CENTAUR: Realizing the Full Potential of Centralized WLANs through a Hybrid Data Path

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

[†]University of Wisconsin
{viveks,shravan,suman}@cs.wisc.edu

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

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

[*]Google Inc.
arunesh@google.com

ABSTRACT

Enterprise WLANs have made a dramatic shift towards centralized architectures in the recent past. The reasons for such a change have been ease of management and better design of various control and security functions. The data path of WLANs, however, continues to use the distributed, random-access model, as defined by the popular DCF mechanism of the 802.11 standard. While theoretical results indicate that a centrally scheduled data path can achieve higher efficiency than its distributed counterpart, the likely complexity of such a solution has inhibited practical consideration. In this paper, we take a fresh, implementation and deployment oriented, view in understanding data path choices in enterprise WLANs. We perform extensive measurements to characterize the impact of various design choices, like scheduling granularity on the performance of a centralized scheduler, and identify regions where such a centralized scheduler can provide the best gains.

Our detailed evaluation with scheduling prototypes deployed on two different wireless testbeds indicates that DCF is quite robust in many scenarios, but centralization can play a unique role in 1) mitigating hidden terminals — scenarios which may occur infrequently, but become pain points when they do and 2) exploiting exposed terminals – scenarios which occur more frequently, and limit the potential of successful concurrent transmissions. Motivated by these results, we design and implement CENTAUR – a hybrid data path for enterprise WLANs, that combines the simplicity and ease of DCF with a limited amount of centralized scheduling from a unique vantage point. Our mechanisms do not require client cooperation and can support legacy 802.11 clients.

Categories and Subject Descriptors

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

MobiCom'09, September 20–25, 2009, Beijing, China.
Copyright 2009 ACM 978-1-60558-702-8/09/09 ...$10.00.

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Hidden terminals, Exposed terminals, Centralized scheduling, Epoch scheduling, Centralized WLAN

1. INTRODUCTION

An enterprise Wireless Local Area Network (or WLAN) consists of a set of Access Points (or APs) that are under the control of a single administrative authority and restricts access to its own users. Such WLANs have been rapidly deployed in recent years by businesses and university campuses. Early enterprise WLANs had a completely distributed structure: APs independently selected their operating parameters, and independently decided on access control policies. Therefore, popular enterprise WLAN solutions today, come with a central control element that observes the entire network and centrally configures parameters, such as the channel of operation and the transmit power level, of each AP. In recent years, major WLAN vendors, e.g., Cisco, Aruba, and Meru, have realized that a centralized structure is more useful for administrators in managing and securing enterprise WLANs. Moving forward, some research efforts, such as DenseAP [19] and Trantor [20], have proposed centralization of various other management and control functions, e.g., client-AP associations and rate selection across client-AP links.

While it is natural for control plane mechanisms in the enterprise WLANs to be centralized in nature, it is not immediately obvious whether the key data plane mechanism of channel access and contention resolution between multiple competing transmitters, should also be centralized. Today, the primary mode of channel access in enterprise WLANs is the Distributed Coordination Function (DCF) as defined by the 802.11 standard. As the name suggests, it is a *distributed* technique which employs a random access mechanism to resolve contention between multiple competing transmitters.

Given the wasted airtime incurred by random backoff in DCF and the potential for collisions due to uncoordinated access it has been argued that centralization of data transmission decisions can improve network capacity [25, 21, 7, 14]. However, conventional wisdom also suggests that the overhead of centrally scheduling each data packet transmission can be prohibitive, while the DCF approach is simple and has been shown to be adequate for most common scenarios. Therefore, the main question we pose in this paper is the following:

Is there a useful role for a centralized data path in enterprise WLANs in which a central control element makes scheduling decisions about when individual frames should be transmitted by APs that are part of the enterprise?

After detailed experiments, we found significant merit in our conventional wisdom — despite its many known failings, DCF is particularly robust across a large range of scenarios, often more so than a carefully engineered centralized scheduling approach implemented on commodity 802.11 hardware. However, there exist two challenging scenarios, *hidden terminals* and *exposed terminals* where DCF performs poorly and centralization can play a unique role.

The problems of hidden and exposed terminals are classical ones. Different aspects of these problems and their solutions have been systematically studied over multiple decades, starting with seminal work by Tobagi and Kleinrock in BTMA [24], by Karn [15], and by Bharagavan et. al. [8]. The RTS-CTS mechanism (and its adaptive variants [17, 30]) can further be used to mitigate the hidden terminal problem, this mechanism has significant inefficiencies due to wasted airtime and is routinely disabled [2]. In related work, there have been solutions proposed to deal with hidden terminals using coding [13, 12], and exposed terminals by disabling carrier sensing [26]. However, as we describe in Section 7 all these solutions tend to require changes at both APs and clients.

Our proposed approach — CENTAUR

Various studies have shown that a large fraction of enterprise wireless traffic tends to be downlink in nature. Traces collected in three example studies, for the Jigsaw work at UCSD, 2007 [9], at the Sigcomm 2004 conference [22], and at Microsoft Research [10], reveal the downlink traffic fraction to be 77.8%, 69.7%, and more than 85% respectively, while a large enterprise WiFi vendor reports such traffic to be about 80% of the aggregate [27]. Motivated by this observation, in this paper we develop a framework, called CENTAUR, that leverages a limited amount of centralization and explicitly mitigates the performance loss experienced by *downlink traffic* in enterprise WLANs, while indirectly also improving the performance of uplink traffic. More specifically, CENTAUR implements a centralization function for all hidden and exposed terminal links that are identified on the downlink wireless path. All remaining wireless traffic, e.g., uplink enterprise traffic as well as downlink traffic not experiencing hidden or exposed terminal interference, accesses the medium using the standard DCF mechanism. Thus, CENTAUR can be viewed to be half-centralized and half-DCF in nature [1]. We show that such a structure not only helps improve the performance of the downlink hidden and exposed terminals, but also provides an aggregate improvement for the entire WLAN across all uplink and downlink paths. An important property of CENTAUR is that it requires *no changes in the 802.11 clients.* In fact, the entire centralization functionality is implemented in a single central controller, and only requires a small amount of configuration changes in APs. Hence, CENTAUR can be independently implemented and deployed by a WLAN vendor.

Key contributions

The work described in this paper captures a significant research and engineering effort in exploring the role of centralization in enterprise WLANs and makes the following contributions:

• *Demonstrates the importance of addressing downlink hidden and exposed terminal problems:* We start by demonstrating that we are solving a practical problem that occurs in enterprise WLAN settings. We show that downlink hidden and exposed terminals are prevalent in multiple enterprise WLANs through analysis and measurement of production WLANs, as well as measurements on our testbeds. We quantify the performance loss observed due to hidden and exposed terminals in such settings.

• *Demonstrates the role of selective data-path centralization in enterprise WLANs and how it can be implemented independently by a single enterprise WLAN vendor:* We show that a selective amount of data-path centralization is useful in enterprise WLANs in directly mitigating performance loss due to downlink hidden and exposed terminal scenarios. Further, such a mechanism can indirectly help improve the performance of the entire WLAN environment. All proposed mechanisms require no changes in clients and hence can be implemented solely by an enterprise WLAN vendor.

• *Implements and deploys* CENTAUR *over two different testbeds and platforms:* We implement CENTAUR over two different testbeds, each with a different wireless platform, NIC, and wired backplane. (i) Testbed 1: located across five floors of a building consisting of 30 266-MHz Soekris 4826 nodes equipped with Atheros-based 802.11 NICs deployed and interconnected with a 100 Mbps Ethernet backplane, and (ii) Testbed 2 deployed across a single floor consisting of 20 1.2-GHz VIA nodes equipped with Intel 2915 802.11 ABG NICs deployed in a single floor of a building and interconnected with a Gigabit Ethernet backplane.

• *Evaluates* CENTAUR *using controlled experiments and playback of real traffic traces:* We evaluate the performance of CENTAUR through a combination of controlled experiments as well as by playing back real traffic traces on these testbeds. We use different metrics for all our measurements including throughput (UDP and TCP), fairness, completion time of web transactions (http downloads), and MOS for VoIP-like traffic. Example results from our experiments on playback of real traffic traces, under observed periods of high loads, and averaged over all traffic across an enterprise WLAN, include: up to $1.48\times$ improvement in data throughputs, $1.38\times$ reduction in web transaction completion times, and $1.21\times$ improvement in MOS for VoIP-like traffic. Gains for individual hidden and exposed terminal links are obviously much higher.

