A First Implementation and Evaluation of the IEEE 802.11aa Group Addressed Transmission Service

Pablo Salvador
Institute IMDEA Networks, University Carlos III de Madrid
josepablo.salvador@imdea.org

Luca Cominardi
University of Brescia
luca.cominardi@gmail.com

Francesco Gringoli
University of Brescia
francesco.gringoli@ing.unibs.it

Pablo Serrano
University Carlos III de Madrid
pablo@it.uc3m.es

ABSTRACT

The IEEE 802.11aa Task Group has recently standardized a set of mechanisms to efficiently support video multicasting, namely, the Group Addressed Transmission Service (GATS). In this article, we report the implementation of these mechanisms over commodity hardware, which we make publicly available, and conduct a study to assess their performance under a variety of real-life scenarios. To the best of our knowledge, this is the first experimental assessment of GATS, which is performed along three axes: we report their *complexity* in terms of lines of code, their *effectiveness* when delivering video traffic, and their *efficiency* when utilizing wireless resources. Our results provide key insights on the resulting trade-offs when using each mechanism, and paves the way for new enhancements to deliver video over 802.11 Wireless LANs.

Categories and Subject Descriptors

C.2.1 [**Computer Systems Organization**]: Network Architecture and Design— *Wireless communication*; C.2.5 [**Local and Wide-Area Networks**]: Access schemes

General Terms

Experimentation, performance, standard

Keywords

Groupcast; WLAN; 802.11aa

1. INTRODUCTION

The IEEE 802.11aa Task Group has recently addressed the lack of efficient mechanisms to support video streaming over WLANs with a new amendment [1]. The motivation is clear: the (now) legacy 802.11 multicast service, even when extended with the service differentiation mechanisms from the 802.11e amendment (i.e., the ability to set different backoff parameters per traffic class), results both inefficient and unreliable, as transmissions are typically performed with one of the 'Basic Service Set' rates (decreasing the overall performance of the WLAN, due to the *performance anomaly*), and there is no acknowledgment for multiple receivers, which is mandatory to improve reliability in case of packet losses.

Indeed, the poor performance of multicast over WiFi has motivated a plethora of proposals [2], which include solutions such as Leader Based Protocols [3], consisting on one station providing feedback on behalf of a group of stations (and therefore requires to properly select one leader), or extensions based on, e.g., the use of "busy tones" [4, 5], used by stations to interfere with the positive acknowledgments. However, most of these extensions require non-negligible modifications to 802.11 operation, or even adding new radio interfaces, and therefore their practicality is uncertain.

The main purpose of the 802.11aa amendment is to extend the widespread 802.11 technology with efficient and reliable mechanisms for the transportation of real-time streams. To this aim, it introduces the *Group Addressed Transmission Service (GATS)*, which is a set of new Medium Access Control schemes to extend the inefficient and unreliable multicast service. Given the relative novelty of GATS and, correspondingly, the little availability of Commercial, Off-The-Shelf (COTS) hardware supporting 802.11aa, it is no surprise the lack of experimental work assessing its performance (very recent works have performed simulation studies only [6]). In this paper, we carry out the first experimental assessment of GATS, which we perform by implementing the new mechanisms over commodity 802.11 cards. This implementation provides us with a hands-on experience on their complexity, which very much complements our efficiency and effectiveness study. More specifically, our main contributions are:

- We describe the implementation of GATS over COTS hardware, reporting the complexity and new functionality required to implement each mechanism. To enable researchers experimentation with the new schemes, the source code of our implementation is publicly available at `http://www.ing.unibs.it/openfwwf/GATS.php`.
- We deploy a test-bed with 30 GATS-enabled nodes, validating the implementation through extensive measurements that confirm the efficiency of the prototype.
- We conduct extensive experiments under a variety of conditions with real video traffic in scenarios with different number of receivers and data stations, assessing the quality of the video received and the resources left for data traffic, and confirm the impact of the configuration parameters of some of the schemes.

The rest of the paper is organized as follows. We provide a brief overview of GATS in Section 2. We describe the implementation of GATS over the 802.11 open-source platform that we use in Section 3. We report the experimental

assessment of GATS in Section 4, and the conclusions and future work are given in Section 5.

