

Secured On Demand Position Based Private Routing Protocol for mobile ad hoc network (SO2P)

S. Sankara Gomathi, Assistant professor, Sri Venkateswara College of Engg, sgomathi@svce.ac.in
M.A.Bhagyaveni, Assistant Professor, Anna University, bhagya@annauniv.edu

Abstract – Privacy is needed in adhoc networks. Designing a security protocol for ad hoc wireless is a very challenging task. There are various kinds of attacks possible on adhoc network. In this paper, secured on demand position based private routing algorithm is proposed for communication anonymity, which provides a security in mobile adhoc networks. Here, we use cryptography algorithm for providing security to prevent message hacking information from internal and external attackers. The position information of nodes is drawn by GPS (Global Positioning System). Here, we provide a terminal node, which behaves like server to collect position information of nodes in the network. Simulation has been carried out for evaluating the security of ad-hoc networks. It provides a secure data transmission between the nodes. In our design, the server with the process of GPS system records all the information about the terminal nodes involved in the ad-hoc networks. In this paper, we proposed a new protocol using cryptographic techniques to analyze and find the optimal route.

Key words – Ad hoc routing protocol, Optimal Routing, Security, Privacy, Anonymity, cryptographic algorithm.

1. Introduction

An ad hoc network is defined as an “infrastructureless” network, meaning a network without the usual routing infrastructure like fixed routers and routing backbones. Typically, the ad hoc nodes are mobile and the underlying communication medium is wireless. Such ad hoc networks may arise in personal area networking, meeting rooms and conferences, disaster relief and rescue operations, battlefield operations, etc. An important aspect of ad hoc networks has security problems [1, 17, 19]. There are lot of surveys carried on routing protocols for ad hoc wireless networks and they are presented in [14,15]. In this paper, we consider the security of routing protocols for ad hoc networks. Section 2 describe about the related work so far carried out in this field. Section 3 analyzes the security requirements in ad hoc networks. Section 4 presents the designing phases involved in it. Section 5 is the simulation output and results discussion. The conclusion is presented in section 6.

2. Related work

There is very little published prior work on the security issues in ad hoc network routing protocols. Neither the survey by Ramanathan and Steenstrup [14, 20] nor the survey by Royer and Toh [15] mention about security. None of the draft proposals in the *IETF MANET* working group have a non-trivial “security considerations” section. Actually, most of them assume that all the nodes in the network are friendly, and a few declare that the problem is out-of-scope by assuming some solution like IPSec may be applicable. There are some works on securing routing protocols for fixed networks that also deserved are mentioned here.

Perlman, in her thesis [12], proposes a link state routing protocol that achieves robustness. Although her protocol is highly robust, it requires a very high overhead associated with public key encryption. Secure Border Gateway Protocol [9] attempts to secure the protocol by using public key infrastructure and IPsec. Zhou and Haas [19] primarily discuss key management and they devote a section to secure routing, but essentially conclude that, nodes can protect routing information in the same way they protect data traffic. They also observe that denial-of-service attacks against routing will be treated as damage and routed around. Security issues with routing in general have been addressed by

several researchers [6, 16]. Lately, some work has been done to secure ad hoc networks by using misbehavior detection schemes [10]. This approach has two main problems: first, it is quite likely that it will be not feasible to detect several kinds of misbehaving (especially because it is very hard to distinguish misbehaving from transmission failures and other kind of failures), second, it has no real means to guarantee the integrity and authentication of the routing messages.

Dahill et al. [4] proposed ARAN, a routing protocol for ad hoc networks that uses authentication and requires the use of a trusted certificate server. In ARAN, every node that forwards a route discovery or a route reply message must also sign it, (which is very computing power consuming and causes the size of the routing messages to increase at each hop), whereas the proposed work of this paper only require originators to sign the message. In addition, it is prone to reply attacks using error messages unless the nodes have time synchronization. Papadimitriou and Haas [11] proposed a protocol named SRP that can be applied to several existing routing protocols (in particular DSR [8] and IERP [5]). SRP requires that, for every route discovery, source and destination must have a security association between them. Furthermore, this paper does not even mention route error messages. Therefore, they are not protected, and any malicious node can just forge error messages with other nodes as source. Hash chains being used as an efficient way to obtain authentication in several approaches that tried to secure routing protocols. The authors of [6], [3], [13] use them in order to provide delayed key disclosure. While, in [18], hash chains is used to create one-time signatures that can be verified immediately. The main drawback of all the above approaches is that all of them require clock synchronization.

