A Packet Level Steering Solution for Tightly Coupled LWIP Networks

Thomas Valerrian Pasca S, Sumanta Patro, Bheemarjuna Reddy Tamma, and Antony Franklin A
Indian Institute of Technology Hyderabad, India
Email: cs13p1002@iith.ac.in, cs15mtech01005@iith.ac.in, tbr@iith.ac.in, antony.franklin@iith.ac.in

Abstract*—Smartphones make use of either LTE or Wi-Fi interface one at a time. Radio level integration (RLI) architecture such as LTE–Wi-Fi Integration with IPSec tunnel (LWIP) enables both the interfaces to operate simultaneously. The real benefit of aggregation resides in enabling packet level steering of traffic, where the incoming packets to LWIP node will be steered across LTE and Wi-Fi links on per-packet granularity. Such packet-level steering leads to the problem of out-of-order (OOO) delivery of packets to the destination. Such OOO delivery occurs due to variable time incurred in delivering the packets through different links *viz.,* LTE and Wi-Fi. OOO delivery, as mentioned earlier, negatively affects TCP performance and triggers spurious retransmissions. In this work, we propose a novel Dynamic pseudo-In-sequence Delivery Algorithm (DIDA) at LWIP node, which mitigates the effects of OOO delivery problem without requiring any modifications at the source and destination devices. DIDA solution operates at the LWIP node and aggregates multiple radio links effectively. We have formulated the crucial parameter, holding time 't' of DIDA. We have setup a prototype of LWIP using OpenAirInterface (OAI) and experimented DIDA on packet level steering over LTE and Wi-Fi networks. DIDA doubles the TCP throughput achieved as compared to native LWIP.

I. INTRODUCTION

The proliferation of smartphones with data-hungry applications has pushed the cellular operators to find a viable solution to address the fast-growing data demand [1]. The high cost of licensed spectrum limits cellular operator from increasing their network bandwidth. The availability of huge bandwidth in the unlicensed spectrum has attracted cellular operators to venture upon. Wi-Fi operating in the unlicensed spectrum has become the key technology to carry the cellular operator data. Hence integrating cellular technology with Wi-Fi has thrived on being a feasible solution to address the cellular operator's requirements. 3GPP has proposed various LTE–Wi-Fi interworking strategies from Rel.8 to Rel.12. All the solutions developed involves interworking at Evolved Packet Core (EPC) or gateway. Such a gateway based solution incurs high signaling overhead in the core network for re-routing a flow from one network to other (Say LTE to Wi-Fi). Understanding the variation in the channel and promptly reacting for that channel variation by steering flows across LTE and Wi-Fi becomes tedious with EPC based solution. To overcome this inefficient, traditional traffic flow regulating mechanism a finer decision-making solution is required. Finer control over interface could be achieved only if the decision making entity is placed closer to the end devices. The need for finer control has pushed the interworking decision making entity from the core network to e-NodeB (eNB). 3GPP has proposed the finer level interworking architecture between LTE and Wi-Fi Radio Access Networks (RAN) as part of Rel.13 [2]. Architecture realizing the interworking at Packet Data Convergence Protocol (PDCP) layer is termed as LTE–Wi-Fi Aggregation (LWA) and the architecture realizing the interworking at IP layer is termed as LTE–Wi-Fi interworking with IPSec tunnel (LWIP).

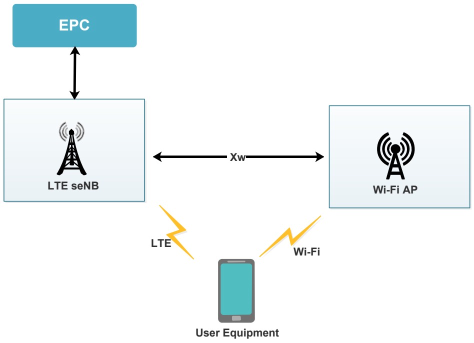


Fig. 1: 3GPP LWIP architecture.

LWA and LWIP architectures have the following merits:

- Existence of Wi-Fi AP is not known to EPC, *i.e.,* Wi-Fi AP is controlled directly by the LTE small cell (SeNB).
- Enables effective radio resource management across Wi-Fi and LTE links.
- LTE acts as the licensed-anchor point for UE's communication with the network.

LWIP could be realized in two ways, (1) Colocated LWIP (2) Non-colocated LWIP. The colocated LWIP consists of SeNB and Wi-Fi AP colocated on the same device and integrated at RAN level. In case of non-colocated LWIP, an X_W interface connects LTE SeNB and Wi-Fi AP as shown in Figure 1.

