International Journal of Computer Engineering Research Vol. 2(2), pp. 19-27, March 2011
Available online at http://www.academicjournals.org/IJCER
ISSN 2141-6494 ©2011 Academic Journals

Review

Towards auto-configuring routing protocols for wireless ad-hoc networks

M. B. Mutanga*, P. Mudali and M. O. Adigun

Department of Computer Science, University of Zululand, KwaDlangezwa, South Africa.

Accepted 2 March, 2011

The importance of wireless ad-hoc networks in community and commercial connectivity cannot be underestimated because of the benefits associated with these networks. Self-organization will minimize the need for manual configuration. In essence, self-organization provides an out-of-the-box functionality such that very little technical expertise is required to setup a network. Providing unique IP addresses efficiently in ad-hoc networks is still an open research question. In general, nodes often are assumed to have addresses configured in advance, but in ad-hoc networks this is not the case and is not easily accomplished. Nodes require a unique address for packets to be delivered to the correct destination. Many protocols to address this problem have been proposed and most of them are independent from the routing protocol and hence fail to address this problem. Despite the interdependence of routing protocols and IP address auto-configuration, no much effort has been put in trying to investigate this. In this paper we argue that incorporating auto-configuration functionalities into routing protocols will address open issues in this area. We propose general solutions for use in proactive and reactive routing protocols.

Key words: Ad-hoc network, IP address, routing protocols, auto-configuration.

INTRODUCTION

The autonomous nature of wireless ad-hoc networks requires the existence of an IP address auto-configuration mechanism. However in recent years, a lot of research in ad-hoc networks has concentrated on routing protocols. The same intensity has not been applied to other important related areas, such as IP interface addressing. Routing protocols typically rely on nodes having a unique address (Cavalli and Orset, 2005). In general, nodes are often assumed to have addresses configured a priori, but in ad-hoc networks this is not the case and is not easily accomplished. Although routing protocols assume the existence of unique node addresses, the question of how to provide them remains open. A lot of IP address auto-configuration protocols have been proposed in literature. The purpose of having an address auto-configuration protocol is to manage the address space and configure nodes with addresses that are either local scope that is, IP addresses valid only within a particular ad-hoc network or global scope. Automatic configuration of nodes in wireless ad-hoc network will help in reducing administration efforts by users and network administrators (Weniger, and Zitterbart, 2004). One may argue that the problem of address auto-configuration can be solved easily by constructing a unique address from the medium access control (MAC) address. For example, IPv6 enables the construction of an address from the MAC address, which is meant to be globally unique but a major concern with this idea is the issue of location privacy (Weniger, and Zitterbart, 2004). This might also compromise on security of targeted nodes or individuals.

Automatic configuration using random numbers is therefore a viable solution to this problem but however, such a mechanism has to cope with a highly dynamic environment and uncertain network structures (Fan and Subramani, 2005). Another school of thought also argues that MAC addresses can be duplicated. The work in Weniger and Zitterbart (2004) reports that there are

*Corresponding author. E-mail: bethelmutanga@gmail.com. Tel: +27 35 9026706 or +27 722657439.

instances of network adapters with unregistered or duplicate MAC addresses on the market, and also that some network adapters allow users to change the MAC address to arbitrary values. It is thus clear that automatic configuration is one of the best options to this problem.

IP address auto-configuration protocols may be classified under umbrella categories of stateless and stateful approaches. In designing these protocols, the following factors must be considered (Harish et al., 2008): (1) Network partitioning and merging, (2) Duplicate address detection (DAD), (3) Scalability, (4) Security and authentication. These factors affect the correct routing of data packets in the network. Despite this interdependence, not much effort has been put in investigating ways of integration IP address auto-configuration protocols with routing protocols. Most proposals are independent of the routing protocol hence making it difficult to detect address conflicts and network mergers. The applicability of these protocols is still debatable since most of them are tested without any other traffic on the network (routing protocol traffic, application traffic). It is not clear how these schemes affect the routing protocol traffic in terms of packet loss, throughput, delay etc. It is also not clear how routing protocol traffic will affect IP address auto-configuration as far as latency, communication overhead and address uniqueness is concerned. How these protocols interact with the routing protocol for duplicate address detection, security, detection of network merging and partitioning is not clear despite the close relationship between routing protocols and IP address auto-configuration protocols. In this paper we propose a paradigm shift. We argue that incorporating IP address automatic configuration functionalities shall solve the open issues around this area.

