Volume 2, No. 9, September 2011

Journal of Global Research in Computer Science

ISSN-2229-371X

RESEARCH PAPER

Available Online at www.jgrcs.info

EFFICIENT SECURE CLUSTERING PROTOCOL FOR MOBILE AD-HOC NETWORK

Shubha Mishra*[1] and Dr. Manish Shrivastava[2]
Department of Information Technology, Lakshmi Narain College of Technology, Bhopal, M.P., India
[1]mishrashubha1@gmail.com and [2]maneesh.shreevastava@yhaoo.co.in

Abstract: A Mobile ad hoc Network is an autonomous network comprised of free roaming nodes which communicate wireless by radio transmission. MANETs are already ubiquitous and their range of use will spread in the near future. In this thesis proposed an efficient dynamic clustering protocol for MANET. In our dynamic clustering protocol have five state interactions. These are un-clustered state, orphan state, election state, cluster node state, and cluster head state. Also, we develop key distribution method for the distribution of symmetric keys in MANETs. Our dynamic clustering protocol is designed to verify the protocol and have an estimate of the cost to gather the density information. We did the analysis from our dynamic clustering protocol different perspectives, in terms of time, clustering, and network packets. For evaluating in terms of time, time spent as part of cluster was measured. From clustering perspective, number of clusters, and number of nodes per cluster were measured. To estimate the network performance, number of protocol packets, and application packets transmitted were measured. And simulate the key distribution process to built authentication. Our results show the effectiveness and more efficient comparatively with previous work.

Keywords- MANET, Clustering, Election Techniques, Distribution Key management.

INTRODUCTION

In the recent years, wireless technology has enjoyed a tremendous rise in popularity and usage, thus opening new fields of applications in the domain of networking. One of the most important of these fields concerns mobile ad hoc networks (MANETs), where the participating nodes do not rely on any existing network infrastructure. A mobile ad hoc network is a collection of wireless nodes that can be rapidly deployed as a multi-hop packet radio network without the aid of any existing network infrastructure or centralized administration. Therefore, the interconnections between nodes are capable of changing on continual and arbitrary basis. Nodes within each other's radio range communicate directly via wireless links, while those that are further apart use other nodes as relays. Regardless of the attractive applications, the features of MANET introduce several challenges that must be studied carefully before a wide commercial deployment can be expected. Cluster algorithms have been widely used in MANETs to determine subsets of nodes for saving energy, enhancing routing protocols, finding efficient flooding, and broadcasting, or to generally build low-cost backbones. Clusters have also been applied in recent research on distributing TAs in ad hoc networks.

These cluster algorithms build one-hop clusters, i.e., the nodes in a cluster are in direct communication range with their CH. In each cluster, exactly one distinguished node, the CH, is responsible for establishing and organizing the cluster. Clusters are formed as geographically needed: If nodes cannot find existing clusters, they create clusters themselves, with existing clusters being merged and split on demand. Various clustering protocols were proposed for MANETs where mobility is slow and non-continuous. In this thesis we proposed a efficient dynamic clustering algorithm catering to these needs should give high importance to mobility and should have less overhead to achieve real time performance [2] and [4]. Symmetric key algorithms are computationally very efficient and are therefore of high interest for MANETs. However, as previously stated, the major challenge in using symmetric key cryptography in MANETs is the secure exchange and efficient storage of symmetric keys. Any data exchange over a wireless channel is initially unauthentic, making it almost impossible to exchange a key without a back-link or pre-configuration. In this thesis we also establish a key distribution method [13] and [15].

BACKGROUND

Mobile ad hoc networks (MANETs) represent complex distributed systems that comprise wireless mobile nodes that can freely and dynamically self-organize into arbitrary and temporary, “ad-hoc” network topologies. This helps people and devices to seamlessly internetwork in areas with no pre-existing communication infrastructure, e.g., disaster recovery environments. Clustering has evolved as an important research topic in MANETs as it improves the system performance by reducing the battery power (expenditure of energy), by decreasing the cluster size increasing the link stability of large MANETs. As MANETs have a limitation of battery power, cluster formation is expensive in terms of power depletion of nodes. This is due to the large number of messages passed during the process of cluster formation.

Clustering Algorithm in MANET:

There have been numerous proposals and surveys of clustering algorithms. Newly published approaches and others already reviewed will be given consideration. The survey presented concentrates on various classifications outlined previously.