2. VALIDATING THE PROBLEM

Prior to describing our approach in solving performance problems in enterprise WLANs that occur due to downlink hidden and exposed terminals, we first validate that these are important problems to begin with. Intuitively, it may appear that both hidden and exposed terminal problems can be eliminated by carefully planning the AP locations, and efficiently assigning channels. However, in practice, both these scenarios occur due to arbitrary location of the clients in the system. Fig. 1 shows a scenario where APs X and Y are placed far enough apart that they cannot carrier sense (CS) each other. However, if two clients C_1 and C_2 get positioned as shown in the figure, and associate to the AP with the strongest signal strength, then X and Y are hidden terminals to clients C_2 and C_1 respectively. One might expect that such close-by APs are likely to be on different 802.11 channels

[1] It is analogous to the mythological creature, Centaur, which is supposed to be half-human and half-horse

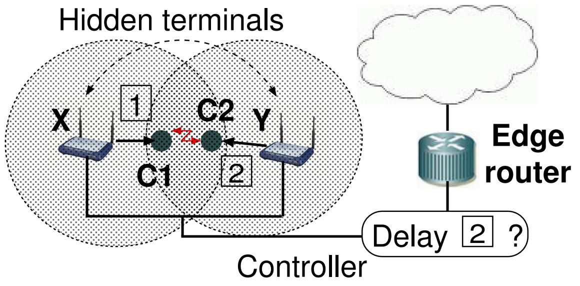


Figure 1: Centralization opportunity in the data path to avoid potential interference effects. Controller can delay the packet 2 for client C_2 to avoid potential collision with packet 1 of client C_1.

mitigating the entire problem. Unfortunately, as analysis in this section shows there are frequent occurrences of these problems even in carefully-deployed enterprise WLANs, even when adaptive channel assignment schemes are used to mitigate interference. Note that if the enterprise WLAN operates in the 802.11b/g mode then the scarcity of orthogonal frequencies is bound to lead to an imperfect channel assignment.

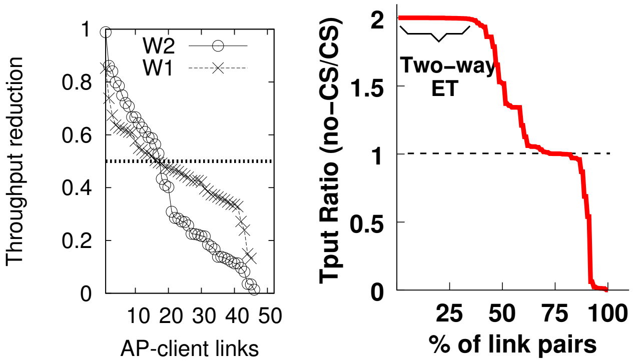
2.1 Quantifying downlink hidden terminals

The Jigsaw effort presented a detailed performance study of a building-wide WLAN in the UCSD campus, consisting of 39 APs and used regularly for Internet access by faculty, staff, and students. It was reported that "co-channel interference from hidden terminals is the likely cause of interference" and for 56% of all interfered traffic, the sender was the AP (i.e., was downlink in nature).

Two production WLANs: Motivated by this observation, we conducted our own measurements of two production 802.11b/g WLANs (W_1 and W_2), each in a different building, each serving hundreds of users daily. These WLANs differ from each other in many significant ways as follows. W_1 spans 5 floors of a building and uses 9 APs manufactured by vendor A. The network administrator was responsible for conducting RF site surveys, identifying locations to place the APs, and manually assigning the channel of operation of each AP to minimize interference. Exactly 3 APs were placed on channels 1, 6, and 11 in W_1 to make the level of inter-AP interference relatively low. In contrast, W_2 occupies a single floor of a different building, uses 21 APs manufactured by a different vendor, B, and features a controller in charge of dynamic channel assignment. The number of APs on each channel, thus, varies over time. In W_2 the vendor was responsible for conducting the RF site surveys and making AP placement decisions.

We placed 45 and 51 nodes in different offices of these two buildings to operate as regular clients to W_1 and W_2 respectively, emulating positions where users typically are located. Once each client associated to a single best AP in each WLAN, we conducted "bandwidth tests" for each pair of AP-client links to identify all occurrences of downlink hidden terminals.

In Fig. 2(a) we show the reduction in throughput due to interference of each AP-client link from its strongest AP-client interferer (relative to the throughput achieved when it operates in isolation). A reduction of throughput around 0.5 is expected if the two links are in carrier-sensing range of each other. However, a reduction in excess of 0.5 implies hidden terminals, with the most severe hidden terminals approaching a throughput reduction of 1 (i.e., zero throughput). This is further confirmed by the increased loss rates for these links. We observe that 16 and 17 AP-client links in W_1 and W_2 respectively (out of 45 and 51) experience some form of hidden terminal interference from other APs in the same WLAN. Further, a few links experience severe hidden terminal interference, i.e. reduction in excess of 0.8. In any production WLAN, even if the number of such hidden terminals is small, the persistent, drastic reduction in throughput for these unfortunate clients makes the WLAN unusable for them.



(a) Impact of hidden terminals (b) Impact of exposed terminals

Figure 2: (a)Throughput reduction due to hidden terminals in production WLANs, W_1 and W_2. Throughput reduction is defined as the ratio of throughput achieved by an AP-Client pair under interference from its strongest interfering AP, to the throughput achieved in isolation. Reduction in excess of 0.5 implies hidden terminals. Severity of hidden terminals increases as throughput reduction approaches 1. (b) Throughput gain for link pairs in CS range (thr without CS/thr with CS). 41% of the link pairs doubled their throughput (two-way exposed terminals), 10% of the link pairs lost throughput (hidden terminals), 20% of the link pairs observe a gain between 1 and 2 (intermittent or one-way exposed terminals). The rest of the links are unaffected.

Further experimentation and analysis revealed that such performance degradation would not be prevented even if the RTS-CTS mechanism were to be enabled. This was primarily because RTS-CTS itself incurred significant airtime overhead.

Summarizing, downlink hidden terminals occur infrequently in enterprise WLAN scenarios but *when they occur they do so with devastating consequences for the clients.* Existing mechanisms, like DCF and RTS/CTS, are unable to address the resulting performance degradation.

2.2 Quantifying downlink exposed terminals

Unlike hidden terminals, exposed terminal occurrences are hard to observe in production WLAN systems. This is because the only real way to identify if a pair of AP-client links are exposed is by disabling carrier sensing at the APs and testing for loss-free simultaneous communication. Unfortunately, it was not feasible to disable the carrier sensing behavior of the APs in these production WLANs. Hence, we evaluate exposed terminals using our own nodes in Testbed 1 and organizing them to mimic the structure of production network W_1 (the closest testbed node to each W_1 AP was chosen to operate as an AP, while the rest of the nodes operated as clients).

Using backlogged UDP traffic we compare the throughput achieved by each pair of links in Testbed 1 with and without CS. We then compute the relative gain obtained in the absence of CS. A value of 1 implies that both experiments led to the same throughput. A value of 2 means that the link doubled its throughput without CS - it was exposed to another link. Fig. 2(b) shows the distribution of throughput gain across all link pairs in the network that were in carrier sense range of each other. We observe that around 41% of the links are exposed terminals that could double their throughput. These observations are consistent with observations in CMAP [26] where exposed terminals were found often in their topologies.

Summarizing, DCF mechanisms miss significant opportunities of throughput improvements when exposed terminals occur. While mechanisms such as CMAP [26] can help, they do not meet our objective of requiring no change in 802.11 clients, and hence cannot be implemented independently by an enterprise WLAN vendor.

3. WHY CENTRALIZATION IS FEASIBLE (AND HOW IT CAN HELP)?

Enterprise WLANs have a useful construction that facilitates significant gains of centralization without much of its overheads. This is because all traffic to this network typically enters through a single edge router (Fig. 1).

Consider the case of two downlink packets (1 and 2) for the two clients C_1 and C_2, associated to APs X and Y respectively. In the traditional DCF mode of operation, the edge router receives these packets and forwards them immediately to the respective APs. Both these packets may get transmitted on the wireless medium simultaneously, leading to interference and packet loss due to the hidden terminal scenario. However, if a controller (co-located at the edge router) realized that such a hidden terminal conflict exists, it might be able to delay packet 2 to a later "time slot," thereby avoiding the collision and packet loss. The key advantage in this design is that by knowing the conflicts in the wireless environment and by observing the previously scheduled downlink traffic, a controller would have a fair estimate on when to transmit a new downlink packet for interference-free reception. Furthermore, given that a dominant fraction of traffic in an enterprise WLAN is downlink in nature (as observed by analyzing traces of [9, 22, 10] and as reported in [27]), such a mechanism can mitigate a significant fraction of potential interference in the enterprise WLAN and improve the levels of contention in the environment as a whole.

