On the Performance of Redundant Traffic Elimination in WLANs

Emir Halepovic, Majid Ghaderi and Carey Williamson
Department of Computer Science
University of Calgary, Calgary, AB, Canada
Email: {emirh, mghaderi, carey}@cpsc.ucalgary.ca

***Abstract*—Redundant Traffic Elimination (RTE) detects and removes repeated chunks of data across network flows, protocols, and applications, with the purpose of reducing bandwidth usage. In this paper, we explore the effectiveness of RTE in WLAN, compare it to RTE in Ethernet, and investigate specific issues affecting RTE in WLAN. Our results show that applying RTE to WLAN links is promising and can potentially yield high bandwidth savings, although RTE is not as effective in WLAN as in wired networks. However, to exploit the full potential of RTE, it is necessary to deal with specific challenges, such as longer headers, control and management frames, retransmissions, and dropped frames. We find that including parts of MAC headers in RTE can increase overall bandwidth savings by up to 53% in a public WLAN. To handle dropped frames, which can severely compromise the effectiveness of RTE, we make a case for MAC-layer RTE, which detects frame loss at the sender. This preserves 23% more savings than a previous approach. However, frame retransmissions generate additional traffic at MAC layer, which reduces the effectiveness of RTE in general case.**

Keywords- Traffic; Redundancy; Elimination; Measurement; Performance; WLAN; Wireless

I. INTRODUCTION

Redundant Traffic Elimination (RTE) has received considerable attention recently, both academic [1, 2, 10, 14] and commercial [6, 13]. Redundancy in network traffic arises due to repeated content within objects (web pages, files) and across objects due to skewed popularity [4, 5]. Previous works focus almost exclusively on RTE in high-bandwidth wired networks, while in this work we focus on specific issues for RTE in IEEE 802.11 WLAN environments.

Given the error-prone nature of wireless channels and the scarcity of RF spectrum, RTE emerges as a promising way to save bandwidth and improve performance in wireless environments. In this paper, we use measurement and simulation to characterize RTE in WLAN, and provide a comparison with RTE in Ethernet. We first seek to understand whether RTE works in general in WLAN, and then we take a more detailed look at specific factors that exist in WLAN.

We present a redundancy analysis of wireless user traffic from a campus WLAN. The findings include several different characteristics of wireless user traffic as compared to aggregate campus traffic, most notably a wide range of detected redundancy across traces. Our results indicate that the effectiveness of RTE is lower in WLAN vs. Ethernet due to physical channel errors, smaller proportion of IP traffic, and MAC-layer characteristics, such as longer headers, retransmissions, and dropped frames.

We start by considering MAC-layer RTE savings for WLAN, as opposed to only IP-layer savings considered in previous works. We identify four challenges for RTE in WLAN and discuss their effects and potential solutions. Longer headers and the prevalence of small packets reduce the useful part of the packet for RTE, leading us to suggest rearranging the MAC headers and including them in RTE. This approach increases the average savings from 17% to 26%, which is a relative improvement of 53%. We further discuss the importance of the amount of IP data within WLAN traffic and show that IP-layer RTE savings do not translate directly to the MAC layer as in Ethernet. We consider the impact of MAC-layer retransmissions, which make up to 40% of total data volume, and show that RTE can deal with them successfully. Finally, we revisit the problem of dropped frames, which can compromise the effectiveness of RTE [11]. We propose a novel MAC-layer solution that preserves 23% more RTE savings than earlier proposed methods of addressing frame loss.

The rest of this paper is organized as follows. Section II provides background on RTE and prior related work. Section III presents our data sets and evaluation methodology. Section IV provides analysis of challenges for RTE in WLAN. Section V concludes the paper.

II. BACKGROUND AND RELATED WORK

RTE detects and eliminates redundancy using content-based chunking and caching. At the network layer, the RTE process selects *chunks* of an IP packet for caching, with the expectation that future packets will contain some of the already cached chunks. Chunks are blocks of consecutive bytes identified by a short hash value or *fingerprint* (typically a *Rabin fingerprint* [12]). Fingerprints facilitate quick comparison of chunks from each packet to the cached chunks.

