Routing in MANETs with Address Conflicts

Kwan-Wu Chin, Darryn Lowe
Telecommunications Information Technology Research Institute
University of Wollongong
Northfields Ave
Wollongong, NSW, Australia, 2522
{kwanwu, darrynl}@uow.edu.au

W.H.O Lau
Department of Computer Science
Curtin University of Technology
GPO Box U 1987
Western Australia
lauhow@cs.curtin.edu.au

Abstract

Mobile Ad-Hoc Networks (MANETs) are dynamic and multi-hop in nature. As nodes continually join and leave the MANET, managing the problem of address conflicts is particularly challenging. In the past, researchers have gone to great lengths to ensure that nodes are assigned unique addresses and various protocols and policies have been designed to resolve address conflicts. In this paper, we argue that current solutions, originally designed for static wired networks, put unnecessary stress on the dynamic operation of a MANET. To solve this problem, we present a MANET that can continue to operate even when there are conflicting addresses. Unlike previous solutions, our technique does not break applications by requiring nodes to renumber. Further, the overheads introduced by traditional address allocation and maintenance protocols are removed. All these improvements are effected by introducing of a new routing sub-layer that enables a reactive routing protocol to route packets through a MANET that is experiencing address conflicts. This routing sub-layer provides features such as conflict avoidance forwarding, conflict notification, and enhanced address resolution.

1 Introduction

Addresses are important in computer communications, as both identifiers and locators of nodes. An important criterion when allocating addresses is that each host must be assigned an unique address. Typically, there are four ways that a host might obtain an unique address. The first is to ask a central server, such as a dynamic host configuration protocol (DHCP) server [5], where the server is responsible for ensuring that all allocated addresses are unique. Secondly, IPv6 hosts can acquire an address using IPv6's stateless auto-configuration mechanism [14] where the assumption is that the on-link router has been configured or managed to obtain one or more unique prefixes that are usable by hosts. Thirdly, a host may use a link-local (LL) protocol [3] to generate itself an address and carry out duplicate address detection (DAD) to ensure that the chosen address is unique. However, LL protocols assumes all nodes are of link-local scope and are thereby available to defend their addresses at all times. Finally, a node may rely on the user to ensure that the assigned addresses are unique.

A mobile ad-hoc network (MANET) is different to what has been assumed by conventional address allocation mechanisms. Firstly, the multi-hop aspect of MANETs violates the link-local scope assumed by schemes such as LL [3] and DHCP [5]. DHCP can be used in multi-hop environment if nodes employ relay agents [20] in order to relay a client's DHCP messages to a server located elsewhere on the wired network. However, in order for relay agents to be employed successfully, there must exist an automatic configuration protocol to enable the relay agents to learn the DHCP server's address quickly so they do not need to rely on inflexible manual configuration. In addition, deploying relay agents would result in redundant DHCP messages being forwarded to the server since multiple relay agents could be within a DHCP client's transmission range. In the worst case, without a coordination protocol, each relay agent would forward redundant copies of the DHCP request to the DHCP server.

In a MANET, since nodes join and leave as they please, there is frequent merging and partitioning. When a MANET partitions, the addresses of the nodes that have left can be reclaimed and then reassigned to new nodes. However, before the address can be reused, the address allocation entity must be certain that the original node has truly left the MANET and will not be rejoining anytime soon. In the case where the original node rejoins the MANET after the address is reused, an address conflict will occur and one of the conflicting nodes will have to renumber to obtain a new address. The impact of this problem can sometimes be mitigated by delaying the reuse of addresses until the current address space has run out. Compared to partitioning, merging is a harder problem to solve because, before the address conflict can be resolved, the MANET needs to detect the conflict.

Thus far, only a handful of address allocation mechanisms (e.g., [17][8][19][16]) have considered the aforementioned issues. Moreover, none of these works consider mechanisms that would avoid renumbering by allowing nodes to continue to function with their existing addresses in the event of an address conflict. To this end, we outline a set of novel features that are embodied in an routing sub-layer and show how these features help a reactive routing protocol route packets through a MANET that is experiencing conflicts. As a result, the nodes in our MANETs are not required to have unique IP or MAC addresses. A node can simply randomly generate itself an IP address and starts communicating without having to check if its address is unique.

A common solution to the aforementioned problems is to associate unique identifiers with each address [16][19][17], hence enabling nodes to distinguish IP or MAC addresses using the identifier. The identifier itself can be a randomly generated number or a cryptographical key. Once the conflicts have been identified, the corresponding nodes can then be notified to renumber. Our routing sub-layer improves on this and, later, we will show how these nodes can retain their original IP addresses.

The remainder of this paper is organized as follows. Section 2 presents what happens when there is either an IP or MAC address conflict. We then present our novel routing sub-layer in section 3 followed by a demonstration of a reactive routing protocol that utilizes our sub-layer to enable routing in a MANET experiencing address conflicts in section 4. In section 5 we qualitatively compare our scheme to existing solutions and section 6 concludes this paper.

2 Problem Description

Address conflict is a serious problem in MANETs that could lead to misdirected packets and ambiguous routes. That is the reason why address allocation protocols put a lot of effort into making sure allocated addresses are unique and no two hosts have the same address. However, address allocation protocols are continuously challenged by the dynamic nature of MANETs where nodes join or leave randomly hence making verification of addresses very difficult. This paper considers routable and link-local address conflicts. For routable addresses, we consider IP (either IPv4 or IPv6) and for link-local addresses we consider MAC addresses. MAC and IP address conflicts are discussed independently since the fact that they are on different layers of the protocol stack means that the case where both are in conflict can be dealt with by solving each separately.

