­

White Paper

Smartphone VoIP performance on enterprise WLANs – iPhone4 and Galaxy Nexus

Peter Thornycroft

Rev 2.0 June 2012

## Introduction

This paper examines how smartphones perform on enterprise WLANs, with special attention to the performance of voice over Wi-Fi and video applications (broadly multimedia over Wi-Fi) that require continuous, uninterrupted media streams as they move around the building or campus. Multimedia traffic is more demanding than serving Web pages or other data traffic: multimedia frames must be delivered on-time, without interruptions or errors. But application developers have found that a simple implementation of a VoIP (voice over IP) client is sufficient for the home user on a single, isolated Wi-Fi access point.

As we move into an enterprise WLAN, however, the RF environment becomes more complex. First, there is more data ‘on the air’ as many users share the WLAN. This means quality of service (QoS) priority levels must be invoked. Second, a smartphone on-call while moving through a building will switch its AP association quite frequently, so the implementation of inter-AP handover must be very robust. Third, business users on a WLAN are likely to participate in conference calls and other long-duration calls, and to be significant consumers of data services such as email on their smartphones. Thus it is important that protocols to allow simultaneous voice-and-data traffic, and also to extend battery life are implemented correctly. Of these separate issues, we focus here mostly on inter-AP handover, as it is the most complex and least understood aspect of Multimedia over Wi-Fi, as well as the most noticeable if performance is lacking.

The underlying theme of this paper is that existing, industry-wide open standards can achieve superior multimedia over Wi-Fi performance. An earlier generation of smartphone vendors already developed robust implementations, as a result of earlier multimedia over Wi-Fi experience, but the current crop of smartphone vendors are still in the early stages. By ensuring that they follow the standards, and considering multi-AP WLANs in their use-case models, smartphone silicon, operating system and applications developers can improve and optimize the multimedia over Wi-Fi experience.

This is our second report on the state of smartphone performance on enterprise WLANs, a follow-up on our report of October 2010. The 18-month period has seen incremental improvements but we have no significant new additions. This will change with the Wi-Fi Alliance ‘Voice-Enterprise’ certification which introduces a number of features to improve inter-AP handover. Most enterprise WLANs can be upgraded to support Voice-Enterprise, and we expect to see Voice-Enterprise compliant clients on the market soon.

The test devices we used are off-the-shelf smartphones with the latest publicly-available software, rather than lab models: the results here represent a snapshot in time. Similarly, there are no special settings or software in the WLAN infrastructure. These results reflect what would be seen, heard and experienced in practical, state-of-the-art networks.

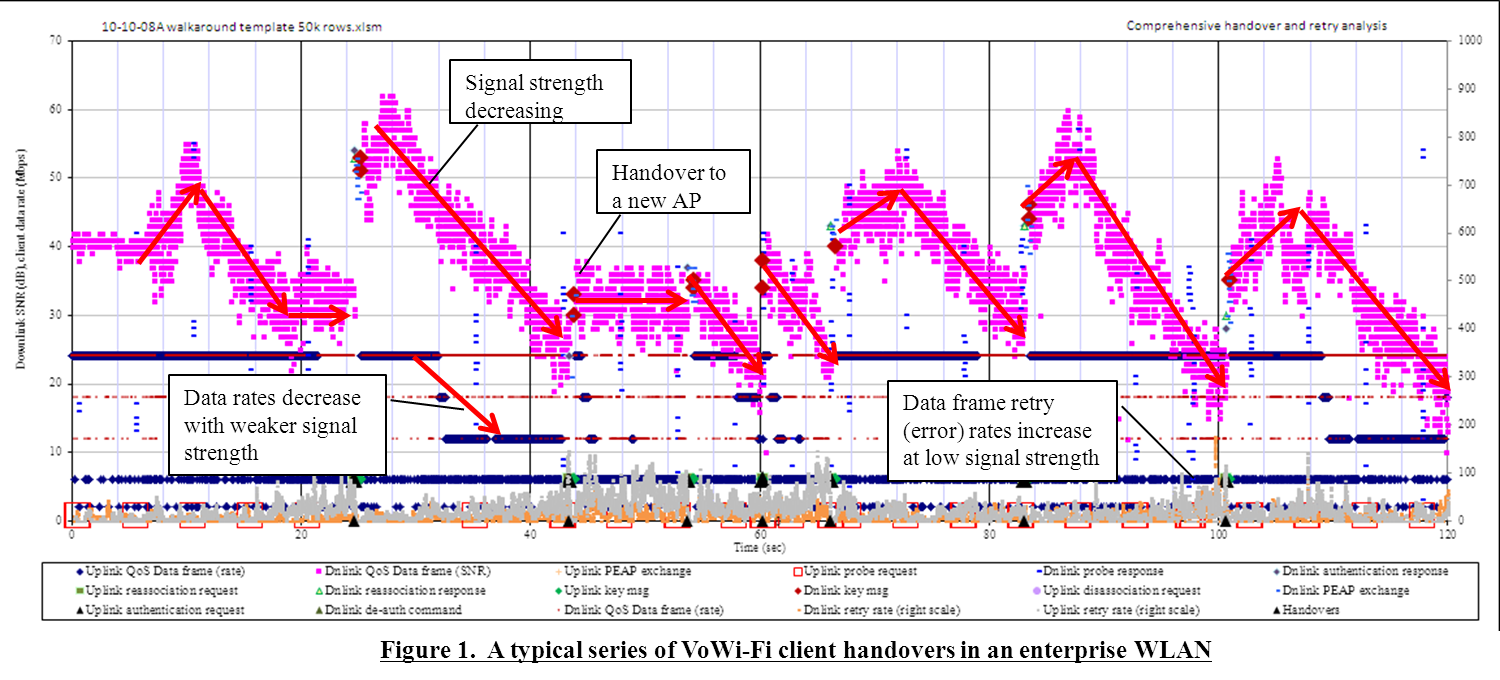
This paper represents a subset of many Wi-Fi client tests. The particular examples used here were selected as they are typical of what might be expected in a ‘real’ network, and we have simplified our analysis in order to highlight the most important results.

This note covers three areas. First, we present and analyze empirical results from ‘open-field’ tests of smartphones with VoIP clients on enterprise WLANs. The emphasis on real-world performance helps us identify and quantify the interruptions due to various phases of inter-AP handover, because poor decisions concerning handovers are the root cause of most impairments. Second, we discuss potential remedies based on improving existing algorithms, to illustrate where client developers can make the most significant improvements. Finally, we identify the mechanisms introduced by the Wi-Fi Alliance in Voice-Enterprise and show where they can improve performance.

## Anatomy of a handover

An on-call multimedia over Wi-Fi client device must support a continuous stream of media frames, usually one every 20 msec in each direction of the call. This is relatively easy to accomplish when the client is static, but enterprise WLANs consist of large numbers of coordinated APs, in the order of 20 meters apart. When the user is moving through the building, the device must shift its association from AP to AP to maintain a usable connection with high signal strength.

In order to analyze handover behavior, we use multi-channel test equipment to capture 802.11 frames on the air, both to and from the client device. With some post-capture analysis, we can construct a narrative for the client as it moves from AP to AP through the building. We are not able to see how the smartphone makes its decisions directly, although phone vendors themselves can use diagnostic code to track their algorithms, but we can infer a decision-making process from observing behavior in a variety of handover events. The graph below follows a smartphone as it moves through an enterprise WLAN.