Typically, all possible chunks are scanned using a sliding window that advances by 1 byte at a time throughout the data packet. This results in the number of chunks to be on the order of number of bytes in the packet. To keep the number of cached chunks manageable, a sample of all possible chunks is selected, i.e. 1 out of p chunks on average, where p is called a sampling *period*, and is usually between 32 and 64. More selected chunks and larger cache result in higher savings.

The sample selection of chunks can be at fixed distance from each other, random, or dependent on the data content or

TABLE I: ETHERNET TRACES

Inbound	Total	Min	Mean	Max
Trace data (GB)	134.9	8.3	16.9	25.3
Duration (hours)	8	1	1	1
Frames (million)	208.0	15.8	26.0	36.8
% IP data	-	93.0%	94.5%	95.5%
Outbound	**Total**	**Min**	**Mean**	**Max**
Trace data (GB)	114.2	8.7	14.3	18.1
Duration (hours)	8	1	1	1
Frames (million)	238.4	17.3	29.8	37.1
% IP data	-	92.5%	93.1%	94.4%

fingerprint value [10]. It is also possible to have fixed or variable size chunks. In addition to matching single chunks, a matching region can be expanded around the matched chunk to maximize savings. Variable-size chunks and expansion usually yield higher savings, but require more processing power and have a significantly higher memory overhead [10].

When a chunk match is detected between a current packet and the cache, instead of sending the whole chunk, only meta-data is sent inside the packet, consisting of fingerprints and other information. The encoded meta-data is sufficient to reconstruct the original packet at the receiver. Variable-size chunks require more meta-data since the length of the matched content must be included. A more detailed description of techniques for encoding can be found in previous works [1, 7, 10, 11]. This approach to RTE detects redundant content arising from repeated transmission to and from the same user (e.g. HTTP requests and responses), and transmissions of the same or similar content across different users and applications (e.g., downloading popular content). We have earlier explored the effectiveness of RTE based on the underlying application and found that content-aware sampling can yield high gains, by sampling more of the text-based content (such as HTML pages) than the binary (such as JPEG/MPEG content) [10].

The original proposal for RTE envisioned synchronized caches to be placed at the ends of a constrained link inside the network [14]. Today, network appliances are used to save bandwidth and optimize performance between distributed locations of an enterprise [6, 13]. This implementation of RTE is commercially known as WAN optimization. RTE has been proposed as an end-system solution for last-mile links [1] and to improve performance of wireless mesh networks [7]. To date, consideration of RTE for wireless links has been limited. A prior study of end-system RTE included an energy efficiency evaluation of RTE on a smart phone [1], and despite the CPU processing, concluded that RTE uses less energy compared to transmitting full packets.

Lumezanu et al. study RTE in a cellular network and show that link-layer frame loss is detrimental for RTE [11]. Dropped frames are shown to cause serious cache synchronization problems. RTE is ineffective if the sender encodes chunks that have not been successfully transmitted to the receiver. Our approach is to reconsider this important factor and propose a novel solution based on MAC-layer RTE.

TABLE II: WLAN TRACES (AIRUC NETWORK)

	Total	Min	Mean	Max
Trace data (MB)	1,518.9	16.2	42.2	60.0
Duration (s)	53,449	71	1,485	7,820
Frames (count)	4,911,230	38,856	136,423	299,376
RTE data (MB)	1,017.2	7.8	28.3	45.6
Mean data rate (Mb/s)	-	0.07	1.67	6.41
Peak data rate (Mb/s)	-	0.83	8.62	22.46
CRC errors (IP packets)	-	0.5%	7.4%	34.1%
% IP data	-	77.0%	91.9%	99.1%
% original IP data for RTE	-	42.6%	65.0%	84.3%

III. DATA SETS AND METHODOLOGY

The data sets used in this study consist of full payload traffic traces. The first data set comes from the University of Calgary Internet access link, which uses switched Ethernet. The 8 traces are collected on April 6-9, 2006, each 1-hour long. The trace capture started at 9 am and 9 pm on each of four days, and lasted for 1 hour. We refer to this set as Ethernet traces. Table I shows the basic characteristics of the Ethernet traces, including total number of frames, total volume of data at MAC layer, and percentage of all bytes that are found in IP packet payloads. Since separate physical links exist for inbound and outbound campus traffic, we separate the two directions in our analysis.