2.1 IP Conflict

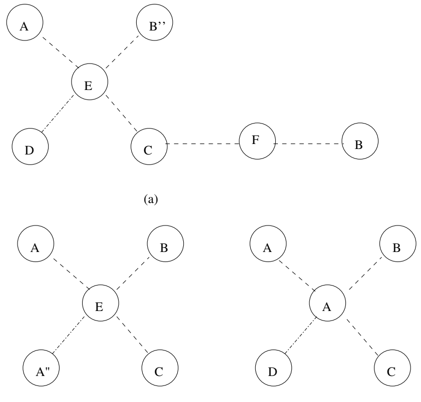
Applications require a host's IP address to be fixed for the duration of the connection and any change in IP address may result in having to re-establish the connection. This is disruptive to the end user and, especially in a dynamic environment such as a MANET, may make some applications inoperable.

Aside from breaking applications, conflicting addresses also cause problems to routing protocols. This is because a routing protocol cannot distinguish between nodes with an identical IP address. As a result, packets may be lost if they are routed to the wrong node or are routed off the correct path. Therefore, as route updates from HELLO and routing messages play a crucial role in determining what a router knows about the network, it is important that these messages identify nodes with duplicate IP addresses so that routers can operate correctly.

Consider, for example, that node-A has established a connection to node-B in Figure 1(a). Say, at some time later, node-B" shows up as node-E's neighbor. Node-E will then think that node-B has become one of its neighbors and will update its routing table promptly to take advantage of the new "shorter" route. However, given that it is the wrong node-B, any packets sent over the new route may be lost.

Another example is the case where a router has two neighboring nodes with the same IP address (see Figure 1(b)). Packets will only be received by the neighboring node for which the current node has a MAC address mapping. Unfortunately, this may or may not belong to the correct next-hop node. In the worst case scenario, if the node receiving the message has a routing entry that points to another node with the same IP address as the intended target, the packets will be routed to the wrong node.

Figure 1(c) shows a scenario where there are two nodes



(a)

(b)

(c)

Figure 1. Address conflict examples. (a) conflicting nodes two hops away. (b) two neighbors having the same address (c) a neighbor having the same address as the node itself.

with address A. This means that these nodes will end up connecting back to themselves when they try to connect to each other.

2.2 MAC Conflict

Nodes may also have conflicting MAC addresses [8][17]. When they do, conflicting nodes who are on the same link will erroneously pick up each other's frames. For example, in Figure 1(b), the two neighboring nodes of node-E, A and A", have the same MAC address but different IP addresses, IP1 and IP2 respectively. A packet sent by node-E to the address IP1 will be received by both nodes A and A" because node-E has an IP-to-MAC mapping corresponding to either the MAC address of node-A' and A". Node-A will therefore happily accept the packet but Node-A" could either forward or drop the packet.

Another possibility is that node-A" does not have a route to IP1, so it would generate a route error message to node-E. If the original sender was node-C, then the route error message will cause node-C to reconstruct the route, and, unbeknown to node-A", node-C's connection has been torn down. Another severe problem, as pointed out in [17], is when node-A moves away and node-A', who has a routing entry pointing to a different node with address IP1, joins the network. Here, packets will be routed to the wrong node which will result in the effective hijacking node-B's communications.

3 Addressing Sub-Layer

The problems presented in the previous section require a solution that mitigates the impact of conflicting nodes without incurring the costs of renumbering. The need for such a solution is proven via the analysis presented in Appendix A that shows how frequent partitioning and merging makes multiple address conflicts almost unavoidable in a large MANET. However, all existing attempts use explicit signaling that creates unnecessary overheads as well as disrupting active connections in the network. For example, existing methods such as MANETconf [8] call for the node with the lowest number of connections to renumber whereas [9] selects the node with the lowest identifier. By avoiding such renumbering completely, our solution is simultaneously more efficient and fair to all nodes.

In our solution, we propose a set of features that are embodied within a routing sub-layer. Our sub-layer has multiple functionalities that can be broadly classified into two key parts. The first is an *enhanced address resolution protocol* that prevents contamination of routing tables when neigboring nodes have conflicting addresses. It also maintains IP to MAC address mappings. The second part is a *conflict avoidance forwarding* scheme that works together with a routing protocol to build a forwarding table that allows packet delivery even when multiple neighbors share the same IP or MAC address.

3.1 Functionalities

3.1.1 Choosing an IP Address

The sub-layer's first task when a node boots up is to choose an IP address, which, for illustration purposes, we assume is IPv4 address. In this paper, we consider only MANETs that are unable to obtain a global address (i.e. they do not have connections to network infrastructure and a private address space is used [13]).

By default, the sub-layer uses the private address spaces specified in [13], of which, for illustrative purposes, we select 192.168/16. Given that we assume no subnets in MANETs, we will use the entire 16-bit address space. The address allocation process follows the link-local protocol [3] whereby the sub-layer uses a pseudo-random number generator with a uniform distribution to select an address in the range of 192.168.1.0 to 192.168.254.255.

3.1.2 Unique Identifier (UID)

Unlike existing schemes [16][19][17] that used UIDs only to detect conflicting nodes, our routing sub-layer extends them by using the same information to guide routing protocols decisions.

It is critical to note that the UID, as a routing-only entity, is independent and distinct from that IP and MAC addresses. Specifically, the UID has three principal distinguishing features.

Firstly, unlike MAC and IP addresses, UIDs are limited to resolving address conflicts that would otherwise break routing protocols. Thus, since the UID is not used to identify session end-points, conflicts between UIDs can be resolved without requiring the session to be torn down.

Secondly, the UID is flexible as it is not used beyond the routing sub-layer. This flexibility means that the specifics of the UID, such as the size of the address space and how it is calculated, can be customised for different MANET scenarios. For example, the size of the UID could be set to give an appropriate address space. Another example could see the UID be a cryptographic key that ensures correct identification of an end-node.

Finally, the UID can be used to resolve address conflicts in MANETS that implement both flat and hierarchical addressing schemes since it is independent of any addressing structure. Indeed, a key feature is that the relationship between the routing sub-layer and the routing protocol is that the sub-layer is agnostic as to how layer three addresses are defined.

