Packet buffering in Manet

Nishu Garg[1], Ruchika Sharma[2] and Parul Pal[3]
1,2,3 JaganNath Institute of Management Sciences, DELHI

Abstract
MANET consists of a group of wireless nodes, which help each other in forwarding packets to enable communications. This type of network requires no fixed network infrastructure [11]. Since the node may change their position unpredictably, the data transmission can be temporarily disturbed if any link on the path fails. Ad hoc routing protocols have been designed to reroute traffic when opposed with network congestion, faulty nodes, and dynamically changing topologies. The common design objective of ad hoc routing protocols is to faithfully route packets from a source node to a destination node. Detecting nasty nodes in an open ad hoc network in which partaking nodes have no previous security associations presents a number of challenges not faced by wired networks. Unless an alternate path is immediately available, the packets are likely to be lost or delayed. An ad hoc network does not have different types of network elements like switches, hubs etc. where the Intrusion Detection System (IDS) can collect and assess assessment data for the whole network. A number of neighbor-monitoring, trust-building, and cluster-based voting schemes have been proposed in the research to enable the detection and reporting of nasty activity in ad hoc networks. The resources consumed by ad hoc network member nodes to monitor, detect, report, and diagnose nasty activity, however, may be greater than simply rerouting packets through a different available path. To overcome this problem[2] suggest some technique such as using the two-hop routing information and data buffering [1] [2]. The route caching [9] has been suggested many a times but the data packet buffering has been given some attention. In this paper we propose the new design for enhancing the performance of MANET by buffering the data and providing an alternate path for the packet to reach their destination.

Keywords:

MANET, Packet Buffering, Two Hop Routing, One Hop Routing, Link Layer Notification (LLN), Local Route Table (LRT)

1. Introduction

The mobile nodes impose many challenges, such as route breakage and packet losses. In MANET the communication between nodes involves several controls. Depending on the type of the protocol used (AODV, DSR (reactive), OLSR (proactive) and GRP (geographic) [12]), nodes may take several actions such as to coordinate the clock of the participating nodes for a number of reasons. Nodes know about the presence of other nodes by means of special messages such as HELLO. Once the nodes know about other nodes they can start communicating to each other which are acknowledged by the receiver. If the links are symmetric (bidirectional) [2] then direct un-interrupted transmission will take place. Data may get lost due to many reasons such as nodes out of range, congestion in the network and other unethical transmission etc. A packet buffering may be employed to reduce packet loss [3] as a result of link failure, congestions etc. If the nodes are in the visibility of two-hop [2] distance, there will be minimum packet loss. Since the nodes are mobile, they may go out of the transmission range, especially when the nodes try to move out of each other, in such circumstance the chances of the packet loss will increase. So the packet loss due to the link failure may be decreased by using a technique of packet buffering. The use of the buffer is to temporarily store packet in flow when a route breakage is detected. It acts like cache memory. In [3] the processing time and the buffer overflow condition a method of intimating the previous hop about buffer overflow was recommended, so previous hop may also buffer some packet. This will require additional storage requirement for buffering the packets, but the use of this storage is justified since it will mend the MANET performance.

2. Background

The common way of discovering link breaks for a routing protocol is through lost polling packets (i.e. lost Hello packets). The Hello packets of OLSR are transmitted between one-hop neighbors at a particular time frequency (e.g. every 1 second, which is the suggested transmission frequency of OLSR) and provide neighborhood connectivity knowledge and a means for link break detection. If no Hello packet from a neighbor is received during a specified time interval (e.g. within 4 seconds, the recommended interval of OLSR), the neighbor is considered unavailable and a link to that neighbor is considered as broken. Another method for the routing protocol to notice link breaks is to leave it up to a mechanism implemented at the basic link layer. The routing protocol ought to be notified unequivocally about a link break by the link layer. The disadvantage of this Link Layer Notification (LLN) approach might be the cost of extra implementation complexity. However, the advantage is that the link layer is normally able to detect link breaks more readily. As a link layer, IEEE 802.11 is

Manuscript received June 5, 2011
Manuscript revised June 20, 2011

normally proficient of detecting a link break considerably faster than a second. In divergence, without LLN and with the recommended values of OLSR, a link break will not be detected before 4 seconds at best and 6 seconds at worst.