The second data set consists of 36 traces collected using a wireless sniffer at the University of Calgary AIRUC WLAN. Each trace contains traffic from a single access point (AP). The APs use 802.11 *g* and *n* modes. Traces are collected at busy times of day, between 10 am and 7 pm, from February 8, 2011 to March 29, 2011. The total duration of the collected traces is 14 hours, 50 minutes, and 49 seconds. More detailed information is shown in Table II.

The third set contains traces collected from IEEE 802.11g WLAN with a wireless sniffer on March 11, 2011. A single user transmits the data over the WLAN from the desktop computer to the Network Attached Storage (NAS) drive attached to the access point via Gigabit Ethernet. The traffic is captured in 14 traces (Table III). The data consists of HTML pages from a major university Web site, collected on March 26, 2010. This is highly redundant content distributed across 27,613 files in 5,031 directories. About 11,600 files are created dynamically via server-side script. The transmission is initiated by a copy command in the Windows 7 operating system. A total of 701 MB of data is stored in the data set. For comparison purposes, the popular variant of LZ77 algorithm compresses this data set by 84.6% to 108 MB.

We process traces offline to detect redundancy and calculate achievable bandwidth savings. Redundancy is

TABLE III: WLAN TRACES (SINGLE USER, HTML FILES)

	Total	Min	Mean	Max
Trace data (MB)	740.0	50.0	52.9	56.0
Duration (s)	1,508.0	26.0	107.7	266.0
Frames (count)	1,453,590	62,715	103,828	135,900
Mean data rate (Mb/s)	-	1.6	6.1	17.9
%IP data	-	85.9%	90.0%	94.3%

detected by selecting chunks of data using the DYNABYTE algorithm we developed earlier [9]. DYNABYTE selects all chunks starting with one of the marker bytes. Markers are set dynamically based on the recent history of most frequent bytes, with the expectation that they will well represent the redundant chunks in the near future. The number of markers and sampling frequency are also dynamically adjusted so that the actual number of selected chunks closely follows the target sampling rate, which directly affects the CPU load.

For Ethernet traces, DYNABYTE parameters are: 64-byte chunks, sampling period of 64, and a 500 MB FIFO cache [9]. The WLAN traces are processed using 48-byte chunks, sampling period of 32, and a 10 MB FIFO cache. These parameters offer highest bandwidth savings in WLAN. We use the assumption that all nodes have the same chunk cache for simplicity. This assumption exploits the broadcast nature of the wireless channel and it is valid if the access point uses all data sent and received for its own cache, and all mobile nodes use all data sent, received, and overheard for their own caches. All mobile nodes are in range of the access point, and hear all packets intended for all other nodes. This assumption is similar to the opportunistic caching system proposed and implemented for wireless mesh networks [7].

RTE savings for all data sets are shown in Table IV, expressed as per-trace minimum, mean, and maximum values. The range for WLAN traces is much larger than for Ethernet traces. The averages show that multi-user traces have moderate redundancy, which is expected given the fact that traffic mixes are similar in wired and wireless environments [8]. This indicates that in terms of application mix, RTE can potentially be as effective in wireless as it is in wired environment. Furthermore, there is an opportunity for significantly higher savings in WLAN, with up to almost 50% bandwidth reduction possible for some traces. However, a more careful approach may be warranted for traffic with low redundancy, where processing and encoding overhead of RTE may simply waste resources.

For HTML traces, which are clearly highly redundant and offer an exceptional opportunity for high bandwidth savings, we must also note a drawback. Performing RTE at the IP-layer cannot fully exploit redundancy compared to compression, simply due to packetization. The compressed data set of 108 MB could be transmitted with 77,566 TCP packets, each 1460 bytes long. However, we end up with many more packets where each packet gets reduced in size, leading to sub-optimal, but still very significant bandwidth savings.

