OmniVoice: A Mobile Voice Solution for Small-scale Enterprises

Nabeel Ahmed
CSAIL, MIT
nabeel@csail.mit.edu

Srinivasan Keshav
University of Waterloo
keshav@cs.uwaterloo.ca

Konstantina Papagiannaki
Intel Labs, Pittsburgh
dina.papagiannaki@intel.com

ABSTRACT

We consider the problem of providing mobility support for Voice-over-IP (VoIP) traffic in small-scale enterprises. There is considerable interest in providing on-the-go support for VoIP through the use of WiFi-enabled smart phones. However, existing solutions either do not support client mobility or require client modifications, making them difficult to deploy in practice.

In this paper, we present OmniVoice, an 802.11 compliant solution that supports mobility for VoIP traffic *without any client modifications*. To effectively support such traffic, OmniVoice eliminates client handoff delays and manages interference from non-VoIP background traffic. It achieves this by using (a) a single-channel WLAN design and (b) a *lightweight* central controller for scheduling non-interfering AP-to-client transmissions and dynamically associating clients. The controller minimizes interference with the help of an interference map that models potential exposed and hidden terminal conflicts in the WLAN, while allowing for the serialization of transmissions that would otherwise compete for medium access. We have implemented and extensively evaluated OmniVoice on a 40 node wireless testbed. OmniVoice meets the QoS requirements for VoIP in all operating scenarios and is unaffected by interference from non-VoIP traffic. In particular, OmniVoice provides 100% greater throughput and 130% greater uninterrupted connectivity time to mobile VoIP users as compared to an off-the-shelf 802.11 solution.

Categories and Subject Descriptors

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

General Terms

Algorithms, Design, Experimentation, Measurement, Performance

Keywords

VoIP, Mobility, Interference, Enterprises

1. INTRODUCTION

We consider the problem of providing mobility support for Voice-over-IP (VoIP) traffic in small-scale enterprises. Small-scale enterprises are those that have 100 employees or less. Based on the 2002 US census[1], over 98% of U.S. businesses were small-scale. The lower cost of VoIP technology over traditional PBX telephone networks makes it an attractive solution for small-scale businesses [28]. VoIP works relatively well in stationary environments where a user connects through a VoIP-enabled IP phone. However, mobile scenarios in which a client walks around the enterprise while maintaining a VoIP session on a WiFi-enabled device are poorly supported. With the rapid growth of smart phones, this mode of communication is becoming increasingly popular [28]. Our aim in this paper is to extend existing enterprise wireless networks to support VoIP mobility where clients roam seamlessly from Access Point (AP) to AP without experiencing degradation in voice call quality.

Enterprise wireless networks (WLANs) consist of APs that are typically under control of a single administrative authority. To minimize interference, co-located APs are assigned different channels (or frequencies). Mobile clients must periodically scan for nearby APs in order to roam from one AP to another. Scanning introduces unpredictable delays (quantified in Section 4.2) and can result in call disruptions or worse, dropped calls. Enterprise WLANs have recently adopted a centralized design where a central control element (or controller) observes the network and centrally configures parameters for the APs (e.g., channels, TX powers, etc) [11, 22, 26]. In centralized WLANs, the controller can assist clients during a handoff by for instance identifying co-located APs to minimize scanning delay [21, 22]. However, this requires clients to communicate with the APs and exchange network information, which necessitates client modifications. Client modifications inhibit practical deployability as today's enterprise networks need to support a diverse set of WiFi devices (e.g., smart phones, tablet PCs, and laptops). Our work emphasizes practical deployability and thus *no client modifications* is a cornerstone of our WLAN design.

The three main challenges we address in our work are:

- *Hand-off Delays:* can significantly impact VoIP call quality. These delays vary depending on the environment, choice of parameters and hardware capabilities [20]. VoIP can typically accommodate delays of up to $200ms$ before the call quality becomes unacceptable low [5].
- *RF Interference:* Enterprise WLANs are known to suffer from hidden and exposed terminals [26]. This can result in severe packet loss for voice traffic [10] in both mobile as well as stationary conditions. Additionally, contention with Data

[1]http://www.census.gov/csd/susb/susb02.htm

MobiHoc'11, May 16–May 20, 2011, Paris, France.

traffic (e.g., due to queueing delays) can also cause packet losses if VoIP packets arrive too late at the receiver.

- *Bi-directionality of VoIP traffic:* Most of today's enterprise traffic (over 80%) is downlink in nature, i.e., from APs to clients [2]. Therefore, enterprise WLANs are typically optimized only for downlink traffic. However, VoIP traffic is bi-directional and thus requires that the uplink have sufficient air-time to accommodate uplink voice traffic.

The key contribution of this paper is a practical enterprise WLAN design (termed 'OmniVoice') for small-scale enterprise networks based on (1) A single channel WLAN design that eliminates hand-off delays and allows clients to retain their association parameters as they move from AP-to-AP, (2) A central scheduler (at the controller) for downlink traffic that co-schedules only non-interfering AP-client links, and (3) An uplink traffic management module that appropriately reserves air-time for uplink VoIP traffic. In our single channel WLAN design, all APs advertise all orthogonal channels (using multiple radios) and when a client associates on a particular channel, it remains on that channel for the duration of the connection. All these mechanisms put together allow OmniVoice to support VoIP for mobile clients in small-scale enterprises without client modifications.

We implement and extensively evaluate OmniVoice on a 40 node testbed. Our evaluation includes controlled micro-benchmarks to evaluate the incremental gain from each component of OmniVoice. We also perform large-scale experiments to study the end-to-end performance of OmniVoice. We find that OmniVoice provides 100% greater throughput and 130% greater uninterrupted connectivity time for VoIP users compared to an off-the-shelf WLAN solution.

To our knowledge, our's is the first research work to provide a practical solution for VoIP in mobile scenarios without client modifications. While the single channel design is not new [12], we are the first to demonstrate it in a practically deployed WLAN system and showcase its ability to eliminate handoff delays. In doing so, we also show how conventional multi-channel WLANs suffer significantly from hand-off delays and thus are ill-suited to support seamless roaming for mobile VoIP clients in enterprise WLANs.

2. PROBLEM DOMAIN

To put the scope of this work into perspective, we start by listing the assumptions under which we have developed OmniVoice.

- We assume that only APs are modifiable. APs are typically under single administrative control and thus their software (including firmware) can be easily programmed. On the other hand, client terminals are quite diverse (e.g., laptops, smart phones, tablets, etc) and not controllable.

- OmniVoice is designed to fulfill the scalability requirements of a small-scale business. Small-scale businesses consist of less than 100 employees and if we assume that half the population simultaneously engages in mobile VoIP communication (an unlikely scenario), OmniVoice must be able to support over 50 mobile VoIP users.

- OmniVoice must ensure that VoIP call quality does not degrade due to RF interference and/or mobility. In this paper, we do not consider fairness issues for non-VoIP traffic [30]. And while we do not observe unfairness for data traffic in our experiments, OmniVoice does provide tuning knobs to explicitly cater for such traffic as well.