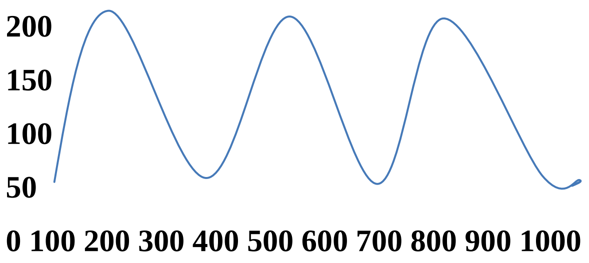

Figure.2.1. accumulated transmissions diagram.

It is important for the complete performance to detect the link break in a timely manner, since two negative effects occur in the period between the physical link break and the detection by the routing protocol. First, the packets queued in the interface queue are marked with an unreachable next hop address. This means that these packets will never reach their destination, and are at this point lost. Second, these packets will be tried transmitted several times by the MAC layer before they are discarded. This will take valuable medium time from packets transmitted from other nodes with a valid next hop address. The retransmission effect is demonstrated through a simulation where a node was placed in the center of the imitation area and set up to receive data from 40 nodes moving randomly inside the simulation area at 10 m/s (Fig. 2). In this imitation, a node inside the transmission area of the receiving node successfully sends traffic to the receiving centered node until it moves out of the receiving node's broadcast area. If a link breaks occurs, but it is not detected by the transmitting node's routing protocol for next 4-6 seconds. At the time when the link break is detected by the routing protocol, the node may have traveled 40 to 60 m past the edge of the transmission area of the receiving node. During this time the MAC layer will transfer each packet with the receiving node as MAC destination several times. Fig. 1 shows the imitation area with the positions for all occurring transmissions plotted in. A loop of an increased number of transmissions is detected outside the transmission area of the receiving node, a direct effect of link breaks and consequent retransmissions. A Manet Node is primarily constructed of three components:-

i. The Node Manager interfaces and extracts all local Radio Node data for the Router and Packet Handler. It accepts information from external sources and explains the information before relaying it to the Router. It also can control the Packet Handler's queues through the State and Control interface.

ii. Packet Handler represents all traits of creating, handling and manipulating network packets.

iii. The Router is responsible for conniving routes on demand from the Packet Handler or the Node Manager and /or it may update/refresh its own routing metrics, proactively. Modeling these elements as components allows for adaptation to existing and future routing protocols, while keeping the key internal and external interfaces constant and independent from routing protocol details.

3. THR Protocol

3.1. THR

The THR is a table driven routing protocol. Instead of keeping the evidence about each and every node as in proactive routing protocol, THR only keeps the information about the immediate neighbor and his neighbor i.e. information about the neighbor's neighbor. Therefore this approach will discover the destination route in fewer numbers of transmissions and also consume less power and also the size of the table maintained at each node will also get reduced. The basic principle of procedure of the THR protocol is as follows:
When a node demand to transmit data packet to some destination node, it first checks its own routing table which contains route to two hop distant node if the route is found then the packet is transmitted to the destination node else a route finding is initiated by the source node dodging the routing loops as discussed in many reactive routing protocol such as DSR, AODV etc.