IV. CHALLENGES FOR RTE IN WLANS

This section describes four key challenges for RTE in WLAN environments, and compares the effectiveness of RTE between WLAN and Ethernet networks. The mainstream approach is to consider savings at the IP layer, since commercial WAN optimization middle-boxes operate at the IP layer, and consider bandwidth savings in terms of data sent and received by each middle-box. However, once we apply RTE to wireless links, we must also consider all characteristics of the medium, namely scarcity of wireless spectrum, high bit error rate, and protocol overhead. In this paper, we focus on the MAC layer view of RTE and examine bandwidth savings at the MAC layer in WLAN.

TABLE IV: RTE SAVINGS AT IP LAYER

Trace	Min savings	Mean savings	Max savings
Ethernet inbound	6.4%	11.2%	16.8%
Ethernet outbound	8.7%	13.2%	19.6%
WLAN-HTML	41.1%	62.2%	76.1%
WLAN-AIRUC	1.6%	17.1%	48.8%

A. Challenge 1: Control and management traffic

One of the key differences between Ethernet and WLAN traffic is the proportion of IP data. Specifically, due to the presence of control and management frames in a WLAN, the relative proportion of IP data traffic is lower. Furthermore, control and managements frames are not amenable to RTE at the network layer, and they are also too small to be useful for RTE. While IP packets are the most important for RTE at the network layer, an AIRUC trace can have as low as 77% of traffic volume in IP packets (Table II), compared to 86% for HTML (Table III), and 93% for Ethernet traces (Table I). In a lightly loaded WLAN, even less IP traffic is possible.

RTE operating at the network layer can still eliminate redundancy from IP payloads, but the potential savings at the MAC layer drop as the relative volume of IP traffic drops. Savings at the MAC layer are directly proportional to savings at the IP layer and proportion of traffic in IP payloads q_{IP}, i.e. $savings_{MAC} = savings_{IP} \times q_{IP}$. For 90% of data in IP payload and assumed RTE savings of 20%, the net savings on MAC layer are 18% of total bytes. If only 80% of data is in IP payloads, such as in WLAN, then the total potential savings at MAC layer are 16%.

This problem can be dealt with in several ways. One way is to simply suspend RTE for low-volume IP traffic, which would eliminate the overhead associated with RTE processing. Another approach might be to perform RTE at MAC layer, which would include non-IP traffic or MAC headers.

B. Challenge 2: MAC headers

Due to longer protocol headers, the total byte savings per MAC-layer frame are lower in WLAN than in Ethernet networks. The redundant chunks of the packet are typically in the IP payload, not the header, so the longer the header is the smaller are the potential savings. In an Ethernet frame, 14 bytes are used for the MAC header, and 20 bytes for the IP header. Therefore, 34 bytes are not used for RTE. In 802.11 frames, there are 36 bytes of MAC header, and 20 bytes of IP header. Therefore, headers are 65% longer in 802.11 frames. Our traces typically contain up to 1536 bytes in a MAC frame. In this comparison, we disregard the physical layer headers such as synchronization bits and preambles.

While header length itself may not appear to be an important factor considering the total frame length, its effect will be more pronounced if the majority of packets are small. In the context of RTE, we can classify IP packets by size of the IP payload as follows:

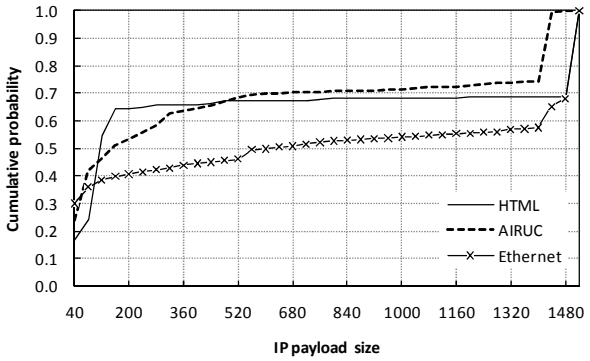

Fig. 1: CDF of all IP payload sizes.

