Energy-aware Distributed Cluster Minimization Algorithm for Intrusion Detection In Wireless Ad Hoc Networks

Juan Huan, Shiguang Ju, Xiaoming Han

Department of Computer Science, Jiangsu University, Zhenjiang, Jiangsu 212013

Summary
In this paper, we propose a distributed hierarchical cluster algorithm for intrusion detection in ad hoc wireless networks. Based on an energy level metric for potential ad hoc hosts, it is used to determine the duration for which a particular node can support a network monitoring node. Besides using boundary-first criteria, our algorithm minimize the number of generated clusters to reduce the communication overhead in transmitting inter-cluster information and network maintenance messages. Simulation results verify that the number of generated clusters is reduced dramatically and lifecycle of clusters gets longer compared to the well known ID-based and connectivity-based algorithms. Otherwise our proposed scheme is more energy efficient while maintaining the same level of detection rate.

Key words:
ad hoc networks, intrusion detection, boundary node, critical node, energy-aware.

1. Introduction

Ad Hoc networks are temporary wireless networks without any infrastructure. Besides advantageous in military, rescue, mobile conferencing, etc, they are vulnerable to many kinds of attacks such as black hole, denial-of-service, selfishness, fabrication, etc [1]. Intrusion Detection approach is used to govern these non-secure wireless ad-hoc networks against malicious behavior (or attackers). Since these networks are lack in any sort of infrastructure, we need to monitor intrusions at all nodes in the network. However, this scheme imposes overhead since nodes in ad-hoc network are battery-constrained. Therefore, in order to reduce computation at all nodes, we must distribute the functionality in some sorts of centralized manner. The problem, however, in ad-hoc networks is lack of centralized audit points [2] for the intrusion detection.

Constructing wireless ad hoc networks into set of clusters provides an effective way to support network intrusion detection since it reduces the communication and the storage requirements significantly [3], and possible power conservation, interference reduction, frequency reusing, and capacity increasing. Based on this concept, an optimal configuration is selected in [4] to be the clusters of the wireless ad hoc networks; in such a way, as long as the changes of the network topology do not affect the clusters there is no need to reconstruct the network. However, the difficulty is that the problem to find an optimum configuration is shown to be NP-hard [5]. The general feasible alternative is to design an approximated heuristic algorithm to cluster the network into a sub minimum connected dominating set, for example, the connectivity-based [6] and ID-based [7] algorithms. In viewing these works, we find that they mainly focus on designing a clustering algorithm that can adapt to the network dynamic. However, when considering battery-constrained, we find that the number of generated clusters that contributes to end-to-end delay is another important design issue, because the communication overhead for exchanging the inter-cluster information is proportional to the number of the generated clusters [8]. We thus would like to present a clustering algorithm that can organize the wireless ad hoc network with the minimum number of generated cluster to support steady guaranteed effectiveness of coverage of any intrusion detection technique.

2. Modular IDS Architecture

Our IDS is built on a mobile agent framework. It is a non-monolithic system and employs several sensor agents thich perform certain functions, such as:

• Network monitoring: Only clusterheads will have sensor agents for network packet monitoring, since we are interested in preserving the total battery energy of mobile hosts.

• Host monitoring: Every node on the mobile ad hoc network will be monitored internally by a host-monitoring agent. This includes monitoring system-level and application-level activities.

• Decision-making: Every node will decide the intrusion threat level on a host-level basis. Clusterheads will collect intrusion information and make collective decisions about network level intrusions.

• Action: Every node will have an action module that is responsible for resolving intrusion situation on a host (such as locking out a node, killing a process, etc).

A hierarchy of agents has been devised in order to achieve the above goals. We will adapt the hierarchy for our purposes. There are three major agent classes as used in [8], categorized as monitoring, decision-making and action agents. Some are present on all mobile hosts, while others are distributed to only a select group of clusterheads, as discussed further. The monitoring agent class consists of packet, user, and system monitoring agents. The following diagram shows the hierarchy of agent classes.

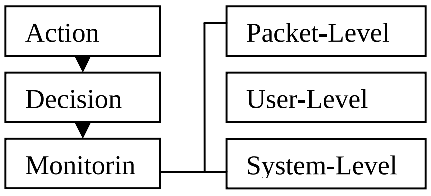

Fig.1 Critical agent hierarchy, depicting the multi-level decision making process for intrusion detection.