There is a great deal of information in the graph, but the following are the main points of interest:

* The horizontal scale is in time (seconds), but this also relates to distance, as the user moves at a constant 1.6 meters/second through the building. As the route turns corners and passes doorways, RF conditions can change very quickly.
* The magenta dots each represent a downlink data frame, from an AP to the client, on a vertical axis of signal strength (SNR). When the user walks towards an AP, the signal strength increases. At some point, the user’s path moves away from the AP, and signal strength weakens.
* Inter-AP handovers are identified by step-changes in received signal strength, and also by the black triangle markers, showing ‘authentication request’ frames from the client. We expect to see AP signal strength after the handover significantly higher than before, if the client made a good choice of target AP.
* Since the sniffer device is carried next to the smartphone, signal strength on the uplink always appears good. Uplink frames are represented by diamonds, sorted on the vertical axis by data rate. Generally, as signal strength weakens, data rates are reduced. Downlink data rates are plotted as bars on the same scale.
* Retries occur when no ack was received for the original frame. The graph plots uplink and downlink retries separately, on the right-hand scale. Generally, when signal strengths decline, retry rates increase, although the effect is not always linear. We like to see a handover initiated before retry rates rise high enough to affect voice quality.

In the example above, the client executes six handovers over a two-minute circuit of the building. This is somewhat of an upper bound for practical handover requirements – not many people walk as far or as fast as this – but by no means unreasonable as a worst-case for design purposes.

Handover can be analyzed in three stages:

* The client must decide which is the best candidate AP for a handover;
* It must decide when to initiate an inter-AP handover;
* It then executes the re-association to the new AP.

Most lab testing regimes focus only on the third phase, but the information in this paper shows that the other stages are more important in determining performance. While re-authentication protocols must be closely coordinated between client and infrastructure, WPA2-enterprise is now widely understood, and implementations seldom diverge very far: once a re-authentication is initiated, it seldom fails because of protocol or state mismatch. Given reasonable conditions, the variation in re-authentication times is in the order of 10%. However, when they occur, errors and failures are usually due to holding the old AP too long before deciding to handover, or to a poor choice of target AP. While outages due to these reasons are generally infrequent – perhaps less than 15% of handovers for a bad case – they can result in media outages of several seconds, which will be noticeable to the listener and can sometimes result in calls dropping.

We can gain an insight into each stage of handover from reviewing the mechanisms required and examining the graphs.

## Deciding on the best candidate AP for handover

A multimedia client should always maintain a list of handover candidates, because RF conditions can change very rapidly and it may need to make a handover decision quickly, for instance when the user turns a corner or closes a door. Since APs operate on different RF channels, the smartphone must steal time between sending and receiving voice frames to scan both the current channel and other channels and identify APs with good signal strength.

Scanning can be passive or active. Since APs regularly broadcast beacons, it is possible to switch to other channels and monitor the beacons of APs on those channels. But beacons are relatively infrequent, usually repeating every 100 msec, so it can take a while to build up a list of candidate APs in this way. Most clients use active scanning, where they will switch to another channel, transmit a probe request, and remain on the channel for several milliseconds to receive probe responses from audible APs. This should reduce the time off-channel to perhaps 15 msec rather than the 100+ msec necessary for a passive scan. Our tests indicate that even in a congested WLAN, probe responses will be received within 10msec, although quite often they are missed altogether due to contention for the wireless medium.

The analysis tool cannot show passive scans, but every probe request is tracked, as are responses from nearby APs. By observing the pattern of probe requests we can learn about the client’s scanning algorithm. There are many other tricks to optimizing AP candidate lists, but the pattern of probe requests is a strong determinant of success.

Some smartphones broadcast probe requests every 5 seconds or so, while others don’t probe at all until signal strength falls towards a 30dB SNR threshold There seems to be little difference in performance: it’s the probe pattern when signal strength falls below the threshold that is important. Most algorithms increase the frequency of probe requests when SNR is below 30 dB, as they need to construct a candidate list: conditions might change very quickly and it is important to have a short list ready to go. A burst of 2 – 3 probe requests per channel every 3 – 5 seconds seems to be successful. The number of channels scanned should be limited to those where the SSID (ESSID) has been seen before, usually 1, 6 and 11 in the USA, to minimize the time spent off-channel, and the overall duration of scanning the environment. Also, probe requests should be directed to the specific SSID used by the enterprise – an ‘open’ request may elicit many unnecessary responses.

Even though active scanning reduces the time off-channel, voice frames can be lost unless the smartphone has an algorithm that times frame intervals. This is difficult to do, and anyway it can be defeated by jittered downlink frames. A better solution is to invoke Wi-Fi Multimedia –power save (WMM-PS), where downlink frames are buffered in the AP and delivered immediately following an uplink frame. Now, the client knows it will not lose downlink frames because it is off-channel when the AP needs to send them.

As we cannot see the client’s candidate list, we must judge its effectiveness by results. This means looking at the choice of target AP in a handover. The simplest measure of performance is whether the target AP offers higher signal strength than the old AP, and this is readily seen from the graphs. A more nuanced view compares the chosen AP with other possible target APs: these are visible by their earlier probe responses, in answer to probe requests from the client. If the data frames after handover are at the level of the strongest probe response, the choice was good. If a stronger signal existed, it may have been a better choice, although if clients broadcast with open probe requests, many responses will be from other SSIDs or networks that could not be used for handover, and these need to be removed from the analysis.

## Handover timing - Deciding when to initiate a handover

Once a smartphone or other Multimedia over Wi-Fi client has a short list of candidates for handover, it must still make the decision that the moment has arrived. This is more difficult than it seems, because as the graphs in this paper show, normal RF fluctuations cause the signal strength of an AP’s frames to vary over a range of at least 6dB, without considering any perturbations from closing doors or turning corners. Multimedia over Wi-Fi client designers have found that they must take account of the current AP’s signal strength, sometimes with both short-term and longer-term average levels, as well as the signal from the target AP.

For instance, as a rule of thumb it is normally a good thing to initiate a handover when signal strength from the current AP falls below the 25-30dB level. But if the best candidate AP is only at 30dB, it may represent a worse choice than the current one. Similarly, even though the current signal may be good, a closer AP with an even stronger signal may justify a handover decision. Other parameters such as error rates may be usefully incorporated in the algorithm, to move away from a noisy channel.

Most client designers have an in-built bias towards ‘stickiness’. Their algorithms don’t decide to handover until the situation is dire, signal strengths are low and error rates high. This may be acceptable for data-oriented clients, but with multimedia services it results in poor call quality, as error rates increase non-linearly when signal strengths drop below 20 dB SNR.

## Execute the handover

Just before initiating a handover, the client usually sends a probe request to the target AP to verify it is still available with good signal strength. Then it starts to re-authenticate to the new AP. Depending on the authentication protocol used, this can entail 50 or more frames over the air, and take several hundred milliseconds.

While the re-authentication phase of handover is the one that is usually emphasized in lab tests, as it is quite repeatable and test conditions can be closely controlled, it is often a minor contributor to overall handover times.

Following the handover, the client is associated with the target AP, and data frames should resume in both uplink and downlink directions.