In SEAD [7] hash chains are also used in combination with DSDV-SQ [2]. At every given time each node has its own hash chain. The hash chain is divided into segments; elements in a segment are used to secure hop counts in a similar way as in SAODV. The size of the hash chain is determined when it is generated. After using all the elements of the hash chain, a new one must be computed. SEAD can be used with any suitable authentication and key distribution scheme. But finding such a scheme is not straightforward. It is quite likely that, for a small team of nodes that trusts each other and wants to create an ad hoc network, where the messages are only routed by members of the team. The simplest way to keep secret their communications is to encrypt all messages (routing and data) with a "team key". Every member of the team would know the key and, therefore, it would be able to encrypt and decrypt every single packet. Nevertheless, this does not scale well and the members of the team have to trust each other. So it can be only used for a very small subset of the possible scenarios. Looking at the work that had been done in this area previously, we felt that the security needs for ad hoc networks have not been satisfied yet (at least for those scenarios where everybody can freely participate in the network). The next section, specify that, what are those needs in the format of a list of security requirements.

3. Attacks on adhoc networks

Attacks on ad hoc network routing protocols generally falls into one of two main categories:

- **Routing-disruption attacks.** The attacker attempts to cause legitimate data packets to be routed in dysfunctional ways.
- **Resource-consumption attacks.** The attacker injects packets into the network in an attempt to consume valuable network resources such as bandwidth or to consume node resources such as memory or computation power. From an application-layer perspective, both attacks are instances of a denial-of-service (DoS) attack. An example of a routing-disruption attack is for an attacker to send forged routing packets to create a routing loop, causing packets to traverse nodes in a cycle without reaching their destination, thus consuming energy and available bandwidth. An attacker might similarly create a routing *black hole*, which attracts and drops data packets.

An attacker also might attempt to cause a node to use a route detour (suboptimal routes), or partition the network by injecting forged routing information to prevent one set of nodes from reaching another. An attacker might attempt to make a route through itself appear longer by adding virtual nodes to the route; we call this attack a *gratuitous detour* because a shorter route exists and would otherwise have been used. The routing protocol of ad hoc network, attempts to keep track of perceived malicious nodes in a

blacklist at each node, as in the watchdog and path rater protocol. That is an attacker might malign a good node, causing the other good nodes to add that node to their blacklists, and thus, avoid that node in future routes. A more subtle type of routing-disruption attack is creating a *wormhole* in the network, using a pair of attacker nodes *A* and *B* are linked via a private network connection. We give an example of this attack and a countermeasure in the next section.

A *rushing* attack is a malicious attack that is targeted against on-demand routing protocols that use duplicate suppression at each node. In the following section, we discuss about the design of SO2P using various module in order to prevent the message hackers from the attacks. The corresponding position information is obtained by location information of the surrounding nodes. These design procedures are needed, so as to find the optimal path finding with respect to the location condition of nodes.

4. Design of the SO2P:

The following are the modules involved to prevent message hacking information from internal and external attackers.

1. Position Tracking Module.
2. Secured Key Generating Module.
3. Broadcast Route Initiation Module.
4. Broadcast Route Reply Module.
5. Optimal path Attack Preventing

4.1 Position Tracking Module

In this first module, the source is supposed to get the destination information from the server. The server will send the desired destination address with its position to the requested source. The server will record all the information about its terminal nodes. The server with the process of GPS system does this information tracking. Here the source requests the position of the destination, which is Position Request (PREQ) to the server. The server with its recorded information sends the reply to the source, which is Position Reply (PREP) to the source, which includes destination address with its position. This Position Reply will be the input to the second secured key generating module, which generates a secured key for the data transmission to the desired destination.

4.2 Secured Key Generating Module

In this module, the input is the Position Reply from the first module, which includes the destination address and its position. This module does the process of secured key generation with the destination address and the position, which achieves the security for the data transmission to the destination. The Data Encryption Standard (DES) algorithm does the security, which is one of the best ad-hoc security algorithms for protecting against the internal attackers from hacking the information about the destination and its data.

