An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process*

Arunesh Mishra
Dept of Computer Science
University of Maryland
College Park, MD, USA
arunesh@cs.umd.edu

Minho Shin
Dept of Computer Science
University of Maryland
College Park, MD, USA
mhshin@cs.umd.edu

William Arbaugh
Dept of Computer Science
University of Maryland
College Park, MD, USA
waa@cs.umd.edu

ABSTRACT

IEEE 802.11 based wireless networks have seen rapid growth and deployment in the recent years. Critical to the 802.11 MAC operation, is the *handoff* function which occurs when a mobile node moves its *association* from one *access point* to another. In this paper, we present an empirical study of this handoff process at the link layer, with a detailed breakup of the latency into various components. In particular, we show that a MAC layer function - *probe* is the primary contributor to the overall handoff latency. In our study, we observe that the latency is significant enough to affect the quality of service for many applications (or network connections). Further we find variations in the latency from one handoff to another as well as with APs and STAs used from different vendors. Finally, we discuss optimizations on the probe phase which can potentially reduce the probe latency by as much as 98% (and a minimum of 12% in our experiments). Based on the study, we draw some guidelines for future handoff schemes.

General Terms

Measurement, Performance, Experimentation

Keywords

IEEE 802.11, Handoff, Performance, Scanning, Probe, Association, Authentication, Latency

1. INTRODUCTION

IEEE 802.11 based wireless local area networks (WLANs) have seen immense growth in the last few years. The predicted deployment of these networks for the next decade resembles that of the Internet during the early 90s. In public places such as campus and corporations, WLAN provides not only convenient network connectivity but also a high speed link up to 11 Mbps (802.11b). In this paper, we are only concerned with the IEEE 802.11b network which operates in the 2.4 GHz range.

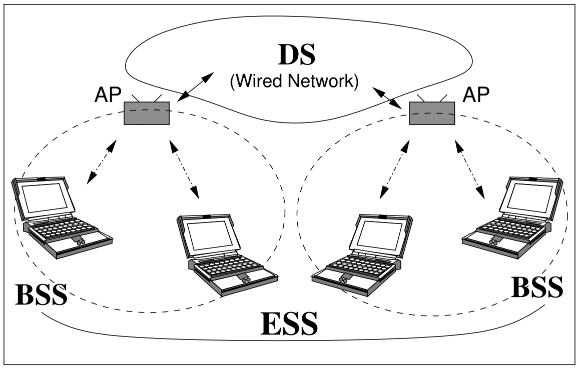

Figure 1: ***The IEEE 802.11 Extended Service Set(ESS)***

The IEEE 802.11 network MAC specification [7] allows for two operating modes namely, the *ad hoc* and the *infrastructure* mode. In the *ad hoc* mode, two or more wireless stations (STAs) recognize each other and establish a peer-to-peer communication without any existing infrastructure, whereas in *infrastructure* mode there is a fixed entity called an *access point* (AP) that bridges all data between the mobile stations *associated* to it. An AP and associated mobile stations form a *Basic Service Set* (BSS) communicating in the unlicensed RF spectrum.

A collection of APs (connected through a distribution system DS) can extend a BSS into an Extended Service Set (ESS refer figure 1).

A *Handoff* occurs when a mobile station moves beyond the radio range of one AP, and enters another BSS (at the MAC layer). During the handoff, management frames are exchanged between the station (STA) and the AP. Also the APs involved may exchange certain context information (credentials) specific to the station via the wired segment or distribution system (DS). Consequently, there is latency involved in the handoff process during which the STA is unable to send or receive traffic.

Because of the mobility-enabling nature of wireless networks, there is an opportunity for many promising multimedia and peer-to-peer applications (such as VoIP [5], 802.11 phones, mobile video conferencing and chat). Also, many believe that WLANs may become or supplement via hot spots the

[*]Portions of this work were sponsored by a National Institute of Standards and Technology Critical Infrastructure Grant, a grant and gift from Samsung Electronics, and a grant from the U.S. Department of Defense.

next generation 4G wireless networks. Unfortunately, the network connection as perceived by the application can suffer from the jittery handoff latencies. As a matter of fact, our measurements not only show that the latencies are very high, but also show that they vary significantly for the same configuration of STAs and APs.

Despite the growing popularity of WLANs, there has been no prior measurement based analysis of the handoff process. There is prior work on performance measurement in ATM-based wireless networks ([15], [11], [18]) and cellular wireless networks ([16]). In [4], Balachandran et. al. present an empirical characterization of user behavior and network performance in a public wireless LAN where they show the varying number of handoffs with time. There has also been work on new handoff schemes in [17], [13],[14] and [10] focusing on reducing WLAN handoff latency, but none of these efforts have measured the current handoff latency.

In this study, we conduct experiments to accurately measure the handoff latency in an in-building wireless network. The measurements are done on three co-existing wireless networks (utilizing APs from different vendors), and using three wireless NICs from different vendors. We analyze the handoff latencies by breaking down the whole process into various phases to assess the contribution of each phase to the handoff latency. Our results show that the *probe* phase is the significant contributor to the handoff latency, and we study the *probe-wait* time which makes up the probe phase and discuss the potential optimizations.

The rest of the paper is organized as follows. Section 2 gives details about the handoff process as specified by the standard. Section 3 explains the methodology used for taking the measurements. We present basic results in section 4. Section 5 is a detailed analysis on the probe phase which is the dominating component of the handoff latency. We conclude the work in section 6.

2. THE HANDOFF PROCESS

The handoff function or process refers to the mechanism or sequence of messages exchanged by access points and a station resulting in a *transfer* of physical layer connectivity and state information from one AP to another with respect to the station in consideration. Thus the handoff is a physical layer function carried out by at least three participating entities, namely the station, a *prior-AP* and a *posterior-AP*. The AP to which the station had physical layer connectivity prior to the handoff is the prior-AP, while the AP to which the station gets connectivity after the handoff is the posterior-AP. The state information that is transferred typically consists of the client credentials (which allow it to gain network access) and some accounting information. This transfer can be achieved by the *Inter Access Point Protocol*(IAPP)[8], or via a proprietary protocol. For an IEEE 802.11 network that has no access control mechanism, there would be a nominal difference between a complete association and a handoff / reassociation. Looking at it another way, the handoff-latency would be strictly greater than *association latency* as there is an additional inter-access point communication delay involved unless a proactive caching technique is used to eliminate the communication [12].

