New Methods of Spoof Detection in 802.11b Wireless Networking

A Thesis
Submitted to the Faculty
in partial fulfillment of the requirements for the
degree of

Master of Science

by

Douglas Madory

Thayer School of Engineering
Dartmouth College
Hanover, New Hampshire

JUNE 2006

Examining Committee:

Chairman_____
Prof. George V. Cybenko

Member_____
Prof. Andrew Campbell

Member_____
Prof. Eugene Santos

_____
Charles K. Barlowe
Dean of Graduate Studies

_____
Douglas Madory

Abstract

This thesis presents two new methods for detecting spoof attacks in an 802.11b wireless network. The two methods are Sequence Number Rate Analysis (SNRA) and Signal Strength Fourier Analysis (SSFA).

The explosive growth of 802.11b networks has coincided with an increased presence of security threats to these networks. A large proportion of these threats are in the form of spoof attacks. Spoof attacks involve one device assuming the identity of another to perform malicious behavior. The available security tools to detect such behavior are quite limited.

Current methods of sequence number analysis simply detect gaps in the monotonic incrementing series of sequence numbers in transmitted frames. However, these methods result in large amounts of false positives on wireless networks which experience even small amounts of frame loss. The proposed method considers the time difference between consecutive frames to allow for naturally occurring loss while still detecting invalid sequence numbers.

We have proposed a method of spoof detection using signal strength analysis where currently no fielded method exists today. The unpredictable nature of environmental effects on signal propagation and a lack of signal strength stability due to calibration drift in low-quality wireless networking cards present significant challenges to using signal strength to detect wireless spoofs. By performing a Discrete Fourier Transform (DFT) on a sliding window of signal strength values (RSSIs), it can be demonstrated that the statistical variance of the high-frequencies which result from the interference between the attacker and the victim can very accurately yield evidence of a spoof attempt. •

• This work was supported under ARDA/DTO Award No. F30602-03-C-0248, DOJ Award No. 2000-DT-CX-K001, and DHS Award No. 2005-DD-BX-1091. Points of view in this document are those of the author and do not necessarily represent the official position of DHS.

Acknowledgements

I would like to thank Professor George Cybenko for his help and encouragement as my advisor, and Professor Eugene Santos and Professor Andrew Campbell for their participation on my thesis committee. Additional thanks goes to Professor Eric Hansen for his assistance on the topics related to signal processing.

I would also like to thank Yong Sheng and Vince Berk for their assistance and guidance, John Murphy, who originally suggested to me to try the Short-Term Fourier Transform, as well as Raz Magar, Udayan Deshpande and the rest of the members of the MAP group in the CS department for all of their support and assistance.

Finally, I would like to thank my wifey, Erin, for all of her devotion and support.

Table of Contents

List of Figures

Chapter 1: Introduction

The explosive growth of the deployment of 802.11b wireless (WiFi) networks in the last few years has been accompanied by a similar growth in the various types of security threats to these networks. The most common threats include WiFi network attacks such as Denial of Service (DoS) attacks, Reduction of Service (RoS) attacks, "Greedy" behavior, and Man-in-the-middle (MITM) attacks [20] [26] [33]. DoS, RoS, and MITM attacks all involve the attacker attempting to impersonate a legitimate node on the network to send his malicious traffic. This general behavior is called a spoof. An attacker can spoof another device by changing his hardware address to be the same as the victim device when both attacker and victim are on the same channel (frequency). Two ways to tell that the two devices are different is through the use of sequence number analysis at the OSI layer two or signal strength analysis at OSI layer one.

Figure 1 illustrates a typical spoof attack. It is important to clarify a couple of terms that are used in this paper at this point. During a DoS attack, such as a DeAuth attack, in which a legitimate node is denied access to a wireless network as a result of frames sent from an attacker impersonating an access point, the "victim" of the overall attack is, of course, the node that is denied access to the wireless network. However, for

the purposes of this paper, the "victim" of the spoofing behavior is the access point, who is impersonated.

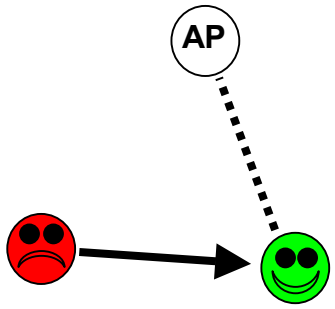

Figure 1 Typical DoS spoof attack. Attacker (colored red) sends DeAuth frames to the victim (colored green) that claim they are legitimate DeAuth frames from the AP that the victim is associated to. The green node is the victim of the overall attack, but the AP is the victim of the spoofing itself.

This paper will present two general methods of detecting such malicious behavior and compare them to corresponding examples of the state of the art in each detection domain. The proposed sequence number analysis technique called Sequence Number Rate Analysis (SNRA) is compared to the Snort-wireless preprocessor, MacSpoof. The proposed signal strength analysis technique called Signal Strength Fourier Analysis (SSFA) is compared to a proposed signal strength analysis technique from the Rutgers WINLAB [17] [31]. All four techniques are used to detect spoofs in twelve experiments. Each pair of corresponding analysis techniques are evaluated by comparing the detection rates against the false positive rates.

This paper demonstrates that the proposed methods of spoof detection out perform their respective counterparts at spoof detection in simulations as well as in practice.

1.1 Sequence Numbers

The 802.11 protocol stipulates that every participating device will monotonically increment the 12-bit sequence number field in the 802.11 header of every management and data frame (control frames do not get a sequence number) [15]. If a device is posing as another device, a naïve attack will produce two distinct but interleaved streams of sequence numbers which can be detected easily if there is little loss. One way a sophisticated attacker could cover his tracks would be to have the sequence number of his frame match that of a recent frame sent by the victim and flip the retransmit bit in the 802.11 header making the repeated sequence number appear like a natural retransmission. Another technique the sophisticated attacker might employ would include hijacking the entire sequence by sending out a frame with the next successive sequence number while corrupting the legitimate frame from the victim so that it gets dropped by any receiving nodes. Clearly, there is a limit to what can effectively be done with sequence number analysis because it may always be possible for a sophisticated attacker to cover his tracks with respect to the sequence numbers.

Existing analysis methods suffer from numerous false positives due to natural frame loss as well as missed detections when spoofed frames do not come fast enough to surpass an arbitrary threshold. The proposed Sequence Number Rate Analysis (SNRA) technique addresses these shortcomings of existing techniques. SNRA can detect spoofs consisting of very small numbers of frames in an environment with a large amount of frame loss.