Based on these observations, we first present a simple deterministic central scheduling algorithm (called DET) for managing downlink traffic in an enterprise WLAN, that has some performance advantages, but is not without its limitations. In Section 4 we will refine DET to obtain the CENTAUR system.

3.1 A Simple Deterministic Centralized Scheduling Approach (DET)

Assume that the controller can obtain a *conflict graph,* $G = (L, E)$, where L is the set of (AP-client) transmission links and E is the set of conflict edges defined as $E = (L_i, L_j) \mid L_i, L_j \in L$, such that L_i and L_j interfere with each other. Let us assume that a set of packets $P_1, P_2, \ldots, P_r$ have already been scheduled for transmission but are not yet transmitted. Let $\lambda(P_i) \in L$ denote the link on which packet P_i will be transmitted, $t(P_i)$ the corresponding transmission time, and $\tau(P_i)$ the transmission duration. Now consider a new packet P_{r+1} that arrives at the central controller. We use $\mathcal{P}$ to denote the entire packet set $\{P_1, P_2, \ldots, P_{r+1}\}$. For DET we define a simple central scheduling decision where we minimize the time at which the next packet P_{r+1} gets scheduled:

$$\textit{minimize} \quad t(P_{r+1}) \tag{1}$$

with the constraint that any two packets to be transmitted on interfering links should not be scheduled together, i.e., if $(\lambda(P_j), \lambda(P_k)) \in E$, then, $P_j, P_k \in \mathcal{P}$, $t(P_j) \geq t(P_k) + \tau(P_k)$ $\bigvee t(P_k) \geq t(P_j) + \tau(P_j)$.

DET is applied to downlink packets only. Uplink packets from clients to APs continue to use the DCF mechanism for channel access. Therefore, uplink transmissions will interfere with centrally computed schedules. We accept this penalty in our design but still expect significant improvements over DCF.

Implementation

The implementation of DET has two parts. A central controller that implements the simple centralized scheduler and the conflict graph generator using a standard Linux PC (3.33 GHz dual core Pentium IV, 2 GB DRAM) (in about 3,000 lines of C code and a few hundred lines of Perl script), and Soekris- as well as VIA-based wireless APs, modified slightly to improve path latencies. The implementation required a significant engineering effort primarily to get precise timing control between the controller and the APs. The following were some of the salient features: (i) We use a recently proposed technique, called micro-probing [3], to compute the conflict graph in an online fashion, which can compute the entire conflict graph for a 10 AP and 10 client topology in less than 4 seconds[2]; (ii) We implemented the scheduler function as a kernel module in the controller that hooks into the Ethernet driver, and utilizes *high-resolution* timers available in the 2.6.20 version of the Linux kernel; (iii) To tightly control scheduling latencies, we implemented a direct *driver-to-driver communication path* for the APs to allow packets received on the wired interface to be immediately forwarded to the wireless interface, bypassing the kernel network queue (this was especially important for DET but less so for CENTAUR); and (iv) we implemented a wired-ACK mechanism that informs the controller when an AP has successfully transmitted a downlink packet.

3.2 Where DET helps and where it does not ?

To evaluate DET's functionality, we experimented using three different simple canonical topologies involving two AP-client links, where the downlink paths are: i) hidden terminals (HT), ii) exposed terminals (ET), and (iii) normally interfering, but neither hidden nor exposed terminals (non-HT/non-ET). Figure 3 shows the throughput gains of DET (normalized to DCF) for these three topologies, under low, medium and high traffic loads on the two downlinks. For the HT case, DET achieves 2-4× throughput gains over DCF in medium and high loads. Unfortunately, DET provides no advantage for ET and normal terminal cases. In fact, there is a slight loss in performance when compared to DCF in the normal case, especially under high loads.

Understanding DET and designing CENTAUR: The above results made clear that even a simple centralized scheduling technique can provide significant performance gains when downlink hidden terminals occur. However, the performance penalties in the normal interference case, and the lack of gains in exposed terminals need further investigation. It turns out that much of this inefficiency

[2] Note that the conflict graph can be computed in stages with each instance taking 2.5ms. In the scenario of high mobility, active probing may impose high overhead. We are currently exploring passive conflict graph generation techniques that will enable us to update the conflict graph with minimal overhead in real time.

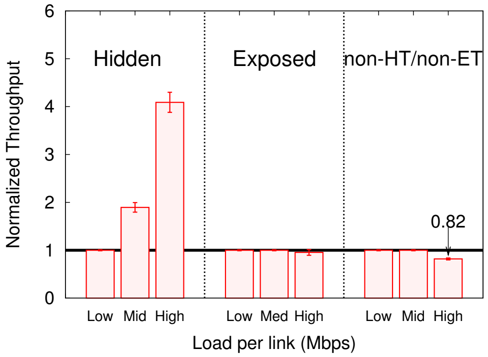
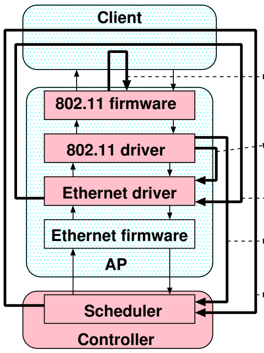


Figure 3: Throughput achieved using DET (normalized to DCF throughputs) on a two-link topology for three different scenarios of HT, ET and non-HT/non-ET. Low, Mid and High represent loads of 1.2 Mbps, 2.4 Mbps and 6 Mbps respectively. Performance gains of DET over DCF increases with increase in traffic load for HT and ET, while the throughput decreases for non-HT/non-ET links under heavy loads due to path latencies.



	Soekris (266MHz) 100 Mbps	VIA (1.2 GHz) GigE
Wireless RTT	2190 (50)	2195 (21)
D2D delay	160 (20)	27 (15)
AP RTT	2550 (80)	2449 (42)
Wired delay	285 (22)	95 (18)
Controller RTT	2800 (200)	2585 (110)

Mean (Variance)
in microseconds

Figure 4: Latencies on Controller-AP-client path that impacts centralized scheduling decisions. Note that Controller RTT = Wired delay + AP RTT .

stems from overheads and inaccuracies in scheduling downlink packets from the controller. In spite of our considerable effort to minimize various delays and their variance between the controller and the APs, some delay and variance in delay persists. Through careful instrumentation of the Atheros wireless driver (Testbed 1) and the Intel ipw2200 wireless driver (Testbed 2), we obtained these delays for different parts of the downlink path (Figure 4). We found that the inaccuracies in estimating the "wired delay" (Figure 4) was a significant contributor to scheduling inaccuracies for traffic.

In the next section, we present our design and implementation path for CENTAUR that masks these delays and their variability effectively using a combination of techniques — *epoch-based scheduling, fixed backoffs, packet staggering,* and *a hybrid model*. Through a combination of all these techniques, CENTAUR achieves throughput gains for exposed as well as hidden terminals scenarios, without sacrificing performance in more common cases.

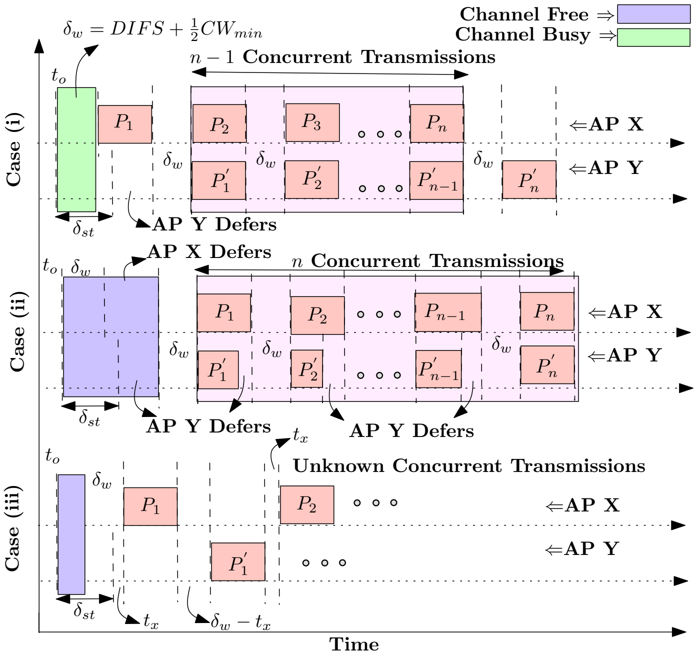


Figure 5: Staggering packets by a time δ_{st} increases transmission concurrency. Cases (i) and (ii) illustrate the scenarios where the channel state remains the same for the back-off duration δ_w therefore synchronizing the transmissions. Case (iii) depicts the scenario where the gains can be unpredictable.

