A Comparison Study of Address Autoconfiguration Schemes for Mobile Ad hoc Network

Soyeon Ahn, Namhoon Kim, Woohyun Kim and Younghee Lee
Information and Communications University,
Computer Networks Lab., School of Engineering,

Abstract

In this paper, we classified existing address autoconfiguration schemes based on two communication models: centralized and distributed schemes. We defined performance metrics and evaluated four existing address autoconfiguration schemes. We suggested requirements for an address autoconfiguration scheme.

1. Introduction

A Mobile Ad hoc Network (MANET) usually doesn't have any fixed infrastructure or administration and the network's topology may change quickly and unexpectedly. A network protocol may affect the entire network performance because of these characteristics of MANET. We must deliberate to choose one.

Addressing is an imperative step for MANET nodes to communicate with each other. In this paper, we classified current address autoconfiguration schemes and defined important performance metrics of address autoconfiguration schemes. We evaluated the address autoconfiguration schemes in terms of various performance metrics.

This paper is organized as follows: In section 2, we classify the address autoconfiguration schemes based on communication models. In section 3, we describe the fundamental performance metrics for address autoconfiguration schemes in MANET, and in section 4, we describe our simulation environments. We show the performance of each approach through our simulation study in section 5. Finally, our conclusion is presented in section 6.

2. Classification of Address Autoconfiguration Schemes

We classified the address autoconfiguration schemes by communication models: one is the centralized schemes, and the other is the distributed schemes. The main concept of the centralized schemes is that a node will be a server node acting as DHCP server. A new node should communicate with the server node to get an address. In contrast, the distributed schemes operate in such a way that every node must communicate with each other to get an address.

2.1. The Centralized Schemes

In the centralized schemes, one server node is elected for a MANET. We see instances of centralized schemes in [1][2][3]. The server node refers to an agent node [1], a leader node [2], or an address authority [3], in each scheme, respectively, but the operation of the server nodes is similar to each other. The server node maintains an address pool and is responsible for the address allocation. When two or more MANETs merge, the number of server nodes is reduced to one.

In these methods, the duplicated addresses by network merger can be simply detected using the server node's address pool. However, the server node becomes burdened and may be a single point of failure. The larger the number of hop counts from a new node to the server node, the longer these mechanisms take to find the server node and to get an address.

Günes et al. proposed a centralized agent based address configuration scheme [1]. The address-agent node maintains the Address List (AL). It periodically broadcasts a Verify Packet that contains the AL and a time stamp. Every node that wants to remain in MANET replies an Address Confirm packet to the agent. If the agent doesn't receive the Address Confirm packet before timeout, it removes the node from the AL. If the network merging is detected, the number of agent nodes must be reduced to one. The agent having more nodes will remain as the agent of the merged network. If the number of nodes is equal,

This work is supported in part by Grant No. R01-2003-000-10562-0 from Korea Science and Engineering Foundation

the agent having a lower MAC address will be the agent of the merged network.

2.2. The Distributed Schemes

Perkins et al. proposed a distributed Duplicated Address Detection (DAD) scheme called Strong DAD [4]. A new node randomly selects an IP address and examines whether it is used in a MANET. If the chosen address is already used, it retries until it gets an unused address. The new node uses a temporary address for communication among other MANET nodes. The address allocation time increases in proportion to the number of failures. The maximum number of hop counts in a MANET also affects the allocation time. In this paper, the authors did not present any ideas for a network partition and network merger.

Sanket et al. suggested an agent-based distributed address autoconfiguration, MANETconf, using the distributed agreement concept [5]. Unlike Strong DAD, a new node, which is called a requestor, asks for an address to one of the neighbors in MANET, which is called an initiator. The initiator then randomly selects an address and gets agreements from all other nodes in MANET and assigns the address to its requestor. To acquire agreements, the initiator uses a modified DAD, receiving not NACK but ACK, which may result in an ACK explosion. Every node manages the list of all nodes in order to count the ACK messages to be received and decides the limit of the waiting time. To deal with the network partitioning and merging, the node with the lowest IP address has the role of group leader. The group leader periodically broadcasts a message including the Universally Unique Identifier (UUID). If a node doesn't receive the message from a group leader, it can perceive the network partitioning. On the other hand, if a node will receive the messages from two or more group leaders, the node can recognize the network to be merged. Because this scheme manages the list of all nodes and collects response messages from all nodes, it can easily decide the success or failure of acquiring agreements. The node list can be used to find the conflict addresses when two or more networks are merged. It may take a longer time to allocate an address when the initiator fails to get agreements. An additional weakness is that synchronization of the node lists among nodes is needed, which may break the consistency of the lists. As a result, it fails to guarantee the uniqueness of the assigned address.