A simple implementation would see a node generate itself an UID, along with an IP address, upon boot-up. For example, one embodiment of an UID could be a hash of the interface's IP and MAC addresses. In the case of multi-homed nodes, each interface would have its own UID. All UIDs belonging to a node are included in that node's HELLO messages so that the node's neighbours can learn of each other's respective UIDs. A node could also learn of other nodes' UIDs by monitoring a reactive routing protocol's route request and reply processes.

3.1.3 Neighbor Discovery and Resolution

A routing protocol uses our sub-layer to maintain a list of neighboring nodes unlike conventional routing protocols that run their own neighbor discovery service. This neighbors list is hidden from the routing layer and helps prevent contamination of a node's routing table.

An important service provided by our sub-layer is the replacement of the address resolution protocol (ARP). The motivation for replacing ARP is due to our previous study [4] where we found ARP [12] to be detrimental to the performance of MANETs. We found that ARP's behavior of caching MAC addresses without checking link reliability, as well as its inability to purge its cache of transient or stalled links, was detrimental to the operation of MANET routing protocols. Therefore, our sub-layer improves upon ARP [12] by caching only information belonging to stable neighboring nodes, and provides a new mechanism for dealing with conflicting neighboring nodes. A stable neighbor is defined as one with a strong signal over a period of time. To determine the stability of a neighbor we use the same metrics as those defined in the associative based routing protocol [15].

3.1.4 Neighbor Table and Conflict Detection

Unlike existing works [17][19] that aim to detect conflicts within an entire MANET, our sub-layer is only concerned with nodes that are one-hop away. This simplifies the address conflict detection process since our sub-layer makes use of existing HELLO messages transmitted from neighboring nodes in order to detect address conflicts.

Figure 1(b) shows node-E discovering that it has two nodes with a identical addresses, namely A and A". Table 1 shows the table constructed by node-E after exchanging HELLO messages with its neighbors. In this example, node-E discovers that it has two different sessions, denoted as UIDs A_1 and A_2, to two different nodes that share the same IP address but have different MAC addresses.

Table 1. Neighbor table corresponding to example in Figure 1(a).

UID	Neighbor IP_addr	MAC_addr
E_1	192.168.199.200	a2-a0-00-00-0c-09
A_1	192.168.111.1	e2-fe-2e-92-3c-39
A_2	192.168.111.1	00-53-45-00-00-00
B_1	192.168.121.2	00-10-00-00-45-ee
C_1	192.168.100.100	aa-ee-00-00-ef-1a

3.1.5 Next-Hop Forwarding Table

Once the routing protocol has decided on the next-hop node for a given destination, and discovers that forwarding packets to this destination involves routing through neighboring nodes with address conflicts, it to register the next-hop node's MAC address with the sub-layer, provided the next-hop node's MAC address is unique. This ensures that the sub-layer knows which neighboring nodes have a correct routing table that leads to the intended destination.

Table 2. A forwarding table example.

Dest. Node	Dest. UID	Next-Hop Node's IP	Next-Hop UID	MAC Conflict?
192.168.1.1	E_1	192.168.254.1	F_1	No
192.168.99.100	B_1	192.168.254.1	Z_1	No
192.168.100.100	A_1	192.168.200.99	G_1	Yes

Table 2 shows an example forwarding table used by our sub-layer when determining the next hop neighbor. An example is as follows. When the sub-layer receives a packet headed to destination 192.168.1.1 with the next hop field set to 192.168.254.1, it looks up Table 2 and finds more than one neighboring nodes assigned the same IP address. Therefore, our sub-layer finds the MAC address that maps to UID F_1 and forwards the packet to the next hop node that owns that MAC address.

3.1.6 Preventing Route Contamination

Existing protocols such as AODV rely on HELLO messages to discover neighboring nodes. However, in MANETs using our scheme, the processing of HELLO messages is conducted by our sub-layer and the routing layer is not aware of its neighbors because we want to prevent route contamination. Figure 2 shows a MANET with an on-going session between nodes A and E. Let's say that node-E" shows up next to node-C at a later time and if the sub-layer does not suppress node-E"'s HELLO messages then node-C will think that node-E has moved closer to it. Hence, it concludes that packets should be sent directly to node-E instead of going through node-D.

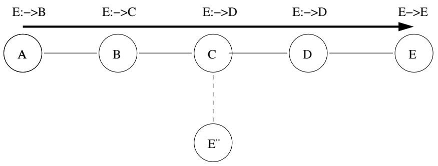


Figure 2. Example of a session contamination.

On the other hand, our sub-layer simply records node-E"'s information so that the routing protocol will not know of node-E"s existence. If node-E shows up, the sub-layer will update its forwarding table and informs the routing protocol that node-E is now a neighbor, given that node-E's UID matches.

3.1.7 Link Breakage

The sub-layer sends a signal to the routing layer indicating which neighboring node has died when a link breaks. Upon receiving the signal, the routing layer may start sending route error messages to nodes that are using the link and remove the corresponding route entries. In the same manner, our sub-layer goes through its forwarding table and searches for entries that use the broken link and removes them.

3.1.8 Conflicting MAC Addresses

Thus far we have assumed that nodes have unique MAC addresses and that the sub-layer is able to distinguish the next hop node unambiguously. However, as reported in [8], MAC address collisions are possible. To overcome MAC conflicts, our sub-layer does the following. When our sub-layer detects that two nodes have an identical MAC address it will include the intended node's UID within the transmitted frame and use a broadcast to deliver the message. A receiving node then checks to see whether the next-hop UID in the shim header matches its own UID. The frame is accepted if the UID matches, otherwise it is discarded. Note that broadcast is only used by nodes that have neighbors with conflicting MAC addresses, otherwise we use unicast.

3.1.9 Destination Node with Address Conflict

Establishing a route to a destination node that is experiencing address conflict is difficult because there is no guarantee that a node will be communicating with the correct target node unless the source node knows of the target node's UID beforehand, in which case intermediate nodes can determine the correct destination node to forward a route request to.