Low-Maintenance Clustering: Clustered networks are chiefly criticized for the need of mobile nodes to have extra explicit message exchange between them in order to maintain cluster structure. When network topologies face recurrent changes, resulting in frequent cluster topology updates, the control overheads required for cluster maintenance face equivalent severe increases. The result of responsive clustering behavior may thereby consume a huge amount of network bandwidth, cause rapid energy drain (of mobile nodes) and (ironically and paradoxically) make ineffective any intended enhancement to network performance and scalability. Greater emphasis will be given to re-clustering due to its negative impact on issues regarding communication overhead, route invalidation and ripple effect. Re-affiliation, a lesser problem, refers to a non-cluster-head being reassigned after a link sever or compromise that seeks reestablishment within a different cluster-head that is within range without affecting the corresponding cluster-head(s). Accordingly, therefore, cluster-related control overhead can be reduced by limiting reaffiliation (usually requiring re-affiliation procedures) and re-clustering events. However, the proposed algorithm strives to actually eliminate this element completely by constructing, and maintaining, cluster architecture data traffic forwarding. The following protocols can be categorized under Low-Maintenance clustering approach:

The ***Lowest-Identifier (LID),*** or 'identifier-based clustering', was an original proposal of Baker and Ephremides (1981) and the Lowest-ID algorithm has proven one of the most favored clustering schemes cited in the old as well as recent hoc networks literature and has been a foundation for many undergraduate studies. This popular heuristic allocates each node a unique ID number and designates the node with the lowest ID as clusterhead. Thus, the IDs of clusterhead's neighbours will be higher than that of itself. However, the clusterhead is capable of delegating its responsibility to a node with the next minimum ID in its cluster. When a node lies directly between two or more clusterheads transmission lines it becomes a 'gateway' and is commonly used for routing between clusters. If a node lies between clusterheads and the clusters overlap the node may become part of a 'distributed gateway' if another node (from another cluster) within transmission range joins it as a pair to behave in this manner. Only gateway nodes (not regular cluster members) can listen to the different nodes of the overlapping clusters outside of which they lie.

The concept of distributed gateway (DG) is also used for inter-cluster communication only when the clusters do not overlie. The chief benefit of distributed gateways is assuming the delegated role of responsibility whereby it can maintain connectivity in situations where any clustering algorithm might fail to provide connectivity. Although system performance is better with LID than Highest-Degree (see next algorithm) in terms of throughput that is sacrificed by this algorithm in terms of its inherent bias towards nodes with smaller IDs possibly leading to the battery drainage of certain nodes without any attempt at a uniform balance of load across all the nodes.

The ***Highest-Degree***, or 'connectivity-based clustering', was an original proposal of Gerla and Parekh (1995) in which the degree of a node is calculated on the basis of its relative proximity to other nodes. Each node transmits its ID to others within its transmission range. A node x is considered to be a neighbour of another node y if x lies within the transmission range of y. The node having the greatest number of neighbours (i.e., most/highest degree of direct transmission links) is chosen as clusterhead and any tie is broken with the unique node IDs. The neighbours of a clusterhead become absorbed as members of that cluster (or specific neighbourhood) and cannot participate any further in the election process now they have a declared 'home'.

The neighbourliness process thus prevents any direct link between clusterheads; only one clusterhead will reside in each cluster. As the clusterhead is linked directly to each of its neighbours in the cluster, any two nodes in a cluster are never more than two-hops apart. Experiments have shown the system demonstrates a low clusterhead rate of change however; there is a low throughput under the Highest-Degree heuristic. Each cluster is typically assigned resources that are shared in turn between those cluster members [nodes]. Any increase to the number of nodes in a cluster causes an eventual drop in throughput with a general effect of gradual degradation in the system performance. Node reaffiliation rates are high due to node movement (for new tasks, migrating to clusters with sufficient resources and responding to events) often resulting in the highest-degree node's (the current clusterhead) failure at re-election because the loss of a neighbour can skew the dominance of a node's previous connections in this arrangement. The subsequent re-elections that occur because of the lack of a ceiling limit on node occupancy of a cluster can drain the system.

LCC (Least Cluster Change) — LCC (Chiang) is believed to be an adaptation that marries the best features of Lowest ID Clustering (LID) with Highest Connectivity Clustering (HC). Prior to the proposal of LCC, most protocols sporadically executed the clustering procedure and to satisfy a particular clusterhead attribute, occasionally reclustered. In HC, the clustering procedure is periodically carried out to confirm a clusterhead's "local highest node degree" attributes and on discovery of a higher degree member node, the current clusterhead under assessment must surrender its clusterhead role. As such, frequent re-clustering occurs when using this particular mechanism.

