An experimental study on Layer 2 roaming for 802.11 based WLANs

Kevin Schneider, Damla Turgut, and Mainak Chatterjee
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2362
{kschneider,turgut,mainak}@eecs.ucf.edu

Abstract

In this paper, we argue that the roaming functionalities implemented at the data link layer (Layer 2) have certain drawbacks causing delay in the handoff process for mobile hosts in a wireless local area network. Through real experiments, we show how the triggering of the roam process is delayed which affects data transmission. Even though the network interface cards and drivers were from different manufactures, the decision process related to the handoffs had significant similarities – the use of a crude timer and an un-sophisticated approach for detecting error conditions. Identifying the mechanisms that cause a mobile host to roam, we suggest a pro-active algorithm that eases the effects of waiting too long to roam. By monitoring the current received signal strength, we were able to initiate the probing process early enough which in turn helped the roaming process.

1 Introduction

In recent years, wireless extensions to local area networks capable of supporting high bandwidth applications have emerged as a competitive technology for users. Furthermore, users have developed a strong desire for connecting to the Internet wirelessly. This is evident from the fact that a large number of so called *hotspots* are becoming omni-present. Wireless LANs (WLANs) though not in their infancy, are not yet a mature network that can provide reliable data delivery with strict quality of service (QoS) requirements. Although there has been tremendous progress made, it would appear under close inspection that aspects related to supporting mobility of mobile clients (nodes) have not received as much attention. There are several challenges that are being addressed that try to provide seamless mobility and still maintain the desired QoS. For seamless mobility, it is not only necessary that there be efficient handoff management protocols but also that the response time of the protocols is as low as possible. This becomes more important when the service in consideration is real-time such as VoIP.

Although 802.11 based wireless LANs are frequently used for their portability aspects in locations such as waiting areas and conference rooms, their mobility aspects are just now beginning to be put into service with such applications as voice over IP and streaming media. Unlike portability, where a device is parked in a temporary location while its being used; true mobility requires the device to work while moving from location to location. This constant data flow while frequently moving requires the active station to transition among many access points (APs). Data flow during the transition process is likely to undergo some disruption during this handoff period. This disruption, which can manifest itself as packet latency, jitter or retransmissions, is very much dependent upon when and how the individual station roams from AP to AP. The decision to roam is a function of the client station.

The main issue in wireless LAN handoff management is the process through which a mobile client (node) moves from one AP to another, i.e., detaches itself from one and attaches to another while still in session. As a moving node may need to change the associated AP, the APs must first be identified and the target AP must be selected. When this process is finished, the process of connection establishment begins. The whole handoff procedure can be broadly divided into three distinct logical phases: scanning, authentication, and re-association. During the first phase, a mobile node scans for the APs by either sending probe request messages (also called active scanning) or by listening for beacon messages (also called passive scanning). After scanning all channels, an AP is selected by the mobile node based on parameters such as the received signal strength indicator (RSSI) and link quality. Then, the mobile node exchanges authentication messages with the selected AP. Finally, if the AP authenticates the mobile node, an association moves from an old AP to a new AP. The delay incurred during these three phases is referred as the Layer 2 (L2) handoff

1-4244-0992-6/07/$25.00 ©2007 IEEE

delay. This delay consists of probe delay, authentication delay, and re-association delay. Mishra *et al.* [3] showed that probe delay is the most dominant among the three delays. Thus, to solve the problem of the L2 handoff delay, the scanning delay has to be minimized.

Currently, much of the handoff process, particularly the decisions of when and where to initiate a handoff, is left to the mobile client. It is generally assumed that the client, which is comprised of a wireless network interface card and associated drivers, will monitor its RSSI and/or signal to noise ratio (SNR) and roam to a different AP when it starts to see poor performance. The motivation for this experimental research is the lack of information about the mechanisms used in the probing and roaming process which results in loss of connection for prolonged duration when mobile clients (e.g., laptops) move within a building with substantial WLAN coverage. It was observed that the clients would hang or simply lose connection to the network for several seconds. The study in [3] revealed that clients with different wireless network interface cards (WNICs) suffered problems which were kind of unique to the the WNIC being used. The study also uncovered the wide variation in the roaming process, and the overall latency of the handoff. Their conclusions were that the handoff process frequently exceeded 100 ms and were of the order of seconds, and no data could traverse the WNIC while the client was looking for its new AP and trying to get associated with it. A good survey of the decision making processes employed by the different vendors and there relative pros and cons can be found in [1].