3. Minimization and Stability Problem of the Cluster

A. Network Model and Definitions

The entire network is modeled as a bi-directed connected graph and every node is assigned a unique ID, denoted by the numbers 1, 2…, N, where N is the number of nodes in the network. We assumed the transmission range for each node is fixed and identical and the signaling packets and data packets are exchanged over a common error-free wireless channel. We further have the following definitions. A node in a cluster is selected as *clusterhead* if it is the most weighted node (the weighting of a node is defined in (1)) among its one-hop neighbors. Node that is not clusterhead is called *ordinary* node. All nodes are considered to be identical, that is, no node with additional capabilities such that it tends to be easily elected as a clusterhead. A node is said to be with status *marked* or *unmarked* depends on if it is being clustered and nodes initially have status unmarked. The *degree* of a node is the number of one-hop neighbors that the node connects with. A node with degree 1 is said to be a *boundary* node and the only neighbor of a boundary node is said to be a *critical* node. A cluster is said to be an orphan cluster if it contains only one node and degree of this node is not 0.

B. Problem Formation

We formulated this clustering minimization and stability problem as follows.

Objective:

Minimize the total number of organized clusters

Extend the lifecycle of organized clusters

Constraints:

1. (dominance constraint) Every ordinary node has at least a clusterhead as neighbor.
2. (independence constraint) There is no overlap between clusters.

We solve this problem by transforming the objective into two sub-objectives: minimizing the number of orphan clusters and choosing the powerful nodes.

For the first sub-objective, an important property is found when we study the cluster structure generated by algorithms in [6]-[7]. More than 50% of orphan clusters are generated by boundary nodes and the percentage increases as the ratio of boundary node increases. The main reason for this is that the only neighbor of a boundary node, i.e. critical node, joins into a cluster constructed by one of its one-hop neighbor causing the boundary node becomes an orphan. Thus, if the clustering algorithm organizes the boundary node and its corresponding critical node into the same cluster, the number of the orphan cluster will be minimized. To conform to the first sub-objective, we select the critical node as the starting point of our clustering algorithm.

For the second sub-objective, choosing the powerful nodes is to extend the lifecycle of organized clusters. In ad hoc, the battery energy plays an important role. When a node uses out of battery energy, this node will vanish in the network. If this node becomes the clusterhead, it will cause cluster reconstruction. The more battery energy a clusterhead has, the longer lifecycle of a cluster is. Therefore, in order to lengthen cluster lifecycle, we should avoid electing a node with low battery energy as clusterhead. Set battery energy of the node v as E_v. Thus we define a threshold value $E_{threshold}$, when the node battery energy is lower than the value $E_{threshold}$, this node is called the dangerous node. A node has a privilege to be a clusterhead when its neighbor is a dangerous node. The number of neighbor dangerous nodes is the weight of this node. The more one node has the neighbor dangerous nodes, the larger the weight is assigned. According to the weight, assign the node corresponding privilege to be clusterhead.

To satisfy the above two sub-objectives, we define the weighting function of a node v, W_v, as

$$W_v = \begin{cases} 1+\theta & \text{if } v \text{ is a critical node and not dangerous} \\ D_v & \text{otherwise} \end{cases} \qquad (1)$$

where θ is the maximum of the neighbor dangerous nodes the considered network has and D_v is the number of the neighbor dangerous nodes node v has.

4. Distributed Clustering Algorithm

The operation of proposed distributed energy-aware boundary-first clustering algorithm is stated as follows:

1) For any unmarked critical node and dangerous node u, while the boundary node v, regard node v as a clusterhead and broadcasts an invite packet, **Invite (*v*)**, to critical node u.

2) For any unmarked node v whose W_v is locally maximum, regards itself as a clusterhead and broadcast an invite packet, **Invite (*v*)**, to all neighbors. (If there is more than one such node, the one with largest E_v prevails.If the result of E_v is same, the one with smallest *ID* prevails.).

3) For any unmarked node u receives an invite packet sent from its maximum weighting neighbor node v, regards itself as an ordinary node and sends a **Join(*u*,*v*)** packet to join the cluster constructed by node v. (If more than one such packet receives, selects the sender with the smallest degree.)

4) When all nodes around have been marked, then any unmarked node announces itself as clusterhead.