LCC uses two steps to take best advantage of the clustering algorithm: cluster formation that is established through LID to choose clusterheads from mobile nodes with the lowest neighbourhood ID and cluster maintenance. Re-clustering in this case is reduced as it is event-driven and summoned in only two scenarios:

When two clusterheads come into proximity range one surrenders its clusterhead role.

When a mobile node is unable contact any clusterhead, the cluster structure for the network is rebuilt according to LID.

LCC thus appreciably improves the stability of a cluster by abandoning the requirement for a clusterhead to always carry specified attributes in its local area. However, signified in the second reclustering scenario in LCC, a single node's movement could still call upon a complete cluster structure re-computation involving an unavoidable expensive communication overhead for clustering.

Energy-Efficient Clustering: Mobile nodes in a MANET dependent on battery power supply during operation pose challenges regarding energy limitation or conservation for optimal network performance. A MANET should make every effort to reduce any greedy energy consumption to prolong the lifespan of a network.

Clusterheads are essential to several administrative tasks and inter cluster communication over and above the regular function of an ordinary node and are subject therefore, to earlier 'death' because of excessive energy consumption. Any resultant lack of mobile nodes (each essential in its role) due to energy depletion can make the network liable to partition and potential communication interruption. The protocols that following can be categorised under an 'Energy-Efficient' clustering approach:

Power-aware connected dominant set is an energy-efficient clustering scheme that can decrease the size of a dominating set (DS) without any functional impairment. Unnecessary mobile nodes are identified and excluded from the dominating set and with the energy saving made from their exclusion, the higher energy-demanding clusterheads have more resources made available to them. Mobile nodes inside a DS bear extra tasks such as data packet relay and routing information updates and consume more battery energy than those outside a DS. The DS then is more power greedy than other sets so it is vital to find a means of reduction to its energy consumption. In this scheme energy level is ascribed to a node to determine its suitability as a clusterhead rather than ID or node degree as described in other schemes. A mobile node can be removed from the DS when it has less residual energy than dominating neighbours in its close neighbour set. However, this scheme is unable to balance the rate of energy consumption between dominating nodes (clusterheads) and non-dominating nodes (ordinary nodes) because it endeavors only to minimise the DS rather than to actually balance the energy consumption of each and every mobile node. Thus, despite achieving some level of energy consumption reduction by decreasing the number of nodes in the DS, much faster rates of energy depletion probably occur overall.
Load-Balancing Clustering: Load-balancing clustering algorithms are based on the belief that a cluster is best served by an optimum number of active mobile nodes, especially in a clusterhead-based MANET. An over large cluster will demand too much of the clusterheads, causing them to become the bottleneck of a MANET with subsequent system throughput reductions. An inadequately small cluster, however, will requires many more of the smaller cluster units to achieve performance capability but the increased number of clusters will inevitably increase the length of hierarchical routes with resultant longer end-to-end delay. This research satisfies the demands of load balancing by establishing calculated upper (Max value) and lower (Min value) limits on the number of mobile nodes that a cluster can deal with for optimal performance regarding stability and energy requirements.

The Max Value represents the upper limit to the amount of nodes a clusterhead can support simultaneously. Since mobile nodes have limited resources they are incapable of handling large numbers of nodes. This value is determined regarding the remainder of the clusterhead's resources. Should a cluster size exceed its predefined limit, reclustering procedures are invoked to make appropriate adjustment to the number of mobile nodes contained therein. The Min Value represents the lower limit to the amount of nodes contained in a given cluster before it becomes necessary to proceed to extension or merging mechanisms when a drop below this calculated lower limit would impair efficiency.

This is a global value that runs through the entire network. The Min Value can help avoid the complexities that result from having to manage great numbers of clusters that might otherwise occur without a load balancing strategy in place. The protocols that follow can be categorized under 'Load-Balancing' clustering approach:
DLBC (Degree-Load-Balancing Clustering): DLBC periodically reviews the clustering scheme to maintain the number of mobile nodes in each cluster around a designated system parameter, ED, that indicates the ideal for a clusterhead. Where the difference between ED and the number of mobile nodes that it currently serves exceeds some value, Max Delta, a clusterhead will be devalued and degrade to an ordinary member node. The endeavor of this mechanism is to make all clusterheads (where possible) serve the same and optimal number of member nodes.
Combined-Metrics-Based Clustering: Combined-metrics-based clustering considers a number of metrics for cluster configuration. It aims to elect the most suitable (rather than desirable) clusterhead in a local area by ignoring any bias of specific node attributes, permitting it to flexibly adjust the weighting factors for each metric in adaptation to a variety of scenarios. For example, in systems that are particularly concerned with battery energy, the associated weighting factor can be set at higher level. However, certain parameters may sometimes be unavailable or lack accuracy and understandably affect clustering performance.