4. CENTAUR DESIGN

CENTAUR incorporates the basic scheduling approach of DET and augments it to mitigate some of its main limitations. We describe this by defining the three main objectives of CENTAUR beyond what DET already provides. They are to: (i) exploit exposed terminals without disabling carrier sensing, (ii) amortize overheads in the scheduling process, and (iii) allow co-existence of uplink as well as non-enterprise traffic by combining our centralization approach with DCF. We describe how CENTAUR meets each objective, in turn.

4.1 Exploiting exposed terminals without disabling carrier sensing

A typical way to allow simultaneous communication over exposed terminal links is to disable carrier sensing. However, disabling carrier sensing for all nodes is particularly dangerous, as it might increase the possibilities of interference. A more intelligent approach is to implement *selective carrier sensing* wherein a transmitter would carrier sense (and therefore back-off) for non-ET links but continue with the transmission for ET links. CMAP [26] is an example of such an approach. However, as the authors discuss in [26], the design of such a mechanism either requires software level modifications for both APs and clients, or it requires a change in the existing 802.11 protocol standard. In keeping with our design goal of requiring no changes at clients or in the underlying 802.11 standard, we achieve simultaneous communication over exposed terminals using an alternate approach as follows: (i) maintain carrier sensing, (ii) use fixed back-offs, and (iii) stagger packets destined to exposed APs. We describe the use of (ii) and (iii) in detail, next.

Fixing back-off intervals

Consider a scenario where n packets are enqueued at each of the APs X ($P_1 \dots P_n$) and Y ($P_1^{'} \dots P_n^{'}$) which are exposed terminals. For simplicity, assume that each packet transmission takes time t_p. In case of DCF, the total transmission time required would be $2nt_p$, excluding backoff and idle times. Now consider a case where backoff at both APs is fixed to some value *bo*. Each of the APs will now only defer for a fixed amount of time, $\delta_w = DIFS + bo$ before transmitting a packet. If we assume a simplistic scenario where both the APs start contending for the medium at the same time, and are successful in transmitting their first packet at the exact same time, then the transmission concurrency on these ET links is doubled. After the first packet transmission, both the APs will sense the carrier to be free for a period of δ_w, and then transmit their second packet at the same time. Thus, all packet transmissions after the first packet are synchronized achieving the effect of disabling the carrier sense[3]. In reality, however, the first packet transmissions are highly unlikely to be synchronized due to wired jitter. In this case, the two APs will get out of sync, and due to carrier sensing, will not be able to transmit simultaneously in the same slot. Therefore, we use packet staggering, which requires delaying the first packet of the two APs relative to each other such that the following packets of both the APs are perfectly synchronized. Next we explain this process in detail.

Packet staggering

Staggering packets P_1 and $P_1^{'}$ by $\delta_{st} > \delta_w$ results in one the three cases shown in Fig. 5: (i) at t_o, AP X starts contending for transmitting P_1 and the channel remains free during the duration δ_w. In this case, AP X transmits the first packet while AP Y defers its transmission due to carrier sense (AP Y had to wait longer to receive its packet due to the fact that $\delta_{st} > \delta_w$). After the first packet transmission, both APs will sense the carrier to be free for a period of δ_w, and then transmit the packets at the same time. Thus, all packet transmissions after the first packet are synchronized. In this case, the total time for transmission is $(n+1)t_p$ ($n-1$ packets are transmitted concurrently and two out of sync), resulting in a throughput gain of $\frac{2n}{n+1}$ (Fig. 5(i)) (ii) the channel remains busy during the duration δ_w, in which case all the n packets are transmitted concurrently, resulting in a gain of 2 (Fig. 5(ii)) (iii) the channel is busy only during some part of the duration δ_w, which results in unpredictable gains as the transmissions of AP X and AP Y may not be synchronized during the entire epoch (Fig. 5(iii)). In CENTAUR P_1 and $P_1^{'}$ are staggered by an amount $\delta_{st} = \delta_w + \gamma \cdot (wired\_jitter)$. We found that the value of $\gamma = 1$ gave the best performance in our testbed. Note that the transmissions in cases (i) and (ii) will be synchronized even when the packet sizes differ for the same link or across links. The effectiveness of packet staggering will also depend on the amount of unscheduled traffic in the network and its interaction with the exposed links. In practice, we show that it leads to remarkable gains over generic traffic mixes (Section 6).

Fairness

In order to contend fairly with other DCF traffic, APs in CENTAUR use a fixed back-off value of $bo = \frac{1}{2}CW_{min}$ which is the average amount of time other transmitters using DCF would spend in deferral. Indeed, experimental results confirm such a property in CENTAUR. The further lack of exponential back-off is not a concern since conflicting links are by design scheduled in different epochs, and are not going to be active simultaneously.

Algorithm 1 CENTAUR : Downlink processing

```
  INPUTS: epoch time ($t_{ep}$), conflict graph $G = (L, E)$
  $max\_ep \leftarrow 0$, $curr\_ep \leftarrow 0$ //Initialize
Procedure ProcessDownlinkPacket($P_i$):
  for each epoch $ep[j]$ in $ep[curr\_ep \dots max\_ep]$
      if canFit ($P_i, ep[j]$) then
          addPacket($ep[j], P_i$); return;
  $max\_ep++$; addPacket($ep[max\_ep], P_i$)
Procedure addPacket($ep[j], P_i$):
  $ep[j].links = ep[j].links \cup \lambda(P_i)$
  if $j \neq curr\_ep$
    $ep[j][\lambda(P_i)].txfill+ = \tau(P_i)$
  else
    $ep[j][\lambda(P_i)].txfill = max(ep[j][\lambda(P_i)].txfill,$
    $curr\_time - ep[j].start\_time) + \tau(P_i)$
  $ep[j][\lambda(P_i)].lastack = P_i$ ; $ep[j][\lambda(P_i)].enqueue(P_i)$
Procedure canFit($P_i, ep[j]$):
  if $\lambda(P_i) \in ep[j].links$ or $((l, \lambda(P_i)) \notin E \ \forall \ l \in ep[j].links)$
  then
    if $j \neq curr\_ep$ then
      if $ep[j][\lambda(P_i)].txfill + \tau(P_i) \leq t_{ep}$ then
        return true
    else
      if $\tau(P_i) + max(ep[j][\lambda(P_i)].txfill,$
       $curr\_time - ep[j].start\_time) \leq t_{ep}$ then
        return true
  return false
```

4.2 Amortizing overhead using epochs

Per-packet scheduling in DET proved to be sub-optimal in generic topologies (without a large number of hidden terminals) due to the delay overhead between the controller and the APs. In essence, DET releases a packet to its intended AP at the time it can get transmitted into the air. The variability in the amount of time it takes for that packet to actually arrive at the AP is what leads to inefficiencies - thus disturbing the inherent timing of the derived schedule.

Epoch-based scheduling: Inefficiencies, described above, can be reduced if the schedule operates on *epochs*, periods of time when packets are transmitted in batches. As long as the batch transmission duration, i.e. epoch, is sufficiently greater than the wired delay variability between the APs and the controller, slight synchronization errors are unlikely to have as significant an effect.

CENTAUR, however, does not only use epochs to amortize the scheduling cost, but also to take advantage of exposed links [4]. Epoch-based scheduling has an important parameter — the time duration of an epoch. This parameter captures an inherent tradeoff between scheduling efficiency and increase in latency experienced by scheduled packets. In particular, the larger the epoch duration, the greater is the scheduling efficiency, but the higher is the path latency experienced by individual packets. After significant parameter sensitivity testing (some results in Section 5), we realized that an epoch duration in excess of 5 ms was sufficient to achieve good scheduling efficiency without adding a high amount of packet latency. To be conservative, we used a default epoch duration of 10 ms in our implementation.

[3] The nodes will indeed carrier sense each other but they won't defer since they will be perfectly synchronized in their transmissions.

[4] Packet staggering is effective if packets are transmitted in batches, which is possible under epoch based scheduling

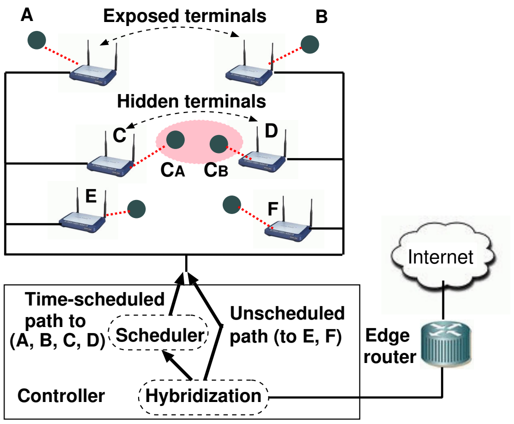


Figure 6: Overview of the CENTAUR hybrid data path.

