Improving of TCP Performance Evaluation in Mobile Ad Hoc Networks

Shivashankar₁, B. Sivakumar² and G. Varaprasad³

ABSTRACT: Wireless Mobile Ad-hoc networks offer challenges to TCP's congestion control mechanism related to its inability of distinguishing between losses induced by congestion and others types of losses. The performance of transport layer protocols will be a key factor in the successful extension of internet application and services to Mobile Ad-Hoc networks (MANETs). This paper uses fixed RTO interval. The TCP is the most commonly used transport for the internet, so providing a high level performance in MANETs is of particular importance. While TCP has been extensively tuned for wire line networks, in its correct form TCP does not perform well when used in MANETs. One of the services provided by transport layer protocols is to turn IP's best effort level of service into a reliable packet delivery service. IP packets may be dropped, reordered, or duplicated on the way from sender to receiver. By contrast the TCP enables applications to establish reliable, full-duplex connection. TCP includes a flow controls scheme by which the receiver can limit the rate at which the sender transmits data. TCP also implements a congestion control mechanism of to keep TCP sender from over loading the network and possible causing congestion collapse. We evaluated its performance under variety of network conditions running different class of routing protocols: DSR and AODV are reactive and DSDV is proactive. Our performance study shows the effect of different factors in isolation and in combination with each other on the TCP performance and the impact of routing protocol design approaches in the TCP stability.

Index Terms: Mobile Ad-hoc networks, TCP, full duplex, DSR, AODV, DSDV.

1. INTRODUCTION

Transmission Control protocol (TCP), the most widely used transport layer protocol on Internet, has attained significant maturity. Over the last few years the popularity of wireless communication and computing systems is on the rise. TCP was developed for wired medium and wireless medium posed an altogether new set of challenges to TCP. For this reason TCP requires improvements or modifications. Wireless MANETs offer unique challenges and opportunities to network designers and administrators. They increase the system capacity, reduce...
deployment cost as they require no supportive infrastructure and reduce administrative cost as they are self-configurable and self-adaptable. However, their inherent characteristic of frequent topology changes adds complexity to routing and transport connection management. Since TCP carries more than 80% of Internet traffic and support most widely used applications, its consistent and stable performance is desirable for application service delivery in MANET.

Although TCP provides reliable end-to-end delivery of data over wired networks, several recent studies have indicated that TCP performance degrades significantly in MANET[1-3]. Since the root of the TCP performance problem in a MANET is its inability to distinguish between losses due to congestion and other types packet losses, designing a mechanism by which TCP can determine that a loss was not caused by congestion and thereby avoid congestion control, seems to be an obvious means of solving the problem. If the TCP sender is notified that a route failure has occurred, it can suspend its normal response to packet drops until the route has been reconstructed. So fixed RTO interval improves the TCP performance by both reactive and proactive protocols such as DSR, AODV and DSDV respectively.

The rest of the paper is organized as follows. In section II, some previous work related to this paper and some applications are identified. In Section III we present the different mechanisms with algorithms. Section IV deals with the both reactive and proactive protocols such as DSR, AODV and DSDV respectively. Section V includes the simulation set up for the performance evaluation and justifies the choice of simulation parameters. Section VI contains the simulation results for single and multiple TCP connections in a variety of MANET scenarios, analyzing and comparison of TCP performance and also it includes the discussion on the performance by the various TCP agents in our simulations. Finally, Section VII concludes the paper and offers suggestions for future work.

2. SOME OF THE RELATED WORKS AND APPLICATIONS

2.1 Related Work

MANETs are used for emergency situations like disaster - relief, military applications, and emergency medical situations. These applications make MANETs attractive targets for cyber attacks and make the development of counter measures paramount. The study of worm behavior is critical to the design of effective counter measures in MANET environments. This paper presents numerical solutions for the models and verifies the results using high fidelity packet level simulations. Today, TCP is widely used in the current Internet. Hence, using TCP to provide reliable end-to-end message delivery in MANETs is necessary in order to achieve a smooth integration with the wired Internet. TCP-Vegas are a well known TCP variant which takes into account network traffic conditions [4]. Unlike other TCP variants, TCP-Vegas do not use packet loss as the indication of network congestion. Instead, it can detect presence of incipient network congestion earlier than other TCP variants by analyzing transmission rate in accordance to the variation in estimated RTTs [5-6].