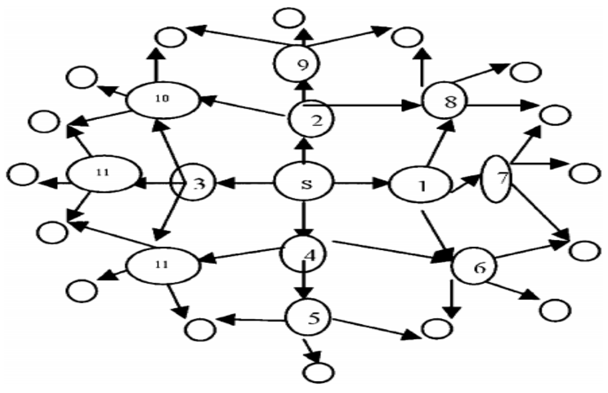

Fig.3.1. a general MANET topology

3.2. Route Discovery

In our projected routing protocol, a routing path is constructed by the combined proactive and reactive process in that short inspirations are transmitted by every node to know about their immediate and the next node. Unlike the pure proactive protocol, the nth node in our approach gets info about n+1st and n+2nd in the visibility range. The information so obtained is stored by the nodes in their internal table known as LRT (local route table). When a source node (S) need to communicate data to some destination node (D) the node S will first check a route in its LRT. If a route to terminus exists, if so the data packet is transmitted to destination, on the other hand, the main route from source node S to destination node D need to be built before source node S can start the data transmission. The method of finding such a routing path is called the main route construction, which originates with the source node S sending a main route request (MRREQ) to all its neighbors. Every host that receives the MRREQ acts exactly the same as the source node does. MRREQ is thus engulfed over the network, and would eventually arrive at node D - S. Node D - S will send a route reply to the source in h-2 hop counts, where h represents the hop count from source to the destination. Every node that receives the route reply will also keep a record of the main route to the destination node D, thus keep the details of the recent and new-fangled route that a node has seen. The construction of the local routing table is built by using short interim Hello packets. The hello packet is processed as follows:

```
If (node_interface_addr = = main address)
Then
Shed the hello packet to avoid routing loop
Else
If (this node has already contains info about route reply initiator in its List of recently seen route reply)
Shed the packet
Else
Save the main address as the neighbor address
Response this node address to source
Update its own local route table (LRT)
End-if
```

Source Node	Intermediate One-Hop Node	Two-Hop Node	Hop-Count	
S	5	3	2	
2	3	6	2	
2	S	-	1	
3	2	S	1	
3	4	5	2	
4	3	2	2	
4	5	6	1	
5	4	3	2	
5	6	D	2	
D	6	4	2	

The route detection process also keeps track of the sequence number and the packet ID to escape any routing loops in the local as well as main route discovery process. The complete route discovery process is described as follows:

1. If the pair originator address, request id for this route request is found in this host's list of recently seen requests, then discard the route request packet and do not process it further.
2. Otherwise, if this host's address is previously listed in the route record in the request, then discard the route request packet and do not process it further.
3. Otherwise, if the goal of the request matches the host's address in the LRT (because the LRT contain information of two-hop reachable nodes), then the route record in the packet holds the route by which the request reached this host from the originator of the route request. Return a copy of this route in a route reply packet to the originator.
4. Otherwise, append this host's own address to the route record in the route request packet, and rebroadcast the request. The above facts can be described by the algorithm given
below:

```
Broadcast( dest_addr.IP, RREQ)
Check if
This node previously seen the RREQ_source_address && request_id
Then
Shed the packet
Else
RREQ_entry already contain Dest Host address
Then
Shed the packet
This node LRT contains entry of Dest Host addr.
Then
Generate RREQ
Intimate to destination about RREQ by source
Else
Update the RREQ packet with this node addr.
Broadcast(dest_addr.IP, RREQ)
End
```

