ISSN (e): 2250 – 3005 || Vol, 04 || Issue, 8 || August – 2014 ||
International Journal of Computational Engineering Research (IJCER)

An Efficient FB Addressing Protocol for Auto configuration of Ad Hoc Networks

Kg Mohanavalli[1], P Nageswara Rao[2], Rameswara Anand[3]
1(Dept of Cse, Swetha Institute of Technology and Science, Tirupati)
2(Associate Professor & Head, Dept Of Cse, Swetha Institute Of Technology And Science, Tirupati)
3(Professor, Dept Of Cse, Swetha Institute Of Technology And Science, Tirupati)

ABSTRACT
Address assignment is a main challenge in ad hoc networks due to the lack of groundwork. Self-determining addressing protocols require a distributed and automatic mechanism to avoid address collisions in a aggressive network with paling channels, usual partitions, and adding/deleting nodes. We propose and evaluate a lightweight protocol that configures mobile ad hoc nodes based on a shared address database stored in filters that decreases the control load and makes the proposal potent to packet losses and network partitions. We evaluate the achievement of our protocol, considering adding nodes, partition merging events, and network declaration. Simulation results show that our protocol resolves all the address collisions and also decreases the control traffic when compared to previously proposed protocols.

Index Terms— *FB, Ad hoc networks, computer based network management*

I. INTRODUCTION

Ad hoc networks do not require any previous groundwork and rely on dynamic multihop topologies for passage forwarding. Sensing, Internet access to deprived communities, and disaster recovering are called as the distributed applications. To makes these several applications it requires centralized administration. An essential issue of ad hoc networks is the frequent network partitions. Frequent partitions, caused by node mobility, fading channels [1], and joining nodes and leaving nodes in the network, can interrupt the distributed Network control. Because of the lack of servers in the network [2], Network initialization is the key challenge in Ad hoc Network.

As other wireless networks, ad hoc nodes also need a unique network address to enable multihop routing and full connectivity. Address assignment is key challenging in ad hoc networks, due to the self-organized nature of these situations. Centralized mechanisms, such as the Dynamic Host Configuration Protocol (DHCP) or the Network Address Translation (NAT), conflict with the distributed nature of ad hoc networks and do not address network detachment and integration. In this paper, we propose and analyze an efficient approach called Filter-based Addressing Protocol (FAP) [3]. The proposed protocol maintains a distributed database stored in filters containing the currently allocated addresses in a compressed manner. We consider Sequence filter contains both Sequence filter and proposed filter, to design a FB protocol that assures both the univocal address configuration of the nodes joining the network and the detection of address conflict after merging detachment. In filter-based approach simplifies the univocal address allocation and the detection of address conflict because every node can easily check whether an address is already assigned or not. We also propose to use the hash filter as a partition identifier, to provide an important feature for an easy detection of network detachment. Hence, we introduce the filters to store the allocated addresses without sustaining in high storage transparency. The filters are distributed maintained by exchanging the hash of the filters between neighbors. This allows nodes to detect with a small control transparency neighbors using different filters, which could affect address conflict. Because of these reason, our proposed method as a robust addressing scheme because it guarantees that all nodes share the same allocated list.

We compare FAP performance with the main address auto configuration proposals for ad hoc networks [4]–[6]. Analysis and simulation experiments show that FAP achieves low communication transparency and low latency, resolving all address conflict even in network partition merging events. These results are mainly associated to the use of filters because they reduce the number of tries to allocate an address to a combination node, as well as they reduce the number of false positives in the partition integration of events, when comparing to other proposals, which diminish message transparency.

II. RELATED WORK

The lack of servers hinders the use of centralized addressing schemes in ad hoc networks. In simple disseminated addressing schemes, however, it is hard to shun duplicated addresses since a random choice of an address by each node would result in a high conflict probability, as established by the birthday paradox [7]. The IETF Zero conf working group proposes a hardware-based addressing scheme [8], which assigns an IPv6 network address to a node based on the machine MAC address. however, if the number of bits in the address suffix is smaller than number of bits in the MAC address, which is forever factual for IPv4 addresses, this key must be adjusted by hashing the MAC address to fit in the address suffix. Hashing the MAC address, though, is similar to a random address preference and does not guarantee a collision-free address allocation.