A DES key consists of 64 binary digits ("0"s or "1"s) of which 56 bits are randomly generated and used directly by the algorithm. The other 8 bits, which are not used by the algorithm, may be used for error detection. Security of the data depends on the security provided for the key used to encipher and decipher the data. Data can be recovered from cipher, only by using, exactly the same key used to encipher it. Unauthorized recipients of the cipher who know the algorithm, but do not have the correct key cannot derive the original data algorithmically. However, it may be feasible to determine the key by a brute force exhaustion attack. Also, anyone who does have the key and the algorithm can easily decipher the cipher and obtain the original data. Thus a standard algorithm based on a secure key, provides a basis for exchanging encrypted computer data by issuing the key used to encipher it to those authorized to have the data. The data that is considered sensitive by the responsible authority or data that

represents a high value should be cryptographically protected, if it is vulnerable to unauthorized disclosure or undetected modification during transmission or while in storage. This algorithm is designed to encipher and decipher blocks of data consisting of 64 bits under control of a 64-bit key¹. Deciphering must be accomplished by using the same key as for enciphering, but with the schedule of addressing the key bits are altered, so that the deciphering process is the reverse of the enciphering process. A block to be enciphered is subjected to an initial permutation **IP**, then to a complex key-dependent computation and finally to a permutation which is the inverse of the initial permutation.

4.2.1 Encryption process

The 64 bits of the input block to be enciphered are first subjected to the following permutation, called the initial permutation **IP**:

IP

58 50 42 34 26 18 10 2
 60 52 44 36 28 20 12 4
 62 54 46 38 30 22 14 6
 64 56 48 40 32 24 16 8
 57 49 41 33 25 17 9 1
 59 51 43 35 27 19 11 3
 61 53 45 37 29 21 13 5
 63 55 47 39 31 23 15 7

That is the permuted input has bit 58 of the input as its first bit, bit 50 as its second bit, and so on with bit 7 as its last bit. The permuted input block is then the input to a complex key-dependent computation described below. The output of that computation, called the preoutput, is then subjected to the following permutation which is the inverse of the initial permutation:

IP-1

40 8 48 16 56 24 64 32
 39 7 47 15 55 23 63 31
 38 6 46 14 54 22 62 30
 37 5 45 13 53 21 61 29
 36 4 44 12 52 20 60 28
 35 3 43 11 51 19 59 27
 34 2 42 10 50 18 58 26
 33 1 41 9 49 17 57 25

That is, the output of the algorithm has bit 40 of the preoutput block as its first bit, bit 8 as its second bit, and so on, until bit 25 of the preoutput block is the last bit of the output 11. The computation, which uses the permuted input block as its input to produce the preoutput block consists, of a final interchange of blocks, of 16 iterations of a calculation that is described below in terms of the cipher function **f**, which operates on two blocks, one of 32 bits and one of 48 bits, and produces a block of 32 bits. Let the 64 bits of the input block to an iteration consist of a 32 bit block **L** followed by a 32 bit block **R**. Using the notation defined in the introduction, the input block is then **LR**. Let **K** be a block of 48 bits chosen from the 64-bit key. Then the output **L'R'** of iteration with input **LR** is defined by the following equation 1.

$$L'R' = LR \hat{\Delta} f(R, K) \quad (1)$$

Where $\hat{\Delta}$ denotes bit-by-bit addition modulo 2. As remarked before, the input of the first iteration of the calculation is the permuted input block. At each iteration a different block **K** of key bits is chosen from the 64-bit key designated by KEY. Let **KS** be a function which takes an integer *n* in the range from 1 to 16, and a 64-bit block KEY as input and yields as output a 48-bit block. **Kn** is a permuted selection of bits from KEY can be written as per equation 2.

$$Kn = KS(n, KEY) \quad (2)$$

This K_n is determined by the bits of 48 distinct bit positions of KEY. KS is called the key schedule because, the block K used in the n 'th iteration of equation (1) is the block K_n determined by equation (2). As before, let the permuted input block be LR . Finally, let L' and R' are L_n and R_n respectively and L and R are L_{n-1} and R_{n-1} respectively and K is K_n , we can write

$$L_n R_n = L_{n-1} R_{n-1} \hat{\Delta} f(R_{n-1}, K_n) \quad (3)$$

Where n is in the range from 1 to 16. The preoutput block is then $R_{16}L_{16}$. That is, the key schedule produces the 16 K_n which are required for the algorithm.

4.2.2 Decryption process

The permutation IP^{-1} applied to the preoutput block is the inverse of the initial permutation IP applied to the input. Further, from equation (1) it follows that

$$R = L' \quad (4)$$

$$L = R' \hat{\Delta} f(L', K) \quad (5)$$

Consequently, to decipher it is only necessary to apply the very same algorithm to an enciphered message block. Using the notation of the previous section, this can be expressed by the equations (6) and (7) as follow.

$$R_{n-1} = L_n \quad (6)$$

$$L_{n-1} = R_n \hat{\Delta} f(L_n, K_n) \quad (7)$$

Where $R_{16}L_{16}$ is the permuted input block for the deciphering calculation and $L_{0}R_{0}$ is the preoutput block. That is, for the decipherment calculation with $R_{16}L_{16}$ as the permuted input, K_{16} is used in the first iteration, K_{15} in the second, and so on, with K_1 used in the 16th iteration.