Routing protocol paradigms or approaches can be classified into two distinct categories namely reactive and proactive paradigms. Some schemes with characteristics of both approaches are also being developed under the umbrella term of hybrid approaches. Routing protocols periodically or otherwise, send control messages for route discovery and or maintenance. Information in such messages can be useful for IP address auto-configuration protocols, for example, nodes can detect network merging from receiving hello messages from a different network. The routing protocol can detect IP address duplicates by analysing routing information (Weniger, 2004). The proposal in (Saadi et al., 2007) has also shown that analysing routing protocol information can easily detect address conflicts without additional traffic.

In proactive routing protocols, each node maintains routing information to every other node in the network. The routing information is usually kept in a number of different tables. These tables are periodically updated and/or if the network topology changes. The routing table stores the routes (and in some cases, metrics associated with those routes) to particular network destinations. This information contains the topology of the network immediately around it.

In reactive routing protocols, routes are determined and maintained for nodes that require to send data to a particular destination. Route discovery usually occurs by flooding a route request packets through the network. When a node with a route to the destination (or the destination itself) is reached a route reply is sent back to the source node using link reversal if the route request has travelled through bi-directional links or by piggy-backing the route in a route reply packet via flooding (Mehran et al., 2004).

Characteristics of routing protocols can be explored to provide a solution to the auto-configuration problem. For example the discovery of routes in reactive routing protocols is similar to performing a duplicate address detection (DAD) procedure in stateless protocols. In this paper, we present generalized IP address auto-configuration solutions for proactive and reactive routing protocols with the intention to stimulate more research in this direction.

The rest of this paper is organised as follows: First, we discuss some research issues in IP address auto-configuration. Secondly, an overview of related work, thirdly, we outline how IP address auto-configuration functionalities can be integrated into both reactive and proactive routing protocols and finally the conclusion of the paper.

ISSUES IN IP ADDRESSING IN WIRELESS AD-HOC NETWORKS

Due to the unique characteristics of wireless ad-hoc networks, there are various issues that need to be considered when developing an IP address auto-configuration protocol. The work in Harish et al. (2008) also gives an analysis of these issues. Subsequently, we present our view and analysis of these issues:

Security

Wireless ad-hoc networks have unique characteristics thereby making it difficult to address security and authenticity issues. The work in Kumar et al (2008) gives possible attacks to the IP auto-configuration process. These attacks include Address Spoofing Attack, Address Conflict Attack, Address Exhaustion Attack, and Negative Reply Attack. Most protocols do not address security during auto-configuration at all. For example, proposals in Fazio et al. (2006), Günes and Reibel (2002), Indrasinghe et al. (2006), Kim et al. (2007) and Mutanga et al. (2008) only addressed the auto-configuration problem whilst the security issues surrounding this aspect are not addressed. The work in Cavalli and Orset (2005) and Pan et al. (2005) are some of the few proposals that consider security during automatic configuration. The proposal in Pan et al. (2005) binds each IP address with a public key, allows a node to self-authenticate itself, and thus thwarts address spoofing and other attacks associated with auto-configuration. In Cavalli and Orset (2005) a protocol that uses the buddy system technique to allocate the addresses, as well as an algorithm allowing to authenticate the participants inside the network is proposed.

Scalability

In most cases the process of IP address auto-configuration requires that nodes exchange a number of messages before a node can be allocated an IP address. These messages might either be flooded in the network or exchanged locally and they usually grow with network size leading to high overhead (Harish et al., 2008).

Most IP address auto-configuration protocols are independent of the routing protocol hence they define their own data packets to detect network merging or duplicate IP addresses. This results in increased communication and high latency overhead and might disrupt routing. Stateless approaches degrade dismally when the network grows because of the flooding mechanism that is used to detect duplicate IP addresses. Both communication overhead and latency are generally high in this approach. Some stateful approaches, such as the Prophet (Zhou et al., 2003), try to address this problem by configuring nodes using local messages only. This however compromises on the uniqueness of the address. The biggest challenge in building scalable protocols therefore is to try and reduce communication overhead without compromising on address uniqueness and latency. The range of IP addresses should also be scalable. IP addresses should not run out of availability when a large number of nodes are joining (Harish et al., 2008).

