A Duplicate Address Resolution Protocol in Mobile Ad Hoc Networks

Chunhung Richard LIN and *Guo-Yuan Mikko WANG*

Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung 804, TAIWAN
E-mail: lin@cse.nsysu.edu.tw, mikko@wmail.cse.nsysu.edu.tw

Abstract − In an IP-based network, automated dynamic assignment of IP addresses is desirable. In most of wired networks, a node acquiring an IP address relies on centralized server by using the Dynamic Host Configuration Protocol (DHCP) [5]. But this DHCP-based approach cannot be employed in a Mobile Ad Hoc Network (MANET) due to the uncertainty of any centralized DHCP server. That is, a MANET may become partitioned due to host mobility. Therefore, there is no guarantee to access a DHCP server. A general approach is to allow a mobile host to pick a tentative address randomly, and then use a duplicate address resolution (DAR) protocol to resolve duplicate addresses. In this paper, we propose a novel distributed dynamic host configuration protocol designed to configure nodes in MANET. Our protocol not only can detect the duplicate address, but also can resolve the duplicate address. We show that the proposed protocol works correctly and is more general than earlier approaches. We also propose an enhanced version of DAR scheme, which solves the situations of duplicate MAC address in the same time or some devices without MAC addresses. A new approach we propose in this paper can make the nodes in MANET provide services to other networks and avoid packets be delivered to an incorrect destination.

Keywords:

Mobile ad hoc network, Duplicate address, IP-based network, network configuration, Service

1. INTRODUCTION

In IP-based network, auto-configuration is a desirable goal. However, if two or more hosts have the same IP address, it will cause data to be delivered to the wrong node and then lead to some insecure situations. In traditional wired networks, a node can acquire an IP address rely on the centralized server by using the Dynamic Host Configuration Protocol (DHCP) [5]. However, a mobile ad hoc network (MANET) is grouped by mobile nodes. A special characteristic of these nodes is mobility which leads to frequent and unpredictable topology changes. Therefore, it is difficult to guarantee to access a DHCP server in MANET.

In this paper, we present a distributed dynamic host configuration protocol designed to assign IP address to the nodes in MANET. It is a DAR (Duplicate Address Resolution) protocol. The goal of DAR is to assign a unique IP address to each node, and to solve the problem of packets which are delivered to incorrect destination nodes if two nodes happen to have chosen the same IP address. Our DAR protocol resolves duplicate addresses and prevents packets from being interpreted by the incorrect destination node.

The rest of this paper is organized as follows. In Section 2 we summarize the related work. Section 3 describes the network model. The basic idea of the DAR is presented in Section 4. The DAR itself is explained in Section 5 and 6. Section 7 shows the correctness of our protocol. Section 8 presents an enhanced version to avoid IP address and MAC address duplicated in the same time or some devices that do not have MAC addresses, and proposes a new approach to make the nodes in MANET that can provide services correctly. We discuss the issues of packets reaching an incorrect node in Section 9. Finally, conclusions are presented in Section 10.

2. RELATED WORK

A. Weak DAD

Weak DAD [20] was proposed as an alternative approach for duplicate address detection. Weak DAD mechanism guarantees that packets are not delivered over time to two different nodes even if both are assigned the same address. Weak DAD randomly assigns an IP address to a node which joins in a MANET and picks a unique key value (e.g., MAC address) as the identification of this node. Then, Weak DAD makes use of the normal routing protocols, e.g., link state routing or dynamic source routing, to update the routing table. The node can look up

the records in the routing table to detect duplicate address.

Consider two nodes, D and D', in MANET are assigned the same IP address, a, and they are placed at different network partition. Weak DAD can guarantee that packets being sent by a given node, S, to the particular address a are not delivered over time to the two different nodes D and D' when both partitions merge. That is, it does the same as before the partitions merge. Although it can avoid packets to be delivered to the incorrect node, it must need to detect the duplicate address and either D or D' must give up the address a. However, in this protocol, there is no discussion about how to do the duplicate address resolution. Therefore, we need a concrete approach to solve these related problems.