1.2 Signal Strengths

In the physical layer (OSI layer one), received signal strength can also be used to detect a spoof. Despite being quite erratic, received signal strength values generally follow a fairly tight and predictable distribution. It is almost certain that the distribution of signal strength values for a certain hardware address will be appear different when a spoof is occurring as compared to the distribution of values coming only from the victim. Effectively detecting this distribution perturbation is what the Signal Strength Fourier Analysis (SSFA) technique is designed to solve.

In the case of a spoof, the unpredictability of the propagation of RF energy is a good thing. It forces each transmitting device to have a unique RF signature from the perspective of a sensor. Due to the effects of multipath interference among other factors, two devices sitting at two different three-dimensional locations will have two separate and distinct signal signatures (mean and variance of signal strength values) [12]. While it is possible for two devices to appear to have the same signal strength from the perspective of one sensor, if a second or third sensor is added, it will be virtually impossible to appear the same from every sensor. Even if the attacker and victim were located next to each other and transmitting the same power, the paths that those emissions would take through the air would be slightly different. The different paths would reflect on surfaces in different angles. The behavior of multipath would allow a sensor to tell that two devices were communicating instead of just one.

In addition to the uniqueness of received signal strength indication (RSSI) at the

sensors, the attacker has no idea what it looks like from the sensor's perspective. If the attacker could somehow become aware of how it was received from every sensor, then it would be possible, in theory, for an attacker to move itself into a position which allows it to blend into the RSSIs of the victim's traffic. However, this requires knowledge that the attacker can never have and even if it did somehow have this knowledge, as soon as it communicated its first frame prior to relocating, its presence would be detectable. Therefore, the attacker would have to predict exactly what signal strength the sensor or sensors would receive based on any three-dimensional point. When the effects of multipath are taken into account, this becomes a near impossible task.

Therefore, unlike sequence number analysis, signal strength analysis is essentially a physics problem. Sequence number analysis may become an arms race within the protocol domain, where each protocol fix is followed by a protocol exploit and so on. Signal strength analysis can end the arms race because there is little an attacker can do to change the laws of physics in order to hide his presence. What makes the use of RSSIs difficult in practice is that there is always environmental variation, calibration drift, and other factors that make the trace of one device's RSSIs unstable and noisy. However, for the most part RSSIs from frames originating from the same device are normally distributed and statistical analysis can be brought to bear on handling this data.

At the time of this writing, there are no fielded signal strength analysis techniques for detecting spoofs in any of the major wireless intrusion detection systems (WIDS) [2] [3] [31]. Although signal strength analysis has been employed for localization with varying levels of success in several research projects, prior to this work an implemented signal strength analysis system for spoof detection does not yet exist [6] [30].

Chapter 2: Sequence Number Analysis

In the 802.11b protocol, each device transmits management or data frames that include a sequence number to assist in reassembly [15]. There is an ancillary security function to the sequence number that although an attacker may guess the next sequence number, the victim will eventually send out that number as well. The occurrence of sequence numbers that are either out of sequence or include duplicate sequence numbers may indicate a spoof.

The matching of sequence numbers is a non-trivial task because the generation of sequence numbers is a low-level device driver operation. When combined with the fact that sequence number analysis for spoof detection is far from an exact science, the result is that most publicly available attack tools which involve a spoof do not bother attempt to match the sequence numbers of the target. The Matlab simulations presented later in this chapter will demonstrate that spoof detection schemes such as the industry open-source benchmark, Snort-wireless, can suffer from either false positives during periods of high frame loss, or missed detections when spoofs involve very few frames [31].

2.1 Snort-wireless MacSpoof

Snort-wireless accomplishes its spoof detection using a configurable preprocessor called MacSpoof. The single piece of evidence that MacSpoof relies on is the sequence number. If MacSpoof sees two consecutive frames with sequence numbers greater than a specified gap limit (called "tolerate_gap" in the configuration), then a gap has occurred. To trigger an alert, a certain number of gaps (called "threshold" in the configuration) as defined by tolerate_gap must occur within a specified time window (called "expire_timeout" in the configuration). The default values for tolerate_gap, threshold, and expire_timeout are 5, 10 and 120 respectively.

An excerpt from Snort-wireless's MacSpoof configuration is below:

```
# MacSpoof
#---------------------------------------------
# MacSpoof detects wireless MAC addresses involved in some MAC spoofed traffic.
#
# Arguments:
#
# MACSPOOF_MASKED_ADDR => list of MAC addresses excluded from wireless MAC spoofing
#        [var             detection process
# tolerate_gap [num]   => tolerate missing frames between two consecutives frames issued
#                         from same MAC address
# threshold [num]      => number of abnormal sequence number gaps during time delta to
#                         trigger an alert
# expire_timeout [num} => time period used to keep count of abnormal seq number gap
# spoofed_target_limit => maximum number of MAC addresses inserted inside MAC spoofed
#        [num]            addresses mempool
# prune_period [num]   => number of seconds to wait for looking after some decayed
#                         MAC addresses inside mempool

preprocessor   macspoof:   $MACSPOOF_MASKED_ADDR,   tolerate_gap   5,   threshold   10,
expire_timeout 120, spoofed_addr_limit 100, prune_period 30
```

The problem with a system such as this one is that in a live wireless networking environment, natural loss and therefore natural gaps occur with great regularity. Due to phenomena such as a "Hidden Node", a sensor may miss hundreds of frames causing false alarms simply due to the fact that one device's transmissions are overpowered by a

closer device's transmissions [12]. The more sensitive the receiver, the greater the problem with frame loss due to hidden nodes. Snort-wireless's MacSpoof preprocessor reflects the general thinking when it comes to using sequence number analysis to detect spoofs [31] [33]. This general thinking tries to detect gaps of a certain size without relation to time elapsed as evidence of spoofing.

2.2 Sequence Number Rate Analysis (SNRA)

The Sequence Number Rate Analysis (SNRA) technique calculates a "transmission rate" by taking the difference modulo 4095 of the sequence numbers from consecutive frames and dividing by the difference of their corresponding arrival times. If the set of sequence numbers and arrival times suggests a transmission rate that is greater than the theoretical transmission limit, then SNRA concludes a spoof has occurred. By using this method, gaps from natural frame loss do not cause false alarms because they will not yield an abnormally large transmission rate.

