Minimum Exposed Path to the Attack ($MEPA$) in Mobile Adhoc Network ($MANET$)

Sandhya Khurana
Department of Computer Science,
University of Delhi
skhurana@cs.du.ac.in

Neelima Gupta
Department of Computer Science
University of Delhi
ngupta@cs.du.ac.in

Nagender Aneja
Department of Computer Science
Tecnia Institute of Advanced Studies
GGSIP University, Delhi
naneja@gmail.com

November 20, 2006

Abstract

Lack of infrastructure, central controlling authority and the properties of wireless links make Mobile Ad hoc Networks ($MANET$s) vulnerable to attacks. Several protocols have been proposed to make the routing protocols handle attacks in $MANET$s. These protocols detect the misbehaving nodes and re-route the data packets around them, mostly along the shortest such path. However, no single protocol handles all the attacks. A variant of the problem for routing around misbehaving nodes in ad hoc networks can be stated as: given a set of nodes under the danger of attack, one wishes to determine the path which is farthest from the endangered nodes. The problem does not address the problem of handling attack directly but tries to minimize the impact of attack. The problem also finds its applications in sensor networks. In this paper, we present a simple and efficient algorithm to solve the problem.The algorithm converges in $O(d^2)$ time where d is the diameter of the network.

1 Introduction

Ad-hoc networks [1] have been proposed to support scenarios where no wired infrastructure exists. They can be set up quickly where the existing infrastructure does not meet application requirements for reasons such as security, cost, or quality. Examples of applications for ad hoc networks range from military operations, emergency disaster relief to community networking and interaction between attendees at a meeting or students during a lecture.

Attacks in $MANET$s [2] are threat for basic network functions like packet forwarding and routing. Achieving $MANET$s free from attacks is challenging due to the lack of infrastructure, dynamically changing topologies, wireless links vulnerable to attacks like eavesdropping and spoofing. The dynamic nature of the network emphasizes the need for solutions to be dynamic.

Routing of packets is one of the basic functions performed in any network. Nodes of an ad hoc network rely on each other to forward the packets due to the limited range of the nodes. As a result, embedding solutions in routing protocols to handle attacks poses a challenge to the researchers. The major threats to the routing in ad hoc networks are due to attacks by malicious nodes [3] and selfish nodes [4] . Malicious nodes attack the network by performing some harmful operations at the cost of their battery life whereas selfish nodes do not cooperate in the normal functioning of the network to save their battery life for their own communication. Malicious nodes can cripple the network by inserting erroneous routing updates, replaying old routing information, changing routing updates, or advertising incorrect routing information so that the network is not able to provide service properly. Attacks like reducing the amount of routing information available to other nodes, failing to advertise certain routes or discarding routing packets or parts of routing packets are due to selfish behavior of a node.

Two main approaches are used to make routing protocols handle attacks in ad hoc networks. The first approach aims at detecting the malicious nodes while computing the route in the network and re-routing the packets around it, mostly along the shortest path among them. Most of these protocols [5, 6, 7, 8, 9, 10, 11, 12, 13] are based on existing ad hoc routing protocols like AODV [14], DSDV [15] and DSR [16], redesigned to handle attacks. The second approach [17, 18, 19] separates the detection of malicious nodes from routing.

Farago in [20] has posed a variant of the problem for routing in ad hoc networks. Given a subset of nodes that are in danger of attack or jamming, find a path which is farthest from these nodes, i.e., the smallest occurring hop distance between a path node and an endangered node is maximum. This path can be viewed as minimally exposed path to the attack or jamming ($MEPA$). He has posed this problem as a challenging algorithmic problem for ad hoc networks. Since most of the existing protocols do not handle all the attacks, it is imperative to find a solution that would reduce the impact of attacks on routing. A solution to the problem posed by Farago aims at achieving this goal.

The problem also finds its application in sensor networks. The solution can be used to place and organize the sensors such that the probability that an intruder can avoid detection is minimized. The related problem in the context of sensor networks is called maximal breach path problem. Maximal breach path is defined as the path which is as far away as possible from the sensors i.e the minimum distance from sensors is maximized on this path.

In this paper, we present a simple and efficient solution to the Farago's problem. It consists of a bootstrap phase where all the nodes compute their distances from the endangered nodes. Once the initial distances are computed, $MEPA$ routes are discovered in a manner similar to that in AODV . The algorithm is simple to implement and converges in $O(d^2)$ time where d is the diameter of the network. Hence our algorithm solves the problem posed by Farago efficiently.