We do not provide a solution for how an end-host identifies whether it has connected to the correct host because this information can only be verified by upper layer protocols/applications. However, our sub-layer does provide feedback to a source node during the route request and reply process by including a list of UIDs for the target address in the shim-header, indicating to the initiating host that the destination address is experiencing conflict. The sub-layer then offers these UIDs to the routing protocol which may then pass them to the application layer for processing. It is up to the application to decide how it wants to process the UIDs, for example it may go through each UID iteratively until the correct destination is found.

3.2 Shim Header Structure

Table 3 shows the structure of the shim header. The header is carried in all packets to protect against the unexpected appearance of conflicting source and/or destination nodes that may confuse the sub-layer, and also to solve MAC address conflicts.

4 Conflict-Avoidance Forwarding

Without loss of generality, we will describe our modifications to Ad-hoc On-Demand Distance Vector (AODV). Application of our sub-layer to other routing protocols is left for future work. In the next section, we will show how the AODV routing protocol [11] can be modified to employ our sub-layer to establish connections and route packets in MANETs with conflicting addresses. In our examples nodes sharing the same address will be marked as A, A", A"', etc. Also, unless specified otherwise, all nodes have unique MAC addresses.

Field	Size (bits)	Description
Flag	4	This flag indicates to intermediate routers which destination UID is currently being used to identify the target host.
nDestUIDs	4	The number of UIDs in the field *DestUIDs*. This field is incremented whenever an intermediate node adds an UID to *DestUIDs*. There will be more than one UIDs if there are multiple destinations sharing the same IP address
DestUIDs	32	This is filled by the source node and intermediate nodes and contains UIDs associated with the target IP address.
SrcUID	32	This field indicates the source that originated a given packet.
NxtUID	32	The next hop node that should be receiving a given packet. This field is filled only when the next hop node does not have an unique MAC address.

Table 3. Shim Header Structure

4.1 AODV: Overview and Modifications

AODV [11] is a reactive routing protocol that relies on request (RREQ) and reply (RREP) messages to create routes. When a node receives a RREQ or RREP it, creates a route entry that stores the next-hop node leading to the appropriate source or destination node. Associated with each route entry is a destination sequence number that indicates whether a given RREQ or RREP is the latest message sent by the source or destination node. If a message with a lower or equal sequence number is received, it is discarded. Further, each node maintains a precursor list that, by recording the nodes using a given route, allows the determination of what neighboring nodes should receive a route error message if the link to a given destination fails.

In addition to implementing the sub-layer, both AODV and the network stack require some modifications. These are:

1. *Comparing addresses and UIDs*. A node must be able to compare IP addresses as well as UIDs. For example, if a node has knowledge of a node called node-E, and it receives a RREQ for node-E, it needs to compare the UIDs of both potential node-Es and, if the UIDs match, then the node knows that the RREQ is referring to the node-E it knows of.

   In addition, our addressing sub-layer will return more than one UIDs if there are multiple destination with the same IP address. Therefore, this list of UIDs needs to be propagated to higher layers for processing.

2. *Marking of route entries with conflicting addresses*. AODV is notified of a route conflict by our sub-layer when it is passed a packet with a conflicting address. AODV uses this notification to flag a given route entry as experiencing a conflict and to request the sub-layer to record the MAC address corresponding to the neighboring node that sent or is about to receive the packet.

3. *Handling forwarding of packets to conflicting addresses*. Once a route entry is identified as having a conflict, the routing algorithm must route the packet to the sub-layer for forwarding. For example, node-C's route entry could be $C : - > C(conflict)$ meaning that the next hop node to node-C is itself, but, given that the route entry has been marked as having a conflict, packets are passed to the sub-layer for processing. Another example is $E : - > C(conflict)$, where the next hop node to node-E is through node-C. Given that node-C is experiencing conflict, the packet is routed to the sub-layer where the sub-layer determines the correct node "C" to send the packet to.

4. *Replacement of ARP*. We need to remove ARP and replaces it with our sub-layer. This is required so that we can control what information gets passed to the routing layer and what information is used when forwarding packets to a next-hop node.

So far, we have presented our sub-layer's features and identified what modifications are required in order to forward packets in MANETs with address conflicts. In the following sections, we demonstrate how AODV makes use of our sub-layer to establish and route packets in various address conflict scenarios. Note that our examples serve to highlight how AODV employs our sub-layer and are not intended to be comprehensive.

4.2 Scenario 1: Conflicting intermediate nodes

Example-1

The first example concerns the establishment of a connection through two nodes with address C. Figure 3(a) shows a network where node-B has two neighboring nodes with address C. Assume that node-A wants to establish a connection to node-E, so it starts by broadcasting a route request (RREQ). The broadcast will be received by node-B, which records a route back to node-A before broadcasting the message to its neighbors, i.e., the node Cs. Both node Cs will record the route back to node-A through node-B before broadcasting the RREQ. The broadcast results in each of the node Cs receiving another copy of the RREQ, but, since they have seen the RREQ before, the message will be discarded.

On the other hand, given that node-D has not seen the RREQ, it will simply record the route back to node-A through node-C instead of returning a RREP message. This behaviour has two differences when AODV is using our routing sub-layer. Firstly, node-D's sub-layer will suppress node-E's HELLO messages. Therefore, node-D will not