B. MANETconf

MANETconf [11] mechanism can ensure that no two nodes in the MANET acquire the same IP address. The approach used by MANETconf is to force the node joining in MANET, called the *requester*, to acquire an IP address from the *initiator*. The initiator acts as a proxy for the *requester* to get an IP address and then packets can be routed to the requester. The requester cannot become a formal host and begin to work until all nodes in MANET accept its join request.

In MANETconf, the *initiator* plays an important role. It needs to assign IP addresses to *requesters*, and detect message loss, node crash, etc. Therefore, the *initiator* must be reliable. Detecting network partitioning and merging in MANET is performed by periodic broadcasting a verifying key. It also provides a scheme for duplicate address resolution.

There still exist some problems to be solved in MANETconf. A new node needs to get an IP address from *initiator* before it can work in MANET. This process will waste time and may cause undesired delay for the urgent data. Additionally, there are many broadcast operations in the process of assigning and releasing the addresses, and the detection of network partitioning and merging. This will result in severe overheads to the network. In a special case, the packet may be delivered to an incorrect destination node. It is insecure to the receiver and may cause receiver to be down. We will discuss this insecure situation later. Therefore, we need a solution to prevent from transmitting data to the incorrect destination, to decrease the times of broadcast operations, and to guarantee the reliability of IP address request.

3. NETWORK MODEL

For simplicity, we only consider a stand-alone MANET. It means this MANET does not interconnect with the other networks. Even if the MANET can communicate with other networks, our protocol can still work correctly. In this protocol, we will focus on IP address assignment. Therefore, the other configurations such as gateway, netmask, DNS, etc., are not our concern. Our protocol can work on both IPv4 and IPv6.

Hosts are roaming around the network. The network topology, therefore, is changed frequently and unpredictable. The partitioning and merging processes will appear. Hosts in MANET are classified into two classes, i.e., *gentle* nodes and *rough* nodes:

Gentle nodes: When a node departs from MANET, it will send a message to notify the other nodes. It makes others be able to reuse this IP address. This kind of nodes is called the gentle nodes.

Rough nodes: A node can not send a message to notify the other nodes before it leaves the network. Such a node is called a rough node. This may be because the node is crashed or the network is partitioned. Therefore, its IP address cannot be free for reuse.

Consider IP address to be a 32-bit integer. We let the smaller integer to have higher priority. For example, 192.168.0.1 has higher priority than 192.168.0.2. Similarly, we can define the priority of MAC address in the same way. This definition will be used to resolve the contention for an IP address.

The broadcast in this paper has some different. It's because we focus on the IP address assignment and not limit all hosts in the MANET must stay in the same subnet, so the traditional broadcast can not be received by a host which is in the different subnet. It makes us must redefine the broadcast. The broadcast in this paper is replaced by multicast or unicast to all hosts except the sender.

4. BASIC IDEA

In our protocol, we do not need an agent to act as the *initiator* in MANETconf when a new node joins the network to acquire an IP address. The new node randomly picks an IP address by itself. Every node needs a key to identify itself in the network. For example, the ID may be the combination of IP and MAC address. This can avoid the problem of two nodes having the same IP address. Every node maintains a table to record the IDs of all nodes in the network. The table is called *address table* and is shown in Figure 1. Every nodes will discard any packets which ID is not in their address table except control message. By exploiting this address table, a node can detect duplicate address, partitioning and merging, and then prevent packets from being delivered to an incorrect destination node.

IP_A	MAC_A
IP_B	MAC_B
...	...

Figure 1: Address table

5. DUPLICATE ADDRESS RESOLUTION PROTOCOL

The following are the objects and control messages in our protocol; we will explain the usage of them:

tx_timer: This message is started by the control messages which need response be send. If this timer expires, the control messages will retransmit to the nodes which not reply.

rtx_retry: This is the maximum number of times a node attempts to send control message. If all the attempts result in a failure, upon exceeding this threshold, the node will give up retransmitting and cleanup the departed nodes.

address_update: This message includes an ID to notify the receiver to add it into address table. Then the receiver will reply an ACK.

table_update: This message contains sender's address table to notify the receiver to merge it with its own one together. Then the receiver will reply an ACK.

duplicate_address: This message is used to ask the receiver to give up the authority of IP address it using. There is an advice IP address which is not in sender's address table piggyback with *duplicate_address* to prevent the receiver from picking a duplicate address again. The node received this message should make a response.

table_probe: This message is used to ask the number of the entries in receiver's address table. The ACK it returns will contain the response. This message also can be a confirm message to check the receiver exist or not.

address_cleanup: There is an ID piggyback with this message to notify the receiver to remove it from address table. This message has no response.