2. THE IEEE 802.11AA GATS

With the specification of GATS, the number of mechanisms available to deliver traffic to multiple receivers in 802.11 WLANs is notably enlarged. To this aim, stations first have to form a "group", in order to agree on the address to listen to (the *groupcast concealment address*) and the specific mechanism to use. The group formation can be triggered by the AP or the clients, and they can leverage on newly defined frames (e.g., the *Group Membership Request Frame*) or the existing IGMP protocol.

We next provide an overview of the mechanisms defined by GATS, which we illustrate in Fig. 1 for the case of two receivers, along with the **legacy multicast service** for the sake of comparison (Fig. 1a). We note that the legacy service never retransmits nor uses any type of feedback and fix the Modulation and Coding Scheme (MCS) to the Basic Rate.

The **Directed Multicast Service (DMS)**, illustrated in Fig. 1b, was specified by 802.11v to improve the delivery of management traffic, and 802.11aa extends it to support data frames. As the name implies, the scheme consists on performing, for every frame, a standard unicast transmission for each intended destination. The implementation cost (as we will detail in Section 3.2) is thus very small, but the resource consumption is very high: even with perfect channel conditions, the number of required transmissions per video frame is proportional to the number of receivers.

The DMS ensures reliability by retransmitting a frame as many times as needed, but it results very inefficient because each destination is addressed in unicast. In order to benefit from retransmissions but without so extreme inefficiency, 802.11aa specifies two new mechanisms that constitute the Groupcast with Retries (GCR) service: one mechanism that is relatively simple but not very efficient, and another mechanism that results more complex but improves the use of wireless resources.

The first of these mechanisms is the **GCR Unsolicited Retries (UR)**, depicted in Fig. 1c. It works by *preemptively* transmitting all frames $R+1$ times, to lessen the impact of channel errors. The idea is to improve reliability with a very simple scheme, which does not require a "closed loop" between the sender and the receiver(s), and therefore the price to pay is efficiency: successfully received frames can be retransmitted several times. Unlike the legacy service, this scheme may use also high MCS, hence it may be more efficient despite the unnecessary retransmissions.

The second mechanism is the **GCR Block Acknowledgment (BA)**, depicted in Fig. 1d. The scheme extends the Block Ack operation of the standard to support multiple destinations, and its operation consists on sending a burst of up to "GCR buffer size" consecutive groupcast frames in a burst, and then performing a per-station polling operation to receive the corresponding acknowledgments. In the figure we depict the *immediate* variant of the scheme, in which a backoff process is executed for the transmission of the data burst, and the separation between frames is the minimum specified by the standard, namely, a `SIFS` time. In the *delayed* version of BA (not shown in the Figure) each Block Ack frame is acknowledged by the intended destination, and all transmissions are performed after a backoff operation. This delayed version imposes less stringent requirements on the implementation, as stations have at least one additional backoff to process the received frame(s) and generate the corresponding ones, but results more inefficient and provides worse video service. In contrast to the GCR UR scheme, the GCR BA introduces a notable implementation complexity, with the need of a closed-loop between the transmitter and the receivers to account for acknowledged frames, or the use of a "sliding window" to keep track of the pending frames. This increased complexity, as we will see in Section 4, will lead to the most efficient use of the wireless resources.

3. IMPLEMENTING GATS

A major advantage of GATS is that it does not substantially alter the functionality of the existing MAC, which on the one hand ensures backwards compatibility with legacy stations and on the other hand does not require introducing major changes to the functionality of existing hardware. Indeed, in this section we report how the new mechanisms can be implemented over existing COTS hardware, using the existing open-source firmware `OpenFWWF` that has been used in the past to implement other extensions for 802.11 [7, 8].