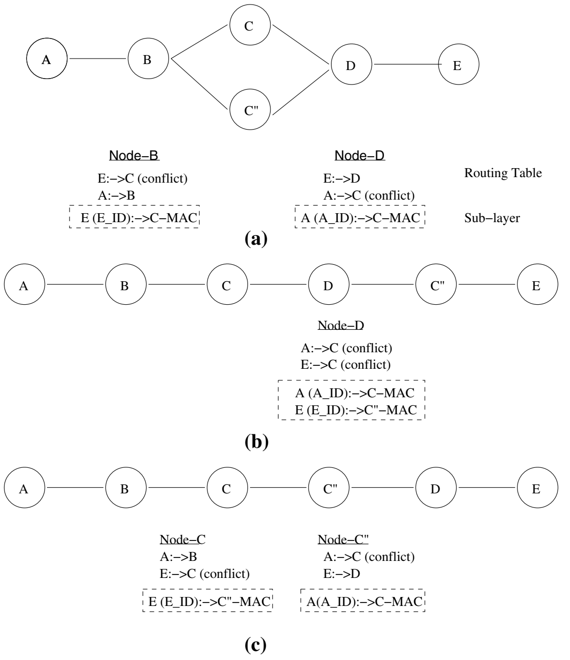


Figure 3. Establishing a connection in a MANET with conflicting intermediary nodes, and the resulting routing and forwarding tables after a route request and reply process. The notation $X :-> N$ is read as destination X's next hop node is node-N, and $E(E\_ID) :-> C\_MAC$ means destination E with UID E_ID's next node is the node with MAC address C_MAC. (a) Both nodes B and D have conflicting nodes with IP address C (b) Node D has two conflicting neighbors that cannot hear each other. (c) Both conflicting nodes C are next to each other but node B and D can only hear one of them.

know of node-E's existence and will not generate a RREP. Indeed, the RREP can only come from the requested destination (unless there is already a route entry installed from an existing session). Secondly, node-D is notified by its sub-layer that its next-hop to node-A, i.e., node-C, is in conflict. Node-D resolves this conflict by discarding all but the most reliable node-C's MAC address.

Finally, the RREQ is received by node-E where it confirmed that it is the intended recipient. Then, after creating a route entry to node-A through node-D, a RREP will be sent back to node-A. The RREP from node-E will first be received by node-D, where a route entry is created to node-E. As remarked before, node-D will then find that the route to node-C is in conflict, so it will request the sub-layer to forward the packet to the previously registered hop, namely node-C. Note that, after a given time, Node-C" will delete the route to node-A since it has not received a RREP.

Node-C records the route to node-E through node-D before sending the RREP onwards to node-B. Then, when Node-B records the next-hop to node-E to be node-C, it will discover that the address C is in conflict and will inform its sub-layer to registers the best node-C's MAC address as the next hop node leading to node-E. Node-B then sends the RREP to node-A and thereby completes the route establishment process.

Example-2

Figure 3(b) shows a slightly different topology wherein the node-Cs are separated by node-D and thereby hidden from each other. In this example, the RREQ will be forwarded to node-E as in the previous case. However, when the RREQ reaches node-D, given that there is a conflict, it will need to inform its sub-layer to record the last hop node, i.e., node-C, that sent it the RREQ message. The RREQ is then broadcast again and will be received by node-C", which will simply records the route back to node-A through node-D before rebroadcasting the RREQ itself.

On the return path, similar to the RREQ stage, node-D will ask its sub-layer to record the last hop node, i.e. node-C", that sent it the RREP. This will create a route entry to node-E through node-C that is marked as being in conflict. To transmit the RREP onwards, the message will be passed to node-D's sub-layer for forwarding. As node-D's sub-layer is aware of the two node-Cs, and knows which node-C is the correct next-hop node, it will forward the RREP onwards. Upon receiving the RREP message, node-C records the next hop node to node-E as node-D and is unaware that its address is in conflict. Node-C then sends the RREP message to node-B where a route entry for node-E is created before being transmitted to node-A, completing the route establishment process.

Example-3

Figure 3(c) shows two node-Cs side-by-side. Here, the RREQ and RREP have to pass through two conflicting nodes. Initially, the RREQ process will occur as before, with each node recording the next-hop neighbor leading to node-A. However, after node-C" receives the RREQ, it will discover that the next-hop to node-A appears to be itself. To solve this, node-C" will mark the route entry as in conflict and informs its sub-layer to register node-C's MAC address as the next hop node to node-A.

On the return path, node-C" will look up its routing table and, since the next-hop is in conflict, will pass the RREP to its sub-layer for forwarding. The sub-layer will look up the next hop node, which in this case is node-C, and will send the frame to node-C's MAC address. Then, node-C will inform its sub-layer to record the next hop node to node-E as node-C" before forwarding the RREP onwards to node-A.

4.3 Scenario 2: Conflicting source or destination nodes

Source

In the example Figure 4, an intermediate node has the same IP address as the sender, node-A. Neither of the conflicting nodes is aware of the other's existence. Here, assume that node-A wants to establish a connection to node-E, so it sends a RREQ for node-E. Upon receiving the RREQ, node-C's sub-layer informs AODV that the route to node-A is in conflict. This causes node-C to create a route entry to node-A that is marked as having a conflict. Further, it asks its sub-layer to register node-B as the next hop node to node-A (with UID A_ID) before broadcasting the RREQ onwards.

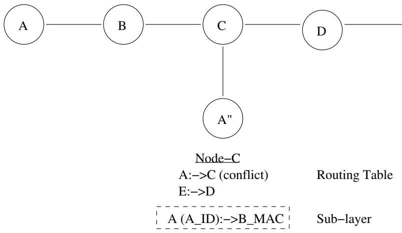


Figure 4. Conflicting Sources

The broadcast from node-C is received by both nodes A" and D. Node-A" determines that the RREQ is not destined for it after comparing UIDs, so it creates a route entry to node-A and marks the entry as experiencing a conflict before broadcasting the RREQ onwards. Finally, the RREQ will arrive at node-E where a route entry to node-A is created before a RREP is sent to node-A.

The RREP will be received by node-D at which time a route entry to node-E is created before the message is forwarded to node-C. Node-C then looks up node-A's next hop node and, given that node-C has previously registered node-A's next hop node as node-B, will forward the message to node-B for processing. The route establishment will then be completed with a broadcast to node-A.

Destination

The destination node could also experience conflict, such as is shown in Figure 5 if node-A tries to communicate with a conflicting node such as node-E. As per section 3.1.9, we assume that the application's policy is to establish a route to each UID until it finds the correct host.