Duplicate address detection

Duplicate address detection (DAD) is usually required when either a new node joins a network or when two more independently configured networks merge. Stateless approaches use DAD when new nodes join the network. A DAD message containing the requested address is broadcast and any node using that address defends it by sending a conflict notification message. When two or more networks merge, there is need to detect and resolve duplicate IP addresses. This might require some of the nodes to relinquish their IP addresses and acquire new ones. A duplicate address detection mechanism is also required as continuous process to guard against duplicate addresses caused by erroneous allocation of duplicate addresses. This can be done by analysing routing protocol information for hints that can point to the existence of a duplicate address. Most IP address auto-configuration protocols, however, are independent of the routing protocol hence they define their own data packets to detect network merging or duplicate IP addresses. This results in high communication overhead and might disrupt routing. Incooporating IP address auto-configuration functionalities into routing protocols may be favourable in wireless ad-hoc networks since nodes are likely to be using the same routing protocol and the networks are usually administered by a single entity.

Network partitioning and merging

Network merging occurs when two or more separately configured networks come together to form one network. This can be as a result of mobility or other factors. If each partition has independently allocated or configured its own addresses, two nodes may end up sharing the same address. Therefore, after a network merger is detected, the first task is to detect address conflicts and then take corrective action, that is some nodes need to acquire new addresses. This, however, is possible if the total number of nodes from the two networks are less than the total address space. To detect network merging, some approaches make use of periodic messages that are broadcast to first hop neighbours. In MANETconf (Nesargi and Prakash, 2002), if a node receives a hello message with a different network identifier, network merging is detected. A network might also be partitioned in to two or more partitions due to various factors. Nodes need to detect this so that they can allocate the IP addresses allocated nodes in the other partition. However, when such networks merge again, duplicate IP addresses might occur. This then requires nodes to generate new partition IDs when network partition is detected.

Network merging is a common occurrence in wireless ad-hoc networks. Consider a network of 100 nodes that are scattered over a 1000 x 1000 m square area. If the nodes are randomly switched on, a lot of independent networks will be formed. To prove this point, simulations were conducted in ns2 and the results were obtained. Nodes were randomly switched-on and the auto-configuration process allowed to take place. The number of independently configured networks formed was recorded. We varied the number of participating nodes and merging was not allowed to take place during simulation so that the number of networks could be counted at the end. The experiment was run for eight times and average values were used for the analysis.

From the results obtained (Figure 1), it is interesting to note that 100 nodes on a 1400 x 1400 m area recorded as much as 20 different networks whilst 50 nodes also recorded up to 17 independent networks. From the number of independently configured networks recorded in

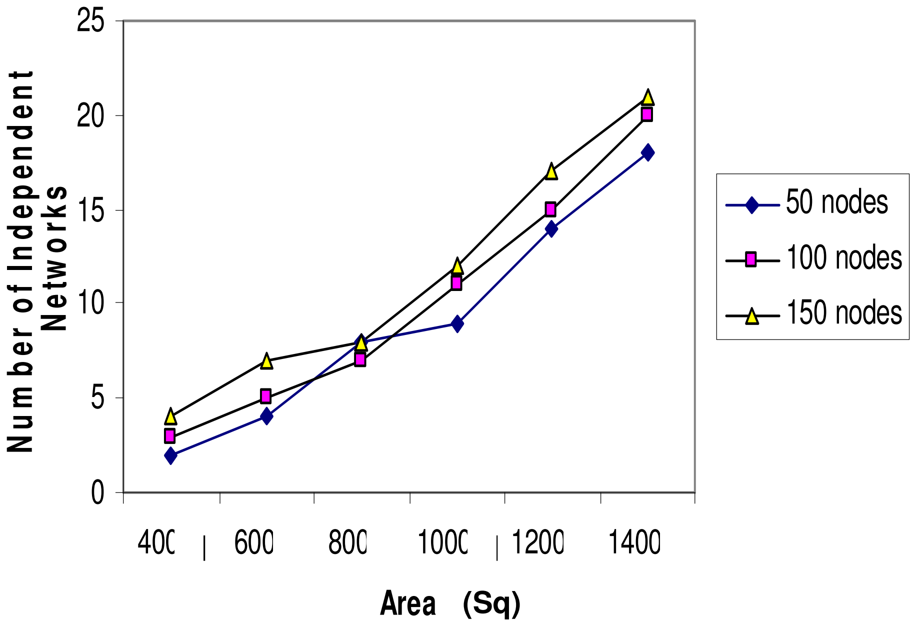

Figure 1. Number of independently configured networks.

the experiments, it is clear that factors such as the time taken for a node to detect a network merger and resolve any possible address duplications and number of packets generated by each node in order to detect and resolve a network merger are of paramount importance.

Wireless ad-hoc network scenarios usually involve relatively closed user groups (for example, community networks) or networks run by a single entity, routing protocol dependency is not an issue. This raises the possibility of incorporating IP address functionalities in the routing protocol unlike in generic networks.