Hongbo et al. proposed a conflict free distributed address configuration scheme named Prophet Address Allocation using a function that produces an integer sequence [6]. The address of a MANET node must be unique during its lifetime. They tried to design a function that produces a sequence that satisfies the extremely long interval between occurrences of the same number. The probability of more than one occurrence of the same number in the different sequences must be extremely low. It is difficult to design such a function in distributed manner. The Prophet allocation has obvious advantages: short address allocation time and low communication overhead. If the address length is less than 16 bits, the conflict ratio is too high to use for address allocation in MANET. The authors suggested the address length be 24 bits. However, if the address length is 24 bits, the conflict ratio is also very low in random selection as well in Prophet. The address conflict still occurs even if 24 bits are used. Prophet may need a method to avoid address conflict.

In addition, there are other distributed mechanisms that split the address space to avoid conflict among addresses. They need a method to gather unused addresses to prevent an address leak problem [7][8][9]. The hybrid mechanisms in [10][11] separate a MANET into sub networks. Other researchers have studied address configuration from a different point of view: they suggested solutions to detect and solve the conflict when it occurs with the aid of routing protocols, which do not involve the allocation mechanism [12][13]. The distributed mechanism presented in [14] gives the role of the leader to the new node. The leader allocates an address of (highest known address + 1) to the next new node. The new leader information is announced by flooding packets. A mechanism using a variable length address instead of an IP address is presented in [15]. In this work, it is required to add some fields to all network layer packet headers.

3. Performance Metrics for Address Autoconfiguration Schemes

We defined performance metrics for address autoconfiguration schemes in MANET as follows:

3.1. Uniqueness

Every MANET node must get a unique address for each network interface because the duplicated addresses may cause severe problems in routing. The incorrect information may result in misrouting or failure of service. Therefore, the nodes having duplicated addresses should change their addresses.

The address changing also may cause a break in service and produce incorrect routing information.

Guaranteeing the uniqueness of an allocated address is the most important performance metric because address conflicts affect the entire performance and traffic of the network.

3.2. Scalability

We can consider two factors in terms of scalability: communication overheads and allocation latency. Communication overheads mean the number of packets that is spent for nodes to get addresses. The allocation latency is the waiting time for a node to get an address. A good address autoconfiguration scheme rarely depends on the total number of nodes in the network or the network size.

3.3. Independency of routing protocols

The routing protocols can be classified into two groups: a proactive routing algorithm and a reactive routing algorithm. Address autoconfiguration schemes must operate with both routing protocols for the nodes to join a MANET regardless of their routing algorithm.

3.4. Reusability: Garbage collection and IP leaks

If a MANET node leaves the network, the address of the node must return to the address pool. The centralized mechanism can handle this problem easily, but the distributed mechanism has difficulty. The schemes that do not have any policies for address reuse may suffer an address leak problem. To prevent address leak, garbage collection which requires additional overheads is necessary.

3.5. Availability

Address autoconfiguration schemes must always be available regardless of network status. For example, we can consider the following network status or events: network partition and merger, node mobility, and message loss.

A good address autoconfiguration scheme must operate even if MANET is partitioned or merged. Handling the network merger is more complex because it causes the address conflict problem. The address conflict happens in network merging because address allocation schemes only guarantee the uniqueness in a MANET.

Address autoconfiguration schemes should quickly detect the network merging and address conflict, and then should solve the address conflicts in the merged network. Most protocols introduce a group ID to solve the network partition and merger. To detect the network partition and merger, a leader node in a MANET periodically broadcasts its existence with a group ID or a network ID. If two or more nodes that have a different group ID meet, they can detect the network merging. If a node doesn't receive the message from the leader, it can perceive the network partitioning.

In addition, address autoconfiguration schemes must continue the allocation process when a new node requiring an address and a MANET node participating allocation of address move during the process of address allocation.