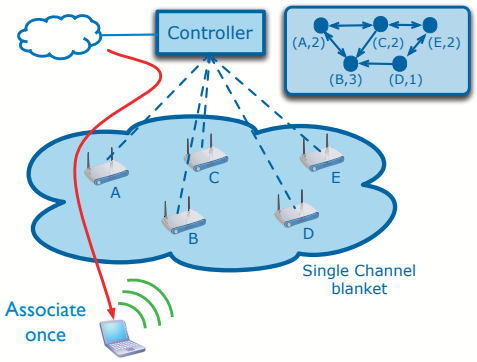


Figure 1: High-level view of the OmniVoice architecture. The client associates once through AP A and seamlessly roams from AP-to-AP thereafter.

- OmniVoice only considers the impact of internal traffic on VoIP performance (i.e., traffic from APs/Client in the same network). External traffic from co-located networks could also impact performance and a full featured solution would also consider such traffic.

3. ARCHITECTURE

Before describing the details of OmniVoice, we briefly discuss our design principles and preview how they will be used in the OmniVoice system.

Virtualization to Address Handoff Delays: Hand-off latencies, i.e., the time taken for a client to roam from one AP to another, can severely disrupt voice quality. OmniVoice effectively addresses such latencies by virtualizing the AP infrastructure so that clients have the illusion of being connected to the same AP (or WLAN segment) regardless of their location. This allows clients to retain their association parameters as they move from AP-to-AP without needing to scan for new APs. We discuss how to achieve this in Section 3.1.

Centralized Interference Management: RF interference from other co-channel transmitters in the WLAN can significantly degrade call quality for voice traffic. Prior work has shown that enterprise networks largely suffer from hidden and exposed terminals [26]. OmniVoice adopts a centralized design where it aggregates interference information in the form of an *interference map* and resolves interference by co-scheduling non-interfering links. Centralization (via a central controller) is commonplace in today's enterprise WLANs because of its superior security, access control, and network management properties. OmniVoice also uses a light-weight controller to effectively manage RF interference.

Managing Uplink Traffic from Client: OmniVoice's design mandates no client changes. Thus, while OmniVoice can control AP traffic on the downlink, it has no control on (uplink) client traffic. Clients may transmit bulk Data traffic and interfere with downlink voice traffic (AP $\rightarrow$ client). Similarly, uplink voice traffic may be interfered by downlink data traffic (AP $\leftarrow$ client). OmniVoice addresses both these problems by exploiting features already present in existing IEEE 802.11a/b/g/n standards, such as using CTS-to-self broadcasts to reserve the channel before transmitting VoIP packets. These techniques allow OmniVoice to effectively control uplink traffic without requiring explicit client control.

We next describe each aspect of the OmniVoice system and show how they work together to implement a mobile voice solution for small-scale enterprises.

3.1 Virtualized Single-channel Design

Layer 2 client handoffs reduce call quality if they last longer than a few hundred milliseconds[2] [5]. Unfortunately, it has been found that commodity WLAN client terminals incur a significant delay to handoff between APs and this delay depends on the environment, choice of parameters and hardware capabilities [20]. We perform measurements to study this hand-off delay. We use the popular Atheros EMP-8602 wireless card as the client and the Intel 2915ABG wireless card as the AP. Our measurements (discussed in more detail in Section 4.2) reveal two major sources of delay: (1) The client radio attempts to repeatedly re-connect to the currently-associated AP rather than immediately scan for a different AP, and (2) Clients take exceptionally long while scanning different channels, and this delay increases if Beacons are missed due to co-channel interference. Taken together, we find that these delays range anywhere from one second to as much as 50 seconds.

The above observations appear to indicate that we can trivially reduce scanning delay by more aggressively (or pro-actively) scanning for nearby APs. However, this consumes additional power and can reduce battery lifetime, which is especially problematic for handheld devices. Thus, with *client-initiated* schemes such as those described above, there exists a fundamental trade-off between energy and lower handoff delay.

A number of schemes have been proposed to eliminate or substantially reduce client delay [8, 22, 17]. These schemes require both changes at the AP and the client. Thus, we cannot use them because they require client modifications.

OmniVoice is designed to eliminate handoff delays without requiring any client modifications. OmniVoice's design is based on the observation that client hand-offs can be eliminated by creating the illusion of a single enterprise-wide WLAN segment where each client is connected to the same *virtual* AP regardless of its location [12]. Clients retain their AP association parameters as they move from AP to AP, effectively eliminating client initiated handoffs.

To achieve the functionality above, APs broadcast the same ESSID and MAC address in their beacons. Furthermore, each AP has multiple radios and each is tuned to an orthogonal channel (or frequency). A client associates to the network using one of the available channels and avoids switching channels for the duration of the connection. From the client's perspective, the entire network is thus reduced to a single virtual AP (as seen in Figure 1). Because clients no longer perform handoffs, this responsibility shifts to the APs in the network. A light-weight controller is used to coordinate APs to ensure that clients transition smoothly between APs as they move about in the enterprise.

Although this design is attractive for eliminating handoff delays, it incurs significant co-channel interference because now all APs use all channels. OmniVoice addresses this using a centralized interference management framework described in Section 3.2.

3.1.1 Client Association and AP Selection

In OmniVoice, a client connects to the WLAN in the usual way, i.e., by receiving a Beacon from one or more nearby APs and responding with an 802.11 Authentication Request. The Authentication Request may be received by multiple APs. All APs receiving the request forward it over the wire to the controller. The controller instructs the AP that observes the strongest signal strength from the client to associate with it. The selected AP completes the 802.11 association process with the client and sends all the client state (including WEP encryption keys) to the controller. The controller stores this client state for each client associated to the network. Once associated, the client also requests an IP address which it retains for the duration it is connected to the network.

Selecting the best AP: In conventional WLAN designs, AP selection involves choosing the best (AP, channel) pair for the client, where 'best' is defined by the AP selection algorithm [18]. OmniVoice simplifies this process by requiring that only the best AP be chosen for the client (keeping the channel constant). This preserves the original association parameters for the client (because all APs advertise the same MAC address as discussed in Section 3.1). Even with this simplification, finding the best AP for the client may be difficult because of the need to balance competing objectives such as AP workload, signal strength, duty cycle, etc. The research literature mentions numerous AP selection algorithms that take into account one or more of these objectives [27]. In this section, we are interested in answering the question, 'Does the choice of AP selection algorithm in OmniVoice have a significant impact on the performance of the mobile VoIP client?'

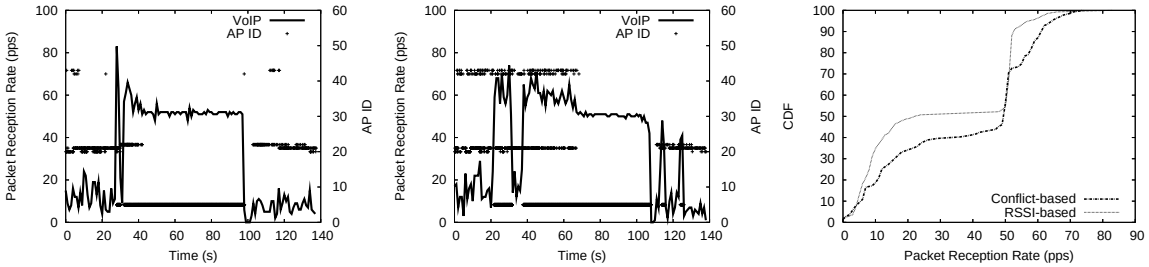
We consider three AP selection schemes to answer this question. Two of them are widely used in off-the-shelf WiFi cards, while the third uses an interference map to make AP selection decisions, taking into account interference between links in the WLAN. While this evaluation is by no means exhaustive, it provides some insight in choosing an AP selection scheme for OmniVoice. Note that the selection of AP for a client is done by the OmniVoice controller and not the client.