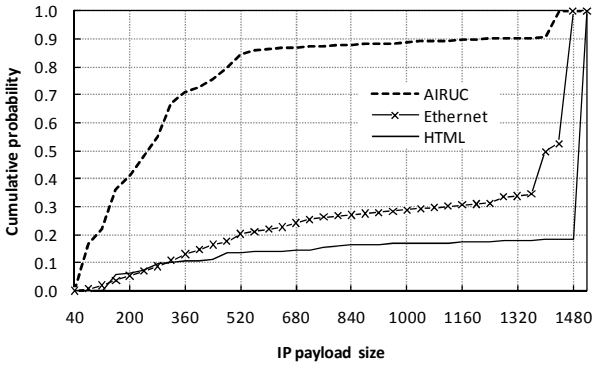

Fig. 2: CDF of IP payload sizes for packets with savings.

- Small packets up to 160 bytes, where at most 2 non-overlapping 64-byte chunks can be found within IP payload, considering the length of TCP or UDP headers.
- Medium packets with 160 - 1320 bytes, and
- Large packets with over 1320 bytes, which offer an opportunity for many redundant chunks to be detected.

We compare frames using IP payload sizes in Ethernet and WLAN traces using a Cumulative Distribution Function (CDF) obtained by partitioning packet sizes in 40-byte bins (Fig. 1). In Ethernet, we have about 40% small, 20% medium, and 40% large packets. In a public WLAN like AIRUC, about 50% of packets are small, 25% medium, and 25% large. For the HTML data set in a single-user WLAN, small packets dominate (65%).

To examine the savings from the three categories of packets, we use CDFs for packets with savings only (Fig. 2). For Ethernet, 65% of packets with savings are large, and for AIRUC only 10% are large. The HTML data set has a very high proportion (80%) of large packets with savings, due to a single bulk data transfer. The important point here is that large packets with savings, which have the smallest header length penalty, are relatively scarce in a public WLAN.

We further confirm the contribution of small and large packets to RTE savings, by computing a CDF using byte savings for each bin, rather than the number of packets contributed (Fig. 3). This CDF tells us how much of the total savings are from small, medium, and large packets.

The difference between Ethernet and WLAN is seen again. Large packets over 1320 bytes contribute the majority of bytes (85%) to RTE savings in Ethernet, whereas small and medium packets contribute the most in WLAN (80%). The combined plots from Fig. 1 and Fig. 3 indicate that about 85% of all savings are obtained from 45% of packets in Ethernet (large packets), and from 50% of packets in public WLAN like AIRUC (medium and large). The consequences of this observation are twofold. First, RTE in Ethernet can ignore over half of all packets (the small ones) and still achieve good savings, while avoiding extra processing overhead. RTE in WLAN cannot afford to do the same, and must process both medium and large packets to achieve 85% of possible savings. Secondly, the MAC header can occupy as much as 1/3 of the packet useful for RTE in WLAN, which suggests different strategies may be needed for WLAN to achieve potential savings, such as combining RTE with header compression [3], or even rearranging MAC header fields so that RTE can detect redundancy in headers as well.

We explore an approach to apply RTE to MAC headers of data frames in a similar fashion as it is applied to IP payloads. The structure of 802.11 MAC headers is such that consecutive frames carrying data between source and destination end points have similar headers. In fact, upon closer inspection, it can be seen that the only field that differs between these frames is the frame sequence number. The sequence number occupies bytes 22 and 23 of the MAC header. Therefore, we have an opportunity to replace the repeating bytes 0 to 21 with a fingerprint (22 bytes).

We first apply RTE with 22-byte chunks to the original MAC headers, using a modest 2 MB cache. Then, we rearrange the header by moving the sequence number field to the beginning of the header, so that the remainder of the MAC header and the LLC header now make a contiguous segment. The rearranged headers now offer a possibility to detect a 30-byte redundant chunk.