A MANET is inherently unreliable, and the transmission error rate is high. The nodes will be abruptly crashed or depart from the network. Some messages may be lost during the address allocation process. Address autoconfiguration schemes should prepare the message loss.

4. Simulation Environments

We simulated four address autoconfiguration schemes: MANETconf [5], Prophet [6], Strong DAD [4], and Zeroconf [1]. To show the results of the uniqueness of allocated addresses for the network with huge nodes, we made a java program. It is hard to simulate with ns-2 in the large network environment.

4.1. Java Simulation Environments

The Prophet and random allocation are simulated in java to compare the address uniqueness in a large network. In java simulation for Prophet, an agent is randomly selected among all nodes in the network when a new node arrives. The number of neighbors is uniformly distributed between 1 and 8. Node mobility was not considered because we believe that the random selection takes the place of the movement effect. After the address allocation for the previous node has been completed, the next new node arrives. All nodes stay in the network until the end of the simulation and the number of nodes always increases.

4.2. Ns-2 Simulation Environments

We used ns-2 simulator of version 2.27 with modified Random Waypoint Models [16][17][18]. The maximum speed of nodes is 5 m/s; the minimum speed of nodes is 1m/s, and the pause time is set at 10 seconds. Table I shows two different simulation

environments. The inter-arrival time of new nodes was uniformly distributed between [0, 10] seconds. To make a well connected network, we added preconfigured nodes before starting the simulations. Network partition and merger were not considered.

TABLE I
Simulation parameters

Parameters	Environment I	Environment II
The number of nodes	35, 40, …, 60	50, 60, …, 90
Preconfigured nodes	30	40
Area	750m x 750m	1000m x 1000m
Simulation time	900 seconds	900 seconds
Address range	1 ~ 65534 (16 bits)	1 ~ 254 (8 bits)
Routing protocol	AODV/DSDV	AODV/DSDV

5. Performance Evaluation

5.1. Uniqueness

The most important metric is the uniqueness of the allocated addresses because address conflicts may cause abnormal behavior in routing protocol and applications.

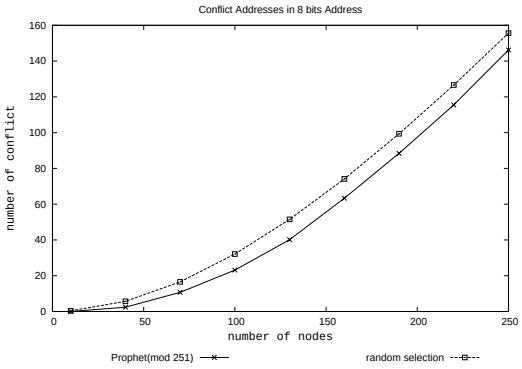
Zeroconf does not allow the conflict of addresses because an address agent node allocates all addresses based on the MAC addresses of the requesting nodes. Strong DAD and MANETconf perform DAD before allocating an address. Strong DAD uses NACK and MANETconf uses ACK as a mean of getting agreements for use of an address.

In contrast, Prophet does not have a method to verify the address conflict. In Prophet, each node has a unique modular function to generate a positive integer sequence.

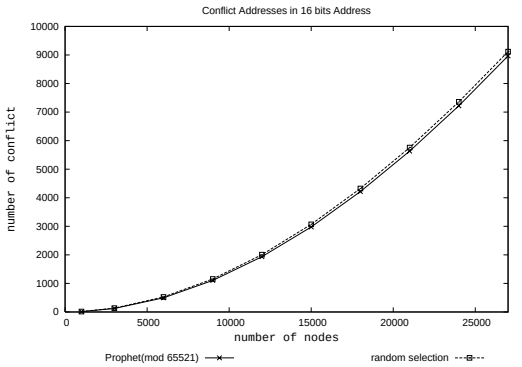
$$address = a + \prod_{i=1}^{k} p_i^{e_i} \bmod R + 1 \quad (1)$$

Here, a is a randomly selected address for the first node. The primes pi satisfy p1 < p2 < … < pk. We use the value k as 209 by their simulation codes, and pk as 1291, the 209th prime number. The R varies from 251 to 65521 and 16777211 according to its address bits: 8 bits, 16 bits, and 24 bits.