II. MOTIVATION

LWA supports both switched bearer and split bearer. In case of a switched bearer, a flow/ bearer is moved completely from LTE interface to Wi-Fi interface. In case of a split bearer, the packets within a flow are sent across LTE and Wi-Fi interfaces. The problem that arises with the split bearer/ packet level split is that the packets sent through different interfaces may be received at the destination out-of-order (OOO). Aggregating the links at PDCP layer *i.e.,* LWA has in-built reordering mechanism which ensures in-sequence packet delivery to the higher layer. The reordering mechanism in LWA follows Dual Connectivity (DC) Reordering procedure. However, the problem remains unsolved in case of LWIP because it does not have a reordering procedure. The problem of OOO packet delivery becomes challenging at IP layer since UE does not involve queuing of packets. Hence, LWIP is not

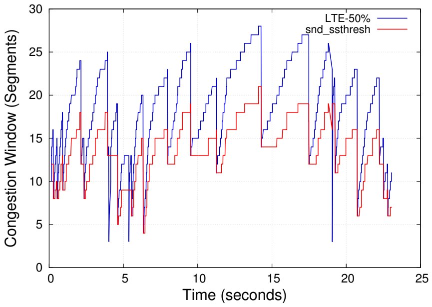


Fig. 2: Congestion Window for split ratio of 50% across LTE and Wi-Fi with 100 msec RTT for a 16MB file download.

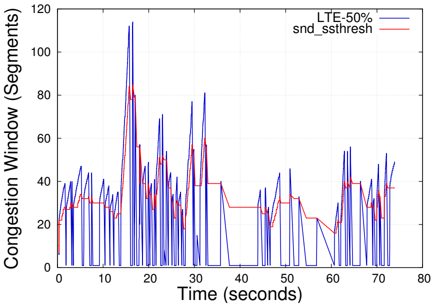


Fig. 3: Congestion Window for split ratio of 50% across LTE and Wi-Fi with 20 msec RTT for a 32MB file download.

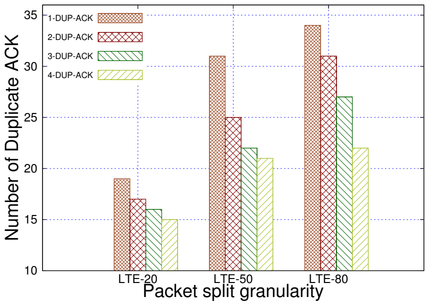


Fig. 4: Out Of Order packet arrival observed for packet split ratio of 20%, 50%, and 80% for 100 msec RTT.

suitable to perform split bearer based aggregation because of the IP layer's inability to reorder the packets. OOO packet delivery has a significant effect on TCP performance. To study the effect of packet level steering, we have employed packet steering at different granularity in our LWIP prototype; the prototype is detailed later in Section VI.

The experiment observes the growth of congestion window when round-robin based packet level steering is employed. The experiment is repeated for different round trip time (RTT) and different file sizes. This setup does not involve any congestion in the backhaul network. Figures 2 and 3 show the congestion window plot for the TCP flows downloading 16MB and 32MB with RTT as 100msec and 20msec respectively. The congestion window values are obtained by hooking on TCP parameters in Linux environment. In the experiment, 50% of the downlink packets are sent through LTE and the rest through Wi-Fi. The congestion window frequently drops for lower RTT because the number of packets received OOO at the destination is high. For every packet received OOO at the destination, a Duplicate Acknowledgment (DUP-ACK) is sent as the response to the source. The TCP source on receiving three such consecutive DUP-ACKs reduces its TCP congestion window by half in case of any AIMD based TCP congestion control algorithm.

Figure 4 captures the number of DUP-ACKs received at the sender for different packet level steering ratios. LTE-20, LTE-50, and LTE-80 correspond to 20%, 50%, and 80% of the incoming packets sent through LTE and the rest sent through Wi-Fi. In this scenario, Wi-Fi has a higher data rate than LTE, so steering the lowest fraction of packets to LTE has less number of DUP-ACKs (LTE-20). An appropriate fraction of steering traffic will also reduce the OOO packet delivery, but it cannot solve it completely.

To overcome this inefficiency in reducing the OOO packet delivery, in this work, we propose a pseudo reordering technique inspired by the packet reordering techniques detailed in Section III. We have evaluated the pseudo reordering technique using a testbed.