Address auto configuration proposals that do not store the list of allocated addresses are typically based on a distributed protocol called Duplicate Address Detection (DAD) [4]. In this protocol, every joining node randomly chooses an address and floods the network with an Address Request message (AREQ) for a number of times to guarantee that all nodes receive the new allocated IP address. If the arbitrarily selected address is already allocated to another node, this node advertises the replica to the joining node sending an Address Reply message (AREP). When the joining node receives an AREP, it arbitrarily selects another address and repeats the overflowing process. Otherwise, it allocates the selected address. This proposal, nevertheless, does not take into account network divisions and is not suitable for ad hoc networks.

A few extensions to the Duplicate Address Detection (DAD) protocol use Hello messages and partition identifiers to handle network partitions [5], [9]. These identifiers are random numbers that identify each network partition. A collection of nodes modifies its partition identifier whenever it identifies a partition or when partitions merge. Fan and Subramanian propose a protocol based on DAD to solve address collisions in the presence of network integration of events. This protocol thinks that two partitions are merging when a node receives a Hello message with a partition identifier different from its own identifier or when the neighbor set of any node changes [5].

Other proposals use routing information to work around the addressing problem. Weak DAD [10], for instance, routes packets correctly even if there is an address conflict. In this protocol, every node is recognized by its address and a key. DAD is executed on the single-hop region, and collisions with the other nodes are identified by information from the steering protocol. If several nodes select the same address and key, nevertheless, the address conflict is not detected. Likewise, Weak DAD depends on modifying the routing protocols.

Other more complex protocols were proposed to improve the performance of network merging detection and address reallocation [6], [11]. In these protocols, nodes store additional data structures to run the addressing protocol. MANET conf [6] is a stateful protocol based on the concepts of mutual exclusion of the Ricart–Agrawala exclusion algorithm. In that the protocol, nodes store two field of address lists: the Allocated list and the assigned Pending list. A combination node asks for an address to a neighbor, which be converted into a leader in the address allocation procedure. The leader decides an available address, accumulates it on the Allocated Pending list, and floods the network. If all MANET conf nodes accept the allocation request and positively answer to the leader, followed by the leader report to the allocated address to the joining node, progress the distributed address to the assigned list, and floods the network again to authenticate the address allocation. After accept this message, each node moves the address from the distributed Pending list to Allocated list. MANET conf handles address reallocation; however network division detection depends on interrupted flooding. Hence this protocol acquires in a high control overhead.

Another stateful protocol is the Dynamic Address assignment Protocol (DAP) in mobile ad hoc networks [11], which is based on presented-address sets, Hello communication, and partition identifiers. In DAP; a node subdivides its available address set with a joining node whenever it is argued for an address by the fusion node. As node has a vacant address set, it rises for an address set reallocation. This reallocation and the recognition to give address is not being used any longer can foundation a high control load in the network, Based on how the addresses are circulated between nodes. DAP involves the use of DAD in integration events not only for the distributed addresses; however available address list stored in every node, improves the control load.

Prophet [12] allocates addresses based on a pseudo-random function through high entropy. The original node in the network, identified as prophet, selects seed for arbitrary sequence and assigns addresses to any joining node that contacts it. The combination of nodes starts to allocate addresses to other nodes from different points of the arbitrary sequence, building an address assignment tree. Prophet does not overflow the network

and, the arbitrary consequence, generates a low control load. The protocol, however, requires an address range much larger than the previous protocols to support the same number of nodes in the group of network. Likewise, it is based on the feature of the pseudo-random generator to evade duplicated addresses. Hence, it needs DAD mechanism, to detect duplicated addresses, which enhances the protocol intricacy and eliminates the advantage of a low control message transparency.

Our proposal aims to reduce the control load and to improve partition merging detections without requiring high storage capability. These intentions are achieved through small filters and an accurate dispersed mechanism to update the states in nodes. Moreover, we propose the use of the filter signature (i.e., a hash of the filter) as a partition identifier in preference to random numbers. The filter signature signifies the set of all the nodes within the partition. Consequently, if the set of allocated addresses changes, the filter signature also will change. Essentially, when using random numbers to identify the partition in preference to hash of the pass through a filter, the identifier does not change with the set of assigned addresses. Hence, filter signatures increases the capability to detect and merge partitions.