2.1 Logical steps in a handoff

The complete handoff process can be divided into two distinct logical steps:(i) *Discovery* and (ii) *Reauthentication* as described below.

1. Discovery: Attributing to mobility, the *signal strength* and the *signal-to-noise* ratio of the signal from a station's current AP might degrade and cause it to begin to loose connectivity and to initiate a handoff. At this point, the client might not be able to communicate with its current AP. Thus, the client needs to *find* the potential APs (in range) to associate. This is accomplished by a MAC layer function: *scan*. During a scan, the card listens for beacon messages (sent out periodically by APs at the default rate of 10 *ms*), on assigned channels. Thus the station can create a candidate set of APs prioritized by the received signal strength.

There are two methods of scanning defined in the standard : *active* and *passive*. As the names suggest, in the active mode, apart from listening to beacon messages (which is passive), the station sends additional probe broadcast packets on each channel and receives responses from APs. Thus, the station actively probes for the APs.

2. Reauthentication: The station attempts to *reauthenticate* to an AP according to the priority list. The reauthentication process typically involves an authentication and a reassociation to the posterior AP. The reauthentication phase involves the transfer of credentials and other state information from the old-AP. As mentioned earlier, this can be achieved through a protocol such as IAPP [8]. In the experiments detailed in this paper, we do not utilize the standard IAPP communications, but we do permit the proprietary inter-access point communications (between APs of the same vendor). Thus, the authentication phase is just a *null* authentication in our experiments.

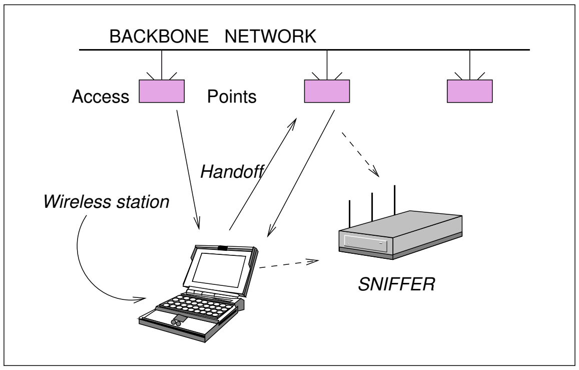

Figure 3: *The Handoff Measurement Setup*

Figure 2 shows the sequence of messages typically observed during a handoff process. The handoff process starts with the first probe request message and ends with a reassociation response message from an AP. We divide the entire handoff latency into three delays which we detail below.

1. **Probe Delay:** Messages A to E are the probe messages from an active scan. Consequently, we call the

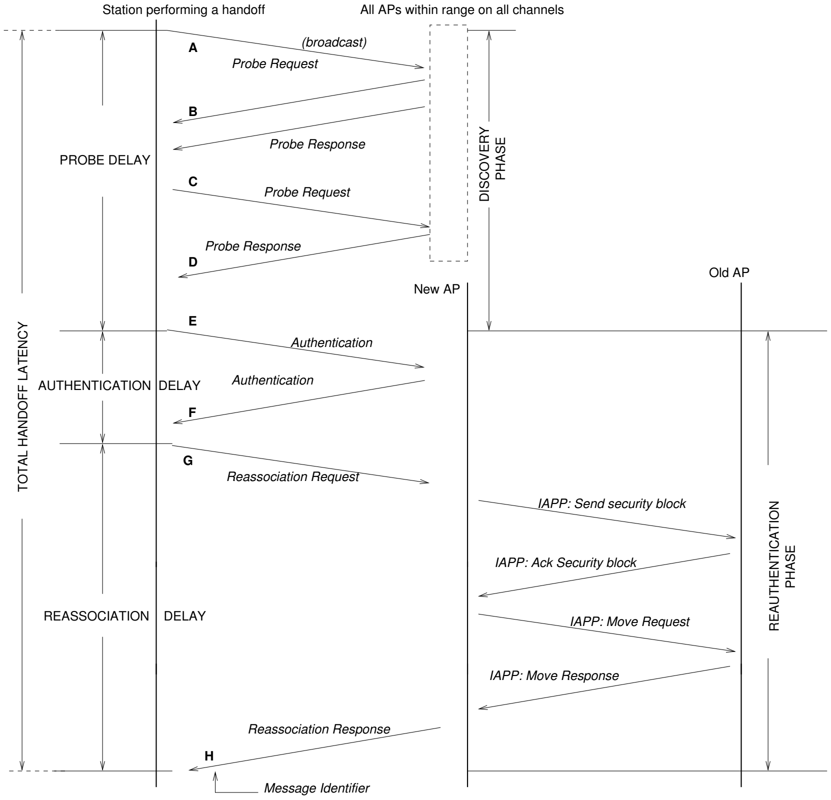

Figure 2: ***The IEEE 802.11 Handoff Procedure (followed by most cards)***

latency for this process, *probe delay.* The actual number of messages during the probe process may vary from 3 to 11.

2. **Authentication Delay:** This is the latency incurred during the exchange of the authentication frames (messages E and F). Authentication consists of two or four consecutive frames depending on the authentication method used by the AP. Some wireless NICs try to initiate reassociation prior to authentication, which introduces an additional delay in the handoff process and is also a violation of the IEEE 802.11 [7] state machine.

3. **Reassociation Delay:** This is the latency incurred during the exchange of the reassociation frames (messages G and H). Upon successful authentication process, the station sends a *reassociation request* frame to the AP and receives a *reassociation response frame* and completes the handoff. Future implementations may also include additional IAPP messages during this phase which will further increase the reassociation delay.