TCP attributes packet loss on the network to congestion. TCP sender buffers the packets sent to the receiver. The receiver sends cumulative acknowledgments to indicate the receipt of the packets. The sender retransmits the lost packets to guarantee reliable delivery. For this purpose, it maintains a running average of the estimated round trip delay and the mean linear deviation from it. The sender identifies the loss of a packet either by the arrival of several
Improving of TCP Performance Evaluation in Mobile Ad Hoc Networks

duplicate cumulative acknowledgments or by the absence of an acknowledgment for a packet within a timeout interval that is equal to the sum of the smoothed round-trip delay and four times its mean deviation. TCP reacts to packet losses by dropping its transmission window size before retransmitting packets, initiating congestion control or avoidance mechanisms and backing off its retransmission timer. These measures result in a reduction in the load on the intermediate links, thereby controlling the congestion in the network. Earlier research work by Ramarathinam et al. [7] examined the goodput performance of TCP Tahoe, Reno, NewReno and SACK in a static multi-hop network and noted that overall Reno is the best performer. Work by Xu et al. [8] examined the behavior of different TCP variants in the context of hidden terminal interference but only in the case of string topologies.

In the same work the apparent performance merit of Vegas was noted in string topologies and a simple modification was suggested to improve goodput for the Reno TCP variants. Finally, an evaluation of TCP variants over wireless networks has been performed by Grieco et al. in [9], where the relative merits of TCP Westwood over Vegas and NewReno were discussed albeit without any mobility considerations. TCP utilizes a sliding window algorithm to guarantee reliable, in order packet delivery and to enforce flow control. The sender can have at most windows worth of outstanding packets that are packets for which no acknowledge of their receipt has yet arrived from the TCP receiver. Since the transmission of new packets must wait for the acknowledge of previously transmitted packets, TCP is said to be self clocking. The size of the sender's window can never exceed the receiver's advertised window size. In addition, TCP keeps track of another parameter called congestion window size which is increased or decreased based on the flow ACKs. We should point out that TCP keeps track of bytes, not packets, so that window size is actually expressed as a number of bytes. TCPs acknowledge(ACK) are cumulative meaning that the sender is guaranteed that all packets up to the and including the packet being ACKed have been successfully delivered. If a packet is delivered out of order, the receiver resends the same ACK it sends previously. When the receives this duplicate ACK, it knows that a packet has left the network, but that the left hand side of its window can not yet be advanced.

2.2 Applications in Network

In Indirect-TCP(I-TCP), the transport layer connection between the mobile host(MH) and the fixed network is split into two connections such as one between the MH and its Mobile Support Router(MSR) over the wireless link and another between the MSR and the Fixed Host(FH) on the fixed network. Mobile-TCP(M-TCP) intends to improve TCP performance over Wireless networks by preventing the sender from shrinking the congestion window when the packets are lost either due to bit-error or due to disconnection. Wireless-TCP is an end-to-end reliable transport layer protocol designed to operate effectively over Wireless Wide Area Networks(WWAN). A typical WWAN is characterized by low and highly variable throughput, high and highly variable latency, bursty and random packet losses unrelated to congestion and occasional blackouts. Thus the standard transmission control protocols such as TCP perform poorly under these conditions. Standard TCP is a window based protocol, where transmission rate is determined by packet loss and retransmission timeout.
3. MECHANISMS FOR PERFORMANCE EVALUATION OF TCP

3.1 TCP Performance Mechanism in Single-hop Wireless Network

A number of schemes are proposed to mitigate the ill effects caused non congestion related losses on TCP performance in 1-hop wireless networks. These networks are often characterized by sporadic bursts of bit errors and temporary brakes in connectivity during handoffs. Split connection protocols divide the TCP connection into two connections, a wired connection from the sender to base station and a wireless connection from a base station to the receiver. Loss recovery over the wireless link is separated from that across the wired network, and so is hidden from the sender. Because ACKs can reach the source before the data packet arrives at the mobile host, split-connection protocols violate the end to end semantics of TCP acknowledgements. Another disadvantage of this scheme is the over head of maintaining TCP state at the base station, which tend to make handoffs complicated and slow. While the split connection approach does insulate the TCP sender from wireless losses and time outs on the wireless link cause the sender to stall frequently, resulting in poor end to end throughput.