4.3 Handling downlink non-HT/non-ET, uplink, and non-enterprise traffic

As our experiments will show, the scheduling approach is particularly beneficial to hidden and exposed terminal traffic in the downlink path, while scheduling traffic to non-hidden and non-exposed terminals in the downlink does not provide much gain.
Hybrid data path: To relieve the load on the scheduling system, we partition all downlink traffic into two parts — traffic to hidden and exposed terminals, which gets *scheduled,* and all other traffic, which is *unscheduled.* As Fig. 6 shows, when downlink packets arrive at the controller for hidden or exposed terminals, they get forwarded to the scheduler. All remaining packets are forwarded directly to the APs to be transmitted using the standard DCF mechanisms with carrier sensing and backoffs. Further, all uplink and non-enterprise traffic is, also, unscheduled and contend for the channel using DCF. Since our scheduled traffic continues to use the carrier sensing mechanism, our scheduled traffic can co-exist with all unscheduled traffic. We illustrate this further in Sections 5 and 6.

4.4 Putting it all together

Summarizing, CENTAUR differs from DET in multiple important ways. In particular, CENTAUR includes packet staggering, fixed backoffs, epoch scheduling, as well as the hybrid data path. When a downlink packet arrives, CENTAUR decides first whether to schedule the packet or not. In our implementation we use a generic epoch-based scheduler, whose logic is presented in the pseudo code shown in Algorithms 1 and 2. Whenever a downlink packet is forwarded to the scheduler, it enqueues the packet into one of the epochs, based on the inputs from the conflict graph (G(L,E)), epoch time (t_{ep}) and ETT of the link ($\tau(P_i)$). An epoch therefore consists of multiple packets for each link which are forwarded to the respective AP at the beginning of the epoch. Note that the packets belonging to HT links are packed in separate epochs, thereby ensuring robust conflict resolution. When dealing with ET links, CENTAUR uses packet staggering to increase the possibility of concurrent transmissions. The controller schedules the packets of the next epoch, after receiving the wired acknowledgments of the last packet scheduled on each of the links in the current epoch (Algorithm 2). Measurements on the conflict graph are taken periodically using the micro-probing technique [3] which has minimal overhead. Our evaluation shows performance gains of CENTAUR in spite of such overheads.

Algorithm 2 CENTAUR : Feedback processing

```
Procedure StartNextEpoch():
  For each link in $ep[curr\_ep].links$ do
    if link is ET, use staggering to forward packets
    else forward packets to AP
Procedure ProcessWiredAck($ack$):
  Update the $ETT$ for link $\lambda(ack.id)$
  if got lastacks for all $ep[curr\_ep].links$ then
    $curr\_ep++$; $ep[curr\_ep].start\_time = curr\_time$
    StartNextEpoch();
```

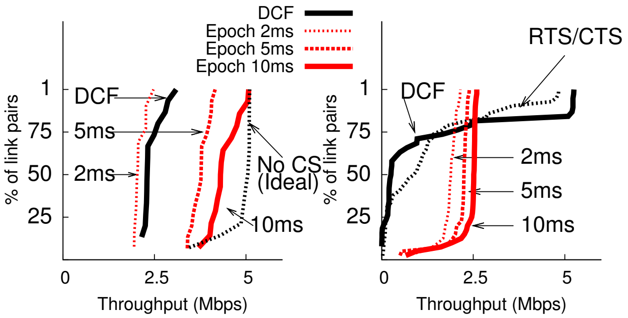


Figure 7: Distribution of throughputs achieved by exposed (left) and hidden (right) link pairs under different access mechanisms. An epoch period of 2ms is equivalent to per packet scheduling. (Testbed 1)

5. CENTAUR MICROBENCHMARKS

To evaluate whether our design of CENTAUR meets our goals, we first present a few micro-benchmarks on targeted scenarios.

5.1 CENTAUR and hidden and exposed terminals

To test the ability of CENTAUR to mitigate hidden and exposed terminal interference, we created topologies with all hidden and all exposed terminal links — 21 and 30 respectively. Further we imposed a high downlink traffic load across all these links to keep them saturated and observed how various versions of CENTAUR compared to DCF, both with and without RTS-CTS. For precise comparison, we fixed the PHY rate at 6 Mbps, packet size to 1440 bytes, and ran each scenario 10 times for 3 minutes each.

ET-only topology

Figure 7(left) shows the distribution of throughput across different exposed terminal links found in the testbed. CENTAUR with a epoch duration in excess of 5 ms is far superior to DCF (median throughput increases from 2.4 Mbps to 4.6 Mbps). In fact, the throughput of all links in the topology improve with CENTAUR. Only CENTAUR with a 2 ms epoch is unable to leverage the gains, because of scheduling inaccuracies at the small epoch size. Disabling carrier sensing completely performs slightly better than CENTAUR. However, a full and robust implementation of such an approach will require client-side changes (as in CMAP [26]) and does not meet our goals.

HT-only topology

Figure 7(right) shows that all variations of CENTAUR (with different epoch times) help mitigate the hidden terminal problems.

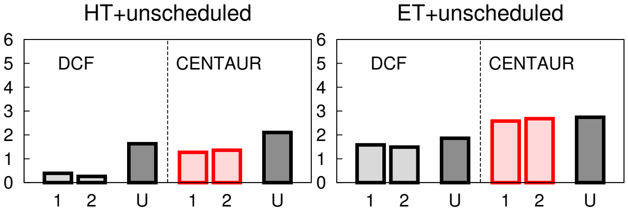


Figure 8: CENTAUR throughput in the presence of unscheduled traffic(Mbps, Testbed 1). Both scheduled and unscheduled link performance improves.

While DCF has a large number of underperforming links (median throughput of 0.2 Mbps without RTS-CTS and 0.8 Mbps with RTS-CTS), CENTAUR with 10 ms epoch has a median throughput of 2.5 Mbps (a factor of 3 and 10 over the two DCF scenarios). The increase in throughput for the hidden terminal links, naturally reduces the throughput of the remaining links. In fact, CENTAUR results in a value of 0.94 for Jain's fairness index, while DCF and RTS/CTS achieve 0.33 and 0.51 respectively.

5.2 Co-existence with unscheduled/uplink traffic

Success of CENTAUR will require efficient co-existence of the downlink scheduled traffic with all unscheduled traffic, including uplink traffic. Therefore, in initial targeted experiments, we created two-link hidden and exposed terminal scenarios (clients 1 and 2), and augmented it with a third client which was responsible for sending continuous uplink traffic (U). There are multiple possible configurations of the client U depending on whether it has symmetric or asymmetric interference with the clients 1 and 2. We evaluated all possible variations of these scenarios, and summarize our observations in Fig. 8. The left plot shows the performance of CENTAUR compared to DCF for the downlink traffic as well as the uplink traffic in the HT scenario. The right plot shows the same for the ET scenario. The results indicate good co-existence properties — in fact, the reduction in interference and contention levels in the downlink, helps the uplink to gain in throughput as well. This is a useful aspect of CENTAUR and helps improve the performance of the entire wireless environment as a whole.

6. CENTAUR EVALUATION

We evaluated the performance of CENTAUR in detailed evaluation over two testbeds emulating the WLAN topologies of $W1$ and $W2$. We have compared the performance of CENTAUR to basic DCF as well as DCF with RTS-CTS. While DCF with RTS-CTS performed slightly better than DCF in HT-only scenarios, in mixed topologies (that include some non-HT/non-ET nodes) it performs worse due to increased overhead. Hence, we do not explicitly plot the DCF with RTS-CTS numbers in this section. All overheads of CENTAUR, e.g., micro-probing [3] are included in our experiments. All results reported are an average of 10 runs, where each run lasted 3 minutes.

Topologies

In all our experiments we emulate the structure of in-building WLANs by placing one testbed AP node near each production APs in the environment. We first present a comprehensive set of results for a *representative mixed* scenario that randomly distributes client nodes into offices with no particular bias. The topology has 7 APs and 12 clients with a mix of hidden (7%), exposed (16%), non-HT/non-ET (44%), and non-interfered scenarios (23%). All experiments are conducted in the 802.11a band to avoid interference with the existing infrastructure WLAN. Although the conflict graph for the same topology might change for different frequency band, it will not affect $CENTAUR$.