1.1 Related Work

Several secure routing protocols [5, 6, 7, 8, 9, 10, 11, 12, 13] exist to handle the attacks in $MANET$s. For example, Secure Routing Protocol proposed in [5] based on DSR assures that a node initiating route discovery is able to identify and discard replies providing false routing information but it fails when two or more malicious nodes cooperate resulting in wormhole attack. ARIADNE [6] and Security-aware Ad-hoc Routing (SAR) [12] protocol handle attacks such as spoofing, changing routing updates with the help of keys and certification of messages. However, they do not handle the attack by selfish nodes. Selfish nodes do not intend to cripple the network but do not cooperate in the normal functioning of the basic services like packet forwarding in order to save energy. In [9] Buttyan and Hubaux have suggested a solution based on virtual currency called Nuglet to locate the selfish nodes in the network. It handles attacks due to selfish nodes but not the attacks due to malicious nodes. CONFIDANT(Cooperation Of Nodes: Fairness In Dynamic Ad-hoc NeTworks) by Buchegger and Boudec [10], CORE (COllaborative REputation) by Michiardi and Molva [11] and RAODV by Khurana et al. in [13] detect the malicious or selfish nodes , isolate them and route the packets around them. However they are vulnerable to spoofing attacks.

Besides finding minimally exposed path from endangered nodes in $MANET$s, the problem also finds its application in sensor networks. In [21] Meguerdichian et al. have addressed the problem of computing a Maximal Breach Path in a sensor network. Maximal Breach Path is a path with the property that for any point p on the path, the distance from p to the closest sensor is maximized. The path finds the area of low observability from the sensor nodes. A solution to maximal breach path problem can be used to compare two given configurations of sensors in a battlefield or, to add new sensors to a network to increase the observability or the coverage area.

The algorithm presented in [21] uses Voronoi diagram and takes $O(n^2 \log n)$ time where n is the number of sensor nodes. Dinesh et al. [22] and Lie et al. [23] have improved the running time to $O(n \log n)$. In [22], after a preprocessing step, they compute the maximal breach path P in optimal $O(|P|)$ time, where $|P|$ denotes the number of edges on the optimal path. The preprocessing takes $O(n \log n)$ time.

In [24] Hai Huang et al. have addressed the problem of maintaining the distance of the maximal breach path in a dynamic scenario. The maximal breach path changes if the topology of ad hoc sensor network changes. For example, a sensor node may be down due to its low battery power or a new sensor node may be inserted to improve the coverage of the sensor network. The network designer may be interested in knowing how the topology change affects the coverage of the network or in other words, how the distance of the maximal breach path (from the nearest sensor) changes. They have given an algorithm that approximates the distance of maximal breach path within an approximation factor of $(\sqrt{2}+\epsilon)$, for any $\epsilon > 0$ in polylogarthmic time. If required, the maximal breach path can be computed in $O(n \log n)$ time.

2 Algorithm to compute the $MEPA$ route

The problem we address in this paper is due to Farago [20]. It is defined as follows: "given a subset of nodes that are in danger of attack or jamming, find a path which is farthest from these nodes, i.e. the smallest occurring hop distance between a path node and an endangered node is maximum". This path can be viewed as minimally exposed path to the attack or jamming.

We assume the existence of a mechanism that enables the participating nodes to detect the nodes under attack. In [17, 18, 19] Lee etal have used intrusion detection techniques to detect the presence of an intruder in wireless ad hoc networks.

Our algorithm works in two phases. Phase-I is a bootstrap phase in which all the nodes compute their distances from the endangered nodes. Once the distances from the endangered nodes have been computed, in Phase-II, nodes can set up $MEPA$ routes. Due to the highly mobile nature of the nodes, the distances may have to be updated dynamically as the nodes move. For the sake of simplicity and better understanding we have presented the algorithm in two steps. In actual implementation they occur simultaneously and the convergence is guaranteed in time polynomial in the diameter of the network.