6. DUPLICATE ADDRESS RESOLUTION

A. MANET initialization

When the first node walks into the spacious area, it will pick an IP address randomly and record the entry (IP address, MAC address) into its address table. Then, it finishes its process to join to MANET. It needs not exchange any packets, because there are no other nodes in this spacious area.

B. The join of new node to MANET

Now, a new node X walks into MANET. X picks an IP address i randomly and records the entry (i, MAC_X) into its address table. No matter what messages X transfers, this message will be received by the neighbors of X. If a node Y hears a message from X, Y will look up its address table to check if (i, MAC_X) exist or not. If this entry does not exist, Y will add (i, MAC_X) to its address table. Then Y will unicast a control message, *table_update*, piggybacking with its own address table to X. Upon receiving Y's address table, X merges the table with its own one together.

After Y sends *table_update* to X, it also multicasts the *address_update* to all nodes in its address table except X and itself. This makes these nodes update their individual address table. Each receiver must reply an ACK to Y for this address_update. If Y can not receive all ACKs before *tx_timer* timeout, it will resend *address_update* to those nodes whose ACKs are missing. If some ACKs still cannot be received after *rtx_retry* times of retransmissions, Y will send *address_cleanup* message to notify the other nodes to remove these nodes from their own *address table*. After these procedures are done, X can joins to MANET.

If Y looks up its address table and finds that Z has used this IP address i already. In this time, Y will send *table_probe* message to X and Z; then X and Z must reply how many entries in their own address table to Y. The number of entries in the address table of X is smaller than Z, because node Z has already existed in the MANET and X is joining to the MANET, therefore, X has only one entry (i, MAC_X). After all, Y will send *duplicate_address* and piggyback an advice IP address which is not in its address table to X to avoid X picking a duplicate address again. All procedures above must run again. Repeat this process until X gets a legal IP address.

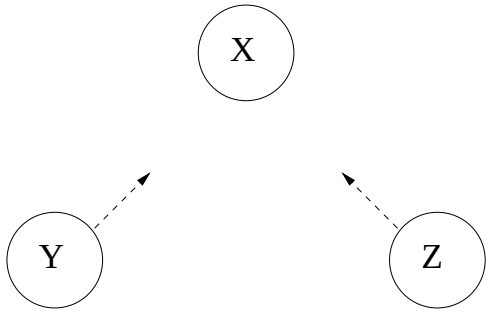
C. Departure of a gentle node

When a gentle node X is going to depart from the network, it will multicast *address_cleanup* message with its (IP address, MAC address) to all nodes in its address table. Nodes which receive this message must remove the entry of X from their address table. The IP address used by X can be reused again.

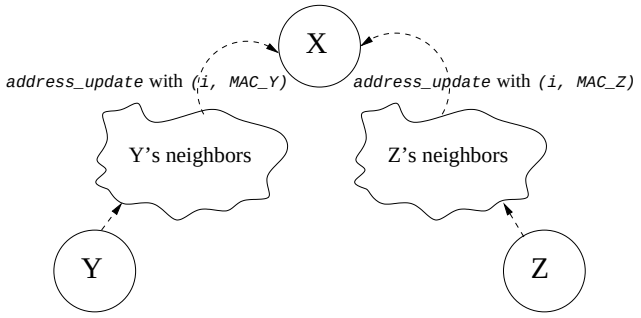
D. Two new nodes pick the same IP address simultaneously

Consider three nodes X, Y, and Z. X has already existed in the MANET. Y and Z just join to the network. If Y and Z pick the same IP address i which is not used by any nodes. Neighbors of either or both of Y and Z will send *address_update* to the other nodes. Both messages contain different information, i.e., (i, MAC_Y) and (i, MAC_Z). When they reach to X, X can detect the duplicate address. Figure 2 shows the

scenario.



(a) Y and Z just join to the MANET.



(b) The neighbors of Y and Z allow them to assign IP address i and send *address_update* to the other nodes for them. X may receive *address_update* messages from neighbors of Y and Z. X can detect the duplicate address occurring.

