Protecting Regular Customer Traffic from Ad-hoc Traffic in Public WLAN Hot-Spots

Jani Lakkakorpi
Aalto University, Comnet
Email: jani.lakkakorpi@iki.fi

Teemu Kärkkäinen
Aalto University, Comnet
Email: teemu.karkkainen@aalto.fi

Jörg Ott
Aalto University, Comnet
Email: jorg.ott@aalto.fi

Abstract—**Mobile devices may often use WLAN hot-spots to communicate directly with one another in a peer-to-peer fashion without having to authenticate (and pay) for Internet access. This ad-hoc-style communication may impede the performance of regular Internet users' hot-spot experience. We present how LEDBAT (Low Extra Delay Background Transport) and its modified version, fLEDBAT, can be used as transport protocol between mobile devices in an infrastructure mode WLAN to reduce the impact on regular traffic. Although promising, the use of (f)LEDBAT may not solve all the problems arising from high-volume ad-hoc data traffic. Active queue management at the access point would do this but this would require controlling it. To overcome this, we complement LEDBAT by bundle admission control mechanisms based on the queueing delay and/or the number of active bundle transfers as alternative approaches. Both combined make device-to-device traffic yield to regular traffic, which we validate through extensive ns-2 simulations.**

I. Introduction

Opportunistic communication networks built between mobile devices using store-carry-forward networking can enable communication in scenarios where the use of fixed network infrastructure is not possible or not desirable. However, direct ad-hoc communication between devices remains challenging due to the limitations in protocol specifications (e.g., IEEE 802.11), chipsets and mobile operating systems. One proposed way to get around this limitation is to instrument existing, commercial wireless LAN access points (WLAN APs) that do not employ L2 security to serve as device-to-device link layer packet relays. Even though the commercial APs typically charge clients for Internet connectivity through them, it has been shown that close to 50% of them allow connected clients to exchange packets with each other without any input from the users (i.e., no authentication, payment or interaction with captive portals) [1]. Further, in addition to the commercially deployed APs, the mobile devices themselves can act as open WLAN APs for other devices to connect to in order to exchange messages with each other [2], [3].

Harnessing commercial APs for ad-hoc communication presents the immediate problem of ensuring that the device-to-device messaging does not interfere with the traffic of clients that pay the AP operator for Internet access; we refer to this traffic as *regular traffic*. The disruption occurs since the paid packets flowing to and from the Internet interact with the non-paid device-to-device packets—dubbed *ad-hoc traffic*—at the WLAN AP's wireless interface queue. Initial experiments suggested that using *less than best effort* protocols that voluntarily scavenge only residual capacity and allow the regular traffic to take precedence could be a feasible approach [1]. In this paper, we study such protocols in depth. Specifically, we investigate if and how using the (fair) less-than-best effort transport protocol (f)LEDBAT for ad-hoc traffic will reduce the impact of these flows on regular TCP and UDP flows in a WLAN hot-spot.

The second issue arising from the use of WLAN APs for direct device-to-device messaging is the high offered traffic load that this may lead to. Since WLAN APs are a broadcast medium where every node can hear every other node, the number of possible device pairs grows proportionally to the square of the node count. When using delay-tolerant networking [4] in such a scenario, bundles [5] are often distributed using epidemic routing [6]. In epidemic routing each node pair aims to synchronize their message buffers by exchanging messages until both nodes carry the same set of messages. Since bundles can be arbitrarily large, the total amount of transfer capacity required to synchronize all buffers between all node pairs in the broadcast network quickly grows beyond the available resources. To solve this issue we propose and evaluate admission control mechanisms that limit the total number of parallel bundle exchanges in the WLAN AP.

The rest of this paper is structured as follows: We briefly review opportunistic networking protocols by example in section II. We present the use of (f)LEDBAT as transport protocol for message exchanges in section III and introduce the bundle admission control mechanisms in Section IV. We then describe our simulation model used for studying the impact of these mechanisms and discuss the findings of our simulation studies in Section V. We conclude the paper with a brief assessment and hint at next steps in Section VI.