Symmetric and public key cryptography in MANETs:

Symmetric and public (asymmetric) key cryptography provide a huge variety of protocols, e.g., for encryption, signatures and authentication, which are suitable for different applications due to their specific requirements.
Symmetric key cryptography: In symmetric key algorithms, two or more parties need to share a common key with a size of typically 128 bits or more. When a party wants to send a message that only the owners of this key can read, it encrypts the message either bit by bit using a *stream cipher*, or it encrypts the message in blocks of fixed size (e.g., 128 bits) using a *block cipher*. The primary advantage of symmetric key algorithms is their efficiency. The fact that hardware implementable bitwise XOR and AND operations are used for encryption and decryption, makes symmetric key algorithms suitable for devices with very limited computational capabilities. The major drawback however is the requirement for a shared key. This might either be critical due to the lack of a secure method to exchange such a key, or when the number of keys required exceeds devices' storage capabilities.

Symmetric key cryptography in Tactical MANETs Once shared keys have been exchanged and stored, symmetric key cryptography is the desired choice to encrypt/decrypt data in networks with limited computational capabilities. We assume small mobile-phone-sized devices for Tactical MANETs which are indeed constrained in their processing power. The major issue in using symmetric key algorithms for Tactical MANETs is the storage of the keys and the exchange of the keys in the absence of a trusted authority.

Public key cryptography: In public key cryptography, also known as asymmetric cryptography, the key used to encrypt a message differs from the key used to decrypt it. In public key cryptography, a user has a pair of cryptographic keys, a public key and a private key. The public key can be distributed freely, while the private key is kept secret. Messages are encrypted with the public key, and can consequently be encrypted by everyone. Only the entity in possession of the corresponding private key can decrypt the message. Private and public key pairs are mathematically related, but it is computationally impossible to derive the private key from the public key.

The mathematical techniques required to fulfill such properties include multiplications and modulo operations of numbers that are too big to be factorized. The time required for these multiplications and modulo operations on a state of the art laptop is in the range of milli-seconds or fractions of milli-seconds. An extensive use of public key cryptography in an algorithm can therefore quickly impose computation times of several seconds. The big advantage of public key cryptography compared to symmetric key cryptography is that shared keys are not required. However, the computationally expensive operations restrict the use of public key cryptography to devices with sufficient computational capabilities. Besides encryption and decryption, public key cryptography can be used for publicly verifiable digital signatures. If a node signs a message with its private key, each node knowing the public key can verify the authenticity of the signature.

This concept of public verifiability is a useful feature of public key cryptography that cannot be realized with symmetric key cryptography. Public key cryptography in tactical MANETs the fact that public key cryptography imposes a critical computational overhead to mobile-phone-sized devices, as used in Tactical MANETs, does not mean that it should be ignored. Firstly, there is no known way to realize algorithms such as publicly verifiable signatures with symmetric key cryptography. Secondly, combinations of public key and symmetric cryptography might facilitate more efficient algorithms than pure public key or symmetric key solutions alone. An example is the one-time generation of a shared key with public key cryptography, which is then used to run symmetric key algorithms. Thirdly, the capabilities of batteries and processors will continue to increase in future, allowing more complex computations on mobile devices. We therefore consider public key cryptography as a suitable, albeit carefully used, operation in Tactical MANETs, while symmetric key cryptography is the choice for frequently repeated and real-time computations [13] and [14].

The challenges in designing a fully distributed CA are similar to those for partly distributed authorities. Depending on the capabilities of the nodes and the topology of the network, either a partially distributed or a fully distributed approach can be the better choice. In a fully distributed CA, the chance to contact a required number of CA nodes is higher than in a partly distributed CA. Some of the CA nodes might be several hops away, imposing a higher communication overhead to obtain service than using a partially distributed CA. However, an online bootstrapping of a fully distributed CA imposes high communication costs and is therefore infeasible for larger networks. A fully distributed certification authority might therefore be favorable in small networks, and when an online bootstrapping of the CA is not required. This can be the case if i) the network can be pre-configured and a later re-establishment of the CA is not required, or ii) the network can be pre-configured and has a recurrent or permanent back-link to an infrastructure network [15] and [16].

In this thesis we proposed an efficient dynamic clustering protocol and also proposed distributed key distribution method for secure Mobile Ad-hoc Network.