3.1 Platform used

We have implemented GATS on Alix 2d2 boxes by *PC Engines*,[1] which embed a Geode LX800 AMD 500 MHz CPU, 256 MB DDR DRAM, 2 mini-PCI slots and a Compact Flash (CF) socket. The software platform is Ubuntu 10.04 Linux (kernel 2.6.36) and as wireless chipsets Broadcom BCM94318MPG 802.11b/g, which supports the open-source driver *b43* and the OpenFWWF firmware.[2]

The implementation consists, basically, on modifying three different modules: the *OpenFWWF firmware*, which is required for time-critical operations (such as, retransmissions, ACKs), the *b43 driver*, for operations with less stringent requirements, and the *mac80211* module of the Linux kernel, so the modifications (e.g., duplicate packet detection) do not impact the behavior of other modules. We next describe the modifications required to implement each scheme.

3.2 Required modifications

Our implementation of GATS requires introducing changes at the *firmware* and the *kernel* modules (i.e., driver and mac80211) of the device. We next provide a qualitative description of the required changes at these modules to implement GATS, and later provide a quantitative evaluation of their complexity.

Directed Multicast Service This mechanism builds on top of the legacy service and therefore its implementation results simple. At the *transmitter* side, a copy of every frame from the application layer is generated for each intended destination, changing the corresponding MAC address. Given that these are not timely operations, the process is performed at the driver, and then the copies are passed to the transmission side. At the *receiver* side, the unicast scheme guarantees that each frame is received only once, so no modifications are required.

Unsolicited Retries This mechanism requires to transmit the same frame $R+1$ times, with backoff but with-

[1] `http://www.pcengines.ch/`

[2] We refer the interested reader on the details of this firmware to its web-page: `http://www.ing.unibs.it/openfwwf/`

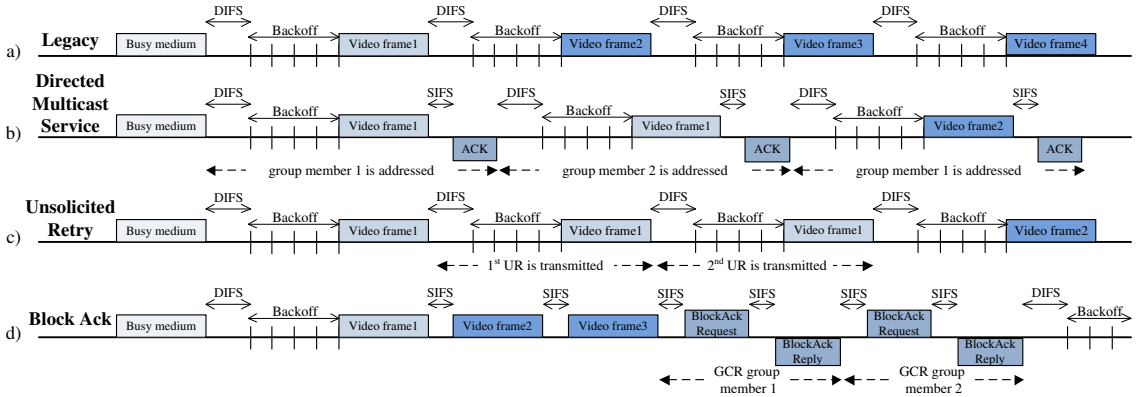


Figure 1: Mechanisms for multicast transmission in 802.11aa WLANs.

out ACKs. Like before, these are not timely operations and therefore the main changes are performed at the driver level. At the *transmitter* side, ACK waiting is disabled at the driver, similarly to multicast frames. The R retransmissions are programmed at the firmware, this being a configurable parameter that we set up in real time (as detailed in Section 4). At the *receiver* side the modifications are three: (i), deactivate at the firmware the transmission of ACKs for frames sent to the configured groupcast address (which is not a multicast address); (ii), replace this address with the stations' one before passing the frame to the upper modules, which is done at the driver; (iii), handle potential duplicates at the mac80211 level.

