Journal of Theoretical and Applied Information Technology, Islamabad PAKISTAN
31st May 2010. Vol.15. No.2.

www.jatit.org

SPANNING TREE BASED VARIABLE LENGTH DYNAMIC ADDRESS AUTOCONFIGURATION IN MOBILE AD HOC WIRELESS NETWORKS

[1]K. SAHADEVAIAH, [2]O.B.V. RAMANAIAH

[1]Assistant Professor, Department of Computer Science Engineering, University College of Engineering, Jawaharlal Nehru Technological University: KAKINADA, Andhra Pradesh, INDIA-533 003.

[2]Professor & Head, Department of Computer Science Engineering, University College of Engineering, Jawaharlal Nehru Technological University: HYDERABAD, Andhra Pradesh, INDIA-505 327.

ABSTRACT

A Mobile Ad hoc NETwork (MANET) is an infrastructure-less, spontaneous, and arbitrary multi-hop wireless network, consisting of group of mobile nodes. The topology of the network changes randomly due to unpredictable mobility of nodes. In order to allow truly spontaneous and infrastructure-less networking, a protocol for dynamic allocation of unique addresses is needed, The pre-configuration of addresses is not a possibility in MANET and every node must configure its network interface with a unique address in order to communicate with other nodes so that the packets can be relayed hop by hop and delivered ultimately to the destination. Allocating addresses in mobile ad hoc network is a challenging task. In recent years, various address autoconfiguration protocols have been proposed in the literature to solve this problem. However, address autoconfiguration in mobile ad hoc networks is still an unresolved issue. The main task of an address autoconfiguration protocol is to manage the resource address space. When a node leaves the network, the corresponding address must eventually be deallocated to prevent exhaustion of the address space. This paper proposes a spanning tree based variable length dynamic address autoconfiguration, a novel method to solve the problem of address autoconfiguration of mobile ad hoc networks. Network partitioning and merging as well as duplicate address detection are well supported by our approach.

Keywords: *Mobile Ad hoc Network, Variable Length Addressing,, Autoconfiguration,, Spanning Tree, Duplicate Address*

1. INTRODUCTION

The term “ad hoc” tends to imply “can take different forms” and “can be mobile, stand alone, or networked”[1]. Ad hoc implies that the network is formed in a spontaneous manner to meet an immediate demand and specific goal. Ad hoc networks have the ability to form “on the fly” and dynamically handle the joining or leaving of nodes in the network. Mobile nodes are autonomous units that are capable of roaming independently. Typical mobile ad hoc wireless nodes are Laptops, PDAs, Pocket PCs, Cellular Phones, Internet Mobil Phones, Palmtops or any other mobile wireless devices. Mobile ad hoc wireless devices are typically lightweight and battery operated.

A mobile ad hoc network (MANET) is an adaptive, self-configurable, self-organizing, infrastructure-less multi-hop wireless network with unpredictable dynamic topologies [2]. By adaptive, self-configurable and self-organizing, we mean that an ad hoc network can be formed, merged together or partitioned into separated networks on the fly depending on the networking needs. This means that a formed network can be deformed on the fly without the need for any system administration. By infrastructure-less, we mean that an ad hoc network can be promptly deployed without relying on any existing infrastructure such as base stations for wireless cellular networks. By multi-hop wireless, we mean that in an ad hoc network the routes between end users may consists of multi-hop wireless links. In addition, each node in a mobile ad hoc network is capable of moving independently

and forwarding packets to other nodes. Before proper routing can be possible, each node needs to be configured to a unique address. In the conventional network, such as Internet, the network address of a device can be preconfigured statically or assigned through the dynamic protocol, such as Dynamic Host Configuration Protocol (DHCP).

The major list of the objectives of an optimal mobile ad hoc network address autoconfiguration protocol is:
Dynamic Address Configuration: Nodes should be able to dynamically obtain IP addresses without manual or static configuration.
Uniqueness: Nodes should obtain unique addresses for correct routing and communication.
Robustness: The addressing protocol should adapt to the dynamics of the network, including partitions and merges.
Scalability: The protocol should avoid significant performance degradation as the size of the network increases.