II. Opportunistic Networking Protocols

The past years of research led to a number of opportunistic networking platforms, including Haggle [7], SCAMPI [8], and IBR-DTN [9]. The latter two are using the *Bundle Protocol (BP)* [5] that defines delay-tolerant communications using asynchronous messaging [10], and we use it as a representative example for message transfers. The Bundle Protocol is an internetworking layer that is mapped to specific underlying networks (at layer 2–4) by means of *convergence layers*, which offer a common abstraction from the lower layers. The actual message transfer protocol is complemented by a neighbor discovery mechanism, which may exploit existing services such as *Bonjour* or implement some variant of *Hello* messages multicast to a well-defined transport address.

Some of the convergence layers offer just minimal functionality: for example, those mapping to UDP and Ethernet offer not much beyond a defined way to encapsulate individual bundles in a datagram or link layer frame; they do not cater for reliability and limit the size of a bundle to the size of the lower layer maximum transmission unit. More sophisticated convergence layers either use more powerful transport protocols, such as the TCP convergence layer [11], or define such a protocol on top of the lower layers, such as the Licklider Transmission Protocol (LTP) [12].

For mobile opportunistic communication, the common case appears using message transfer over a reliable lower layer, and typically TCP is chosen when using WLAN due to its ubiquitous availability in mobile phone OSes. As described above, when using TCP, the ad-hoc traffic of those mobile nodes carrying out opportunistic device-to-device communication competes on even ground with the regular traffic of paying hot-spot customers who access the Internet. This implies that the ad-hoc users obtain their (fair) share of the network capacity, possibly even putting the TCP connections to the Internet at a disadvantage since the latter experience higher RTTs. To avoid this disturbance, we explore using LEDBAT underneath the TCP convergence layer instead of TCP.

III. (F)LEDBAT AS A BUNDLE TRANSPORT

Since we do not want our ad-hoc communication to disturb operator's regular traffic, we shall use LEDBAT (Low Extra Delay Background Transport) [13] as our bundle transport protocol. LEDBAT detects congestion based on access point queueing delay—this is our bottleneck: all ad-hoc traffic as well as the regular (commercial) traffic goes into this queue—and will decrease its congestion window before packet loss takes place. Thus, LEDBAT will always yield to TCP (and other non-LEDBAT) traffic. However, LEDBAT has an intra-protocol unfairness issue, known as late-comer advantage. This means that the latest flow will get most of the resources while the other flows starve. A solution for this problem, fLEDBAT (fair LEDBAT), has been suggested in [14].

The congestion avoidance algorithm of LEDBAT is illustrated in Equation 1 and 2. We need to first estimate the current queuing delay by subtracting the minimum delay, D_{min}, from the measured delay $q(t)$. When we subtract our target queuing delay, τ, from the queuing delay we obtain $\Delta(t)$.

$$\Delta(t) = (q(t) - D_{min}) - \tau \tag{1}$$

$$cwnd(t+1) = \begin{cases} cwnd(t) + \alpha \frac{\tau - \Delta(t)}{\tau} \frac{1}{cwnd(t)} & \text{if no loss,} \\ \frac{1}{2} cwnd(t) & \text{if loss.} \end{cases} \tag{2}$$

In LEDBAT, the congestion window is updated according to Equation 2. If there is no packet loss, we shall apply the upper part of the equation, where α is the additive increase factor. If there is packet loss, the congestion window is halved, just like in TCP.

Congestion avoidance in fLEDBAT (see Equation 3) is a bit more complex than in LEDBAT. fLEDBAT introduces a new parameter, ζ, which is used as a multiplicative decrease factor.

$$cwnd(t+1) = \begin{cases} cwnd(t) + \alpha \frac{1}{cwnd(t)} & \Delta(t) \leq 0, \\ cwnd(t) + \alpha \frac{1}{cwnd(t)} - \frac{\zeta}{\tau}\Delta(t) & \Delta(t) > 0, \\ \frac{1}{2} cwnd(t) & \text{if loss.} \end{cases} \tag{3}$$

We compare TCP, LEDBAT, and fLEDBAT as bundle transport protocol with respect to bundle delivery performance and the impact of the ad-hoc traffic on the performance of regular traffic in section V-C.

