Extended Abstract: Interference Mitigation in Enterprise WLANs through Speculative Scheduling

Nabeel Ahmed[†*], Vivek Shrivastava[‡*], Arunesh Mishra[‡]
Suman Banerjee[‡], Srinivasan Keshav[†], Konstantina Papagiannaki[§]

[†]University of Waterloo
{n3ahmed,keshav}@cs.uwaterloo.ca

[‡] University of Wisconsin-Madison
{viveks,arunesh,suman}@cs.wisc.edu

[§]Intel Research, Pittsburgh
dina.papagiannaki@intel.com

ABSTRACT

Wireless LANs are commonplace installations in enterprise environments. Their ease of use and deployment, however, are accompanied by a difficulty in their management and security. Proposed solutions to these problems are based on centralization; in the control plane through centralized authentication and allocation of channels and power levels, and in the data plane through time slotted medium access using centralized scheduling for interference mitigation. While centralization of some control plane tasks has been shown to be feasible, centralization on the data plane is significantly harder to realize. This is because it needs to take into account the inherent variability of the wireless medium while offering bounds on delay and jitter on the control paths. In this work, we present a study of the various problems that arise in centralization of the data plane in an enterprise WLAN. We believe that a pragmatic solution for data plane centralization is the key approach to provisioning an enterprise WLAN consisting of a dense deployment of APs.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Interference, Scheduling, Centralization, Conflict Graph

1. INTRODUCTION

Interference in WLANs is typically mitigated using two complementary approaches. First, APs that are likely to cause interference to each other are configured to operate on different non-interfering channels. Second, all 802.11-based wireless transmitters operating on the same channel use standards-based MAC contention mechanisms, e.g., the Distributed Coordination Function (DCF), to avoid collisions.

Ideally, if a sufficient number of non-interfering channels exist, careful allocation of channels to APs can mitigate most sources of RF interference. However, channel allocation may not be sufficient to alleviate interference in dense AP deployments which are being embraced by many commercial vendors, e.g., Aruba Networks, due to advantages such as the ability to pack more transmissions per unit space, greater redundancy and resilience to failures, and lower handoff latencies. As a consequence, the specifics of MAC contention strategy will continue to play an increasingly important role in mitigating interference.

In general, two different approaches for channel contention have been widely used in practice — distributed mechanisms such as random access and centralized techniques such as scheduled access. In a random access mechanism, such as DCF, each transmitter picks a random instant of time while contending for the channel (called its backoff window). The first contender with the smallest backoff window gets the first right of use and all other contenders defer until the first contender finishes its communication. Random access mechanisms tend to be simple and robust in practice — they do not require significant coordination between competing transmitters. Versions of randomized channel access with better fairness properties have also been defined by Nandagopal et. al. [8].

Random access based contention resolution however, can lead to under-utilization since it uses *time* to resolve contention. This is because the medium stays un-utilized until the time instant picked by the earliest contender.

In contrast, scheduled access mechanisms aim to construct a collision-free transmission schedule across different contending wireless nodes.

This construction requires greater coordination between competing wireless transmitters. In general, if the traffic pattern at different nodes is regular and predictable, it is possible to pre-compute an optimal transmission schedule that maximizes any desirable performance metric. Under such predictability assumptions, scheduling based approaches can outperform random access approaches.

Noting these merits of scheduling, multiple prior efforts [10, 8, 5] have designed scheduling strategies that infer and learn traffic patterns along arbitrary single-hop or multi-hop wireless paths. Given that all potential wireless transmitters are not in communication range of each other, an explicit goal of such prior work was to not use any centralized entity for either traffic inferencing or schedule computation.

Examples include work by Vaidya et. al. [10] and Kanodia et. al. [5] both of which require simple extensions to the 802.11 standards to infer and schedule frames in a distributed manner.

Speculative scheduling for enterprise WLANs: Completely distributed scheduling techniques for channel contention can be useful in general wireless environments, e.g., ad-hoc networks or APs in different apartments of a building with no centralized wire-

[*]Student authors are listed in alphabetical order

MobiCom'07, September 9–14, 2007, Montréal, Québec, Canada.
ACM 978-1-59593-681-3/07/0009.

less management capability. Enterprise WLAN environments, however, are quite unique — they typically have many APs distributed over one or more buildings that are centrally managed. This is why all existing and new enterprise WLAN vendors, e.g., Cisco, Meru, and Aruba, have moved towards such centralized management infrastructures. (Unfortunately given the proprietary nature of their products, it is unclear how much centralization they incorporate into their solutions — control path alone or also data path.) In this paper, we describe a general framework for designing efficient channel access mechanisms for enterprise WLANs that leads to a more informed use of the wireless medium. Our approach uses a significant centralized scheduling component, but is augmented with a limited amount of randomized contention. We call this approach *speculative scheduling* — the speculative component is necessary to handle various delay uncertainties in the wireless medium, e.g., due to data rate adaptation algorithms implemented locally at APs, frame re-transmissions, and the need for co-existence with other non-enterprise traffic. We note that our approach does not change the behavior of the underlying 802.11 MAC standard.