As a note, according to our analysis presented above, the messages during the probe delay form the discovery phase, while the authentication and reassociation delay form the reauthentication phase. Apart from the latencies discussed above, there will potentially be a *bridging* delay caused by the time taken for the MAC address updates (using the *IEEE 802.1d* protocol) to the ethernet switches forming the distribution system (the backbone ethernet). The results in our experiments will not reflect this latency. In the next section we describe the details of the experiment.

3. DESIGN OF THE EXPERIMENTS

As mentioned earlier, the experimental setup consisted of three in-building wireless networks, a mobile wireless client, and a mobile sniffing system. As shown in figure 3 , the basic methodology behind the experiments, is to use the sniffer (in *close* proximity to the client) to capture all packets of interest related to the client for the analysis. This section describes the wireless networks, the sniffing system, and the client setup in detail.

3.1 The Wireless Network Environment

All the experiments were done in the A.V. Williams Building at the University of Maryland, College Park campus. The building hosts three co-existing wireless networks namely *cswireless*, *umd* and *nist* spanning four floors in all. The three networks have overlapping coverage, with *umd* covering the whole building, *nist* having half the coverage of *umd*, and *cswireless* covering one floor of the building (which has four floors). The three networks are described below :

1. The *umd* network: This network has 35 Cisco 350 APs distributed over four floors. This network uses open authentication. The gateway does a MAC address based access control for the data packets, and

this does not have any effect on the MAC layer handoff process. The APs are configured only on channels 1, 6 and 11.

2. The *nist* network: This network has 17 APs distributed over half the building. The APs are built using a Soekris board [3], each using a Demarctech Prism 2.5 200mW wireless card, running OpenBSD 3.1 and using the *hostap* driver [1] for the AP functionality on the wireless interface. This network uses open authentication and no access control. The APs are configured only on channels 1, 6 and 11.

3. The *cswireless* network: This network has 8 Lucent APs in total. It uses a static shared key for WEP encryption. The APs are present on 8 different channels.

3.1.1 Methodology of each Experiment

The experiments were done in the following manner. A person with the mobile station walks through the building following a fixed path of travel (to minimize effects from the layout of APs) during each run. The duration of the walk, which is the duration of a single run of the experiment is approximately 30 minutes. Each experiment is characterized by the (i) Wireless NIC used at the mobile station and (ii) the Wireless network used. The mobile client sends negligible periodic ICMP messages to the network to maintain and display connectivity. Thus as the station moves, it performs handoffs as it leaves a BSS and enters another.

During the experiment on one wireless network, the other two coexisting networks were shutdown *i.e.*, the environment had absolutely no RF apart from the entities taking part in the experiment. Also the experiments were done when there was negligible user activity (during the early morning hours of weekends). This was done in order to minimize the effects of channel contention on the various latencies measured. Thus, the results in this work reflect zero contention, and we reason that channel contention will only worsen the handoff latencies.

The mobile station is accompanied by a sniffing system which is designed (as discussed in section 3.2.3) to capture packets of 'interest' i.e. the management frames constituting the handoff process. The sniffer is always in *close proximity* to the client, *i.e.* as close as physically possible and also moves with the station during the experiment. This helps in validating the sniffing system in the following manner :

1. Since the sniffer and station are in close proximity, any packets received by the client, will also (accuracy measured in section 3.2.1) be captured by the sniffer.

2. Frames sent by the station on a neighboring channel with respect to the sniffer, will be captured with relatively high probability (see section 3.2.1).

3.2 The sniffing process

In this section, we describe the sniffing system used with each network, the accuracy and the limitations.

3.2.1 Capturing packets on one channel

The wireless NICs based on the *Prism 2.5* chipset from Intersil [1] have a monitor mode in which the NIC captures all traffic (management and data) on one particular channel and passes it to the driver. The wlan-ng linux driver [2] provides the functionality to capture the traffic, the *ethereal* sniffing program was used to capture and filter the traffic. A laptop with a PCMCIA prism 2.5 based wireless NIC was used to sniff one channel.

We measured the accuracy of this setup by having a source machine send sequenced UDP packets over a wireless network to a sink machine on the wired segment. The experiment was done in an RF free environment (i.e. no other STAs or APs were present in the RF medium). The sniffer was placed in close proximity to the source to reflect the sniffing setup in our later experiments. We observed a loss percentage of 0.2% averaged over five experiments, each sending 1032 packets in an interval of 10 seconds.

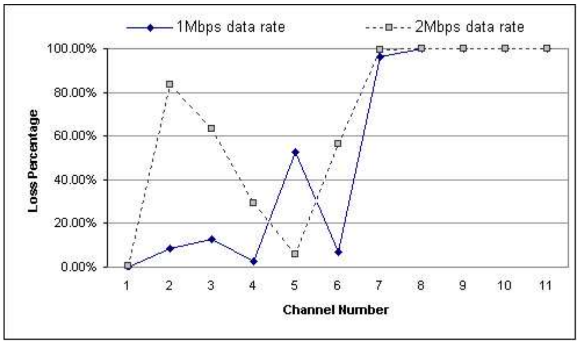

Figure 4: *The loss percentage of the sniffer on neighboring channels. The traffic was sent on channel one.*

3.2.2 Capturing packets on neighboring channels

Depending on the data rate, packets sent on one channel can be captured by the above sniffer, while on a neighboring channel. We performed an experiment to empirically observe the accuracy of sniffing traffic on neighboring channels.

The experiment consists of a source machine transmitting sequenced UDP packets on the wireless network on channel 1. The sniffer is progressively moved from channel 1 through 11. Figure 4 shows the loss ratio for various data rates. As can be seen, for packets sent at a data rate of 1Mbps, the sniffer can capture packets up to three neighboring channels (in either direction) with a maximum loss of 12% [1]. As a note, handoffs with missing packets (which can be detected using missing sequence numbers), were not taken into account during the analysis in this work. Hence although the loss effects the number of handoffs that can be studied, it does not compromise the accuracy of the latency measurements.

We reason to our best knowledge that because of poor *selectivity* of the radios used in the wireless NICs employed for

[1] At 5.5Mbps and 11Mbps, no traffic was observed on any neighboring channel.

sniffing, a strong signal on an adjacent channel is treated as a weak signal on the sniffing channel (since the transmitting client is always in close proximity to the receiver). This phenomenon, also known as *adjacent channel interference* [6], is being exploited by our sniffing mechanism to capture packets transmitted by the client on adjacent channels.

