Computer Communications 31 (2008) 2972–2980

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

CASAN: Clustering algorithm for security in ad hoc networks

Mohamed Elhoucine Elhdhili *, Lamia Ben Azzouz, Farouk Kamoun

University of Manouba, ENSI CRISTAL Laboratory, Tunisia 2010, Tunisia

ARTICLE INFO

Article history:
Available online 11 April 2008

Keywords:
Self-organisation
Clustering
Ad hoc networks
Trust

ABSTRACT

Clustering in ad hoc networks is an organization method which consists in grouping the nodes into clusters (groups) managed by nodes called clusterheads. This technique has been used for different goals as routing efficiency, transmission management, information collection, etc. As far as we know, no existing clustering algorithms have taken into account the trust level of nodes for clusterheads election. In this paper, we propose a clustering algorithm for security in ad hoc networks that we called CASAN. CASAN aims to elect trustworthy, stable and high-energy clusterheads that can be used to offer security for application level. Simulations were conducted to evaluate CASAN in terms of clusters stability, load balancing and number of hops to clusterheads. Furthermore, we compare CASAN performances to an existing clustering algorithm called weighted clustering algorithm. Results show that it gives a convenient network division with stable clusters and mainly one hop members.

1. Introduction

Ad hoc networks are temporary wireless networks where mobile nodes rely on each other to keep the network connected without the help of any preexisting infrastructure or central administrator. These networks are generally formed in environments where it is difficult to find or settle down a network infrastructure [1].

In this type of networks, nodes must collaborate and organize themselves to offer both basic network services as routing and management services as security. Clustering might be a suitable technique for ad hoc networks to ensure efficiently these services. It consists in dividing the network into clusters managed by clusterheads.

An efficient clustering algorithm must adapt itself to frequently and unpredictable topology changes known in ad hoc networks. It must also generate stable clusters as much as possible to prohibit their updates which can lead to update other information as routing, security, addressing and management information. However, this technique can lead to the clusterheads congestion (processing, routing, etc.). Moreover, signalling messages used for executing the clustering algorithm and updating clusters can degrade the network performances.

In the literature, different works have proposed clustering algorithms for ad hoc networks [6–31]. These algorithms have different purposes as routing efficiency, backbone formation and network management. As far as we know, no existing clustering algorithms have taken into account the trust level of nodes while electing clusterheads. For this purpose, we propose a new reactive clustering algorithm for security in ad hoc networks called CASAN. This algorithm aims to elect trustworthy clusterheads that can be used to settle down many secured applications in ad hoc networks (public key infrastructures, secure instant messaging, etc.). Thus, the trust level of a node must be taken into consideration in the metric used for clusterhead election. Furthermore, nodes energy, mobility, distance to neighbors and connectivity metrics are also used for clusterheads election to make CASAN result in trustworthy, stable and high-energy clusterheads.

The rest of the paper is organized as follows: in Section 2, we present clustering in ad hoc networks. In Section 3, we describe the proposed clustering algorithm for security in ad hoc networks (CASAN). Section 4 presents simulations conducted to evaluate the performances of CASAN and compare it to an existing clustering algorithm called weighted clustering algorithm(WCA). Finally, in Section 5, we conclude the paper and outline our immediate future work.

2. Clustering in ad hoc networks

In the literature, several clustering algorithms for ad hoc networks have been proposed [6–31]. These algorithms can be classified based on their goals. Dominating-Set-based algorithms[6–9] try to reduce the number of nodes participating in relaying routing information and data packets. These algorithms allow to build hierarchical addresses and reduce the overhead costs imposed by routing. The size of routing tables is considerably minimized and routing could be accomplished via backbones.

* Corresponding author. Tel.: +21697582453; fax: +21671600449.
E-mail addresses: mohamed.elhdhili@cristal.rnu.tn (M.E. Elhdhili), lamia.benazzouz@ensi.rnu.tn (L. Ben Azzouz), frk.kamoun@planet.tn (F. Kamoun).

0140-3664/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2008.04.001

Other works have proposed mobility aware [10–14], low maintenance [15–18], energy efficient [19–21] and load balancing [22] clustering algorithms to divide the network into stable, balanced and long lifetime clusters that could be useful for upper layer protocols such as routing or management protocols. These algorithms use only one metric (node identification, connectivity, energy, mobility) for clusterheads election. Other clustering algorithms [23–26] use combined metrics with weighting factors for clusterheads election. The combined metric is a linear function of nodes characteristics such as mobility, energy, distance to neighbors and connectivity. Then, weighting factors can be adjusted according to the final application scenario. Recently, researchers have given much more importance to the application scenario of a clustered network such as network initialization [27,28], efficient data collection [29] where clusterheads can aggregate their members information before sending it and efficient monitoring [30,31] where clusterheads monitor traffic within their clusters and communicate with other clusterheads for cooperative traffic analysis, misbehavior or intrusion detection. Although these two last algorithms were designed for security purposes, they give all nodes the same chance to become clusterheads without taking into account neither their capacity nor their trust level or physical security.