Once node-A's routing protocol learns of node-E's UID from the application it sends out a RREQ with node-E's UID. After node-C receives the RREQ, it will simply create a route entry for node-A and rebroadcast the RREQ message onwards. AODV is not aware of the existence of the conflicting node-E" because our sub-layer has suppressed its HELLO messages. However, node-C does add node-E"'s UID to the route request's shim-header in order to inform node-A of the conflict.

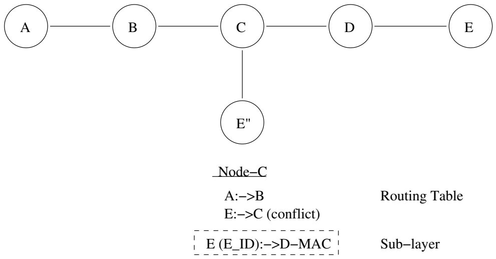


Figure 5. Conflicting destinations

The broadcast is received by both nodes D and E" where they create a route entry to node-A through node-C before broadcasting the message onwards. Upon receiving the RREQ, node-E matches UIDs, finds that it is the intended destination, and proceeds to send node-A a RREP. To do this, node-E's sub-layer copies the contents of the shim header from node-A's RREQ into the RREP. Once the RREP is received by node-C, it is informed by its sub-layer that address E is in conflict, so it asks its sub-layer to register the route to node-E through node-D. It also checks whether the conflicting node-E's UID has been included in the shim-header before forwarding the RREP to node-B, which then forwards it onwards to node-A, completing the process.

4.4 Scenario 3: Conflicting sender and receiver nodes

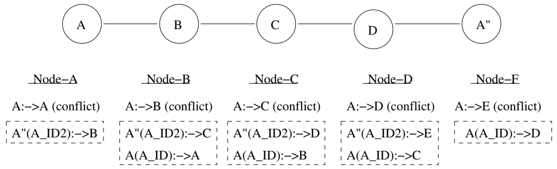


Figure 6. Conflicting sender and receiver nodes result in all nodes along the path having to ask their sub-layer to maintain a mapping to each of the conflicting nodes.

A complicated situation is when both the sender and receiver are in conflict. Figure 6 shows two nodes with address A. As before, assume that node-A wants to establish a connection but, this time, to node-A" with UID ID_A2. Note that, at the connection initiation stage, node-A does not know that the destination shares the same IP address.

Node-A sends a RREQ as before and the intermediate nodes install the appropriate route entries to node-A. When node-A" receives the RREQ, it recognizes that address A is in conflict and that the next-hop node to node-A with UID A_ID2 is node-D. After registering node-A's information and next-hop as node-D, node-A" marks the entry as in conflict and adds its own address to the *DestUIDs* field of the shim header. Node-E then sends a RREP to node-D. On the return path, the RREP message will cause the intermediate nodes' sublayers to mark address A as in conflict and register a separate next-hop node to either node-A or A". For example, node-C's sub-layer will set node-A" and A's next hop as nodes D and B respectively. Upon receiving the RREP, the routing daemon at node-A notes that its address is in conflict. This enable packets destined to node-A" to be routed to the next hop node-B instead of being routed back up the stack. Note that the result of having conflicting source and destination nodes is that every sublayer on the route needs to maintain a forwarding entry. Hence, for large scale MANETs it is advisable that a progressive renumbering process takes place to minimize the number of states present at intermediate nodes.

5 Evaluation

As our sub-layer is best evaluated through implementation, efforts towards a real-world test are underway. In order to justify this on-going implementation effort, we present a qualitative comparison to the signalling overheads incurred by existing methods as well as results from simulations. The conclusion is that our sub-layer is superior for solving address conflicts in MANETs.

5.1 Qualitative

5.1.1 Address Allocation

Table 4 shows a summary of existing solutions' address allocation process, in particular when a new node joins a MANET. From Table 4 we see that all schemes require the transmission of messages and some form of delay whilst waiting for an address. The broadcast schemes are the worst because a node needs to check its address against all nodes in the MANET, and wait for k tries to ensure the chosen address is unique. The conflict free addressing scheme is the best scheme where a new node only needs to ask a neighboring node for a seed that will be used to generate an address. Similarly, a node using a split address scheme [7] only needs to ask one of its neighbor to allocate it a block of addresses by asking the neighbor to split its address space. However, if the neighbor has no free addresses left, the neighbor has to ask its neighbors, and so forth, meaning at worst it could query the entire MANET for a free address block.

A node using our address sub-layer is not required to generate any messages nor wait for an address to be allocated. This is because when a node boots up it randomly chooses an address and starts communicating immediately, hence our scheme does not cause additional signaling overheads and incurs no waiting time due to explicit messaging. Instead, each packet inherits a minimal 104 bits shim-header, less than 1% overhead for a 1500 bytes packet.

Schemes	Number of Messages	Latency
Broadcast [10][8]	O(N)	O(kxN)
Conflict Free [22]	O(1)	O(1)
Leader-Based [16][9]	O(1)	O(D)
Split Address [7]	O(M)	O(N)

Table 4. Comparison of Join Processes. N is the number of nodes in a MANET, D is the diameter of a MANET, k is number of times a node executes DAD, M is a node's degree, i.e., number of neighbors.

5.1.2 Conflict Resolution or Nodes Renumbering

Existing schemes incur a varying cost that either impacts the whole MANET or, in the best case, just the conflicting nodes. In [9], conflicting nodes belonging to the MANET with a smaller network identifier are forced to renumber. Weniger [19] uses a similar method that requires the node that has held a given address for the shortest period or the host with the lowest number of TCP sessions to renumber. Any such approach that requires renumbering will be disruptive to existing connections. Further, the cost, in terms of signalling overhead and outage time can be quantified as the number of nodes that have to renumber multiplied by the message overheads or latency incurred whilst waiting for a new address. Zhou et al. [21] attempted to avoid such renumbering by using a network address translator (NAT) [6]. Although this avoids the renumbering cost, this scheme inherits all the limitations of NATs and may thereby cause some applications to fail.