## Measuring the outage

The simplest measure of handover time is the re-authentication phase: usually from the authentication request to the last key frame completing the protocol. This is the easiest figure to measure, and it can be easily automated in test tools. But the measure that matters to the user is how much of a gap is heard between the last frame received (or sent) on the old AP and the first frame on the new one. This is more difficult to measure, and it depends on higher-layer functions as well as the Wi-Fi protocol.

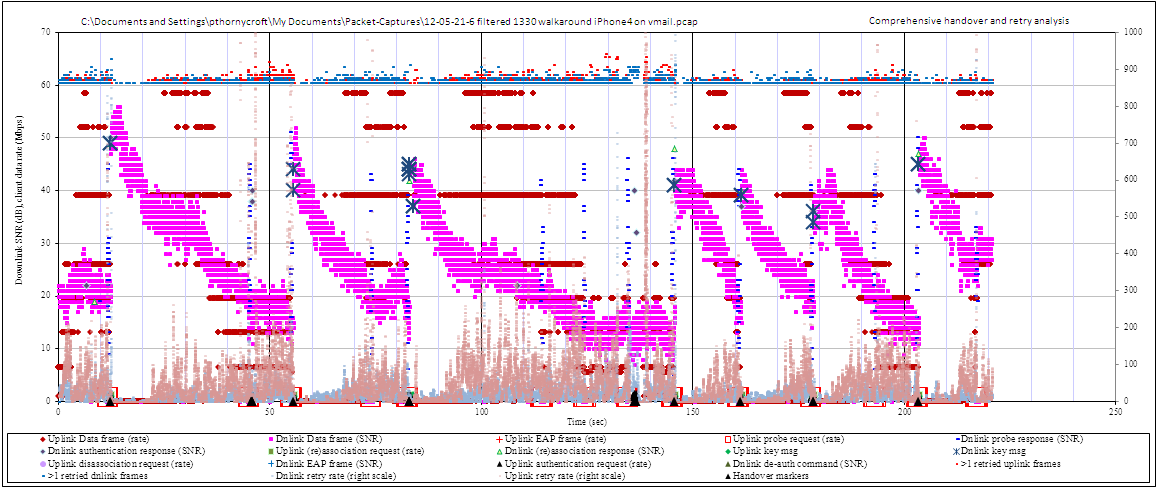
Indeed, to tell the complete story we should include estimates of how good voice quality was, before and after the handover. In this paper we offer figures for re-authentication protocol and media interruption, along with qualitative comments about retry rates around the handover.

The next sections review results from recent handover tests. We took current smartphones from Apple (iOS) and Samsung (Android), set them up for SIP/RTP voice calls and walked them around Aruba office buildings.

We tested in two separate buildings to get an indication of performance in ‘clean’ and ‘challenging’ conditions. The ‘clean’ building is empty, as it is used mostly for large-scale testing so there is very little background Wi-Fi activity or non-Wi-Fi interference. Further, the WLAN in this case is configured for PSK so the reauthentication exchange is very short. Apart from the PSK option, this environment reflects what would be expected in many practical WLANs.

The ‘challenging’ building houses many Aruba employees, including support engineers, most of whom run their own Wi-Fi labs to troubleshoot problems. This results in a large amount of Wi-Fi traffic on the air: even off-hours when we tested, there were >80 BSSIDs audible across the 2.4GHz band, and a substantial amount of associated traffic. In these conditions, probe requests and responses can be lost due to media contention, or delayed so long that the client has returned to its home channel before hearing the probe response. Thus, the client can be unaware of the presence of a suitable AP, even though it has scanned the appropriate channel. Also, retry rates will rise due to co-channel interference, a phenomenon that has a near-far effect and is exacerbated by low signal strengths. We believe this ‘challenging’ environment is more hostile than the normal WLAN installation, but it allows us to benchmark ‘worst-reasonable-case’ performance.

## iPhone4 (iOS-Apple) handover performance

The first iPhone test was in our ‘clean’ building with very little Wi-Fi activity. We set up a voice call using Aruba’s SIP server (Avaya SES) and the 3CX client on the iPhone. Many SIP clients are available on the App Store, and in our experience they have similar performance – in fact, most use the mjsip open-source SIP engine. Two circuits of the building yielded ~8 handovers. The following is representative of several tests.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| timestamp | SNR dB before | SNR dB after | h/o time ms | impaired s | notes |
| 12.325 | 23 | 50 | 28 | 0 | good handover |
| 45.729 | 16 | - | - | 0 | exchange of authentication frames but client never sends reassociation request |
| 55.551 | 15 | 44 | 35 | 9.8 | re-auth to same AP as above, this one is good |
| 82.967 | 18 | 44 | 1025 | 2 | extra 980msec because Key1 was not ack’d by client x9 and abandoned |
| 136.253 | 12 | - | - | 16 | exchange of authentication frames but client never sends reassociation request |
| 145.723 | 16 | 42 | 28 | 10 | re-auth to same AP as above, this one is good |
| 161.403 | 18 | 40 | 28 | 0 | good handover |
| 178.513 | 20 | 35 | 28 | 0 | good handover |
| 203.433 | 15 | 45 | 33 | 8 | good handover |

The diagram and analysis above show 9 handover attempts of which 7 resulted in successful handovers. We can analyze performance based on the three phases of handover defined earlier.

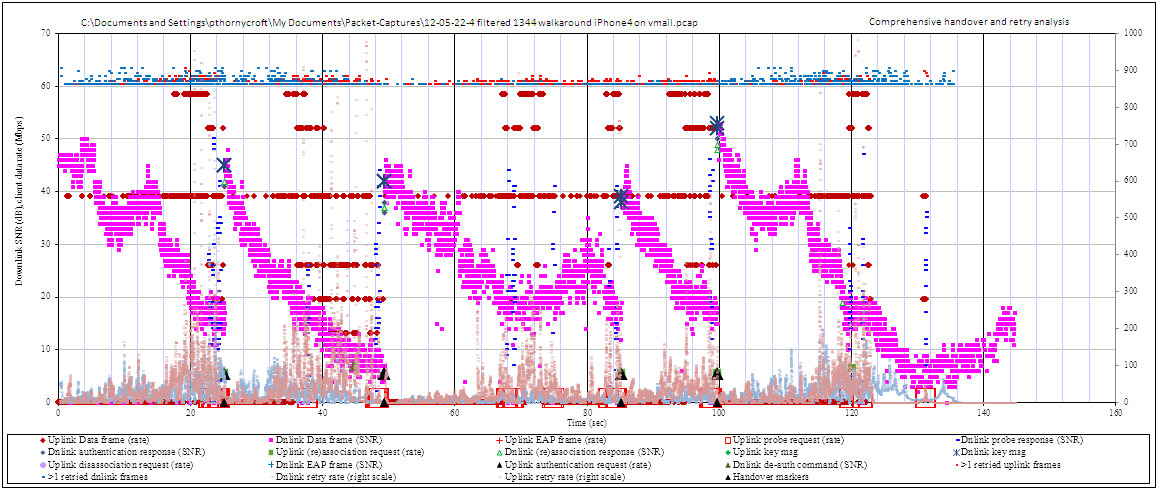
Scanning pattern – Probe request patterns start consistently when the received SNR falls to 20dB. Below 20dB, we see probe request scans every 10 seconds. Probe requests are all directed to the current SSID, and cover all three channels. The client selects the best, or nearly-best target AP in nearly all cases, and does not take long to make the decision. It appears that this phase of handover is well-implemented on the iPhone, although we would like to see the initiation threshold set at least 5dB higher.