IV. BUNDLE ROUTING AND ADMISSION CONTROL

We now turn our attention to the second issue: bundle routing. While many different routing protocols for mobile opportunistic networks exist (see [15] for a recent survey), their macroscopic operation is immaterial for our purpose: we only need to consider their small-scale operations. Specifically: when a number of nodes "meet" in a WLAN hot-spot, to how many neighbors will they forward their bundles?

As we assume that WLAN hot-spots always carry regular traffic, it is important to avoid flooding, i.e., epidemic-style routing protocols should not be used.[1] For this reason, we use protocols that limit the number of copies per message. We start from modified binary spray and wait [16] routing with just four forwarding tokens. In binary spray and wait, the number of forwarding tokens is halved each time a bundle is forwarded; the next hop gets half of the tokens. Whenever a bundle is created at the source node or whenever a bundle arrives at the forwarder node, the following rules are applied: If both source/forwarder and destination nodes are within the access point radio range (Hello messages from the destination are received), we send the bundle directly to the destination node (if it does not have this bundle already). After this, we stop forwarding the bundle. If the bundle cannot be sent to the destination node directly, we forward the bundle to a limited number of other nodes that are within the access point range (and do not have this bundle). If the destination node becomes reachable later, we send the bundle directly there.

An easy solution for protecting regular traffic would be the use of priority queueing or active queue management (e.g., Random Early Detection, RED [17]) at the access point. In RED, once the exponentially averaged queue size exceeds a minimum threshold, we start dropping random packets with the dropping probability increasing as the averaged queue size grows. In RIO (RED In and Out) [18], we have different packet dropping thresholds for in-profile and out-profile packets, which would map to regular and ad-hoc traffic. However, we may not be able to configure the access point in any way and thus we have to somehow limit the number of bundles that the nodes send to each other.

[1]We confirmed this assumption in simulations.

We therefore define a number of simple admission control schemes individual nodes can apply to their ad-hoc traffic. Our first bundle admission control is based on the number of on-going bundle transfers (parameter-based admission control, PBAC): each node has a local limit for the number of on-going bundle transfers. If the limit (of, e.g., five concurrent connections) is reached, the node has to wait for one on-going bundle transfer to complete before it can start a new one.

The aforementioned simple scheme can be made more efficient (more statistical multiplexing) by having a common limit (of, e.g., 30 connections) for the total number of bundle transfers (TPBAC). We propose (see Algorithm 1) that the number of on-going bundle transfers via the access point should be calculated based on received Hello messages that carry the number of on-going bundle transfers per node. The Hello messages naturally carry the identifiers of stored bundles, too. Hello message frequency should be rather high, e.g., one Hello message per 500 ms.

Algorithm 1 TPBAC: Update the total number of connections N

```
Hello message from node $i$ arrives:
update the number of on-going connections for node $i$, $N_i$
set $N \Leftarrow 0$
for each $i$ do
    set $N \Leftarrow N + N_i$
end for
```

Bundle admission control can be measurement-based (MBAC) as well. In this scheme, nodes learn the access point queueing delay by pinging the AP regularly (e.g., every 500 ms). Bundle can be forwarded only if we have fresh delay information and if the delay is below the threshold (of, e.g., 25 ms). The queueing delay could be extracted from Hello messages, too.

Naturally, PBAC may be combined with MBAC (MPBAC). We could either have adaptive limits (based on delay measurements) for the number of on-going bundle transfers or we could simply run the PBAC and MBAC schemes simultaneously. We have chosen the latter approach.

We propose adaptive TPBAC (ATPBAC, see Algorithm 2), where the common limit for the number of on-going bundle transfers is adaptive. When the AP queueing delay drops below 20 ms, we increment the common limit by one and when the queueing delay exceeds 25 ms, we decrement the common limit by one.

We evaluate the performance of our different admission control schemes and their impact on regular hot-spot traffic in section V-D.

V. Evaluation

For our simulations, we use ns-2.35 [19] with the LEDBAT code from Dario Rossi [20] and our own implementation of bundle-based communication. The simulation time is 2000 seconds per simulation run, and there are 10 simulation runs per scenario across which we report the mean results.