The main task of an address autoconfiguration protocol is to manage the resource address space. It must be able to select, allocate, and assign a unique network address to an autoconfigured node. When a node leaves the network, the corresponding address must eventually be deallocated to prevent exhaustion of the address space. A major challenge is network partitioning and merging. An autoconfiguration protocol can at most guarantee unique addresses within a single network partition. If two partitions merge, address conflicts may occur. The resolution of a conflict requires at least one node to acquire a new address.

The autoconfiguration protocols proposed for mobile ad hoc networks can be classified as *stateful*, *stateless* and *hybrid* approaches [3]. A *stateful approach* assumes the existence of a central entity to keep an address allocation table for whole network and assigns unique address for unconfigured node. However, when subnet merging occurs, it is still a great challenge to synchronize multiple central entities. Furthermore, most of these protocols rely on either periodic or reliable flooding, which usually consumes a considerable amount of bandwidth. *Stateless approaches* do not need a central entity to maintain an address allocation table. Instead, each node selects an address by itself and verifies its uniqueness with the so-called *duplicate address detection* (DAD) procedure. If duplication is detected, at least one of the nodes with duplicate addresses must change its address. *Hybrid approaches* combine both stateful and stateless approaches by maintaining an allocation table and performing a DAD, which can increase the robustness, but may result in higher complexity and higher protocol overhead.

VARIABLE ADDRESSING SCHEME:

In mobile ad hoc networks, pre-configuration is not always possible and has some drawbacks. So, an autoconfiguration protocol is required to provide dynamic allocation of node's address. Most current ad hoc routing architectures use flat addressing which uses fixed length for addressing each node. The overhead in control packets of source routing is reduced highly with variable addressing scheme. The length of this routing address is variable, but remains same for all the nodes in the network at one point of time. Our scheme allocates addresses in such a way that nearby nodes have same prefixes, thereby helping the routing protocols to make use of the location information of nodes from their addresses.

2. RELATED WORKS

The major challenge of address autoconfiguration in mobile ad hoc networks is the provision of both address uniqueness and efficiency in terms of communication overhead and configuration latency. Current approaches, in the following literature, are not satisfactory with respect to these requirements. In recent years, various address autoconfiguration protocols have been proposed in the literature to solve the problem of allocating addresses in mobile ad hoc networks. Most stateful approaches are usually conflict-free as they use allocation table. An address autoconfiguration protocol utilizing a centralized allocation table has been proposed in [4]. The protocols, MANETconf [5], Boleng's protocol [6], and the Prophet Allocation protocol [7], utilizes a distributed common allocation table. The protocol, proposed in [8], utilizes multiple disjoint allocation tables. The ad hoc address autoconfiguration (AAAC) [9] protocol, also uses multiple disjoint allocation tables but relies on broadcast messages for implementing address reclamation. Accordingly, the proposed scheme also falls into the stateful approaches, since it relies on the spanning tree to make conflict-free address allocations.

Instead of maintaining an address allocation

31st May 2010. Vol.15. No.2.

table, the protocols using stateless approaches usually need the DAD algorithm to ensure the uniqueness and conflict-free of the allocated IP addresses. Query-based DAD [10] performs DAD by querying all nodes in the network, which may result in high communication overhead and small scalability. In weak DAD (WDAD) [11], routing protocols are customized, by attaching an additional key to each address, to prevent the packets from being transmitted to the conflict addresses. The additional key can enhance the ability to detect address conflicts but increase the additional overhead over routing protocols. Passive DAD (PDAD) [12] observes incoming route protocol packets to derive the hints about address conflicts. Since the behavior of detecting address conflicts is passive, PDAD consumes almost no communication overhead but the latency to resolve any possible address conflicts is unbound, which is unacceptable in many practical environments.

The two hybrid protocols: Hybrid Centralized Query-based Autoconfiguration (HCQA) [13] and Passive Autoconfiguration for Mobile Ad Hoc Networks (PACMAN) [14] combines elements of both stateful and stateless approaches, and thus can improve the algorithm stability. Nonetheless, they may result in higher protocol complexity and overhead [3].