Block Acknowledgment With this mechanism, the sender can transmit up to "GCR buffer size" consecutive frames in a burst (we denote this number by M), which can be either new frames or retransmissions. As the standard does not specify a policy to schedule these transmissions, we implement the following one. We wait for the queue to fill with M new frames, and do not admit new frames until all these frames are acknowledged by all receivers. In this way, if out of the M frames, N are positively acknowledged, the next transmission burst will consist on $M - N$ frames.[3]

The above requires at the *transmitter* the following changes. First, the driver collects M frames before copying the entire burst to all the hardware queues, so that the firmware may draw the same burst multiple times for handling retransmission. The firmware (re)transmits all frames of the burst that have to be (re)transmitted by spacing them of a SIFS, then it polls receivers with `Block Ack` requests in round-robin (in the order stations joined the groupcast) and processes all replies by storing in a bitmap the frames that have to be retransmitted: if a transmission phase is needed, only such frames are sent.

At the *receiver* side, the firmware updates the reception bitmap (i.e., correctly received frames in the burst) after receiving each frame, which leverages on the computation of CRCs and the sequence number. This bitmap is sent in the `Block Ack Reply` during the polling. Like in the UR case, duplicated frames (i.e., not received by other stations) are handled by the `mac80211` module, and the driver substitutes the groupcast address by the stations'.

[3]This policy is a trade-off between always waiting for the output queue to be filled with M frames, which would maximize efficiency but could introduce overly large delays, and immediately sending frames as they are available, which would minimize delay but at the cost of large inefficiency. We note that, in order to prevent a potential "HOL blocking" caused by receivers with poor link qualities that require too many retransmissions, the AP should not use overly large values for the "MSDU lifetime" parameter.

Table 1: Implementation *cost* of each mechanism.

Scheme	Node	New lines of code			
		Kernel	Firmware	Subtotal	Total
DMS	TX	149	0	149	149
	RX	0	0	0	
UR	TX	50	16	66	104
	RX	21	17	38	
BA	TX	577	618	1195	1632
	RX	132	341	473	

3.3 Implementation summary

We build each mechanism on the standard kernel MAC and driver modules, introducing no changes to the upper layers. The resulting implementation is fully compatible and compliant with the current Linux networking stack[4].

Table 1 hints the "implementation cost" of each mechanism by reporting the number of lines of additional code required for implementing it. Although simply counting lines of code does not take into account other factors like the time needed to design and debug a prototype, still we believe that the table gives a fair quantitative evaluation of the complexity of each mechanism, as DMS and UR result similar in terms of complexity (although DMS requires changing only the transmitter) while BA is (at least) one order of magnitude more complex. As we will see in the next section through real-life experimentation, this increased complexity pays off in terms of performance, although in some scenarios a simpler scheme can result good enough.

4. EXPERIMENTAL RESULTS

4.1 Testbed description and set-up

We deploy a testbed of 30 Alix 2d2 devices acting as wireless stations and one desktop machine acting as AP. The AP uses a 7 dBi omnidirectional antenna and the stations are equipped with 2 dBi omnidirectional antennae. All nodes use a transmission power of 10 dBm and the 802.11g PHY

[4]We refer the reader interested on the detailed description of the implementation to the technical report available at: `http://www.ing.unibs.it/~openfwwf/GATS/tech_report.pdf`.

layer. We set-up a standard desktop machine as the source of video traffic when needed (not shown in the picture). Unless otherwise noted, we run all experiments in channel 14.

All nodes are equipped with a wired interface, which is used to set-up and control the experiments. Depending on the experiment, a set of N_v stations will act as multicast receivers and a set of N_d will act as data stations. For each test scenario, the AP and the N_v stations load through the wired interface our modified version of the 802.11 stack, and use the configuration of the backoff parameters recommended by the EDCA standard for video. For simplicity, we statically configure on these video stations the groupcast address to use (namely, `BE:EF:BE:EF:BE:EF`), as well as other control parameters such as the number of retries R for the case of UR, the maximum burst length M for BA, or the initial sequence numbers.

The N_d stations load the Broadcom 802.11 firmware and b43 driver, and use the MAC parameters of DCF. In this way, we assess the performance of GATS in a scenario with legacy stations, which do not support the service differentiation provided by EDCA. Our assessment therefore constitutes a "worst case" scenario were data stations are very aggressive –we left for future work the analysis and optimal configuration of 802.11aa scenarios were MAC parameters can be tuned. Similarly, data stations will generate uplink and saturated UDP traffic instead of TCP traffic, as the former results in very aggressive contention activity in the channel and, correspondingly, puts the most stringent conditions for the performance of GATS. Finally, throughout our experiments we fix the MCS of data, groupcast and BA request and reply frames to 54 Mb/s, which represents the most stressing conditions for our implementation in terms of frame processing rates,[5] while for the legacy scheme is set to 24 Mb/s. This is also the MCS used for control traffic.