III. Related Work

LWIP is a fast commercializing technology, and it is a promising way forward for 1000x capacity increase using small cells. Here are some works targeted to solve OOO delivery problem in the literature. In [3], authors presented an interesting mechanism DOOR for detecting the OOO event at the sender. DOOR detects the OOO TCP ACK packets precisely by introducing an additional ordering information in TCP ACK packets.

Delayed ACK [4] mechanism introduces waiting time before the receiver generates a duplicate ACK, thus minimizing the OOO delivery. This method has a significant problem when a TCP ACK is delayed in the TCP slow-start phase, which will negatively affect the growth of the TCP window.

Other reordering techniques such as Reordering Robust-TCP [5] and TCP-Packet Reordering [6] target to prevent persistent packet reordering from contrived activating congestion response by deferring packet retransmission and congestion response till the occurrence of packet loss.

The other existing solution which readily enables aggregation of multiple radios is Multipath TCP (MPTCP) [7]. It enables multiple TCP sub-flow to be sent over different interfaces. The major challenge with MPTCP is its inability to make quick and efficient decisions to steer the packets across different subflows. Also, the steering decision is taken at the MPTCP sender. Hence, it cannot react for the fluctuations on the wireless channel quickly. In [8], the authors have developed an LWA system and presented that LWA is inefficient in aggregating LTE and Wi-Fi links due to large OOO packet delivery when the link rates of LTE and Wi-Fi are different.

All the above works concentrate on reducing the effect of OOO delivery which arises due to various factors in the network. But in case of LWIP, the packet level steering introduces OOO packet delivery at the last hop, for which none of the above mechanisms are well-calibrated. Taking the best features of all the above-mentioned algorithms and being inspired by them, we propose a novel Dynamic pseudo-In-sequence Delivery Algorithm (DIDA), which involves a network regulated reordering procedure detailed in Section IV.

A. *Contributions of the paper*

The main contributions of this paper are as follows:

- We investigate the bottleneck element for LWIP technology, which is the OOO packet delivery problem, and propose a novel DIDA algorithm that solves it in real-time.

- Optimal operation region for holding time to achieve in-sequence delivery is modeled.
- The proposed solution is evaluated using a testbed.

IV. PROPOSED WORK

LWIP suffers from OOO packet delivery problem due to ‘speed of the slowest link’. The speed of the slowest link signifies that when packets of a TCP stream are split across two interfaces, a packet sent through LTE could reach faster than Wi-Fi and vice-versa. Thus the total system throughput depends on the slowest link. An intelligent packet level steering algorithm can steer the packets across interfaces by knowing their capacities and their link qualities. Even for an intelligent packet level steering algorithm to work with LWIP, the problem of OOO delivery exists. This problem must be solved way ahead of proceeding. Hence introduce two major components to combat the problem, as shown in Figure 5, (a) A packet steering module to steer the traffic across LTE and Wi-Fi links and, (b) A novel DIDA receiver to solve the OOO packet delivery problem. The packet steering module is employed with two packet steering algorithms to observe the performance of LWIP. And the DIDA mechanism targets to maximize the system throughput. The term pseudo in DIDA refers to achieving in-sequence delivery for ACK packets (instead of actual data packets). These ACKs, if delivered OOO to TCP source node, could potentially cause a drop in TCP window growth. The data packets are buffered at TCP receiver buffer even if they are received out of order, and ACKs for corresponding packets are generated. In our experiment, TCP uses Selective ACK (SACK) to respond to the received packet.

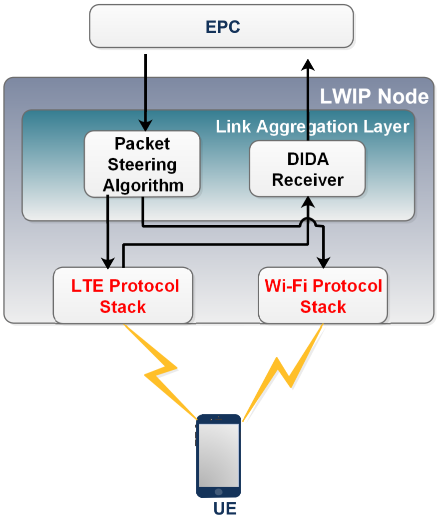


Fig. 5: Components of the proposed system.