3. PROPOSED METHOD

We consider an autonomous mobile ad hoc network working on its own. The network is formed by a group of nodes coming together. The nodes can join and leave the network at any time and are free to move around. Hence, the size and topology of the network is dynamic and unpredictable in nature. Every node must configure its network interface with a unique address in order to communicate with other nodes.

3.1. KEY ASSUMPTIONS

(i) The proposed approach assumes that link availability always can last a piece of time, though node mobility cannot be avoided.
(ii) The spanning tree is constructed on demand in a completely distributed fashion when network topologies change. Spanning tree construction indeed a traversal algorithm involving almost all nodes in a subnet, which is a logical grouping of connected nodes.
(iii) The network environment is homogeneous, i.e., all the nodes have same power capabilities.

3.2. SPANNING TREE CONSTRUCTION

A mobile ad hoc network may involve an amount of nodes that self-organize into partitions, also called subnets, by connectedness. The spanning tree is constructed on demand in a completely distributed fashion [15]. Constructing the spanning tree is indeed a traversal algorithm involving almost all nodes in a subnet, which is a logical grouping of connected nodes.

In unicast communication, before obtaining a unique address, a node uses the link-local address or the MAC address to communicate with its neighbors. Each node maintains a list to maintain its neighbor's link-state. If a new neighbor is discovered, a decision is made regarding whether or not an adjacency should be formed with the neighbor. The decision may trigger subnet merging in an appropriate time.

One node, namely the *requester*, marks itself and broadcasts a query message to all its neighboring nodes. The source field of the query message is set to be the link-local address of the requester. The receiver records the source address as its parent, marks itself and broadcasts the query message again if it has not been marked. Otherwise, if a receiver has been marked, it should ignore the query message. When all nodes are marked, propagating the query request messages leads to a spanning tree. Each node is able to trace back to the requester along the spanning tree in the reverse direction. The requester is the root of the spanning tree, which has no parent. The leaf nodes are those that have no children nodes. Therefore, if the requester needs to obtain answers of all receivers, every receiver can post a reply message to the requester.

Given a subnet G' = {V', E'}, a round accessing is defined as a process for a requester to receive all replies after issuing a query. The spanning tree may be different, regarding different orders of transferring the query messages. Since, the spanning tree is constructed on demand in a completely distributed fashion, requester competition occurs when more than one node try to build the spanning tree by issuing the query

31st May 2010. Vol.15. No.2.

independently. Only one spanning tree is needed. Privilege identification (PID) is defined to distinguish multiple requesters. PID is a 2-tuple *(D, R)* where the first part *D* can be specified and the latter part *R* is a random number that is initialized by a random-number generator. Two PIDs can be compared in the pseudo code below:

```
int compare (PID pid1, PID pid2)
{
        test = pid1.D – pid2.D;
        if (test == 0) test = pid1.R – pid2.R;
        return test;
}
```

Given two PIDs, *p*1 = *(D*1, *R1)* and *p*2 = *(D*2, *R2)*. We define that *p*1 is higher than *p*2, if and only if compare (p1, p2) > 0; p1 is lower than p2, if and only if compare (p1, p2) < 0; and p1 equals to p2, if and only if compare (p1, p2) = 0. In practice, we should choose an appropriate random number generator to reduce the situation, whose probability is evaluated in the following theorem.

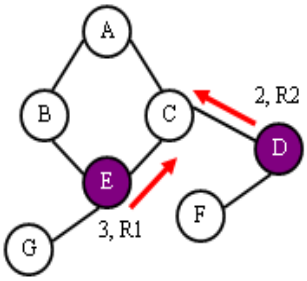

Fig. 1 An example where there are two requesters in the subnet.

Theorem of Coincidence: *Given a subnet, the latter part of PID is represented by B (N = 2B) binary bits and its value is generated using random assignments. If M (> 1) requesters exist, the probability of multiple requesters having the same highest PID is*

$$p \le 1 - \frac{M}{N^M} \sum_{i=1}^{N-1} i^{M-1}$$

Figure 2 shows an example with two requesters issuing queries concurrently. *(3, R1)*, is higher than (2, *R2)*.Whenever *the* query request with *(3, R1)* or (2, *R2)* reaches node c earlier, node c will hold the PID *(3, R1) at last*. The requester with *(3, R1)* will complete its query process as if the requester with *(2, R2)* does not exist. The case with more requesters in the subnet is similar.

