multicast Tree-assisted Random Propagation (MTRP) and Directed Random Propagation (DRP) [2] [8] can be implemented. Various intrusion detection schemes that detect the presence of intruders and their malicious activity can also be included. Further the data received at the sink can be checked if its been manipulated during its routing. If so corrective measures can also be included to enhance security of the data transferred.

REFERENCES

[1] Santosh Kulkarni and Prathima Agrawal, "Honey Ant based cooperative data accumulation in wireless sensor networks", 2011

[2] Tao Shu, Sisi Liu, and Marwan Krunz, "Secure Data Collection in Wireless Sensor Networks Using Randomized Dispersive Route", 2010

[3] S. Kulkarni and P. Agrawal, "Ad-hoc cooperative computation in wireless networks using ant like agents:' in Proceedings of the 17th International Coriference on Advanced Computing and Communications (ADCOM), 2009

[4] P. C. Lee, V. Misra, and D. Rubenstein. Distributed algorithms for secure multipath routing in attack-resistant networks. IEEE/ACM Transactions on Networking, 15(6):1490–1501, Dec. 2007

[5] "Scalable and efficient ant-based routing algorithm for ad-hoc networks:' IEICE Transactions on Communications, vol. E-89B, no. 4, pp. 1231-1238, January 2006.R.

[6] Tao Shu, Sisi Liu, and Marwan Krunz, (2010) "Secure Data Collection in Wireless Sensor Networks Using Randomized Dispersive Routes"

[7] P. C. Lee, V. Misra, and D. Rubenstein. Distributed algorithms for secure multipath routing in attack-resistant networks. IEEE/ACM Transactions on Networking, 15(6):1490–1501, Dec. 2007.

[8] W. Lou and Y. Kwon. H-spread: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks. IEEE Transactions on Vehicular technology 55(4):1320-1330, July 2006.