CURRENT APPROACHES IN IP ADDRESS AUTO-CONFIGURATION

IP address auto-configuration protocols are generally classified into three categories namely: stateless, stateful and hybrid.

Stateless approaches

Protocols following the stateless paradigm do not maintain an address allocation table. An allocation table is a list of all IP addresses in use in a network at any given time. In this approach, nodes generate their own IP addresses and check for possible conflicts through a DAD procedure, hence most of the research classified under this approach is aimed at coming up with the most efficient DAD procedure (Mutanga et al., 2008). If a conflict is detected, the process is repeated, thus making DAD the cornerstone of the stateless paradigm.

In Strong-DAD (Perkins et al., 2001), a node randomly selects an IP address and checks whether or not it is used in a network using a DAD procedure. In fact a new node chooses two addresses: a temporary address and the actual address to use. During the IP address negotiation process described previously, new nodes use temporary IP addresses for communication. The temporary address is not verified for uniqueness. The network is flooded with an address request (AREQ) message containing the selected address. A node that uses the same address defends its address by replying with an address reply (AREP) message. If the address is currently in use, the process is started again until a free IP address is obtained. An address is assumed to be free if the timer for a DAD trial expires before receiving a conflict notification message (AREP). Due to broadcast, Strong-DAD has high communication overhead. It performs a DAD procedure every time a new node requests for an IP address. The number of failed DAD procedures increases as network size increases in size.

Due to the increase in the probability of failed DAD as the number of nodes increases, scalability is a problem in Strong-DAD. Also from the birthday paradox, address conflicts are likely to occur when each node chooses its address by random selection (Jeong et al., 2004). As the network size increase, latency and communication overhead also increase. Since the approach uses a time based DAD, address allocation latency depends on the DAD timeout and the number of DAD trials. If DAD is successful on the first attempt, address allocation latency is equal to the DAD timeout. Strong DAD does not specify how it handles the situation of more than one node requesting for the same IP address at the same time hence address uniqueness is compromised. However, the DAD proposed is likely to get a unique address if all the network nodes are reachable. Strong-DAD does not provide a way for solving the problem of two nodes using the same temporary IP address during the address negotiation process.

In AIPAC (Fazio et al., 2006), the authors proposed a stateless IP address auto-configuration protocol, AIPAC, which is based on Strong-DAD (Perkins et al., 2001). This mechanism avoids the storage of a lot of information about the network and does not produce too much traffic in the communication channels. Since Strong-DAD does not provide a way for solving the problem of two nodes using the same temporary IP address, AIPAC uses the concept of Requester and Initiator, which is defined in ManetConf. The Initiator selects an address at random among the allowed addresses, and sends in broadcast a Search_IP packet. The selected address is specified in the packet. Any node receiving this packet checks whether the address is known (whether this address belongs to it or to another node in its routing tables). If a match is detected, the node sends a Used_IP message to the Initiator. When the Initiator receives the Used_IP message, the address assignment procedure is restarted, and a new address is selected. Conversely, if no reply is received for a given time interval (Search_IP timer), the Initiator sends the Search_IP packet again, in order to face up possible errors in wireless channels. If neither replies arrive, it means that the address is not used yet. The Initiator then notifies the Requester with the NetID of the network and the IP address that it has to use.

Like Strong-DAD, this scheme has high communication overhead. It performs a DAD procedure every time a new node requests for an IP address. The number of failed DAD procedures is likely to increase as the network increases in size. This affects the overall quality of service of the network and might also increase power consumption of the nodes. This scheme does not specify how it handles the situation of more than one node requesting for the same IP address at the same time hence uniqueness in this scheme is not guaranteed. However, like Strong-DAD, the DAD proposed is likely to get a unique address if all the network nodes are reachable.

Stateful approaches

Protocols that follow the stateful paradigm assume that the addresses that are going to be assigned are not being used by any node in the network. This is achieved by guaranteeing that the nodes that participate in the allocation of IP addresses have disjoint address pools. In this case, performing a DAD is not necessary. Another way is to distribute the address allocation table to all network nodes so that they can configure new nodes since they know which IP addresses are free. This approach requires that the allocation tables be synchronized. In this case, a DAD is required to guard against a situation in which the same IP address is being requested for at the same time.

In MANETconf (Nesargi and Prakash, 2002), the authors proposed a system for the management of the IP addresses which is distributed in all the nodes of the network. A new node has to rely on a configured node (initiator), to negotiate for an address for it. Each node belonging to the network stores all the used addresses, as well as the ones that are going to be assigned to the new nodes. The initiator selects an address among the available ones, and performs a DAD procedure. This is a way for checking whether the same address is being assigned in another part of the network. If all the nodes send a positive reply for this request, the address is assigned. This process is repeated until a free IP address is obtained. All nodes in the network periodically broadcast their IP address allocation for state synchronization purposes.