Figure 2: Two nodes pick the same IP address simultaneously

There are two situations which cause the duplicate address:

(i) Two nodes just join to the network;

(ii) One of both nodes already departs from the network. But it does not notify the others. This may because the node is crashed, the network is partitioned, or message is lost, etc.

First, X can send a message to confirm if both nodes are alive. If two nodes just joins to the network, X will compare their MAC address priority.† X will send a *duplicate_address* message to the low priority node to ask it to pick another IP address again. In addition, X will send *address_cleanup* to all to make them remove this entry of the low priority node from their address table. But X will send *address_update* to all to let the high priority node become a legal node of IP address i.

† Because MAC address is 6 bytes, we can compare the byte one by one to determine their priority. If the first byte of them is the same, we can compare the next one and repeat this action until we can determine the priority of them. Only in the worst case we need to compare all 6 bytes.

E. Partitioning and Merging

A MANET may split into multiple partitions at any time due to the node mobility. However, the partitions may be merged together at another time. In our protocol, we use an unique value to identify each partition. This ID (identified key) can be the highest priority entry in the address table. Every node maintains its *address table*. They can easily get the ID of the partition they stay in. There will be no cost to get the partition ID.

(i) Partitioning

We assume that the MANET splits into two partitions. The one which keeps the highest priority node has the same ID without changing. We name it as Partition 1. The other partition whose ID must be changed is called Partition 2.

If a new node X walks into Partition 2, a neighbor node Y of X sends *address_update* for X. According to ACK responses, Y will clean the nodes staying in Partition 1 from its address table, and sends *address_cleanup* to the others in Partition 2. The nodes receiving this message will update their address table. The entries which are removed contain the ID used before partition occurred. This lets nodes in Partition 2 to know they become a new partition. At this time, the IP address of the highest priority will become the new ID in Partition 2. The operation will be performed at all nodes in Partition 2 and the new ID will be got.

The occurrence of network partitioning is not only detected at the time of a new node joining the network. We can also detect this from the routing table. When the node which has the highest priority becomes unreachable, it means partition occurs. Therefore, we need not do any extra detection to detect partition occurred or not; it can be discovered in finite time.

(ii) Merging

Let two nodes X and Y staying in different partitions. Once X and Y can communicate with each other, they will discover the partition merge together by the ID they exchanged. Then, they will flood their own *address table* with *table_update* to make all nodes in this new created partition update their address table. Duplicate address may be found. The number of TCP connections can be used to resolve it. The node which has less TCP connections must give up its IP address and pick a new one.‡

‡ We use TCP connections to determine the authority of IP address. This is because a TCP connection will disrupt when at least one of both ends changes its IP address. If we

F. Message losses

The nodes which receive the *address_cleanup* need not reply the ACK. Due to message loss, a node Y may not receive the *address_cleanup* message about a node X which has already departed from the network. This situation is the same as the rough node case. Y can remove X from its address table if X becomes unreachable in the routing table.

Consider a scenario that a node Y misses the *address_cleanup* and let a node X still in its address table. If a node which picks the IP address as the same one of X to join to the network. This will be rejected by Y. That is, the IP address is hold and cannot be reused. Consider the following two situations:

(i) Y receives any messages from the new node directly.

When Y receive any messages from a new node Z, Y will find that its IP address has already been used by X. Subsequently, Y will send *table_probe* to X and Z. Because X has already departed from the network, therefore, X will not reply any message to Y for *table_probe*. After *rtx_retry* times retries, Y can ensure that X has already departed and cleans X from its address table.

(ii) Y receives the *address_update* message from the other nodes.

This situation is the same as the case (ii) in Section 6-D. When Y receives an *address_update* for the new node Z, Y will find that the IP has been assigned already. Y will send a confirm messages to X and Z first to confirm if X and Z exist or not. Eventually, Y removes the entry of X because X has already departed.

7. THEOREM AND PROOF

We assume there is a routing table in each host which records the route to each reachable host. A routing protocol will automatically update the routing table every *n* seconds.

Theorem 1: *This configuration protocol can ensure address table be updated and synchronized in finite time.*

Proof:

There are some conditions that address table are updated:

(i) node receives an *address_update* message.

(ii) node receives a *table_update* message.