A. Packet Steering

A packet steering algorithm would send some packets of flow through LTE link and rest through Wi-Fi link based on information about the links and decision making metrics. In this work, we consider two packet steering algorithms: (1) Steering the packets across the interface with a fixed steering ratio, and (2) Dynamic steering of packets across the interfaces based on link round trip time as a metric. TCP packets that have SYN, RST, and FIN bits set are sent exclusively through LTE after the handshake procedure. The packet level steering algorithm enables packet steering across LTE and Wi-Fi radio stacks at the IP layer. The solution adopted in this paper is a reduced form of VISIBLE [9] to enable real-time operation.

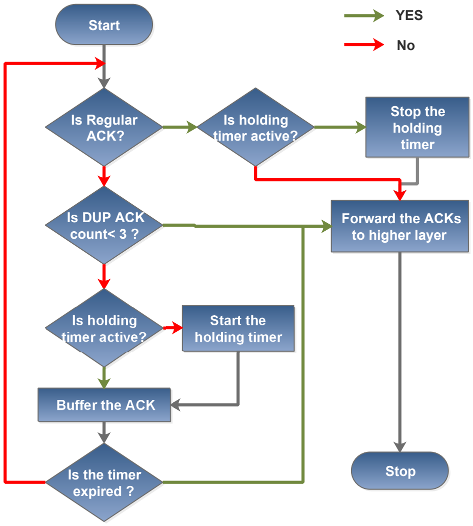


Fig. 6: Flow chart for working of DIDA.

B. Dynamic pseudo In sequence Delivery Algorithm

Working procedure of DIDA is detailed in Figure 6. DIDA runs at LWIP node; it reorders the OOO ACK packets rather than actual data packets by buffering them. DIDA does not buffer TCP handshake ACK packets *(like SYN, SYN+ACK, and RST)*, but the rest of all ACK packets which are received OOO are buffered. An ACK Sequence Number variable is used per connection to verify the ACKs are received in-order. If the ACKs are in-order then they are forwarded to TCP sender, else the ACKs are buffered, and a Holding Timeout Timer (T_{hold}) is started at the LWIP node from the third consecutively received DUP-ACK. The OOO ACKs of i^{th} flow are buffered in an ACK packet queue (Q_i). Implementation of Q_i is realized as a map data structure, where the map entities include (i) five tuples of flow i, (ii) Holding Timeout Timer (T_{hold}), and (iii) reference to the ACK packet. If holding timer (T_{hold}) expires (*i.e.,* larger than the maximum allowable holding time t), the packets queued in the buffer will be sent to the intended destination, and the corresponding entry in the map will be flushed. Obtaining the value of t will be detailed in the next section. If the missing ACK packet is received before the expiry of T_{hold}, then the timer will be stopped for that flow, and the in-sequence ACKs will be forwarded to the higher layer. DIDA works in real-time precisely.

V. OPTIMAL HOLDING PERIOD

The holding time (t) is a crucial control factor in determining the system operating performance. The growth of the congestion window is favored by lesser RTT and high packet success probability. As t increases, more OOO packets are resolved, and hence higher success ratio is achieved; at the same time, RTT gets increased, which negatively affects the window growth. Most of the congestion control algorithms, such as HTCP [10], STCP [11], and BIC [12] are dependent on RTT for deciding the growth of the congestion window. These constraints bring a trade-off in increasing and decreasing of the holding time. An optimal operating region of t must be formulated to maximize the system throughput. The system cannot hold the packets for a longer duration, which will cause retransmission timer to expire. Thereby triggering the retransmission timeout, which in-turn sets TCP window size to one and slow start phase begins. Hence, the holding time is bounded. Upper bound on holding time (t) is obtained from various parameters such as current RTT (R'), smoothed RTT ($SRTT$), round-trip time variation ($RTTVAR$), clock granularity (G), a constant K whose value is 4 and retransmission timeout (RTO) [13]. γ is a smoothening factor for RTT and typically set to $1/8$ [14]. The maximum holding time (T_{MHT}) is given by:

$$\begin{aligned} T_{MHT} = \frac{1}{\gamma}\{RTO - max(G, K \times RTTVAR) \\ -(1-\gamma)SRTT - \gamma R'\} \end{aligned} \tag{1}$$