In this paper, we examine the mechanisms that cause a client to roam and propose a better decision making process. We propose a pro-active algorithm that is based on empirical analysis of common coverage patterns; and an improvement on the existing timer that takes into account the typical mobile velocity of a user. We conducted laboratory experiments with laptops that had the WNICs from different manufactures. It was found through testing that the method used by several manufacturers for roaming was a crude timer, or some type of error condition such as dropped packets. It was also shown that with a small amount of pro-active monitoring of the current RSSI, a better decision could be made with regards to probing and subsequent roaming.

The rest of the paper is organized as follows. In section 2, we discuss the handoff process. The experimental setup and certain shortcomings of L2 roaming are explained in section 3. We propose a pro-active roaming algorithm and demonstrate the improvements in latency in section 4. Conclusions are drawn in the last section.

2 The Roaming (Handoff) Process

The WLAN roaming process operates at Layer 2 (data link) of the OSI protocol stack. At this layer, the medium access protocols allow packets to traverse from clients to APs. As defined by the IEEE 802.11 standard, a client can only be connected to a single AP at any given time. This connection, commonly called an *association*, is what establishes the link layer connection from the client to the network and allows the AP to forward packets on behalf of the client. This forwarding can be to another wireless client connected to the same AP, or to an altogether different type of client that resides on the back-end wired network. The specification requires that all communications through the AP require an association; however some communications to an AP by a client may be done without an association. This *unassociated* traffic is how a client probes other APs for there existence, and performs authentication and association. Thus, probe packets are essentially broadcast messages and need no prior association.

Outside of the roaming process itself, the network forwarding agents, i.e., the APs must recognize that a client has moved to a different part of the system and forward packets to the new location as necessary. This mechanism which involves possible AP to AP communication and changing of L2 bridge tables throughout the infrastructure is outside the scope of this paper, despite its obvious impact on data flow. On a similar note, although it is entirely possible for a client to roam from an AP on one IP network to an AP on a different IP network, the mechanisms for subnet roaming (i.e., Layer 3 roaming) are also beyond the scope of this paper. The simple L2 association between an AP and a client is the underlying foundation for all other L2 and higher transitions.

Prior to association, a client must authenticate with the AP via appropriate authentication protocols such as WEP [2], WPA [4], LEAP [6] and so on. Although this requirement could easily pose an obvious bottleneck to a timely client-AP association, the vast array of authentication schemes are a subject of investigations by themselves. Hence, we adopt the simplest and fastest method with an open authentication for the experiments. We use an open authentication, which is essentially a simple handshaking process between the client and AP. A state diagram with the authentication and association states is shown in Figure 1.

3 Experimental Study

3.1 Experimental Setup

For experimentations, we setup laptops on a mobile cart as shown in Figure 2. The cart was pushed through a building containing the two APs at a fairly slow rate which gave

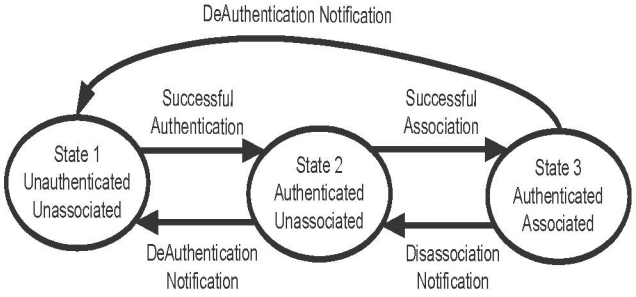

Figure 1. State diagram

the most optimal roaming conditions. The analyzers were laptops that had AiroPeek version 2.0.2 from Wildpackets Inc [8]. The analyzer software uses common network interface cards and special drivers are required to put the cards in promiscuous mode and capture packets.

