INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

# EFFICIENT AND ROBUST ADDRESSING PROTOCOL FOR NODE AUTOCONFIGURATION IN SENSOR NETWORK

**P. Sathya Priya[1], Dr. S. Uma[2]**

*[1]Post Graduate Student, Department of Computer Science,*
*Hindusthan Institute of Technology, Coimbatore,*
*sathyapr03@gmail.com*
*[2] Head of Department, Department of Computer Science,*
*Hindusthan Institute of Technology, Coimbatore,*
*umakarunahind@gmail.com*

## Abstract

Address assignment is a key challenge in ad hoc networks due to the lack of infrastructure. Autonomous addressing protocols require a distributed and self-managed mechanism to avoid address collisions in a dynamic network with fading channels, frequent partitions, and joining/leaving nodes. We propose and analyze a lightweight protocol that configures mobile ad hoc nodes based on a distributed address database stored in filters that reduces the control load and makes the proposal robust to packet losses and network partitions. We evaluate the performance of our protocol, considering joining nodes, partition merging events, and network initialization. Simulation results show that our protocol resolves all the address collisions and also reduces the control traffic when compared to previously proposed protocols.

**Keywords***:* Sensor networks, Computer network management, Ad hoc networks.

## 1. Introduction

MOBILE networks do not require any previous infrastructure and rely on dynamic multihop topologies for traffic forwarding. The lack of a centralized administration makes these networks attractive for several distributed applications, such as sensing, Internet access to deprived communities, and disaster recovering. A crucial and usually unaddressed issue of ad hoc networks is the frequent network partitions. Network partitions, caused by node mobility, fading channels [1], and nodes joining and leaving the network, can disrupt the distributed network control. Network initialization is another challenging issue because of the lack of servers in the network [2].

As other wireless networks, ad hoc nodes also need a unique network address to enable multihop routing and full connectivity. Address assignment in ad hoc networks, however, is even more challenging due to the

self-organized nature of these environments. 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 partitioning and merging. 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 compact fashion. We consider both the Bloom filter and a proposed filter, called Sequence filter, to design a filter-based protocol that assures both the univocal address configuration of the nodes joining the network and the detection of address collisions after merging partitions. Our filter-based approach simplifies the univocal address allocation and the detection of address collisions because every node can easily check whether an address is already assigned or not. We also propose to use the hash of this filter as a partition identifier, providing an important feature for an easy detection of network partitions. Hence, we introduce the filters to store the allocated addresses without incurring in high storage overhead. The filters are distributed maintained by exchanging the hash of the filters among neighbors. This allows nodes to detect with a small control overhead neighbors using different filters, which could cause address collisions. Hence, our proposal is a robust addressing scheme because it guarantees that all nodes share the same allocated list. We compare FAP performance with the main address autoconfiguration proposals for ad hoc networks [4]–[6]. Analysis and simulation experiments show that FAP achieves low communication overhead and low latency, resolving all address collisions even in network partition merging events. These results are mainly correlated to the use of filters because they reduce the number of tries to allocate an address to a joining node, as well as they reduce the number of false positives in the partition merging events, when compared to other proposals, which reduces message overhead.

## 2. System Assumptions

The lack of servers hinders the use of centralized addressing schemes in ad hoc networks. In simple distributed addressing schemes, however, it is hard to avoid duplicated addresses because a random choice of an address by each node would result in a high collision probability. The IETF Zeroconf working group proposes a hardware-based addressing scheme [8], which assigns an IPv6 network address to a node based on the device MAC address. Nevertheless, if the number of bits in the address suffix is smaller than number of bits in the MAC address, which is always true for IPv4 addresses, this solution must be adapted by hashing the MAC address to fit in the address suffix. Hashing the MAC address, however, is similar to a random address choice and does not guarantee a collision-free address allocation.

Address autoconfiguration 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 address. If the randomly chosen address is already allocated to another node, this node advertises the duplication to the joining node sending an Address Reply message (AREP). When the joining node receives an AREP, it randomly chooses another address and repeats the flooding process. Otherwise, it allocates the chosen address. This proposal, however, does not take into account network partitions and is not suitable for ad hoc networks.

### *2.1 Duplicate Address Detection*

A few extensions to the Duplicate Address Detection (DAD) protocol use Hello messages and partition identifiers to handle network partitions [9]. These identifiers are random numbers that identify each network partition. A group of nodes changes its partition identifier whenever it identifies a partition or when partitions merge. Fan and Subramani propose a protocol based on DAD to solve address collisions in the presence of network merging events. This protocol considers 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.