3.3. ADDRESS ALLOCATION

The objective of the address allocation is to assign unique address for every node in a subnet. After a node switches into the ad hoc mode, a new process of address allocation is triggered. In our approach, the process of address allocation includes two steps. The first step is to investigate the size of subnet to obtain an order of nodes. The next step is to distribute appropriate address resources to each node along the spanning tree.

STEP1: SUBNET SURVEY

The aim of the subnet survey is to obtain an order of nodes, preparing for the further address allocation. Any node that has not been configured can request to become a requester, but the requester competition enables only one requester to exist at one time. Each requester has a PID to differ itself from others. Propagating the query message sent by the requester leads to a spanning tree. The requester starts a timer and waits expiration of SURVEY_TIMER, which should be preconfigured to exceed two times of the maximum end-to-end latency. After the timeout of SURVEY_ TIMER, the requester presumes that the survey is successful if it still is a requester. Each node has a list variable *L* that holds all the query results that its children return. Each element of *L* is a 2-tuple (*fromID, count*) which indicates that the query result of the node with the link-local address *fromID* is *count*. A node uses the following equation to compute the recent query result.

The recent query result of a node is:

$$k(a) = \sum_{b \in L} b.\text{count} + 1.$$

Each node initiates its *k(.)* to be 1 and uses a reply message to answer each query message. The reply message contains the query result of the sender.

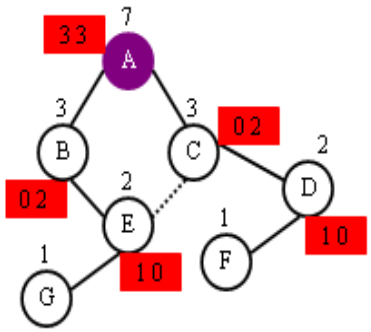

Fig. 2 An example of survey process

31st May 2010. Vol.15. No.2.

Figure 3 shows an example where node A is a requester. $k(A) = k(B) = k(C) = k(D) = k(E) = k(F) = k(G) = 1$ initially. Notice that the query message flows from the root node A to all other nodes and each node will return at least one reply message. Therefore, $k(F)$, and $k(G)$ always are 1, as they can not receive any replies. $k(C)$ becomes 3 after D reply messages. Likewise, $k(A)$ is updated to be 7 after node A receives the reply messages from all the other nodes.

Theorem of subnet survey: *Given a subnet $G' = \{V', E'\}$, if $s \in V'$ is a requester, k(s) is equal to the number of nodes in V'.*

STEP2: ADDRESS ALLOCATION

The task of the address allocation process is to allocate variable length address to each node along the spanning tree. After a survey process completes successfully, the requester starts to allocate addresses. The address allocation uses the same spanning tree as the survey process.

The first node that comes in takes an address "0" and registers a free address "1" as available address. The second node that attaches to the network gets an address "1" looking at the routing updates from its neighbor. Here, we call the first node as the *creator* and the second node as *follower*. "***Creator*** is any node that forces a bit increase in the address. ***Follower*** is any node that gets an address from the free addresses. One may note that a node may be a creator or follower, but not both."

The third node will get to know that there are no free addresses available. So, it creates the address space by adding one more bit as a prefix for addressing. Now the previous two nodes get assigned "00" and "01" while the new node gets an address from the available free addresses ("10" or "11"). In our algorithm, the lowest address is assigned and the remaining addresses are maintained as free addresses. If there are no free addresses, the address space is expanded by adding a prefix "0" to the already existing addresses and the new node gets an address of the regular expression [10+] depending on the requirement of the address space. It is significant to note, from figure 4, that all the nodes in the network have the same address length at any point of time. This will lead to consistency in addressing.

EXAMPLE:

Assume node 'A' be the first node to initiate the network. Then 'A' becomes the creator. It takes the address '0' and reserves '1' in the free address space. Let 'B' joins the network then it listens to network and takes '1', which is reserved under the free address space, hence is follower. Similarly, node C becomes a creator for the second bit. This concept is carried by further nodes.