- *RSSI-based Selection:* The algorithm chooses the AP with the highest received signal strength (measured as RSSI) from the client. The algorithm maintains an exponentially weighted moving average of the RSSI values observed for each client. This metric is commonly used in WiFi clients to select the best AP. In OmniVoice, we use uplink RSSI as a predictor of throughput in both the uplink and downlink directions. Prior work [21] shows that uplink signal strength provides a good approximation of downlink signal strength.

- *DDR-based Selection:* This algorithm selects the AP yielding the highest downlink delivery ratio (DDR) to the client. This algorithm is likely to be more accurate because it measures the quality of the link by *actual* number of packets successfully delivered to the client, instead of using RSSI as a proxy for link quality. To measure DDR, all candidate APs transmit a series of probes to the client (in a serialized fashion) and determine whether the transmission was Acked. APs report back the resulting DDR to the controller which maintains an exponentially weighted moving average of the DDR values to the client. The controller then selects the AP with the highest DDR to the client. Probes last $\approx$ 15-20ms per AP, and therefore introduce a modest amount of overhead to the system.

- *Conflict-based Selection:* This algorithm uses interference information that is available in the interference map. It assesses client performance along two axes: (1) Link quality to the AP, and (2) Degree of inter-AP interference at the AP. Link quality is measured using RSSI values as discussed above. Once a set of 'good' links are chosen, the algorithm selects the AP that minimizes the sum total number of conflict edges from neighbouring APs, where the goal is to maximize the amount of airtime a client gets from the AP (i.e., it chooses an AP with the smallest duty cycle). This algorithm also requires load information for each AP to quantify the degree of interference for each conflict edge.

[2] Layer 3 mobility (or handoffs) are a superset of Layer 2 handoffs and thus do not mitigate their impact



(a) RSSI-based (b) Conflict-based (c) CDF of AP Selection Algorithms

Figure 2: Results on the AP selection algorithms that were evaluated for OmniVoice

Given that the clients are sending VoIP traffic with a packetization interval of *t*, we are able to measure most of the above metrics without injecting any additional traffic into the system. Since all APs can listen to all traffic on all channels, they passively collect statistics in the vicinity of a client. The only exception to the rule is DDR where we must compute the downlink delivery ratio from each AP to the client.

Our evaluation reveals similar performance for DDR as compared to the RSSI and Conflict-based schemes. Hence, for conciseness, we omit presenting results for DDR.

Results

To isolate the impact of AP selection, we disable the scheduler and only run the chosen AP selection algorithm. We add background interference to gauge the benefits of being interference-aware in the AP selection process. In our experiments, the VoIP client walks along the mobility shown in Figure 3. It starts at point A (with many APs) and moves in a straight line horizontally until it reaches the end of this straight path. It then moves back along the same straight to point A. VoIP traffic is sent at $50pps$ and we measure the goodput at the client.

Figures 2(a) and 2(b) show typical time series graphs of the goodput a client obtains for each of the AP selection algorithms. On both graphs, the client starts off with low throughput when it is in the neighbourhood of many APs. Its throughput improves as it moves away because it only encounters a single AP to whom its associated (on the other end of the straight path). However, it drops back down again as the client returns back to point A. Looking at these graphs, although the Conflict-based scheme performs slightly better than the RSSI-based scheme, the improvement is not significant. This is also seen in the CDF plot shown in Figure 2(c). We repeated this experiment along different movement trajectories in our building and observe similar results.

In short, we find that in OmniVoice there is no additional gain from conflict and load information during AP selection instead of a naive RSSI-based selection scheme. We believe this is because it is less likely that co-located APs on the same channel (that have a high signal strength to the client) observe markedly different channel utilizations. Because these APs transmit omni-directionally and at the same power, they all sense one another and hence back-off from each others' transmissions. This transmission coupling between APs would make either of them an equally good candidate in terms of the amount of air-time available to transmit a VoIP client's traffic (assuming VoIP is prioritized over non-VoIP traffic at all APs). APs farther away (i.e., those that are not coupled) are not suitable because their weaker signals would likely collide with stronger signals from the co-located APs (at the client). Hence, these observations coupled with the simplicity of the RSSI-based scheme made it our solution of choice for AP selection in OmniVoice[3].

3.2 Centralized Interference Management

OmniVoice provides a comprehensive framework for managing interference. We first discuss how OmniVoice achieves interference-free communication in the downlink (AP $\rightarrow$ client) and then present the uplink (AP $\leftarrow$ client) case in Section 3.3. For downlink traffic, OmniVoice provides two key functions (1) Mapping of interference between pairs of wireless links, and (2) Co-scheduling of non-interfering links.

3.2.1 Mapping of Interference

Mapping interference in an enterprise WLAN is necessary to identify links that potentially interfere with one another. An interference map (or conflict graph) is a data structure that encodes interference information between links in a wireless network. Conflict graph construction has received considerable attention in the research literature [7, 23].

In OmniVoice the interference map dynamically changes because of client mobility. Therefore, the interference map must be re-computed in realtime to capture such changes. Because in-building mobility is the result of human movement with typical walking speeds of $\approx$ 4kph, the conflict graph is likely to remain constant over periods of 10s of milliseconds. We use micro-probing [7] to measure downlink interference because of its ability to map interference in realtime at millisecond-level timescales. Micro-probing does not require client modifications and thus is also compatible with our design goals.

Micro-probing discovers interference between links by performing active interference tests. Each test consists of a pair of links transmitting packets simultaneously. Depending on the outcome of the test, we determine whether a link interfere's with another link. For instance, suppose we want to test whether link l_2 interferes with link l_1. Both l_1 and l_2 transmit packets simultaneously and if l_1's receiver replies with an ACK, l_2 does not interfere with l_1. To account for channel variations, each experiment is repeated multiple times (5 times in OmniVoice). Using this approach, we can measure interference between pairs of links in the network. It should be noted that clients are *not* modified to perform micro-probing. It is the clients' normal response to data reception that microprobing relies on. This is the key characteristic of the micro-probing approach to measuring interference, making it our solution of choice

[3]We assume VoIP clients are uniformly distributed and their traffic is load-balanced across all available channels. RSSI-based selection may not work in the rare event that many VoIP clients all assigned to the same channel congregate together at a particular location. In this case, a more sophisticated scheme would be required for assigning APs to clients

for mapping interference. Micro-probing is also able to perform these tests very quickly and can generate an interference map for a modest-sized network (e.g., 10 APs, 10 clients) in just a few seconds [7].

Minimizing Measurement Overhead: Although microprobing rapidly generates the interference map, it nonetheless incurs some measurement overhead that needs to be minimized. We take the following steps to ensure minimum overhead. When each client first associates to the network, interference tests are performed for that client's link with all it's neighbouring links. Neighbouring links are those AP-client links where the AP for the link does not contend with the newly added client's AP (APs that do contend don't need to be tested). These tests allow us to gather all the conflict information for the new AP-client link. When the client moves, OmniVoice must re-measure interference for it. But how do we decide when to perform this measurement?