Attacker: 5

Victim: 1 2 3 4

In the above example, assume each number is a sequence number in a series of frames supposedly from a single hardware address. If each frame is sent one millisecond apart, then the transmission rate between the first two frames would be *(2-1) frames/0.001 second* or *1000 frames/second*. The rate between the third and four frames would be *(5-3)/0.001* or *2000 frames/second*. The rate between the four and fifth frames would be *(4-5)/0.001* or *4,094,000 frames/second* because the difference in sequence numbers is modulo 4095. This abnormally large transmission rate is evidence of a spoof. Depending on the numbers and arrival times involved, the transmission rate from victim frame to spoofed frame or the reverse can produce evidence of a spoof.

In addition, if the sensor system of a distributed wireless intrusion detection system (WIDS) uses a sampling or channel-hopping technique in which has individual sensors covering multiple channels, false alarms will typically not result from intermittent coverage resulting from the hopping [9].

The basis for the SNRA is simple. There is a theoretic limit to how many frames can be sent by an 802.11b device in 1 second. This limit, detailed below, is approximately 5314 frames. When the calculated frame transmission rate surpasses this theoretical transmission limit, the SNRA concludes that a spoof has occurred.

Theoretical Transmission Rate Limit

The following calculations are used to calculate the theoretical transmission rate limit of 802.11b wireless networks:[1]

Max data rate of 802.11b:	11Mbps
	(or 1,375,000 8-bit bytes/sec)
Size of smallest 802.11b frame (e.g. ACK, CTS):	14 bytes
Worst-case frame rate:	98,214 frames/sec

Furthermore, frames aren't sent one right after another. There are wait periods between the transmission of frames as specified by the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) access mechanism of 802.11b as well as physical layer overhead from the Physical Layer Convergence Procedure (PLCP) and preamble transmission [12]. The Short Inter-Frame Space (SIFS) of 10 μs is used to estimate the minimum wait period due of CSMA/CA.

The physical layer overhead includes a preamble header which is 144 bits transmitted at 1 MHz, and the PLCP header which is 48 bits transmitted at 2 MHz [12]. The transmission durations of the preamble and PLCP are therefore 144 μs and 24 μs, respectively.

The minimum transmission overhead for each frame is:

10 μs (IFS) + 144 μs (preamble) + 24 μs (PLCP) = 178 μs

A minimum transmission duration for each frame is calculated by taking the reciprocal of the maximum frame rate (98,214 frames/sec), to arrive at 10.18 μs per

[1] Based on calculations by Dartmouth PhD student, Rajendra Magar.

frame. Adding this term to the minimum frame overhead yields and overall frame duration of 188.18 μs per frame. Taking the reciprocal of this term yields our theoretical frame transmission limit of 5314 frames/sec.

2.3 SNRA vs MacSpoof

To directly compare SNRA against Snort-wireless's MacSpoof, several Matlab simulations were used which are described in this section.

To give MacSpoof a fighting chance, the expire_timeout parameter was lowered from from two minutes to one second. In the simulation and in later experiments, gaps occur at a fairly high frequency (due to both attacks as well as natural loss) and an enormous analysis window would have kept the detector perpetually yielding false positives.

The following Matlab code generated an approximation of what a spoof attack would look like from the perspective of sequence numbers from a sensor. In this example, every fifth frame has an anomalous sequence number.

```
for i=1:len
    data(i,1)=i/100;          %time val
    data(i,3)=mod(i,4095);    %sequence number
    if (i>500 && i<600 && (mod(i,5)==1))
        data(i,3)=mod(i+250,4095);
    end
end
```

When both techniques are run side by side to analyze this simulated spoof, the result is Figure 2. MacSpoof detects the spoof at 5.11 seconds, while SNRA detects the spoof at 5 seconds even (the time of the first frame). Although MacSpoof was a little slower to make the detection, this experiment is characterized as a success for both techniques.

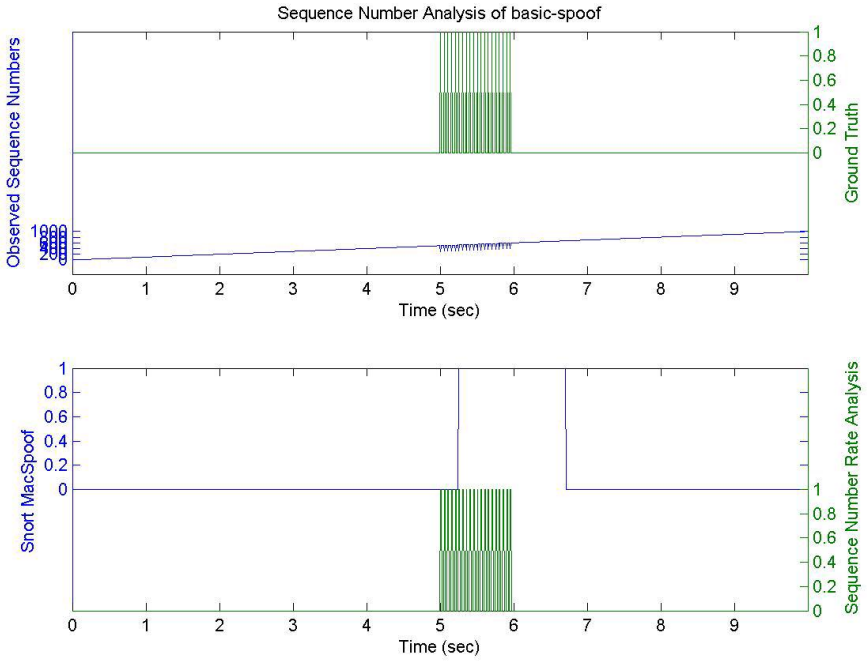

Figure 2. Comparison of Sequence Number Analysis Techniques Against Simple Spoof Attack

The weakness of an approach like MacSpoof is that there is always an arbitrary threshold that an administrator must set for the number of gaps in a certain window. If the threshold is too low, there will be many false positives from natural loss, however, if it is too high this will result in missed detections.

In the next simulation, illustrated in Figure 3, the incidence of spoof frames is lowered from 1:5 to 1:25. Spoofed frames arrive at a rate too slow to be detected by MacSpoof but they are caught with SNRA.