Figure 2. Experimental setup

The roaming process consisted of moving a client between two APs on separate non-overlapping channels, and observing the packets using separate protocol analyzers. The client in this case was one of the few laptops. The laptops were equipped with a variety of WLAN cards and driver revisions. Separate analyzers were used because protocol analyzers can only listen on one channel at a time, and switching between channels with a single card results in a tremendous amount of dropped packets. Different laptops were used because some WLAN cards are embedded into the system, whereas other systems have PC cards that are hot swappable. Most of the cards were 802.11b/g, however most of the tests were run using 802.11b only data rates to ensure consistency with roaming. By using 802.11b data rates, which are the most prevalent for WLANs, it was ensured that any driver issues pertaining to 802.11g only data rates would be eliminated. Most WLAN implementations utilize the 1Mbps data rate for beacons and probing, so this would provide the best baseline to test the performance given that most manufacturers would design for this environment.

3.2 Results and Deductions

Basically, every minute, the WNIC would probe one or more channels to see what APs were around and determine their advertised data rates and transmit powers. It appeared that if an AP was transmitting at a greater RSSI than the current AP, and the difference of the RSSIs was greater than a threshold, the client would change channels to get the better signal. It was also found that should a signal become degraded to the point that packets were lost prior to expiration of the timer; the system would start probing, trying to find a better AP to be associated to. The level of degradation necessary was very subjective, with some clients roaming at low error levels and other clients roaming only after a nearly complete loss of throughput through the channel and many errored seconds of data.

The data from the analyzers which consisted of a listing of packet captures showed RSSI between the two APs as a function of time. Packets indicating probe (discovery) sequences and successful roaming were manually added to the graphs to show when the roaming actually occurred. A typical roaming sequence is shown in Figure 3.

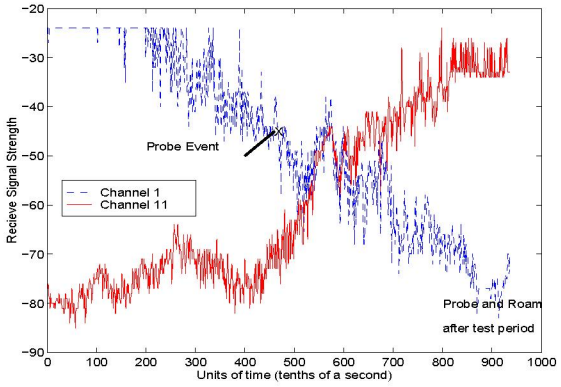

Figure 3. A typical roaming sequence

The graph shows the signal levels as the client and analyzers were moved between the areas. The first probe indicated by the 'X' shows that the client began probing approximately 47 seconds after the start of the trace. Upon

analysis of the packets in the trace, it was not found anywhere that the system roamed. It is assumed that the system roamed right after shutting down the analyzer. The APs were checked before and after moving the cart to verify association status. Another trace, shown in Figure 4 does have the authentication and association packets that signal the roaming sequence, which is also indicated.

Figure 4 also shows that the roaming event clearly has a 1 minute timer that it uses for general discovery. The delay between the start of the second probe process and the actual roaming event is due to the large time it takes for this particular client to probe. It carefully goes through, what appears to be all 11 channels, one at a time, for a period of about 50ms each. During this probe process of a channel, the system requests a power-save mode from the AP, which causes the AP to temporarily buffer packets destined for the client. As a result of this, the client must constantly switch back to its home AP and disable power-saving mode to receive any packets sent while it was on another frequency. This technique, though novel, causes the discover process to take almost 5 seconds for the whole spectrum.

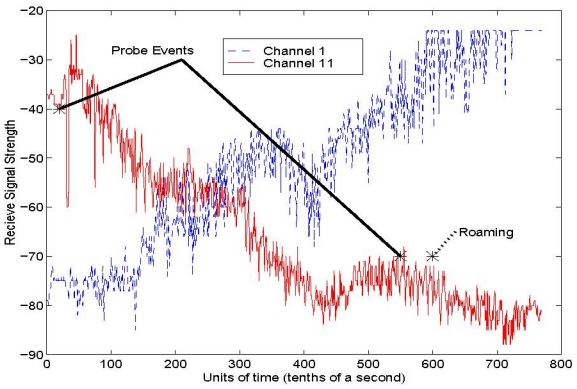

Figure 4. Delay of probes (1 minute)

For every case considered, the client did not experience any large error condition or loss of data. The following cases show both a temporary error condition resulting from lost beacons as shown in Figure 5, and a sustained error condition that resulted in a great deal of dropped data as shown in Figure 6.