A link layer approach evaluated was the snoop protocol. In the snoop method, a TCP aware agent at the base station catches the packets sent across the wireless link until they are ACKed so that local recovery is possible in the event of packet loss. By shielding the TCP sender from duplicate ACKs caused by wireless losses, the snoop protocol yielded increase of 10% to 30% in throughput compare to a link layer protocol with no knowledge of TCP. The end to end scheme consider were selective acknowledge and the addition of an explicit loss notification option to TCP acknowledgements. Allow the sender to handle multiple losses within a window more efficiently. Explicit loss notification notifies the sender that a non congestion related loss has occurred so the sender can retransmit the lost packet without invoking congestion control. Two different SACK schemes were consider: a simple version of the SMART proposal and an implementation based on RFC 2018. Compare to TCPReno, the RFC 2018 SACK implementation yielded an increased in throughput of approximately 30%, while the fraction of packets successfully delivered (good put) remained the same. The use of ELN increased throughput by about 25%, and again, good put was unchanged.

3.2 TCP Layer Mechanism

Even though MANET routing protocols are designed to repair broken quickly, route failure is soften a source of packet delay and packet drop. Significant reordering of packets may occur as well. This suggests that mechanisms designed to avoid to unnecessary retransmits time outs, packet retransmission, and reductions in the congestion in window, may be promising means of improving TCP performance. TCP layer methods will generally require changes to the TCP sender, sender receiver, or both.

SACK: In the Reno version of TCP, a duplicate ACK does not tell the sender which packet or packets are still outstanding, only that one or more packets have been dropped or delayed. The best the sender can do is retransmit at most one missing packet trip time (RTT) or risk transmitting a packet that may already been received. The selective ACK option allows the receiver to specify which non-consecutive packets have been received. Frequent route changes in a MANET can be expected to cause packet losses, and these losses may result in duplicate ACKs that trigger the TCP sender’s fast retransmit mechanism.
2 Del ACK: In networks where bandwidth is a scarce commodity, it makes sense to reduce traffic possible. The use of delayed ACKs is expected to help by reducing the volume of ACK traffic in normal network conditions. In MANETs, delayed ACKs are also useful when routes break. During route reconstruction, data packets may be sitting in a queue at the source or at an intermediate node. If the route is repaired quickly enough that the data packets have not yet been flushed from the buffer, then all of these packets will arrive at the TCP receiver in quick succession. With delayed ACKs, the receiver is able to send fewer acknowledgements for these packets. This will turn, enable the TCP sender to increase its congestion window more quickly and retransmit fewer packets.

3.3 Routing Layer Mechanism

These techniques require routing agents at TCP endpoints or in intermediate nodes to provide network information that is not directly available to TCP. If only the routing agents at the endpoints of a TCP connection provide additional feedback to TCP, then such mechanism are classified as routing layer. If other routing agents are also involved, then such mechanisms are classified as Routing agents.

1. Proposed TCP sender side option: The TCP sender is designed to adapt to network condition by tracking the TCP acknowledgements return by the receiver. The timing of these ACKs and the order in which they are received are clues that the sender uses to infer what is occurring elsewhere in the network. How these clues are interpreted by the sender has a major impact on TCP performance. In MANETs, new heuristics can be used to elicit sender behavior i.e. conductive to good TCP performance. In this paper we are using two mechanisms that use the hold down timer and the fixed RTO.