4.2 Synthetic traffic

We first run some experiments with the `mgen` traffic generation tool, to assess the performance of the GATS mechanisms with different input loads. More specifically, we configure the AP for sending CBR traffic of fixed rate r towards $N_v = 10$ receivers, and $N_d = 10$ data stations constantly backlogged for sending UDP traffic towards the AP. Data payload of all frames is fixed to $L = 1400$ B.

For each scenario we are interested in two performance figures, related to the *effectiveness* and *efficiency* of the considered GATS mechanism, which are, respectively:

- Video delivery rate (VDR), which is the average throughput received by stations over the throughput generated by the sender. This figure quantifies the reliability of a given GATS scheme.
- Aggregated data throughput (ADT), which is the sum of the throughputs obtained by data stations, and serves as an indication of the amount of wireless resources left for data traffic (the higher the ADT, the more efficient the multicast mechanism).

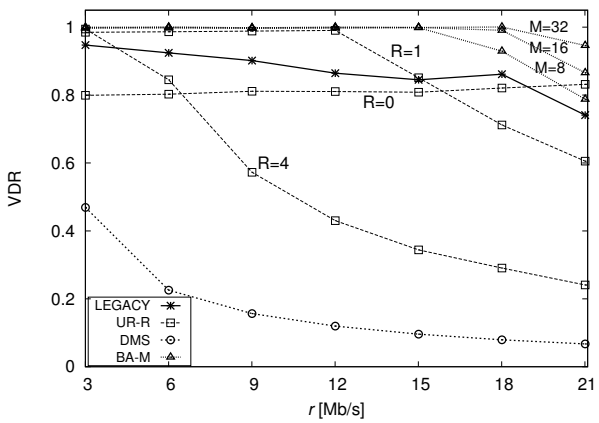
We run 5 experiments of 30 s each for different values of r, and provide the average values of the VDR and ADT in Fig. 2. For the case of VDR (Fig. 2.a), results confirm that BA guarantees reliability for almost all r values (the higher the r the higher the M required to properly prioritize video over data), while DMS suffers from its poor scalability properties, failing to deliver even 50% of the traffic for $r = 3$ Mb/s. Between these two extreme cases, we see a variety of behavior vs. r: the legacy service outperforms UR with $R = 0$ due to the use of a more robust (and lower) MCS, but cannot deliver rates above 18 Mb/s because of the same reason. Similarly, configuring UR with two transmissions ($R = 1$) guarantees delivery for r below 12 Mb/s, but starting from this value suffers from frame drops at the transmission queue (for the case of $R = 0$, these drops start at 24 Mb/s). It is worth noting that, for these cases, $R = 1$ is *enough* to guarantee a good delivery rate, and setting overly large values of R (e.g., $R = 4$) not only provides no good, but even worsens performance due to the frame drops.

We analyze the ADT results, shown in Fig. 2.b. As expected, using UR with $R = 0$ results the most efficient scheme (one transmission per frame at a high MCS), leaving a lot of wireless resources to data stations. In contrast, with the legacy service there is only one transmission per frame, but data throughput is degraded due to the performance anomaly (by increasing r the legacy traffic occupies longer the medium and, consequently, the overall throughput decreases). With $R = 1$, UR provides the same ADT as the legacy for $r \leq 12$ Mb/s, as two transmissions from the video station at the 54 Mb/s MCS consume almost the same amount of resources as one transmission at 24 Mb/s. The figure also illustrates the impact of the M parameter of BA, which serves to improve reliability as seen before, and also leaves more resources to data traffic but at the cost of increasing their latency. Finally, DMS provides a constant ADT that is higher than many schemes depending on the value of r. The reason is that, with this mechanism, the video transmitter performs a binary exponential backoff (BEB) when transmission fails, which leaves more resources for data stations (for the other schemes, the transmitter never performs the BEB, acting more aggressively).