As the holding time increases, the number of packets delivered in-sequence to higher layers also increases. This leads to an increase in the success probability, and hence the growth of TCP CW. The buffer can reorder received ACK packets that are received within the holding time. The packets arrived through two interfaces that may not be in order. To mitigate this reordering problem, we found the probability of success through empirical results which can be represented as

$$P_s(t) = \psi \times t^{\Upsilon} \tag{2}$$

where ψ is normalizing factor, defined as $\frac{1}{T_{MHT}^{\frac{1}{e}}}$ and Υ is reordering factor which is equal to $\frac{1}{e}$. Window growth function (ϑ_t) depends on success probability ($P_s(t)$) and RTT and it is given by:

$$\begin{aligned} \vartheta_t = \delta_{buff} \times w \times (1-\beta(w)) \times P_f(t) \\ +\ \delta_{buff} \times P_s(t) \times \alpha(w)\left(\frac{T_E}{RTT+t}\right) \\ subject\ to\ 0 \leq t \leq T_{MHT} \\ \delta_{buff}, RTT > 0 \end{aligned} \tag{3}$$

The term, ϑ_t is a continuous function, and it is time-dependent. As the holding time 't' increases, the probability of success P_s increases. But the growth of the TCP window decreases as the RTT component $RTT + t$ increases. The overall window function ϑ_t is dependent on both P_s and $RTT + t$. Other parameters are T_E, which is the time elapsed since the last successful transmission. $\alpha(w)$ and $\beta(w)$ denote increase and decrease function of window size in HSTCP [15], respectively. w corresponds to the observed window size.

First differentiation of ϑ_t can be found as following:

$$\begin{aligned} \vartheta'_t = -\delta_{buff} \times w \times (1-\beta(w)) \times (e^{-1} \times t^{e^{-1}-1}) \\ +\ \delta_{buff} \times \alpha(w) \times \\ T_E\left(\frac{e^{-1} \times t^{e^{-1}-1} \times (RTT+t) - t^{e^{-1}}}{(RTT+t)^2}\right) \end{aligned} \tag{4}$$

It has been evaluated and observed that $\vartheta''(t)$ (second derivative) is strictly lesser than zero which ensures that there is a maximum point. The maximum point is obtained by setting $\vartheta'(t) = 0$ with unit ψ . The best holding time 't_{opt}' is obtained as follows:

$$\begin{aligned} t_{opt} = \frac{1}{2Ae^{-1}}\Bigg((2Ae^{-1} \times RTT + B(1-e^{-1})) + \\ \sqrt{(2Ae^{-1} \times RTT + B(1-e^{-1}))^2 + 4Ae^{-1}(e^{-1} \times B \times RTT - Ae^{-1}RTT^2)}\Bigg) \end{aligned} \tag{5}$$

where

$$A = \delta_{buff} \times w \times (1-\beta(w))$$

$$B = \delta_{buff} \times \alpha(w) \times T_E$$

The maximum point is denoted as $(t_{opt}, \vartheta(t_{opt}))$. The best operating range R for holding time t_{opt} could be found using the following expression.

$$LB = t_{opt} - \frac{R}{S_1 \times \theta}$$

$$UB = t_{opt} + \frac{R}{S_2 \times \theta}$$

Here, t_{opt} corresponds to the t value for which ϑ_t is observed to be maximum. LB and UB correspond to lower and upper bounds for operating region. $S_1\ and\ S_2$ correspond to slope with respect to $t = 0$ and $t = T_{MHT}$. The value of θ is given as $TS = \frac{1}{S1+S2}$.

TABLE I: Experimental Parameters

Parameter	Value
LTE eNB bandwidth	5 MHz
Number of resource blocks	25
Wi-Fi transmit power	20 dbm
LTE MAC scheduler	Round Robin
Wi-Fi frequency, bandwidth	2.4 GHz, 20 MHz
Wi-Fi standard	IEEE 802.11 g, n

VI. REALIZATION OF LWIP SYSTEM ARCHITECTURE

OpenAirInterface (OAI) [16] is used to setup the LWIP testbed, which includes OAI User Equipment (OAI-UE), OAI eNodeB (OAI-eNB), and OAI Core Network (OAI-CN). LWIP testbed is shown in Figure 7. It illustrates OAI-eNB and Cisco access point which are connected through ethernet. The control plane of LWIP is through LTE and data plane is through both LTE and Wi-Fi. Android prefers to use Wi-Fi link when both LTE and Wi-Fi interfaces are available. We enhanced the Android application HI PRIORITY KEEPER [17] which allows LTE and Wi-Fi links to be used simultaneously. The original destination IP address of the packet is changed from UE LTE IP address to UE Wi-Fi IP address at LWIP node to realize LWIP operation. On receiving the packet at UE, the destination IP address is changed back to UE LTE IP address by rule the inserted at the destination Network Address Translation (NAT) chain of the Linux *iptables*. LWIP operation is enabled in Nexus 5, and it downloads a file from the remote server simultaneously through LTE and Wi-Fi interfaces. The uplink packets are strictly confined to be sent through the LTE interface. Link Aggregation Layer (LAL) is introduced at the LWIP node, and it is responsible for effectively aggregating LTE and Wi-Fi links. Figure 5 shows the components of LAL; it primarily includes (a) the packet steering module to steer the traffic across LTE and Wi-Fi links, and (b) our proposed DIDA receiver to solve the OOO packet delivery problem. The source code of LWIP is open for further research [18].