(iii) node receives an *address_cleanup* message.

(iv) routing table is updated.

Condition (i) occurs when a new node joins and its neighbor multicasts this message into MANET. A new node joining and MANET partitioning or merging will make condition (ii) happen. When a gentle node departs from MANET, duplicate address is detected, and MANET partitioning or merging happens, condition (iii) will occur. Condition (iv) will happen every *n* seconds.

We can not predict when some node will join or leave the MANET to make the network topology change. This time of when the topology will be changed is not predictable. But routing table will be updated in finite time. So the address table can be updated by using routing table. And all nodes in the same partition can keep their address tables synchronized as we describe in Section 6, even if message loss occurs. Thus the address table can be updated and synchronized in finite time.

Theorem 2: *The packets will never be delivered to an incorrect destination node.*

Proof:

Every node in our MANET has an address table, that can help node to discard those packets which are delivered to incorrect destinations. When a node receives a packet, it would check the ID inside first, if the ID matches with an entry in its address table, then the packet will be processed; else the packet will be discarded.

The ID we used in normal DAR is the pair of *(IP address, MAC address)*, it will work in most situations. But if a special situation happens, i.e., two hosts have the same IP address and MAC address simultaneously, we will use the enhanced DAR which will be described in the next section to solve it. Therefore, any packets will never be delivered to an incorrect node.

Theorem 3: *If MANET is partitioned or merged, it can be detected in finite time.*

Proof:

The MANET partitioning will be known when the highest priority entry is removed from the address table. The MANET merging will be detected when the address table is exchanged. Both actions are based on address tables. Because the address table will be updated in finite time (from theorem 1), if MANET is partitioned or merged, it can be detected in finite time.

---

choose the node which has less TCP connections to give up the authority of IP address, it can decrease the effect. If two nodes have the same number of TCP connections, we can determine the authority of IP address by the priority of their MAC address.

8. ENHANCED DUPLICATE ADDRESS RESOLUTION

A. IP address and MAC address are the same simultaneously

We propose an enhanced version of DAR to solve a very special case which is IP address and MAC address are the same simultaneously for two nodes. We make every node generate a UUID when it picks an IP address randomly and add this UUID to its *address table*. After this, the new ID becomes *(IP address, MAC address, UUID)* and this format of *address table* is illustrated in Figure 3. Therefore, we can completely avoid the situation that IP address and MAC address are the same simultaneously.

The probability of the IP address and MAC address to be the same simultaneously is very small. If we add UUID to *address table*, the size of *address table* will grow up with the number of nodes joining to the MANET. We use the enhanced version of DAR only in this kind of situations. In most of situations, the normal version of DAR is enough to resolve the IP address assignment.

IP_A	MAC_A	UUID_A
IP_B	MAC_B	UUID_B
...	...	...

Figure 3: *address table* format of enhanced DAR.

B. Service on the nodes

If the nodes in the MANET want to provide some services, we must ensure the packets exchange between client and server won't be delivered to incorrect destination nodes.

Consider the case of the MANET which is no longer stand-alone and interconnects with the other networks by Internet Protocol. Thus, the services provided by a node in the MANET can be accessed by remote users in the other networks. If the MANET is divided into two partitions, we assume two nodes are within the different partitions, respectively, and they have the same IP address, i, and the same service (e.g., TELNET). Take Figure 4 as an example. We assume both A and B have the same IP address, i, and provide the TELNET service. How does the MANET distinguish a TELNET request from a remote user to a server with IP address i? Of course, it is a severe problem. Actually, there is no any solution which has ever been proposed to solve this problem. But our protocol can solve it completely.

We use a powerful AP (Access Point) to solve this problem. The behavior of APs which use our protocol are like the nodes in the MANET we describe before. They will use control message (e.g., *address_update*) to exchange data and maintain their own address table. The different is that AP has two address table to records internal and external entry, respectively, i.e., *address table*$_i$ and *address table*$_e$:

address table$_i$: The nodes which can communicate with AP_n will be recorded in AP_n's *address table*$_i$. This table makes AP_n action like a filter, it can discard the broadcast messages from network which destination address not in its communication range and ignore the verbose messages.

address table$_e$: This table contains all *address table*$_i$ of APs in the network. AP can avoid the duplicate address occurs in its network by checking this table. If two nodes are within the different partitions, respectively, and they have the same IP address, AP will send *duplicate_address* message to the node which use this IP address later to ask it to choose another IP address. It can prevent the exist service connection not to be broken.