Traffic and metrics

We used different types of traffic for various experiments, traversing both directions of the AP-client links. We have experimented with various PHY rates for 802.11 schemes, including the popular auto-rate fallback (ARF) mechanism that dynamically adapts the data rate. Our performance gains are persistent across all scenarios. In order to better interpret our results, most of the data presented in this section illustrate the performance for a PHY rate of 6 Mbps. Results on multiple fixed PHY rates, as well as ARF, are presented at the end of this Section.
Controlled traffic: We used UDP, TCP, as well as VoIP-like traffic (small payloads and frequencies drawn from VoIP traces). The relative volume of uplink and downlink traffic is varied across experiments. We report results on the UDP and TCP throughputs, path delays, and VoIP Mean Opinion Scores (MOS) calculated using [6].
Playback of real wireless traces: From the public SIGCOMM 2004 conference traces [22], we extract the HTTP traffic and partition it into sessions. Each session consists of a set of timestamped operations starting with a connect, followed by a series of sends and receives (transactions), and finally a close. These sessions are replayed on our testbed, by clients, emulating the mechanism described in [10]. Timing gaps between transactions are preserved. We evaluate the delays in completing each of these transactions under different schemes.

6.1 Performance under controlled workloads (representative topology)

We start by examining the throughput, delay, and performance of VoIP-like traffic in our representative scenario. The results are shown in Figure 9.

UDP throughput

Figure 9 (top) shows the UDP throughput of different schemes when the downlink traffic load is upto 6 Mbps per client and the uplink load is upto 1.2 Mbps per client (20% of downlink). CENTAUR with 2 ms epochs provides significant throughput gains for all underperforming links in DCF (especially links 1 and 5) by almost 5×. The aggregate throughput increases from 17.9 Mbps to 18.6 Mbps. However, CENTAUR with 10 ms epochs can take advantage of some exposed terminals and increase their throughput even further (e.g., link 8) by 1.8×. On the whole, CENTAUR with 10 ms epochs improves aggregate throughput across all links 46% over DCF.

TCP throughput

TCP traffic is bi-directional in design due to the return flow of ACKs. In this experiment we have both downlink and uplink TCP traffic with a 80:20 split as before. The overall gains are even higher than UDP. CENTAUR's ability to reduce losses and mitigate interference has an even greater impact on TCP's performance, reflected in the overall throughput gain of 61.5% over DCF.

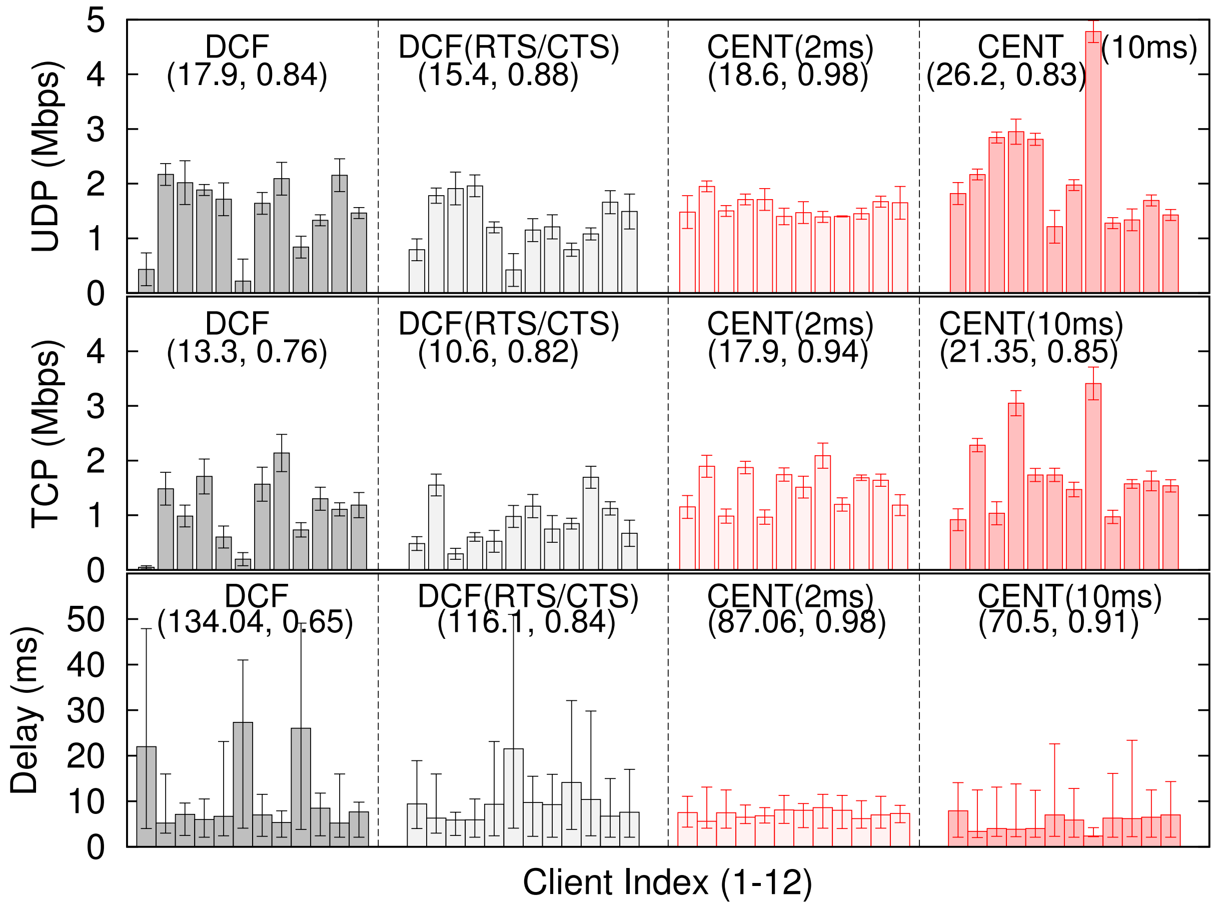


Figure 9: (Testbed 1) Throughput achieved under different mechanisms for a 19 node (7 AP,12 Client) topology. Plot shows the UDP throughput (top), TCP throughput (middle) and UDP delay (bottom). Experiments were run with the uplink data load being 20% of downlink load. 10th and 90th percentile values shown by error bars.

UDP delay

We next examine the performance of UDP delay (Figure 9, bottom). CENTAUR with 10 ms epochs reduces the delay across all links by 47.4% when compared to DCF. The impact is particularly impressive on HT links, since their delay reduces from 49 ms in DCF to 23 ms in CENTAUR. The average delay of CENTAUR with 2 ms epoch is slightly worse than that of CENTAUR with 10 ms epochs since a 10 ms epoch is able to exploit exposed terminals efficiently. In addition, as expected, CENTAUR with 10 ms epochs leads to a higher *variability* in delay as can be observed by the 10th and the 90th percentile values also marked in the plots with error bars. We show next that this does not negatively impact delay sensitive applications.

VoIP traffic

In our VoIP-like traffic experiment, we compute the MOS values of different VoIP streams that were transmitted both in the uplink and downlink directions. Most VoIP implementations use a de-jitter buffer which limits the impact of higher latency on voice quality. However, variability in latency and packet loss are dominant contributors to VoIP MOS. The MOS value can range from 1-5, where above 4 is considered good and below 3 is considered bad. While DCF achieves a MOS of 3.35, CENTAUR with 10 ms epochs achieves a MOS of 3.75. Further, CENTAUR with 2 ms epochs, owing to its lower latency variability achieves a MOS of 4.02. We also observe that HT links get poor call quality (mean MOS was 1.83) due to increased loss rates under DCF, while the mean MOS for these links under CENTAUR was 4.05(2ms) and 3.95(10ms) respectively. Further, the impact on latency can be controlled by limiting the epoch period for scheduling. Variable epoch sizes for different class of applications, will further reduce the impact on latency. We defer such exploration of variable application specific epoch times for future work.

Downlink load (Mbps)	Uplink load (Mbps)	Downlink Throughput			Uplink Throughput median
		10%	50%	90%	
6	1.2	6.78×	1.48×	1.78×	1.15×
6	2.4	3.17×	1.37×	1.75×	1.04×
6	6	2.24×	1.21×	1.53×	1.01×
2.4	1.2	1.05×	1×	1×	1×
2.4	2.4	1.32×	1.11×	1.27×	1.06×
2.4	6	1.68×	1.21×	1.49×	1.18×

Table 1: Normalized throughput gains of CENTAUR over DCF for different combinations of uplink/downlink UDP traffic mix. Each link is operating at 6 Mbps.