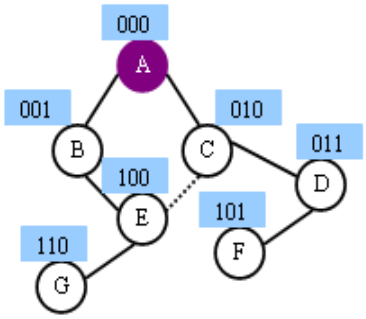

Fig. 3 Variable length address allocation

The following table demonstrates the way variable length addressing works. The columns represent the addresses of all nodes in the network as each node enters the network in sequence as indicated by the rows.

Table 1 Variable length addressing

Node	1	2	3	4	5	6	7	8	9
1	0	·	·	·	·	·	·	·	·
2	0	1	·	·	·	·	·	·	·
3	00	01	10	·	·	·	·	·	·
4	00	01	10	11	·	·	·	·	·
5	000	001	010	011	100	·	·	·	·
6	000	001	010	011	100	101	·	·	·
7	000	001	010	011	100	101	110	·	·
8	000	001	010	011	100	101	110	111	·
9	0000	0001	0010	0011	0100	0101	0110	0111	1000

The following table represents the free address space available with each of the nodes entered into the network.

31st May 2010. Vol.15. No.2.

Table 2 Free addresses maintenance

Node Number	Free address Space	Number of free address(es)	Number bits involved in addressing
1	[1]	1	1
2	-	-	1
3	[11]	1	2
4	-	-	2
5	[101-111]	3	3
6	[110-111]	2	3
7	[111]	1	3
8	-	-	3
9	[1001-1111]	7	4

VARIABLE LENGTH ADDRESSING ALGORITHM:

The procedural steps of the proposed algorithm, variable length addressing, for dynamic address allocation is as follows:

Module: A New Node Joins the Network.
Input : '**n**' be the new node joining the network, N.
Output: Uniquely addressed nodes in the network, N with varying length addresses, if needed.
Step1: if (**n** is the first node)
{
Assign address "0" to node itself. Register "1" as available address. Here, node '**n**' is called as *creator* since it forces an extra bit to address existing nodes.
}
Step2: else
{
Node '**n**' listens to the periodic routing updates of its neighboring nodes.
If (node '**n**' identify an unoccupied address (s))
{
Assign the least unoccupied address(s) identified to node '**n**'. Here, node '**n**' is called as *follower* since it takes the freely available address.
}
else // There are no free addresses
{
Expand the address space by adding a prefix "0" to the already existing addresses and the new node gets an address of the regular expression, [10+] and registers all higher addresses as available free addresses. It is significant to note that all the nodes in the network have the same address length at any point of time. This will lead to consistency in addressing.
}
}

Module: A New Node Leaves the Network.
Input : '**n**' be the leaving node from the network, N.
Output: Uniquely addressed nodes in the network, N with varying length addresses, if needed.
Step1: if (**n** is the follower node)
{
Relinquish the address allocated to node. Add this address to the available free addresses.
}
Step2: else // '**n**' is the creator node leaving
{
Diminish the address space by deleting a prefix "0" from each node address. It is significant to note that all the nodes in the network have the same address length at any point of time. This will lead to consistency in addressing.
}

4. NETWORK PARTITIONING, MERGING AND DUPLICATE ADDRESS DETECTION

A major challenging unsolved issue in mobile ad hoc networks is network partitioning and merging. Both node mobility and link state changes may lead to network partitioning or network merging. An ad hoc network could be divided in to two or more disconnected networks during its lifetime. When two different partitions merge, address conflicts may occur, because there is a possibility that two or more nodes have the same IP address. Such duplicate addresses should be detected and resolved. The resolution of a conflict requires at least one node to acquire a new address. Since, every address change may break transport layer connections, unnecessary address changes should be avoided.