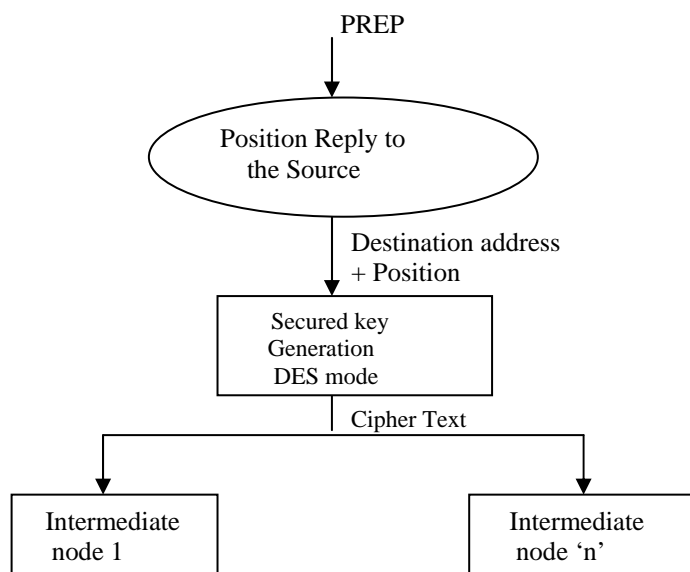


Figure 1. Secured Key Generating

• **The Cipher Function f**

A sketch of the calculation of $f(R,K)$ is given below, Let E denote a function which takes a block of 32 bits as input and yields a block of 48 bits as output. Let E be such that the 48 bits of its output, written as 8 blocks of 6 bits each, are obtained by selecting the bits in its inputs in order according to the following table.

• **E BIT-Selection Table**

```

32 1 2 3 4 5
4 5 6 7 8 9
8 9 10 11 12 13
12 13 14 15 16 17
16 17 18 19 20 21
20 21 22 23 24 25
24 25 26 27 28 29
28 29 30 31 32 1
    
```

Thus the first three bits of $E(R)$ are the bits in positions 32, 1 and 2 of R while the last 2 bits of $E(R)$ are the bits in positions 32 and 1. Each of the unique selection functions $S1, S2... S8$ takes a 6-bit block as input, and yields a 4-bit block as output which is illustrated by using a table containing the recommended $S1$. The position reply from the first module is given to the secured key-generating mode, which generates the key by using the destination secured encrypted cipher text. This is then broadcasted to all the intermediate terminal nodes involved in the ad-hoc networks. This process is said to be Broadcast Route initiation process. The output of the module is the broadcasting of the secured cipher text to all intermediate nodes which is as shown in Figure 1.

4.3 Broadcast route Initiation Module

In this module the cipher text from the source is the input to all the intermediate nodes involved in the ad-hoc networks as shown in the Figure 2. This cipher text is the input to the secured key decryption mode. This cipher text is decrypted with the same DES security algorithm, which finds the plain text. This plain text is the desired destination address for which the source needs to send the desired data. The output of this module is the input to the next Broadcast route Reply module.

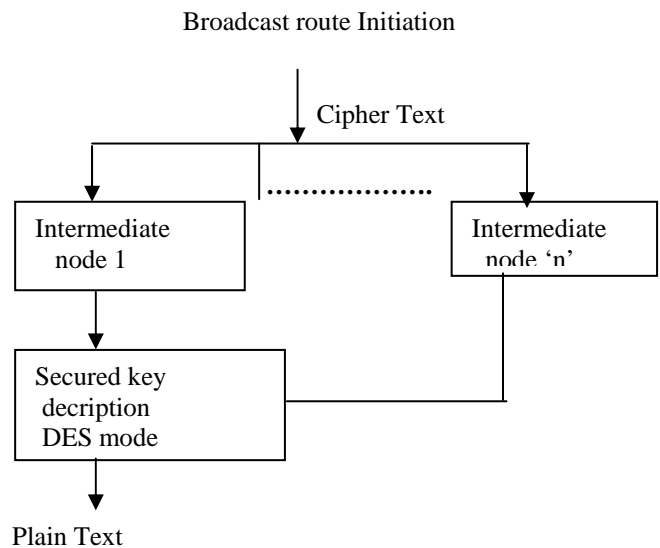


Figure 2. Broadcast Route Initiation

4.4 Broadcast route Reply Module

In this module, the plain text from the third module, which is secured using decryption DES mode is used to check whether it is equal to the desired destination by the destination address checking mode. As per Figure 3 and 4 the destination address checking mode gets the plain text as its input. This mode checks, whether the plain text is equal to the destination address and reply that information to the requested source. If that plain text is equal to the requested destination address, then the intermediate node will request the data to the source, by sending the route reply with its matched destination address (see Figure 5). So, now this node reply to the source is the output of this module, which is given as the input to the next optimal path attack-preventing module.