2.1 Phase-I of the algorithm : The Bootstrap phase

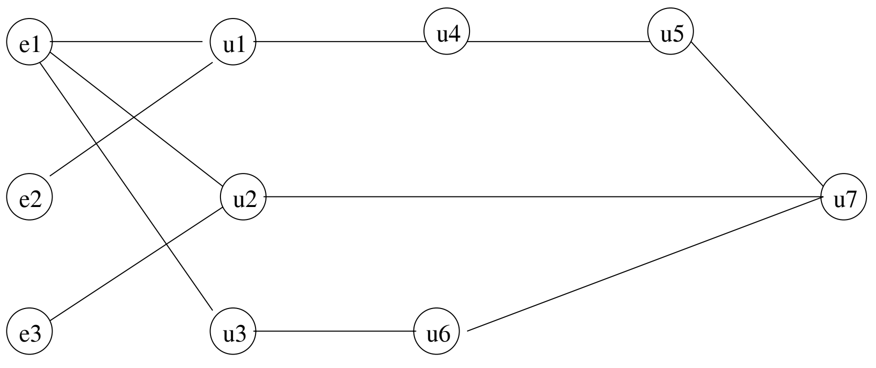

Figure 1: $E = \{e1, e2, e3\}$ and $Nb(E) = \{u1, u2, u3\}$.

Let E be the known subset of nodes that are in danger of attack or jamming. Let $Nb(E)$ be the set of nodes in the neighborhood of E i.e. $Nb(E)$ is the set of nodes, which are at one hop distance from at least one node in E. For example, in Figure 1, Let $E = \{e1, e2, e3\}$, then $Nb(E) = \{u1, u2, u3\}$. For a node $u \in Nb(E)$, let $in\_nbhd(u)$ be a flag that denotes whether u is in the neighborhood of E or not. For any node u in the network, let $\delta(u)$ denote the minimum distance of node u from the set E i.e. $\delta(u) = min_{e \in E}\{distance(u, e)\}$. In the figure, $\delta(u1) = 1, \delta(u2) = 1, \delta(u3) = 1, \delta(u4) = 2, \delta(u5) = 3, \delta(u6) = 2, \delta(u7) = 2$ For a node $u \notin Nb(u)$, let $pr(u)$ denote the node, in the neighborhood of u, through which the distance of u from E is minimum. In the figure, $pr(u4) = u1, pr(u5) = u4, pr(u6) = u3$ and $pr(u7) = u2$.

In the bootstrap phase, initially the flag $in\_nbhd(u)$ is off and the value of $\delta(u)$ is *infinity* for all the nodes u in the network. Nodes compute their distances from E as follows:

1. Any node $u \in Nb(E)$ knows that it is in $Nb(E)$ either when it receives some packet from an endangered node or it senses so in case of sensor networks. It sets its in_nbhd flag to 1, $\delta(u)$ to 1 and broadcasts its distance to all its neighbors.

2. Let u be a node not in $Nb(E)$. When u receives $\delta(v)$ from its neighbor v it updates its $\delta(u)$ as follows: if the distance of u from E is shorter through v than its current value (that is, if $\delta(v) + 1 < \delta(u)$) then it updates $\delta(u)$ to $\delta(v) + 1$ and sets $pr(u)$ to v.

Whenever a node updates its distance from E, it broadcasts it to all its neighbors except to the one through which the new distance was computed. Assuming that packets arrive from the shortest path first, no updation is done in the bootstrap phase. We will see in the next section that, as the nodes move, distances are updated in the maintenance phase.

2.2 Maintenance of distances

As the nodes in an ad hoc network are highly mobile, distances must be updated as the nodes move and the links break or new links established. Our algorithm updates the distances as described below.

Consider a case when a node moves out of the range of another node:

1. Let $u \in Nb(E)$. As long as, it is in the range of at lease one endangered node, it does nothing. As soon as it stops hearing from all the endangered nodes (for example, when an endangered node has moved or u has moved), it switches off its in_nbhd flag. Now, distances of all those nodes v (including u), whose minimum distance from E was through u, need to be updated. This is done as follows:

   u broadcasts a request for "distance from E" to its neighbors, which in turn pass on the request to their neighbors. The process continues till the request is received by neighbors of E. Now, consider a node v. It may receive the request from its $pr(v)$ or from a node which is not $pr(v)$. That is, it may receive it from a node through which v had shortest path to E or it may not be from such a node. (For example, in Figure 2, the node $u7$ may receive the request from $pr(u7)$ i.e. $u2$ when both the nodes $e1$ and $e3$ have moved out of the range of $u2$ or from $u6$ when $e7$ has moved out of the range of $u3$.) In the former case, $\delta(v)$ needs to be recomputed. So it resets $\delta(v)$ to *infinity* on seeing the request from $pr(v)$ and, broadcasts the request to its neighbors. In the latter case, it broadcasts the request without resetting $\delta(v)$. When other neighbors of E receive the request they do not broadcast it further, and reply to the request by broadcasting their "distances from E". Distances of all the nodes are re-computed as explained earlier.