The channel contention framework presented in this paper has the following key properties:

Exact knowledge of a significant fraction of traffic: It has been observed that in typical enterprise WLANs, a significant volume of the traffic (nearly 80%) is downlink in nature [9]. This traffic first arrives at an edge router and then gets forwarded to individual APs. A controller, implementing the centralized scheduling function and co-located with the edge router of the WLAN, will therefore have exact knowledge of this large fraction of the overall traffic. It is this exact knowledge of a significant fraction of traffic that makes our centralized scheduling approach feasible.

Hierarchical decision-making for scheduling: Most of the scheduling decision is concentrated with the central controller, which has a global view of the downlink traffic on the WLAN. In particular, it decides which packets are sent to which AP, and when. Centralization of these decisions are possible because these decisions are typically at a millisecond timescale. However, once a packet is scheduled for transmission from an AP, certain follow-up actions need to happen at even faster, typically sub-millisecond timescales. Examples include packet re-transmissions and transmission rate adaptations due to packet losses, and carrier-sensing related deferrals for co-existence with uplink and non-enterprise traffic. These actions are best taken by the AP on its own. AP-level decisions create *a bounded amount of uncertainty at the central scheduler that is managed through speculation*. Importantly, *feedback from the APs can be used to continually refine the scheduler's global view and mitigate some of this uncertainty in future decisions*. This two-stage hierarchy for managing scheduling is a critical design choice in our system.

Flexible design of scheduling policy within the enterprise WLAN: The design of our centralized scheduling framework provides significant flexibility to WLAN administrators. By implementing specific functions within the central scheduler, an administrator may choose to optimize their metric of choice, e.g., throughput, fairness, or some combination of the two. In addition, a single central point of control for most of the data traffic allows an administrator to effectively implement application-specific prioritization, say for applications such as Voice-over-WiFi [4].

No change required at 802.11 clients: This property is important for practical deployments of enterprise WLANs. In particular, our proposed solution does not require 802.11 clients to be aware of the new scheduling techniques or the existence of a central scheduler.

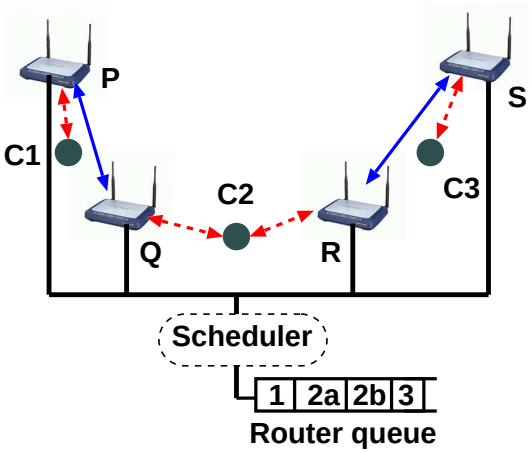

Figure 1: Illustrating gains of centralized scheduling. Arrows indicate nodes that are in range of each other.

2. CENTRALIZING THE DATA PLANE

In our design, scheduling plays a central role in mitigating interference across all APs and clients operating on the same channel. The scheduler lies on the downlink (i.e. edge router to AP) data path, observing all traffic between the edge router and the APs. Note that some slow-time-scale channel allocation algorithm first assigns each client and AP to a channel chosen from a set of orthogonal channels [7, 6]. The scheduler's role is then to decide, for each channel, and for each arriving packet destined to a client assigned to that channel, *which* AP also assigned to that channel and in range of the client should send that packet, and *when*. Clearly, these two decisions are coupled.

In this section, we start with a simple example to illustrate the basic intuition behind centralized scheduling-based channel access. To keep this example simple, we assume no latency or jitter on the wired AP backplane, no data rate adaptation, and an absence of uplink or non-enterprise traffic. Subsequently, we will provide a more complete view of our approach by introducing the speculative component, that will provide robustness against all such uncertainties in scheduling.

