2.5.3 Comparative study between TCP and UDP

TCP is a connection-oriented protocol, which means that the end-to-end com-

munications is set up using hand- shaking. Once the connection is set up, user

data may be sent bi-directionally over the connection. Compared to TCP, UDP

is a simpler message based connectionless protocol, which means that the end-to-

end connection is not dedicated and information is transmitted in one direction

from the source to its destination without verifying the readiness or state of the

receiver. TCP controls message acknowledgment, retransmission and timeout.

TCP makes multiple attempts to deliver messages that get lost along the way, In

TCP therefore, there is no missing data, and if ever there are multiple timeouts,

the connection is dropped. When a UDP message is sent there is no guarantee

that the message it will reach its destination; it could get lost along the way.

There is no acknowledgment, retransmission, or timeout. If two messages are

sent in sequence, the first message will reach the destination first. When data

segments arrive in the wrong order, TCP buffers delay data until all data can be re-ordered before being delivered; when using UDP the order in which messages

arrive cannot be predicted.

The TCP protocol has extensive algorithms to ensure correct delivery of the

data. Having two TCP connections stacked together will thus force the algo-

rithms of both TCP connections to work in parallel. TCP was not designed

to work this way and problems are likely to occur in di\_erent situations. The

retransmission problems, TCP meltdown and double retransmit, are problems

caused by tunneling TCP in TCP. The problems can occur when both of the

stacked connections are retransmitting packets. In previous work, related to

TCP in TCP tunneling, it is not entirely clear, how severe the retransmission

problems really are. TCP protocol suite featured automatic recovery from any

dropped or lost data. This protocol must be able to recover from an outage of

any host on any part of the network and at any point in a data transfer. When

TCP packets are transmitted from one end to a remote end across the network,

the data packets are reordered in the same sequence generated by the sender.

The protocol detects when segments of the data stream have been discarded by

the network, reordered, duplicated, or corrupted. The sender can even retransmit

damaged segment. This process makes TCP a reliable protocol. However, the

double retransmission creates latency.

TCP regulate retransmission, message acknowledgment, and timeout. TCP

deliver lost messages along the way upon multiple attempts. In TCP, there is no

missing data, and if ever there are multiple timeouts, the connection is dropped.

When a UDP message is sent there is no guarantee that the message will reach

its destination; it could get drop along the way. There is no retransmission,

timeout and acknowledgment. When two data packets are sent in sequence,

the first message will reach the destination first. When data segments arrive in the wrong order, TCP buffers hold the data until all data are re-ordered before

being transmitted; when using UDP the order in which messages arrive cannot

be predicted.

The design of TCP was to make an efficient protocol with low overhead, a

protocol suite that had a minimal amount of 'extra' data being transferred. This

extra data called overhead, functions as packaging for the data being transferred

and enables the data transmission. TCP tunnel is a technology that aggregates

and transfers packets sent between end hosts as a single TCP connection. By

using a TCP tunnel, the fairness among aggregated ows can be improved and

several protocols can be transparently transmitted through a \_rewall. Currently,

many applications such as Secure Shell (SSH), Virtual Tunnels (VTun), and Http

Tunnel (HTun) use a TCP tunnel. However, since most applications running on

end hosts generally use TCP, two TCP congestion controls (i.e. end-to- end

TCP and tunnel TCP) operate simultaneously and interfere each other. Under

certain conditions, using a TCP tunnel severely degrades the end-to-end TCP

performance. More speci\_cally it is known that using a TCP tunnel drastically

degrades the end-to-end TCP throughput for some time. This is called TCP

meltdown problem.