2. Hold down timer: Sometimes the packet delay, as opposed to packet loss, due to a route failure can cause an ACK to arrive the retransmit timer have already expired. The same thing can happen for ACKs buffered during route repair. Once the route has been reestablished, these ACKs will arrive at the TCP sender in quick succession. If the retransmit timer has expired, the packet corresponding to the first of the ACKs will already have been retransmitted. If the ACKs arrive in order, the first one will trigger the retransmission of the packets corresponding to the second ACK, the second one will trigger yet another retransmission, and so on. Since the RTO is doubled following a timeout, the hold down period is one half of the new timeout period. The first packet in the window is immediately retransmitted as usual. During the hold down interval, any incoming ACKs are processed in the normal way except that they do not trigger further packet transmissions. If the hold down timer expires before the packet in the window has been acknowledged, then TCP reverts to its normal behavior during the second half of the retransmit timeout period, i.e. packets will be retransmitted when the outstanding ACK is received. Otherwise, any packets that are ACKed during the hold down interval will not have to be retransmitted.

3. Fixed RTO interval: In the event of a retransmit timeout, TCP retransmits the oldest unacknowledged packet and doubles the retransmit timeout interval (RTO). This process is repeated until an ACK for the retransmitted packet has been received. This exponential backoff of the RTO enables TCP to handle network congestion gracefully. However, in MANET, the loss of packets (or ACKs) may be caused by temporary route loss as well as network congestion.
Since routes are likely to be broken frequently in high node mobility environments, routing algorithms for MANETs are designed to repair broken routes quickly.

4. ROUTING IN MANETS

Routing protocols for mobile ad hoc networks can be broadly categorized as: (i) On-demand protocols, and (ii) Link state protocols. In On-demand routing protocols a source learns the routes when it is ready to send a packet to the destination. Depending upon caching the route remains valid until the destination becomes unreachable or cache times. In contrast a node running link state protocol maintains routing tables containing routes to every other node in the network. All nodes periodically update their tables to maintain a consistent and up-to-date view of the network.

4.1 Dynamic Source Routing (DSR)

DSR employs two mechanisms: (i) Route Discovery mechanism to discover the route for the first time, and (ii) Route Maintenance mechanism to repair an existing route. During route discovery the source broadcasts a route request packet, which contains a route record that is a list of nodes forming the route. Every node includes its address in the packet and broadcasts the packet further unless the node finds itself to be the target of the route request or has the requested route in its cache.

4.2 Destination-Sequenced Distance Vector (DSDV)

In DSDV, every node in the network maintains a routing table of route entries that contain the destination prefix, the next hop to the destination, sequence number generated by the destination, and metric of the path. Nodes advertise routing information by broadcasting routing table update packets periodically or immediately after the network topology changes. DSDV uses hop count as the route metric.

4.3 Ad Hoc On-Demand Distance Vector (AODV)

This routing protocol is intended for use by mobile nodes in an ad hoc network. It offers quick adaptation to dynamic link conditions, low processing and memory overhead, low network utilization, and determines unicast routes to destinations within the ad hoc network. It uses destination sequence numbers to ensure loop freedom at all time simulation environment.

5. SIMULATION SETUP

For our simulation, we used the Network Simulator (NS-3.9) with mobility extension. These extensions include the modeling of an IEEE 802.11. To evaluate the effectiveness of various TCP heuristics, we used three routing protocols: two on demand protocols, AODV and DSR, and one proactive protocol, DSDV. We simulated an ad hoc network comprised of 60 mobile nodes on 1100m × 1100m area. The nodes move according to a mobility pattern based on the random way point model. To avoid clustering of nodes in the middle of the field, we analyzing a node reaching an edge of the field to wraparound and continue its movement in the same direction from the opposite edge of the field. Since a MANETs performance is sensitive to movements patterns, 60 different mobility patterns are simulated and averaged for each data
point presented in the plots. Node speeds are uniformly distributed between 0 m/s and 60 /ms, yielding a mean node speed of 30 m/s. Table 1. Shows the network simulator parameter used in the NS-3.9.