There are two ways of answering this question. First, we could empirically estimate the mean rate of change of conflicts for a link and use that as our re-measurement interval. Second, we could estimate the overhead of computing conflicts for a link and clamp that to a particular percentage of the total available airtime. The former is harder to do because it is correlated with user mobility characteristics (i.e., walking speeds, changes in direction, etc) and thus has high variance. We therefore use the latter approach to estimate the re-measurement interval.

We do a back-of-the-envelope calculation of the measurement overhead. For simplicity, we deal only with mean values of random variables. As a rule of thumb, we maintain that conflict graph construction/re-measurement should consume no more than 5% of the total airtime in the network. Each pairwise interference test takes $\approx 2.5ms$ to complete (assuming a probe size of 100 bytes). Through empirical measurements, we observe that at any given time, a client is in the neighbourhood of no more than 5 APs. Computing conflicts for this client across all these APs requires approximately $12.5ms$ to complete. To ensure the interference map is up-to-date we must re-measure it at a rate faster than the transition periodicity of the client between APs. Through empirical measurements, we determine that the mean transition time of a mobile user between APs is $34s$ (excluding cases where the client oscillates between APs). Using the Nyquist criterion, we choose to sample at twice the client's transition rate (i.e., 17s). Given that our measurement overhead is clamped at 5% of the total air time, this gives us approximately $850ms$ to perform measurements. Given that computing conflicts for a single client takes $12.5ms$, OmniVoice is able to support up to 68 mobile clients per measurement interval, which is more than sufficient for a small-scale enterprise deployment. Note that this assumes client tests are serialized. This is not necessary for clients that are geographically separated in the enterprise. Thus, in practice, OmniVoice can scale to even more clients.

3.2.2 AP Co-scheduling

We now describe our approach to managing inter-AP interference in the OmniVoice system.

In a single channel WLAN, neighbouring APs are likely sources of interference. Given that we have measured the interference map for the network, we design a scheduler (co-located with the controller) to coordinate downlink transmissions at the APs. The input to the scheduler is the interference map of the network and the workload of each AP. APs periodically report their queue sizes (or workload) to the scheduler and APs with zero workload are ignored by the scheduler.

Our scheduling mechanism divides time into equal sized slots and schedules APs such that no two conflicting APs (that interfere due to inter-AP or AP-client conflict) are scheduled in the same slot. We implement a greedy scheduler that schedules as many APs as possible in a given time slot. Once a schedule is constructed, the controller executes it as follows. For each slot, it sends a wired broadcast frame containing the identifiers of all APs assigned to that slot. It also adds the slot length (in ms) to the packet. The broadcast synchronizes APs to the current scheduling slot. Upon receiving the broadcast frame, an AP determines whether or not it is scheduled for the current slot (by searching for it's ID in the frame). If it is scheduled, it starts sending queued up packets and continues doing so until the slot duration expires. For each packet that is to be transmitted, the AP estimates its transmission time by using the packet size and the data rate to be used for transmission (the data rate is supplied by the currently running rate adaptation algorithm). If the current time plus the packet's transmission time exceeds the expiry time of the current slot, the packet is not transmitted and held for the next slot. APs that are not scheduled for the current slot block and wait for the next broadcast frame from the scheduler. A back-of-the-envelope calculation reveals that in the worst case, a broadcast frame consumes approximately 0.32% of the total available transmission time on the wired backbone, when the scheduling interval is as small as $2ms$ (the minimum time required to transmit an Ethernet frame of size 1500 bytes at the lowest supported data rate for IEEE 802.11a/g/n). Therefore, the overhead of broadcast frames is effectively negligible.

Note that APs serve both realtime and non-realtime clients. To avoid VoIP traffic from suffering large queueing delays or losses due to kernel buffer overflows, APs prioritize VoIP traffic by dedicating a separate output queue for such traffic. This is similar in spirit to the IEEE 802.11e standard [3]. However, we opt not to use 802.11e for reasons that we discuss in greater detail in Section 5.

Traffic loads in the network are subject to change, as are interference patterns. Therefore, a new schedule needs to be periodically re-computed by the scheduler. How often this is done depends on the network dynamics. In practice, we find that the overhead of recomputing the schedule is negligible and we therefore re-compute it each time the previously generated schedule has completed.

3.3 Uplink Traffic Management

We now discuss how OmniVoice accommodates uplink VoIP traffic. To do so, the co-scheduler (from the previous section) allocates some slots as 'uplink' slots. During an uplink slot, only clients contend for the medium using standard DCF and the APs refrain from transmitting packets.

OmniVoice exerts precise control on the starting time of uplink slots using unsolicited CTS-to-self broadcasts. Specifically, all APs transmit a CTS-to-self (with a NAV value equal to the duration of the downlink slot) prior to initiating their downlink transmissions. On receiving the CTS-to-self, clients freeze their back-off timers and do not contend until the end of the downlink slot. Thus, they are able to send packets only in the designated uplink slot periods. Some prior work [7, 16] has experimentally studied the efficacy of CTS-to-self in effectively silencing the medium in real world networks. This work shows that in almost all cases, CTS-to-self effectively silences the medium for the period specified in the CTS-to-self frame. We therefore opt to use this mechanism to control transmissions on the uplink from the client to the AP.

The controller needs to estimate the number of uplink slots needed to carry all uplink traffic. Unfortunately, the controller does not have access to the queue sizes at each of the clients. However, the controller is aware of the number of active VoIP sessions. In OmniVoice, our goal is to ensure that all these VoIP sessions have sufficient uplink airtime to transmit their packets (we also add some

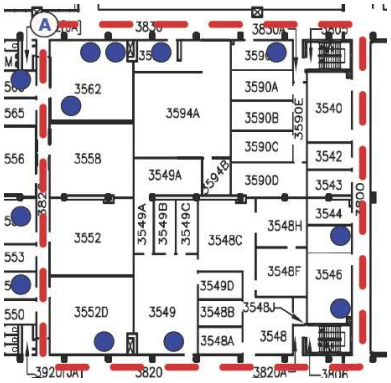


Figure 3: Mobility path followed by the client in our experiments. The path starts at point A, follows the dashed line, and returns back to point A. The blue circles represent APs along the path.

headroom for bulk Data traffic). The controller keeps track of the amount of VoIP traffic seen on the uplink. If it is less than the expected amount of VoIP traffic (based on the number of VoIP sessions), the controller increases the number of uplink slots using a multiplicative-increase additive-decrease (MIAD) scheme. We use MIAD in order to quickly converge to the correct number of slots. As long as uplink traffic forms a small fraction of the overall wireless traffic, a fact borne out by multiple studies of real networks [2], this scheme remains stable. However, as the amount of non-VoIP uplink traffic increases, its effect on VoIP traffic on the uplink is also likely to increase. This is because unlike downlink VoIP packets, uplink VoIP packets aren't given higher priority as compared to bulk Data traffic. In these rare situations, clients would need to implement VoIP prioritization mechanisms similar to those implemented at OmniVoice's APs.

4. EXPERIMENTAL EVALUATION

In this section, we first present micro-benchmark results to demonstrate the incremental gains from each component of OmniVoice. We then present results from a large-scale evaluation of the system across a variety of different scenarios.