If a node leaves the network gracefully, it has to release its address, by sending a bye message in broadcast. This allows the other network nodes to update their address allocation tables accordingly. For managing the merging of different networks, a single network ID is used, which is selected by the node with the lowest IP address. When nodes belonging to different networks get in contact, they detect the merging and check for possible duplicated addresses. The system has to verify also if network partitioning occurs. If some nodes do not respond to the subsequent assignment procedure of the IP address, then partitioning is detected. If such nodes also include the one that originally determined the network ID, a new one is selected by the node with the lowest IP address. Since the IP assignment operations may not take place for a long-time, and thus no partitioning can be detected, the node with the lowest IP address must periodically broadcast a message to show its presence. One cannot easily determine how often this message needs to be sent, since this depends on the dynamics of the network.

Although MANETconf is a stateful protocol it employs broadcast similar to the one used in stateless approaches. It also requires periodic state information synchronization which is bandwidth consuming. The length of the IP address assignment process in MANETConf is proportional to the network size because every node in the network takes part in the address assignment process.

In the Prophet's approach (Zhou et al., 2003), the authors proposed a novel approach that follows the stateful paradigm, but the protocol does not store an allocation table. The basic idea is to predict the allocation table using a function f(n) that is distributed among nodes. The authors argued that IP address auto configuration is the same as assignment of different numbers from an integer range, say R, to different nodes. They went on to argue that if all the addresses that have been allocated and those that are going to be allocated are known in advance, then broadcast could be avoided whilst conflict is still detectable. A way to obtain an integer sequence consisting of numbers in R, using a function f(n), was then proposed. The initial state of f(n) is called the seed. Different seeds lead to different sequences with the state of f(n) updated at the same time. The basic idea behind the approach of Prophet is as follows:

The first node, say A, chooses a random number as its IP address and uses a random state value or a default state value as the seed for its f(n).
When another node, say B, approaches A and asks for a free IP address, A uses f(n) to obtain another integer, say n2, and a state value. It then provides them to B. Node A updates its state accordingly.
Node B uses n2 generated by A as its IP address and the state value obtained from node A as the seed for its f(n).
Now nodes A and B are both able to assign IP addresses to other nodes.

Address reclamation is not necessary in prophet because the same number will reoccur in the sequence. Nevertheless, the minimal interval between two occurrences of the same number in the sequence is extremely long. The authors say when a node is assigned an old address X, the previous node with the same address X, is likely to have already left the network. As a result of this, this mechanism does not exclude the possibility of generating duplicate addresses. The mechanism employed in prophet works well with short-lived networks like the proposals in Dijkstra et al. (2006) and Saxena et al. (2005). However, prophet does not flood the network with IP request messages. The new node only communicates with its first hop neighbors and IP addresses are generated locally. This reduces both latency and communication overhead.

Hybrid approaches

Hybrid protocols combine elements of both stateful and stateless approaches. Protocols that follow this approach combine DAD with either a centrally maintained or a distributed common allocation table.

In Wise-DAD (Mutanga et al., 2008) an unconfigured node periodically broadcasts a request to join message. If there is another unconfigured node within its transmission range, a network is automatically formed. The node with the lower host identifier (HID) chooses network parameters, gives the other node an IP address and other configuration details. The HID is a randomly generated temporary IP address used by nodes before they acquire permanent IP addresses. If a configured node receives a request to join message, it assumes that an unconfigured node wants to join the network hence it will offer to act as its initiator by sending a confirmation message. The new node then selects only one of its neighbors node to act as its negotiating agent (initiator). It sends a select initiator message to the first node to respond. The initiator then generates a random IP address from the allowed addresses and checks its allocation table if there is no node in the network that have requested for or used the same IP. If the address is not known, the initiator then performs a DAD (using an address request message).

All nodes receiving an address request packet update their tables and add their IP addresses to the packet before broadcasting it. If any node is using the requested address, it defends it with an IP conflict message and this process is repeated. If no IP conflict message is received after a certain time interval, the address is assumed to be free and the initiator will send an address reply message to the new node. The address reply message will have the IP address for the new node, the network identifier (NetID) and the state information (allocation table). If a node leaves the networks gracefully, it broadcasts a goodbye message and all the nodes delete its IP address from their allocation tables. If a node leaves abruptly, immediate address reclamation is not performed. Since the node will not be sending or forwarding any data packets, other nodes will remove all passive nodes from their allocation tables. Allocation tables are not actively synchronized, they are used only as an estimate of the state information. If a node does not take part in an IP address allocation process for a long time, its IP address will be deleted when the size of the allocation table reaches a certain level because it will be assumed that the node left the network abruptly.