Algorithm 2 ATPBAC: Adapt the limit L for the total number of connections N

```
Hello message from any node arrives:
update the access point queueing delay $D$
if $D < 20ms$ then
    set $L \Leftarrow L + 1$
else
    if $D > 25ms$ then
        set $L \Leftarrow L - 1$
    end if
end if
```

TABLE I: IEEE 802.11g simulation parameters.

Parameter	Value	Parameter	Value
dataRate	54 Mbps	PLCPDataRate	1.0 Mbps
Propagation	2-RayGround	BeaconInterval	100 ms
gSyncInterval	0.01 ms	ScanType	Passive
CWMin	16	ProbeDelay	0.1 ms
CWMax	1024	MaxChannelTime	11 ms
RTSThreshold	3000 bytes	MinChannelTime	5 ms
ShortRetryLimit	8	ChannelTime	120 ms
LongRetryLimit	5	CSThresh	1.559e-11 W
SlotTime	9 ms	RXThresh	3.652e-10 W
SIFS	16 ms	Pt	0.0372 W
Preamble	72 bits	freq	2407 MHz
PLCPHeader	24 bits	L	1.0

A. Wireless Channel Model

We use IEEE 802.11 infrastructure mode. The IEEE 802.11g MAC/PHY settings presented in Table 1 lead to radio range of 100 m.

Figure 1 shows our simulated network: one node in the fixed network is connected to the access point with a link that has a latency of 50 ms and a bandwidth of 10 Mbps. We have an area size of 250 m × 250 m, in which ten nodes move according to the random waypoint mobility (RWP) mobility model (pause time: 2 s, maximum speed: 5 m/s) and six nodes are at static positions. The mobile nodes use opportunistic ad-hoc communication among each other, whereas the static nodes represent regular users of a hot-spot. One node is the access point node, located exactly in the middle (x=125 m, y=125 m).

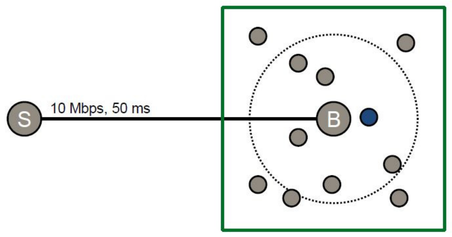

Fig. 1: Simulated network topology.

B. System Setup and Traffic Models

The access point downlink queue (interface queue, IFQ) has a default size of 500 packets. This queue becomes the system bottleneck as all traffic is routed via the access point (B). In our simulations, we consider two types of regular traffic: TCP traffic mimicking web page retrieval and RTP traffic representing VoIP or Internet radio traffic. Both TCP and RTP traffic are sent from the server node S to the static client nodes. RTP packets get scheduling priority at the access point, i.e., they are put to the head of IFQ. The ad-hoc bundle traffic is sent between mobile nodes, via the AP, and it should ideally yield to (other) TCP traffic.

One static node is receiving real-time media content from the fixed network using RTP and one to five static nodes are downloading web pages using HTTP. All the static nodes are 25 m away from the access point (x=150 m, y=125 m).

HTTP traffic is modeled as follows: Four Linux TCP agents per session are used simultaneously; page requests are lognormally distributed with $\mu = 5.9288$ and $\sigma = 0.3208$. There are 31 items per page (size 9.82 KB each leading to a total page size of 304 KB). In the original web traffic model [21], page reading time is lognormally distributed with $\mu = 2.754$ and $\sigma = 1.566$; we have added a maximum reading time of 120 seconds, i.e., the distribution is now truncated.

Our RTP traffic generator uses G.711 codec with 200-byte packets and 50 packets are generated per second as a continuous flow without any pauses.

For our ad-hoc traffic, nodes send large files to each other (5 MB bundles with a random destination, at random intervals, via the access point) using (f)LEDBAT or Linux TCP as transport protocol. We have extended Dario Rossi's LEDBAT code for ns-2 [20] by implementing fLEDBAT according to [14]. Target queueing delay $\tau = 25\,\text{ms}$ and the additive increase factor $\alpha = 1$ are used with both LEDBAT and fLEDBAT. In fLEDBAT, we choose the multiplicative decrease factor $\zeta = 5$. The bundle lifetime is 600 s.