The conditions of Figure 5 show that the client under test will begin probing at a fairly low error threshold. When the signal level came back up, the error condition cleared and the client stopped the probing process without roaming. Subsequently it did not roam again until the original timer expired. At that point the other channel had a stronger signal and the client roamed to the new channel.

Figure 6, which was from a completely different client, shows the worst effects of waiting to roam. Additionally, this client had a problem generally known as *stickiness*.

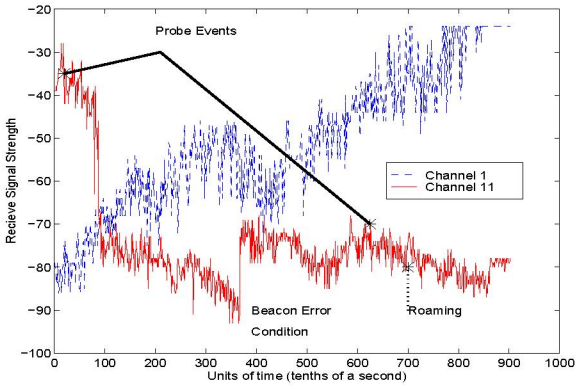

Figure 5. Error condition

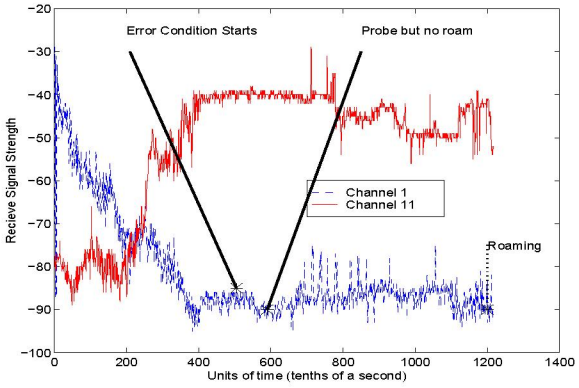

Figure 6. Sticky client

This behavior is essentially when the system performs its probing function but refuses to roam despite having a poor RSSI or lost data. As a result, the client sits with a poor signal quality for another minute until the timer expires again, upon which point it now chooses to roam. Had the signal continued to deteriorate, which was forced in other test runs, this particular client would allow the data throughput to degrade to almost complete failure before disassociating itself, then probing for a new AP.

It is this behavior that is causing the poor signal quality and dropped connections that were seen in the field. Although some clients performed better than others, particularly with regards to error thresholds, all clients tested used the timer mechanism as a basis for probing for better APs.

4 Proposed L2 Roaming Algorithm

We propose a probing process that induces a more pro-active roaming. Currently the only pro-active measure includes an active probe every minute. In a minute's time,

a person traveling at 3-4 feet per second, can easily walk through two or more zones covered by different APs. A better pro-active measure would be to establish a performance baseline of RSSI, and continuously monitor it for any significant changes. Obvious increases in RSSI, such as coming from the edge of a covered zone to the center, would require re-establishing the baseline; however a decrease of 15 db, would merit an inspection of surrounding APs to determine if roaming is necessary. If it is found that the RSSI of the AP is better than surrounding APs, then the lower value is recorded as the new baseline, and the process is restarted. The other more reactive functions of the previously tested devices would still be used, such as the error trigger and the timer function. The error trigger is a catch all, in case any of the other functions are unable to generate a necessary roam. The timer function is still necessary in the case of *perimeter movement* or *device creep*, to detect a more suitable AP. Perimeter movement would result from establishing a baseline at the outer edge of a zone, such as near an exterior wall, however if the user keeps an equidistance from the AP, but moves closer to another AP zone by going around the perimeter of the zone, the client will not roam until an error occurs or the signal drops significantly. Device creep is from a slow moving mobile station, such as a computer on wheels, which is frequently used in clinical facilities. They are moved slowly relative to the APs. In either of these cases the timer expiration would allow the transition to a more suitable AP. The timer mechanism, however is still long, and should probably be shortened to 30 seconds. A flowchart of this new algorithm is shown in Figure 7.

In addition to the changes in the decision process of when to probe/roam; the new roaming model should also take into consideration what channels to probe. In an infrastructure environment that has been properly designed, most adjacent APs are laid out with non-overlapping channels (i.e. 1, 6, and 11). Given this relatively safe assumption, it makes little sense for a client to investigate all legal channels of the available spectrum all the time. As a result, all probing should be done on non-overlapping channels, and other channels should only be examined in the unlikely event that nothing suitable is found.