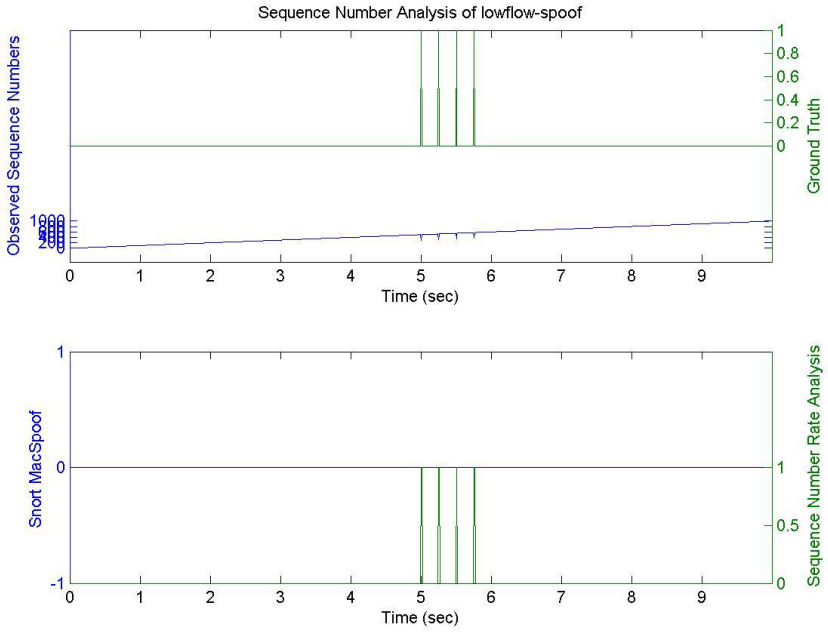

Figure 3. Comparison of Sequence Number Analysis Techniques Against Spoof Consisting of Few Frames

The general weakness of MacSpoof is its inability to discern whether gaps are natural or malicious. In the simulation below, blocks of frames have been removed from a trace of sequence numbers to represent frame loss. There are no spoof attacks, just a period of high loss. SNRA technique isn't fooled because despite the existence of gaps the frame transmission rate hasn't changed.

Here is the Matlab code used:

```
len=500;
for i=1:len
    data(i,1)=i/100; %time val
    data(i,2)=30; %signal strength
    data(i,3)=mod(i,4095); % sequence number
end

for q=1:2:30
    data([200+q:206+q],:)=[]; %delete frames to simulate frame loss
end
```

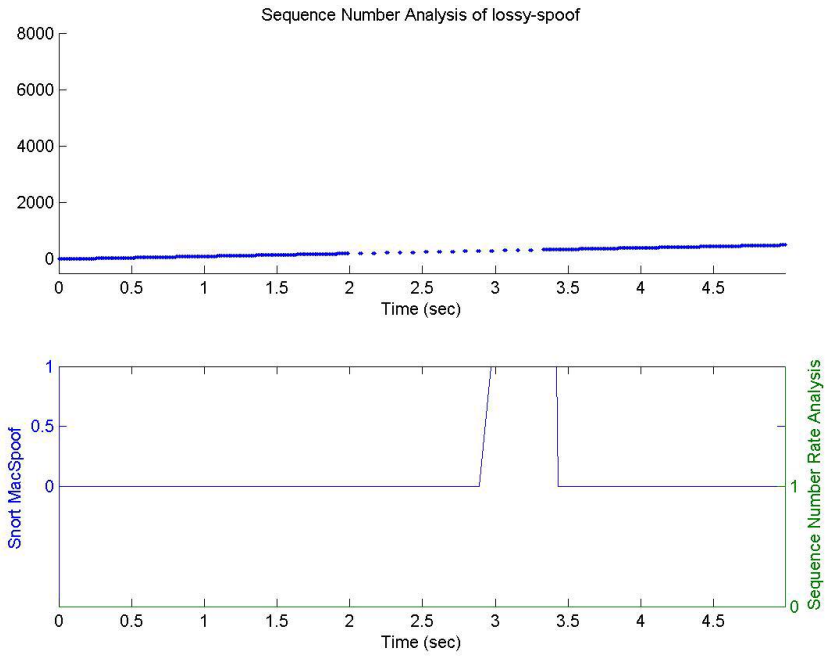

Figure 4. Comparison of Sequence Number Analysis Techniques Against Lossy Behavior Without a Spoof

Retransmitted Frames

A challenge for the Sequence Number Rate Analysis technique is the handling of re-transmitted frames. In 802.11b, every data frame transmitted is assumed lost if an acknowledgement frame is not sent in response. When this loss is detected a frame is retransmitted with the same sequence number as was previously sent. In addition, the retransmit flag is set in the 802.11b header of a retransmitted frame to identify it as such. The sequence number of such a frame will inevitably arrive out of order and then cause a false positive in the Sequence Number Rate Analysis.

The solution to this is to ignore the sequence number of the retransmitted frame but use a hash function such as MD5 to maintain a hash value for each frame with a

unique sequence number for each hardware address [28]. If a frame claims to be retransmitted, then its hash value is compared with the corresponding the value of a frame with the same sequence number and hardware address. If the hash values are different then the retransmitted frame is malicious and presumed to be attempting to evade the SNRA.

2.4 Sequence Number Summary

In summary, the SNRA will not generate false positives due to periods of heavy frame loss. MacSpoof's dependence on arbitrary thresholds leaves it vulnerable to an attacker who times his frames to arrive beneath the thresholds set by a network administrator.

Additionally, since the detection of a spoof with SNRA only involves an operation on two consecutive frames, the computational load is much less than MacSpoof which requires the maintenance of an analysis window which may include hundreds of frames for as many devices.

Chapter 3: Signal Strength Analysis

Signal strength analysis for spoof detection operates under the assumption that frames from two distinct devices be will received at two distinct power levels. If this assumption holds true, then it should be possible to use signal strengths of incoming frames to be able to detect when two or more devices are sending frames as a single source.