Compared to Strong DAD, Wise-DAD significantly reduced latency, communication overhead and address conflicts. Passively collecting state information reduces the number of DAD trials thereby both reducing latency and communication overhead. However, the number of IP address conflicts recorded was relatively high as compared to stateful approaches like prophet.

AUTO-CONFIGURING ROUTING PROTOCOLS

In order to realize the goal of integrating IP addressing functionalities into with routing protocols, there is need to design algorithms for both reactive and proactive routing protocols, it is imperative to come up with mechanisms of how routing protocols can handle: (1) Network partitioning and merging, (2) Duplicate addresses and solve them, (3) Scalability issues, (4) Security and authentic-cation during and after configuration. These issues affect the way packets are routed in the network. For example, if two networks merge, duplicate IP addresses might occur. Most applications in the ad-hoc networks are based upon unicast communication hence routing protocols require nodes to have unique address for packets to be delivered to the correct destination (Toner and O'Mahony, 2003). Thus, the most basic operation in the

IP layer of ad-hoc networks is to successfully transmit data packets from one source to destination (Zhou, 2003). Subsequently, we present general solutions for integrating IP address auto-configuration functionalities with reactive and pro-active routing protocols.

Auto-configuration in proactive routing protocols

Here, we present an IP address auto-configuration solution for proactive routing protocols. Subsequently, a breakdown of how IP address auto-configuration can be achieved in networks running pro-active routing protocols was given.

Node admission

Nodes running pro-active routing protocols can easily adopt the stateful approach of configuring IP addresses since they store and update topology information. Proactive routing protocols, maintain an up-to-date view of the network by periodically broadcasting the link-state costs of its neighbouring nodes to all other nodes using a flooding strategy (Mehran et al., 2004). Stateful auto-configuration protocols also maintain state information that is, the list of all the nodes that are in the network at any given time. The same information can be obtained from routing tables of proactive routing protocols. It then makes sense to use the same information rather than maintaining two separate states. The concept of initiator can be adopted for the purposes of IP address negotiation.

An un-configured node periodically broadcasts a request message until it receives a reply from another node that will negotiate for an IP address for it. The initiator generates a random IP address and checks if it is in its routing table before it starts the negotiation process through a DAD procedure. If the address is in the routing table, it generates another one otherwise it will perform a DAD procedure and set a timer. If the timer expires without any node defending the requested IP address, the initiator will sends an address reply (AREP) to the new node.

On receiving an address request message other network nodes first check if the message is new or not before checking if the requested IP address has been assigned them. A message sequence number can used to determine if a message is new or not. If the address is found to be in use, an IP conflict is sent to the initiator and the process is repeated. If the message is not new, it is discarded, otherwise it will be broadcast further. Before the message is broadcast, the recipient adds its IP address to the message. As the message is passed from one node to another, a reverse path to the initiator will be contained in the packet. This allows for an IP conflict message to be sent back to the initiator.

Duplicate address detection

To detect duplicate addresses, nodes analyse routing protocol information. The proposal in Vaidya (2002) can be applied. Each node generates a random key at initialization phase, and distributes it with its IP address in all routing messages. Each node maintains keys along with IP addresses of all the other nodes in its routing table. When a node receives a routing message with an IP address that exists in its table, it checks if the keys are different. If they are different, a duplicate address is detected and steps can then be taken to inform other nodes about this duplication. The nodes with duplicated addresses will be required to acquire new ones. Using this approach, nodes can detect duplicate addresses without any additional traffic. This approach however relies on the key-address combination being unique, that is, no two nodes should have the same key and IP address at the same time. The probability of two nodes having the same key and IP address can however be minimized by increasing the range of the key.

Network merging

The concept of network IDs used in the MANETConf protocol can be adopted to handle network merging. The first node in the network generates a random network identifier to be used by all the nodes in the network. Nodes can incorporate network IDs in topology update messages. If a node receives a topology update message with a different network ID, network merging is detected. Nodes detecting the network merging can estimate the number of IP address conflicts by inspecting routing tables of both networks. Only nodes with conflicting IP addresses will then be required to relinquish their IP addresses and acquire new ones.

Auto-configuration in reactive routing protocols

Here, we present an IP address auto-configuration solution for reactive routing protocols. Subsequently, a breakdown of how IP address auto-configuration can be achieved in networks running reactive routing protocols are given.