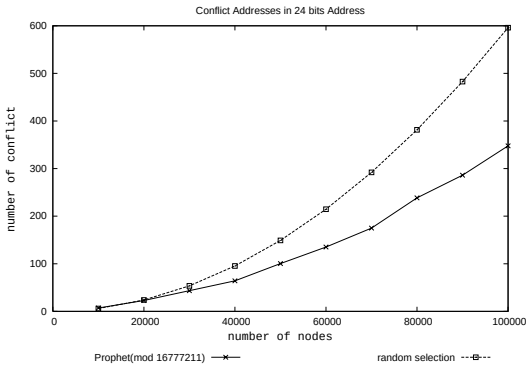
For a node, the function can be expressed as (address = a + C*pi mod R + 1), where C is a constant. It has a form of Linear-Congruential Generator (LCG) used in random number generation [19]. If we use modulus R as a non-prime value such as 2m, the number of conflicts increases because if the prime used for the sequence is a divisor of R, the period of one sequence decreases. The Modulus 251, 65521, and 16777211 are the closest prime numbers to the address ranges, respectively.



(a) Conflict in 8 bits address



(b) Conflicts in 16 bits address



(c) Conflicts in 24 bits address

Figure 1. The number of conflict addresses between Prophet and Random Allocation

Even though Prophet uses the prime modulus for reducing the number of conflicts, the nodes with

duplicated addresses still exist in MANET. We show how many nodes with duplicate addresses exist using our Java Program. We compared random allocation mechanism using random function to Prophet. In random allocation, a new node chooses a uniformly distributed random number less than the address range and takes the number as its address.

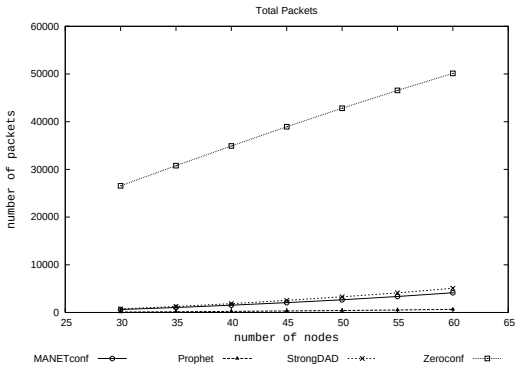
The function in Prophet cannot avoid address conflicts. Each node produces a sequence in the same domain. Two or more of the same numbers can be generated from the multiple sequences. The uniqueness is guaranteed only in one sequence, but it is not guaranteed among multiple sequences. Figure 1 shows the number of address conflicts according to the number of nodes. With the 8 bits address in Figure 1(a), Prophet shows many conflict nodes even though the number of nodes is small. With the 16 bits address in Figure 1(b) and the 24 bits address in Figure 1(c), the conflict ratio decreases but still remains. In addition, Prophet and random selection show similar results. The number of conflicts in Prophet is smaller than that of random allocation. However, when the number of nodes is 100,000, the difference is at most 250.

As a result, Prophet fits for the network with huge address ranges and with small nodes: 24 bits address length and at most 150 nodes, which is a waste of the address space. In such a case, even random allocation has a conflict ratio close to zero.

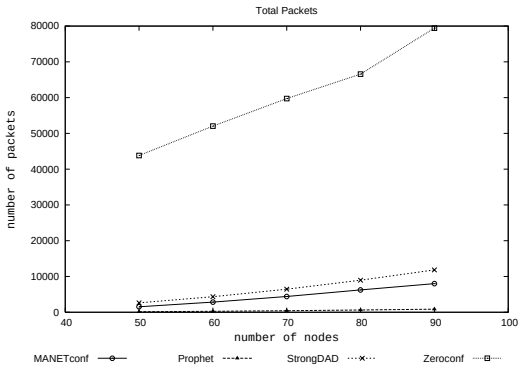
5.2. Scalability

We measured two factors in terms of the scalability metric: communication overheads and allocation latency. Figure 2 shows the total number of packets as the number of nodes increases. Prophet is outstanding in communication overheads because a new node needs to communicate with only its agent in order to obtain an address. MANETconf and Strong DAD show similar results because they use DAD process. Zeroconf needs a lot of messages because it periodically sends the confirm message in order to manage its address pool. The messages are increase as the number of nodes increases. Zeroconf is not suitable for a network with many nodes.