Impact of uplink

In order to show the impact of uplink traffic on the performance of CENTAUR, we repeat our experiments with different uplink / downlink profiles. Table 1 shows consistent throughput gains of

Rate	10th percentile	mean gain	90th percentile
6Mbps	6.78×	1.48×	1.78×
12Mbps	8.12×	1.54×	1.67×
Auto	7.43×	1.25×	1.32×

Table 2: Normalized throughput gains of CENTAUR over DCF with different PHY rates and ARF.

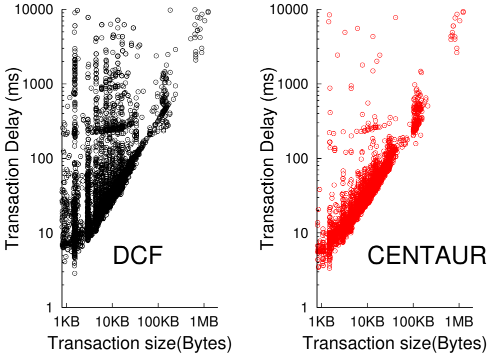


Figure 10: Scatter plot of delay required to complete a transaction during heavy traffic periods under DCF and CENTAUR (Testbed 1). Average transaction delay: 13.8ms (CENTAUR), 29ms (DCF).

CENTAUR with increase in uplink traffic volume (the values in the table are CENTAUR 10 ms epoch throughput gains normalized to DCF). We can infer that the savings in downlink hidden and exposed terminal interference result in more efficient medium utilization improving overall network performance.

Impact of PHY rate and auto-rate fallback (ARF)

In order to understand the impact of higher rates as well as dynamic rate adaptation we repeated our experiments with different fixed rates and with ARF. We use the mechanism presented in [4] to estimate conflicts for multi rate scenarios. Note that multiple data rates can be seamlessly handled by CENTAUR through its dynamic ETT estimation, which packs a variable number of packets in an epoch depending on the data rate being used. Table 2 shows the mean, 10th and 90th percentile throughput gains of CENTAUR over DCF in three cases (fixed 6 Mbps, fixed 12 Mbps, and ARF). We observe that the 10th percentile of the throughput distribution is significantly improved with CENTAUR. This is because the performance gain from mitigating hidden terminals will increase if those links can transmit at higher transmission rates. With ARF, links under HT interference fall back to lower rates while CENTAUR continues to operate at a higher rate, providing persistent gains. Note that the improvement in gain slightly decreases for the 90th percentile and for higher transmission rates. This is because the use of faster transmission rates may “hide” some exposed links from each other. So, CENTAUR will have less exposed links to improve upon.

6.2 Performance with real traffic traces (representative topology)

Finally, we extract the HTTP traffic out of the traffic traces captured at the SIGCOMM 2004 conference and replay it to understand how CENTAUR performs under realistic loads. We partitioned the original trace into a heavy and a light period, based on the total volume of traffic. We evaluated the performance of CENTAUR and DCF separately under the heavy and light conditions. In our experiments, each client emulated the behavior of one real client from the trace, faithfully imitating its HTTP transactions.

Name	Load(MB) (MB)	Session Count	Transaction Count	Ratio of delay (CENTAUR/DCF) 10%	 50%	 90%
Heavy	392	1655	23660	0.53	0.81	0.95
Light	68.2	744	6671	0.62	0.92	0.98

Table 3: Traffic periods replayed and the corresponding ratio of HTTP transaction delay (CENTAUR/DCF).

	Hidden-heavy (Testbed 2)	Exposed-heavy (Testbed 1)	Mixed (Testbed 1)
% HT	14%	0%	6.7%
% ET	0%	22%	10.2%
Overall Gains	34.7%	47.2%	44%
(HT/ET gains)	(HT: 6×)	(ET: 1.7×)	(HT: 3.2×, ET: 1.4×)

Table 4: Normalized throughput gains of CENTAUR over DCF for different representative topologies.

Table 3 shows the load and the corresponding reduction in transaction delay for the different HTTP transactions during the heavy and light periods used for replay. The average transaction delay is reduced to 81% of its DCF counterpart during the heavy period and to 92% during the light period. Clearly, the advantages of the scheduling system are greater under higher loads. Interestingly, the 10th percentile of the delay distribution is significantly improved to 53% and 62% of its DCF value. We further examine the overall improvement in the transaction delay distribution for the heavy period in Fig. 10. Transaction delay is plotted against the transaction size for CENTAUR and DCF. We observe that the transaction delay with CENTAUR is close to expected, while DCF’s delay can be highly variable even for smaller transaction sizes, thus revealing the effect of severe hidden terminal interference.

6.3 Impact of topology

Last, we examine the performance of the different schemes in three different topologies where the fraction of hidden and exposed terminals is varied. Table 4 lists the overall performance results obtained on three types of topologies we constructed — hidden-heavy, exposed-heavy, and mixed. The percentage of hidden and exposed terminals in these topologies are also shown in the table. All these topologies were created by changing the client positions. Uplink traffic load was 20% of the downlink load.

- *Hidden heavy topology (Testbed 2, 10 AP-client pairs):* As expected CENTAUR leads to a significant improvement in performance for all hidden terminals, improving the overall throughput by 35%. The overall fairness (computed by Jain’s fairness index) improves by 89.6% as a result.

- *Exposed heavy topology (Testbed 1, 6 AP-client pairs):* In this topology, CENTAUR again outperforms DCF by 47.2% in system throughput by primarily improving the throughput of exposed terminals.

- *Mixed topology (Testbed 1, 19 nodes):* CENTAUR provides an aggregate throughput improvement of 44%. More results on this topology were presented in Section 6.1.

6.4 Summary of results

A superset of the results presented until now is shown in Table 5. Our results show that (i) CENTAUR resolves HT conflicts efficiently (ii) CENTAUR when used with an epoch of 10ms also successfully exploits ET links. (ii) performance gains of CENTAUR over DCF (w/ and w/o RTS-CTS options) is higher for

Mechanism	Target problem	Approach	Changes to clients or NIC firmware?	Evaluation testbed
CMAP [26]	ET	Conflict graph and DCF	Yes	802.11
ZigZag [12], SIC [13]	Collisions (HT)	Symbol/signal manipulations	Yes	USRP
CENTAUR	HT and ET in enterprise	Conflict graph, DCF, and scheduling	No	802.11

Table 6: Comparing CENTAUR with recently proposed mechanisms of mitigating interference.

TCP flows over UDP flows, (iii) CENTAUR provides higher gains at increased downlink loads (iv) performance gains depend on the amount of unscheduled traffic , (v) gains of CENTAUR also depend on the fraction of HT and ET links in a topology. (vi) CENTAUR improves the overall VoIP quality, with lower epochs performing better as they introduce smaller delay.

7. RELATED WORK

CENTAUR builds on some basic mechanisms like centralized scheduling, epoch based scheduling, which have been the focus of some earlier studies. Besides, solving hidden and exposed terminal problem has been an active area of research for the wireless community. In this section, we provide a brief overview of such related mechanisms and how they differ from CENTAUR .

7.1 Centralized Scheduling

Centralized controllers are commercially available, from vendors such as Cisco [1] and Aruba [27], but they typically operate only in the control plane. Centralization of data, though recognized as providing more control, is harder to implement, and therefore less common. A few examples of such design exist. For example, Meru Networks has proposed cellular-like coordination of various APs and scheduling mechanisms to provide a certain degree of deterministic channel access in enterprise WLANs [28, 29]. The proprietary nature of Meru's solution makes it difficult to present a detailed comparison with Centaur. However, through private communication we have established that Meru's solution has some fundamental differences from Centaur's approach of hybridization and in the specific mechanisms implemented to detect and to handle the exposed and the hidden terminals.

In the research community, people have thoroughly studied scheduling based channel access and there is a large body of literature dealing with efficient scheduling in cellular networks [5]. In the 802.11 context, researchers have studied distributed scheduling techniques for multi-hop or ad-hoc networks [25, 14]. The centralized scheduling technique most closely related to our work, is by Bejerano and Bhatia [7], who propose a PCF-style polling based channel access for APs and clients. However, MiFi requires modifications to clients and unlike our work, MiFi focused more on the efficient design of fair algorithms, and was evaluated through simulations.

While all these techniques of data path centralization are intuitively appealing, to the best of our knowledge, there exists no careful study on the feasibility of data plane centralization for enterprise WLANs through prototype design and large scale implementation. In our work, we have presented a first-of-its-kind implementation-based evaluation of the challenges associated with such data plane centralization.

7.2 Epoch based Scheduling

The notion of epochs itself is not new, and has been used in other scheduling problems, including for wireless networks [18]. However, such prior work used epochs to aggregate knowledge about traffic demands in a distributed environment, while we use epochs to hide inaccuracies in scheduling due to variable latencies on the path for downlink scheduling in enterprise WLANs. More importantly their mechanisms required inherent changes to the clients, while a goal in CENTAUR is to keep the clients unchanged.