III. FAP

The proposed protocol aims to dynamically auto configure network addresses determine collisions with a low control load, yet in joining or merging events. To achieve these objectives, FAP uses a distributed compact filter to signify the current set of distributed addresses. This filter is present at every node to abridge frequent node joining events and reduce the control transparency essential to solve address collisions intrinsic in random obligations. Furthermore, we propose the filter signature, which is the hash of the address filter, as a partition identifier. The address filter signature is an significant feature for easily identifying network merging events, in which address collision may occur. FAP uses two different filters, depending on the scenario: the Bloom filter, which is support on hash functions, and the Sequence filter, which reduces data based on the address sequence.

A. Bloom Filters

The Bloom filter is a compact data structure used on distributed applications [13], [14]. The Bloom filter is composed of an -bit vector that represents a set $\mathbf{A= \{a_1, a_2, a_3 \ldots\ldots a_n \}}$ collection of elements. This elements are inserted into the filter which is through a set of independent hash functions, $\mathbf{h_1 , h_2 , h_3, \ldots\ldots h_k ,}$ whose outputs are uniformly distributed over the m bits

$$P_{fp} = (1-p_0)^k \qquad (1)$$

B. Sequence Filters

The other filter structure that we propose is called Sequence filter, and it stores and compress the addresses based on the sequence of addresses. This filter is produced by the concatenation of the first address of the address sequence, which we call primary element , with an -bit vector, where is the address range size. In this filter, each address suffix is represented by one bit, indicate to give the distance between the initial element suffix and the current element suffix .If a bit is in 1, then the address with the given suffix is considered as inserted into the filter; if not, the bit in 0 indicates that the address does not go to the filter. Consequently, there are neither false positives nor false negatives in the Sequence filter since each available address is deterministically represented by its relevant bit.

C. Procedures of FAP

1) Network Initialization: The network initialization procedure deals with the auto configuration of the primary set of nodes. Two different circumstances can happen at the initialization: the joining nodes appear one after the other with a long sufficient interval among them, called gradual initialization, or all the nodes appear at the same time, called abrupt initialization. Most protocols assume the gradual scenario with a large time interval between the arrival of the first and the second joining nodes. If all nodes join the network approximately at the equivalent time, each partition will choose a different partition identifier. This triggers many partition merging procedures concurrently, which creates a high control load and it cause inconsistency in the address assignment procedure, generating address conflicts. We dispute that address allocation protocols must operate lacking any restriction to the way the nodes join the network. Our FB proposal fits well for both gradual and rapid initialization scenarios, called Hello and AREQ messages, shown in Fig. 3(a) and (b). The Hello message is used by a node to advertise its current association status as division of identifier. The AREQ message is used to broadcast that the existing address is now allocated. Each AREQ has an unique number, which is used to differentiate AREQ messages produced by different nodes, but with the similar address.

In FAP, a node trying to join the network listens to the medium for a period . If the node does not receive a Hello message within this period T_L, then it starts the network, acting as the initiator node. An initiator node may start the network only, or with other originator nodes. or else, if the node receives a Hello message, then the network previously survives and the node acts as a joining node. An initiator node randomly selects an address, assuming the address range defined by the bits of the network prefix, constructs an empty address filter, and initiates the network initialization phase. In this phase, the node floods the network N_F times with AREQ messages to improve the possibility that all originator nodes receive the AREQ message. If there are other initiator nodes, they also send their AREQ N_F times, promoting their randomly selected addresses. After waiting a period T_W without listening to AREQs from other orginator nodes, suppose if they exist, the node disappear the initialization phase and introduce on the address filter all the available addresses accepted with AREQs. After the node begin to send Hello messages through the address filter signature, which is a hash of the filter. This signature recognize the network and is used to detect partitions, if they occur. If the initiator node receives any AREQ with the similar address that it has selected, however with a different identifier number, which means that there is an address conflict, the node waits for a period T_C and then selects another existing address and pass another AREQ. During the period , T_C the node receives more AREQs with other already allocated addresses. Therefore, after T_C, the node knows a more complete list of distributed address, which reduces the probability of selecting used address. Therefore, the periods TC reduce the probability of address conflicts and, accordingly, decrease the network control load.