3.2.3 Design of the sniffer system

Based on the above observations we design the sniffing system in the following manner. The frames of interest are the *Probe Requests* and *Responses*, the *Reassociation* and the *Authentication* frames. These frames are sent at the lowest data rate allowed i.e. 1Mbps (for maximum range and compatibility).

1. For frames sent by the STA (at 1Mbps), the sniffer (which is in close proximity) has to be capturing packets in a neighboring channel (or on the same channel), as discussed in section 3.2.2.
2. For frames sent by an AP on a particular channel, the sniffer has to be capturing packets on the same channel for high accuracy. We require this constraint because the AP is *not* in close proximity to the sniffer.

Based on the above principles, we designed the sniffing system for the three networks in the following manner:

1. For *umd* and *nist* networks: Here the APs are on channels 1, 6 and 11. Thus the sniffer needs one NIC capturing packets on each of these channels. We use two Linux machines, one with one wireless NIC and the other with two wireless NICs (PCMCIA based) which sniff the three channels independently. The captured data is then merged using the timestamp on each packet. To minimize the inaccuracy caused by the inconsistencies of the system clock in the two machines, they were synchronized using the *Network Time Protocol* (NTP) through a point-point ethernet connection between the machines. Throughout the experiment, we maintained a clock accuracy of 80 μs or better between the machines (an error of less than 0.08% for latency of 1 ms). These Linux machines we used were IBM ThinkPad laptops with Pentium III 866 MHz and 256 MB RAM. The software for sniffing on each NIC was as discussed in section 3.2.1.
2. For the *cswireless* network we used six linux machines, sniffing all eleven channels. Five machines had two wireless NICs and one had a single wireless NIC. The machines were synchronized using NTP in a similar manner. The traces were combined and the duplicate packets were removed using the 802.11 sequence numbers.

3.3 The clients

The wireless NICs studied for the handoff process were from three different vendors, namely, Lucent Orinoco, Cisco 340, and ZoomAir prism 2.5[2]. The mobile station performing the handoff was an IBM Thinkpad T30 with Pentium IV and 512 MB RAM. The machine was running RedHat Linux 8.0 as the operating system.

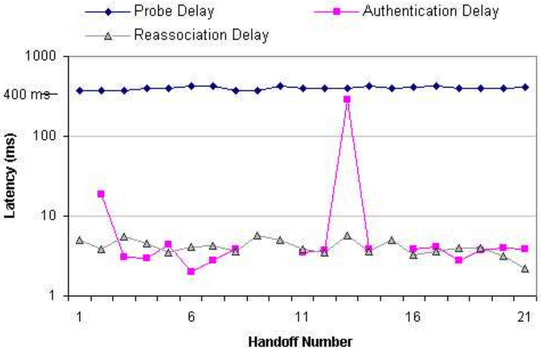

Figure 5: ***Handoff Latencies - Cisco 340 STA on* umd *(Cisco AP) network. Zero values are not plotted on the log-scale.***

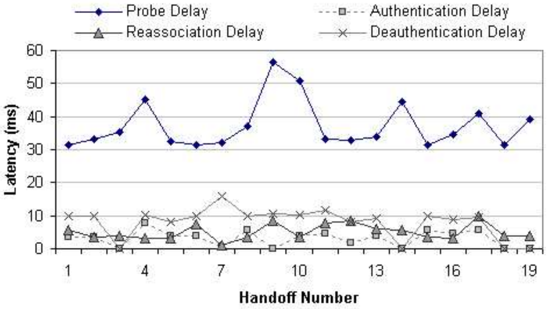

Figure 6: ***Handoff Latencies - Lucent STA on* umd *(Cisco AP) network.***

4. EXPERIMENT RESULTS

Three wireless NICs and three different networks gave nine experiments to run. Each experiment is characterized by the wireless NIC and wireless network being used. The experiments were performed as discussed in section 3.1.

As a representative set, figures 5,6 and 7 show the raw-breakup of the handoff latencies on the *umd* network. Figure 5 shows the handoff latencies for the Cisco 340 STA, figure 6 shows the latencies for the Lucent STA and figure 7 shows the latencies for the ZoomAir STA. Figures 6 and 7 show a fourth delay, namely, *deauthentication* delay which comes into picture because of a different sequence of handoff messages followed by these NICs (figure 10). This is further discussed below.

Figure 8 shows the average values of the total handoff latency for the nine experiments along with the standard deviation (shown graphically on the bars). Figure 9 shows the average values of the four delays in the nine experiments.

Based on these results, the following direct conclusions can be drawn.

[2] The secondary firmware versions on these NICs were as follows : Lucent Orinoco – 7.28.1, Cisco 340 – 4.25.10 and ZoomAir – 0.8.3.

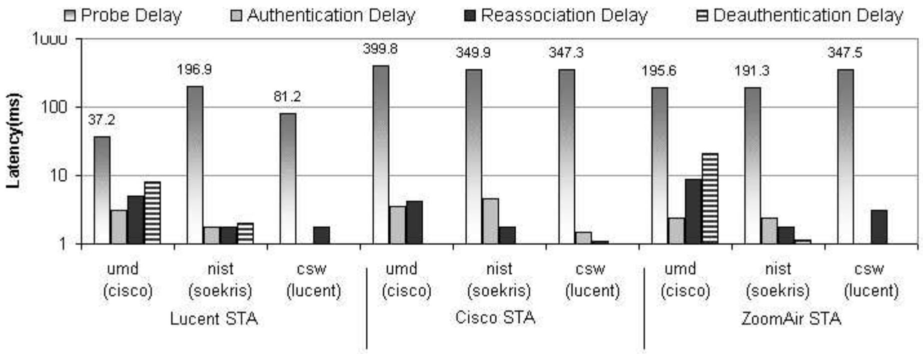

Figure 9: ***Handoff Latency Breakup - Comparison of the nine experiments.***

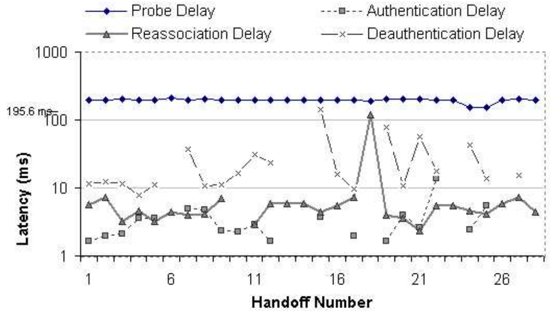

Figure 7: ***Handoff Latencies - ZoomAir Prism 2.5 NIC on*** **umd** ***(Cisco AP) network. Zero values are not plotted on the log-scale.***