Despite technical papers suggesting that this technique may prove useful, no fielded system currently detects spoofing through signal strength analysis [3] [11]. This is due to the fact that making conclusions based upon RSSI values is difficult for many reasons. The RSSI, a measure of the strength of the RF signal at the sensor, is affected by the power of the signal upon transmission from the source and the attenuation of the signal as it propagates. At the source, the device transmits with as much power as it is configured for or is capable of. This power level sometimes fluctuates due to subtleties within the electronics of the radio transmitter. This phenomenon is referred to as calibration drift.[2] 802.11b devices were not engineered to be precision emission

[2] Observed by Prof Wade Trappe of Rutgers University on the ORBIT Wireless Networking Testbed www.orbit-lab.org.

instruments and are therefore susceptible to this signal variation. Sometimes RSSIs may fluctuate deliberately due to mechanisms like Automatic Radio Management (ARM) in access points (APs) from wireless networking equipment manufacturer, Aruba. According to Aruba's website, “ARM technology is used to optimize channel assignments, avoid interference and ensure pervasive Wi-Fi coverage” [5]. The end effect from a sensor’s point of view is that the average power level coming from one source can change abruptly when an automatic power adjustment has been made.

Signal attenuation can be caused by several conditions such as temperature of and moisture in the air, obstacles such as walls and RF reflecting surfaces such as metal cabinets. Additionally, as objects move through the environment, the cumulative environmental effects can change over time. Therefore, even if both the source and the sensor are stationary, the sensor can witness significant RSSI variation over time. RSSI values can even vary from one frame to the next in a matter of hundredths of a second. This variability presents a difficult challenge in making strong conclusions based on RSSI values.

The use of signal strength analysis for spoof detection reduces the task down to a physics problem. There will always be a problem of detecting and reacting to a spoof regardless of what form the current and future wireless networking protocols take. Additionally, since all current and future wireless networking protocols will adhere to the laws of physics and signal propagation, a physics-based detection scheme will be protocol-independent.

3.1 Signal Strength Fourier Analysis (SSFA)

The proposed Signal Strength Fourier Analysis (SSFA) technique uses a Short-term Fourier Transform (STFT) to detect spoofing behavior. An STFT is essentially a Discrete Fourier Transform that is repeatedly performed on a sliding window of data points. While the Fourier Transform is a function of observed frequencies only, the STFT allows for temporal isolation of discrete frequency events adding a chronology of events into the analysis [27]. A simple rectangular windowing function is used because frequency leakage in this application is not a concern [23]. In this application, the discrete frequency event that is being detected is the perturbation of the distribution of RSSIs.

Figure 5 depicts a model plot of RSSI values with a spoof occurring between the times 500 and 600 time units. Figure 6 depicts the resulting color-graph from a STFT with a window 100 units wide sliding in steps of 10. It is important to note that the spoofed frames are being interleaved into the legitimate frames in a random and changing interval. This detail will be revisited in the explanation section. The plot in Figure 5 is based on data generated by the following Matlab code:

```
for i=1:1000
    data(i,1)=30+rand*3;
    if (i>500 && i<600 && (floor(rand*4)~=1))
        data(i,1)=40+rand*3;
    end
end
```

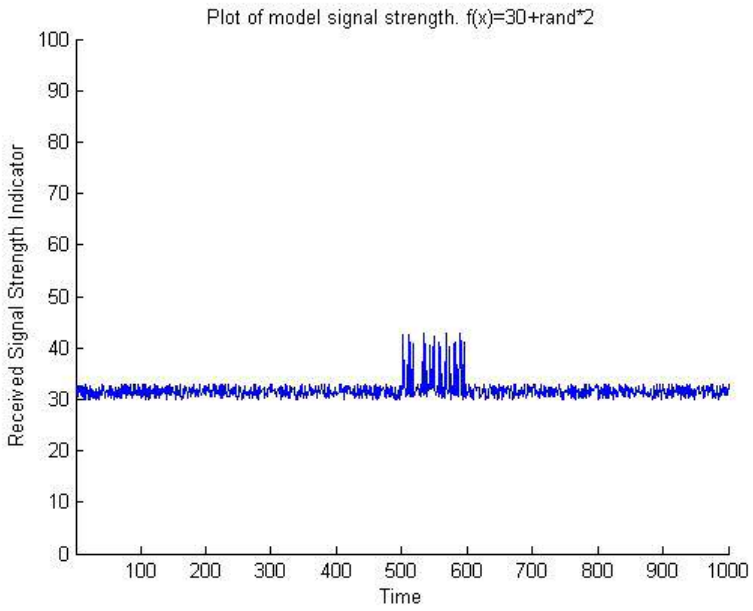

Figure 5. Plot of Simulated RSSI Signal Without Variation With Spoof

In Figure 6 below, the x-axis is window count and the y-axis is relative frequency bins. In this case there are 100 bins because the window is 100 units wide. The colors of each square in Figure 6 represent the magnitude of the frequency in that window.

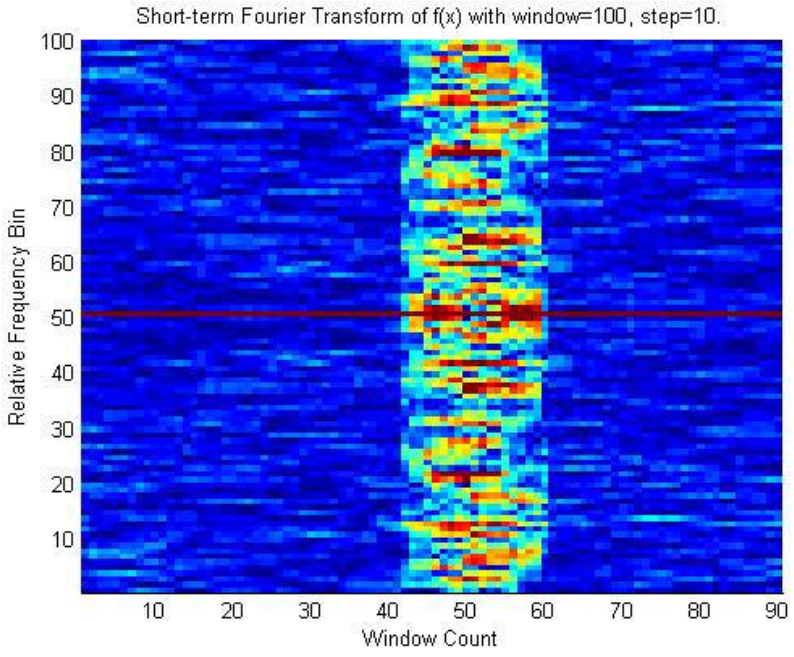

Figure 6. Color-graph of Short-Term Fourier Transform of Plot Described in Fig 5.