After the initialization of FAP, all initiator nodes have seleted a unique address due to the arbitrary address choice and the substantiation using AREQ messages with identifier numbers. Moreover, each node knows all presently allocated addresses with a high possibility due to the N_F times flooding the network. Therefore, each node create an address filter including all the assigned addresses.

2) Node Ingress and Network Merging Events: After the FAP initialization, every node begin to transmit periodic Hello messages including its address filter signature. Ahead the function of a Hello, neighbors estimate whether the signature in the message is the same as its own signature to sense merging events. Only the nodes that have already merged network are capable to send Hello messages, receive a request of a node to join the network, and sense merging events.

The node ingress procedure is illustrated in Fig. 4(a). When a node turns on, it listens to the medium for a period T_L . If the node eavesdrops to a Hello, there is at least single node with an address filter, and the network already be present. Consequently the node knows that it is a combination node instead of an originator node. The merging node then asks for the source of the first snooped Hello message(the host node) to send the address filter of the network using an Address Filter (AF) message, shown in Fig. 3(c).When the host node retrieves the AF, it checks bit , which specifies whether the message is individually used for a node-merging procedure or a partition-joining procedure. If I=1, the message came from a merging node. Then, the host node replies the request with another AF with bit set to 1, indicate that the AF is an answer to a existing filter request. When the joining node retrieves the AF response message, it stores the address filter, selects a arbitrary existing address, and floods the network with an AREQ to assign the new address. When the other nodes retrieve the AREQ, they include the new address in their filters and modernize their filter signatures with the hash of the revised filter.

Joining events are also identified depend on Hello and AF messages, as illustrated in Fig. 4(b). Nodes in dissimilar partitions select their address depends only on the group of addresses of their network partition. Consequently, nodes in dissimilar partitions can select the similar address, which may origin address conflicts after the partitions joined. In FAP protocol, when a node retrieves a Hello, it verifies whether the filter signature on the message is dissimilar than its present signature. Hence, the node knows that they have different sets of allocated addresses.

IV. SIMULATION RESULTS

We implemented FAP in the Network Simulator-2 (NS-2) and evaluated it considering the Shadowing model for radio propagation and the NS-2 IEEE 802.11 model for the MAC. These replicas account for making a circumstances related to a real neighborhood network, using constraints of profitable equipments. Hence, the constraints are used for our simulations are: an average broadcasting range of 18.5 m, a highest carrier logic range of 108 m, and a density of 0.0121 nodes/m [16]. We measured the control traffic, the delays

TABLE I: PARAMETERS OF FAP (F), DAD-PD (PD), DAD (D), AND MCONF (M)

Variable	Description	Value	Protocol
T_L	Max. time listening to the medium	1.0 s	F, PD
T_P	Partition merging min. interval	3.0 s	F, PD
T_W	AREQ/AREP waiting time	1.2 s	F, PD, D
T_R	Message replication interval	0.3 s	F, PD, D
T_H	Hello Timer	1.0 s	F, PD, M
T_C	Waiting time before changing address	0.3 s	F
T_S	Generated-filter-signature storage time	0.5 s	F
$T_{S'}$	Received-filter-signature storage time	3.0 s	F
T_M	Min. interval between filter renews	5.0 s	F
T_F	Max. time waiting for a filter	0.5 s	F
T_{MA}	Address Allocation Timer	0.7 s	M
T_{MR}	Request Reply Timer	0.3 s	M
T_{MN}	Neighbor Reply Timer	0.35 s	M
T_{MAP}	Allocation Pending Timer	1.0 s	M
T_{MP}	Partition Timer	2.0 s	M
N_F	Transmissions of a flooding message	2	F, PD, D
N_T	Neighbor Reply Threshold	3	F, M
N_{MR}	Request Reply Retry	2	M
N_{MI}	Initiator Request Retry	2	M
N_{MP}	Partition Identifier Threshold	3	M