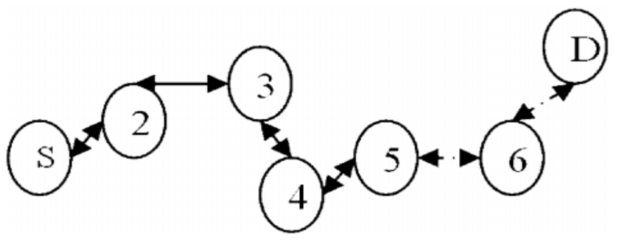

Fig. 3.2: route-Discovery THR

The above algorithm can be explained as follows: Suppose node 3 wish to transmit to node 'D', it first examines its LRT. Since route to node D is not available in the LRT, a request reply is generated by the source node-3 and broadcasted to its one-hop neighbors.Node-4 and node-2 receive the request reply broadcast and searches their LRT, node-4 has an entry for node2 and node-6 as the reachable nodes; node- 2 on the other hand has reachability to node-S. So node-4 broadcast the RREQ and node-5 listens to this request. Node-5 has a route to destination node-D and thus will send a RREQ containing the list of intermediate node through which the data packet may be sent to destination node-D.

3.3. Buffer Methodology

In MANET nodes work in dynamic topology, where a neighbor node may or may not be noticeable in the next moment. We assume that the nodes are moving with steady speed but in casual direction and the nodes have symmetric transmission capability. After the route has been established the transmission proceeds uninterrupted. If as a outcome of mobility the link breaks, then the algorithm given below will help recover the situation by employing the packet buffering.

```
START:
Intimate source of route failure and to initiate new route discovery
Set TTL
Buffer the inward packet
While (route_not_found)
{
Continue Buffering inward packets
If (Buffer_Full)
Goto ABC
if(TTL_expire)
Exit( )
}
Goto XYZ
ABC: Intimate source to pause the transmission
if (TTL_expired)
Goto LAST
Wait till route found
If (route_found)
XYZ:
if (this_node is in the list of intermediate nodes to destination)
{
Get the packets from the buffer and resume the transmission and inform the destination to carry on file saving.
}
LAST: Clear the buffer of the packets for this transmission
END
Retry fresh RREQ (request reply) from source to destination
```

3.4. Buffer Scheme

Normally a node should forward the packet towards the destination node. But, here the topology is dynamic; a node might have changed its position, and thus leading to a broken route to destination. Under this condition we recommend to buffer the packet in the node. Now the problem that needs to be resolved is the restriction of the buffer size. Due to advancement in the technology, where the amount of storage per unit of area has increased to hundred times, the use of large storage capacity can absolutely be discovered. Assuming the travel rate of a bit to destination at the speed of light i.e. 3x10^8 m/sec., and the distance to next node as 50 meters so a bit will take 1.66667E-07 seconds to reach the destination. Now if the route breaks and a fresh route discovery is needed, then it take will take "N x1.66667E-07 x M" seconds to discover the route to destination, here N is the number of hops to destination and M is the RREQ (request reply) packet size. For a typical case if N=12, and RREQ = 2KB, then it takes 12 * 2 * 1.66667E-07 * 1024 = 0.0020480 seconds or say 2.04 milliseconds to discover the route usually increases with increase in the number of packets in the buffer. To get an approximation of the waiting time in the buffer, we assume that there are N sources that may send request in the form of control and information packets. If n request are made each succession and B of these requests are honored [5] each succession, then n – B of these requests are delayed. We undertake that the waiting time for the B honored requests is negligible (say 0), while the waiting time for the remaining n – B requests are Ts (i.e. the service time). Thus, the waiting time per request is $Tw = [(n - B) / B] * Ts$

We undertake that m (no. of memory module is=1 per node), the expected buffer size equals size of the closed queue $Qc = (n - B)$. The waiting time Tw can now be modified as:

$Tw = [\{(n - B) / m\} * (m / B) * Ts]$

Putting m= 1; the above equation simplifies to:

$Tw = (n - B) / B) * Ts$

The Total Service Time $Tt = Ts + (n - B) / B) * Ts$

$= [Ts (B + n - B)] / B$

$= nTs / B$

Thus the total amenity time may be reduced either by decreasing the service time (Ts) or improving the per succession request handing capability of the nodes.