Table 1: Simulator Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation length in seconds:</td>
<td>134.2064596</td>
</tr>
<tr>
<td>Number of nodes:</td>
<td>60</td>
</tr>
<tr>
<td>Number of sending nodes:</td>
<td>5</td>
</tr>
<tr>
<td>Number of receiving nodes:</td>
<td>5</td>
</tr>
<tr>
<td>Number of generated packets:</td>
<td>9510</td>
</tr>
<tr>
<td>Number of sent packets:</td>
<td>9468</td>
</tr>
<tr>
<td>Number of forwarded packets:</td>
<td>890</td>
</tr>
<tr>
<td>Number of dropped packets:</td>
<td>64</td>
</tr>
<tr>
<td>Number of lost packets:</td>
<td>41</td>
</tr>
<tr>
<td>Minimal packet size:</td>
<td>1060</td>
</tr>
<tr>
<td>Maximal packet size:</td>
<td>10600</td>
</tr>
<tr>
<td>Average packet size:</td>
<td>1060</td>
</tr>
<tr>
<td>Number of sent bytes:</td>
<td>10036090</td>
</tr>
<tr>
<td>Number of forwarded bytes:</td>
<td>943400</td>
</tr>
<tr>
<td>Number of dropped bytes:</td>
<td>67840</td>
</tr>
<tr>
<td>Packets dropping nodes:</td>
<td>0 10 16 19 25 30 36 39 52</td>
</tr>
</tbody>
</table>

We simulated the steady state conditions of a network with various background traffic loads generated by 20 and 60 Constant Bit Rate (CBR) connections. The TCP packet size is 1060 bytes, and the maximum size of both the send and receives windows is 8. In each simulation run, we measure connect time, throughput and goodput. Connect time is the time taken to deliver the first TCP packet. The overhead includes the routing of the background CBR traffic. For DSR, the numbers of bytes of routing data transmitted include the routing information carried by data packets. We also measure the number of routing packets transmitted per second at the MAC layer, including all the IP layer routing packets and the RTS, CTS and ACK control exchanges the packets used for transmitting data routing packets.

6. SIMULATION RESULTS, ANALYSIS AND COMPARISON OF TCP PERFORMANCE

For each simulation run a TCP connection is set between two randomly selected nodes and an FTP transfer session is initiated for the duration of the simulation. As such, there is a single source of TCP regulated traffic with an infinite backlog in the network. The performance metric is goodput which is defined as the number of packets successfully transmitted by the sender for which an ACK has been received. Retransmissions do not contribute to the metric i.e. each packet’s contribution to the advancement of the sliding window is only measured once.
We perform a series of five simulation runs. Each simulation run tests a different technique or combination techniques: TCP Reno, Reno with SACK, Reno with SACK and delayed ACKs, fixed RTO on consecutive timeouts plus SACK and delayed ACKs, and a hold down timer plus fixed RTO and SACK and delayed ACKs in each run, a set of performance measurements are made for each of the three routing protocols at each of several background traffic loads from 20 CBR connections and from 60 CBR connections. The fixed option improves the throughputs of DSR significantly. Since each data packet must carry the path to destination in DSR routing, nodes snoop and learn routes. With a fixed RTO option, a TCP sender transmits data packets more frequently in the event of a broken, which in turn forces the routing layer to learn fresh routes using route discovery. The fixed RTO improves the AODV’s performance. The hold down timer option improves the TCP throughputs for DSR, but not significantly for DSDV or AODV.

6.1 Throughput

Throughput is the rate at which a network sends or receives data. It is a good channel capacity of network connections and rated in terms bits per second. And mathematically it is defined by

\[ T_p = \frac{P_a}{P_f} \]  

Where \( T_p \) is the throughput, \( P_a \) is the packets received and \( P_f \) is the amount of forwarded packets over a certain time interval.

The highest throughput achieved is 0.39 Mbps with 20 CBRs and 0.37 Mbps with 60 CBRs, when all 4 options are used. Looking at the results for the case of 60 CBR connections, both AODV and DSR’s throughput are improved by 40-50%. Fig.1, 2, 3, 4, 5, 6. Show the throughput of different receiver and sender details.