and the number of address collisions in FAP, considering a confidence level of 95% in the consequences. We also realized in Network Simulators-2 the addressing protocols projected by Perkins *et al.* [4], called DAD, by Fan and Subramani [5], which we call DAD-PD, and by Nesargi and Prakash [6], called MANET conf and indicated in the outcomes by Mconf.3 while DAD does not work in network partition-level locations, we estimated this protocol since it is a simple proposal with low transparency. Our main intention is to show that our proposal also near by a low transparency and works in any development. Comparing FAP to DAD-PD, we observe the performance collision of the use of the hash of the address filters as a substitute of mediated partition identifiers to sense partition merging events. In the original DAD-PD, but, the new partition identifier after a partition joining is given by the sum of network partition identifiers, which causes volatility in the protocol. Hence, we increased the protocol performance by selecting the huge partition identifier in network joining events in preference to sum them, which decrease the number of negative partition joining detections. Additionally, when compared FAP to MANET conf because both proposals use an allocated address list. The protocol constraints are shown in Table. These parameters were selected based on experiments to improve all the four protocols routine and also on proposals from the authors of the other proposals. Therefore, we have chosen these values edging on decreasing the delays and the transparency while at rest evading instabilities in the simulated scenario. Particularly, includes the time listening to the medium before the node choose if it is alone or not. Consequently, this phase must be, at least, equal to the Hello Timer .

The minimal interval between partition merging events,
avoids high overheads in FAP in environments prone to high forwarding delays and/or many packet failures. This value is even more significant for the DAD-PD protocol, in which partition joining mechanisms are frequently called. We estimate this constraint choice throw simulations. Likewise, both the constraints , which identifies the intermission among retransmissions of flooding messages, and , which decreases the address conflicts during the initialization phase, contacts on FAP performance and are also appraised in the following simulations. The constraints is used through FAP initialization to identify when a node is allowed to use its selected address. Therefore, this interval identifies the period that a node should wait for more AREQs before

terminating the initialization phase,and does not hamper in the protocol stabilization interruption. The values and avoid wrong detections of partition joining events through the new node and partition joining events. The use of high values provisionally increases the storage transparency. The minimum interval among filter restores, collisions FAP transparency only if the network is full. This interval defines the frequency in which the filter is checked to discover if any node has left the network to provide addresses for merging nodes. A small implies on a frequent substantiation, which improves the transparency, while a high involves in low transparency, but long wait for new nodes to merge an almost complete network. The value controls the time a merging node waits until the chosen neighbor sends the present address filter. This timer only presents flexibility to fault nodes or to neighbors that leave the transmission range of the joining node, and then it usually does not influence on protocol performance. The number of transmissions of a flooding message, , also impacts FAP performance and is evaluated through simulations. Finally, the last FAP parameter, , has a resilience function similar to and presents a small impact over FAP performance.

Both FAP and DAD-PD use equal-sized Hello messages because we assume that both partition identifier and filter signature are composed of 4 B. We assume an address range of 150 addresses and a network with a maximum of 100 nodes to guarantee that the address range is not a constraint that can cause instabilities for any practice. Based on these parameters, we have used Sequence Filter of 23 B. We first evaluate the collision of one node merging the network. Therefore We have considered a rectangular space with nodes dispersed in grid.

We evaluate the control load after the last node merge the network and the involve delay to acquire an address, as revealed in Fig. 6. In the consequences, we monitor that our proposal, FAP, current an transparency larger than DAD since our protocol uses Hello messages to notice partitions. DAD-PD currents the greatest control transparency for more than 36 nodes since unnecessary partition joining practices are be ginned caused by mistakes in partition joining event detection after a node merge the network. in fact, DAD-PD has a partition detection mechanism that is depend on partition identifiers. When a new node merge the network, the partition identifier must be altered to signify the new set of assigned addresses. The modernize of this value, yet, can root negative partition joining detections which improve the control load transparency. For other than

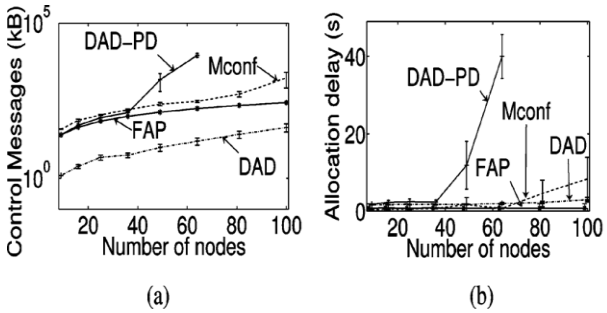

Fig. 1. Collision of a joining node procedure followed to the many number of nodes