TABLE II: Configurations of testbed setup

Parameter	Value
OAI LTE eNB Hardware Config	ExMIMO2/USRP-B210, Intel Xeon 8 core, 12GB DDR, Gigabit Ethernet 1 Gbps
OAI LTE eNB Software Config	Ubuntu 14.04, Low Latency Kernel 3.19 Ubuntu 14.04, Kernel 3.4.60-mptcp
OAI EPC Hardware Config	Intel Xeon Server 24 core, 64GB DDR, Gigabit Ethernet 10 Gbps
OAI EPC Software Config	Ubuntu 14.04, Kernel 3.19 generic
Remote Server Harware Config	Intel Xeon 8 core, 32GB DDR, Gigabit Ethernet 1 Gbps
Remote Server Software Config	Ubuntu 14.04, Kernel 3.2 Apache 2 Web server
User Equipment	Nexus 5 - hammerhead, Android 4.4.4 (kitkat)

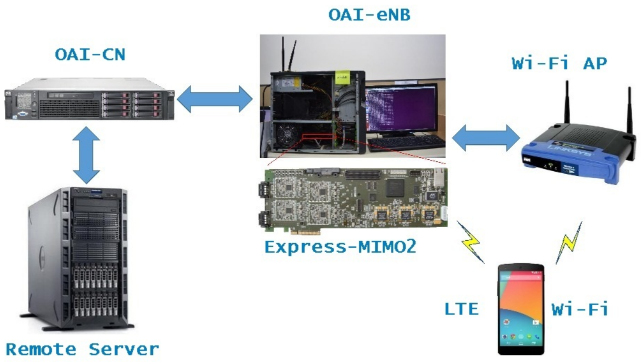


Fig. 7: LWIP testbed.

VII. PERFORMANCE EVALUATION

3GPP has limited the usage of LWIP to switched bearer. But DIDA makes it feasible to work for the split bearer. Thorough testing has been performed to confirm the working of DIDA and results are briefed below. Table I and II show the experimental parameters and configuration used in the testbed setup, respectively.

Experiment 1: Evaluating the link level aggregation benefit by downloading a file of size 16MB through both LTE and Wi-Fi interfaces simultaneously by varying the holding time. The RTT to the destination is found to be 100 msec on average.

The holding time is varied from 0 to 600 msec; it is found that for different split ratios the best throughput is achieved at a specific holding time. Packet split of 50% yields the highest throughput among all for the holding time of 50 msec (approx.), as shown in Figure 9. Other packet splits could not grow higher because of the speed of the slowest link problem; one comparatively slower interface brings down the overall system throughput. Also, the holding time ensures in-order delivery of the packets to the destination. It is notable that no other reordering algorithm does reordering at the last hop in the network (*i.e.,* in a small cell) to solve OOO delivery at the destination (UE).