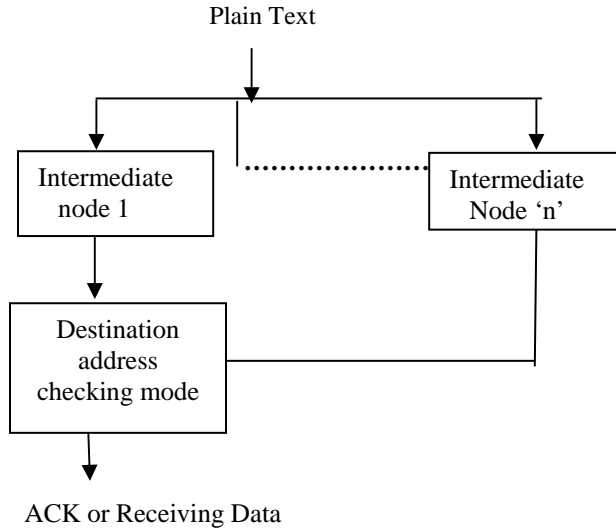


Figure 3. Broadcast Route Reply

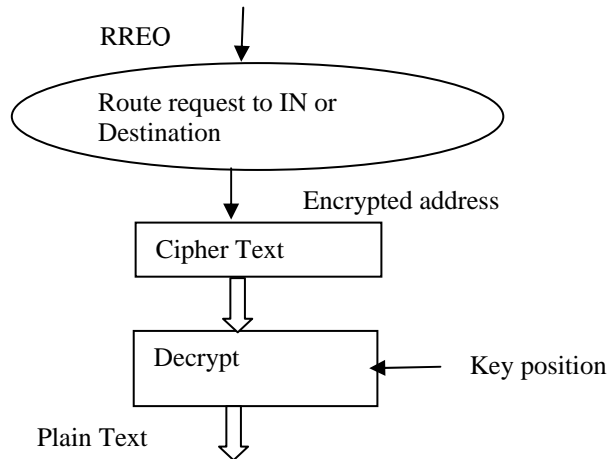


Figure 4. Secured Key Decryption DES mode

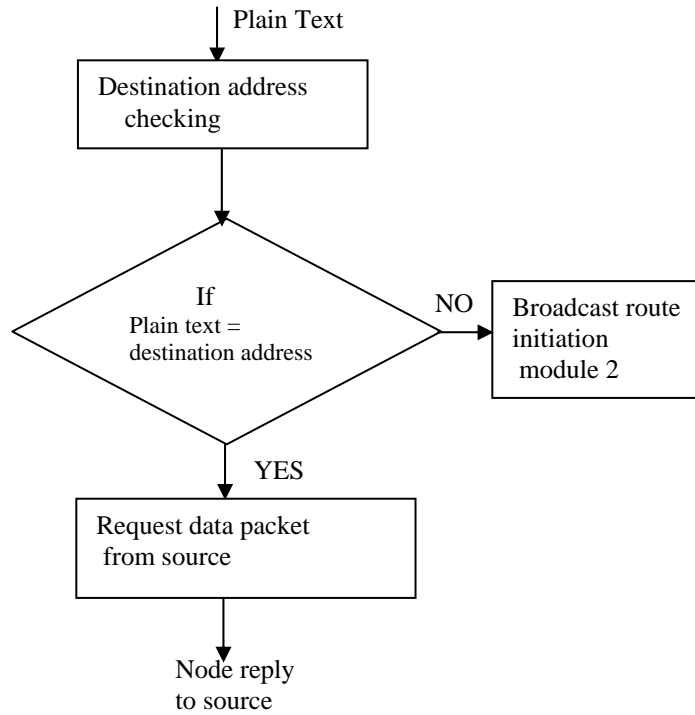


Figure 5. Broadcast Route Reply

4.5 Optimal Path Attack Preventing Module:

The module as shown in the figure 6, the node reply is the input to the source, which is received from the intermediate node. This node reply is given to the optimal route analysis mode. This optimal route analysis mode does the process of selecting the best optimal path for sending the desired data to the destination. Here the optimal route analysis mode gets the IN node reply as the input, which has the sequence number and the HOP number.