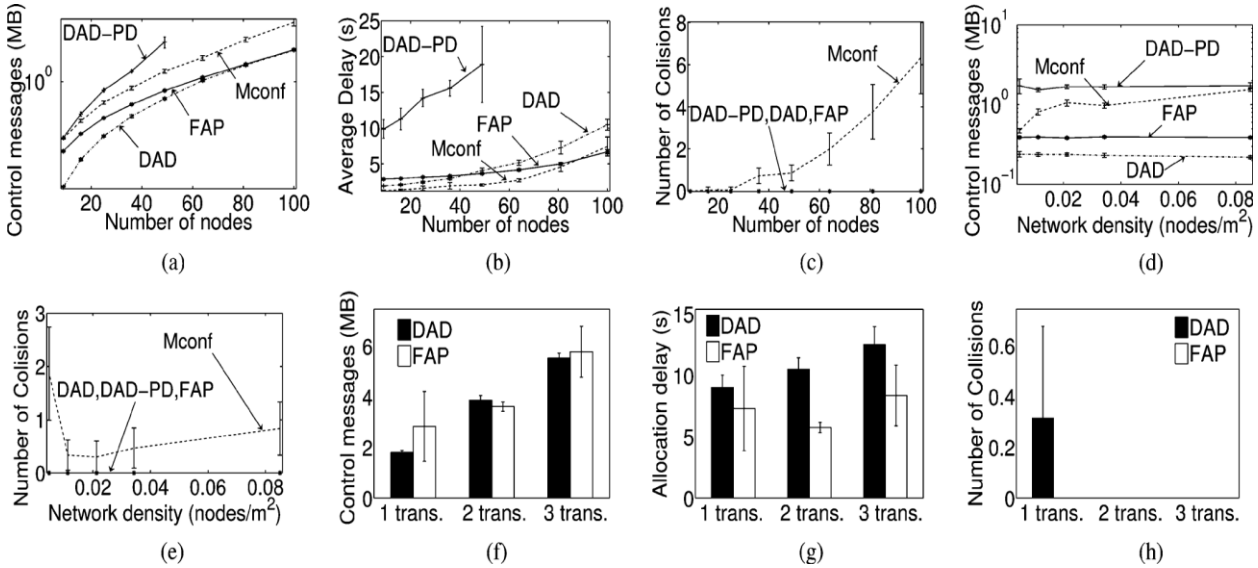

Fig. 7. Collision over the network of the number of nodes, the density, and the number of broadcasts of flooding messages (N_F) in a network rapidly initialized.(a) Control transparency according to the number of nodes. (b) Average interruption according to the many number of nodes. (c) Number of address conflicts based on the number of nodes. (d) Control transparency is depend on network density N=36, nodes. (e) Number of collisions according to network density N=36, nodes. (f) Control overhead according to N_F,N=100 , nodes. (g) Average delay according to N_F,N=100 , nodes. (h) Number of collisions according to N_F,N=100 , nodes.

since this protocol switches joining events, but it could not evade all the address conflicts. FAP presented no address collision, but DAD-PD offered a small probability of conflict caused by packet losses. FAP omits this kind of address conflict because of the mechanism that detects packet losses and begins fake joining methods until all the nodes have retrieved the similar information. since a result, FAP determines all the conflicts and also presented a minimal control load. In Fig. 7, we have guessed the collision of the retransmissions of flooding packets in network initialization. Now, we analyze the collision of the delay among flooding message retransmission, the time waiting before selecting a new address in the initialization and the minimal interval among partition joining events. First, we have analyzed the conflict of the network with 100 nodes. We illustrated in Fig. 9(a) that a huge partition decreases the transparency since the nodes learn more allocated addresses before selecting a new address. However, this also improves the delays in the network, as illustrated in Fig. 9(b). An intermediate value in this scenario is s, which was used in the other simulations. In the same scenario, we analyze and observed that a small reduces both the overhead and the delay. We repeated the partition merging scenario with four partition merging events. Fig. 9.Analyzing performance according to different values of FAP parameters. (a) Control overhead. (b) Average delay to obtain an address. In this scenario, . We observe in Fig. 9(a) that a small is better for FAP, but is very prejudicial for DAD-PD. Indeed, FAP presents a better partition merging event detection than DAD-PD. DAD-PD performs many unnecessary partition merging mechanisms and a greater reduces these false detection events. Nevertheless, a greater also increases the stabilization time of DAD-PD during network initialization. Then, we selected an intermediate value for to balance the requisites of both protocols.