Timing of handover – Handover attempts appear to be triggered after a probe request scan reveals suitable AP candidates, and handovers are initiated in most such cases. However, we see several occurrences where the scan showed good APs but no handover was triggered, at 114, 124 and 193sec on the trace. It’s not clear why there was no handover in these instances, but they resulted in periods of impaired media quality, quite long periods because the scan interval is ~10sec.

Execution of handover – this network uses PSK, so the frame exchange is only 8 frames (not including acks and possible retries). Even so, in 2 of the 9 re-authentication attempts the protocol got stuck – in the same place, after the authentication response and before the reassociation request from the client. This was probably caused by a bug – we haven’t seen this syndrome before, and we expect it to be fixed soon. The interval before a new attempt was ~10sec in both cases, and we can speculate that this is linked to some software timer in the client. While there were no very serious consequences here, the 10sec delay in handing over could have caused dropped calls in a more complex building topology.

Media break due to handovers was 1.2 sec in 220sec, or ~0.5% of the run. Media impairment (defined as SNR < 20dB) was ~20%.

Our second test with the iPhone was in a much more challenging environment. This building has more internal walls and partitions than the first, and there is considerably more Wi-Fi traffic on the air – due to the proximity to Aruba’s TAC group, 80+ BSSIDs are audible across the 2.4GHz bands.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| timestamp | SNR dB before | SNR dB after | h/o time ms | impaired s | notes |
| 25.139 | 15 | 45 | 26 | 6 | good handover |
| 49.371 | 8 | 42 | 28 | 13 | handover was good, but way too sticky, should have been 10 seconds earlier |
| 85.263 | 17 | 39 | 25 | 6 | handover was good, but should have happened at 69sec |
| 99.677 | 18 | 52 | 42 | 2 | good handover |

The trace above shows four successful handovers in two circuits of the building. Overall the handovers were successful, but it is clear from the chart that there should have been more handovers – much of the time the client was associated with a distant AP with poor signal strength, when we know that all parts of the building have good Wi-Fi coverage. We will analyze the trace and show what should be improved.

Scanning pattern – The pattern is similar to the earlier trace in the clean building. When received SNR falls to 20dB, or just below, we see a burst of probe requests that repeats every ~10sec. When a handover is initiated, the chosen AP always has SNR of >40dB, reflecting a good choice by the client. However, the scan interval of 10sec is too long for traversing a congested WLAN at walking speed, as scans seem to miss good APs, and RF conditions change quickly.

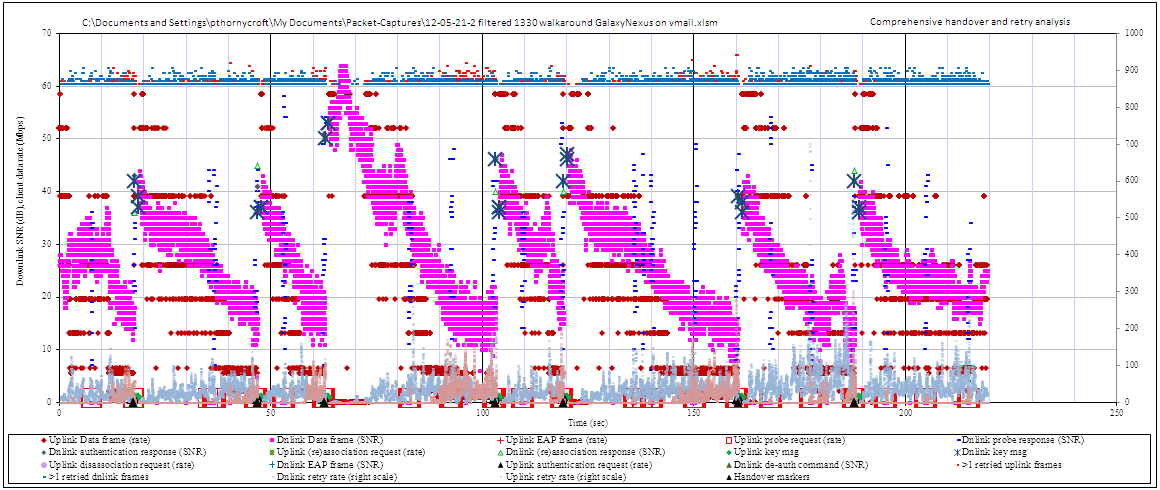
Timing of handover – There are several cases where, although probe responses indicate good candidate APs, the client failed to find a good AP and handover. The scan at 38sec, for instance, revealed an AP at 26dB SNR when the current AP was 16dB. Even though we would prefer the new AP to be ~40dB (and such an AP exists, but its probe response was not seen due to Wi-Fi congestion and contention on the air) even that AP would have been an improvement. As a result the media stream suffered very bad SNR until an eventual handover at 49sec – this could have been avoided by handing over after the initial scan, or starting a new scan a few seconds later. Similarly, the scans at 69sec and 75sec both show good handover candidates, but no handover was initiated. And the scan at 119sec was a good handover opportunity missed.

Execution of handover – All four successful handovers were very quick, as they used PMK caching. This was because we made several runs already that morning, and the client had connected to these APs and established keys. An iPhone normally takes ~250msec to execute a full PEAP-MSCHAPv2 authentication sequence with ~23 frames exchanged, not including acks. If clients make a bad choice of AP, and poor SNR results in retries and lost frames, the longer re-authentication sequence can result in extended or failed handover attempts. If the iPhone were to implement OKC (opportunistic key caching), a more general version of PMK caching, we would expect all handovers to execute this quickly.

Media breaks due to handover took ~0.08% of the run, media was impaired ~22% of the time.

## Galaxy Nexus (Samsung-Android) handover performance

The Android OS includes an organic SIP client since Rls2.3, however we found while testing in this challenging environment that it was prone to dropping calls during handover interruptions. The free 3CX client from Google Play was a little more forgiving, so it was used for these tests.



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| timestamp | SNR dB before | SNR dB after | h/o time ms | impaired s | | notes |
| 17.671 | 16 | 40 | 1037 | 3 | slow execution due to no Key2 from client | |
| 46.798 | 17 | 51 | 960 | 3 | slow execution due to no Key2 from client | |
| 62.798 | 17 | 52 | 971 | 4 | slow execution due to no Key2 from client | |
| 103.113 | 15 | 40 | 870 | 10 | slow execution due to no Key2 from client, should have been 10sec earlier | |
| 119.134 | 20 | 43 | 942 | 1 | slow execution due to no Key2 from client | |
| 160.491 | 12 | 38 | 987 | 5 | slow execution due to no Key2 from client | |