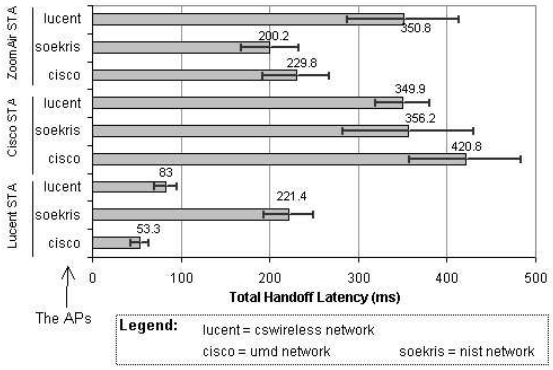

Figure 8: ***Handoff Latencies - Average values and standard deviation shown for all nine experiments.***

1. **Probe delay is the dominating component:** From figure 9 it is clear that the probe delay accounts for more than 90% of the overall handoff delay, regardless of the particular STA, AP combination. Also even in the number of messages exchanged between the STA and the APs involved, the probe phase accounts for more than 80% of these in all cases. Thus any handoff scheme that uses techniques/heuristics that either cache or deduce AP information without having to actually perform a complete active scan clearly stand to significantly improve the handoff process.

2. **The wireless hardware used (AP,STA) affects the handoff latency:** We can infer this by observing two facts. Firstly, keeping the AP fixed, we can see that the client wireless card affects the latency. Figure 8 compares the average values of the latency among all nine configurations. Keeping the AP fixed, we can see a maximum average difference of 367.5 *ms* (Lucent STA and Cisco STA with Cisco AP). This is a huge variation by just changing the client card being used. Secondly, keeping the client card fixed, the AP also affects the latency but to a much lower extent (around 60% less): The maximum average difference (between the two APs for any fixed client) is 150.2 *ms* (Lucent AP vs Cisco AP for ZoomAir STA). This supports our previous result that the probe function is the dominating component since it is a firmware function (and implemented differently by different vendors) of the NIC cards.

3. **There are large variations in the handoff latency:** Apart from the variations in the latency with different configurations, we find significant variations in the latency from one handoff to another within the same configuration. This is also supported by the high standard deviations (figure 8). Cisco STA on Cisco AP (*umd* network) has the largest standard deviation of $63.2ms$. Also, we observe that the larger the handoff latency, the higher the variation.

4. **Different wireless cards follow different sequence of messages:** This is an observation from looking at the traces offline. We found that the ZoomAir and the Lucent NICs follow a slightly different procedure from the Cisco NIC, as shown in figure 10. The figure shows that the card sends a *reassociate* message prior to *authentication* which it performed when the AP sends a *deauthentication* message. The figure also shows the modified semantics of the *reassociation delay* and the *authentication delay* for the ZoomAir cards. In order to attribute the delay caused by this modified sequence, we call the latency between the first

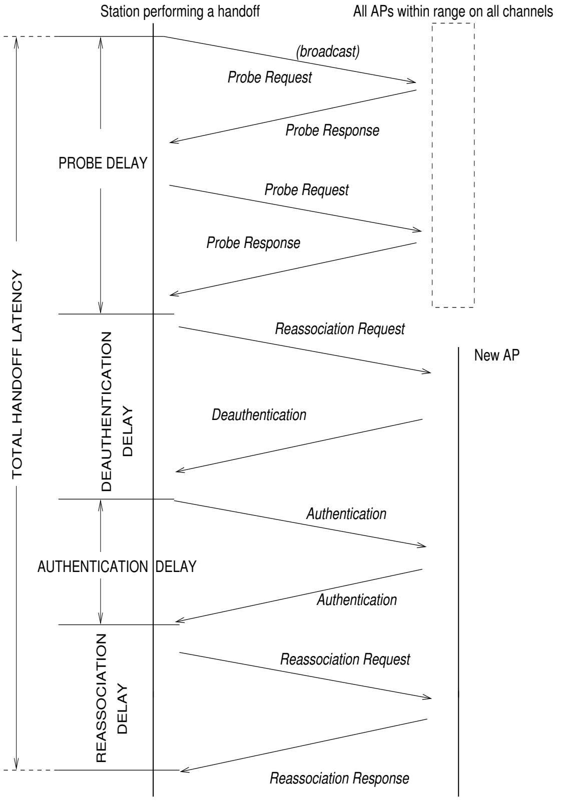

Figure 10: *The Handoff Procedure as observed on the Lucent and ZoomAir wireless NICs.*

reassociation and the first *authentication* message as the *deauthentication* delay. This latency includes the *deauthentication* message as shown in figure 10.

Thus the probe delay is accountable for the high handoff latency and also the variations in some cases. We present a detailed analysis of this phase based on the traces collected in the above experiments.

5. ANALYSIS OF THE PROBE PHASE

In this section, we present a detailed analysis of the probe phase based on the experiment data. Presented first is the specification of the active scan algorithm from the standard ([7]), a discussion of the observations, and suggestions for improvement of the probe latencies.

5.1 The Probe Function Specification

The probe function is the IEEE 802.11 MAC active scan function and the standard specifies a scanning procedure as follows (modified for brevity):

For each channel to be scanned,

1. Send a *probe request* with broadcast destination, desired SSID, and broadcast BSSID.
2. Start a *ProbeTimer* (a timer used in step 3).