Every APs has an unique manufactory key, therefore, we use this key to verify the packets go through AP. AP will fill the key into the *source key field* of packet which sends from the node in the MANET and ckech the *destination key field* of packet which receives from other networks. Only when the key field matches with AP's key or keep blank or has no key field then the packet can go through AP into its MANET. We describe this approach below:

In Figure 4, we assume AP_1 and AP_2 are in the same network. $MANET_n$ is an independent partition and can not communicate with other MANETs directly. If the nodes in it want to interconnect with other networks, the packets must go through AP_n.

(i) duplicate address in the same network

If the node A walks into $MANET_1$ and picks an IP address i which not be used, the AP_1 will record this entry in its *address table*$_i$ and update its *address table*$_e$. Then, AP_1 will send *address_update* to ask AP_2 to add this entry in its *address table*$_e$. After all, if the node B walks into $MANET_2$ and picks the same IP address i, AP_2 will discove the duplicate address and ask B to choose another IP address. How about A and B choose the same IP address in the same time? We make the one which has less TCP connections to choose another IP address.

(ii) packets exchange

We assume A and C has the same IP address i, and B has the IP address j. Because AP_1 far from AP_3 (e.g., one in Taiwan, another in USA) and not in the same network, therefore, they can not discover the duplicate address.

Now, a host D wants to connect to i to get a service, it will send a request message to create a

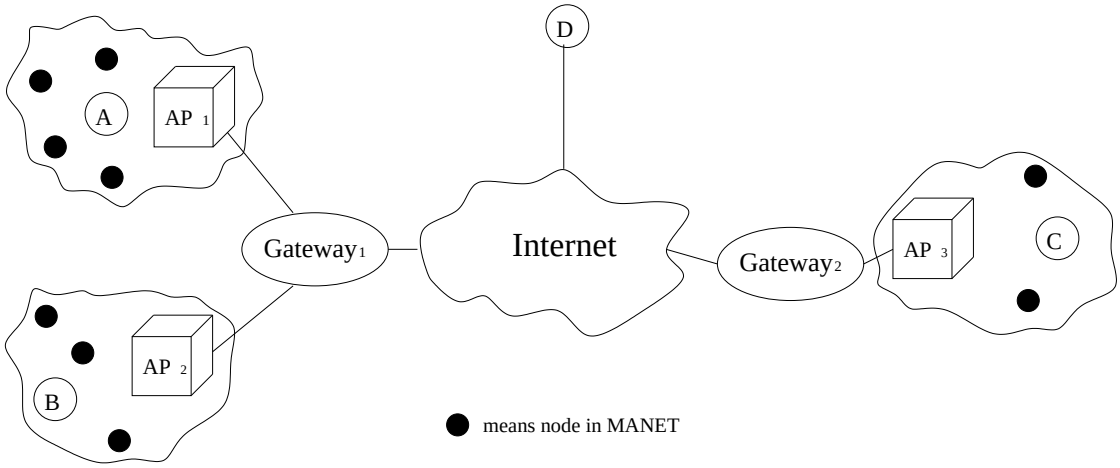


Figure 4: Service on the nodes.

connection. We assume the routing routine in Internet makes this request message be delivered to A. When this request message arrive in $Gateway_1$, it will be broadcast to the network behind $Gateway_1$. Both AP_1 and AP_2 will receive this broadcast message, but only AP_1 will accept it. It's because the destination address of this request message is not in $address\ table_e$ of AP_2. After AP_1 received this request message, it will check the destination address and key field of this request message first. Because the destination node in its MANET and it has no key field, therefore, AP_1 will unicast this request message to A. After received request message, A will response a reply message. When this reply message goes through AP_1, AP_1 will append a key field to this reply message and fill its key into *sourec key field* and keep *destination key field* be blank. After connection created, D only need to exchange *sourec key field* and keep *destination key field* from A and fill them into the outgoing packet to A. Even if the routing routine in Internet makes a mistake to send the packet which should be delivered to A be delivered to C, the AP_3 will discard this packet because the error key. And AP_3 can ask C to choose another IP address to avoid the above situation happen again.