Figure 3 shows an average allocation latency as the number of nodes increases. Prophet also shows a very low latency that is almost constant regardless of the number of nodes. Strong DAD shows a constant line because it must wait for a fixed time i.e., the time it takes for the DAD process to complete three times. Each DAD process needs 1.5 seconds because we define the maximum hop size is 10 and the latency for one hop is 0.15 seconds. Thus, the allocation latency of Strong DAD approaches 4.5 seconds or higher.



(a) Environment I



(b) Environment II

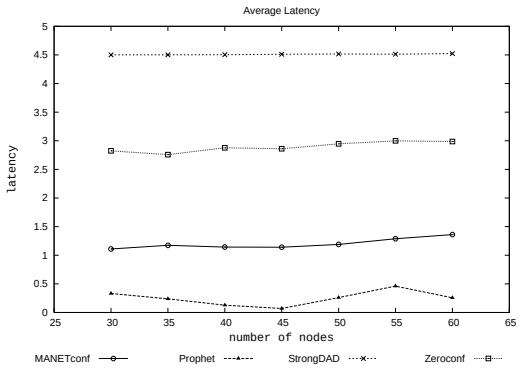
Figure 2. Total packets according to the number of nodes

MANETconf is also not affected by increasing the number of nodes. Even though MANETconf uses DAD process, it always shows better results than Strong DAD because MANETconf performs DAD process one time and resends messages only to nodes that do not respond. On the other hand, Zeroconf is affected by the number of nodes. Figure 3 (b) shows increasing latency according to the number of nodes. A new node must wait for the second flooding message from the leader node in order to obtain an address. The leader node periodically floods the message every 2 seconds. Therefore, allocation latency of Zeroconf is higher than 2 seconds.

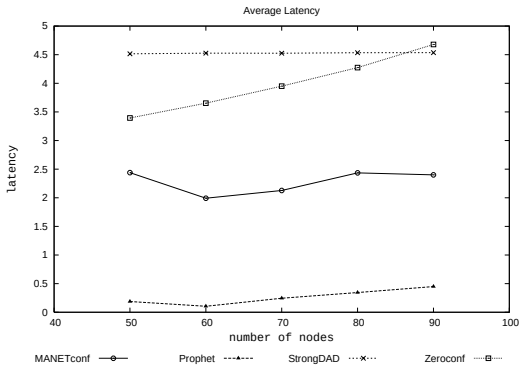
5.3. Independency, Reusability, and Availability

All Protocols operate with any routing protocol. However, it will be better if Zeroconf and MANETconf operate with proactive routing protocols because they should also maintain information of all MANET nodes.

In Zeroconf, it is easy to handle the address pool since it is a centralized mechanism. Its reusability of



(a) Environment I



(b) Environment II

Figure 3. Average latency according to the number of nodes

addresses is also high. It does not require an additional garbage collection process and does not allow address leak. MANETconf and Strong DAD are also good at reusability because they use DAD process before address allocation if they minimize their message loss. Prophet provides reusability using the rotating sequence. It assumes that a MANET node stays at a MANET for a while and leaves the network. Prophet implicitly returns the addresses to the pool. However, if the length of sequence is short, the address conflicts occur in the network. It does not check whether an address is being used or not.

In the aspect of availability, Strong DAD does not provide a solution for network partitioning and merging. MANETconf and Prophet have the same approach that uses a group ID to detect them. Zeroconf also provides a similar approach such that its leader node periodically checks all nodes and their addresses. The period may affect the network performance because if it is too short, the traffic becomes highly congested. In Table II, we arrange the performance evaluation of each scheme in terms of 3 metrics: independency, reusability, and availability.

TABLE II
Performance Evaluations in terms of three Metrics

Performance Metric	MANETconf	Prophet	Strong DAD	Zeroconf
Independency of Routing Algorithms	Yes	Yes	Yes	Yes
Reusability	High	Low	High	High
Availability	Medium	Medium	Low	Medium

6. Conclusion

We compared four address allocation schemes in terms of uniqueness, scalability, independency, reusability, and availability using simulation. The centralized scheme is simple and shows good results with respect to uniqueness and reusability. It has a weak point in scalability because it produces a lot of messages. In addition, the leader node may be a single point of failure. Strong DAD and MANETconf may take a long time to obtain an address because they use DAD process before allocation. Prophet results in low communication overhead and low allocation latency. However, it doesn't guarantee the uniqueness of allocated addresses.