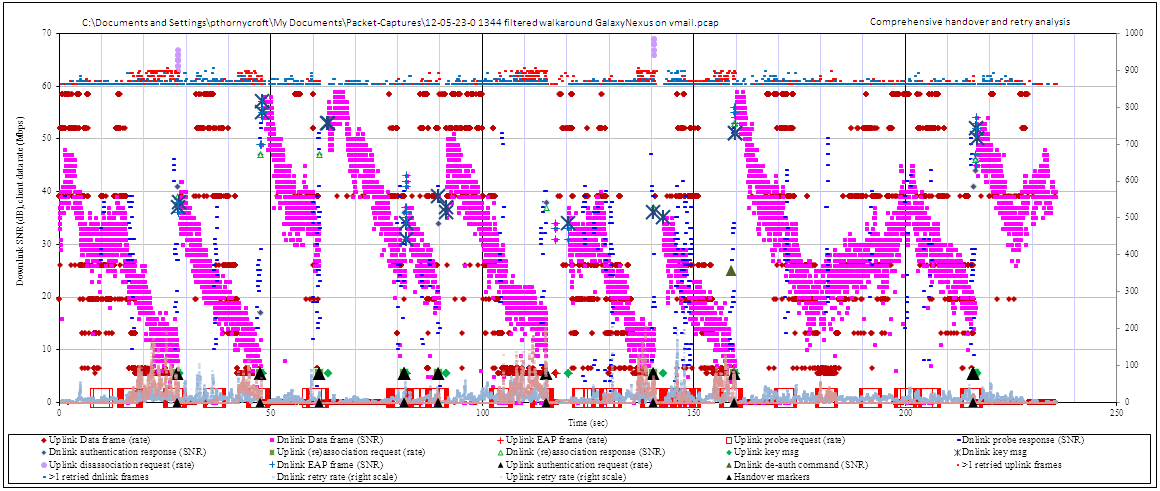
The chart above, from the ‘clean’ building, shows good handover behavior. The one consistent problem is in executing the re-authentication protocol, which we will examine below.

Scanning pattern – Probe request scans are generally every 10sec, with some at ~4sec intervals. They start a little earlier than for the iPhone, at around 25dB SNR, so there are more scans on the trace. The choice of handover AP is always good, with SNR >37dB, although this is a very easy WLAN for a client to work with.

Timing of handover – While probe requests are triggered at a good signal level, not every scan is followed by a handover attempt, despite the presence of much better APs in all cases on the chart. It’s not clear, for instance, why the scans at 38, 53, 84, 94, 109, 137, 139, 150, 168, 178sec do not result in handover attempts – possibly the current signal is above its threshold, but if an AP with 25dB better signal is discovered, the client should be handing over in every case.

Execution of handover – There is a consistent problem running through this trace. Although authentication is by PSK and uses very few frames, the Key2 frame from the client is always delayed. The sequence is that the AP initially sends Key1 and sees an ack, but there is no subsequent activity from the client. After a timeout of ~900msec, the AP re-sends Key1, with an immediate Key2 response and the exchange completes. This appears to be a problem with the client implementation, but more debugging is needed to properly characterize it.

If we ignore this protocol delay, the handover performance of this Galaxy Nexus in the ‘clean’ building is very similar to the iPhone. The higher scan threshold gives us hope for better handover performance, but it seems the handover decision thresholds are not changed.

Media breaks due to handover took ~2.7% of the run, media was impaired ~12% of the time.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| timestamp | | SNR dB before | | SNR dB after | h/o time ms | impaired s | notes |
| 27.981 | 12 | | 37 | | 335 | 5 | good handover but 10sec too late |
| 47.664 | 10 | | 56 | | 474 | 2 | good handover but 10sec too late |
| 61.432 | 30 | | 53 | | 107 | 0 | good handover, using PMK caching |
| 81.727 | 28 | | 33 | | 365 | 0 | good handover |
| 89.609 | 25 | | 45 | | 2016 | 2 | Long pause in handover execution, using PMK caching |
| 115.099 | 14 | | 34 | | 5343 | 10 | Long pause in handover execution, no response to Key1 frame |
| 140.579 | 12 | | 37 | | 2199 | 10 | long pause in handover execution, using PMK caching |
| 159.581 | 10 | | 52 | | 282 | 10 | good handover, but 15sec too late |

While the Wi-Fi chip in the Galaxy Nexus is from the same vendor and family as the iPhone’s, the drivers are customized to varying degrees, accounting for the differences in performance. In the more challenging WLAN environment, above, we see that the Galaxy Nexus generally makes good handovers but performance could be improved.

Scanning pattern – The basic pattern is a scan every 10sec, as in the first test, but we see several episodes where many probe requests are sent, for instance 59 probe requests from 145.1 – 148.4sec. While we usually suggest more frequent scans, this seems excessive – and for every probe request, the client must wait off-channel for the probe responses, so it’s not without cost. Apart from these spurts, we see probe request scans at ~10sec intervals, but not when received SNR is above 30dB. We’d prefer to see a 5sec interval. The choice of AP again seems to be good, there aren’t many probe responses stronger than the chosen APs.

Timing of handover – This client is too sticky, and handovers occur too late. And this despite a good scanning algorithm. For instance, at 17sec there is a scan with responses up to 39dB, but the client waits another 10sec for another scan before handing over at 28sec, when the signal is very low. Similarly, there should have been a handover at 181sec, where the received SNR was 22dB and an AP 30dB stronger was available.

Execution of handover – A problem with handover execution runs through this trace, but it is different from the iPhone and has different symptoms. First, there are far fewer PMK caching attempts. This may well be because the iPhone test we used here was probably the fifth or sixth of the day and it had already authenticated with most of the APs, while the Galaxy Nexus was the first or second. (The results here are cherry-picked in the sense that probably 50% of runs in the challenging WLAN ended prematurely in dropped calls – this is one of the better runs by definition.) But it may also reflect the algorithms used in the client. We know that there is a greater risk of breakdown in long authentication protocols when the environment is noisy and congested, as it’s more likely that frames or acks will be lost. In this case, however, the Galaxy Nexus completes the open authentication and reassociation frame exchanges, then immediately starts to negotiate a Block Ack and send upstream frames, before completing the re-authentication. This may reflect a state mismatch between the network and the client, where it is not ready to respond to the next EAP frame the AP sends, Request, Identity. The traces for each handover attempt are similar, suggesting this may be a software issue.

Media breaks due to handover took ~4.4% of the run, media was impaired ~17% of the time.

## Notes on Quality of Service and other features for voice

In this section we examine how well the smartphones deliver high-quality voice streams. Voice quality depends primarily on delay, jitter and packet loss. Long network delay from talker to listener eventually makes conversations difficult, as co-ordination suffers, but the WLAN contributes very little to this effect, perhaps 10msec in a budget of 150 – 200msec. Jitter can be damaging if the receiver’s jitter buffer is small: when the buffer under- or over-runs there is a gap in speech either because packets are lost, or while waiting for the next packet. And frames can be lost over-the-air, but Wi-Fi has a retry mechanism that attempts to re-transmit a frame several times before giving up, so irretrievably lost frames are unusual under normal conditions, and retries add to jitter but not more than a few milliseconds. Finally, congestion on the air can make it difficult for a phone to ‘seize the medium’ for a transmit opportunity: this can be a source of lost packets under heavy-load conditions, particularly without QoS priorities.

In this section we will discuss the significant standards and techniques necessary for good voice quality over Wi-Fi. We can’t deal with all of the nuances of good multimedia over Wi-Fi design in a short document, but luckily most of the benefits accrue from two key features:

* Compliance with the Wi-Fi Multimedia (WMM) standard; and
* Compliance with the WMM-power save (WMM-PS) standard

And we can observe some of the facets of good voice traffic over the air:

* Uplink jitter performance; and
* Selection of transmission rates; and
* Resulting retry (error) rates over the air,

## WMM priority for Quality of Service

Much can be written on QoS for Multimedia over Wi-Fi, but most of the benefits depend on implementing just this one aspect of the standards. WMM is a Wi-Fi Alliance certification that gives voice frames priority over-the-air compared with other forms of traffic. If a client (or AP) has high-priority traffic queued to transmit, it has an opportunity to send it before other, lower priority traffic is allowed on the air. WMM is quite simple to implement, and very effective. It only really makes a difference when networks become congested with traffic, which is why many multimedia over Wi-Fi applications aimed at home use still do not implement it, but it should be de rigueur for enterprise applications, as momentary congestion can appear quite frequently in enterprise WLANs.