4. Conclusion and Future Scope

The dynamic nature of MANET imposes a great challenge in ensuring the error free transmission. In MANET, Links fail often leading to packet loss or delay in transmission. The concept present in this paper will not only improve the route discovery, route maintenance process with minimum route breakages but also improve the performance by using the buffering technique. This paper also proposes data buffering technique which will improve the routing performance when link failure occurs. The present day technology has made the availability of low powered, small and fast memory chips which could be employed in the device. These small size and faster memory will make the space available for buffering the data. As the packets are buffered, a fresh route may be discovered, and the buffered packet may resume transmission. Though the work done till yet emphasizes improving the MANET performance and route maintenance so that the data transmission may be continuous, it may also add some cost but the benefit of continuous and less control overhead will motivate its usage. So there is a lot of scope for future research.

References

[1] S. Balfe and S. Reidt. Key Deactivation Strategies in MANETs – A

[2] S. Buchegger and J.-Y. Le Boudec. A Robust Reputation System for P2P and Mobile Ad Hoc Networks. In Proceedings of the Second Workshopon the Economics of Peer-to-Peer Systems, (P2PEcon 2004), pages 403– 410. Harvard University Press, June 2004.

[3] L. Butty´an and J.-P. Hubaux. Stimulating cooperation in self-organising mobile ad hoc networks. ACM/Kluwer Mobile Networks and Applications(MONET), 8(5):579–592, October 2003.

[4] H. Chan, V.D. Gligor, A. Perrig, and G. Muralidharan. On the Distribution and Revocation of Cryptographic Keys in Sensor Networks. IEEE Transactions on Dependable and Secure Computing, 2(3):233– 247, 2005.

[5] Pirzada, Asad Amir, and McDonald, Chris. Establishing trust in pure ad-hoc networks. Proceedings of the 27thconference on Australasian Computer Science - Volume 26, pp 47-54, 2004

[6] The odorakopoulos, George and Baras, John. Trust evaluation in ad-hoc networks. Proceedings of the 2004 ACM workshop on Wireless security, pp. 1-10, 2004.

[7] Michiardi, P. and Molva, R., "Core: A Collaborative Reputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks", Communication and Multimedia Security 2002 Conference.

[8] Albers, Patrick and Camp, Olivier. Security in Ad hoc Networks: a general Intrusion detection architecture enhancing trust based approaches. Proceedings of the First International Workshop on Wireless Information Systems2002

[9] R. Ismail, C. Boyd, A. Josang, and S. Russell. A security architecure for reputation systems. In K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, editors, E-Commerce and Web Technologies - 4th International Conference,EC-Web 2003, Prague, Czech Republic, September 2-5, 2003,pages 176–185. Springer-Verlag (LNCS 2738), Sep 2003.

[10] ISO/IEC 11770-1:1996. Information technology – security techniques – key management – part 1: Framework, 1996.

[11] B. Raghavan and A. C. Snoeren. Priority forwarding in ad hoc networks with self-interested parties. In Proceedings of the First Workshop on Economics of Peer-to-Peer Systems, Berkley, CA, June 5-6, 2003, June2003.

[12] D. Roberts, G. Lock, and D.C. Verma. Holistan: A Futuristic Scenario for International Coalition Operations. In Proceedings of Fourth InternationalConference on Knowledge Systems for Coalition Operations(KSCO 2007, 2007.

[13] G. Stoneburner, A. Goguen, and A. Feringa. Risk Management Guide for Information Technology Systems. Special Report 800-300, NIST, 2002.

[14] S. Zhong, Y. Yang, and J. Chen. Sprite: A Simple, Cheat-proof, Creditbased System for Mobile Ad Hoc Networks. In Proceedings of the 22ndAnnual Joint Conference of the IEEE Computer and CommunicationsSocieties (INFOCOM 2003), 2003. ACITA 2008

[15] Karygiannis, A. and Antonakakis, E. mLab: A Mobile Ad Hoc Network Test Bed. 1st Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing in conjunction with the IEEE International Conference in Pervasive Services 2005, July 14, 2005.