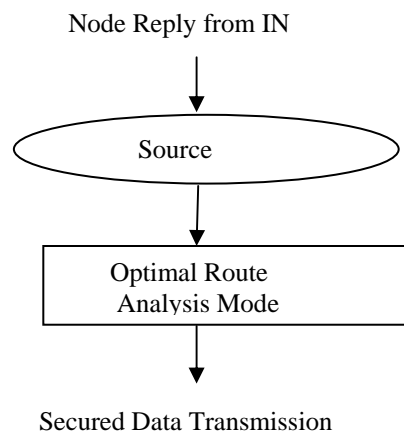


Figure 6. Optimal Path Attack Preventing

The Sequence number is the number of route reply received and the HOP number is the number of terminal crossed in that ad-hoc networks. Now this optimal route analysis mode with its input checks for the validation for the greatest sequence number with the less HOP number which is shown in Figure 7. So the destination with this condition is selected as the desired destination and the data packet is decided to send to the destination. If that destination does not have the greatest sequence number and less HOP number, then that intermediate node path is rejected. So the output of this module is the secured data transmission to the desired destination.

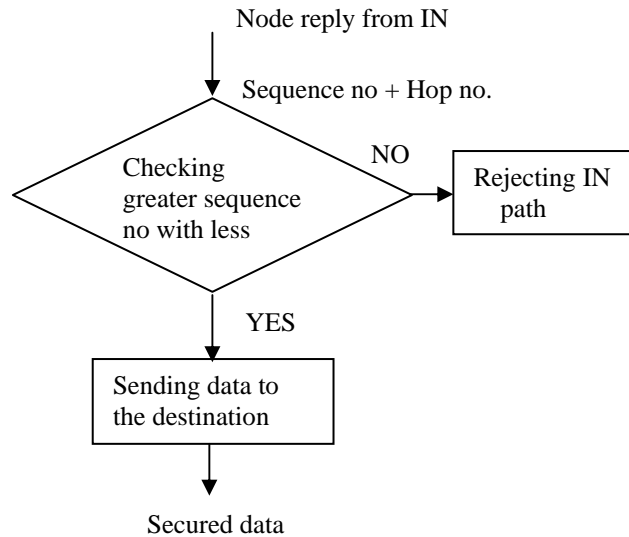


Figure 7. Optimal Route Analysis mode

Hence, we found the optimal route along with security by considering the location information. Also in our approach the server along with GPS records all the information about the terminal nodes which yields better performance.

5. Simulation of SO2P

We have used Global Mobile Information System Simulator (GloMoSim) is a scalable simulation environment for large wireless and wire line communication networks. GloMoSim uses a parallel discrete-event simulation capability provided by Parsec. GloMoSim simulates networks with up to thousand nodes linked by a heterogeneous communications capability that includes multicast, asymmetric communications using direct satellite broadcasts, multi-hop wireless communications using ad-hoc networking, and traditional Internet protocols. The following table lists the GloMoSim models currently available at each of the major layers:

Table 1.Layer Models

Layer	Models
Physical(Radio propagation)	Free space, Two way
Data Link (MAC)	CSMA, MACA, TSMA, 802.11
Network (Routing)	Bellman-Ford, OSPF, DSR, WRP, AODV
Transport	TCP, UDP
Application	Telnet, FTP

The *node aggregation technique* is introduced into GloMoSim to give significant benefits to the simulation performance. Initializing each node as a separate entity inherently limits the scalability because, the memory requirements increase dramatically for a model with large number of nodes. With node aggregation, a single entity can simulate several network nodes in the system. Node aggregation

technique implies that, the number of nodes in the system can be increased while maintaining the same number of entities in the simulation. In GloMoSim, each entity represents a geographical area of the simulation. Thus a system for providing position information by using a wireless personal area network, and a method of acquiring the position information of a mobile device thereof where the system includes: a first node including referenced position information; and second node calculating their respective position information based on the reference position information of the first node, wherein the mobile device calculates its position information by using the reference position information received from the first node and/or one or more of the respective position information of the second nodes. According to the system and method, at an indoor place where a global positioning system (GPS) signal is weak or absent, the GPS position information can be acquired through the position information providing system using a low-price wireless LAN communication network. Hence the network nodes in which a particular entity represents are determined by the physical position of the nodes.

5.1 Input Formats of SO2P Routing

The following are the network parameter used in our model.