4.1 Methodology

We evaluate OmniVoice on a 40 node wireless testbed. The testbed nodes act as APs, each equipped with an Intel 2915 ABG wireless card. We modified the ipw-2200 driver to implement the features described in Section 3. We use Dell Vostro 1400 laptops to act as clients, equipped with an EMP 8602 (Atheros) card and using the MADWiFi 0.9.4 driver.

Clients associate with the wireless network and move along the path shown in Figure 3. We collect wireless traces at the client and plot the packets received per second (pps) during an experiment. In our experiments, we generate VoIP traffic using UDP streams to mimic the popular G.729 VoIP codec: the packet inter-arrival time is 20 ms and packet size is 20 bytes (resulting in 50pps under ideal conditions).

Interference sources generate backlogged UDP traffic with a packet size of 1400 bytes (to mimic Ethernet packets). This represents the worst case for OmniVoice and therefore our results serve as a lower bound on OmniVoice's performance. Our primary performance metrics are packet reception (in pps) and delay jitter (in ms).

All our experiments are conducted on the 5.8 GHz (IEEE 802.11a) band to avoid interference from the campus network as well as other networks. We use the default rate adaptation algorithm implemented in the Intel 2915ABG cards. Our experiments are repeated five times and we show 95% confidence intervals for all our results.

Scheme	Mean Delivery Rate (%)	95% CI (%)
No Interference	49.31	0.39
Interference	31.34	0.92
Uplink Scheduler	44.34	0.23

Table 1: The mean uplink delivery rate without interference, with interference, and with both interference and uplink scheduling

4.2 Micro-Benchmarks

We begin by studying the behavior of VoIP clients in a standard multi-channel network. We then evaluate the gain from introducing the following three components in the context of a single-channel design: 1) VoIP traffic prioritization at the APs, 2) Co-scheduling of the APs, and 3) Uplink traffic management.

4.2.1 VoIP Performance in a Multi-Channel System

To study VoIP performance in a typical multi-channel system, we assign APs in our testbed to orthogonal channels. A VoIP client walks along the rectangular path shown in Figure 3 (Figure 4(a) presents the result). Observe that the client initially gets a goodput of 50 pps but this quickly falls as it moves away from the AP. The goodput also falls to zero during the experiment, during which the client is disconnected from the network. Once it re-establishes connectivity, this process repeats. By analyzing the wireless traces for this experiment, we find that the primary cause of these prolonged disconnections is that the client attempts to repeatedly re-connect with the same AP with which it lost connectivity. After failing over multiple attempts, it drops down to scanning for neighbouring APs. The re-connection and scanning delay taken together make up for the unusually high disconnection delay observed in Figure 4(a).

4.2.2 Single Channel Design with Traffic Prioritization and Background Traffic

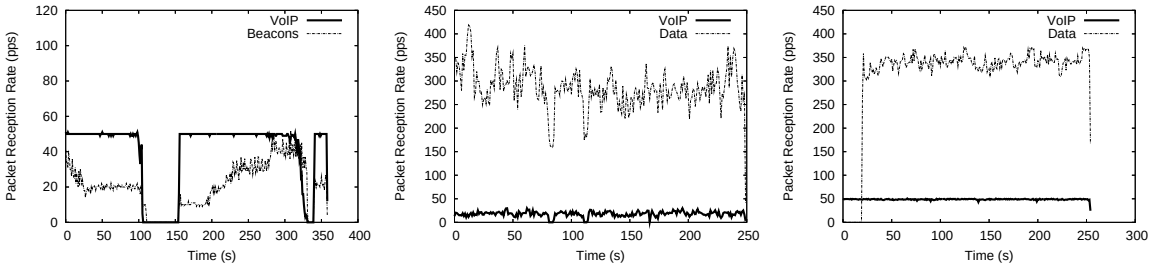
We now study the gain of each component in OmniVoice. We start with a single-channel design and introduce saturated UDP background traffic on each AP downlink [4]. Without any of OmniVoice's enhancements, the single channel design performs very poorly in the presence of RF interference and suffers 100% packet loss rate. To alleviate this, we introduce VoIP prioritization on the downlink (AP $\rightarrow$ Client). Figure 4(b) shows the result of this experiment. Due to single channel virtualization, the client no longer experiences disconnections as it moves between APs. However, despite VoIP prioritization, the client still suffers interference from co-located APs and thus fails to achieve the target reception rate of 50pps.

4.2.3 The Gain from Co-scheduling

Figure 4(c) shows the result of enabling VoIP prioritization and co-scheduling. The client now achieves $50pps$ on the downlink regardless of its location. We repeat similar experiments for other mobility paths as well and observe statistically similar performance. Thus, we find that VoIP prioritization coupled with AP co-scheduling are sufficient to adequately support voice traffic on the downlink from the AP to the client. Next, we look at VoIP traffic on the uplink.

4.2.4 The Need for Uplink Scheduling

[4]Recall that single channel architectures are not restricted to one channel but use all orthogonal channels



(a) Multi-Channel (b) Single-Channel w/ Prioritization (c) Single-Channel w/ Prioritization & Co-scheduling

Figure 4: Results indicating the incremental gain from each of OmniVoice's components

We now evaluate the effectiveness of uplink scheduling in OmniVoice. We use the mobility path shown in Figure 3 and evaluate three cases: 1) Bi-directional VoIP traffic without downlink data traffic, 2) Bi-directional VoIP traffic with downlink interference but without uplink scheduling (marked 'Interference'), and 3) Bi-directional VoIP traffic with downlink interference and with uplink scheduling. Table 1 presents the mean (and 95% confidence intervals) of the uplink pps in all three cases. As expected, the client attains the full rate in the absence of downlink interference. With saturated downlink interference, the uplink pps drops to approximately $31pps$. With uplink scheduling (as described in Section 3.3), the performance improves to ≈45 pps. In this experiment, because there is only one VoIP client connected to the network, uplink traffic is allocated a slot of $1ms$. When one or more clients join or leave the network, the MIAD scheme discussed in Section 3.3 is used to adjust the number of uplink slots in order to reflect the traffic demand for VoIP on the uplink.

4.3 Large-scale Experiments

We now present large-scale experimental results to evaluate OmniVoice's ability to provide VoIP call quality to mobile clients under a variety of different operating conditions.

We compare the performance of OmniVoice against two other schemes. The first is a conventional multi-channel scheme (termed *M-channel*) commonly used in today's enterprise networks. In M-channel, we hand-tune three orthogonal frequencies across all the APs encountered along the mobility path to maximize frequency re-use. The second scheme (termed *No-Scheduler*) is identical to OmniVoice except that it does not incorporate co-scheduling and uplink traffic management. Therefore, it mimics IEEE 802.11e for downlink traffic.

4.3.1 End-to-end Performance Results

We begin by comparing the performance of OmniVoice with the other schemes over a single mobility run. In this experiment, a client walks along a mobility path (Figure 3) and encounters interference from APs sending downlink data traffic. A representative run for each scheme is shown in Figures 5(a), 5(b), and 5(c). The right-side Y-axis shows the ID of the AP with which the client is associated while the left-side axis shows the packet reception rate.

We observe (as seen in Section 4.2.1) that M-channel suffers frequent disconnections as the client attempts to maintain connectivity to the AP with which it is associated. In contrast, No-Scheduler and OmniVoice switch multiple times even over short periods of 10 seconds.