If client and server both reside in MANET, our approach also can work correctly. The APs of them will fill the correspond key into key field. They only need to exchange *sourec key field* and *destination key field* which received from other and fill them into the outgoing packet to other. Then, the packets will never be delivered to incorrect node. So, our approach can guarantee that the packets exchange in a created connection will always be delivered to the correct destination.

9. DISCUSSIONS

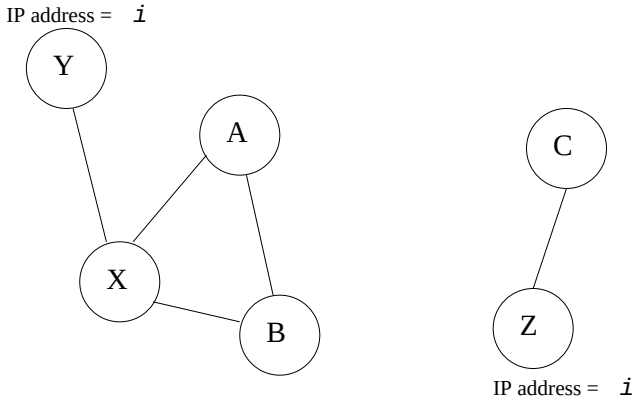
Consider a MANET shown in Figure 5(a). There are two partitions in the network. Both cannot communicate with each other. Partition 1 contains A, B, X, and Y and Partition 2 contains C and Z. Even though Y and Z are assigned the same IP address *i*, they can work correctly within each partition.

Now, we assume that mobility makes both partitions merge to one. Before the duplicate address is resolved, if X intends to send packets to Y, then it is possible for Z to receive packets from X. This is illustrated in Figure 5(b). It makes errors occur. Z may be crashed because of these errors. It must be avoided.

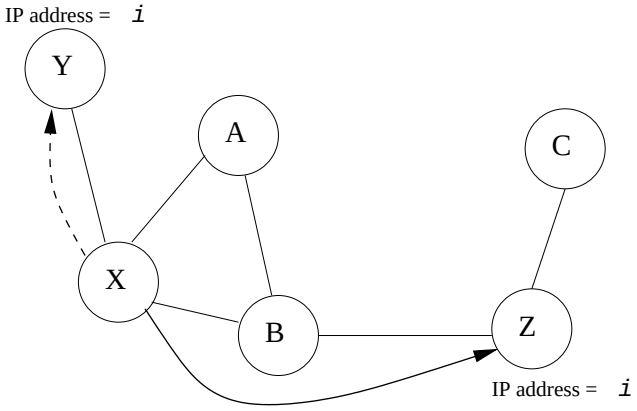
As presented in Section 2-B, MANETconf unfortunately cannot solve this problem of partition merging. However, we can completely avoid this situation in our protocol because every node has a unique ID. If a packet from X reaches the undesired destination Z, Z just checks the destination ID and then drops this packet. Here we assume the destination must always double check the ID (i.e., the pair of MAC and IP address) in the packet to make sure if the packet should be accepted.

10. CONCLUSIONS

In this paper, we present a distributed dynamic host configuration protocol for DAR (Duplicate Address Resolution). The pair of IP and MAC addresses is exploited as a unique ID to avoid the duplicate address occurring. Our protocol works correctly with any routing protocols, such as DSR, AODV, etc. The enhanced version can further solve the situation of both IP and MAC addresses to be the same simultaneously. Our protocol also can work no matter whether the MANET is partitioned or merged and whether control messages are missing. We also



(a) Y and Z have the same IP address, but two partitions can not communicate with each other.



(b) X want to send a packet to Y but the packet be received by Z during two partitions merge together.

Figure 5: Packet exchange during partition merge.

propose a new approach to make the nodes in MANET provide services to other networks and avoid packets be delivered to incorrect destination. Packets can always be delivered to the intended destination finally.

REFERENCES

[1] N. Asokan and P. Ginzboorg, ‘‘Key agreement in Ad Hoc Networks,’’ *Computer Communications,* vol 23, no. 17, pp. 1627-1637, November 2000.