### *2.2 Dynamic Address assignment Protocol*

Another stateful protocol is the Dynamic Address assignment Protocol (DAP) in mobile ad hoc networks [11], which is based on available-address sets, Hello messages, 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 joining node. When a node has an empty address set, it asks for an address set reallocation. This reallocation and the detection that a given address is not being used anymore can cause a high control load in the network, depending on how the addresses are distributed among nodes. DAP requires the use of DAD in merging events not only for the allocated addresses, but also for the available address list stored in each node, increasing the control load. Prophet [12] allocates addresses based on a pseudo-random function with high entropy. The first node in the network, called prophet, chooses a seed for a random sequence and assigns addresses to any joining node that contacts it. The joining nodes start to assign addresses to other nodes from different points of the random sequence, constructing an address assignment tree. Prophet does not flood the network and, as a 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 network. Moreover, it depends on the quality of the pseudo-random generator to avoid duplicated addresses. Therefore, it needs a mechanism, like DAD, to detect duplicated addresses, which increases the protocol complexity and eliminates the advantage of a low control message overhead. Our proposal aims to reduce the control load and to improve partition merging detections without requiring high storage capacity. These objectives are achieved through small filters and an accurate distributed mechanism to update the states in nodes. Furthermore, we propose the use of the filter signature (i.e., a hash of the filter) as a partition identifier instead of random numbers.

The filter signature represents the set of all the nodes within the partition. Therefore, if the set of assigned addresses changes, the filter signature also changes. Actually, when using random numbers to identify the partition instead of hash of the filter, the identifier does not change with the set of assigned addresses. Therefore, filter signatures improves the ability to correctly detect and merge partitions.

## 3. Requirements for Sensor Network Security

The proposed protocol aims to dynamically autoconfigure network addresses, resolving collisions with a low control load, even in joining or merging events. To achieve all these objectives, **File Acceleration Protocol(FAP)** uses a distributed compact filter to represent the current set of allocated addresses. This filter is present at every node to simplify frequent node joining events and reduce the control overhead required to solve address collisions inherent in random assignments. Moreover, we propose the filter signature, which is the hash of the address filter, as a partition identifier. The filter signature is an important feature for easily detecting network merging events, in which address conflicts may occur. We propose the use of two different filters, depending on the scenario: the Bloom filter, which is based on hash functions, and the Sequence filter, proposed in this paper, which compresses data based on the address sequence.

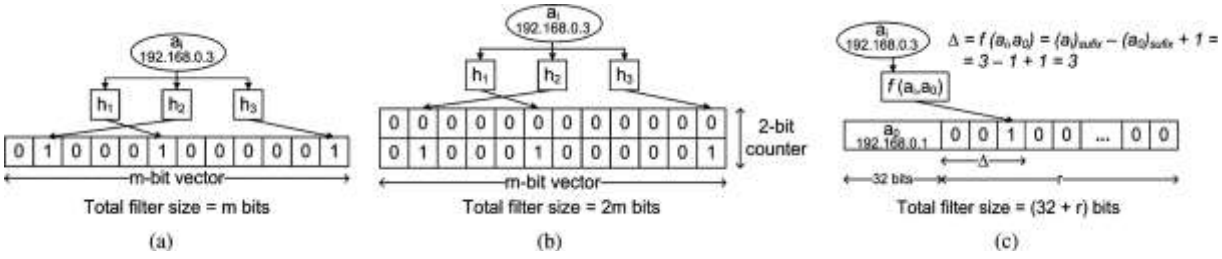

**Fig. 1. Insertion procedure of the address element $a_i = 192.168.0.3$ in the filters used with FAP. For the sequence filter, the address range, whose size is , goes from $a_0 = 192.168.0.1$ to $a_{r-1} = 192.168.0.254$ with a sub network. (a) Bloom filter with k = 3 hash functions and m = 12 bits of filter size. (b) 2-bit Counter Bloom filter with k = 3 hash functions and 2m bits of filter size. (c) Sequence filter, assuming an address range of r = 254 addresses.**

### *3.1 Bloom Filter*

The Bloom filter is a compact data structure used on distributed applications [13]. The Bloom filter is composed of an -bit vector that represents a set A = $\{a_1, a_2, \dots, a_n\}$ composed of elements. The elements are inserted into the filter through a set of independent hash functions, $h_1, h_2, \dots, h_k$, whose outputs are uniformly distributed over the m bits. First, all the bits of the vector are set to zero. After that, each element $a_i \in A$ is hashed by each of the hash functions, whose output represents a position to be set as 1 on the -bit vector, as shown in Fig. 1(a).

### *3.2 Sequence Filter*

The other filter structure that we propose is called Sequence filter, and it stores and compacts addresses based on the sequence of addresses. This filter is created by the concatenation of the first address of the address sequence, which we call initial element ($a_0$), with an -bit vector, where is the address range size. In this filter, each address suffix is represented by one bit, indexed by Δ, which gives the distance between the initial element suffix ($a_{0_{suffix}}$) and the current element suffix ($a_{i_{suffix}}$). If a bit is in 1, then the address with the given suffix is considered as inserted into the filter; otherwise, the bit in 0 indicates that the address does not belong to the filter. Therefore, there are neither false positives nor false negatives in the Sequence filter because each available address is deterministically represented by its respective bit. The Sequence filter and the procedure to insert an element into the filter are illustrated in Fig. 1(c).