When encountering interference, No-Scheduler (Figure 5(b)) performs the worst and its performance only improves when co-located interference drops. In interference-free zones, No-scheduler attains the full $50pps$ rate as expected. Note that in some cases the client retains its full rate (between intervals $50-100$ and $175-200$) despite background interference (as shown by the green line). In these intervals, the green line indicates data traffic from the same AP (i.e., intra-AP contention) as opposed to traffic from co-located APs. Because No-scheduler implements VoIP prioritization, it is resilient to intra-AP contention.

Finally, OmniVoice consistently sustains $\approx 50pps$ throughout the run, indicating its resilience to hand-off latency and intra-AP/ inter-AP interference. Notably, it performs an almost equivalent number of AP switches as No-Scheduler, because both use the same AP selection algorithm, i.e., strongest RSSI.

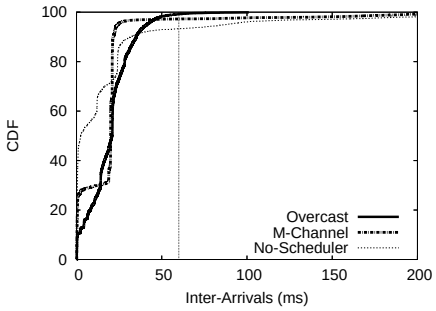


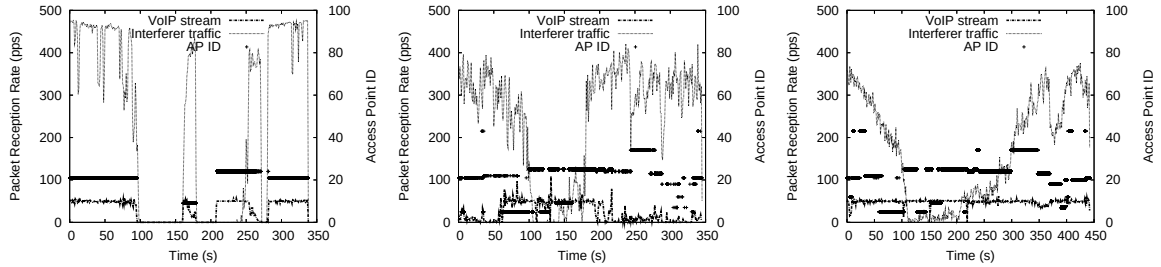
Figure 6: Cumulative distribution function of inter-arrival times shows that all packets arrive within the 60ms time specified for the jitter buffer size of the G.729 codec. M-channel performance is shown for non-disconnected intervals

4.4 VoIP Call Performance

In this section, we study the performance of VoIP calls using specific metrics that provide a deeper insight into VoIP call quality. These include delay jitter and total connectivity time during a mobility run.

4.4.1 Delay Jitter

The delay jitter metric quantifies the delay variance between when a VoIP packet is transmitted by the sender and when it arrives at the receiver. If the delay jitter is too high, VoIP clients suffer delay-induced losses. The amount of delay jitter that VoIP can tolerate depends on the VoIP codec used and the corresponding jitter buffer length (in ms). We use the popular G.729 codec, and thus assume a jitter buffer length of 60 ms [14]. Thus, we measure the span of the inter-arrival time distribution of VoIP packets at the client. If



(a) M-Channel (b) No-Scheduler (c) OmniVoice

Figure 5: Client packet reception rate for a representative run using each of the three schemes

Scheme	Total Connectivity Time (%)	95% CI
No-Scheduler	27.5	7.61
M-Channel	40	6.49
OmniVoice	94.25	0

Table 2: Tot. connectivity time with MOS greater than 2

Scheme	Number of Call Disruptions / Run	95% CI
No-Scheduler	10.75	1.19
M-Channel	4.25	0.39
OmniVoice	4.6	0.71

Table 3: Number of call disruptions per run for the different schemes

the span is less than the jitter buffer length, delay-induced losses are negligible.

Figure 6 plots the CDF of the inter-arrival times of VoIP packets for the three schemes. This result corresponds to the mobility runs performed in the previous section. Note that we only show inter-arrival times for consecutive packets in the trace (identified by their sequence numbers). We omit packets not received during periods of disconnection in the M-Channel case. Thus, while the CDF for M-channel appears to indicate good performance, it does not capture what happens when the client disconnects from the network. Thus, in reality, M-channel would perform even worse in terms of delayed induced losses.

We draw a vertical line at the point corresponding to $60ms$ for the inter-arrival time. For OmniVoice almost all packets arrive within $60ms$ of each other (with a span of $\approx$ $60ms$). In fact, $\approx 75\%$ of the packets arrive within the VoIP packetization interval of $20ms$. We repeat the experiment with different configurations of interferers and mobility paths and obtain similar results. Therefore, we conclude that with OmniVoice delay-induced losses are almost negligible.

The overall performance of M-channel and No-Scheduler is surprisingly similar to OmniVoice. In fact, No-Scheduler performs slightly better than OmniVoice. This improvement is attributed to the absence of the co-scheduler in the No-Scheduler scheme. In OmniVoice, the scheduler introduces some delay to separate conflicting APs transmissions. This increases delay between packets and affects packet inter-arrival time. However, for both M-Channel and No-Scheduler, the span of inter-arrival times has a heavy tail (up to $200ms$). Thus, these schemes suffer significantly from delay induced losses that degrade VoIP performance.

4.4.2 Session Characteristics

The longer a VoIP client maintains connectivity with the network and obtains good service the better. Service quality depends on the amount of losses a VoIP client can tolerate which in turn depends on the VoIP codec used. For the G.729 codec we use, VoIP can tolerate up to 10% losses before the call quality becomes unacceptably low [9]. A popular metric for evaluating a voice call is the Mean Opinion Score (MoS), which ranges from 1 to 5. A value of 5 implies perfect call quality and a value of 1 implies the inability to communicate. Losses of up to 10% correspond to a MoS value of 2.

We consider two metrics to gain a deeper understanding of VoIP call quality across a mobility run: *Total Connectivity Time* and *Number of Call Disruptions*. Total connectivity time of a VoIP session is defined as the time the client is able to sustain acceptable quality of service while connected to the network (i.e., its MOS value is above 2). This is measured as a percentage of the total VoIP session time. We consider different MOS threshold values, and for higher values, the performance gap between OmniVoice and the other schemes is even wider. The number of call disruptions captures the degree of disruptions experienced by a mobile client during a VoIP session. A disruption occurs if the MOS value falls below 2 for a period of at least three seconds (which is roughly the amount of time it takes to utter a short English sentence). This metric was also used in prior work that studied VoIP performance in vehicular environments [9].

Table 2 shows the result for the total connectivity time while walking along the same mobility path used in prior experiments. Again, No-Scheduler performs the worst of all the schemes. M-Channel improves total connectivity time (over No-Scheduler) by almost 40%. In contrast, OmniVoice yields the greatest total connectivity time, up to 130% greater than M-Channel. We repeat this experiment for a variety of mobility paths and locations and find that the mean gain from using OmniVoice is similar. Hence, we conclude that *OmniVoice more than doubles a client's uninterrupted connectivity as compared to off-the-shelf multi-channel WLAN solutions*.