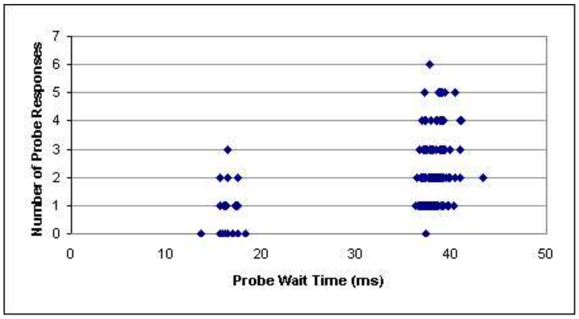

Figure 11: *The distribution of the probe-wait times with respect to the number of probe responses received for the Cisco STA.*

3. If medium is *not busy* before the ProbeTimer reaches *MinChannelTime*, scan the next channel, else when ProbeTimer reaches *MaxChannelTime*, process all received *probe responses* and proceed to next channel.

As can be seen from the algorithm, *MinChannelTime* and *MaxChannelTime* are two parameters that determine the duration of scan for each channel. Figure 12 shows the messages in a probe phase. The STA transmits a probe request message and waits for responses from APs on each channel. Let *Probe-Wait latency* be the time an STA waits on one particular channel after sending the probe request. We measure this as the time difference between subsequent probe request messages. Thus the STA waits on one channel for *MinChannelTime*, and if any traffic (data or management frames) was observed or a probe response was received, the STA further extends the probe-wait period to *MaxChannelTime*. Thus according to the above procedure, the traffic on the channel and the timing of probe response messages affects the probe-wait time, i.e. the probe-wait time should be expected to be distributed between a *MinChannelTime* and a *MaxChannelTime* value. Hence the total probe delay, say t, for probing N channels, would be bounded by:

$$N * MinChannelTime \leq t \leq N * MaxChannelTime$$

In the next subsection, we present the empirical observations on the probe-wait time. We contrast this with the expected behavior according to the standard.

5.2 The Probe-Wait time: Observations

In this section, we discuss the probe-wait times for the three wireless NICs under study over the *umd* network.

1. *Cisco 340 STA:* Figure 11 shows the various probe-wait times with respect to the number of probe response messages received by the Cisco STA on the *umd* (Cisco AP) network. The scatter-plot shows two clusters being formed, which more-or-less correspond to the *MinChannelTime* and *MaxChannelTime* values from the active scan algorithm. When no probe responses are received, (and the channel has no traffic, probably

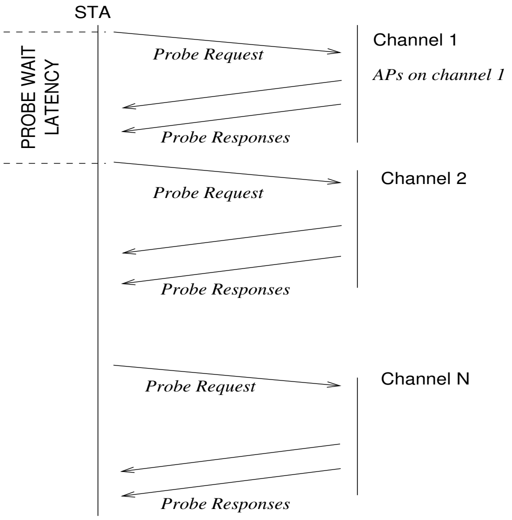

Figure 12: ***The messages in an active scan.***

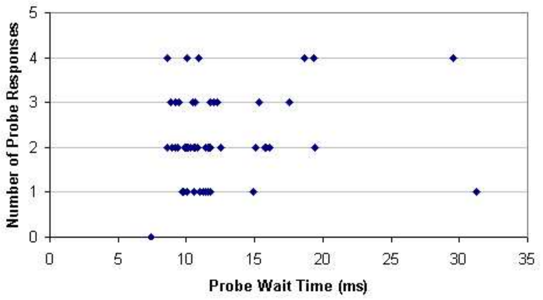

Figure 13: ***The distribution of the probe-wait times with respect to the number of probe responses received for the Lucent STA.***

since there were no APs present) the probe-wait time is equal to the *MinChannelTime* which is around $17ms$ for the Cisco NICs. When there are responses (or traffic) on the channel, the NIC spends *MaxChannelTime* on the channel which is around $38ms$. The Cisco STA sends 11 probe requests in all, one on each channel.

2. *Lucent STA:* Figure 13 shows the distribution for the Lucent STA on the *umd* network. Here the probe-wait times do not have significant correlation with the number of probe responses. Also the Lucent STA sends only 3 probe requests, one each on channels 1, 6 and 11. The probe requests are sent at 1Mbps, and can be received by the APs on the neighboring channels. Also the variation is not much, having a standard deviation of $4.2ms$.

3. *ZoomAir STA:* Figure 14 shows the probe-wait times for the ZoomAir STA on the *umd* network. Like the Lucent STA, ZoomAir also sends only 3 probe requests on channels 1, 6 and 11. The probe-wait times for the first two requests cluster around $63ms$ while the times for the third probe-wait clusters around $73ms$. The third (i.e. the last) probe wait time is measured as the difference between the last probe request and the reassociation request frame sent by the STA. Thus the additional $10ms$ (on average) potentially goes into the processing of the probe results, making a decision about the AP to reassociate to and sending the reassociation request. However, we did not observe a similar difference in the probe-wait times for the other STAs.