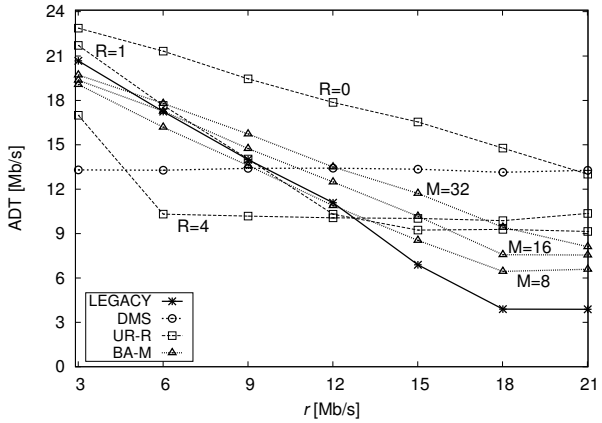
The above assessment is performed using channel 14, which is relatively unpopulated in our testbed. In order to understand the impact of harsher channel conditions on performance, we repeat the same experiments using channel 11, where we detect a number of WLANs with significant activity. For space reasons, we focus on the VDR, with the results depicted in Fig. 3. According to the figure, the performance of the schemes is qualitatively very similar, although in most cases results are worse due to the increased interference, e.g., UR with $R = 1$ fails to guarantee video delivery. Only the BA scheme is able to keep its good performance.

Next, we perform another round of experiments in a more heterogeneous scenario with 10 video receivers, one of them suffering from very poor channel conditions. To enable repeatability of the results, we emulate poor channel conditions by randomly discarding (with probability p=0.25) frames arriving at the impaired node. We measure the VDR for the nine stations with good channel conditions and for the receiver with poor reception, and depict the results in Fig. 4. In the top subplot we present the results of the mean VDR for the stations with good channel conditions, and in the bottom subplot we depict the mean VDR of the station that randomly discards frames. As results illustrate, the GAST mechanisms maintain their relative good performance also in this new scenario for all the stations regard-

[5]Additional experiments, unreported because of space constraints, confirm that the performance with 24 Mb/s MCS is qualitatively similar.



(a) Successful video delivery ratio for different traffic loads.



(b) Aggregated data throughput for different traffic loads.

Figure 2: Performance of the mechanisms with synthetic traffic in channel 14.

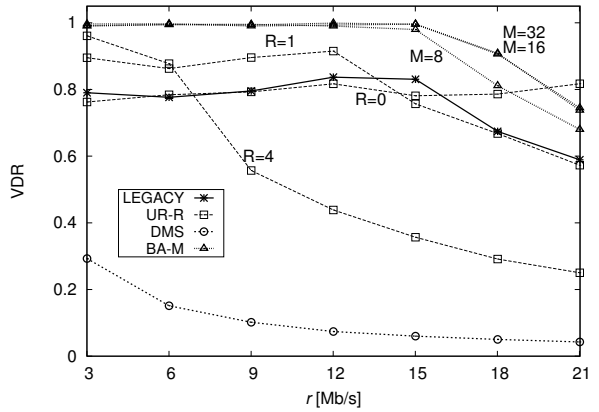


Figure 3: Successful video delivery ratio for different traffic loads in channel 11.

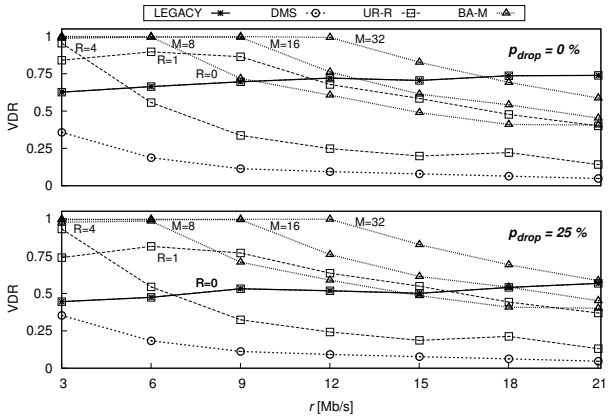


Figure 4: Successful video delivery ratio for different traffic loads in channel 11 with a station with $p_{drop} = 0.25$.

less the channel conditions; for instance, the BA mechanism with $M = 32$ is able to guarantee the video delivery for the impaired station for up to 12 Mb/s. Meanwhile, for this impaired station, the legacy scheme (and UR with $R = 0$) fails to provide 55 % delivery ratio even with 3 Mb/s, while the use of UR with $R = 4$, up to a load of 3 Mb/s, or the BA scheme significantly improves the delivery ratio.