![Fig. 1: Throughput for 10 TCP Connections with 200-Kbps Background Traffic Generated by 60 CBR Connections.](image)
Improving of TCP Performance Evaluation in Mobile Ad Hoc Networks

Fig. 2: Throughput for 20 TCP Connections with 200-Kbps Background Traffic Generated by 60 CBR Connections

Fig. 3: Throughput for TCP Reno with a 200 Kbps 20 CBR Background

Fig. 4: Goodput Overhead for TCP Reno with a 200kbps 20 CBR
Fig. 5: Throughput for TCP Reno-F Connection with 60 CBR Background

Fig. 6: CBR Packet Latencies for TCP Reno-F with 200 Kbps

6.2 Goodput

The Goodput ratio indicates the efficiency of a given combination of transport and routing protocols in delivering data. All three protocols exhibit very high goodput, ranging from 97% to 99%. A 30-38% increase in the MAC layer overhead is observed for DSR.

6.3 Performance Comparison

Keeping the number of CBR connections 20 or 60 and varying background traffic from 60 Kbps to 200 Kbps, reduced TCP throughputs for all 3 routing algorithms by 40-50%. Also DSDV provides the highest throughput among the three for all combinations of traffic loads, number of CBR connections of transport protocols with 60 CBRs generating 200 Kbps traffic load and Reno as the transport protocol. DSDV provides 85-100% highest throughput then AODV and DSR compare to on demand protocols; DSDV does a much better job of handling the background CBR flows in terms of packet delivery fraction. With 40 TCP traffic sources, DSDV delivers 80-85% of the CBR packets compare to about 70 % for AODV and 60% for DSR. For only 2 TCPs, DSDV achieved a delivery fraction above 90% for 20 CBR flows and in excess of 95% for 60 CBRs.
Improving of TCP Performance Evaluation in Mobile Ad Hoc Networks

When the fixed-RTO technique is employed the performance difference of the 3 protocols tended to be minimized for 20 TCPs with a 20 CBR background. DSR throughput is 10% higher than DSDV throughput. Consequently as the volume of CBR traffic increases, the impact of the background load on TCP throughput is largest for DSDV with 60 CBR connections. DSDV continues to provide better throughput than AODV and DSR which shown nearly identical performance shown in Fig. 6.

Fig. 7: Throughput of Receiving Bits Versus Simulation Time

Fig. 8: Average Simulation Jitters Versus Throughput of Receiving Bits

Fig. 9: Average End to End Delay Versus Throughput of Sending Bits
7. CONCLUSION

In this study, we simulated TCP sessions in MANET running two different classes of routing protocols - DSR and AODV representing on-demand and DSDV representing link state protocol, under a variety of network conditions. Our main conclusions are as follows: (1) Throughput. (2) Goodput (3) Packets out of order ratio and packet loss ratio. We have combined fixed RTO with TCP’s selective acknowledgements and delayed acknowledgements options to form the TCP Reno-F protocol. Reno-F works well with routing protocols having route discovery mechanism that can be stimulated by an increased rate of packet transmissions. Route stimulation can be useful in situations where route failures are frequent, or where route repairs are difficult. Among the three different routing algorithms we used in the study, the proactive DSDV protocol out performance both AODV and DSR in almost all cases except for the combination to Reno-F with a large number of TCP connections. The performance DSDV is particularly noteworthy when TCP Reno with no enhancement technique is used. While AODV and DSR falter owing to high routing overhead or stale routes, DSDV performance very well. It indicates that for consistently high performance, some form of proactive route maintenance should be considered to compliment the route discovery used in on-demand routing protocols. This paper also concludes that DSDV routing protocols works better for smaller networks but not for larger networks and DSR and AODV routing protocols are best suited for general mobile ad-hoc networks as they consume less bandwidth and lower overhead when compared with DSDV routing protocol.

REFERENCES


Improving of TCP Performance Evaluation in Mobile Ad Hoc Networks