Hello message exchange is not modeled. There are no antipackets or bundle retransmissions. Node buffers are large enough, i.e., there is no bundle dropping.

C. TCP vs. LEDBAT vs. fLEDBAT

We do not observe any impact on the RTP flows in any of our simulations (and therefore they are omitted from the figures). This is due to the RTP packets being given scheduling priority at the access point, and as a result they experience very low delays regardless of other traffic load. We further observe that all the packet losses experienced by the RTP flows were due to link layer collisions rather than due to queueing. This indicates that real-time traffic flows that are given scheduling priority at the AP will not be negatively impacted by the use of the AP as a device-to-device relay regardless of the transport protocols employed.

In the first set of simulations in Figure 2 (A-1 to A-4) we study the impact of the ad-hoc transport protocol choice on the regular traffic. A-1 shows that this choice has no impact on the overall throughput, and increasing the number of regular traffic flows reduces the overall ad-hoc traffic throughput as the additional TCP flows claim their share of the capacity and there is less available for the ad-hoc traffic (fLEDBAT case is shown). A-4 shows that using TCP has a significant negative impact on regular traffic as all the flows then compete equally for the AP resources. Using LEDBAT instead of TCP for ad-hoc traffic leads to significantly less disturbance on regular traffic, while fLEDBAT has almost no negative impact on the regular traffic. Overall we can see that, as expected, the choice of TCP for ad-hoc traffic will significantly hurt regular traffic. However, using (f)LEDBAT will closely match the throughput of TCP without causing disturbance on the regular traffic.

As explained earlier, both LEDBAT and fLEDBAT are based on observing the bottleneck queue. We therefore examine the impact of the AP IFQ by reducing its size from 500 to 50 packets (B-1 to B-4). From B-1,2,3 we can see that this has no impact on the overall throughput of the (f)LEDBAT flows, nor the ad-hoc traffic delivery ratios or delays. However, from B-4 (compared to A-4) we can see that LEDBAT has significantly more effect on the regular traffic when the AP queue is shorter. This is due to LEDBAT seeing lower queue delays ($q(t)$) on average and, given that the rest of the protocol parameters are kept constant, the bottleneck will appear less congested and therefore LEDBAT will behave more aggressively. Unfortunately this seems to indicate that LEDBAT must be carefully parametrized based on the AP queue characteristics. Fortunately fLEDBAT is far less sensitive to the AP queue characteristics and performs almost equally regardless of the AP queue length.

We then validate that the results hold in more sparse scenarios by increasing the simulated area to 500 m $\times$ 500 m (C-1 to C-4). Since the network is sparser, there are fewer contacts between the nodes and the AP, and therefore the confidence intervals (not illustrated) are larger than in the previous cases. As expected, the ad-hoc bundle delivery ratios decrease and latencies increase in the sparser scenario. However, the average throughput of the ad-hoc traffic transport protocols remains the same, and the above findings of the impact on regular traffic still hold in the sparse case.

Finally, we examine the impact of increasing the amount of message replication done by the routing protocol by increasing the Spray-and-Wait token count from 4 to 8 (D-1 to D-4). The increased amount of replication leads to lower delivery rates (D-2 vs. A-2) and higher delays (D-3 vs. A-3) as the marginal benefit of spending resources on transmitting replicas decreases. However, the higher load does not lead to significantly higher disruption of regular traffic when using (f)LEDBAT for the ad-hoc traffic (D-4 vs. A-4).

D. Bundle Admission Control

As shown above, increasing the amount of replication by the routing protocol leads to lower delivery and delay performance of the ad-hoc traffic. We study this further by employing using epidemic routing instead of binary Spray-and-Wait (Figure 3). From A-2 and A-3 we can see that unlimited replication will severely lower the delivery rate and increase the delay, while