- From config file of GloMoSim:
 - TERRAIN-DIMENSIONS (900, 1000)
 - NUMBER-OF-NODES 20
 - NODE-PLACEMENT FILE
 - NODE-PLACEMENT-FILE ./nodes. Input
 - MOBILITY TRACE
 - MOBILITY-TRACE-FILE ./mobility. in
 - RADIO-BANDWIDTH 50000
 - ROUTING-PROTOCOL SO2P
 - APP-CONFIG-FILE ./cbr.in

From cbr.in file of GloMoSim

- CBR 0 6 0 512 3S 40S 170S
- CBR 12 19 0 512 3S 80S 200S
- CBR 24 30 0 512 2S 0S 150S

From mobility. in file of GloMoSim

- 2 500MS (350, 50, 0)
- 15 500MS (510, 585, 0)
- 28 500MS (525, 735, 0)

From nodes. input file of GloMoSim

```
0 (100, 200) 10 (550, 300)
1 (200, 200) 11 (300, 400)
2 (300, 200) 12 (150, 500)
3 (400, 200) 13 (250, 500)
4 (500, 200) 14 (350, 500)
5 (600, 200) 15 (450, 500)
6 (700, 200) 16 (550, 500)
7 (250, 100) 17 (650, 500)
8 (450, 100) 18 (750, 500)
9 (350, 300) 19 (850, 500)
```

5.2 Simulation output of SO2P

The following screen of Figure 8 is the output of data packets delivery in Secured On demand Position Based Private Routing Protocol.

```

C:\glomosin\glomosin\bin\glomosin.exe
Current $in Time[s] = 36.019736908 Real Time[s] = 0 Completed 4%
Current $in Time[s] = 45.008464022 Real Time[s] = 0 Completed 5%
Current $in Time[s] = 54.030933774 Real Time[s] = 0 Completed 6%
Current $in Time[s] = 63.050930530 Real Time[s] = 0 Completed 7%
Current $in Time[s] = 72.074550054 Real Time[s] = 0 Completed 8%
Current $in Time[s] = 81.004277417 Real Time[s] = 0 Completed 9%
Current $in Time[s] = 90.000000000 Real Time[s] = 0 Completed 10%
Current $in Time[s] = 99.023300744 Real Time[s] = 0 Completed 11%
Current $in Time[s] = 108.031918445 Real Time[s] = 0 Completed 12%
Current $in Time[s] = 117.000186342 Real Time[s] = 0 Completed 13%
Current $in Time[s] = 126.001754878 Real Time[s] = 0 Completed 14%
Current $in Time[s] = 135.000526945 Real Time[s] = 0 Completed 15%
Current $in Time[s] = 144.000563992 Real Time[s] = 0 Completed 16%
Current $in Time[s] = 153.000507350 Real Time[s] = 1 Completed 17%
Current $in Time[s] = 162.000001000 Real Time[s] = 1 Completed 18%
Current $in Time[s] = 171.000448342 Real Time[s] = 1 Completed 19%
Current $in Time[s] = 180.000001000 Real Time[s] = 1 Completed 20%
Current $in Time[s] = 189.000307350 Real Time[s] = 1 Completed 21%
Current $in Time[s] = 198.000000000 Real Time[s] = 1 Completed 22%
Current $in Time[s] = 207.000548342 Real Time[s] = 1 Completed 23%
Current $in Time[s] = 216.019273074 Real Time[s] = 1 Completed 24%
Current $in Time[s] = 225.000407350 Real Time[s] = 1 Completed 25%
Current $in Time[s] = 234.008501767 Real Time[s] = 1 Completed 26%
Current $in Time[s] = 243.000008342 Real Time[s] = 2 Completed 27%

```

Figure 8. Output of data packets delivery in Secured On demand Position Based Private Routing Protocol (SO2P)

- Node:2, Layer: RoutingAo2p, Number of Routes Selected = 2
- Node: 2, Layer: RoutingAo2p, Number of Data Txed = 2
- Node: 13, Layer: RoutingAo2p, Number of Data Packets Received = 2
- Node: 19, Layer: RoutingAo2p, Number of Routes Selected = 5
- Node: 19, Layer: RoutingAo2p, Number of Data Txed = 5
- Node: 18, Layer: RoutingAo2p, Number of Data Packets Received = 2
- Node: 11, Layer: RoutingAo2p, Number of Routes Selected = 7
- Node: 11, Layer: RoutingAo2p, Number of Data Txed = 7
- Node: 10, Layer: RoutingAo2p, Number of Data Packets Received = 3

6. Conclusion

Thus, the proposed SO2P provides a security for Ad-hoc On Demand Position based private Routing Protocol. It prevents position information from internal and external hackers. This is done by selecting a new secured routing algorithm in mobile ad-hoc networks in order to find the optimal path to reach the destination. The new approach of DES algorithm is used to prevent an internal and external attacker in the mobile environment, which proves to be the best in its kind. This SO2P faces some difficulties in the performance of routing because of security mechanisms. Example high packet loss ratio, less throughput, more time in end to end connection delay. So in future, we are planning to improve the performance of SO2P using backup routing mechanism. This mechanism provides a sophisticated back-up routing protocol for mobile ad-hoc network in case of routing path failure occurred in networks.