4.1 Algorithm Validation and Results

In order to validate the proposed changes, parts of the algorithm were transferred to some thresholds and evaluated against RSSI from collected test data. The data, which was collected from the protocol was analyzed and sorted by time, signal type (beacons) and sending station's MAC address. The proposed algorithm was not implemented on the wireless network interface card. However, we exported trace files of RSSI data. The algorithm was emulated as

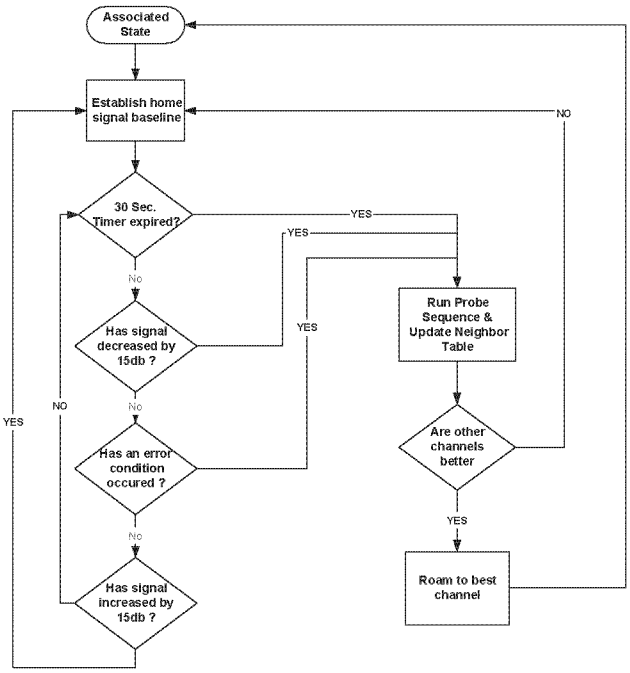

Figure 7. Proposed algorithm

functions in a program, and the functions were executed against the data files to show the points in time at which the new probe sequences would be generated.

To analyze the data, a logic function that displayed a RSSI in db was maintained. Also, we marked a packet if it was a beacon from the appropriate AP. Any other packet generated an error message to indicate that it was not a valid packet for inspection. Another logic function maintained a running 2-second (approximately) average of the RSSI of beacons. The baseline RSSI for the AP was established by counting the previous 20 beacons, and computing the average. If 20 valid packets were not found, the function returned a 'FALSE' condition.

Once an average was established, a function was generated that would monitor only beacon packets and compute a running average over the past second or so, to see if the RSSI had changed. If there were no beacons in the examined cells, the system returned an error. If the running average drops 15 db below the baseline the system will indicate a *Probe* is necessary. If we assume that no other viable AP was found, we re-establish our baseline using a variation of the logic functions, and continue monitoring the running average using the reference to the new baseline. The new function shows when the average generates a probe request, then a new baseline is established and the running average is monitored again against the new baseline.

Additionally, since AP beacon packets are so important to the process, an error checking mechanism is established that monitors all packets over a given second to ensure that