2. Let u be a node not in $Nb(E)$ and $pr(u)$ moves out of the range of u. In this case u will broadcast the request for "distance from E" to its neighbors and the procedure explained above is repeated.

Next, consider a scenario in which a node u has moved into the range of another node v. If u is an endangered node, then v switches its *in_nbhd* flag on, sets $\delta(v)$ to 1 and broadcasts its distance to its neighbours. The distances of all other nodes not in $Nb(E)$ are updated as explained earlier. If u is not an endangered node, it is possible that for a node v, the distance of v from E, through u, is less than its previous distance. Hence, $\delta(v)$ needs to be recomputed. u will broadcast the request for "distance from E" to its neighbors and the procedure explained above is repeated.

2.3 Phase-II of the algorithm : Establish the $MEPA$ routes

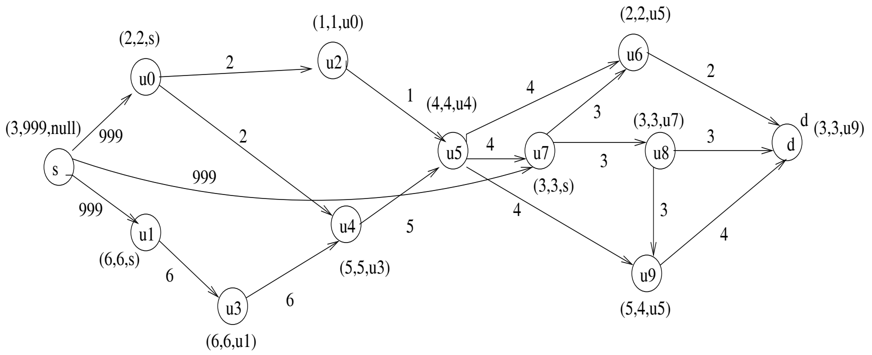

Figure 2: Each node u is labeled with the triplet $(\delta(u), DEN(s,u), p(s,u))$ and edge (u,v) is labeled with $DEN(s,u)$.

Once the distances of all the nodes from the endangered nodes have been setup initially, nodes can communicate with each other. As the nodes move, updation of distances takes place simultaneously.

Let "Minimum Exposed Path to the Attack" from a node u to another node v be denoted by $MEPA(u,v)$ and its Distance from Endangered Nodes by $DEN(u,v)$. Suppose a node s wants to establish a $MEPA$ route to another node d. For a node v, let $N(v)$ be the set of neighbors of v. Then clearly, $DEN(s,v) = min\{max_{v_i \in N(v)}\{DEN(s,v_i)\}, \delta(v)\}$. Let $p(s,v) = v_i$ for which $DEN(s,v_i)$ is largest. Then, $p(s,v)$ denote the next hop for v on the reverse route of $MEPA$. In Figure 2, each node u is labeled with the triplet ($\delta(u)$, $DEN(s,u)$, $p(s,u)$) and edge (u, v) is labeled by $DEN(s,u)$. Consider node $u5$. Three routes from s to $u5$ have "distance from E" as $1, 2$ and 5, maximum being 5. $DEN(s,u5) = min\{5, \delta(u5)\} = min\{5,4\} = 4$. Since distance 5 was received from the node $u4$, $p(s,u5) = u4$. As can be seen from the figure that this indeed is the next hop on the reverse $MEPA$ route from $u5$ to s.

Initially a node u computes approximate $MEPA(s,u)$ and its distance $DEN(s,u)$, which may be updated as more and more information is received from its neighbors. For example, in Figure 2, node $u4$ may receive $DEN(s,u0) = 2$, and set its $DEN(s,u4) = 2$ and $p(s,u4) = u0$. Later when it receives $DEN(s,u3) = 6$, it updates its $DEN(s,u4)$ to 5 and $p(s,u4) = u3$. After every update u broadcasts its $DEN(s,u)$ to its neighbors, which in turn update their DENs if required. Suppose we are at a node u. Let $N^*(u)$ denote the neighbors of u such that approximate $MEPA(s,v_i)$ and approximate $DEN(s,v_i)$ have been computed $\forall v_i \in N^*(u)$. We never consider a node u for which $N^*(u)$ is empty. Thus, we first consider the nodes which are neighbors of s (because for such nodes u, $N^*(u)$ is not empty), then their neighbors and so on. Suppose a node u receives $DEN(s,v)$ from $v \in N^*(u)$ then u updates $DEN(s,u)$ and $p(s,u)$ as follows:

If $DEN(s,v) \leq DEN(s,u)$ there are no updations else there are three cases:

1. $DEN(s,u) > \delta(u)$ is not possible by definition of DEN.