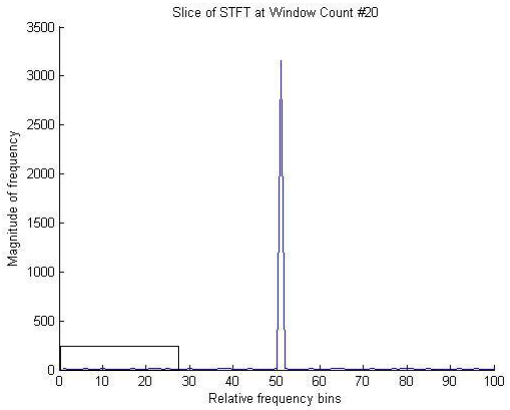

Figure 7. Slice of STFT at Window Count #20

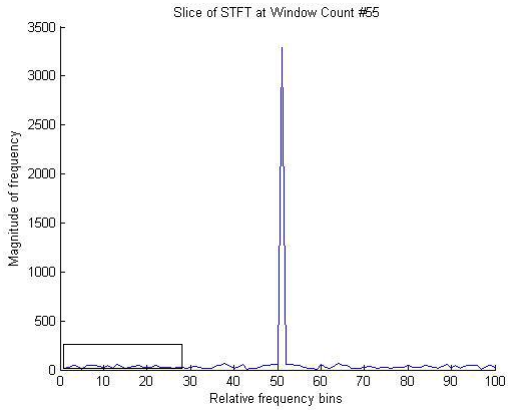

Figure 8. Slice of STFT at Window Count #55

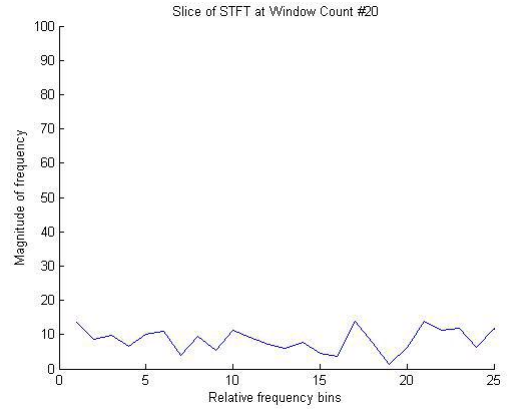

Figure 9. Exploded View of Inset Box from Fig. 7.

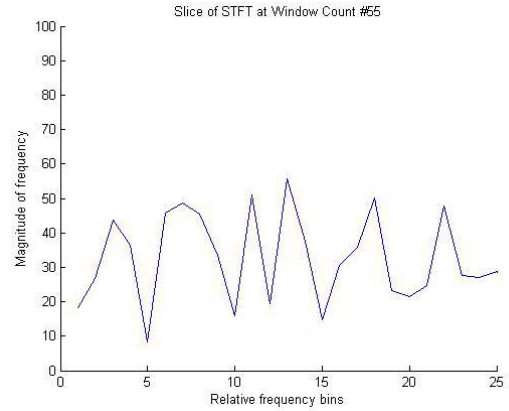

Figure 10. Exploded View of Inset Box from Fig 9. Note the greater variance of the frequencies than in Fig 8.

A closer look at two individual vertical slices of this STFT in Figures 7 and 8 reveals some of the common attributes and differences of these slices. Each slice is a Discrete Fourier Transform performed on the given window of data and since the data represents a real signal, they are symmetrical around the midpoint on the x-axis. Without noise, these signals would be entirely DC signals and therefore there is a very large DC component (the center spike) in the resulting transform. The low-level noise that occurs

on the signal is represented by the presence low frequencies - the values close to the center spike. In Figure 8, higher frequencies appear on the extremities of the plot revealing the presence of a spoof attack.

The following procedure explains how to turn the STFT analysis into a binary flag (0 for no spoof, 1 for spoof). For each STFT window step, calculate the statistical variance of the high frequency values (first quarter for the frequency magnitudes for each Fourier transform) after removing the uppermost and lowermost two values to negate the effect of outliers. If this variance is greater than a threshold and is much greater than the average of the preceding five variances, then a spoof has occurred. Therefore, two conditions must be met for a spoof to be detected; the variance must be above the aforementioned threshold, but also this must be a significant departure from recent variance values. The motivation for this two-condition approach is to eliminate false positives resulting from high levels of natural background noise. See pseudocode below:

```
for each window step
do
      take the first quarter of the frequency magnitude values
      sort and remove the top and bottom two values
      calculate the variance of the remaining values

      if variance > threshold and variance >> avg of previous 5 values
      then
            spoof has occurred
      else
            spoof has not occurred
      end
end
```

A threshold of 50 and a variance difference of 25 over the previous values were used during the simulations and the experiments.

In the next two scenarios, variation has been added to the legitimate signal to

approximate the behavior of natural and ARM-style variation. Natural environmental variation and calibration drift generally take the form of gradual changes over time. Power adjustments made by a transmitting device during ARM appear very abrupt and the average power level of the signal instantaneously jumps from one level to another. To demonstrate natural variation a sine value was added to the legitimate signal in the Matlab code from the previous example:

```
for i=1:1000
    data(i,1)=30+rand*3+5*sin(i/30);
    if (i>500 && i<600 && (floor(rand*4)~=1))
        data(i,1)=40+rand*2+2*sin((i)/30);
    end
end
```

The resulting model plot and corresponding STFT analysis are illustrated in Figures 11 and 12 below. In case be seen in Figure 12 that the natural variation appears at low frequencies while the high frequencies resulting from the spoof are still distinctive.