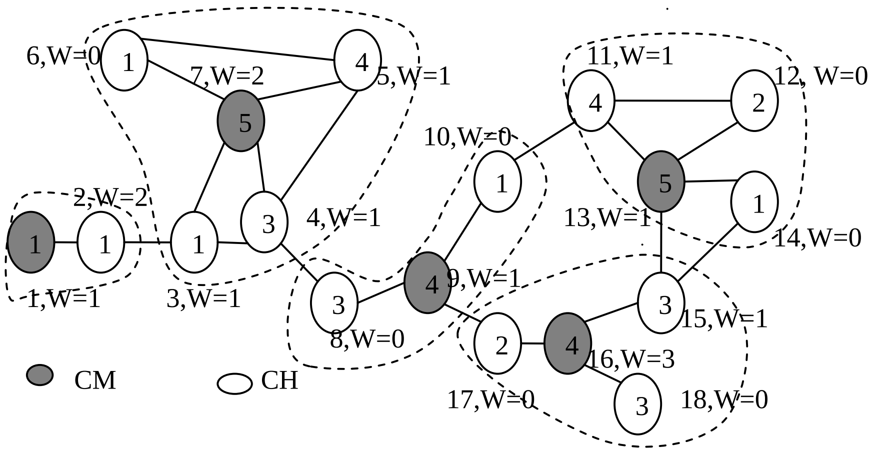

Fig.2 An ad hoc network with 18 nodes and the resulting cluster architecture. Nodes with gray background are clusterheads and the dashed line indicates the coverage of a cluster.

Example: Consider an ad hoc network with 18 nodes shown in Fig.2 in which $\theta = 2$. Two critical nodes, node 2 and 16 with weight $w(16)=1+\Delta=3$, critical node 2 is a dangerous node, so regard node 1 as a clusterhead. To do this is to avoid node 1 forming an orphan cluster, and at the same time C_1 has no too much burden. Construct clusters $C_1$and C_{16} respectively. Member of C_1 is node 2; while, members of C_{16} are nodes 15, 17 and 18. For the rest of unmarked nodes, node 7 with the highest weight (which is 2), organizes cluster C_7 with member nodes 3, 4, 5 and 6. Then, among node 11 and 13 with the same weight 1, node 13 is selected to be a clusterhead according to the criterion of the largest energy. Therefore, cluster C_{13} is generated with members11, 12 and 14. Again, among unmarked nodes 8, 9, and 10, node 9 prevails with the highest weight (which is 1), organizes cluster C_9 with member nodes 8 and 10. Thus far, all nodes are marked, and the algorithm terminates. The resulted cluster architecture is also shown in Fig.2.

To verify the proposed algorithm, we need to examine that the generated architecture satisfies the dominance and independence constraints and the algorithm terminates eventually.

Property 1: Every ordinary node has at least a clusterhead as neighbor. (*dominance constraint)*

Proof: After receiving an invite packet, an unmarked node turns out to be an ordinary node and regards the sender of the invite packet as clusterhead. If more than one invite packets are received, there will be multiple clusterheads as neighbors. In fact, the number of neighboring clusterhead is equal to the number of received invite packets.

Property 2: The generated clusters would not mutually overlap. (independence constraint)

Proof: According to the procedures, only unmarked node executes the algorithm and whenever an unmarked node is clustered, the algorithm updates the node status into marked. This means that a marked node is never being processed again, i.e., a node would not belong to more than one cluster. This proves no clusters mutually overlap.

Property 3: The aggregate number of nodes in each cluster is the total number of nodes in the network.

Proof: Let nodes in cluster C_{mj} are represented as a set V_{mj}. Assume the algorithm generates i clusters, C_{m1}, C_m, …C_{mj}, where m_k is the clusterhead ID of the corresponding cluster and $1 \leqslant k \leqslant i$. From Theorem 2, we know that each set V_{mk} with respect to cluster C_{mk} is disjointed, thus, by using the set operation, we have $\bigcup_k V_{m_k} = V$ where k=1, 2,…, i and $|V|=N$.

Property 4: The algorithm terminates in finite steps.

Proof: Since only nodes with status unmarked are considered in this algorithm and once a node is clustered, its status will be updated to marked, it is obvious that the algorithm will terminate when all unmarked nodes are processed.

Property 5: Each node broadcasts only one packet during executing the algorithm.

Proof: There are only two events incur packet transmission. The first event is {node with local maximum weighting}. The second event is {node receives an invite packet sent from maximum weighting neighbor}. Since these events are mutually exclusive, we can conclude that a node will only send one packet before the algorithm terminates.

5. Simulations and Results

We verify the distributed clustering algorithm by conducting extensive simulations. In the experiment, each result is the average of 10,000 simulations. In each simulation, we generate a connected ad hoc network by randomly placing N nodes$(40 \leqslant N \leqslant 80)$ in a 100×100 square. The transmission range of each node is 30m and is fixed.