Other smartphone vendors such as Nokia and RIM apply WMM to their organic multimedia over Wi-Fi clients, and it works well. For both the iOS and Android operating systems, WMM is available but the application that must invoke WMM priority or the lowest priority level will be assigned. Most SIP/VoIP client applications for these operating systems do not currently apply WMM, the Bria client for iPhone being an honorable exception. But we expect these shortcomings to be rectified over time.

In short, WMM is a required feature for multimedia over Wi-Fi; it is available on both iOS and Android platforms, but it is not commonly invoked by applications – Apple’s FaceTime application does not use WMM (tested on an iPhone4 with software version 5.1.1), nor does the native SIP client in Android.

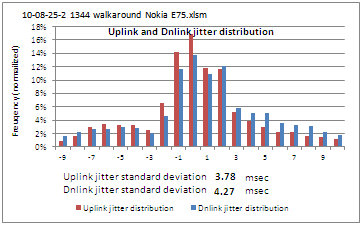
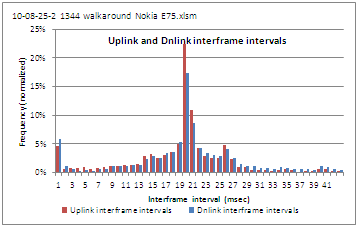
## WMM-PS for downlink traffic triggering and battery life

Battery life is not strictly a QoS feature for smartphones, but it is such a key determinant of user satisfaction that we include a reference here. The Wi-Fi chips of several years ago were quite power-hungry, but much progress has been made in the standards and by the silicon designers in bringing down the power draw when Wi-Fi is turned on. The remaining significant gain comes from implementing WMM-PS, a Wi-Fi Alliance certification that allows the phone to ‘sleep’ between frames when on-call. To avoid losing downlink traffic while the client is sleeping, the AP must buffer frames for the client until it sees an uplink transmission, when it replies immediately with its downlink frame.

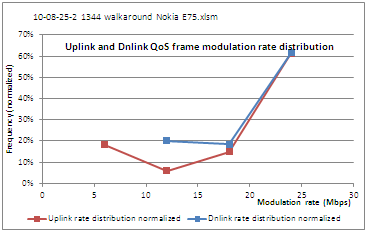
WMM-PS is useful because it allows the phone to switch off its Wi-Fi radio for inter-frame intervals when on-call, increasing talk-time in the order of 300% in phones Aruba has tested. But it also ensures that downlink and uplink frames are grouped together, so the phone knows it can scan off-channel, or perform other tasks between its transmissions without the risk of losing downlink traffic.

Our testing reveals that while an earlier generation of smartphones from Nokia and RIM did indeed implement WMM-PS, the current general-purpose platforms from Apple and Google-Samsung do not. This is probably due to three factors. First, the feature is not available on most home APs, and enterprise applications are still not a focus for smartphone vendors. Second, when the application is separate from the OS, it is not trivial to provide the APIs that allow the application developer to invoke WMM-PS, and third, even when such APIs are available, the application must be written to use them properly – and as we have seen, this has yet to happen with the much-simpler WMM QoS APIs.

## Voice Quality analysis graphs

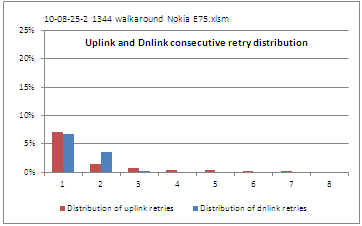
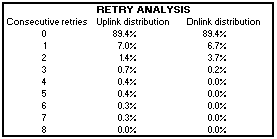
The first dimension we analyze is jitter. In all these graphs, the red histograms or lines are for the uplink, from the smartphone to the AP, while blue is for the downlink: it can be useful to compare the two. The test calls were made to the voicemail server, so we believe frames were generated with good timing integrity for the downlink, while uplink timing depends on the phone and over-the-air conditions, given that silence suppression was not used on the uplink.

The first graph (on the left) is an aggregation of inter-frame arrival times. Simply put, we subtract the timestamps of consecutive frames. For most codecs, including G.711 which is used in all these tests, we expect frames to be generated at regular 20 msec intervals, and the graph indeed peaks at 20 msec. Since we are including retry frames, there is a small peak at 1 msec for immediate retries, also some cases where two frames are buffered and sent consecutively, with WMM-PS. Sometimes we see a local peak at 40 msec, where the sniffer missed the odd frame. Where WMM-PS is used, we expect to see up- and downlink plots match, as the downlink is triggered by uplink frames.

The right-hand graph uses very similar data, the frame arrival timestamps, but divides them mod 20 msec to see how stable the internal clocks are. It is sorted for center-weighting, so we always see the peak at 0 msec, and the distribution reflects how jitter, clock wander and step-changes affected transmit and receive frame timing. We will see that some smartphones keep very accurate clocks, while others vary considerably, which could have consequences for voice quality by increasing overall jitter.

The second area we analyze is the over-the-air data rates used by both the smartphone and the AP. In Wi-Fi, each device sets the data rate (modulation rate) it uses for transmissions, usually based on how many acks it sees. If all frames are successfully ack’d, the rate is increased, if acks are missed, the rate is reduced: for a given SNR, a higher data rate will be associated with a higher error rate. As network engineers we like to see high data rates, as this maximizes overall WLAN data capacity, but for multimedia it is often better to reduce the rate to minimize retries – every retried frame increases air occupancy and jitter.