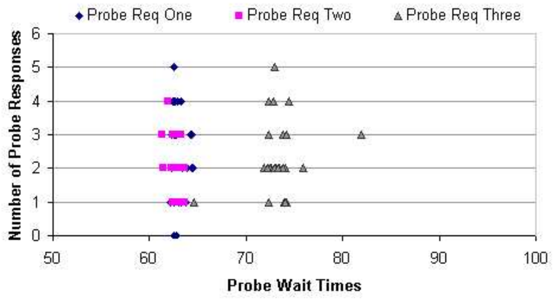

Figure 14: ***The distribution of the probe-wait times with respect to the number of probe responses received for the ZoomAir STA.***

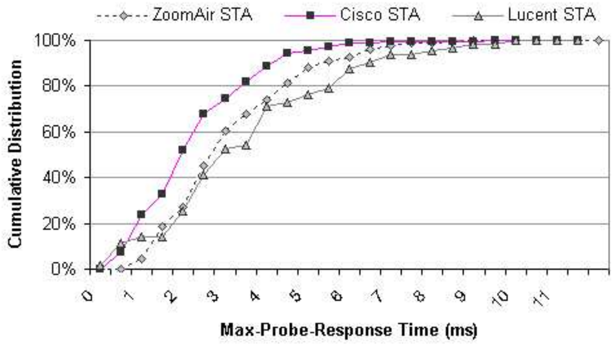

Figure 15: ***Cumulative distribution of the maximum probe response times observed by the three wireless NICs under study on the* umd *network.***

From the above analysis, it is clear that vendors implement different probing methods which reflect the large variation in the probe delays from one STA to another.

5.3 Probe-Wait Optimizations

In this section, we discuss some simple optimizations on the probe-wait time based on the observations. In particular, we analyze the probe responses to determine a *good* value for the *MinChannelTime* and *MaxChannelTime* parameters which effect the probe-wait time.

Let AP ap_i be on channel L and assume that STA C is performing a probe. We define the *probe response time* from ap_i to be the time between the probe request message sent by C on L and the corresponding probe response sent by the AP

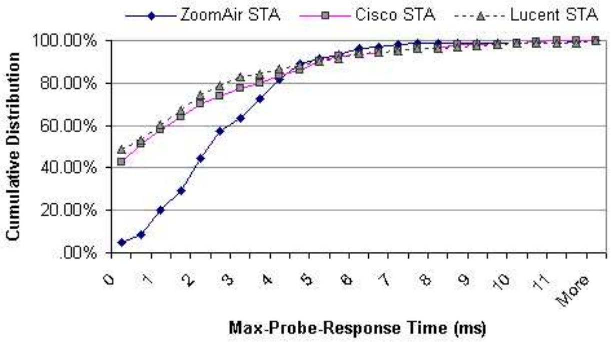

Figure 16: ***Cumulative distribution of the maximum probe response times observed by the three wireless NICs under study on the* cswireless *network.***

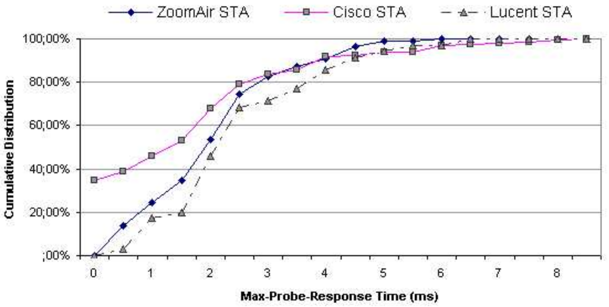

Figure 17: ***Cumulative distribution of the maximum probe response times observed by the three wireless NICs under study on the* nist *network.***

ap_i. The *maximum* probe response time is the time between the probe request and the last probe response received by the STA C from any AP on channel L. Ideally the probe-wait time on every channel should be no larger than the *maximum* probe response time on that channel.

Figure 15 shows the cumulative distribution function of the probe response times for all three wireless NICs (on the *umd* network). From the graph it can be seen that all probe responses are received by a station within (approx.) $11ms$ for the *umd* network. Also in 90% of the cases all probe responses are received within (approx.) $6.5ms$. Hence a direct conclusion is that a *MinChannelTime* of around $6.5ms$ would a very good indicator of the presence of APs on the channel. And a *MaxChannelTime* of around $11ms$ would be sufficient to capture all the probe responses. This optimization can bring about a drastic reduction in the overall handoff latency. Even using a pessimistic probe-wait time of $11ms$, brings the handoff latency for the Cisco STA to around $121ms$ for the 11 probe requests (from an average of $399.8ms$, a reduction of around 70%), for the Lucent and ZoomAir to $33ms$ for the 3 probe requests (reduction of around 12% for Lucent and around 83.2% for ZoomAir).

Figures 16 and 17 show the same distribution calculated for the experiments done on the *cswireless* and the *nist* networks respectively. For the *nist* network, all responses are received within $9ms$, while for *cswireless* the value is around $15ms$. Using these pessimistic estimates (*MaxChannelTimes*), figure 18 shows the expected probe-wait time improvements for the nine scenarios.

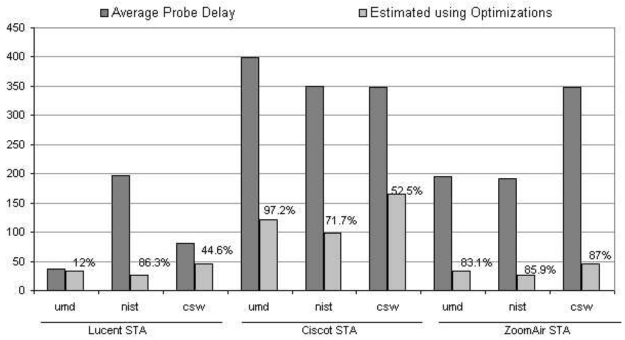

Figure 18: ***The average probe delay values, and the estimated probe delay values based on the pessimistic calculation of the* MaxChannelTime *for the nine scenarios. Also shown is the percentage improvement next to the estimated probe delay values.***

5.4 Hints for Fast-Handoff Strategies

Based on the observations from the previous sections, we present some simple heuristics to improve the probe-wait latency and also the overall handoff latency.

5.4.1 Reducing the Probe-Wait Latency

Based on observations from section 5.3, we have the following methods to improve the probe-wait latency. These heuristics basically estimate a better fit for the *MinChannelTime* and *MaxChannelTime* parameters, thereby improving the probe-wait time:

1. An offline empirical analysis of the network as done in this work, can help come up with a good static value for these parameters which could be broadcast in the beacon messages or probe response messages from the APs.
2. The Average AP density (i.e. number of APs *visible* per channel) could be broadcast in the beacon or probe response messages. The STA can wait for the corresponding number of probe responses and decide to switch to the next channel.
3. The STA could wait *sufficiently* long on one channel for the last probe response and empirically learn the *maximum* probe response time. It can thus dynamically refine the *probe wait* time accordingly.