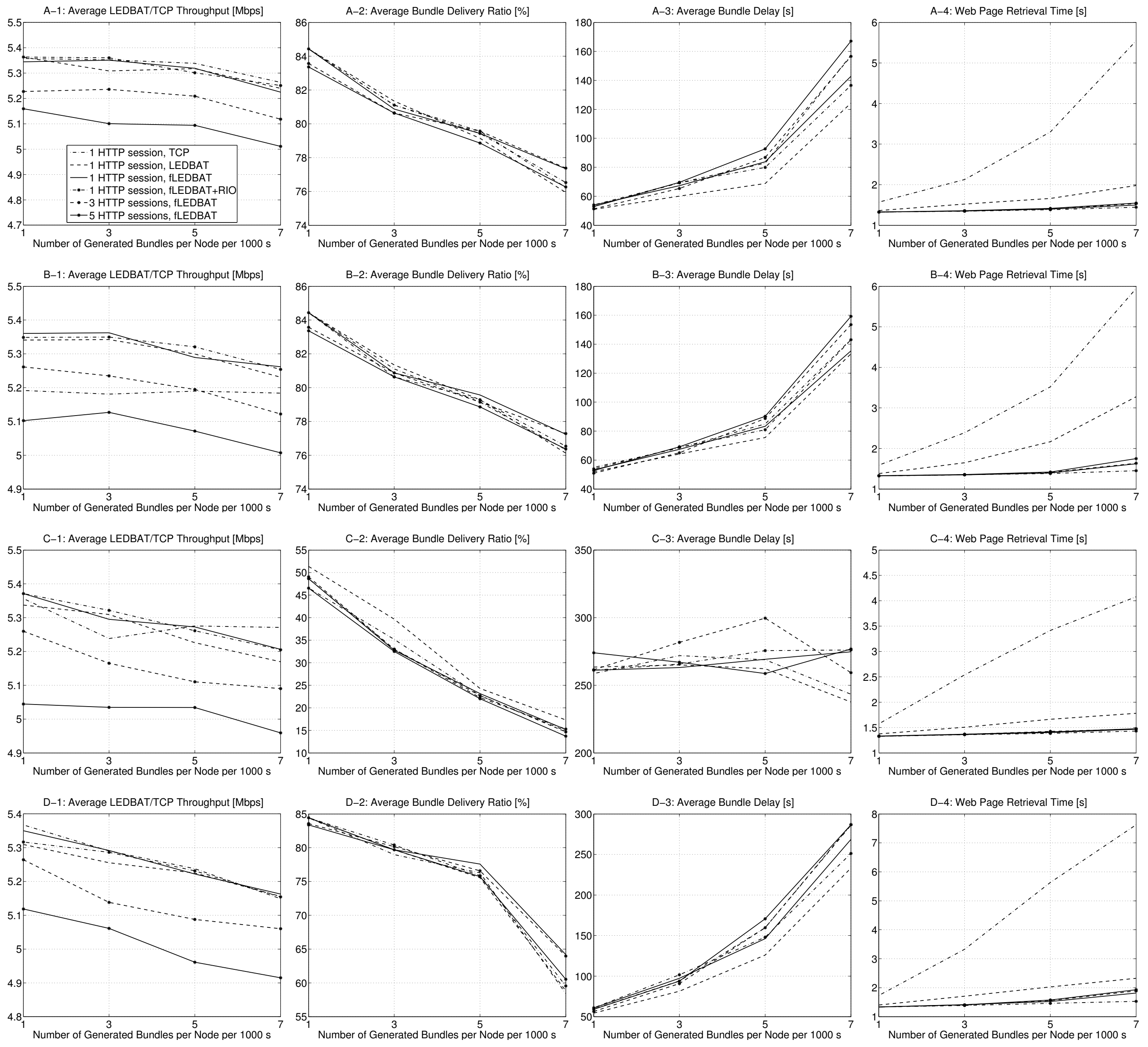

Fig. 2: Spray and wait. 1st row: 250 m × 250 m, 4 tokens, IFQ size 500 pkts; 2nd row: 250 m × 250 m, 4 tokens, IFQ size 50 pkts; 3rd row: 500 m × 500 m, 4 tokens, IFQ size 500 pkts; 4th row: 250 m × 250 m, 8 tokens, IFQ size 500 pkts.

(f)LEDBAT will still keep down the impact of the ad-hoc traffic on regular traffic.