Our addressing sub-layer does not require nodes to renumber because nodes can continue to operate even in the presence of address conflicts. However, our scheme does require modifications to the networking stack and routing protocols, in particular when responding to a conflict indication message from our sub-layer. Note that although it is no longer required, nodes are still encouraged to renumber in order to ensure that there is no ambiguity at higher layers.

5.2 Simulation

We extended the AODV [11] implementation in the *ns-2* simulator [1] and performed simulations with 50 nodes moving randomly at a speed of 10m/s in a grid of $1500m \times 1500m$. We then introduce a node with the same as address as one of the 50 other nodes. The number of packet-level conflicts is the monitored as packets from 20 flows are routed through the system.

Figure 7 shows the total number of observed conflicts in the MANET. We see that at different times in the simulation nodes experience varying rates of address conflicts. This shows that address conflicts are a serious problem for which existing solutions are non-ideal in that the unpredictable nature of the conflicts makes it difficult to estimate the signalling overheads as well as the impact that renumbering has on existing connections. Our sub-layer is a preventative measure that solves both these issues. First, it has a deterministic fixed-cost overhead based since the sub-layer adds a small shim-header to every packet. This means that the impact of the scheme can be accommodated for during design. Second, our scheme does not break any established connections as it does not arbitrarily demand any node to renumber.

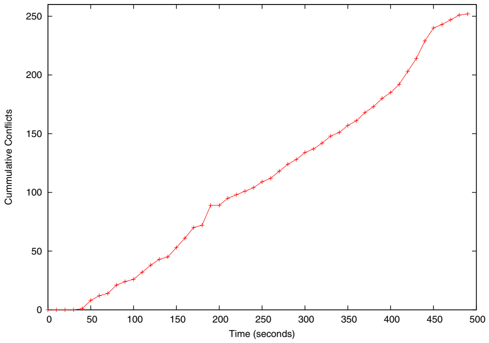


Figure 7. Number of observed conflicts over time.

6 Conclusion

We have presented a set of novel features that a reactive routing protocol can employ in order to route packets in a MANET with address conflicts. These novel features are:

- *Enhanced ARP*. This feature processes HELLO messages on behalf of the routing protocol, maintains IP to MAC address mappings and prevents contamination of a node's routing table that may result in lost packets or incorrect updates performed on a given route.

- *Conflict notification and next-hop forwarding table*. Our sub-layer sends a conflict indication signal to the routing layer whenever it detects nodes with a similar address. In response to the signal, the routing protocol and our sub-layer constructs a forwarding table that allows packets to be forwarded unambiguously.

- *Shim header*. The header carries with it the source, destination, and the next-hop node's UIDs. These fields enable detection of conflicts, and overcome problems resulting from neighboring nodes sharing the same MAC address. Also the shim header provides a list of known UIDs to a source and destination nodes if their respective address is experiencing conflict, hence enabling higher layer protocols or applications to resolve the conflict intelligently.

- *UID role expansion*. Unlike existing works, e.g., [16], that used UID only to detect conflicts and renumber hosts in order to ensure the resident routing protocol continues to function properly, our scheme expands upon the role of UID further where together with conflict avoidance forwarding a routing protocol is able to route a packet through nodes with conflicting addresses.

  Further, as the UID is used only for conflict resolution, and not for routing or applications, it is possible to mitigate UID conflicts by using any reactive conflict resolution algorithm without incurring the serious disadvantages that are discussed in Section A. Indeed, in this way, the proposed solution can be seen as complimentary to existing approaches.

Currently, we are investigating how pro-active routing protocols will be able to employ our sub-layer, and also how our sub-layer can be employed when a MANET connects to a network infrastructure. Another interesting area of research would be to incorporate ideas pertaining to securing MANET such as routing updates, and defending against rogue nodes, for example a node with a spoofed IP address.

References

[1] Wireless and mobility extensions to ns-2. Software Documentation, 1999. http://monarch.cs.cmu.edu.

[2] A. M. andR. Nagpal. Slides-12-1: Great expectations. MIT Course Notes, Mathematics for Computer Science, Nov. 2002.

[3] S. Cheshire, B. Aboba, and E. Guttman. Dynamic configuration of link-local IPv4 addresses. Internet Draft: draft-ietf-zeroconf-ipv4-linklocal-13.txt, Feb. 2004.

[4] K.-W. Chin, J. Judge, A. Williams, and R. Kermode. Implementation experience with MANET routing protocols. *ACM/SIGCOMM Computer Communications Review*, 32(5), Nov. 2002.

[5] R. Droms. Dynamic host configuration protocol. RFC 1531, Oct. 1993.

[6] K. Egevang and P. Francis. The IP network address translator (NAT). RFC 1631, May 1994.

[7] M. Mohsin and R. Prakash. IP address assignment in a MANET. In *IEEE Milcom*, Anaheim, California, USA, 2002.

[8] S. Nesargi and R. Prakash. MANETconf configuration of hosts in a MANET. In *IEEE Infocom*, New York, USA, 2002.

[9] C. Perkins, J. Malinen, R. Wakikawa, E. Belding-Royer, and Y. Sun. IP address autoconfiguration for ad hoc networks. Internet Draft: draft-ietf-manet-autoconf-01.txt, Nov. 2001.

[10] C. E. Perkins, J. T. Malinen, R. Wakikawa, E. Belding-Royer, and Y. Sun. IP address autoconfiguration for ad hoc networks. IETF Internet Draft, draft-ietf-manet-autoconf-01.txt, Nov. 2001.

[11] C. E. Perkins, E. M. Royer, and S. R. Das. Ad hoc on-demand distance vector (AODV) routing. draft-ietf-manet-aodv-06.txt, July 2000.

[12] D. C. Plummer. An ethernet resolution protocol. RFC 826, Nov. 1982.