Further, mechanisms like TXOP in 802.11e [11] and packet aggregation in 802.11n [23] also provide uninterrupted channel access to wireless transmitters for extended periods of time. However such mechanisms are orthogonal ways for implementing epoch based scheduling and we believe that CENTAUR can make use of any such mechanism to amortize the overhead of wired acknowledgments in centralized scheduling.

7.3 Hidden and Exposed terminals

Solving hidden and exposed terminal problems has been a major focus for the wireless research community. We present a brief synopsis of such mechanisms in Table 6. Recent work [12, 13, 16] proposes novel physical layer mechanisms that can recover frames from collisions (due to hidden terminals and other scenarios) efficiently. In [26], authors present a system, CMAP, that infers interference between links on the basis of packet reception probability and opportunistically disables carrier sensing whenever possible. While such mechanisms can provide substantial throughput gains in interference prone WLANs, they require firmware changes to the receiver's wireless NIC, which makes it difficult for them to be readily deployed in current WLAN scenarios with legacy devices. In contrast, CENTAUR only requires a software update to the wired Ethernet driver at the centralized controller, making it an attractive approach for current enterprise WLANs that want to support legacy wireless devices. This ease of deployment was a critical factor that enabled us to implement and test our system on two different testbeds with relative ease.

8. CONCLUSION

In this paper we explored the question of whether there is a useful role for a centralized data path in enterprise WLANs. We showed that while centralization does not offer gains in all cases, it has a very significant role to play in mitigating downlink hidden terminals and exploiting downlink exposed terminals. We proposed CENTAUR, a hybrid architecture that centrally schedules hidden and exposed terminals, while employing DCF for uplink and legacy downlink traffic. It is based on the novel use of epoch-based scheduling, fixed backoff, packet staggering and the use of a hybrid data path. We showed that CENTAUR is able to deliver significant performance gains for scheduled traffic, but also improves the performance of the network as a whole due to the improved utilization of the wireless medium. Importantly, CENTAUR can be implemented by any individual WLAN vendor without any changes required for clients.

Acknowledgment

We wish to acknowledge Sharad Saha and Siddharth Barman, for help during the experiments and providing constructive feedback on the earlier versions of the paper. Vivek Shrivastava and Shravan Rayanchu were supported in part by the US NSF through awards CNS-0639434, CNS-0627589, CNS-0627102, CNS-0520152, and CNS-0747177. We would also like to thank the anonymous

Section	Experimental setup	Evaluation scenario	CENTAUR Gains
§ 3.2	2-link HT/ET/non-HT/non-ET	DET vs. DCF	HT:4×, ET:1×,non-HT/non-ET:0.82
§ 5	HT/ET links (Testbed 1)	DCF, DCF(w/ RTS-CTS), DET, CENTAUR	10× for HT, 1.89× for ET
§ 5	2-link HT/ET	CENTAUR vs. DCF with unscheduled traffic	1.4×, Uplink: up to 1.6×
§ 6	20-node HT-heavy (Testbed 2)	CENTAUR vs. DCF (UDP, 20% uplink)	1.34×, HT: up to 6×
§ 6	12-node ET-heavy (Testbed 1)	CENTAUR vs. DCF (UDP, 20% uplink)	1.47×, ET: up to 1.7×
§ 6.1	19-node Mixed (Testbed 1)	CENTAUR vs. DCF (UDP, variable uplink/downlink)	up to 1.48×, HT: up to 6.78×, ET: up to 1.78×
§ 6.1	19-node Mixed (Testbed 1)	CENTAUR vs. DCF (TCP, 20% uplink)	1.61×, HT: up to 7.4×, ET: up to 1.64×
§ 6.1	19-node Mixed (Testbed 1)	Impact on delay	47% (reduction in delay)
§ 6.1	19-node Mixed (Testbed 1)	Effect on VoIP traffic	1.4× (MOS for HT links)
§ 6.1	19-node Mixed (Testbed 1)	Effect of PHY rate and ARF	1.54× (12 Mbps), 1.25× (ARF)
§ 6.2	19-node Mixed (Testbed 1)	CENTAUR vs. DCF (Replay of real traces)	up to 0.53× (transaction delay)

Table 5: Summary of evaluation results. Gain is reported for throughput unless otherwise noted

reviewers whose comments helped bring the paper into its final form.

9. REFERENCES

[1] Cisco Wireless Control System. http://www.cisco.com/en/US/products/ps6305/.

[2] Intel pro/wireless network connection for mobile. http://www.intel.com/network/connectivity/products.

[3] N. Ahmed, U. Ismail, S. Keshav, and K. Papagiannaki. Online estimation of rf interference. In *ACM CoNext*, 2008.

[4] N. Ahmed, U. Ismail, K. Papagiannaki, and S. Keshav. Measuring multi-parameter conflict graphs for 802.11 networks. *Mobile Computing and Communications Review 2009*.

[5] M. Andrews. A survey of scheduling theory in wireless data networks. In *IMA summer workshop on wireless comm.*, 2005.

[6] A. Balasubramanian, R. Mahajan, A. Venkataramani, B. N. Levine, and J. Zahorjan. Interactive wifi connectivity for moving vehicles. *SIGCOMM 2008*.

[7] Y. Bejerano and R. Bhatia. MiFi: a framework for fairness and QoS assurance in current IEEE 802.11 networks with multiple access points. In *IEEE INFOCOM*, 2004.

[8] V. Bhargavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A Media Access Protocol for Wirelesss LANs. In *SIGCOMM*, 94.

[9] Y. chung Cheng, J. Bellardo, P. BenkÃű, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *ACM SIGCOMM*, 2006.

[10] J. Eriksson, S. Agarwal, P. Bahl, and J. Padhye. Feasibility study of mesh networks for all-wireless offices. In *MobiSys*, 2006.

[11] E. P. Evaluation, S. Choi, J. Prado, S. Shankar, and N. S. Mangold. Ieee 802.11e contention-based channel access. pages 1151–1156, 2003.

[12] S. Gollakota and D. Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. In *ACM SIGCOMM*, 2008.

[13] D. Halperin, T. Anderson, and D. Wetherall. Taking the sting out of carrier sense: interference cancellation for wireless lans. In *ACM MobiCom*, 2008.

[14] 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.

[15] P. Karn. Maca — a new channel access method for packet radio. In *ARRL/CRRL Amateur Radio Computer Networking Conf.*, 1990.

[16] S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: Analog network coding. In *ACM SIGCOMM*, 2007.

[17] J. Kim, S. Kim, S. Choi, and D. Qiao. Cara: Collision-aware rate adaptation for ieee 802.11 wlans. *INFOCOM 2006*.

[18] R. R. Kompella, S. Ramabhadran, I. Ramani, and A. C. Snoeren. Cooperative packet scheduling via pipelining in 802.11 wireless networks. In *in Proceedings of ACM SIGCOMM E-WIND*, 2005.

[19] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill. Designing high performance enterprise wi-fi networks. In *NSDI*, 2008.

[20] R. Murty, J. Padhye, A. Wolman, and M. Welsh. An Architecture for Extensible Wireless LANs. In *ACM Hotnets*, 2008.

[21] T. Nandagopal, T.-E. Kim, X. Gao, and V. Bharghavan. Achieving mac layer fairness in wireless packet networks. In *MobiCom'00*.

[22] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, J. Zahorjan, and E. Lazowska. CRAWDAD data set uw/sigcomm2004.

[23] P. Thornycroft. Designed for speed: Network infrastructure in an 802.11n world. In *Aruba Networks white paper*, 2007.

[24] F. Tobagi and L. Kleinrock. Packet switching in radio channels : Part ii - the hidden terminal problem in carrier sense multiple access and the busy tone solution. *IEEE Transactions on Communications*, (12), Dec. 1975.

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

[26] M. Vutukuru, K. Jamieson, and H. Balakrishnan. Harnessing Exposed Terminals in Wireless Networks. In *NSDI*, 2008.

[27] White-paper from Aruba Networks. Advanced RF management for wireless grids. http://www.arubanetworks.com/pdf/rf-for-grids.pdf.

[28] White-paper from Meru Networks. Microcell deployments: Making a bad problem worse for pervasive wireless LAN deployments. http://www.merunetworks.com/pdf/whitepapers/.

[29] White-paper from Meru Networks. Virtual cells: The only scalable multi-channel deployment. http://www.merunetworks.com/pdf/whitepapers/.

[30] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *MobiCom '06*.