To combat this effect we propose bundle admission control (BAC) mechanisms, which are shown in Figure 3, B-1 to B-4. B-2 and B-3 show that all BAC mechanisms increase delivery rates and lower the delays, with similar performance. However, it appears that TPBAC performs the best overall, leading to relatively high delivery rates, low delays and low disruption of regular traffic. Active queue management (RIO) does not improve the delivery or delay performance, but it cuts down the disruption on regular traffic more than BAC mechanisms.

VI. Conclusion

Ad-hoc traffic between mobile devices in a WLAN hot-spot quickly impacts regular traffic when TCP is used as bundle transport protocol and leads to high web page retrieval times as all TCP connections are equally aggressive. Using LEBDAT and especially fLEDBAT as bundle transport protocol helps alleviating this problem to some extent. However, as the number of (f)LEDBAT connections grows due to concurrent transfers of many bundles, controlling the IFQ length at the access point becomes increasingly difficult because new (f)LEDBAT connections might obtain higher base delay estimates as the

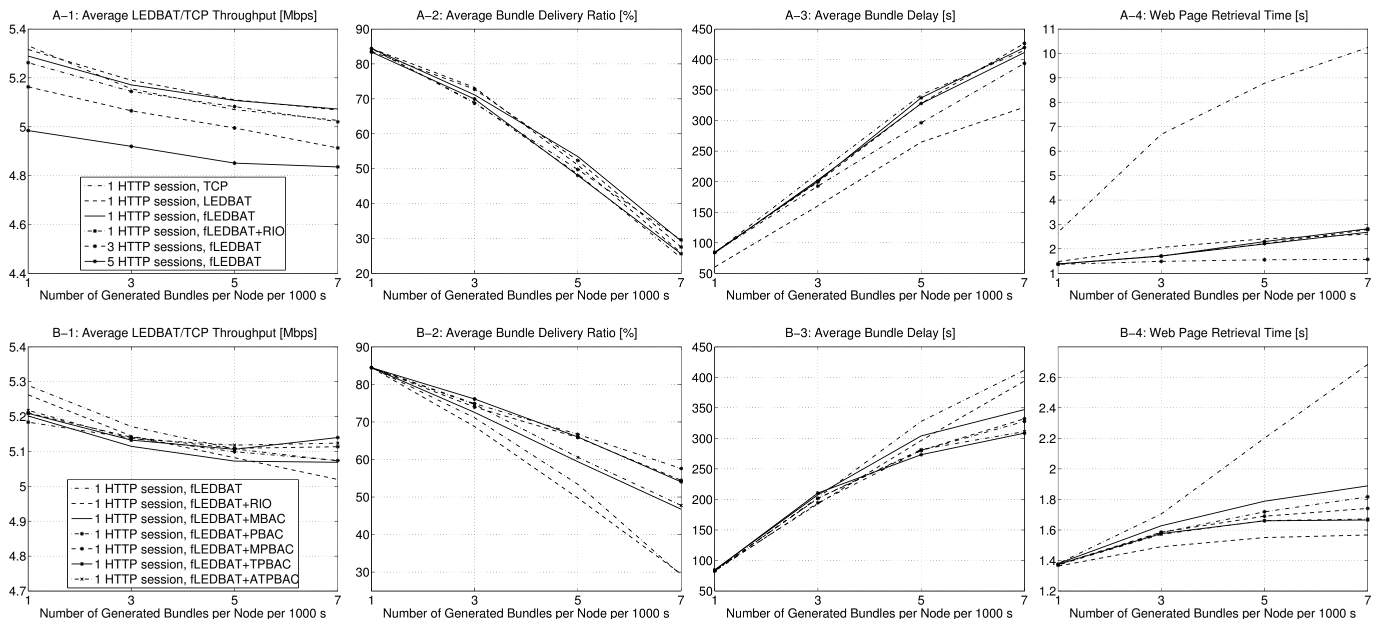

Fig. 3: Epidemic routing: 250 m × 250 m, IFQ size: 500 pkts. Bottow row: bundle admission control schemes.

IFQ is never empty. Without modifying the AP, the mobile devices themselves have to constrain the generated traffic.

Limiting the number of (f)LEDBAT connections can be achieved by using less aggressively replicating DTN routing protocols (to minimize the total incurred load in the first place) together with a bundle pacing mechanism. For the latter, we find that bundle admission control is a feasible approach to be implemented easily only at the mobile nodes.