We find considerable redundancy in the first 22 bytes of the MAC headers, and much higher redundancy after rearranging the headers for a 30-byte match, as shown in Table V. We adjust the redundancy by an encoding penalty of 12% to obtain estimated savings, using a rationale that at most 4 bytes would be an adequate fingerprint length to avoid collisions when 22 or

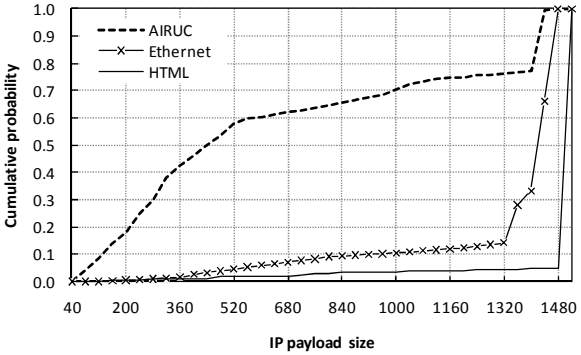

Fig. 3: CDF of total savings that each packet size bin contributed.

TABLE V: SAVINGS WITHIN 802.11 MAC HEADERS

	Redundancy		Estimated savings	
	AIRUC	**HTML**	**AIRUC**	**HTML**
Original headers	64.6%	68.7%	56.9%	60.5%
Rearranged headers	86.2%	93.7%	75.9%	82.5%

30-byte chunks are used. With Rabin fingerprints, the probability of collision would be less than 2^{-23} [12].

The estimated savings indicate the effect on the overall traffic. Since about 80% of frames in AIRUC data sets are data frames, we calculated that their headers account for 11.8% of traffic volume. Reducing the header size by 75.9% (Table V) would yield bandwidth savings of about 9% overall. This would be in addition to 17.1% mean savings from IP payloads. For HTML traces, where about 6% of data volume is in headers, the total benefit to bandwidth savings would be an estimated 4.5%, in addition to 62.2% average savings from IP payloads. The benefit is clearly high enough in a public WLAN to justify a consideration of this approach. It should be noted that the additional processing overhead would be incurred by the MAC layer driver.

C. Challenge 3: MAC-layer retransmissions

There is a fundamental problem affecting RTE in WLAN, and that is *frame retransmissions*. Frame retransmissions in WLAN are caused by poor channel conditions, collisions, and congestion, especially in high-rate traffic. While MAC-layer retransmissions consume bandwidth, they are transparent to RTE at the network layer. The original IP traffic (Table II) could be transmitted up to 5 times at MAC layer, due to a typical limit of 4 retries in 802.11. RTE applied at the network layer only reduces the size of the original IP packet, which then gets transmitted between 1 and 5 times at the MAC layer. Therefore, with respect to the total volume of data, the number of retransmissions can play a major role for RTE savings, as opposed to Ethernet, where physical packet loss is negligible and MAC-layer retransmissions are rare.

Table VI shows the basic retransmission statistics for AIRUC traces. The volume of IP payload data that is retransmitted is substantial. Therefore, it is worth analyzing the effect of retransmissions on RTE. We proceed to calculate the MAC-layer savings for all of our data sets and compare them.

For Ethernet, MAC-layer savings are simply calculated as a ratio of the total bytes saved and total bytes transmitted in frames. Loss and retransmissions at MAC-layer are negligible and hence ignored. In WLAN, if an original IP packet has RTE savings, then all retransmitted versions of that packet at MAC layer will have the same savings.

For WLAN, we consider two cases, *optimistic* and *pessimistic*. These two cases represent the estimates of RTE savings for retransmitted frames in WLAN data sets. The optimistic case assumes that retransmitted frames in each trace have aggregate savings equal to the average savings of the trace at IP layer. This means that the same percentage of bytes is saved from the original frames as well as retransmitted frames. On the other hand, the pessimistic case assumes no RTE savings for the retransmitted frames, as if they resulted from IP packets containing no redundancy at all. This results in low MAC-layer savings. We assume that either original or one of the retransmitted frames is successfully received so it can be used for matching future chunks.