We need to compare these four protocols with other distributed or centralized schemes that are not evaluated in this paper.

7. Acknowledgment

We thank Woohyuk Jang and Bioinformatics and Software Systems Lab. for helping our simulation study.

8. References

[1] M. Günes and J. Reibel, "An IP address configuration Algorithm for Zeroconf. Mobile Multi-hop Ad-hoc Networks," Proc. of the International Workshop on Broadband Wireless Ad-Hoc Networks and Services, September 2002.

[2] S. Toner and D. O'Mahony, "Self-Organizing Node Address Management in Ad-hoc Networks," Springer Verlag Lecture notes in Computer Science 2775, Springer Verlag, 2003, pp 476-483.

[3] Y. Sun and E. Belding-Royer, "Dynamic Address Configuration in Mobile Ad hoc Networks," UCSB Technical Report 2003-11, June 2003.

[4] C. Perkins, J. Malinen, R. Wakikawa, E. Belding-Royer and Y. Sun, “IP Address Autoconfiguration for Ad Hoc Networks,” draft-ietf-manet- autoconf-01.txt, November 2001.

[5] S. Nesargi and R. Prakash, “MANETconf Configuration of Hosts in a Mobile Ad Hoc Network,” Proc. of IEEE INFOCOM, June 2002.

[6] H. Zhou, L. Ni, and M. Mutka, “Prophet Address Allocation for Large Scale MANETs,” Proc. of IEEE INFOCOM, March 2003.

[7] A. Misra, S. Das, A. McAuley, and S. K. Das, “Autoconfiguration, Registration and Mobility Management for Pervasive Computing,” IEEE Personal Communications (Special Issue on Pervasive Computing), Volume 8, Issue 4, August 2001, pp. 24-31.

[8] M. Mohsin and R. Prakash, “IP Address Assignment in a Mobile Ad Hoc Network,” IEEE Military Communications Conference (MILCOM), October 2002.

[9] T. Ramakrishnan, “A Protocol for Dynamic Configuration Of Nodes in MANETs,” Master’s thesis, Computer Science, University of Texas at Dallas, August 2002. (advisor: Ravi Prakash)

[10] K. Weniger and M. Zitterbart, “IPv6 Autoconfiguration in Large Scale Mobile Ad-Hoc Networks,” Proc. of European Wireless 2002, Feb. 2002.

[11] K. Manousakis, A. McAuley, R. Morera, and J. Baras, “Routing Domain Configuration for More Efficient and Rapidly Deployable Mobile Networks,” Army Science Conference, Dec. 2002.

[12] K. Weniger, “Passive Duplicate Address Detection in Mobile Ad hoc Networks,” Proc. of IEEE WCNC 2003, March 2003.

[13] N. Vaidya, “Weak Duplicate Address Detection in Mobile Ad Hoc Networks,” Proc. of ACM MobiHoc, June 2002.

[14] P. Patchipulusu, “Dynamic Address Allocation Protocols for Mobile Ad Hoc Networks,” Master’s thesis, Computer Science, Texas A&M University, August 2001. (advisor: N. Vaidya)

[15] J. Boleng, “Efficient Network Layer Addressing for Mobile Ad Hoc Networks,” Proc. of 2002 International Conference on Wireless Networks (ICWN), June 2002.

[16] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, “A Performance Comparison of Multi-Hop Wireless Ad Hoc Routing Protocols”, Proc. of the Fourth Annual ACM/IEEE Inter-national Conference on Mobile Computing and Networking, October 1998, pp. 85–97.

[17] J. Yoon, M. Liu and B. Noble, “Sound Mobility Models,” Proc. of ACM MobiCom, September 2003.

[18] J. Yoon, M. Liu and B. Noble, “Random Waypoint Considered Harmful,” Proc. of IEEE INFOCOM, April 2003.

[19] R. Jain, “The Art of Computer Systems Performance Analysis,” John Wiley & Sons, 1991, pp 439-440.