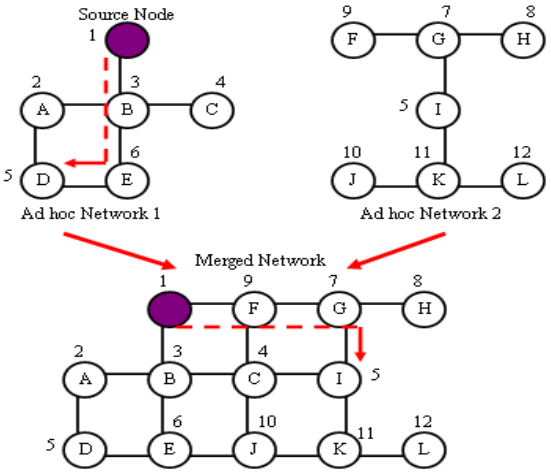

Fig. 4 Example of duplicate addresses and erroneous routing, where nodes D and I have the same address 5. Erroneous routing occurs when the two networks merge.

31st May 2010. Vol.15. No.2.

An important requirement of address autoconfiguration is to provide duplicate-free address assignment and to be able to detect duplicate addresses if they occur. Figure 5 shows a simple scenario of the existence of duplicate addresses, where two ad hoc networks are about to merge. A source node in ad hoc network 1 sends packets to node D, which owns address 5. Because this address is unique in the respective partition, no routing problem exists. If both partitions merge, address 5 is no longer unique in the network, and packets destined for node D may be misrouted to node I. Thus, erroneous routing prevents the source from communicating with the correct destination.

THE PROCESS OF SUBNET MERGING:

To detect subnet merging, there are two scenarios to be considered. The first scenario is that one node concludes that subnet merging occurs when detecting another node having the different network identification (NID) from it's. Since, after a successful survey process, the requester creates a NID and distributes it to every node along the allocation process. The second scenario is that one node detects that another node has both the same NID and the same address as itself. If more subnets need merging, every two carry out merging, until finish all. Merging of two networks may raise address conflict. When two subnets merges, the number of addresses needed to change should be kept as few as possible. In our algorithm, the process of subnet merging only changes the addresses in the subnet with fewer nodes.

THE PROCESS OF SUBNET PARTITIONING:

Two situations may result in subnet partitioning. In the first situation, one or more configured nodes go out of others' transmission range. The simplest solution is to mark changes and ignore subnet partitioning if a node leaves the subnet. If there are at least two requesters with the same highest PID, requester competition results in the second situation. The address space may go on increasing after subnet merges and partitions alternately many times. Our solution is straightforward. Since the survey process in a subnet can obtain the number of nodes in the subnet, we can insert a checking function before address updating. The checking function compares the number of nodes with the first part of the NID. If the difference exceeds a certain limit, all the addresses in the subnet is cleaned up and allocated again through a process of address allocation.

THE PROCESS OF DUPLICATE ADDRESS DETECTION:

We have taken care to have unique Node-ID and unique Network-ID using the concept of random number assignment associated with each ID. We demonstrated this using the theorem of coincidence. It is rare to have duplicate Node-ID and Network-ID, we have shown that this probability is trivial and can be ignored. We have taken care to merge two networks even with same Network-ID and duplicate node-ID's can be detected by observing the network's periodic routing updates.

5. EXPERIMENTAL INVESTIGATIONS

The proposed algorithm was coded in Java and run on a Pentium IV machine.