V. CONCLUSION

We proposed a distributed and self-managed addressing protocol, called as FB Addressing protocol, which is robust for dynamic ad hoc networks with fading channels, numerous partitions, and merging/leaving nodes. Our key idea is to use address filters to omit address conflicts, which decrease the control load, and reduce the address allocation interruption. We have also proposed to use the hash of the filter as the partition identifier, providing an easy and accurate feature for partition detection with a small number of control messages. Moreover, our filter-based protocol improves the protocol vigorousness to message losses, which is an main issue for ad hoc networks with fading channels and high bit error rates.

The use of the hash of the filter instead of a random number as the partition identifier creates a better representation of the group of nodes. Therefore, a change in the group of nodes is mechanically reflected in the partition identifier. This identifier is occasionally presented, allowing neighbors to identify if they belong to different sets of nodes. In the other proposals, mechanism to change the arbitrated partition identifier is requested, which increases the complexity and the packet overhead of the protocol.

The proposed protocol efficiently resolves all address collisions even during merging events, as showed by simulations. This is achieved because FAP is able to detect all merging events and also because FAP is robust to message failures. FAP initialization practice is straightforward and proficient, requiring a control load related to the control pack of DAD, which is a protocol with a small transparency but that does not handle network partitions. Moreover, FAP presents smaller delays in the joining node procedure and on network partition merging events than the other applications, designating that the proposed protocol is more appropriate for very dynamic environments with frequent partition merging and node joining events.

REFERENCES

[1] D. O. Cunha, O. C. M. B. Duarte, and G. Pujolle, "A cooperation-aware routing scheme for fast varying fading wireless channels," IEEE Commun. Lett., vol. 12, no. 10, pp. 794–796, Oct. 2008.

[2] N. C. Fernandes,M.D.Moreira, and O. C. M. B. Duarte, "Aself-organized mechanism for thwarting malicious access in ad hoc networks," in Proc. 29th IEEE INFOCOM Miniconf., San Diego, CA, Apr. 2010, pp. 1–5.

[3] N. C. Fernandes,M.D.Moreira, and O. C. M. B. Duarte, "An efficient filter-based addressing protocol for autoconfiguration of mobile ad hoc networks," in Proc. 28th IEEE INFOCOM, Rio de Janeiro, Brazil, Apr. 2009, pp. 2464–2472.

[4] C. E. Perkins, E. M. Royers, and S. R. Das, "IP address autoconfiguration for ad hoc networks," Internet draft, 2000.

[5] Z. Fan and S. Subramani, "An address autoconfiguration protocol for IPv6 hosts in a mobile ad hoc network," Comput. Commun., vol. 28, no. 4, pp. 339–350, Mar. 2005.

[6] S. Nesargi and R. Prakash, "MANETconf: Configuration of hosts in a mobile ad hoc network," in Proc. 21st Annu. IEEE INFOCOM, Jun. 2002, vol. 2, pp. 1059–1068.

[7] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks," in Proc. IEEE Symp. Security Privacy, May 2005, pp. 49–63.

[8] S. Thomson and T. Narten, "IPv6 stateless address autoconfiguration," RFC 2462, 1998.

[9] M. Fazio, M. Villari, and A. Puliafito, "IP address autoconfiguration in ad hoc networks: Design, implementation and measurements,"Comput. Netw., vol. 50, no. 7, pp. 898–920, 2006.

[10] N. H. Vaidya, "Weak duplicate address detection in mobile ad hoc networks," in Proc. 3rd ACM MobiHoc, 2002, pp. 206–216.

[11] H. Kim, S. C. Kim, M. Yu, J. K. Song, and P.Mah, "DAP: Dynamic address assignment protocol in mobile ad-hoc networks," in Proc. IEEE ISCE, Jun. 2007, pp. 1–6.