4.3 Real video

We next analyze the performance with real video traffic. To this aim, we stream one minute of the well-known "Big Buck Bunny" video[6] in full HD (1920x1080) encoded with AVI MPEG 4, resulting in an average bitrate of 12 Mb/s. To assess the impact of a high and low number of receivers and data stations, we consider two different cases for both N_v and N_d, namely, 5 and 15, leading to a total number of four different scenarios. We consider an additional value $M = 4$ to understand the impact of this parameter. We plot the resulting VDR[7] vs. ADT for each scenario in Fig. 5.

For the case of BA, the results confirm its good properties, as in most cases its performance is around the top right corner of the figures, i.e., both high VDR and ADT. However, the results also highlight the need to adequately configure the parameter M (the burst length), as overly small values (i.e., $M = 4$) can on the one hand unnecessarily harm the performance of data stations even when the total number of stations is small ($\{N_v = 5, N_d = 5\}$), and on the other hand fail to guarantee the delivery of video when either the number of receivers or the data activity on the WLAN is high (i.e., $N_v = 15$ or $N_d = 15$, respectively). Indeed, a poorly configured BA scheme can be outperformed by the much simpler UR scheme, if adequately configured, when the number of receivers is high (with $N_v = 15$, $M = 4$ vs. $R = 1$). Similarly, the need to limit the number of retransmissions with UR is clear from the figures: configuring UR with $R \geq 2$ results in a worse performance than the legacy scheme in terms of VDR, and similar performance in terms of ADT. Finally, the poor scalability of DMS is confirmed in this bandwidth-hungry scenario, with VDR values well below 20% even for small number of stations.

[6] `http://www.bigbuckbunny.org`

[7] Although in some cases we could compute the "quality" of the received video with, e.g., the peak signal-to-nose ratio, in other cases the resulting VDR is too low for the software tool to properly estimate it, which would have precluded a proper comparison of the multicast schemes.

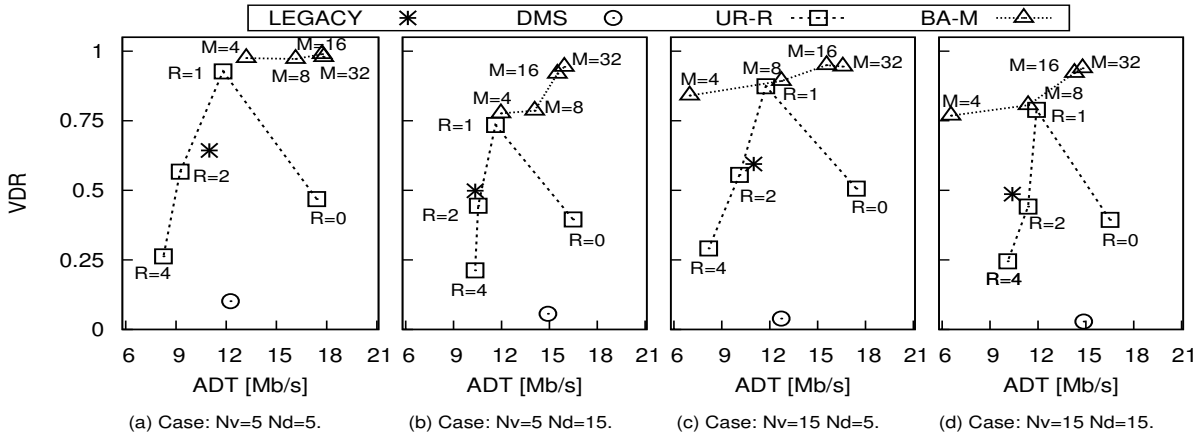


Figure 5: Received video file size and aggregated data throughput for the studied schemes.