2.1 Basic centralized scheduling example

In current WLANs, each client associates to a single AP and exchanges all traffic with it. However, if we were to allow a client to associate with more than one AP (using recent techniques, such as MultiNet [3]), then the centralized scheduling approach can achieve further performance gains. For example, consider the enterprise WLAN scenario shown in Figure 1, consisting of four APs (P, Q, R and S), connected to an Ethernet backbone. APs P and Q are in interference range of each other. Similarly APs R and S interfere with each other and cannot be active simultaneously. At the instant depicted, there are three clients (C_1 to C_3), with C_1 and C_3 associated to their respective in-range APs as shown in the figure. Client C_2, on the other hand is associated to two APs, Q and R. As a result, the scheduler has greater flexibility in scheduling downlink traffic to C_2 — when it knows that Q is in interference range of another scheduled communication (e.g. during P -> C_1), while R is not, the scheduler can send an arriving packet for C_2 through R. If R is similarly conflicted when Q is not (e.g. during S -> C_3), the scheduler can choose Q for downlink communication with C_2. Again, the centralized design of our proposed approach can lead to better utilization of the channel under the multi-AP association assumption.

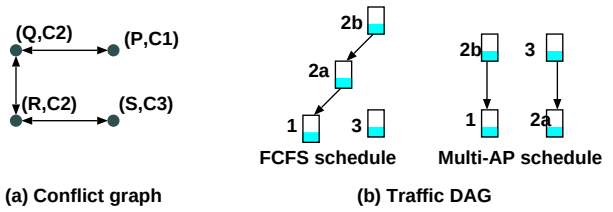

Figure 2: Conflict graph for topology in Figure 1, and two traffic DAGs corresponding to different schedules. The labels on DAG edges are not shown.

An edge router on the same Ethernet segment forwards client traffic from the Internet to the four APs depending on the destination of the packet (C_1 to C_3).

Firstly, suppose the edge router queues packets arriving at a particular instance and immediately forwards them to their respective APs. In the existing de-centralized WLAN data plane architecture, the router will forward these packets to the APs in the same sequence as shown in Figure 1. Let us assume without loss of generality, that in the basic case, client C_2 is associated to AP Q. 802.11's DCF mechanism will be used for channel contention, and the APs will contend for the channel in order to transmit packets 1, 2a, 2b and 3 to clients C_1, C_2, C_2 and C_3 respectively. Contention may be resolved either by waiting different time instants prior to channel access or by resolving a collision through further backoffs.

For simplicity, let us assume that each packet transmission (including the link layer acknowledgment) takes one time slot. Then the best case for DCF (where no collisions occur) will have the following schedule of events:
- Slot 1: Packet 1 - (AP $P \rightarrow C_1$), Packet 3 - (AP $S \rightarrow C_3$)
- Slot 2: Packet 2a- (AP $Q \rightarrow C_2$)
- Slot 3: Packet 2b- (AP $Q \rightarrow C_2$)

That is, the packets require three time slots for successful transmission of the four packets to the clients.

However, a better schedule of these packet transmissions is possible in two time slots using the aforementioned multi-ap association assumption, as shown in Figure 1. This is possible using the following mechanism. Given that the router now has additional knowledge that both Q and R can be used to send data to C_2, it can choose to schedule packets for non-conflicting APs P,R and Q,S in parallel. The new transmission schedule for packets would then be the following:
- Slot 1: Packet 1 - (AP $P \rightarrow C_1$), Packet 2a - (AP $R \rightarrow C_2$)
- Slot 2: Packet 3 - (AP $S \rightarrow C_3$), Packet 2b - (AP $Q \rightarrow C_2$)

The better schedule in this simple example achieves a 33% increase in throughput. Note that a distributed channel access mechanism is unaware of such opportunities and in particular, cannot exploit multi-ap associations like the centralized scheduler. Therefore, it will not be able to construct the most efficient packet transmission schedule across the four packets. However, in our centralized data plane design of an enterprise WLAN, *the scheduler, co-located with the edge router, is in a unique position to observe all downlink traffic* and can, very efficiently, create the optimal two time slot transmission schedule using the multi-ap association opportunity for client C_2.