TABLE VI: RETRANSMISSIONS IN WLAN (AIRUC)

Trace	Minimum	Mean	Maximum
Frames	2.2%	8.5%	21.0%
Bytes	4.4%	19.7%	38.9%
IP payload Bytes	4.8%	22.3%	43.9%

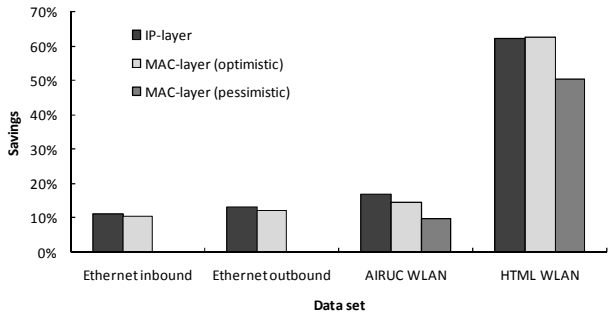

Fig. 4: IP-layer and MAC-layer savings.

The calculations of MAC-layer savings are presented in Fig. 4. Ethernet data sets only have the IP-layer and optimistic case savings, since retransmissions are negligible. WLAN data sets include IP-layer, optimistic and pessimistic savings. We first note that the MAC-layer savings for Ethernet are slightly lower (5-8%) than at the IP layer.

In WLAN, we first note an apparently surprising result. The optimistic case for HTML data has slightly higher savings than IP-layer savings. This is actually possible since IP-layer savings are calculated for IP traffic only, which can be up to about 40% smaller than total traffic (Table VI). Up to 4 retransmissions per frame, especially if retransmitted frames are large, can contribute more byte savings to overall traffic and maintain, or even increase, the byte savings at MAC-layer. Therefore, RTE can be very useful in reducing the overall traffic at MAC-layer. The pessimistic case has lower savings at the MAC layer than the IP layer, by nearly 20%.

For AIRUC traces, we see the expected reduction in MAC-layer savings, for both optimistic and pessimistic cases, by 15% and 43%, respectively. Since this is the most general data set from a public WLAN, we should consider these results closely. While the reduction in savings from the pessimistic case seriously hampers the effectiveness of RTE in WLAN, the optimistic case results are encouraging and indicate that RTE in WLAN can still be worthwhile.

D. Challenge 4:Dropped frames

Another important consequence of high physical loss over wireless links is dropped frames. After being processed by RTE at the network layer, an IP packet may be dropped at MAC layer and cause caches at sender and receiver to become out of sync. The sender using RTE at the network layer does not know if the IP packet has been successfully received, so it

may encode future packets using chunks from prior packets that the receiver has not received. MAC-layer retransmissions that time out rely on higher-layer protocols to retransmit the data, but this has only a limited effect on RTE since higher-layer protocols may retransmit much later or not at all. While the same issue has been investigated in cellular networks [11], we examine its effect in WLAN.

RTE should handle loss gracefully. Proposed approaches to date involve loss detection either at the sender or at the receiver. The sender can detect loss of TCP segments by snooping TCP ACKs. The receiver can detect loss by a cache miss, and then notify the sender using a control packet, which would either black-list the chunks or request their retransmission [11].

Our proposal is to detect loss at the MAC layer of the wireless sender. In 802.11, a MAC-layer ACK is sent for every received frame. The sender will not use a frame and its payload for RTE unless an ACK has been received, which will prevent the encoding problems from happening in the first place. We simulate post-ACK caching, wherein successfully transmitted MAC-layer frames that contain IP packets are used for encoding after being acknowledged at the MAC layer. Dropped frames are not cached and simply discarded.

We consider several different loss scenarios, with packets dropped based on the overall loss probability using either uniform or length-dependent loss probability. Uniform loss probabilities are 0.05 and 0.1 for each IP packet. Length-dependent loss probability varies the probability of loss for each packet based on its length and the average of 5% loss. The factor for loss probability is assigned according to the distribution of packet lengths of retransmitted packets, where small packets under 80 bytes represent 36%, medium packets 26%, and large packets over 1400 bytes represent 38% of all retransmitted IP packets.

Table VII compares the RTE savings without loss and with three levels of loss. For 5% uniform loss, 71% of average savings are preserved by post-ACK caching, and 67% for high 10% loss rate. The length-dependent loss scenario shows that 74% of savings can be preserved. Note that the percentages presented are for the original IP data volume, and not only the data that was successfully transmitted. These results are about 23% better than those obtained by black-listing approach used in previous work in a cellular network [11]. In addition, post-ACK caching does not need overhead traffic to inform the sender about lost chunks.