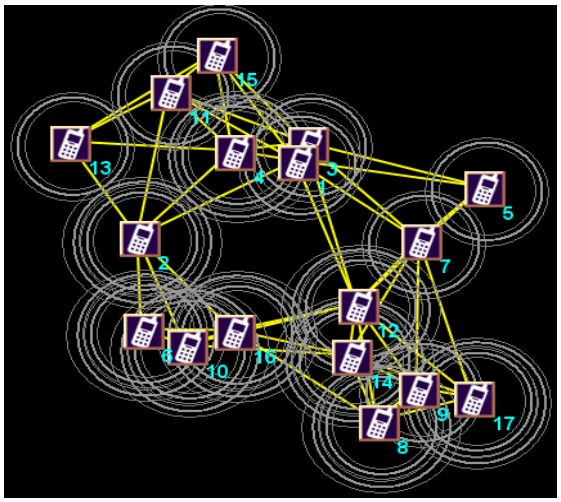

Fig. 5 Mobile Ad Hoc Autoconfiguration

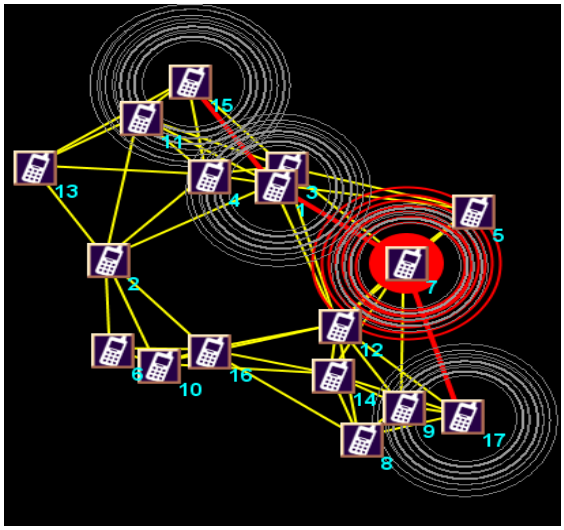

Fig. 6 Mobile Ad Hoc Communication Scenario

31st May 2010. Vol.15. No.2.

Variable length addressing can provide significant savings in network overhead. While not initially apparent, such savings can become substantial in light of source based routing protocols. When prefix is added or subtracted, it is done to all the nodes in the network. Hence, all routing addresses in the distributed address lookup change and consistency is maintained.

Address space is efficiently used. When a node enters a network, it is treated as if two networks come in contact. Already existing nodes get the same prefix "0", whereas the new node is treated as a different network. Hence, merging of different networks is just the special case of a node joining the network. When a node leaves the network, it will relinquish the address and inform its followers of the free address spaces. If it happens to be the *creator* with no followers, it will inform all the nodes in the network to decrease the prefix by one bit, preventing wastage of addresses.

EFFICIENCY OF THE PROPOSED ALGORITHM

The size of routing table at each node is O(log n), where n is the number of nodes in the network. This can be derived from the fact that, with k bit addressing, there can be at most k creators and 2^k nodes in the network. This facilitates scalability for very large mobile ad hoc networks. The frequency of change in address length is high during the initial stages of ad hoc network configuration. However, as the network expands and stabilizes, it becomes less frequent because for every single bit increase in address length, the address space is doubled.

6. CONCLUSION AND FUTURE WORKS

Address autoconfiguration schemes need a concrete solution for security since the attacks on networks are becoming increasingly intelligent and more fatal. This paper presents a spanning-tree based dynamic variable length address autoconfiguration, a novel method to solve the problem of address autoconfiguration of mobile ad hoc networks.

The problem of dynamic addressing in mobile ad hoc networks has been investigated and proposed a spanning tree based variable length dynamic address assignment scheme. The proposed scheme allocates the address resources efficiently to different kinds of topology and ensures the timely assignment of unique addresses. In particular, it helps source routing due to variable length addressing. The issues of nodes joining and departing mobile ad hoc networks were discussed. In the future, we will conduct Qualnet 4.5 based simulation study to test the robustness of the proposed novel method.

REFERENCES:

[1]. C.K. Tok(2002), *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Pearson Education, pp. 28-30, 2002.

[2]. X. Cheng, X. Huang and D.Z. Du(2006), "*Ad Hoc Wireless Networking*", pp. 319-364, Kluwer Academic Publishers, 2006.

[3]. Kilian Weniger and Martina Zitterbart(2004), "*Address Autoconfiguration in Mobile Ad Hoc Networks: Current approaches and Future Directions"*, IEEE Network, Page No. 6-11, July/August 2004.

[4]. M. Gunes and R. Reibel(2002), "*An IP address configuration algorithm for Zeroconf mobile multihop ad hoc networks*", Proceedings of International Workshop on broadband wireless ad hoc networks and services, Sophia Antipolis, France, September 2002.

[5]. S. Nesargi and R. Prakash(2002), "*MANETconf: Configuration of hosts in a mobile ad hoc network*", Proceedings of the IEEE INFOCOM 2002, New York, NY, June 2002.