2.2 Sketching a Design

In order to make its scheduling decisions, the scheduler needs to maintain two important data structures:

A conflict graph: This data structure identifies all pairs of links in the given enterprise WLAN that interfere with each other. From the scheduler's standpoint, vertices in the conflict graph are AP-client links and an edge between two vertices indicates that the corresponding links cannot be simultaneously active. In reality, however, a conflict graph may capture other types of conflicts as well, such as AP-AP and client-client conflicts. This may be represented in a richer conflict graph such as the ones used in [2]. For clarity, however, we only consider AP-client links in our discussion. Trivially, there are no edges between vertices assigned to orthogonal channels.

The conflict graph corresponding to the topology shown in Figure 1(a) is shown in Figure 2(a). Note that the conflict graph is *traffic-independent* and depends only on the (current) topology of APs and clients in the enterprise WLAN [1]. As the topology of the WLAN changes, perhaps due to client mobility or changing characteristics of the wireless environment, the conflict graph changes as well. Therefore, to maintain its accuracy, it must be periodically re-computed.

A downlink traffic DAG: This represents a speculative transmission schedule for all downlink traffic. Because the data structure itself is effectively a *directed acyclic graph*, we refer to it simply as a DAG. In this data structure, vertices correspond to packets awaiting transmission. A directed edge from vertex A to vertex B, indicates that packet B is scheduled to be transmitted before packet A to avoid a potential conflict. Each edge $A \rightarrow B$ is labeled with the time at which B's transmission is expected to complete. All packets with zero out-degree can be scheduled immediately without any conflict. Any other packet can be scheduled for transmission no earlier than the largest label on its outgoing edges.

Figure 2(b) shows the traffic DAG for the four packets that need to be scheduled to clients for the topology in Figure 1(a). The First-Come-First-Served (FCFS) schedule, shown on the left, requires three slots to transmit the four packets. Packets 1 and 3, which have no outgoing edges, can be scheduled for transmission immediately. Once packet 1 is transmitted (and hence, deleted from the DAG), packets 2a and the packet 2b can be transmitted in that order. Assuming no re-transmissions, this schedule needs three slots to transmit the four packets. The schedule, shown to the right, uses multi-ap association to send packets 2a and 2b in parallel with transmissions of 1 and 3 and completes the transmissions in just two slots.

Given the discussion above, it is clear that *any* scheduling approach will consist of: (i) a conflict graph construction and maintenance component — which continuously discovers the (potentially changing) set of conflicts in the topology, and (ii) a traffic DAG management component — which computes the scheduling decision for each downlink packet that arrives on the WLAN and allows uplink and non-enterprise traffic to co-exist in an efficient manner.

3. IMPLEMENTATION AND CHALLENGES

We are currently in the process of implementing the system outlined in the previous section, featuring two tightly-coupled subsystems: a conflict graph constructor (in the control plane) and a centralized scheduler (in the data plane). Both sub-systems run on a $controller$, which is a standard desktop PC. The controller is connected to 30 custom-built APs using a 100 Mbps Ethernet switch. Each AP is built from a Soekris 4826 single-board computer node [1] running the 2.6.16.19 Linux kernel, and is equipped with an Intel 2915ABG mini-PCI wireless adapter. The APs are deployed on five floors of an academic office building.

Although the centralized scheduling algorithm described in Sec-

[1]Hence, this structure is somewhat different from the flow contention graph used for traffic scheduling by Nandagopal et. al. [8] which captures interference between different traffic flows.

tion 2 theoretically illustrates the fundamental advantages of a centralized data plane, in practice, we have found that several additional issues need to be addressed to make it practical and robust. These include: (i) accurate determination of AP-client conflict links, (ii) co-existence of our solution with uplink and non-enterprise traffic, (iii) handling the non-deterministic effects of data rate adaptation and re-transmission algorithms locally running at the APs (or by the wireless NICs in the APs), and (iv) reducing, and dealing with, the latency and jitter on the wired backplane between the scheduler and the APs. We outline some of our observations as we currently tackle these issues, both, from a data plane and a control plane perspective.

In the data plane, we are performing detailed experiments to analyze the impact of the above issues. The following are some of our preliminary observations:

- In our deployment, we observed a mean delay of $500\mu s$ and significant delay-jitter on the wired path from the controller to an AP. We attribute this to a congested wired backbone. This emphasizes the need for a very fast dedicated wired backbone for a practical system that implements centralized scheduling.

- We found that the delay between the time that an AP receives a packet on the wired ethernet and sends the packet out on the wireless card was uniformly distributed between $3000\mu s$ to $6000\mu s$. We attribute this delay to the network queues at the device drivers and also the time spent waiting in the driver for an execution context from the CPU. This problem could be partially solved using a real-time OS or custom hardware that can process packet delivery tasks at the highest priority.