Node admission

Nodes running reactive routing protocols can easily adopt the stateless approach with minor changes. The discovery of routes in these protocols can be likened to performing duplicate address detection (DAD) procedure in stateless protocols. Route discovery usually occurs by flooding route request packets in the network. When a node with a route to the destination (or the destination itself) is reached a route reply is sent back to the source node using link reversal if the route request has travelled through bi-directional links or by piggy-backing the route in a route reply packet via flooding (Mehran et al., 2004). A DAD message is flooded in the network just like route request messages in reactive routing protocols. We propose the introduction of a new packet similar to the route discovery packet. Minor changes can be made so that the packet can be used to check if the requested IP address is not in use. A new node contacts an already configured node to act as its initiator. To reduce the chances of DAD failing, we can introduce state information maintenance which is passively collected but not actively maintained. The best time interval for state synchronization is an area that needs to be investigated. Passively collecting state information will reduce the number of DAD trials thereby reducing latency and communication overhead. Before the initiator sends an AREQ, it first checks if the IP address is not in the allocation table.

On receiving the AREQ, nodes check if the requested address does not belong to them. If it does, an IP conflict is sent to the initiator and the process is repeated. The allocation tables need not be synchronized or periodically updated since nodes still perform DAD. The allocation tables are merely used to reduce the probability of IP address conflict during a DAD procedure. Before the message is broadcast, the recipient adds its IP address to the message. As the message is passed from one node to another, a reverse path to the initiator will be contained in the packet. When nodes receive AREQ, they also update their allocation tables using IP addresses in the reverse path list before rebroadcast the AREQ. Every node also generates a random key at start-up. The key is used for detecting duplicate addresses. Subsequently, details of how these keys can be used to detect duplicate addresses are given.

Duplicate address detection

Nodes generate keys at start-up and send them when either requesting for a route or responding to a route-request message. On route-discovery a node sends the last known key of the destination and its own key in the route discovery packet. If the destination receives a route request message with a different key, a duplicate address is detected. This also serves as a way of authenticating both the receiver and the sender before they start communicating. The receiver also checks if the sender's key is different from what it has on its table. This can conserve bandwidth since duplicate addresses are detected only when the nodes with duplicated addresses wants to receive or send data. This means that nodes with duplicate addresses can still be able to forward data on behalf of other nodes without any problems. Actually this also implies that duplicate addresses can be tolerated (and allowed to exists) as long as they do not affect the routing process. Unlike in WeakDAD (Vaidya, 2002), there is no need for keys to be carried along with routing packets but only in route discovery packets hence saving a considerable amount of bandwidth.

Network merging

To detect network merging, nodes periodically send one hop messages with their network identifiers. The network identifiers can be incorporated in the hello messages of the routing protocol. Reactive routing protocols periodically send hello messages to first hop neighbours hence no additional packets need to be defined. If a node receives a hello message with a different network ID, network merging is detected. The node that detects the network merging can then respond to the hello message notifying the other network of the possibility of the two networks merging. The two nodes can exchange their allocation tables so that they can estimate the number of address duplicates. The nodes with conflicting addresses can then notified. Another way is to make all the nodes in the network with the lower network ID to relinquish their IP addresses and starts the process of IP address requisition.

CONCLUSION AND FUTURE WORK

The advent of wireless networking has significantly reduced the costs of setting up computer networks. Wireless ad-hoc networks in particular have the potential to expand but a lot of research is still needed to realise this dream. Automatic configuration of nodes is one area that still needs investigation. The following factors are important in the design of auto-configuration protocols: Network partitioning and merging, Duplicate addressed and solve them, scalability issues and security and authentication during and after configuration. These issues affect the way packets are routed in the network.

Despite this interdependence, not much effort has been done in investigating ways of integration IP address auto-configuration protocols with routing protocols. Most proposals in literature are independent of the routing protocol hence making it difficult to detect address conflicts and network mergers. The applicability of these protocols is still debatable since most of them are tested without any other traffic on the network (routing protocol traffic, application traffic). It is not clear how these schemes affect the routing protocol traffic in terms of packet loss, throughput, delay etc. It is also not clear how routing protocol traffic affect IP address auto-configuration as far as latency, communication overhead and address uniqueness is concerned. How these protocols interact with the routing protocol for duplicate address detection, security, detection of network merging and partitioning is not clear despite the close relationship between routing protocols and IP address auto-configuration protocols. Integrating auto-configuration functionalities into routing protocols is a candidate solution to open issues to the IP address auto-configuration problem. We hope that our contributions will stimulate further research in this direction. The future focus of this work will be on implementing the proposed approaches to test the validity of our proposition.