[2] R. Chandra, V. Ramasubramanian, and K.P. Birman, ‘‘Anonymous Gossip: Improving Multicast Reliability in Mobile Ad-Hoc Networks,’’ in *Proceedings of the 21st IEEE International Conference on Distributed Computing Systems (ICDCS 2001),* Mesa, Arizona, April 2001, pp. 275-283.

[3] S. Cheshire, ‘‘IPv4 Address Conflict Detection,’’ draft-cheshire-ipv4-acd-03.txt (expire June 9, 2003), Internet Engineering Task Force, Zeroconf Working Group, December 2002.

[4] S. Cheshire and B. Aboba, ‘‘Dynamic Configuration of IPv4 Link Local Addresses,’’ draft-ietf.zeroconf-ipv4-linklocal-03.txt (expire December 22, 2001), Internet Engineering Task Force, Zeroconf Working Group, June 2001.

[5] R. Droms, ‘‘Dynamic host configuration protocol,’’ March 1997.

[6] S.K.S. Gupta and P. Srimani, ‘‘An Adaptive Protocol for Reliable Multicast in Mobile Multi-hop Radio Networks,’’ in *Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA99),* New Orleans, February 1999, pp. 111-122.

[7] Y.-C. Hu and D.B. Johnson, ‘‘Caching strategies in on-demand routing protocols for wireless ad hoc networks,’’ in *ACM International Conference on Mobile Computing and Networking,* August 2000.

[8] D.B. Johnson and D.A. Maltz, ‘‘Dynamic Source Routing in Ad-Hoc Wireless Networks,’’ T. Imielinski and H. Korth, editors, *Mobile Computing,* 1996, Kluwer Academic Publishers.

[9] S. Nesargi, R. Prakash, ‘‘DADHCP: Distributed Dynamic Configuration of Hosts in a Mobile Ad Hoc Network,’’ Tech. Rep. UTDCS-04-01, University of Texas at Dallas, Department of Computer Science, 2001.

[10] S. Nesargi, R. Prakash, ‘‘Issues pertaining to service discovery in mobile ad hoc networks,’’ in *ACM Workshop on Principles of Mobile Computing,* August 2001.

[11] S. Nesargi, R. Prakash, ‘‘MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network,’’ *Proceedings of IEEE INFOCOM 2002.*

[12] T. Ozaki, J.B. Kim, and T. Suda, ‘‘Bandwidth-Efficient Multicast Routing for Multihop, Ad-Hoc Wireless Networks,’’ in *Proceedings of IEEE INFOCOM 2001,* 2001, pp. 1182-1191.

[13] C. Perkins and E. Royer, ‘‘Ad Hoc On-Demand Distance Vector Routing,’’ *Proceedings of the 2nd IEEE Workshop on Selected Areas in Communication,* February 1999, pp. 90-100.

[14] C.E. Perkins, J.T. Malinen, R. Wakikawa, E.M. Belding-Royer, and Y. Sun, ‘‘IP address Autoconfiguration for Ad Hoc Networks,’’ draft-ietf-manet-autoconf-01.txt, Internet Engineering Task Force, MANET Working Group, July 2000.

[15] Y. Rekhter, B. Moskowitz, D. Karrenberg, G.J. de Groot, and E. Lear, ‘‘Address Allocation in Private Internets, RFC 1918,’’ Internet Engineering Task Force, Network Working Group, February 1996.

[16] G. Ricart and A.K. Agrawala, ‘‘An Optimal Algorithm for Mutual Exclusion in Computer networks,’’ *Communications of the ACM,* vol. 24, no. 1, pp. 9-17, January 1981.

[17] W.R. Stevens, *TCP/IP Illustrated,* Volume 1. Addison Wesley, 1994.

[18] C. Schurgers, G. Kulkarni, and M.B. Srivastava, ‘‘Distributed assignment of encoded MAC addresses in sensor networks,’’ *Proceedings of ACM MobiHoc 2001.*

[19] S. Thomson and T. Narten, ‘‘IPv6 Stateless Address Autoconfiguration,’’ RFC 2462, Internet Engineering Task Force, Zeroconf Working Group, December 1998.

[20] N.H. Vaidya, ‘‘Weak Duplicate Address Detection in Mobile Ad Hoc Networks’’, *Proceedings of ACM MobiHoc 2002.*