In the experiment an appropriate threshold value $E_{threshold}$ should be defined. One unit of energy consumption is measured by the energy how much is needed when transmitting one information packet. The battery energy is set between 1000 to 10000 units. The source node and the destination node are produced stochastically each time, and each time 10 packages are transmitted. We only consider the energy consumption by transmission package, not considering other energy consumptions. The lifecycle is from the cluster construction to clusterhead's using out of battery energy.

As Fig.3 shows, no matter how many nodes there are, the lifecycle is longer then others when $E_{threshold}$ is set between 3000 to 8000 units. While $E_{threshold}$ between 3000 to 8000 units cluster lifecycle is similar. Obviously, that $E_{threshold}$ between 3000 to 8000 units can avoid choosing the dangerous node to be clusterhead, therefore the lifecycle is quite long. Fig.4 is the relation between $E_{threshold}$ and the number of clusterheads. From this, when $E_{threshold}$ is between 1000 to 2000 units, the number of clusterheads produced is more then others, on the other hand, the number of clusterheads is almost similar when E is between 3000 to 10000, it indicated that the E value between 3000 to 10000 units has small influences on the number of clusterheads. Therefore in following experiment, we define the $E_{threshold}$ value 3000 units.

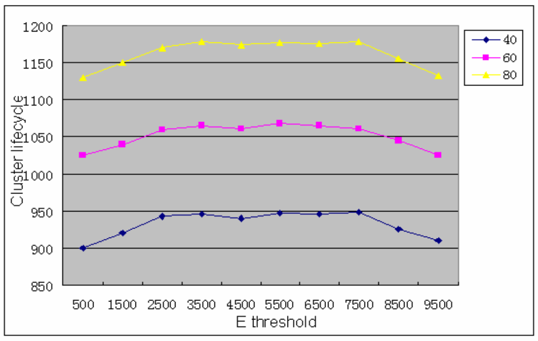

Fig.3 Find an appropriate threshold

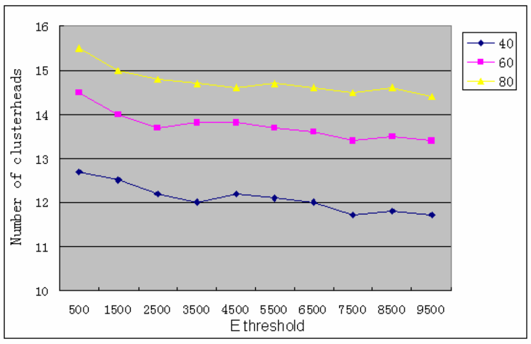

Fig.4 Relation between Ethreshold and number of clusterheads

Fig.5 compares the number of orphan clusters generated by the proposed algorithm (ECA) to the connectivity-based (HD) and ID-based algorithms (ID). It shows that the proposed algorithm reduces the number of generated orphan clusters dramatically and almost no orphan cluster is generated as the ratio of boundary node greater than 0.16. This result also shows that assigning the critical node with the highest weight in (1) is a good approach to reduce the number of orphan clusters.

Fig.6 compares the average lifecycle of organized clusters. It shows the averaged lifecycle of organized clusters by the proposed algorithm is far longer than the other two algorithms (HD and ID).That is because we choose nodes with enough energy to be clusterheads according to energy level metric. When we look into each of the 10,000 simulation results to check if there is any violation to this observation, similar conclusion is obtained.