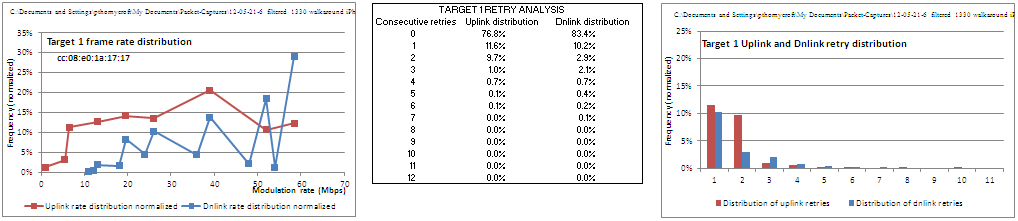
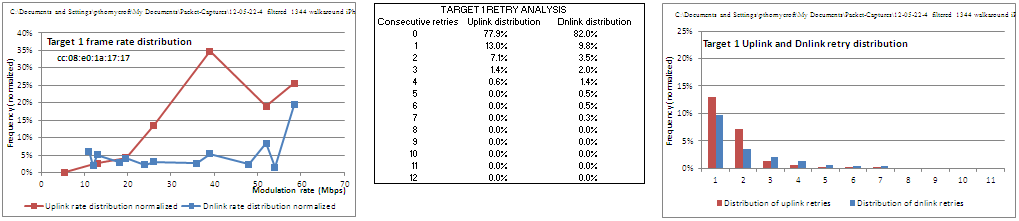
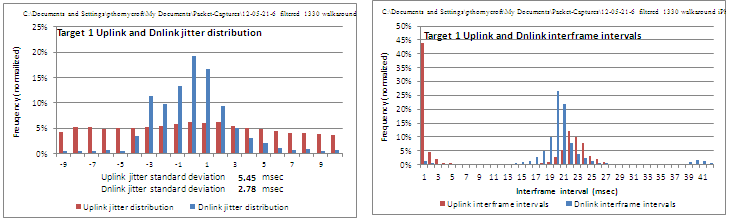
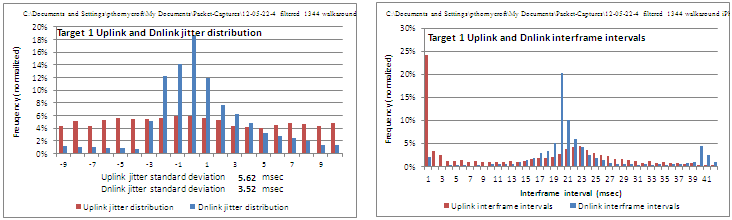
In the graph above, the uplink (red) frames are mostly at 24 Mbps, with some at 6, 12 and 18 Mbps. These are 802.11g rates. We prefer to see 802.11n phones, as those tested here are, using 802.11n rates, because they have slightly higher data rates (they are only single-spatial stream, so 58.5 Mbps is usually the top rate) and support a number of options introduced with 802.11n.

Lastly we analyze retried frames. Whenever a transmission is not ack’d (which can be because it was not received by the far end, or because the ack was lost), the frame is queued for immediate re-transmission, subject to air occupancy. A low rate of retries is always expected, as wireless is an inherently uncertain medium, but large numbers of retries increase jitter, and when a frame has been re-transmitted a number of times (usually from 4 – 8, depending on device settings), it is dropped, and the far end will never receive it.

Our analysis shows how many frames have one, two, three… retries. For instance, in the graph above (the figures are the same in the table and chart) 89% of frames are ack’d the first time, without re-transmission. Of the remaining 11%, most have just one retry, and the numbers fall off rapidly till only 0.3% of frames are re-transmitted 7 times which indicates in this case that they are dropped. The downlink shows a similar pattern, 89% of frames are ack’d immediately, but nearly all are corrected by one or two re-transmissions, and it appears none are lost. It takes further analysis to determine whether the smartphone or the AP is missing more frames or acks on the air, but this graph shows us whether the situation is normal, or we need to investigate.

In the following pages, we report these results for each of our smartphones.

## iPhone4 call quality measures

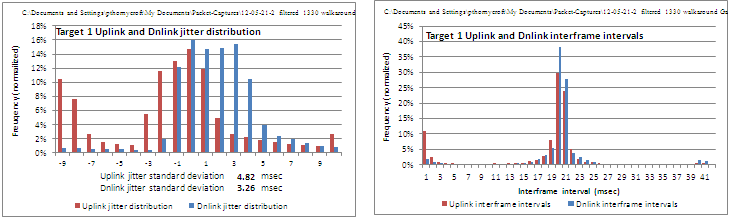
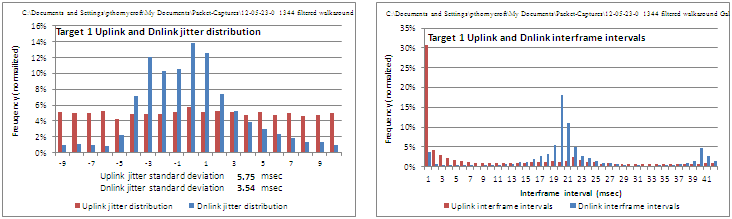
The graphs for the iPhone jitter and interframe intervals are similar (in each case below the top graphs are from the ‘clean’ building and the lower ones are from the ‘challenging’ building). Downlink frames show a strong central tendency to 20msec spacing, while the upstream frames, while they do cluster around 20msec spacing, are more random. The iPhone tends to transmit frames in pairs, resulting in the strong line at 0msec interframe intervals. This may reflect a desire to group frames for Block Ack, although BA is not used here.

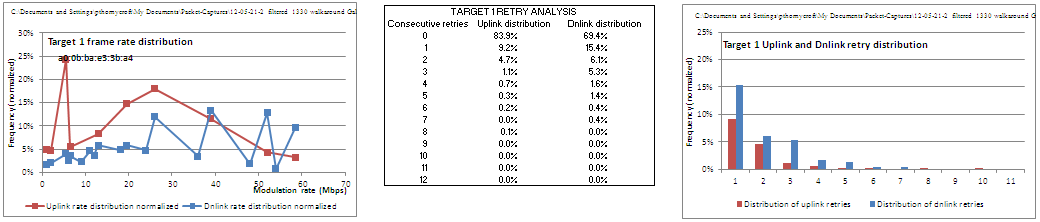
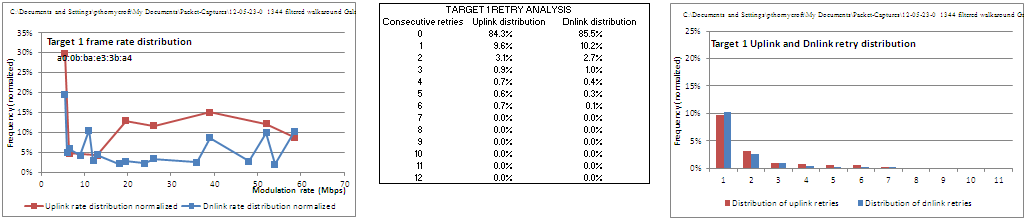
Both uplink and downlink show mainly 802.11n rates (6.5, 13, 19.5, 26, 39, 52, 58.5Mbps). The iPhone uses these rates exclusively, along with a few frames at 1 and 5.5Mbps. In the challenging building, 80% of uplink frames are at rates of 39Mbps or higher, while in the clean building the figure is 43%.

The retry rates are similar in both environments. Around 77% of uplink frames a successful first-time, while 11% require one retry and 8% two retries. This is similar to the downlink figures, and reflects a good choice of transmit rates – we assume that most transmission failures are caused by co-channel interference, that is other Wi-Fi frames on the air, although there is no increase in retries in the challenging building. The two-retry figure is a little high in both cases, both compared to the downlink and other phones, but not unreasonably so.

Overall these graphs indicate good voice call quality. But while the iPhone is capable of WMM QoS, very few voice and video applications invoke the appropriate WMM priority – the Counterpath Bria client is the only one we know of, and in recent tests, Apple’s own FaceTime video application (on Version 5.1.1 software) did not set WMM correctly, leaving traffic at priority 0 (best-effort).