TABLE VII: RTE SAVINGS WITH POST-ACK CACHING (AIRUC)

Loss scenario	Minimum	Mean	Maximum
No loss	1.6%	17.1%	48.8%
Uniform 5% loss	0.4%	12.1%	32.9%
Uniform 10% loss	0.4%	11.4%	31.0%
Length-dependent loss	0.4%	12.6%	34.3%

V. CONCLUSION

This paper explores specific issues affecting RTE in WLAN. The results show that applying RTE to WLAN links is promising and can potentially yield high bandwidth savings. However, to exploit the full potential of RTE, it is necessary to deal with specific challenges. We describe four challenges (proportion of IP traffic, longer headers, retransmissions, and dropped frames) and discuss possible solutions. We find that including parts of MAC headers in RTE can increase overall bandwidth savings from 17% to 26% in a public WLAN. RTE can in fact deal with many MAC-layer retransmissions by reducing the total size of each frame.

We make a case for MAC-layer RTE that detects frame loss at the sender and caches selected chunks only after they are acknowledged, which preserves 23% more savings than previous approach. We present the simulation results of post-ACK caching, while the Linux driver implementation is underway.

The effects of RTE on frame size in general and on frame retransmissions in particular open an opportunity for cooperative communication for RTE, as a part of the future work.

REFERENCES

[1] B. Aggarwal, A. Akella, A. Anand, et al., "EndRE: An End-System Redundancy Elimination Service for Enterprises," in *NSDI '10: 7th USENIX Symposium on Networked Systems Design and Implementation*, San Jose, CA, 2010, pp. 419-432.

[2] A. Anand, C. Muthukrishnan, A. Akella, et al., "Redundancy in Network Traffic: Findings and Implications," in *ACM SIGMETRICS*, Seattle, WA, USA, 2009, pp. 37-48.

[3] J. Arango, M. Faulkner, and S. Pink, "Compressing MAC Headers on Shared Wireless Media," in *Ad Hoc Networks*. vol. 28: Springer Berlin Heidelberg, 2010, pp. 576-591.

[4] M. Arlitt and C. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications," *IEEE/ACM Transactions on Networking,* vol. 5, 1997, pp. 631-645.

[5] L. Breslau, C. Pei, F. Li, et al., "Web Caching and Zipf-Like Distributions: Evidence and Implications," in *IEEE INFOCOM*, 1999, pp. 126-134 vol.1.

[6] CISCO. "WAN Optimization and Application Acceleration," http://www.cisco.com/en/US/products/ps6870/.

[7] F. R. Dogar, A. Phanishayee, H. Pucha, et al., "Ditto: A System for Opportunistic Caching in Multi-Hop Wireless Networks," in *ACM MOBICOM*, San Francisco, CA, USA, 2008, pp. 279-290.

[8] E. Halepovic, C. Williamson, and M. Ghaderi, "Wireless Data Traffic: A Decade of Change," *Network, IEEE,* vol. 23, 2009, pp. 20-26.

[9] E. Halepovic, C. Williamson, and M. Ghaderi, "DYNABYTE: A Dynamic Sampling Algorithm for Redundant Content Detection," in *IEEE ICCCN*, 2011.

[10] E. Halepovic, C. Williamson, and M. Ghaderi, "Enhancing Redundant Network Traffic Elimination," *Computer Networks,* vol. 56, 2012, pp. 795-809.

[11] C. Lumezanu, K. Guo, N. Spring, et al., "The Effect of Packet Loss on Redundancy Elimination in Cellular Wireless Networks," in *ACM IMC*, Melbourne, Australia, 2010, pp. 294-300.

[12] M. Rabin, "Fingerprinting by Random Polynomials," Center for Research in Computing Technology, Harvard University, Technical Report TR-CSE-03-01, 1981.

[13] Riverbed. "Steelhead," http://www.riverbed.com.

[14] N. Spring and D. Wetherall, "A Protocol-Independent Technique for Eliminating Redundant Network Traffic," in *ACM SIGCOMM*, Stockholm, Sweden, 2000, pp. 87-95.