As far as we know, despite the high number of existing clustering algorithms, no one have taken into account the trust level of nodes while electing clusterheads. CASAN is a reactive clustering algorithm for security in ad hoc networks. It will allow a network division over which we can implement many secured applications. For example, a distributed public key infrastructure can be implemented in the ad hoc network. Many works in the literature [3,2] have proposed distributed public key infrastructures for ad hoc networks where central authority services (certificate renewal, revocation, validation, etc.) were distributed over K servers (nodes). However, these works did not explain how to select or elect the K servers. CASAN would be suitable for this type of application where the trustworthy clusterheads will ensure distributed public key infrastructure (PKI) services.

While designing CASAN, we seek for a suitable division of the network (number of clusters generated with balanced load), clusters stability to prohibit management information updates and minimizing the overhead.

3. CASAN: clustering algorithm for security in ad hoc networks

In this section, we describe, in a first step, CASAN metrics. In a second step, we give its election protocol and update policy. The algorithm is executed for only one time (at the system bootstrapping). Then the updating policy is locally invoked after mobility or to attach new nodes joining the network.

3.1. CASAN metric components

To decide how much a node is suited for being a clusterhead to offer security services, we take into consideration the following characteristics:

- *The node trust level (T):* Clusterheads will ensure security services. Thus, we should elect trustworthy nodes as clusterheads. Nodes with a trust level less than a threshold *Trust_min* (defined by the network application) will not be accepted as candidate for being clusterheads even if they have other interesting characteristics as high energy or low mobility. Each node is assigned a static trust level by the application. However, this level can be decreased if nodes are misbehaving.
- *The node mobility (M):* We aim to have stable clusters. So, we should elect nodes with low relative mobility as clusterheads. To characterize the instantaneous nodal mobility, we will use a simple heuristic mechanism [4] where each node i estimates its relative mobility index M_i by implementing the following procedure:
  - Sends periodic hello messages (including its current mobility index M_i). To minimize the overhead, these hello messages will be also used for clustering.
  - Estimates the distance change $D(i,j)$ for each neighbor j from consecutive received hello messages (it is known that the received signal power density is inversely proportional to the square of the distance between the transmitter and the receiver).
  - Updates its mobility index M_i: after receiving all neighboring hello messages, node i computes its mobility index by summing up all distance changes with different weights so that larger received mobility indexes have smaller weights.
  - Sends the new estimated mobility index M_i in the next hello message.

  The mobility index is computed as given in Eq. 1.

$$M_i = \sum_{j=1}^{n} W_j D(i,j) = \sum_{j=1}^{n} \frac{\frac{1}{M_j}}{\sum_{k=1}^{n} \frac{1}{M_k}} D(i,j) \quad (1)$$

- *The distance to neighbors (D):* It is better to elect the node with the nearest members as a clusterhead. This might minimize node detachments and enhances clusters stability. D is computed as the cumulative mean square distance to neighbors divided by the total number of neighbors. The list of neighbors can be dynamically computed using hello messages while the distance to each neighbor can be estimated from the signal strength of a received hello message.
- *The node remaining energy (E):* We should elect nodes with high remaining battery power as clusterheads since they will offer security services to ordinary nodes. E can be easily retrieved from the node at any time.
- *The node connectivity degree (C):* We should not elect nodes with very high or very low connectivity as clusterheads. In the first case, they will be congested and their battery power will drop rapidly. In the second case, the clusters size will be very low and we will not take advantage of clustering.

3.2. Clusterheads election protocol

Each node broadcasts a hello message with $TTL = 1$ (Time To Live) including its identification and mobility index. Then, each node with a trust level less than a threshold *Trust_min* launches a timer *CTimer* and executes the non-trustworthy nodes procedure described in Section 3.2.2 while each node with a trust level greater than the threshold *Trust_min* launches a timer *HTimer* and executes the trustworthy nodes procedure described in the next section.

3.2.1. Trustworthy nodes procedure

Each node:

- Waits until its *HTimer* expires.
- Computes its connectivity degree C which is equal to the total number of distinct hello messages that it has received.
- Broadcasts its metric components ($T, M, E, C,$ and D) with $TTL = 1$.
- Uses received metric components of its neighbors to compute its weight as well as its neighbors weights according to the following steps: