Module No.

vARIANTS OF tcp OPTIMIZED FOR WIRELESS NETWORKS

# LEARNING OBJECTIVES

* Recap of previous module
* Limitations of I-TCP and Snooping TCP and M-TCP
* Fast retransmit protocol
* Freezing protocol
* Selective retransmission
* Transaction oriented TCP
* Comparison of all the variants of TCP

# INTRODUCTION

We have seen in previous modules that standard TCP cannot be replicated in wireless environment because standard TCP assumes only congestion as reason for packet loss and goes into slow start by lowering the rate at which the packets are sent. This decreases the throughput. Whereas in wireless and mobile environments there are many reasons of packet loss other than congestion like call blocking improper handover, high bit error rate , disconnections and many more. Applying standard TCP here would degrade the performance. Many protocols have been optimized as per the wireless environment. Each protocol has its strengths and weaknesses and caters a particular problem of wireless scenarios. In the previous modules we learnt about three protocols namely I-TCP, Snooping TCP and M-TCP. All are split connection protocols and divide the end-to-end TCP connection into two parts via an intermediate host. The connection between fixed host and intermediate host has standard TCP whereas optimized TCP resides on link between intermediately host and mobile host. In I-TCP the intermediate hosts act as proxy which takes packets from fixed host, gives acknowledgement and then forwards the packet to mobile host and same thing is repeated on other side. This completely loses end-to-end semantics of TCP because the packets are acknowledged even before they are delivered. The situation is worst when the mobile moves to new network. All the packets buffered for the mobile device needs to be forwarded. Moreover ,whole scheme fails when the intermediator crashes. As a revision to this scheme, snooping TCP was proposed in which the intermediator just caches the packets from fixed host towards the mobile host at the same time it snoops the flow of acknowledgements and packets header. In case of missing acknowledgements, it forwards the cached packet to the mobile host. The scheme retains end-end semantics. Both I-TCP and Snooping TCP handles packet losses due to bit error rate and do not work well when packets are lost due to frequent disconnections. A revision to these schemes which deals with the problem of periodic disconnection was proposed as M-TCP. The proposed solution is to send the sender into persist mode when the mobile device goes into disconnection state. This is done by shrinking the window of the sender.

All these schemes do not take into account the problems such as packet loss due to improper handover or due to lengthy disconnections. In this module we will discuss four more protocols devised to handle these issues in mobile computing environment. These protocols are :

* Fast retransmit/ Fast recovery
* Timeout/Freezing
* Selective retransmission
* Transaction oriented TCP

Next sections consists of discussion about these protocols

# Fast retransmit/ Fast recovery

This protocol is designed for situations when mobile phone moves from one network to another network. As with Mobile IP technology, it comes under new foreign agent. Packets needs to handed off to new foreign agent. Improper Hand-offs may result in temporary loss of packets to MH because with non-overlapping cells there may be a small disconnection before mobile connects to new base station MH and old BS may attempt to send packets to each other, during this period, it may result in loss of packets. The existing approaches cannot handle this situation because of reasons given below:

**Behavior of existing protocol incase of improper handoff**

* In I-TCP approach hard state at base station must be moved to new base station or foreign agent.
* In Snoop protocol soft state need not be moved but while the new base station/foreign agent builds new state, packet losses may not be recovered locally.
* M-TCP avoids reduction of congestion window due to handoff .When host moves, route changes, and new route may be more congested. therefore it is not wise to start transmitting at full window after handoff.

**Issues during handoffs**

During the long delay for a handoff to complete a whole window worth of data may be lost. After handoff is complete, acks are not received by the TCP sender. As a result, sender eventually times out, and retransmits;If handoff still not complete, another timeout will occur . This is a serious performance penalty as time is wasted until timeout occurs and Window gets shrunk after timeout.

**Fast retransmit behaviour of TCP**

Duplicate ACKs for the same packet are received implies that all the packets upto the acknowledged packet has been received . This implies that receiver is receiving something and packet loss is not due to congestion, it is due to transmission error . Hence missing packet can be sent before timer expires(fast retransmit) and No initiation of slow start is required. Sender transmits with the current window size(fast recovery).

**Fast retransmit/fast recovery protocol takes advantage of this behavior of TCP. I**dea was presented by Caceres in 1995. The idea is to forcefully impose fast retransmit behavior when it moves from one foreign agent to another. Artificially force fast retransmit/fast recovery.

Mobile host is receiver: When mobile host enters new foreign agent area, it sends duplicate ACKs .The algorithm proposed three duplicate ACKs . When Fixed host receives duplicate ACKs, it goes into fast retransmit mode. It continues transmitting with same speed.