2. $DEN(s,u) = \delta(u)$ and $DEN(s,v) > DEN(s,p(s,u))$ then there will be no updation in $DEN(s,u)$ but $p(s,u)$ will be equal to v. For example, in Figure 2, suppose node $u6$ receives $DEN(s,u7)$ first. It sets $DEN(s,u6) = min\{3,2\} = 2$ and $p(s,u6) = u7$. Next, it receives $DEN(s,u5)$ which is greater than $DEN(s,u6)$ and $DEN(s,u7)$. So, $DEN(s,u6)$ does not change but $p(s,u6)$ is updated to $u5$.

3. $DEN(s,u) < \delta(u)$ then $DEN(s,u)$ will be updated as $min(DEN(s,v), D(u))$ and $p(s,u)$ will be equal to v. This case is exhibited in Figure 2 at node $u4$ and at $u5$ as explained above.

Whenever a node u updates $DEN(s,u)$ it broadcasts it to all its neighbors, which in turn update their DENs if required. We will show that for every node u $MEPA(s,u)$ and $DEN(s,u)$ are not updated more than d times, where d is the diameter of the network. Let $\delta(u) = k$. Then clearly $DEN(s,u)$ is not updated more than k times because in the worst case $DEN(s,u)$ assumes values in the sequence $1, 2, 3, \ldots, k$. Once $DEN(s,u) = k$ it is no longer updated by the path discovered in the future. Since $k \leq d$, the total number of times DEN is updated is no more than d. However, the length of the $MEPA$ route could be as big as d. Hence, the total time for the entire algorithm to converge is $O(d^2)$. This is a theoretical bound. In practice, the updations are done less frequently and the length of the $MEPA$ route is much less than d.

3 Conclusion and Future Work

Handling attacks in $MANET$s is important for the normal functioning of the basic functions like routing. At the same time, because of the lack of infrastructure and any central managing authority, achieving $MANET$s free from attack is a challenging task. Highly mobile nature of the nodes and the properties of the wireless links further make the task of handling attacks more difficult. Most of the existing secure routing protocols handle the attacks by malicious nodes and selfish nodes. However, no single algorithm handles all the attacks. In this paper, we have presented an algorithm that does not handle the attacks but reduces the impact of (any type of) attacks. Given a set of nodes in the danger of attack, we compute a path that is exposed minimally to the set of endangered nodes. The problem was posed by Farago as a challenging algorithmic problem for ad hoc networks. Our algorithm is simple and fast. It computes the secure path in $O(d^2)$ time, where d is the diameter of the network.

With a slight modification,the algorithm can be used to find the shortest $MEPA$ route. We may include the hopcount field in the $MEPA\ REQ$. When a $MEPA\ REQ$ with DEN equal to the current DEN arrives at a node, it compares the hopcounts of the two paths keeping the one with smaller hopcount and discarding the one with the larger hopcount. In the context of sensor networks, the algorithm can be adapted to compute the maximal breach path in the same bounds as the previously known algorithms i.e. in $O(n \log n)$ time where n is number of sensor nodes.

References

[1] Charles E. Perkins, Ad Hoc Networking, Addison Wesley, ISBN 0-201-30976-9 ©2001.

[2] Refik Molva, Pietro Michiardi, “Security in Ad Hoc Networks”, Lecture Notes in Computer Science, Volume 2775, Sep 2003, Pages 756 - 775.

[3] Matthias Hollick, Jens Schmitt, Christian Seipl, and Ralf Steinmetz “On the Effect of Node Misbehavior in Ad Hoc Networks”, http://disco.informatik.uni-kl.de/publications/HSSS04-2.pdf.

[4] Pietro Michiardi and Refik Molva “Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks” http://www.eurecom.fr/ michiard/pubs/mimo02-EW2002.pdf.