In general, the measured delays in our system appear to be an artifact of using off-the-shelf commodity hardware for implementing the APs and the Controller. We are currently determining the degree to which careful design and software modifications can alleviate some of these delays, thus reducing dependency on customized hardware solutions.

For the control plane, we adopt an active measurement [2] approach for constructing the conflict graph, which is subsequently used for centralized scheduling. In active measurements, APs perform interference experiments with other APs and clients, to discover conflicts amongst each another. Although these interference experiments can fairly accurately determine node conflicts, they have certain constraints that first need to be met. In particular, they have the following two requirements: i) Tight time synchronization between APs participating in each experiment, and ii) A 'clean' (or interference-free) environment in which to perform the experiments. The first requirement is necessary to ensure that all actions across all participating APs occur at exactly the time instances specified in the experiment [2]. The second requirement ensures that any random sources of background noise don't interfere with the outcome of the experiment. Further, we outline some observations made while implementing such a graph construction framework for centralized scheduling:

- Wired delays observed in the control plane are similar in nature to those observed in the data plane. However, problems of delay-jitter observed in the data plane are exacerbated in the control plane, where such variabilities lead to inaccuracies in the execution of our interference experiments. This can be mitigated, in part, by adding sufficient buffer time before each action in order to *mask* such variabilities during the execution of an experiment.

- A common design principle for NICs available today is that radio reception (or energy detection) is performed very aggressively, to accommodate other transmitters in the surrounding region. Therefore, at any given state of the NIC's operation, the first action that is typically performed is to carrier-sense the medium. This is problematic because this mechanism makes it challenging to guarantee that certain actions (e.g. interference experiment transmissions) take place at exact instances of time, as specified in the experiment (and in particular, at microsecond level granularity). In practice, other workarounds may be necessary to enforce such guarantees, for conflict graph-based interference testing (e.g. tuning the radio's receiver sensitivity threshold).

- In [2], the authors propose using IEEE 802.11 standards-based mechanisms such as CTS-to-self, to 'clean' the environment before executing interference experiments. Such a mechanism can, in part, remove interference from sources that are IEEE 802.11 compliant, which are also operating in the same region. In practice, however, we find that many vendors do not properly implement this mechanism, making it ineffective for use in our graph construction framework.

From the observations made above, one crucial point stands out: implementing centralized scheduling on a real system is a challenging and non-trivial engineering task. While the theoretical gains of such a system are easier to appreciate, practical implementation requires solving multiple different real-world challenges that are an artifact of both the software as well as the hardware constraints that are part of such a system. Addressing such challenges forms part of our future work.

Acknowledgements. V. Shrivastava, A. Mishra, and S. Banerjee were supported in part by NSF awards CNS-0639434, CNS-0627589, CNS-0627102, and CNS-0520152.

4. REFERENCES

[1] Soekris Engineering. URL: http://www.soekris.com.

[2] N. Ahmed and S. Keshav. Smarta: A self-managing architecture for thin access points. In *ACM CoNEXT*, 2006.

[3] R. Chandra, P. Bahl, and P. Bahl. Multinet: Connecting to multiple ieee 802.11 networks using a single wireless card. In *IEEE Infocom*, 2004.

[4] T. Henriksson. Hardware architecture for 802.11b based h.323 voice and image ip telephony terminal. In *SSoCC*, 2001.

[5] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly. Distributed multi-hop scheduling and medium access with delay and throughput constraints. In *ACM Mobicom*, 2001.

[6] Y. Lee, K. Kim, and Y. Choi. Optimization of ap placement and channel assignment in wireless lans. In *Local Computer Networks*, 2002.

[7] A. Mishra, V. Brik, S. Banerjee, and W. Arbaugh. A client-driven approach for channel management in wireless lans. In *IEEE Infocom*, Apr. 2006.

[8] T. Nandagopal, T.-E. Kim, X. Gao, and V. Bharghavan. Achieving mac layer fairness in wireless packet networks. In *ACM Mobicom*, 2000.

[9] W. paper from Aruba Networks. Advanced rf management for wireless grids. Available at: http://www.arubanetworks.com/pdf/rf-for-grids.pdf.

[10] N. Vaidya, P. Bahl, and S. Gupta. Distributed fair scheduling in a wireless lan. In *ACM Mobicom*, 2000.