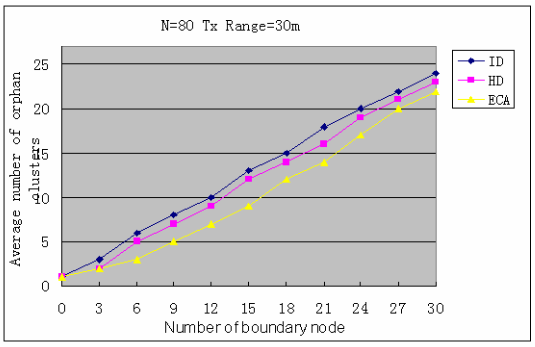

Fig.5 Comparison of average number of orphans clusters

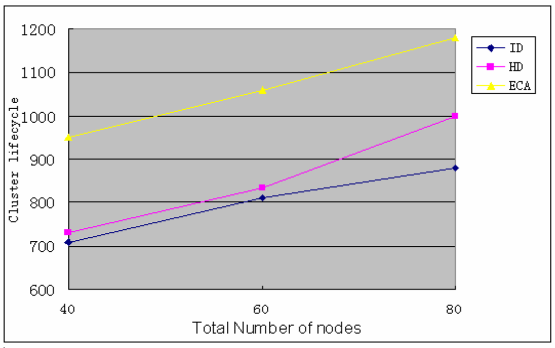

Fig.6 Comparison of average lifecycle of clusters

Fig.7. compares the detection time between the proposed algorithm (ECA) and the fully distributed scheme (Fully). We assume that there is one intruder sending a sequence of consecutive packets constituting an attack to the destination. These packets are sent in a flow consisting of normal packets. Further, we assume that the nodes, which are a part of the intrusion detection clusterheads, know this sequence of packets constituting the intrusion. The intrusion is considered being detected if this subsequence of attack packets pass through any of the clusterheads that implement the intrusion detection.

In the proposed algorithm, since the clusterheads can overhear all the traffic of the network, the time spent by ECA to detect an intruder should be equal to the fully distributed scheme. Only when the attacked node moves out of the scope that the clusterheads can overhear, the intruder may be detected later than fully distributed scheme. The figure shows that ECA can detect an attack almost as quickly as the fully distributed scheme. Even at the worse case, the proposed algorithm only costs several more millisecond than the fully distributed scheme.

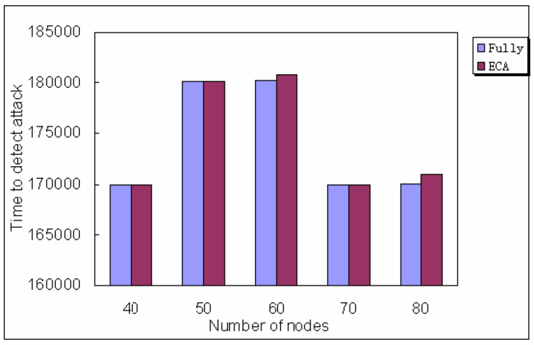

Fig.7 Comparison of detection time

The deficiency lies in our proposed algorithm implemented on routing protocol, and it brings a amount of additional control overhead. The part of control overhead is introduced by broadcasting message to maintain the cluster. What to do in the future is to control effectively this part of overhead within a small scope.

6. Conclusions

Intrusion detection is an indispensable second wall of defense especially in any high survivability network. Considering the limited computational and energy resources of mobile nodes, it is not efficient to make every mobile node always run IDS agent on itself. In this paper, we have proposed a distributed clustering algorithm for MANET. Its goal is to minimize the consumption of energy and at the same time maintains an acceptable level of monitoring. It divides the whole network into several clusters, selects the appropriate clusterheads that can overhear all the traffic for each cluster, and all the clusterheads run network IDS agents. Simulation results show that the clustering algorithm can implement the goals above efficiently.

Acknowledgments

It is the authors's honor to acknowledge that this work was supported by the National Natural Science Foundation of China under Grant No. 60573046 .

References

[1] Yi-an Huang, Wenke Lee: A Cooperative Intrusion Detection System for Ad Hoc Networks, ACM CCS, SASN Workshop, Oct 2003.
[2] Andrew B. Smith: An Examination of an Intrusion Detection Architecture for Wireless Ad Hoc Networks, National Colloquium for Information Systems Security Education, May 2001.
[3] J. Behrens and J. J. Garcia-Luna-Aceves, "Hierarchical routing using link vectors," in Proc. IEEE INFOCOM'98, San Francisco, CA.
[4] B. Das and V. Bhargavan, "Routing in ad-hoc networks using minimum connected dominating sets," IEEE ICC 97, 1997.
[5] C.S. Li, "Clustering in packet radio networks," IEEE ICC 85, pp. 283-287, 1985.
[6] Parekh A K. Selecting routers in ad-hoc wireless networks[C].Proceedings of the SBT/ IEEE International Telecommunications Symposium, August 1994.
[7] Gerla M, Tsai J T C. Multicluster mobile multimedia radio network [J]. Wireless Networks, 1995, 1 (3) : 255-265.
[8] Mario Joa-Ng and I-Tai Lou, "A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks," IEEE Journal on Selected Area of Communications, vol. 17, no. 8, pp. 1415-1425.
[9] ChatterjeeM,Das S K, Turgut D. WCA: a weighted clustering algorithm for mobile Ad Hoc network s [J]. Journal of Cluster computing, Special issue on Mobile Ad hoc Networking, 2002, 5: 193-204.