5.4.2 Other Optimizations

Handoff heuristics that require the least number of active scans will tend to perform the best. The following methods (or a combination of them) might be used to design heuristics and these are all attempts to avoid an active scan:

1. Using a distributed datastructure : *Neighbor Graphs* (refer [12]). The AP-neighborhood information is captured as a datastructure stored in the APs in a distributed manner. Each AP maintains a list of its neighbors, and using this information the STA performing the handoff can proactively determine its next AP instead of performing the scan process. Since this information it not dynamic (i.e. the AP topology does not change rapidly), it can bring about significant improvements in the overall handoff latency by potentially eliminating the probe phase completely.

2. Query an external agent that provides *hints* on the potential next APs and their channels i.e a *map* of the APs based on the location. Pack et. al. in [14], [13] propose a technique in this category.

3. Interleave scan messages with data during normal connectivity and use that information to perform a partial active scan (or no scan at all) during the handoff. Also passive scanning (listening for beacon messages) might be performed during normal connectivity to build up the list of APs.

4. Since the probe-wait time depends on the number of probe responses, another strategy might be to create an *ordering* among the APs such that a single AP or a small set of APs is responsible for probe requests (i.e. the number of probe responses is a constant).

6. CONCLUSION AND FUTURE WORK

The primary contribution of this work is a detailed analysis of the handoff process, the factors that bring about the high latency and the variation and the various messages/steps involved. We find that out of the three basic functions (probe, authentication and reassociation), carried out by the STA, the probe phase has the dominant latency regardless of the AP-STA being used.

We also present a detailed analysis of the probe phase, and account for the large variation to the *probe-wait* time which essentially depends on the particular heuristic employed by the wireless client NIC being used.

We present a detailed analysis of the probe phase, and in particular suggest optimizations for the two parameters *MinChannelTime* and *MaxChannelTime* that have a significant effect on the overall handoff latency. We also provide some ideas on fast-handoff strategies which aim to eliminate the delay with high probability.

In our experiments we used wireless PC cards from three vendors, namely Lucent Orinoco, Cisco Aironet, and ZoomAir and the APs from Lucent, Cisco and Demarctech. This provided enough diversity in our experiments, and we find that there is large variation in the latency with the particular AP-STA hardware being used. Also we find that the sequence of messages exchanged during the handoff process can also differ with the STA being used.

One of the more interesting results of our work is that current WLAN equipment will not meet the expectations (replacing or augment 4G systems) that many have. This is because the handoff latencies we measured far exceed guidelines for jitter in voice over IP (VoIP) applications where the overall latency is recommended not to exceed 50ms [9].

In the future, we plan to investigate methods to add a robust authentication mechanism to WLAN handoffs and reduce the overall latency of the handoff within acceptable bounds for VoIP applications.

7. REFERENCES

[1] Host AP driver for Intersil Prism Cards. URL: http://hostap.epitest.fi.

[2] Linux driver for Prism based wireless cards. URL: http://www.linux-wlan.com/linux-wlan/.

[3] OpenBSD based access points using the Soekris Boards. URL: http://www.missl.cs.umd.edu/wireless/testbed/.

[4] A. Balachandran, G. Voelker, P. Bahl, and P. Rangan. Characterizing User Behavior and Network Performance in a Public Wireless LAN. In *Proceedings of ACM SIGMETRICS'02*, June 2002.

[5] R. Caceres and V. N. Padmanabhan. Fast and Scalable Wireless Handoffs in Support of Mobile Internet Audio. *Mobile Networks and Applications*, 3(4):180–188, December 1998.

[6] C. Chien. *Digital Radio Systems on a Chip*. Kluwer Academic Publishers, 2001.

[7] IEEE. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Standard 802.11*, 1999.

[8] IEEE. Recommended Practice for Multi-Vendor Access Point Interoperability via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation. *IEEE Draft 802.1f/D3*, January 2002.

[9] International Telecommunication Union. General Characteristics of International Telephone Connections and International Telephone Circuits. ITU-TG.114, 1988.

[10] R. Koodli and C. Perkins. Fast Handover and Context Relocation in Mobile Networks. *ACM SIGCOMM Computer Communication Review*, 31(5), October 2001.

[11] U. R. Krieger and M. Savoric. Performance Evaluation Of Handover Protocols For Data Communication In A Wireless ATM Network. In *Proceedings of ITC 16*, June 1999.

[12] A. Mishra, M. Shin, and W. Arbaugh. Context Caching using Neighbor Graphs for Fast Handoffs in a Wireless Network. *Computer Science Technical Report CS-TR-4477, University of Maryland*, 2003.

[13] S. Pack and Y. Choi. Fast Inter-AP Handoff using Predictive-Authentication Scheme in a Public Wireless LAN. *IEEE Networks 2002 (To Appear)*, August 2002.

[14] S. Pack and Y. Choi. Pre-Authenticated Fast Handoff in a Public Wireless LAN based on IEEE 802.1x Model. *IFIP TC6 Personal Wireless Communications 2002 (To Appear)*, October 2002.

[15] R. Ramjee, T. F. L. Porta, J. Kurose, and D. Towsley. Performance evaluation of connection rerouting schemes for ATM-based wireless networks. *IEEE/ACM Transactions on Networking*, 6(3):249–261, 1998.

[16] H. B. Srinivasan Seshan and R. H. Katz. Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience. 4:141–162, 1997.

[17] C. L. Tan, K. M. Lye, and S. Pink. A Fast Handoff Scheme for Wireless Networks. In *WOWMOM*, pages 83–90, 1999.

[18] C.-K. Toh. Implementation and Evaluation of a Mobile Handover Protocol in Fairisle. 1995. http://www.cl.cam.ac.uk/Research/SRG/greenbook.html.