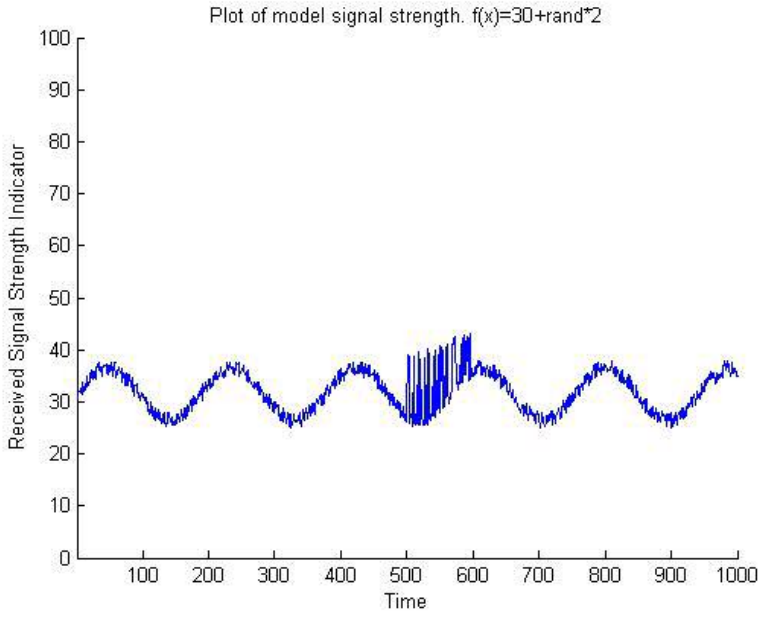

Figure 11. Plot of Simulated RSSI Signal With Gradual Variation With Spoof

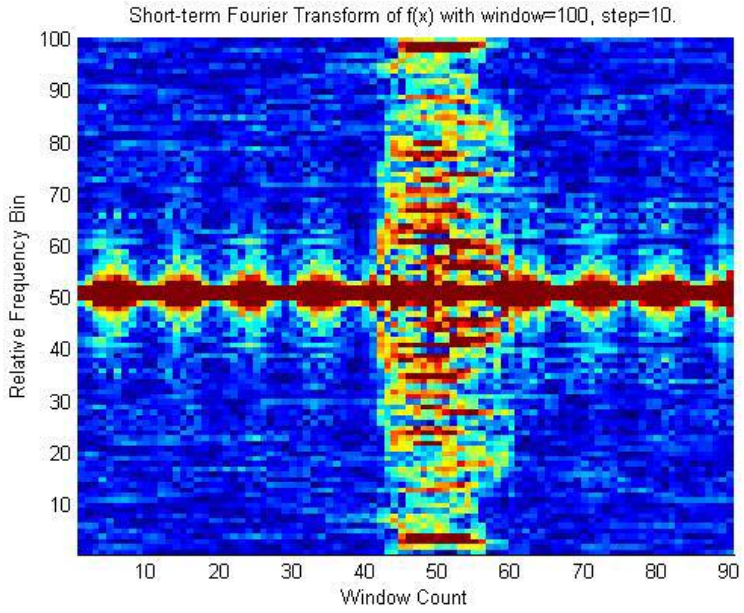

Figure 12. Short-Term Fourier Transform of Plot from Fig 11.

To demonstrate abrupt ARM-like variation an intermittent offset value was added to the legitimate signal in the Matlab code from the previous example:

```
ARM=0;
for i=1:1000
    if (mod(floor(i/50),2)==1) %offset every other 50
        ARM=3;
    end
    data(i,1)=30+rand*3+ARM;
    if (i>500 && i<600 && (floor(rand*4)~=1))
        data(i,1)=40+rand*2+2*sin((i)/30);
    end
    ARM=0;
end
```

This code results in the following model plot and corresponding STFT analysis illustrated in Figures 13 and 14 below.

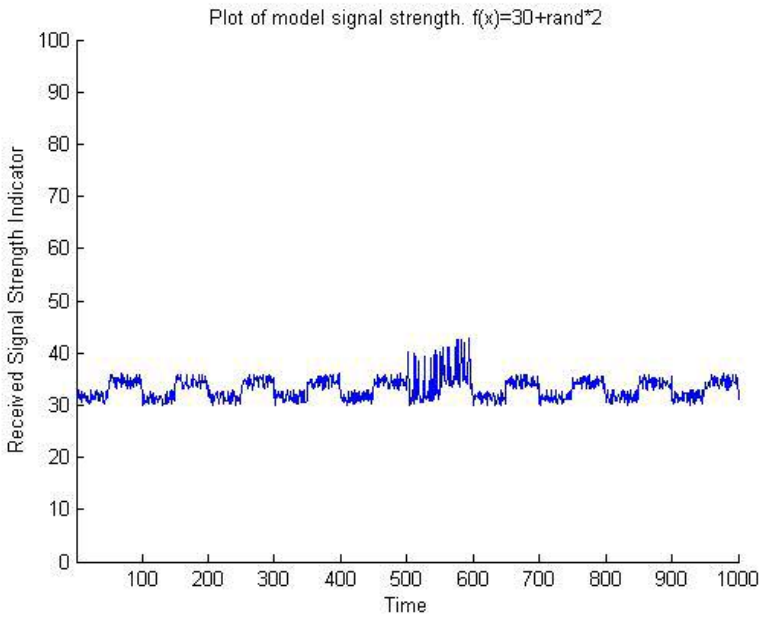

Figure 13. Plot of Simulated RSSI Signal With Abrupt Variation With Spoof

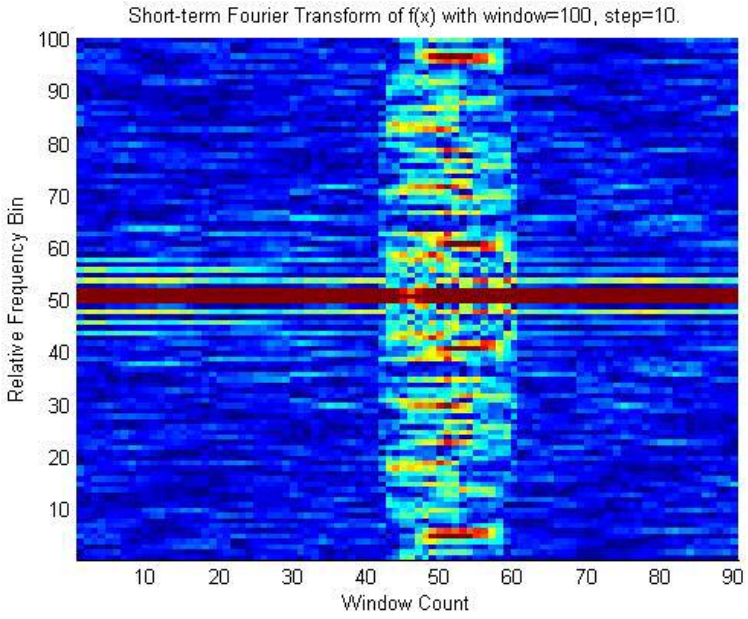

Figure 14. Short-Term Fourier Transform of Plot in Fig 13.

The ARM-like power adjustment scenario would seem to be more problematic because at the instant of each abrupt change, there might be a high risk of a false positive. However, as was the case with the natural variation, the power adjustment variation is reflected in the presence of lower frequencies and the higher frequencies resulting from the spoof attack remain present.