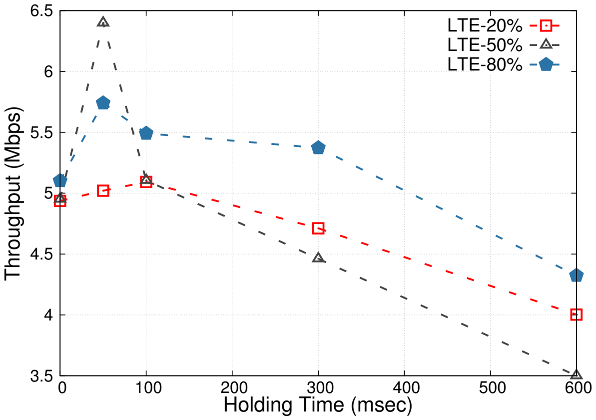
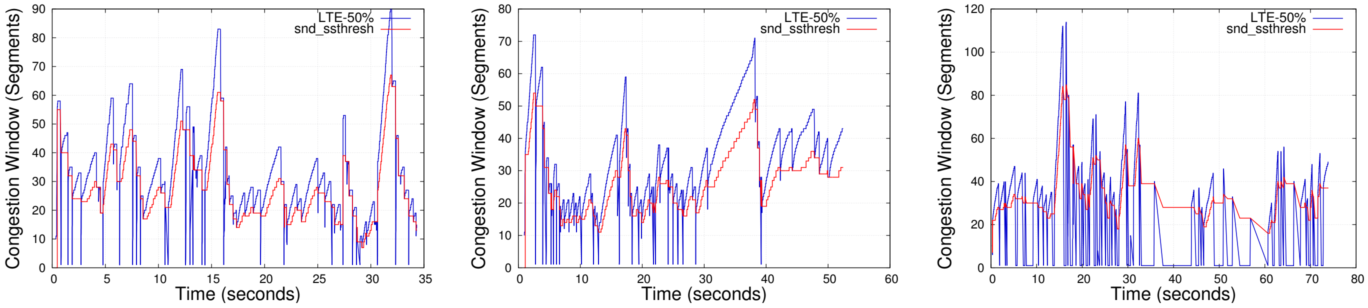


Fig. 9: Throughput for packet split of 20%, 50%, and 80% with varying holding time for RTT of 100 msec.

Experiment 2: Evaluating the aggregation benefit by downloading a file size of 32MB, with an RTT of 20 msec. The holding time is varied to study the performance.

Figures 8a - 8c capture the congestion window growth with different holding times. Figure 10 shows that the effect of backhaul RTT. In our experiments, LTE operates with 5 MHz bandwidth, and Wi-Fi operates with 20 MHz bandwidth. The fraction of packets sent through LTE and Wi-Fi interfaces, which correspond to their link capacity, could potentially reach very high throughput. In this case, LTE-20% achieves the maximum throughput. Holding time lesser than twice of RTT allows the steering ratio to reach maximum throughput. As the holding time increases, the timeout increases which is very clearly visible from Figures 8a, 8b, and 8c. TCP timeout in our proposed algorithm is taken care of maximum holding time. The change in the holding time has reflected in reducing the number of drops causing triple DUP ACKs as shown in Figure 11. It is evident that holding time increases the in-sequence delivery of packets; also, they increase RTT for that connection which negatively affects the system throughput.

Figure 12 compares the download times by varying file sizes in case of LTE, Wi-Fi, LWIP, LWIP+DIDA, and MPTCP schemes. LWIP+DIDA reduces the download time as compared to using only LTE or only Wi-Fi. Also, LWIP+DIDA exhibits comparable performance with MPTCP when LTE and Wi-Fi link qualities do not fluctuate. When the link qualities fluctuate, then LWIP+DIDA reacts faster as compared



(a) Holding Time of 10 msec. (b) Holding Time of 20 msec. (c) Holding Time of 50 msec.

Fig. 8: Congestion window growth for a 32 MB file download with packet steering ratio across LTE and Wi-Fi as 1:1.

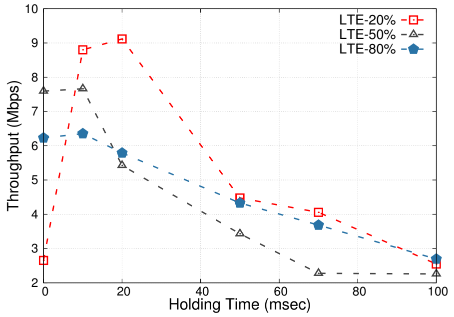


Fig. 10: Throughput for packet split of 20%, 50%, and 80% by varying holding time.

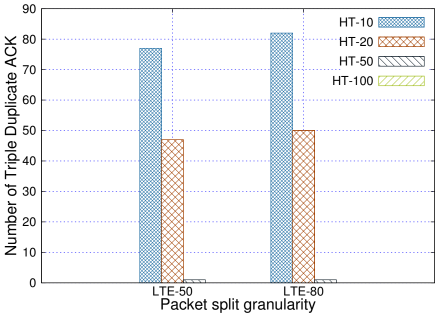


Fig. 11: Triple DUP-ACKs observed for different packet split ratios by varying holding time.