**Advantage**

* Simple to implement
* Minor changes in mobile host software required
* No need to change foreign agent or fixed host

**Disadvantage**

* More cooperation between mobile IP and TCP layer is required.
* Change in one influences the other Insufficient isolation of packet losses.
* Retransmitted packet crosses whole network between fixed host and mobile host
* Packet losses only due to improper handovers are considered

# Transmission/timeout freezing

This algorithm is used in a situation when Mobile hosts can be disconnected for a longer time. For eg. when a mobile phone enters a tunnel or a disconnection occurs due to overloaded cells etc. In such a situation TCP disconnects after time-out completely. The proposed protocol uses the fact that MAC layer is often able to detect interruption in advance .MAC layer knows reason for disruption. MAC can inform TCP layer of upcoming loss of connection. If interruption is anticipated, early both mobile host and fixed host can be informed .Reason of disruption is also informed to prevent triggering of slow-start When TCP is informed of congestion, all the timers and current window states are freezed .TCP sender stops sending . When MAC detects connectivity again, it signals TCP about it. Transmission resumes from where it stopped. Timers do not advance because of freezing. No timeout occurs and TCP does not go into slow start.

**Advantages**

* Resume connection after long disconnection
* Can be used with encrypted data

**Disadvantages**

* Both fixed host and mobile host needs to be changed
* Dependence on MAC layer for information of interruption
* If any encryption mechanism uses time dependent encryption schemes, freezing would not help.

# Selective retransmission

TCP acknowledgements are often cumulative. TCP acknowledges in-sequence receipt of packets up to certain packet. If a single packet is lost, whole packet sequence beginning at the gap has to be retransmitted (go-back-n), thus wasting bandwidth.

To solve this issue, selective retransmission was presented as solution in RFC 2018. It allows for acknowledgements of single packets. Sender can now retransmit only the missing packets.

**Advantage**

* much higher efficiency
* Requires less bandwidth
* Helpful in slow wireless links

**Disadvantage**

* more complex software in a receiver, more buffer needed at the receiver to resequence the packets

# Transaction oriented TCP

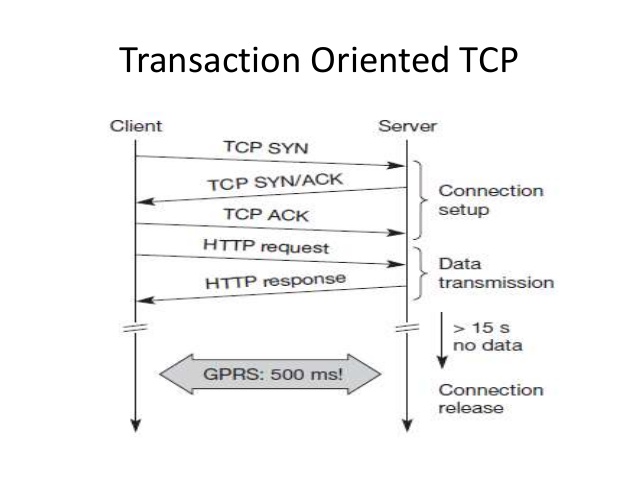
A TCP connection consists of three phases: setup, data transmission and connection release using 3-way-handshake. It needs 3 packets for setup and release, respectively. This overhead is minimal and acceptable when connections are with large traffic or long duration connection. But if the connections are short, it is overhead. Because for even short messages a minimum of 7 packets are needed. To solve this issue, T-TCP was proposed by Barden in 1994. It was documented in RFC 1644. The proposed proposal combines connection establish and release phases with data packets. Hence number of packets reduces to two instead of seven.

**Advantages**

* Reduces overhead

**Disadvantages**

* Needs changes at mobile host and fixed host
* Mobility is not hidden
* Security issues



|  |  |  |  |
| --- | --- | --- | --- |
| **Approach** | **Mechanism** | **Advantages** | **Disadvantages** |
| Indirect TCP | splits TCP connection into two connections | isolation of wireless link, no change in TCP or fixed host | loss of TCP semantics, higher latency at handover,securitythreat,lack of robustness |
| Snooping TCP | “snoops” data and acknowledgements, local retransmission | transparent for end-to-end connection, MAC integration possible, TCP semantics maintained | problematic with encryption, bad isolation of wireless link, no transmission during period of disconnection while handovers |
| M-TCP | splits TCP connection, chokes sender via window size | Maintains end-to-end semantics, handles long term and frequent disconnections | Bad isolation of wireless link, processing overhead due to bandwidth management |
| Approach | Mechanism | Advantages | Disadvantages |
| Fast retransmit/ fast recovery | avoids slow-start after roaming by forced retransmission | simple and efficient | mixed layers, not transparent |
| Transmission/ time-out freezing | Mac layer notifies of the disruption,freezes TCP state at disconnect, resumes after reconnection | independent of content or encryption, works for longer interrupts | changes in TCP required, MAC dependant |
| Selective retransmission | retransmit only lost packets | Lower bandwidth requirement | slightly more complex receiver software, more buffer needed |
| Transaction  oriented TCP | combine connection setup/release and data transmission | Efficient for certain applications | changes in TCP required, not transparent |