### *3.3 Filter Selection*

The best filter for FAP depends on network characteristics such as the estimated number of nodes in the network and the number of available addresses. It also depends on the false-positive and false-negative rates of the filter. Bloom filters do not present false negatives, which mean that a membership test of an element that was inserted into the filter is always positive. These filters, however, present a false-positive probability. Hence, a membership test of an element that was not inserted into the Bloom filter may be positive. If we choose a false-positive probability of ≈0.06, assuming that m >> k and a 4-bit counter to avoid buffer overflow, then the ratio between the number of cells in the filter (m) and the maximum number of inserted elements (n) is m/n = 6 , and the ideal number of hash functions is 4, according to (1). Hence, the size of the Bloom filter with 4-bit counters ($b_c = 4$) is S = ( m/n )·n·$b_c$ . = 6·n·4.

### *3.4 Procedures of FAP*

***Network Initialization****:* The network initialization procedure deals with the autoconfiguration of the initial set of nodes. Two different scenarios can happen at the initialization: the joining nodes arrive one after the other with a long enough interval between them, called gradual initialization, or all the nodes arrive 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.

***Node Ingress and Network Merging Events:*** After the initialization, each node starts broadcasting periodic Hello messages containing its address filter signature. Upon the reception of a Hello, neighbors evaluate

whether the signature in the message is the same as its own signature to detect merging events. Only the nodes that have already joined the network are able to send Hello messages, receive a request of a node to join the network, and detect merging events.

***Node Departure:*** When a node leaves the network, its address should become available for the other nodes. If the departing node is correctly shut down, it floods the network with a notification to remove its address from the address filter. If the departing node does not notify the network, the address remains allocated in the filters, which can make the available addresses scarce with time. This can be identified in the address filter by the fraction of bits set to 1 in the Bloom and in the Sequence filter and by the fraction of counters greater than one in the Counter Bloom Filter. Therefore, every node verifies this fraction in their address filters every time the filter is updated. If this fraction reaches a threshold that indicates that the filter is full or almost full, all the nodes reset their address filters and returns to the network initialization. Instead of choosing a new address, the node uses its current address, which is not collided, to reduce message overhead and to avoid breaking active data connections.

## 4. Conclusion

We proposed a distributed and self-managed addressing protocol, called Filter-based Addressing protocol, which fits well for dynamic ad hoc networks with fading channels, frequent partitions, and joining/leaving nodes. Our key idea is to use address filters to avoid address collisions, reduce the control load, and decrease the address allocation delay. We 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 increases the protocol robustness to message losses, which is an important 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 set of nodes. Hence, a change in the set of nodes is automatically reflected in the partition identifier. This identifier is periodically advertised, allowing neighbors to recognize if they belong to different sets of nodes. In the other proposals, a 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 losses. FAP initialization procedure is simple and efficient, requiring a control load similar to the control load of DAD, which is a protocol with a small overhead 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 proposals, indicating that the proposed protocol is more suitable 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 cooperationaware 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, “A self-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.

[12] H. Zhou, L. Ni, and M. Mutka, "Prophet address allocation for large scale MANETs," in *Proc. 22nd Annu. IEEE INFOCOM*, Mar. 2003, vol. 2, pp. 1304–1311.

[13] M. D. D. Moreira, R. P. Laufer, P. B. Velloso, and O. C.M. B. Duarte, "Capacity and robustness tradeoffs in Bloom filters for distributed applications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 12, pp. 2219–2230, Dec. 2012.

[14] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: A scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281–293, Jun. 2000.

[15] A. Broder and M. Mitzenmacher, "Network applications of Bloom filters: A survey," *Internet Math.*, vol. 1, pp. 485–509, 2002.

[16] M. E. M. Campista, I. M. Moraes, P. Esposito, A. Amodei Jr., L. H. M. K. Costa, and O. C. M. B. Duarte, "The ad hoc return channel: A low-cost solution for Brazilian interactive digital TV," *IEEE Commun. Mag.*, vol. 45, no. 1, pp. 136–143, Jan. 2007.

## A Brief Author Biography

**P. Sathya Priya** received the B.E. degree in Computer Science and Engineering from Anna University in 2012 and currently pursuing M.E. degree in Computer Science and Engineering at Anna University. The current research focuses on secure communications in sensor networks.

**Dr. S. Uma** is Professor and Head of the PG Department of Computer Science and Engineering at Hindusthan Institute of Technology, Coimbatore, Tamil Nadu, India. She received her B.E., degree in Computer Science and Engineering in First Class with Distinction from P.SG. College of Technology in 1991 and the M.S., degree from Anna University, Chennai, Tamil Nadu, India. She received her Ph.D., in Computer Science and Engineering from Anna University, Chennai, Tamil Nadu, India with High Commendation. She has nearly 23 years of academic experience. She has organized many National Level events like seminars, workshops and conferences. She has published many research papers in National and International Conferences and Journals. She is a potential reviewer of International Journals and life member of the ISTE professional body. Her research interests are pattern recognition and analysis of nonlinear time series data.