[5] P. Papadimitratos, Z. Haas, “Secure Routing for Mobile Ad Hoc Networks”, Proceedings of CNDS 2002.

[6] Y-C Hu, A. Perrig, D. B. Johnson, “Ariadne : A secure On-Demand Routing Protocol for Ad Hoc Networks”, in proceedings of MOBICOM 2002.

[7] B. Dahill, B. N. Levine, E. Royer, C. Shields, “ARAN: A secure Routing Protocol for AdHoc Networks”, UMass Tech Report 02-32, 2002.

[8] Yih-Chun Hu, David B. Johnson, and Adrian Perrig, “SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad hoc Networks”, Proceedings of the 4th IEEE Workshop on Mobile Computing Systems & Applications (WMCSA 2002), IEEE, Calicoon, NY.

[9] L. Buttyan, J.-P. Hubaux, “Nuglets: a virtual currency to stimulate cooperation in selforganized ad hoc networks”, Technical Report DSC/2001/001, Swiss Federal Institute of Technology – Lausanne, 2001.

[10] S. Buchegger and J.-Y. Le Boudec, “Performance analysis of the CONFIDANT protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks)”, Proceedings of The Third ACM International Symposium on Mobile Ad Hoc Networking and Computing, 9-11 June, 2002, Lausanne, Switzerland, ACM Press, 2002, pp. 226-236.

[11] Pietro Michiardi and Refik Molva, “CORE: A COllaborative REputation Mechanism to enforce node cooperation in Mobile Ad hoc Networks”, Sixth IFIP conference on security communications, and multimedia (CMS 2002), Portoroz, Slovenia., 2002.

[12] Seung Yi, Prasad Naldurg, Robin Kravets, “A Security-Aware Routing Protocol for Wireless Ad Hoc Networks”, Proceedings of 6th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI), Orlando, FL, July 2002, pp. 286-292.

[13] Sandhya Khurana, Neelima Gupta, Nagender Aneja, “Reliable Ad-hoc On-demand Distance Vector Routing Protocol,” icn, p. 98, International Conference on Networking, (ICN’06), 2006.

[14] Charles E. Perkins, Elizabeth M. Belding Royer and Samir R. Das, “Ad-hoc On-Demand Distance Vector (AODV) Routing”, Mobile Ad-hoc Networking Working Group, Internet Draft, draft-ietf-manet-aodv-00.txt, February 2003.

[15] Charles E. Perkins and Pravin Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers”, Proceedings of the SIGCOMM ’94 Conference on Communications Architectures, Protocols and Applications, August 1994, pp. 234-244.

[16] D. B. Johnson, D. A. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks”, Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, pages 153- 181, Kluwer Academic Publishers, 1996.

[17] Yongguang Zhang and Wenke Lee, “Intrusion Detection in Wireless Ad-Hoc Networks” Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom’2000), August 6-11,2000, Boston, Massachussetts.

[18] Yi-an Huang, Wenke Lee,“A Cooperative Intrusion Detection System for Ad Hoc Networks”, Proceedings of the First ACM Workshop Security of Ad Hoc and Sensor Networks Fairfax, Virginia.

[19] Yongguang Zhang, Wenke Lee, Yi-an Huang, “Intrusion Detection Techniques for Mobile Wireless Networks” Wireless Networks 9,545-556,2003,©2003 Kluwer Academic Publishers.

[20] Basagni, Conti, Giordano, and Stojmenovic, “Mobile Ad-Hoc Networking”, Institute of Electrical and Electronics Engineers Inc., ISBN 0-471-37313-3 ©2004, pp. 429.

[21] S. Meguerdichian, F. Koushanfar, M. Potkonjak and M.B. Srivastava, Coverage problems in wireless ad-hoc sensor networks, in: Proc. of the 20th IEEE INFOCOM (2001) pp. 13801387.

[22] D.P.Mehta, M.A.Lopez, L.Lin, Optimal Coverage Paths in Ad-hoc Sensor Networks , ICC,volume 1,May,2003.

[23] X.-Y. Li, P.-J. Wan and O. Frieder, Coverage in wireless Ad-hoc sensor networks, IEEE Trans. Comput. 52 (2003) 111.

[24] Hai Huang., Andrea W. Richa.and Michael Segal, Dynamic Coverage in Ad-Hoc Sensor Networks, Mobile Networks and Applications 10, 917, 2005.