PROPOSED TECHNIQUES

Proposed Dynamic Clustering Protocol:

Now, we describe our proposed Dynamic Clustering Protocol for MANET. In our clustering protocol is explained by using state interaction which is intuitive and easy to implement. The state interaction for our proposed clustering protocol is as follows:

a. Un-clustered State
b. Orphan State
c. Election State
d. Cluster Node State
e. Cluster Head State

Un-clustered State: When a node starts from parking, it would first change to the un-clustered state. It would remain in this state for un-clustered time period, t_{uc}. During this time it would listen to the wireless medium for GET_STATS packet. On receiving a GET_STATS packet from a cluster head, it would switch to cluster node state, otherwise at the end of tuc it would switch to orphan state.

Orphan State: As the name suggests, a node is in orphan state if there is no node within its communication range. There are exceptions to this rule, for example, if the last node in a road starts to lag behind the cluster head and goes out of communication range of cluster head, then it would change its state to orphan. Though in this case there would be node(s) in its vicinity, they would be in cluster node state. In orphan state, the node transmits GET_STATS packet at regular intervals of hello time period, t_h. The number of cluster nodes field in this GET_ STATS packet will be 0. This field is used to detect whether a GET_STATS packet is sent by a cluster head or an orphan. If an orphan receives a GET_STATS packet from another orphan, then it triggers both of them to enter election state. At any point, if an orphan receives a GET_ STATS packet from a cluster head, then it switches to cluster node state.

Algorithm for orphan state nodes-

Election State: A node enters this transient state to choose a cluster head and form clusters. When two or more nodes in orphan state come within the communication range of other(s), and receive GET_ STATS packet from one of them, they enter election state. At any point in election state, if a GET STATS packet is received from a cluster head, then the node withdraws from election process and changes its state to cluster node. There are three phases in the election state. The time period of each phase is election time period, t_e, which could be same as hello time period t_h or different. During the first phase, all the nodes involved in election exchange GET_STATS packet. The GET_STATS packet is used to calculate the number of neighbors. Each node broadcast MY_STATS packets in the beginning of second phase. On receiving MY_STATS packet from neighbors each node starts building the list of neighbors (nlist). While building the list of neighbors, MY_STATS packet received from nodes outside the toleration range are discarded.

This is done to prevent a node from choosing a neighbor with high probability of moving out of the communication range anytime, as cluster head. At the end of the second phase, a potential cluster head list (pcidlist) is formed by sorting the neighbor list by descending stability factor value. If there are two or more neighbors with same stability value, the node with lowest id wins. At the start of the third phase, if a node finds that it has the highest stability factor among its neighbors, then it would switch to cluster head state. Otherwise it would wait for time period te. If it received any NOT_CH packet (nchpkt) during this period, then it would delete that neighbor from the potential cluster head list. After deleting the neighbor, if a node discovers that it is the most stable among its neighbors, then it switches to cluster head state. At the end of third phase, the most stable neighbor would have been chosen as cluster head and the rest would switch to cluster node state.

Cluster Node State: Nodes in cluster node state report their mobility statistics to the cluster head at regular intervals of hello time period, t_h. Unlike most other clustering protocol, in Dynamic Cluster Protocol the cluster node decides which of the cluster heads it wants to report to based on the stability factor. Whenever a cluster node receives GET_STATS packet, it calculates stability with respect to the cluster head transmitting GET_STATS packet and stores it. It replies back with a NODE_ STATS packet. A NODE_ STATS packet is transmitted after a random small delay time (2 seconds) in order to prevent simultaneous transmission of NODE_STATS packets by other cluster nodes of the cluster, leading to packet collisions.

Not receiving GET_STATS packet for time period t_h could imply that it has moved out of communication range of cluster head. So it readily accepts any cluster head found after that, regardless of the stability. If a cluster node does not receive GET_STATS packet for deadline time period, td, it assumes that there is no cluster head in its communication range and switches to orphan state. The deadline time period could be n $*t_h$ where $n > 2$.

Consider scenarios where distance between the cluster heads is slightly more than twice the communication range. If a cluster node moves from one cluster to the other, it may miss GET STATS packet from both the cluster heads by fraction of a second. In that case it would not receive GET STATS packet for 2 $* t_h$ time period. Consequently, it is safe to have the value of n as greater than 2.

Cluster Head State: The cluster head is responsible for collecting mobility statistics from its cluster nodes. In order to achieve this, it sends a GET_STATS packet at regular interval of hello timer period, t_h. Cluster nodes which find it to be stable, reply with mobility statistics. The cluster head collects and stores the statistics at the end of t_h. In case none of cluster nodes reply, the cluster head assumes that it has lost all of them and changes its state to orphan.