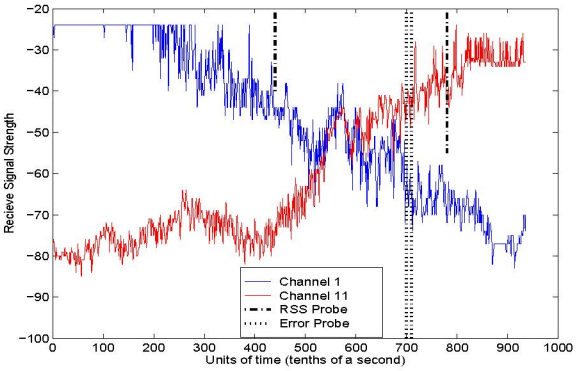

Figure 8. Roaming sequence with new algorithm

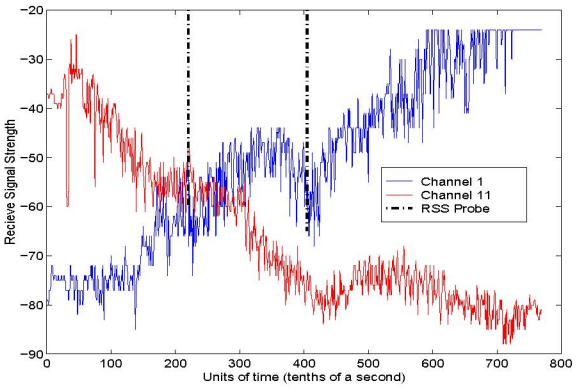

Figure 9. Delay of probes with new algorithm

at least 80% of the beacons are received successfully. If not, a probe request is generated as this indicates an error condition as well as a problem for monitoring, and a probe request sequence is probably necessary.

Both the error condition and the baseline function generate a numerical value during an event. The baseline function returns the new baseline in db, and the error handler returns the value of -99. When the data is plotted on the existing graphs, these numerical values act as a pointer to probe events because of their numerical value (text values are plotted as 0). A graph of the first run, which was shown in Figure 3, is shown again in Figure 8, with the contents of the new functions that represent the algorithm.

Similar runs of data were collected that showed similar results. Although most did not have missed beacons, they all would have generated an earlier roam from the decreased signal level. These new results are shown in Figures 9 and 10.

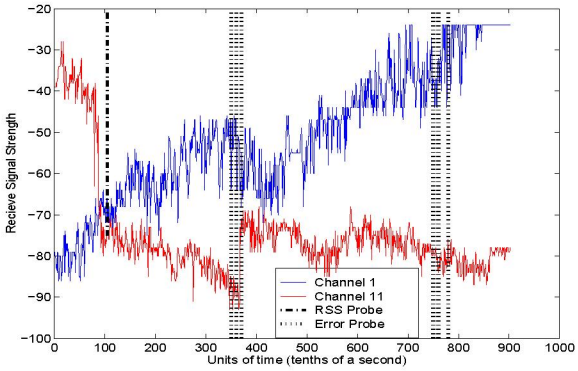

Figure 10. Error condition with new algorithm

5 Conclusions

In this paper, we proposed a new algorithm that was shown to be more effective by simulating the decision process of Layer 2 roaming. The plots shown in Figures 3, 4, and 5 indicate when the actual probe sequences occur along with subsequent roaming. The same data was used to obtain Figures 8, 9 and 10 in which used the proposed algorithm. These suggest that handoff is likely to occur earlier by an order of 20, 15 and 25 seconds respectively. Although the effects of long timer (say 30 seconds) were not shown, its effects are rather intuitive. Likewise, the positive effects of examining only 3 channels instead of all 11 are rather obvious given that the client operates in an infrastructure environment with a known design and frequency layout. Finally, it was found that the root cause of many of the worst problems observed in the production environment were actually caused by 1) a failure on the part of the client to effectively roam after successfully finding a better AP; and 2) an unusually high error threshold to trigger the reactive probing and roaming. The use of a 1-minute timer only exacerbated the problem which could be described as "buggy" at best.

References

[1] V. Brik, A. Mishra, S. Banerjee, "Eliminating Handoff Latencies in 802.11 WLANs Using Multiple Radios: Applications, Experience, and Evaluation" ACM/USENIX Internet Measurement Conference (IMC), October 2005.

[2] M. Borsc and H. Shinde, "Wireless security & privacy", IEEE Inter. Conf. on Personal Wireless Communications (ICPWC) 2005, pp. 424-428.

[3] A. Mishra, M. Shin, and W. Arbaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process", *ACM SIGCOMM Computer Communication Review, Vol. 33 Issue 2, April 2003, pp. 93-102.*

[4] M.S. Obaidat and D.G. Green, "SNR-WPA: an adaptive protocol for mobile 802.11 wireless LANs", IEEE Inter. Conf. on Communications (ICC), 2004, Vol. 7, pp. 3911-3915.

[5] Bruno, Charles, "Enterprise Wireless LANs", *Tolly Group*, August 2004.

[6] "Lightweight Extensible Authentication Protocol - LEAP", http://www.cisco.com

[7] "Cisco Aironet 1200 Series Access Point - Data Sheet", *Cisco Systems Inc*, San Jose, CA, 2006.

[8] "AiroPeek for Windows Version 1.1.0.12", *Wildpackets, Inc*, 1995-2001.