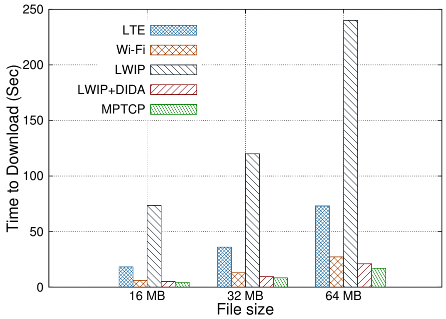


Fig. 12: Download time observed for LWIP, LWIP with DIDA, and MPTCP.

to MPTCP and exhibits a better performance. The native LWIP display a worse performance due to the increased OOO delivered at the sender whereas LWIP+DIDA effectively solves it and capable of increasing its performance significantly due to two factors: (a) effective steering across links based on RTT estimate of that particular link, and (b) effectively holding the DUP-ACKs for a fixed duration without causing TCP timeout.

VIII. CONCLUSIONS

3GPP has not supported split bearer which would negatively affect the system performance in the LWIP context. In this work, we proposed DIDA to address the problem that persists with OOO packet delivery in the context of TCP. We have evaluated our proposed solution in a testbed to observe its performance. Further, we have formulated the optimization problem to find the best holding time for a given packet split with estimated RTT. Operating DIDA with optimal holding time could save 20% of DUP-ACKs on average, which might have caused TCP congestion window to drop. Also, DIDA doubles the throughput achieved as compared to native LWIP.

ACKNOWLEDGEMENTS

This work was supported by the project "Low Latency Network Architecture and Protocols for 5G Systems and IoT", SERB, Govt. of India.

REFERENCES

[1] Cisco. (2017) A White Paper on Global Mobile Data Traffic Forecast Update. [Online]. Available: https://goo.gl/zY4nKI
[2] 3GPP, "LTE-WLAN Aggregation and RAN Controlled LTE-WLAN Interworking," Tech. Rep. 36.300, 2016.
[3] F. Wang *et al.*, "Improving TCP performance over mobile ad-hoc networks with OOO detection and response," in *Proc. of ACM MobiHoc*, 2002, pp. 217–225.
[4] V. Paxson, "End-to-end internet packet dynamics," in *ACM SIGCOMM CCR*, vol. 27, no. 4, 1997, pp. 139–152.
[5] M. Zhang *et al.*, "RR-TCP: a reordering-robust TCP with DSACK," in *Proc. of ICNP*. IEEE, Nov 2003, pp. 95–106.
[6] S. Bohacek *et al.*, "A New TCP for Persistent Packet Reordering," *IEEE/ACM Trans. on Net.*, vol. 14, no. 2, pp. 369–382, April 2006.
[7] C. Paasch and O. Bonaventure, "Multipath tcp," *Communications of the ACM*, vol. 57, no. 4, pp. 51–57, 2014.
[8] D. Ibarra *et al.*, "Software-Based Implementation of LTE/Wi-Fi Aggregation and Its Impact on Higher Layer Protocols," in *Proc. of ICC*. IEEE, May 2018, pp. 1–6.
[9] T. Santhappan *et al.*, "Visible: Virtual congestion control with boost acks for packet level steering in lwip networks," in *Proc. of IEEE GLOBECOM*, Dec 2017, pp. 1–7.
[10] D. Leith and R. Shorten, "H-TCP: TCP for high-speed and long-distance networks," in *Proc. of PFLDnet*, 2004.
[11] T. Kelly, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks," *ACM SIGCOMM CCR*, vol. 33, no. 2, pp. 83–91, 2003.
[12] L. Xu *et al.*, "Binary Increase Congestion Control for Fast Long-Distance Networks," in *Proc. of IEEE INFOCOM*, 2004, pp. 2514–2524.
[13] V. Paxson *et al.*, "Computing TCP's Retransmission Timer," IETF, Tech. Rep. RFC 6298, 2011.
[14] V. Jacobson, "Congestion avoidance and control," in *ACM SIGCOMM CCR*, vol. 18, no. 4, 1988, pp. 314–329.
[15] S. Floyd, "Highspeed tcp for large congestion windows," IETF, Tech. Rep. RFC 3649, 2003.
[16] N. Nikaein *et al.*, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM CCR*, vol. 44, no. 5, pp. 33–38, 2014.
[17] M. Baerts. (2015) HIPRIKeeper. [Online]. Available: https://github.com/MPTCP-smartphone-thesis/MultipathControl
[18] S. Thomas *et al.* (2017) LTE Wi-Fi Integration With IPSec Tunnel. [Online]. Available: https://github.com/ThomasValerrianPasca/