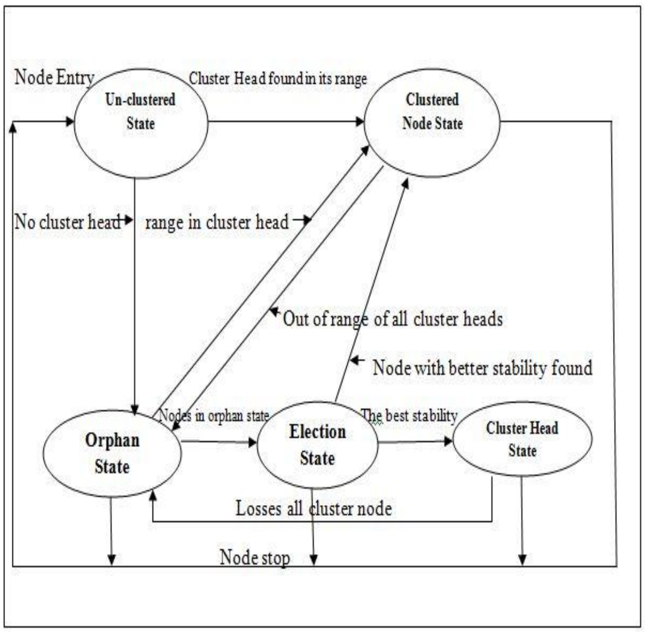

Figure: State Transition Diagram

KEY DISTRIBUTION METHOD

The statements in Authentication model can be described in more detail as follows:

Authenticity of public keys- $Aut_{A,X}$ denotes A's belief that a particular public key PX is authentic.

Trust- $Trust_{A,X,1}$ denotes A's belief that a particular entity X is trustworthy for issuing certificates. Similarly, her belief that X is trustworthy for issuing recommendations of level i − 1 is denoted by $Trust_{A,X,i}$.

Certificates- $Cert_{X,Y}$ denotes the fact that A holds a certificate for Y 's public key issued and signed by entity X.

Recommendations- $Rec_{X,Y,,i}$ denotes the fact, that A holds a recommendation of level i for entity Y issued and signed by entity X.

The inference rules that allow the derivations of statements from already known statements are defined by Maurer as follows:

$Aut_{A,X}$ $Trust_{A,X,1}$ $Cert_{X,Y}$ $\vdash$ $Aut_{A,Y}$ ………………(a)

$Aut_{A,X}$ $Trust_{A,X,i+1}$, $Rec_{X,Y,i}$ $\vdash$ $TrustA_{,Y,}i$………...(b)

The statements can thereby be divided into two different categories. The first category gives information about the characteristic of nodes and contains $Aut_{A,X}$ and $Cert_{X,Y}$. The second category holds information about the trustworthiness of nodes' characteristics and contains the statements $Trust_{A,X,I}$ and $Cert_{X,Y,i}$. Note that all these statements are deterministic, so trust in a node's characteristic means total trust. Due to the different levels i of trustworthiness statements, it is possible to infer statement chains of arbitrary length. In Maurer's model the general aim of building those chains is to infer new Authentication statements. Thus a chain of statements could be built as follows:

$Aut_{A,X}$, $Trust_{A,X,2}$,$Rec_{X,Y,1}$,$Cert_{X,Y}$,$Cert_{,B}$ $\vdash$ $Aut_{A,B}$,…………(c)

since:

Aut $_{A,X}$, Trust $_{A,X,2}$,, Rec $_{X,Y,1}$ $\vdash$ Trust $_{A,Y,1}$ ……………………(d)

Aut $_{A,X}$, Trust $_{A,X,1}$, Cert $_{X,Y}$ $\vdash$ Aut $_{A,Y}$ …………………………(e)

Aut $_{A,Y}$, Trust $_{A,Y}$,1, Cert $_{Y,B}$ $\vdash$ Aut $_{A,B}$ ………………………..(f)

The first simplification of yielding a reduction of complexity, especially in the computations of the probabilistic part, is to restrict the trustworthiness statements to level 1, while disallowing the use of second-hand evidence.

For the purpose of building a pure trust model, we also redefine statements as follows:

Trust- Trust $_{X,Y}$, denotes X's belief that a particular entity Y is a trustworthy TA member.

Distrust- Distrust $_{X,Y}$ denotes X's belief that a particular entity X is generally not a trustworthy TA member.