Table 3 presents results for the mean number of call disruptions during the mobility run. No-Scheduler experiences the greatest number of disruptions, which are approximately 150% higher than the other two schemes. On the other hand, M-Channel and OmniVoice's performance is comparable. It is noteworthy however that disruptions in M-channel typically imply loss of connectivity for the client and hence incur long delays due to scanning and re-association. In OmniVoice, a client resumes its transmissions as soon as it exits the problem area (which may be a pillar causing RF shadowing) and is in range of any of OmniVoice's APs.

4.4.3 Impact of Interference

In this section, we seek to understand the relationship between the amount of inter-AP interference present in the neighbourhood of a client and its effect on VoIP call quality. To isolate the impact

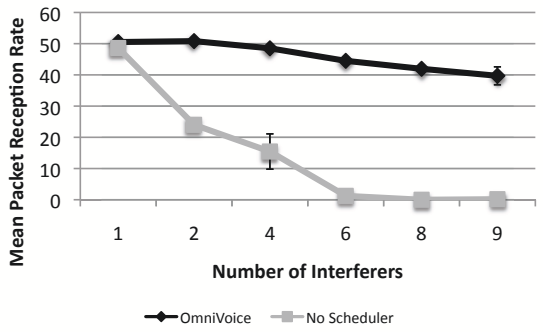


Figure 7: Interference has little effect on the performance of the VoIP client using OmniVoice. However, the client suffers heavily under the No-Scheduler approach which does not exploit information present in the interference map

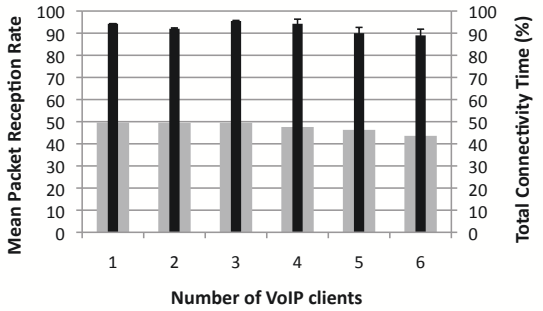


Figure 8: Mean packet reception rate (PRR) and total connectivity time for different numbers of VoIP clients. Blue bars show mean PRR and red bars show mean total connectivity time

of interference, we perform experiments with a static client. Furthermore, to eliminate location-induced biases, we place the client at multiple different locations and repeat the experiment. We find our results to be statistically similar for different locations and we therefore present results for only one location. Interferers generate saturated backlogged data traffic and maximize interference to the VoIP client. Thus, we study the worst case performance of OmniVoice.

Figure 7 shows the mean packet reception rate for a VoIP client as the number of interferers is increased. We do not plot results for M-Channel, since they are similar to No-Scheduler but scaled up based on the number of orthogonal channels. No-Scheduler's performance drops to almost half as the number of interferers increases to 2 and gradually falls to zero, as the number of interferers increases to 6. However, OmniVoice provides the full rate for VoIP for up to 4 interferers, and falls only slightly as the number of interferers goes up to 9. This demonstrates the resilience of OmniVoice's traffic scheduling framework. By precisely mapping interference and using a scheduler to co-schedule only non-interfering links, *OmniVoice avoids wasted airtime in retransmissions (due to collisions) and as a result can support more VoIP sessions as compared to the other schemes*.

4.4.4 Increasing the Number of Clients

An important aspect of OmniVoice's evaluation is how many mobile clients it is able to support. To study this, we simultaneously move multiple clients along the same path during an experimental run. Moving clients along separate paths does not stress test OmniVoice because VoIP traffic potentially gets distributed across APs in different contention domains (i.e., the APs are geographically separated such that they don't contend for wireless access). Again, we study the worst-case effect of running multiple VoIP clients. For these experiments, we used a variety of different hardware and software platforms for the clients, ranging from laptops running Linux, Windows XP and Vista, to iPhones running the OS X iPhone operating system. Thus, our results are not an artifact of any particular platform. Note that you can only do this because clients are unmodified.

Figure 8 plots the mean packet reception rate (the blue bars) for different numbers of VoIP clients (we compute the mean across 5 runs). OmniVoice provides good mean packet reception rate (PRR) for increasing numbers of clients. However, because mean PRR is an aggregate statistic, in Figure 8, we also plot the total connectivity time of VoIP clients (the red bars), for increasing numbers of clients. We observe that the per-client connectivity time is approximately the same for different numbers of clients. Nevertheless, the trends in these plots indicate a decline in performance for increasing numbers of clients. However, these results correspond to VoIP clients that are within a common contention domain. In practical deployments, this will likely not be the case and VoIP clients will be distributed across the entire enterprise. Thus there will be clusters of VoIP clients that can be scheduled in parallel. Even if we conservatively estimate there to be three such clusters across the enterprise (our measurements reveal that the commercial WLAN deployment in our building contains ≈ 7 clusters), OmniVoice will be able to support at least 18 clients across all clusters. Moreover, if we you use multiple orthogonal channels, we can support even more clients. For three orthogonal channels, we can support up to 54 clients, which is more than sufficient for a small-scale enterprise with less than 100 employees.

5. RELATED WORK

Hand-offs in WLANs: Eliminating hand-off latencies in WLAN networks has been the focus of much prior work [19, 25, 24, 29, 18]. Shin et. al. [25] propose the use of neighbour graphs to reduce client scanning time. Ramani et al. [24] propose synchronizing beacon transmissions across co-located APs to reduce overall scanning time. Mhatre et al. [18] consider a variety of client-initiated hand-off algorithms to reduce latency of roaming. However, all these approaches advocate client modifications and thus are not feasible for small-scale enterprises.

VoIP Traffic Management: There has also been a lot of work on studying the performance of VoIP over 802.11 networks (Voice-over-WLANs) [15, 10, 30]. Conventional wisdom states that the popular 802.11a/b/g standards poorly support VoIP traffic [15]. To support such multimedia traffic, the IEEE 802.11e standard [3] has been proposed which extends prior standards to enable QoS support for realtime applications. 802.11e proposes two channel access schemes: Enhanced Distributed Channel Access (EDCA) and HCF Controlled Channel Access (HCCA). EDCA supports prioritized channel access whereas the HCCA access scheme can be parameterized and is similar in spirit to PCF that was proposed for the 802.11a/b/g standards. A recent experimental study found [13] that while many WiFi devices do not implement the 802.11e standard, those that do (i.e., WMM-certified devices) only support EDCA

and not HCCA (which is optional in 802.11e). This study also found that many of these WMM-certified devices did not work correctly in the presence of legacy 802.11a/b/g devices. Because of the poor support of 802.11e in today's WiFi devices and the fact that we wanted to be backwards compatible, we opted not to use 802.11e in OmniVoice's design. Having said that, the research community has also explored ways to improve VoIP traffic support on networks that use purely 802.11a/b/g standards. An approach, termed SoftSpeak [30], proposes a distributed TDMA protocol to support VoIP clients. The goal is to improve both the number of simultaneous VoIP sessions as well as minimize their impact on data traffic. However, Softspeak, aside from requiring client changes, does not allow support enterprise-wide mobility.