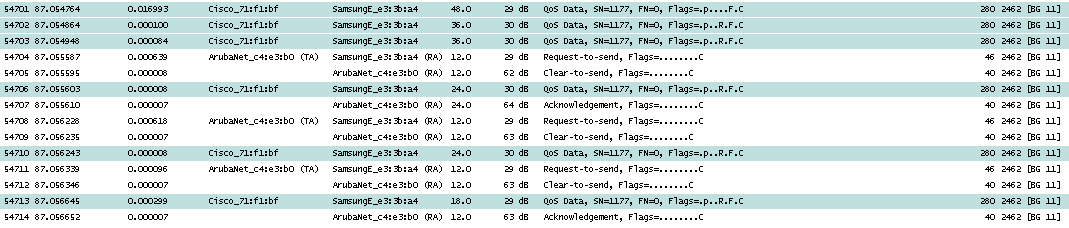
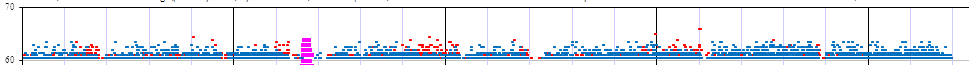
## Galaxy Nexus call quality measures

The Galaxy Nexus shows a better tendency to 20msec interframe spacing than the iPhone in the clean building, which should translate to less overall jitter across the connection, to some degree. However, the upstream jitter for the challenging building is very similar to the iPhone. One reason for this is that the Galaxy Nexus tends to retransmit frames with the same sequence number but without the retry bit set – this is responsible for the strong line at 0msec on the interframe spacing graphs.

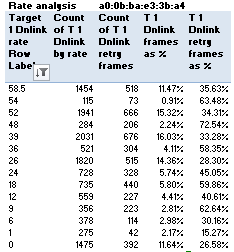


The uplink rates from the Galaxy Nexus are lower than those of the iPhone or the downlink. It uses 5.5Mpbs for 25% of its frames. But the percentage of frames at the high rates, 39Mbps and above, are not very different from the downlink. Comparisons on the graph are difficult because of the 802.11g rates used on the downlink.

First-time uplink success rates are at 84% for both environments, which is a normal profile. The downlink shows a larger number of retries in the clean building, and the trend continues through one-, two- and three-retry success. This warrants further investigation. First we can look at the distribution of retries on the panoramic chart (12-05-21-2) of which a snippet is posted below.

The red stack charts at the top show uplink retries, and they tend to cluster around the points where signal strength is low, as we would expect. However, the blue stack charts show a constant level of retries. Thus it does not appear to depend on signal strength.

Next we can look at the file in Wireshark to see whether the retries are caused by the client not hearing the data frames, or the AP not hearing the acks (we use the sniffer as the arbiter here – if it doesn’t see a frame, it probably didn’t exist). In the case above, frame 1177 is transmitted, then retried twice with no ack. Then the AP uses RTS-CTS and this time there’s an ack from the client. But the AP doesn’t see the ack because it retransmits the frame twice more before seeing the final ack. This and other sequences from the file show no strong pattern of one side or the other missing frames, in this case.

Finally we can look at retry incidence for different data rates, in case there’s a difficulty in receiving one particular rate. The table above shows such an analysis, but there is no strong pattern. The abnormal behavior may have been the result of external interference or other effects, we can’t tell from this evidence.

The Galaxy Nexus did not use WMM-PS or WMM voice tagging in these tests. We know the latter is a function of the application rather than the OS, and the 3CX client does not set WMM priority, but even the integral SIP client in Android 4.0.4 does not invoke WMM priority.

## Conclusion

As Wi-Fi-equipped mobile devices proliferate in the enterprise, whether corporate-supplied or employee-owned, there is an opportunity to connect these devices to real-time streaming media services such as voice and video. This paper analyzed how well two of today’s most popular smartphones perform on Wi-Fi in an enterprise WLAN environment, with a focus on multimedia over Wi-Fi applications.

Using Aruba-developed analysis tools, we were able to demonstrate that these devices are already quite capable of voice services, meaning they can be used with carriers’ voice over Wi-Fi offerings, or configured to connect directly to the corporate PBX when in range of the WLAN.

The complicating factor for multimedia over Wi-Fi in an enterprise WLAN stems from the large number of densely-deployed access points used to provide service. The mobile device must be able to recognize these APs, to continuously calculate the best AP to connect to, and to execute accurate and speedy handovers when the user moves and a new AP becomes a better choice for connection. These problems can be solved by state-of-the-art software incorporating standards, and suitable software algorithms targeted at the WLAN environment.

In an earlier paper we showed that Nokia and RIM (BlackBerry) devices offer excellent performance for multimedia over Wi-Fi. In part, this stems from several years’ investment in development and testing by these vendors, in co-operation with Aruba and other WLAN providers. It is also somewhat easier for these Nokia and RIM to deliver seamless voice services, as they control the hardware, operating system and applications on their devices.

The more recent smartphones tested here are not as capable. They usually perform well, sufficient to satisfy some users, perhaps most: but are they prone to bad handovers, causing long gaps in the multimedia signal, and their jitter and retry characteristics are not as good. The business models used by Apple and Android are partly responsible. Both operating systems (iOS and Android) are capable of WMM, but the applications must indicate priority for their voice streams, and at present most do not. This can be solved by better co-ordination between developer groups: all smartphone services depend on the chain from Wi-Fi chip and driver through the operating system to the application. We hope that UC vendors developing multimedia clients, along with a renewed focus on the WLAN enterprise environment from phone developers will drive wider understanding of these issues.

The new Wi-Fi Alliance ‘Voice-Enterprise’ certification (June 2011) includes advanced features from 802.11k, 802.11v and 802.11r making it much easier to develop high-performance mobile devices for the enterprise. Information exchange at the Wi-Fi layer will allow client devices to learn neighbor AP information that is useful in scanning and choosing handover targets. Voice-Enterprise also includes a mechanism for the WLAN to prompt clients to hand over to a new AP and a fast re-authentication protocol. All these features should make inter-AP handovers more accurate, quicker and more reliable. But it will take several years before all WLANs and clients are Voice-Enterprise compliant, so smartphone designers should ensure their existing algorithms are improved, where necessary, to perform well in the absence of Voice-Enterprise.

Aruba continues to develop features for higher-quality, more effective voice and video over Wi-Fi services, and we are confident that future mobile devices will offer ever-higher performance. The explanations in this paper only scratch the surface of our ongoing investigation and understanding of multimedia over Wi-Fi mechanisms and features.

About Aruba Networks, Inc.

People move. Networks must follow. Aruba securely delivers networks to users, wherever they work or roam. Our mobility solutions enable the Follow-Me Enterprise that moves in lock-step with users:

* Adaptive 802.11a/b/g/n Wi-Fi networks optimize themselves to ensure that users are always within reach of mission-critical information;
* Identity-based security assigns access policies to users, enforcing those policies whenever and wherever a network is accessed;
* Remote networking solutions ensure uninterrupted access to applications as users move;
* Multi-vendor network management provides a single point of control while managing both legacy and new wireless networks from Aruba and its competitors.

The cost, convenience, and security benefits of our secure mobility solutions are fundamentally changing how and where we work. Listed on the NASDAQ and Russell 2000® Index, Aruba is based in Sunnyvale, California, and has operations throughout the Americas, Europe, Middle East, and Asia Pacific regions. To learn more, visit Aruba at <http://www.arubanetworks.com>.

H

© 2012 Aruba Networks, Inc.  *AirWave*®, *Aruba Networks*®, *Aruba Mobility Management System*®,*Bluescanner*, *For Wireless That Works*®, *Mobile Edge Architecture*, *People Move. Networks Must Follow*., *RFProtect*, *Green Island, The All-Wireless Workplace is Now Open for Business*, and *The Mobile Edge Company*®are trademarks of Aruba Networks, Inc.  All rights reserved.  All other trademarks are the property of their respective owners.