Authenticity of public keys- $Aut_{A,X}$ denotes A's belief that a particular public key PX is authentic.

For more deterministic model, it is necessary to define an additional parameter for distrust. Further statements such as Aut that might deliver information about a node's trustworthiness can also be defined. Limiting the length of trust chains to 1, inference rules are defined as follows:

Trust $_{A,X}$ Trust $_{X,Y}$ $\vdash$ Trust $_{A,Y}$, (I)

Trust $_{A,X}$, Distrust $_{X,Y}$ $\vdash$ Distrust $_{A,Y}$, (II)

Trust $_{A,X}$ Aut $_{X,Y}$ $\vdash$ Aut $_{A,Y}$. (III)

Rules (I) and (II) represent the forwarding of trust information over one hop, while (III) shows the mechanism to include additional statements in the model.

RESULTS ANALYSIS

In our experiment firstly shows the analysis with respect to time spent in each state. Secondly, analysis of variation of number of clusters and number of vehicles per cluster. Thirdly, analysis from the network perspective in which number of packets used in each protocol and their respective application were examined. The simulation time was 1000 seconds. For easy analysis, we divided the simulation time into three parts namely, Orphan time, Election time and Clustered time. Time spent by a vehicle in un-clustered state or orphan state is considered as Orphan time. We added time spent in un-clustered state to Orphan time as it is a short (5 seconds) and also during this time the vehicle is not associated with any cluster. The time period during which a vehicle is in election state is considered as Election time.

Time spent by a node in cluster node state or cluster head state represents Clustered time. In each scenario, ideal Orphan time was found by measuring the time period during which the node was disconnected from others. The ideal Clustered time is found by subtracting ideal Orphan time from the total simulation time. Explains the statistics from the networking perspective, in terms of packets transmitted.

The GET_STATS packet, NODE_STATS packet, MY_STATS packet and NOT_CH packet formed the protocol packets. The QUERY and CLUSTER_STATS packet formed the application packets. For all the configurations, ten vehicles sent query every 100 seconds making the total number of queries sent during simulation as 100. To measure the efficiency of DCP, percentage overhead of other protocols in terms of number of packets was computed. Comparing the overall results of our DCP and previous protocols we notice that DCP performs better in all the perspectives. Once the clustering protocol forms part of a network routing protocol or application, the nodes have to exchange their speed and number of neighbors information. In this way collection of speed and number of neighbors required for DCP will be easily available. Besides in wireless communication among vehicles even a slight decrease in the number of packets transmitted implies lesser packet collisions resulting in more reliability in communication. In the key management mechanism we first identify the parameters that need to be configured for our path authentication scheme.

We continue to determine the probabilities to (a) identify (verify) a path if the packet is sent over the expected path, (b) identify (back trace) the nodes on an unexpected path, and to (c) detect a adversary up to two nodes accuracy. The detection of selfish nodes is incorporated in the path identification, since the required information to detect selfish nodes is contained in the MACs. Based on these probabilities, we then propose a strategy to configure our probabilistic path authentication scheme. Finally, we present results for the probability to identify a path and to detect nodes, depending on both the length n of the tag and the number of packets R used for the analysis. Simulation driven experiments were used to validate our quantitative results and the optimality of our configuration settings.

CONCLUSION AND FUTURE WORKS

In this paper, we have proposed, simulated and evaluated a novel clustering protocol, Dynamic Cluster Protocol (DCP) for intelligent node. Summarizing the results, we notice that, DCP forms stable cluster indicated by higher average number of clusters and lower standard deviation.DCP has lower number of switches to clustered state corroborating that it forms stable cluster. The number of nodes per cluster is high in RSDCP showing that it forms large clusters.DCP has higher Clustered time which is a resultant of stable and large clusters. In order to achieve all these DCP has a slightly higher protocol overhead which is insignificant as the load on the application is lessened to a large extent. And key distribution method gives to built authentication system for MANET.

In future to improve the proposed clustering protocol, traffic modeling techniques could be investigated for developing better stability factor. If it is possible to get real world mobility traces, the clustering protocol could be validated against them to get more confidence in the protocol. To completely realize the distributed server architecture, the next step would be to design an algorithm which optimally chooses certain cluster heads to perform the role of distributed servers.

REFERENCES

[1]. S.Muthuramalingam and R.Rajaram, "A TRANSMISSION RANGE BASED CLUSTERING ALGORITHMFOR TOPOLOGY CONTROL MANET", International journal on applications of graph theory in wireless ad hoc networks and sensor networks (GRAPH-HOC) Vol.2, No.3, September 2010, pp 68-76.