[6]. J. Boleng(2002), "*Efficient network layer addressing for mobile ad hoc networks*", Proceedings of International Conference on wireless networks, pp. 271–277, Las Vegas, NV, June 2002.

[7]. H. Zhou, L.M. Ni, and M.W. Mutka(2003), "*Prophet Address allocation for large scale MANETs*", Proceedings of the IEEE INFOCOM 2003, San Francisco, CA, March 2003.

[8]. M. Mohsin and R. Prakash(2002), "IP address assignment in a mobile ad hoc network", Proceedings of the IEEE MILCOM 2002, Anaheim, CA, October 2002.

[9]. A.P. Tayal and L.M. Patnaik(2004), "*An address assignment for the automatic configuration of mobile ad hoc networks*", Personal Ubiquitous Computer, 8(1), 47–54, 2004.

[10]. Charles E. Perkins, Jari T. Malinen, Ryuji Wakikawa, Elizabeth M. Belding-Royer, Yuan Sun(2001), "*IP address autoconfiguration for ad hoc networks*", IETF draft, November 2001.

[11]. N.H. Vaidya(2002), "*Weak duplicate address detection in mobile ad hoc networks*", Proceedings of the ACM mobi-Hoc 2002, pp. 206–216, Lausanne, Switzerland, June 2002.

[12]. K. Weniger(2003), "*Passive duplicate address detection in mobile ad hoc networks*", Proceedings of the IEEE WCNC 2003, New Orleans, LA, March 2003.

[13]. Y. Sun and E.M. Belding-Royer (2003), "*Dynamic address configuration in mobile ad hoc networks*", UCSB Technical Report 2003-11, Santa Barbara, CA, June 2003.

[14]. K. Weniger(2005), "*PACMAN: Passive autoconfiguration for mobile ad hoc networks*", IEEE JSAC, Special Issue on Wireless Ad Hoc Networks, Vol. 23, No. 3, pp. 507-509, March 2005.

[15]. Longjiang Li, Yunze Cai, Xiaoming Xu(2007), *Spanning-tree based autoconfiguration for mobile ad hoc networks"*, Wireless Personal Communications, Springer Science, Netherlands, Page No.1465–1477, Volume 43, Number 4, December 2007.

[16]. Som Chandra Neema and Venkata Nishanth Lolla, *Variable Length and Dynamic Addressing for Mobile Ad Hoc Networks.*

[17]. Longjiang Li, Yunze Cai, Xiaoming Xu(2009), *Cluster based autoconfiguration for mobile ad hoc networks*, Wireless Personal Communications, Springer Science, Netherlands, Page No.1465–1477, Volume 49, 2009.

[18]. M.R. Thoppian, Ravi Prakash(2006), *Distributed protocol for dynamic address assignment in mobile ad hoc networks*, IEEE Transactions on mobile computing, Vol.5, Issue 1, Page No. 4-19, January 2006.

[19]. Namhoon Kim, Soyeon Ahn, Younghee Lee(2007), *AROD: address autoconfiguration with address reservation and optimistic duplicated address detection for mobile ad hoc networks*, Computer Communications, Vol. 30, Issue 8,Page No. 1913-1925, June 2007.

[20]. Emilio Ancilotti, Raffaele Bruno, Marco Conti and Antonio Pinizzotto(2009), *Dynamic address autoconfiguration in hybrid ad hoc networks*, Pervasive and Mobile Computing, Vol. 5, Issue 4, Page No. 300-317, August 2009.

[21]. Sang-Chul Kim and Jong-Moon Chug(2008), *Message complexity analysis of autoconfiguration protocols*, IEEE Transactions on Mobile Computing, Vol. 7, Issue 3, Page No. 358-371, March 2008.

[22]. M. Fazio, M. Villari, A. Puliafito(2004), *Merging and partitioning in ad hoc networks*, Proceedings of the Ninth International Symposium on Computers and Communications, Vol. 2, Page No. 164 - 169, IEEE Computer Society, 2004.

[23]. Stephen Toner and Donal O'Mahony, *Self-Organising Node Address Management in Ad-hoc Networks*, Lecture Notes in Computer Science Series, Personal Wireless Communications Book, Vol. 2775, Page No. 476-483, September 2003.