Together these mechanisms ensure that the ad-hoc traffic has only a minor impact on the regular TCP traffic. Since regular RTP traffic is treated preferentially at the access point, it does not suffer at all. Avoiding such disturbance is key to providing an incentive to WLAN hot-spot operators (and thus AP manufacturers) to still tolerate ad-hoc traffic in the future.

While our findings from this paper are promising, further work with more sophisticated setups and also real-world experiments constitute next steps to understand if these mechanisms can deliver in practice.

Acknowledgments

This work received funding from the Academy of Finland in the RESMAN project (grant no. 134363) and from the European Community's Seventh Framework Programme under grant agreement no. 258414 (SCAMPI).

References

[1] T. Kärkkäinen, M. Pitkänen, and J. Ott, "Enabling ad-hoc-style communication in public wlan hot-spots," in *Proc. of the 7th ACM workshop on Challenged Networks*, CHANTS, 2012.
[2] S. Trifunovic, B. Distl, D. Schatzmann, and F. Legendre, "WiFi-Opp: Ad-Hoc-less Opportunistic Networking," in *Proc. of ACM MobiCom CHANTS workshop*, Sep 2011.
[3] H. Wirtz, T. Heer, R. Backhaus, and K. Wehrle, "Establishing Mobile Ad-Hoc Networks in 802.11 Infrastructure Mode," in *Proc. of ACM MobiCom CHANTS workshop*, Sep 2011.
[4] "Delay Tolerant Networking Research Group." http://www.dtnrg.org.
[5] K. Scott and S. Burleigh, "Bundle Protocol Specification." RFC 5050, November 2007.
[6] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," tech. rep., Duke University, April 2000.
[7] J. Scott, P. Hui, J. Crowcroft, and C. Diot, "Haggle: A Networking Architecture Designed Around Mobile Users," in *Proceedings of IFIP WONS*, (Les Ménuires, France), January 2006.
[8] T. Kärkkäinen, M. Pitkänen, P. Houghton, and J. Ott, "Scampi application platform," in *Proc. of ACM MobiCom CHANTS workshop*, 2012.
[9] J. Morgenroth, S. Schildt, and L. Wolf, "A bundle protocol implementation for android devices (demo)," in *Prof. of ACM MobiCom*, 2012.
[10] K. Fall, "A Delay-Tolerant Network Architecture for Challenged Internets," in *Proc. of ACM SIGCOMM*, 2003.
[11] M. Demmer, J. Ott, and S. Perreault, "Delay Tolerant Networking TCP Convergence Layer Protocol." Internet Draft draft-irtf-dtnrg-tcp-clayer-06.txt, Work in Progress, May 2013.
[12] M. Ramadas, S. C. Burleigh, and S. Farrell, "Licklider Transmission Protocol – Specification." Internet RFC 5326, September 2008.
[13] G. Carofiglio, L. Muscariello, D. Rossi, and S. Valenti, "The quest for ledbat fairness," in *GLOBECOM*, (Miami, FL, USA), December 2010.
[14] G. Carofiglio, L. Muscariello, D. Rossi, C. Testa, and S. Valenti, "Rethinking low extra delay backtround transport protocols," tech. rep., Telecom ParisTech, 2010.
[15] Y. Cao and Z. Sun, "Routing in Delay/Disruption Tolerant Networks: A Taxonomy, Survey and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 15, 2012.
[16] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *ACM SIGCOMM Workshop on DTN*, (Philadelphia, PA, USA), 2005.
[17] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, 1993.
[18] D. D. Clark and W. Fang, "Explicit allocation of best-effort packet delivery service," *IEEE/ACM Trans. Netw.*, 1998.
[19] S. McCanne and S. Floyd, "Network Simulator - ns (version 2)." http://www.isi.edu/nsnam/ns.
[20] D. Rossi, "Ledbat module for ns-2." http://perso.telecom-paristech.fr/~drossi/index.php?n=Software.LEDBAT.
[21] M. Molina, P. Castelli, and G. Foddis, "Web traffic modeling exploiting tcp connections' temporal clustering through html-reduce," *Network, IEEE*, vol. 14, no. 3, pp. 46–55, 2000.