WLAN Management Designs: Like OmniVoice other WLAN architectures have also been proposed for managing (and possibly supporting VoIP) traffic [8, 21, 17, 22]. SMesh [8] proposes a solution where each AP advertises a common gateway IP address and BSSID, and avoids DHCP overheads during handoff. SMesh operates at Layer 3 and uses the default Layer 2 handoff process, making it susceptible to the unpredictable hand-off delays discussed in Section 3.1. DenseAP [21], uses a multi-channel design without client changes (similar to M-Channel) and thus also suffers from similar hand-off delays. Dyson [22] was recently proposed as a clean-slate design for enterprise WLANs with the potential to support realtime applications such voice. A key tenet of Dyson's design, however, is the ability for clients to report network measurements to the APs which makes it infeasible for small-scale enterprise deployments.

The work that most closely relates to OmniVoice is CENTAUR [26] which proposes centralized management of data traffic in a WLAN. Like OmniVoice it uses interference maps to coschedule non-interfering traffic on the downlink. However, CENTAUR differs from OmniVoice in a number of ways. It does not support mobility, does not address changes to the interference map, and also assumes that AP-client associations remain fixed. CENTAUR also only supports downlink traffic whereas OmniVoice addresses both uplink as well as downlink VoIP traffic.

Commercial Networks: Some commercial vendors (e.g., Meru [4], Extricom [6]) claim to support realtime traffic for mobile clients without requiring client modifications. However, these solutions are pricey, with an entry-level system costing around $50,000 [1]. Moreover, the proprietary nature of these systems makes it difficult to present a detailed comparison against OmniVoice. Nevertheless, private communications with some of these WLAN vendors reveals that while they may bear a few similarities with OmniVoice's design, there are some key differences as well, such as the approach they use to map interference. OmniVoice uses micro-probing to proactively measure and track interference for mobile clients. This is crucial for delay sensitive VoIP traffic because we want to avoid situations where the link quality drops to unacceptable levels before we flag the link as being interfered. To the best of our knowledge, we are the first to use such a proactive mechanism to mitigate interference for mobile VoIP clients in the enterprise.

6. CONCLUSIONS

The mobile work-force in small-scale enterprises is unable to benefit from the low cost and wide-spread availability of Voice-over-WLAN technology. In this paper, we design a practical WLAN system termed OmniVoice, that supports seamless mobility for VoIP users without requiring client modifications. Unlike conventional WLANs, OmniVoice uses a single-channel design and eliminates interference by co-scheduling non-interfering links. Moreover, aside from handling downlink VoIP traffic, OmniVoice also provides uplink VoIP traffic support. Through an extensive evaluation on a 40 node WLAN testbed, we find that OmniVoice dramatically improves performance over today's multi-channel networks. Furthermore, we believe that because OmniVoice insists on no client modifications, it can be easily deployed in existing WLANs and thus is an approach whose benefits can be immediately realized in today's small-scale enterprises.

7. REFERENCES

[1] A Total Cost of Ownership Analysis of the Meru Networks Virtual Cell Wireless LAN Architecture. http://www.merunetworks.com/technology/TCO_WP-20080326.pdf.

[2] Aruba Networks: Advanced RF Management for Wireless Grids. http://www.arubanetworks.com/pdf/rf-for-grids.pdf.

[3] MAC Enhancements for Quality of Service. http://grouper.ieee.org/groups/802/11/Reports/tge_update.htm.

[4] Meru Networks. http://www.merunetworks.com.

[5] VoIPTroubleShooter - Online Diagnostic Tools for Network Managers. http://www.voiptroubleshooter.com/problems/delay.html.

[6] Extricom Inc., Wireless LAN Switch DataSheet, 2005. http://www.extricom.com/imgs/Uploads/PDF/SWDataSheet.pdf.

[7] N. Ahmed, U. Ismail, S. Keshav, and K. Papagiannaki. Online Estimation of RF Interference. In *ACM CoNEXT*, 2008.

[8] Y. Amir, C. Danilov, M. Hilsdale, R. Musăloiu-Elefteri, and N. Rivera. Fast handoff for seamless wireless mesh networks. In *ACM MobiSys*, 2006.

[9] A. Balasubramanian, R. Mahajan, A. Venkataramani, B. N. Levine, and J. Zahorjan. Interactive wifi connectivity for moving vehicles. In *ACM Sigcomm*, 2008.

[10] R. O. Baldwin, I. Nathaniel J. Davis, S. F. Midkiff, and R. A. Raines. Packetized voice transmission using RT-MAC, a wireless real-time medium access control protocol. *ACM Mobile Computing and Communications Review*, 2001.

[11] 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.

[12] V. Bharghavan. Performance evaluation of algorithms for wireless medium access. *Computer Performance and Dependability*, 1998.

[13] R. Bolla, R. Rapuzzi, and M. Repetto. On the effectiveness of ieee 802.11e implementations in real hardware. In *ISWCS*, pages 303 –307, 2009.

[14] R. G. Cole and J. H. Rosenbluth. Voice over ip performance monitoring. *SIGCOMM Comput. Commun. Rev.*, 31(2):9–24, 2001.

[15] S. Garg and M. Kappes. An experimental study of throughput for UDP and VoIP traffic in IEEE 802.11b networks. In *WCNC*, 2003.

[16] U. Ismail. Virtual PCF: Improving VoIP over WLAN performance with legacy clients, Master's Thesis, University of Waterloo (UW). 2009.

[17] A. Kashyap, S. Ganguly, S. Das, and S. Banerjee. Voip on wireless meshes: Models, algorithms and evaluation. In *IEEE Infocom*, 2007.

[18] V. Mhatre and K. Papagiannaki. Using smart triggers for improved user performance in 802.11 wireless networks. In *ACM MobiSys*, 2006.

[19] A. Mishra, M. ho Shin, and W. A. Arbaugh. Context caching

using neighbor graphs for fast handoffs in a wireless network. In *IEEE Infocom*, 2004.

[20] D. Murray, T. Koziniec, and M. Dixon. An analysis of handoff in multi-band 802.11 networks. In *MASS*, pages 1 –10, 2007.

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

[22] R. Murty, J. Padhye, A. Wolman, and M. Welsh. Dyson: An Architecture for Extensible Wireless LANs. In *USENIX*, 2010.

[23] J. Padhye, S. Agarwal, V. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of Link Interference in Static Multi-hop Wireless Networks. In *IMC*, 2005.

[24] I. Ramani and S. Savage. SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks. In *IEEE Infocom*, 2005.

[25] M. Shin, A. Mishra, and W. A. Arbaugh. Improving the Latency of 802.11 hand-offs using Neighbor Graphs. In *ACM MobiSys*, 2004.

[26] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra. CENTAUR: Realizing the Full Potential of Centralized WLANs through a Hybrid Data Path. In *ACM MobiCom*, 2009.

[27] K. Sundaresan and K. Papagiannaki. The need for cross-layer information in access point selection algorithms. In *ACM IMC*, 2006.

[28] P. Thornycroft. The all-wireless workplace is now open for business: Using 802.11n as your primary network LAN deployments, Technical Report, Aruba Networks. 2007.

[29] H. Velayos and G. Karlsson. Techniques to reduce the ieee 802.11b handoff time. In *ICC*, 2004.

[30] P. Verkaik, Y. Agarwal, R. Gupta, and A. Snoeren. SoftSpeak: Making VoIP Play Fair in Existing 802.11 Deployments. In *NSDI*, 2009.