ACKNOWLEDGMENTS

The authors appreciate the support given to the Department of Computer Science at the University of Zululand by industry partners. In addition, the authors also appreciate contributions from the Wireless Mesh Networks Group at the University of Zululand.

REFERENCES

Cavalli A, Orset J (2005). Secure hosts auto-configuration in mobile ad hoc networks. Ad Hoc Networks, 3(5): 656-667.

Dijkstra F, Van der Ham J, Cees TAM (2006). Using zero configuration technology for IP addressing in optical networks, Future Generation Comput. Syst.. 22(8): 908-914.

Fan Z, Subramani S (2005). An address autoconfiguration protocol for IPv6 hosts in a mobile adhoc Network, Comput. Commun., 28(4): 339-350.

Fazio M, Villari M, Puliafito A (2006). AIPAC: Automatic IP address configuration in mobile ad hoc networks, Comput. Commun., 29(8): 1189-1200.

Günes M, Reibel J (2002). An IP Address Configuration Algorithm for Zeroconf Mobile Multihop Ad Hoc Networks, Proceedings of the International Workshop on Broadband Wireless Ad-Hoc Networks and Services, Sophia Antipolis, France.

Harish K, Singla RK, Malhotra S (2008). Issues & Trends in AutoConfiguration of IP Address in MANET IProceed. ICCCC 2008, pp. 353-357

Indrasinghe S, Pereira R, Mokhtar H (2006). Hosts Address Auto Configuration for Mobile Ad Hoc Networks, in the proceedings of HET-NETs, West Yorkshire UK.

Jeong J, Park J, Kim H, Kim D (2004). Ad Hoc IP Address Autoconfiguration for AODV, IETF Internet-Draft.

Kim N, Ahn S, Lee Y (2007). AROD: An Address Autoconfiguration with Address Reservation and Optimistic Duplicated Address Detection for Mobile Ad Hoc Networks, Comput. Commun., 30(8): 1913-1925.

Mehran A, Tadeusz W, Eryk D (2004) A review of routing protocols for mobile ad hoc networks, Ad Hoc Networks. 2(1): 1-22

Mutanga MB, Nyandeni TC, Mudali P, Xulu SS, Adigun MO (2008). Wise-DAD Auto-Configuration for Wireless Multi-hop Networks, In the proceedings of Southern Africa Telecommunication Networks and Applications Conference.

Nesargi S, Prakash R (2002). MANETconf: configuration of hosts in a mobile ad hoc Network, in: Proceedings of the 21st Annual Joint Conference of IEEE Computer and Communication Societies, New York.

Pan W, Reeves DS, Ning P (2005) Secure Address Auto-configuration for Mobile Ad Hoc Networks, Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services.

Perkins C. Malinen T, Wakikawa R, Belding-Royer E. Sun Y (2001). IP address autoconfiguration for ad hoc networks. IETF Internet Draft.

Saadi B, Adjih C, M¨uhlethaler P, Laouiti A (2007). Duplicate Address Detection and Autoconfiguration in OLSR, J. Uni. Comput. Sci., 13(1): 4-31

Saxena N, Tsudik G, Yi JH (2005). Efficient Node Admission for Short-lived Mobile Ad Hoc Networks, Proceedings of the 13TH IEEE International Conference on Network Protocols, pp: 269 – 278.

Toner S, O'Mahony D (2003). Self-Organising Node Address Management in Ad-hoc Networks, in Springer Verlag Lecture notes in Computer Science 2775, Springer Verlag, Berlin, pp: 476-483, 2003.

Vaidya NH (2002). Weak Duplicate Address Detection in Mobile Ad Hoc Networks, Proceedings of ACM MobiHoc, Lausanne, Switzerland, pp: 206–216.

Weniger K (2004). Passive Duplicate Address Detection in Mobile Ad Hoc Networks, In IEEE Wireless Communications and Networking Conference (WCNC), New Orleans, USA.

Weniger K, Zitterbart M (2004). Address Autoconfiguration in Mobile Ad Hoc Networks: Current Approaches and Future Directions, IEEE Network Magazine Special issue on 'Ad hoc networking: data communications & topology control'.

Zhou H (2003). A Survey on Routing Protocols in MANETs. Technical Report: MSU-CSE-03-08.

Zhou H, Ni L, Mutka M (2003). Prophet address allocation for large scale manets, Ad Hoc Networks. 1(4): 423-434