[13] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. de Groot, and E. Lear. Address allocation for private internets. RFC 1918, Feb. 1996.

[14] S. Thomson and T. Narten. IPv6 stateless address autoconfiguration. RFC 2642, Dec. 1998.

[15] C.-K. Toh. A novel distributed routing protocol to support ad-hoc mobile computing. In *Proceedings of IEEE Phoenix Conference on Computer and Communications*, Arizone, USA, Mar. 1996.

[16] S. Toner and D. O'Mahony. Self-organizing node address management in ad-hoc networks. In *Springer Verlag Lecture Notes in CS 2775*, Springer Verlag, Berlin, 2003.

[17] N. Vaidya. Weak duplicate address detection in MANETs. In *ACM MobiHOC*, Lausanne, Switzerland, 2002.

[18] E. W. Weinsstein. Birthday problem. From MathWorld http://mathworld.wolfram.com/BirthdayProblem.html.

[19] K. Weniger. Passive duplicate address detection in MANET. In *IEEE WCNC 2003*, New Orleans, USA, Mar. 2003.

[20] W. Wimer. Clarifications and extensions for the bootstrap protocol. RFC 1542, Oct. 1993.

[21] H. Zhou, M. W. Mutka, and L. M. Li. IP address handoff in the MANET. In *IEEE Infocom*, Hong Kong, China, 2004.

[22] H. Zhou, L. M. Ni, and M. Mutka. Prophet address allocation for large scale manets. In *IEEE Proceedings of INFOCOM'2003*, San Francisco, USA, Mar. 2003.

A Likelihood of Address Collisions

In this appendix, we provide an analysis of how likely it is for a node to experience an address conflict as well as determine the probability of delivering a message in the presence of address conflicts.

There are two distinct parameters with regard to address conflicts, namely the probability of there being *at least one* conflict in a MANET, as well as as estimation of the expected number of conflicts. In other words, the measure of probability tells us of the likelihood of there being a conflict while the expected number of conflicts tells us how serious the problem is.

To begin this analysis, we make the following assumptions:

- The MANET has a steady state population of N nodes.
- The address space contains α unique addresses.
- R is the number of nodes that enter and leave the MANET during an arbitrary period. It is these incoming R nodes that can lead to address conflicts.
- All nodes in the steady state have no address conflicts.
- $R \leq N$.

Given the above, and using the classical birthday problem [18] as our base, we find that the probability of at least one address conflict occurring due to the incoming R nodes is

$$PC_{N,\alpha,R} = 1 - \frac{(\alpha + R - N)(\alpha + R - N - 1)(\alpha + R - N - 2)\ldots(\alpha - N + 1)}{\alpha^R} \tag{1}$$

or, equivalently,

$$PC_{N,\alpha,R} = 1 - \frac{(\alpha + R - N)!}{(\alpha - N)!\alpha^R} \tag{2}$$

Figure 8 depicts Equ. 2 over a range of N and R values, with $\alpha = 65536$ (i.e., maximum number of addresses in a class B network). As shown in Figure 8, the probability of at least one address conflicts occurring is quickly becomes very high. For example, in a MANET with a Class-B address space that contains only 1,000 active nodes, the probability of at least one address conflict is 99.95%.

Therefore, we conclude that it is critical that any large, dynamic MANET must have mechanisms in place that enable it continue operating in the presence of an address conflict given that at least one conflict is almost guaranteed to occur. For example, a MANET must not use routing protocols that suffer cause wide-spread route thrashing in the presence of a conflict. This requirement defines the *Uniqueness of Addresses* argument of section III.

With the probability of at least one conflict considered, we now move onto the calculation of how many conflicting addresses we expect to find in the MANET.

We start by determining the expected number of node pairs that share addresses as given in [2]:

$$E[\#ofPairs] \approx \frac{N^2}{2\alpha} \tag{3}$$

For example, in a network of 1,000 nodes with a class-B address space, we would expect approximately eight pairs of nodes with address conflicts. Given this, we can now derive the probability of a node experiencing a conflict as being:

$$PR_{node} = \frac{2 \times E[\#ofPairs]}{N} = \frac{N}{\alpha} \tag{4}$$

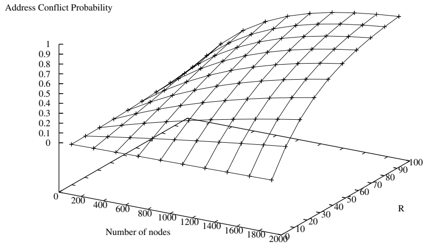


Figure 8. Probability of one address conflict with respect to the number of nodes in the steady state and number of incoming nodes R.

To continue our 1,000 node MANET example, the probability of a given node being in conflict is 1.5%.

We now consider the number of nodes that a packet will encounter as it traverses a given route to a destination node. We assume that a node's communication area is C $units^2$ and that the node density is D nodes/$unit^2$. In other words, any given node is within direct communicate range with CD other nodes. With this information, we can now calculate the probability of successfully delivering a message over H hops as follows:

$$S_{Dlvry} \simeq \begin{array}{rr} (1 - PR_{node})^{CD}, & if H = 1 \\ (1 - PR_{node})^{CD} \times (1 - PR_{node})^{\frac{(H-1)CD}{2}}, & if H > 1 \end{array}$$

The reasoning behind S_{Dlvry} when $H > 1$ is that the packet in every hop after the first only encounters approximately $CD/2$ "new" nodes since half the nodes are the same as the previous hop. We can see that even with a small PR_{node} value, the probability of successfully delivering a message drops rapidly. For example, in our 1,000 node MANET, assuming each node can see 10 others directly, the probability of a packet being successfully received after 8 hops is only 50%. Note that all losses in this model are due to address conflicts – when one considers the inevitable errors due to channel losses, the probability of a successful transmission would quickly drop to zero.

From this analysis, the errors due address conflicts are significant, especially in large networks. This means that it is critical that we have a mechanism to mitigate these errors, lest the MANET become non-functional.