7. References

[1] N. Asokan, P. Ginzboorg, 2000. Key agreement in Ad-hoc networks. *In Computer Communication Review*, 23(17), pg: 1627–1637.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, 2004. A performance comparison of multi-hop wireless ad hoc network routing protocols. *In Proceedings of the Fourth Annual International Conference on mobile Computing and Networking*, pg 85–97.

[3] S. Cheung, 1997. An efficient message authentication scheme for link state routing. *In 13th Annual Computer Security Applications Conference*, pg 90–98.

[4] B. Dahill, B. N. Levine, E. Royer and C. Shields, 2001. A secure routing protocol for ad hoc networks. *Technical Report UM-CS-2001-037, University of Massachusetts, Department of Computer Science.*

[5] Z. J. Haas, M. R. Pearlman, and P. Samar, 2002. The interzone routing protocol (IERP) for ad hoc networks. *INTERNET DRAFT, MANET working group, July 2002. draft-ietf-manet-zone-ierp-02.txt.*

- [6] R. Hauser, A. Przygienda, and G. Tsudik, 1997. Reducing the cost of security in link state routing. *In Symposium on Network and Distributed Systems Security (NDSS '97)*, pg 93–99, California.
- [7] Y. C. Hu, D. Johnson, and A. Perrig, 2002. SEAD: Secure efficient distance vector routing for mobile wireless adhoc networks. *In Fourth IEEE Workshop on mobile Computing Systems and Applications (WMCSA '02)*, pg 3–13.
- [8] D. B. Johnson et al. 2002. The dynamic source routing protocol for mobile ad hoc networks (DSR). *INTERNET DRAFT, MANET working group, Feb 2002. draft-ietf-manet-dsr-07.txt*.
- [9] S. Kent, C. Lynn, J. Mikkelsen, and K. Seo, 2000. Secure border gateway protocol (S-BGP) — *White paper real world performance and deployment issues* 2(3), pg 67-74.
- [10] S. Marti, T. J. Giuli, K. Lai, and M. Baker, 2000. Mitigating routing misbehavior in mobile ad hoc Networks. *In Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking*, 5(2), pg: 255–265.
- [11] P. Papadimitratos and Z. J. Haas 2002 . Secure routing for mobile ad hoc networks. *In SCS Communication Networks and Distributed Systems Modeling and Simulation Conference*
- [12] R. Perlman, 1983. Fault-tolerant broadcast of routing information. *In Computer Networks Review*, 3(7), pg:395–405.
- [13] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, 2001. SPINS: security protocols for sensor networks. *In Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking*, pg: 189–199.
- [14] S. Ramanathan and M. Steenstrup, 2005. A survey of Routing techniques for mobile communications networks. *In proceedings of Mobile Networks and Applications*, 1(2): pg : 89–104.
- [15] E. M. Royer and C.-K. Toh, 1999. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pg: 46–55.
- [16] B.R. Smith, S. Murthy, and J. Garcia-Luna-Aceves, 2006. Securing distance-vector routing protocols. *In Symposium on Network and Distributed Systems Security (NDSS '06)* [11], pg 85–92.
- [17] F. Stajano and R. Anderson, 1999. The resurrecting Duckling: Security issues for ad-hoc wireless networks. *In Proceedings of the 7th International Workshop on Security Protocols, number 1796 in Lecture Notes in Computer Science*, pages 172–194 Springer-Verlag, Berlin, Germany.
- [18] K. Zhang, 2001 . Efficient protocols for signing routing messages. *In Proceedings of the Symposium on Network and Distributed Systems Security (NDSS'01)*.
- [19] L. Zhou and Z. J. Haas, 1999. Securing ad hoc networks. *IEEE Network Magazine*, 13(6), Pg:24–30.
- [20] H. Krawczyk, 2000. Simple forward-secure signatures from any signature scheme. *In ACM Conference on Computer and Communications Security*, pg: 108–115.