Table 2: Assessment of the multicast schemes.

Scheme	Complexity	Effectiveness	Efficiency
Legacy	None	Medium	Low
UR	Medium	Medium-High	Medium
DMS	Medium	None	Medium
BA	High	High	High

5. SUMMARY AND FUTURE WORK

In this paper, we have provided a first implementation and evaluation of the multicast mechanisms that are now available with 802.11aa. We have described how to prototype GATS using COTS hardware, made our implementation available and we have performed an extensive experimental assessment in a variety of scenarios. Our prototype shows that GATS is able to significantly improve the performance of video delivery over 802.11 WLANs, and is implementable over existing devices, unlike previous proposals relying on non-standard or complex functionality. Results confirm that each mechanism offers a specific trade-off between complexity and performance. We provide a summary of our results in Table 2.

Our experimentation has considered a large variety of scenarios, and opens a number of research questions that need to be tackled in the future. Indeed, in our evaluation we have seen that different settings of the R and M lead to very different performance in terms of throughput and video delivery. In this way, one first research challenge to tackle is their optimal configuration, given some performance criterion. Another problem is the design of the best policy to program the Block Ack (e.g., when to hold frames, how to schedule the polling of stations, the setting of the MSDU lifetime parameter). Another major challenge is the design and evaluation of a rate selection algorithm for multicast. We are currently analyzing how to adapt a recent proposal [9] for its use with the 802.11aa mechanisms, in order to perform an extensive evaluation in a mobile scenario.

Given that the standard leaves a lot of room for implementation dependent optimizations (for these and other questions), and based on our previous experiences, we foresee that 802.11aa hardware will have either hard-coded mechanisms whose functionality cannot be altered, or some extensions but poorly documented (at most). By making our implementation publicly available, we hope to foster researchers and practitioners to prototype and assess their optimizations in the above or other areas, "freeing" them from the limitations imposed by closed developments.

Acknowledgments

The authors are grateful to the anonymous referees for their valuable comments which helped in improving the paper.

6. REFERENCES

[1] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 3: MAC Enhancements for Robust Audio Video Streaming*, IEEE Amendment 802.11aa, 2012.

[2] J. Vella and S. Zammit, "A Survey of Multicasting over Wireless Access Networks," *IEEE Commu. Surveys Tutorials*, no. 99, pp. 1–36, 2012.

[3] J. Kuri and S. K. Kasera, "Reliable Multicast in Multi-Access Wireless LANs," *Wireless Networks*, vol. 7, no. 4, pp. 359–369, 2001.

[4] S. Gupta, V. Shankar, and S. Lalwani, "Reliable Multicast MAC Protocol for Wireless LANs," in *IEEE ICC*, vol. 1, May 2003, pp. 93–97.

[5] J. Miroll, Z. Li, and T. Herfet, "Wireless Feedback Cancellation for Leader-Based MAC Layer Multicast Protocols," in *IEEE ISCE*, June 2010, pp. 1–6.

[6] M. Santos, J. Villalon, and L. Orozco-Barbosa, "Evaluation of the IEEE 802.11aa Group Addressed Service for Robust Audio-Video Streaming," in *IEEE ICC*, June 2012, pp. 6879–6884.

[7] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. Miller, "Maranello: Practical Partial Packet Recovery for 802.11," in *Proc. of the 7th NSDI*, 2010.

[8] P. Salvador, F. Gringoli, V. Mancuso, P. Serrano, A. Mannocci, and A. Banchs, "VoIPiggy: Implementation and evaluation of a mechanism to boost voice capacity in 802.11 WLANs," in *IEEE INFOCOM*, March 2012, pp. 2931–2935.

[9] S. Paris, N. Facchi, F. Gringoli, and A. Capone, "An Innovative Rate Adaptation Algorithm for Multicast Transmissions in Wireless LANs," in *IEEE VTC2013-Spring*, Dresden, Germany, 2-5 June 2013.