[2]. Preetee K. Karmore , Smita M. Nirkhi, "Detecting Intrusion on AODV based Mobile Ad Hoc Networks by k-means Clustering method of Data Mining", International Journal of Computer Science and Information Technologies, Vol. 2 (4) , 2011, pp1774-1779.

[3]. Cynthia Jayapal and Sumathi Vembu, "Adaptive Service Discovery Protocol for Mobile Ad Hoc Networks", European Journal of Scientific Research ISSN 1450-216X Vol.49 No.1 (2011), pp.6-17.

[4]. I-Shyan Hwang and Wen-Hsin Pang, "Energy Efficient Clustering Technique for Multicast Routing Protocol in Wireless Ad Hoc Networks", International Journal of Computer Science and Network 74 Security, VOL.7 No.8, August 2007, pp 74-81.

[5]. Stefano Basagni, Michele Mastrogiovannit and Chiara Petrioli, "A Performance Comparison of Protocols for Clustering and Backbone Formation in Large Scale Ad Hoc Networks", 2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems, pp-7079.

[6]. Abdelouahid Derhab and Nadjib Badache. A Self-Stabilizing Leader Election Algorithm in Highly Dynamic Ad Hoc Mobile Networks. IEEE Transactions on Parallel Distributed Systems,19(7):2008, pp 926–939.

[7]. I. I. Er and W. K. Seah, .Mobility-based d-hop clustering algorithm for mobile ad hoc networks,. in IEEE Wireless Communications and Networking Conference, Atlanta, Georgia, USA, Mar. 2004.

[8]. Ratish Agarwal and Dr. Mahesh Motwani, "Survey of clustering algorithms for MANET", International Journal on Computer Science and Engineering Vol.1(2), 2009, pp 98-104.

[9]. T. Ohta, S. Inoue, and Y. Kakuda, "An Adaptive Multihop Clustering Scheme for Highly Mobile Ad Hoc Networks," in proceedings of 6th ISADS'03, Apr. 2003.

[10]. I. Er and W. Seah. "Mobility-based d-hop clustering algorithm for mobile ad hoc networks". IEEE Wireless Communications and Networking Conference Vol. 4., 2004, pp. 2359-2364.

[11]. P. Basu, N. Khan, and T. D. C. Little, "A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks," in proceedings of IEEE ICDCSW' 01, Apr. 2001, pp. 413–18.

[12]. F.D.Tolba, D. Magoni and P. Lorenz " Connectivity, energy & mobility driven Weighted clustering algorithm " in proceedings of IEEE GLOBECOM 2007.

[13]. CAO Zheng, YIN Pengpeng, and LU Zhengjun, "A Key Management Scheme for Multicast Based on Layer 2 Control", IEEE 2010, International Conference on Computer and Information Technology (CIT 2010), pp. 1512-1518.

[14]. Lin Yao, Bing Liu, Kai Yao, Guowei Wu, and Jia Wang, "An ECG-Based Signal Key Establishment Protocol in Body Area Network", IEEE 2010 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, pp. 233-238.

[15]. H. Wen, P.-H. Ho, C. Qi, and G. Gong, "Physical layer assisted authentication for distributed ad hoc wireless sensor networks", IEEE 2010, Special Issue on Multi-Agent & Distributed Information Security, pp. 390-396.

[16]. N H Ayachit, Santosh L Deshpande, and Kamakshi Prasad V, "Evolutionary Computing based secure key management Protocol", IEEE 2010.

SHORT BIODATA OF THE AUTHOR'S

Miss Shubha Mishra presently pursuing M.Tech of the department, Information technology at L.N.C.T College (A renowned academic institution of central India) M.P, India. The degree of Graduation secured in Information Technology in 2009 from SSSIST College, Bhopal, India.

Research Interest includes Network Security, Data Mining, and Artificial Intelligence. E-mail: mishrashubha1@gmail.com.

Dr. Manish Shrivastava worked on TCP/IP Network, Optical Network, Network Security and Management since 2003, from research to commercial development and is currently a professor at LNCT (A renowned academic institution of central India). He did his graduation in

Computer Technology in 1993 from UIT, RGTU (Formally known as Govt. Engineering College) Bhopal, India. After graduation he started his carrier on the recommendation of TCS with an organization, which works under their consultancy. He switched in academic in 1997. He did his PG and Doctorate from MANIT, Bhopal. In all he has 19 years academic, research and industry experience. E-mail: maneesh.shreevastava@yhaoo.co.in.