Figures 15, 16 and 17 depict the conversion of STFT analysis into binary conclusions in the three original examples (base signal with no variation, natural variation, abrupt variation).

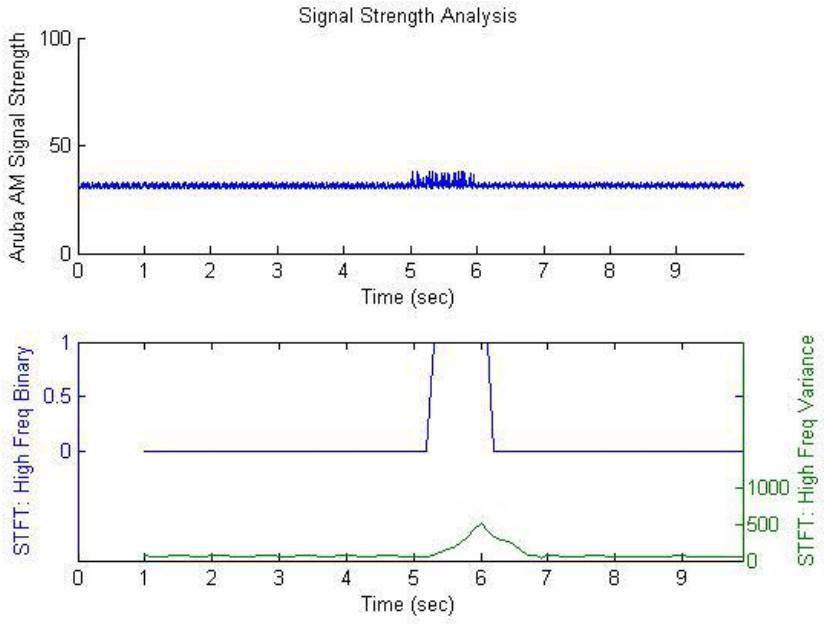

Figure 15. Demonstration of STFT Technique Binary Conclusions Against Plot from Fig 5.

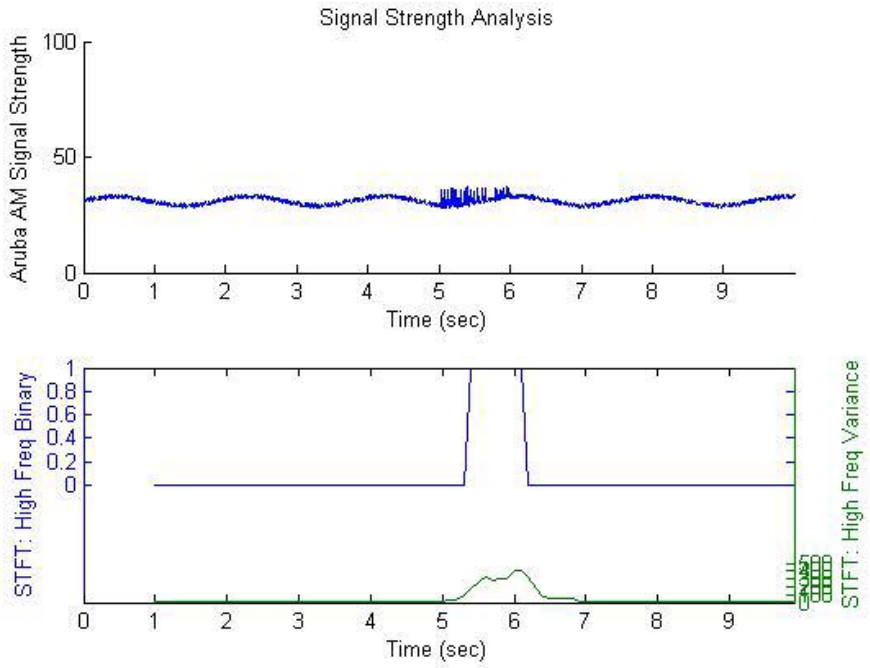

Figure 16. Demonstration of STFT Binary Conclusions Against Plot from Fig 11.

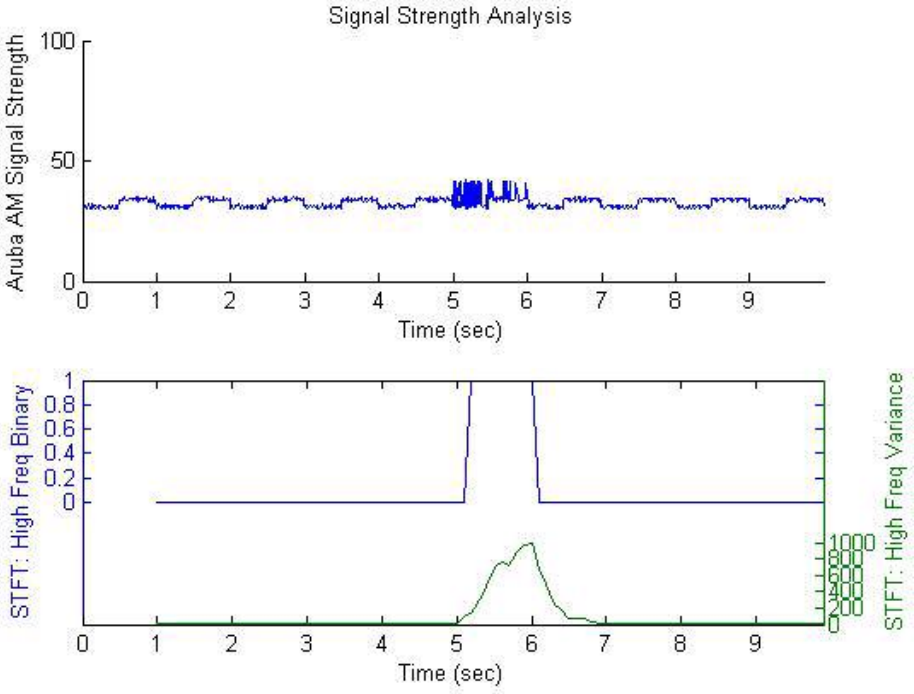

Figure 17. Demonstration of STFT Binary Conclusions of Plot from Fig 13.

As can be seen from the graphs, the detection of the spoof occurs at approximately 5.2 seconds in every case. Note that each spoof begins at 5 seconds, so there is a slight delay in the detection of the spoof. This is due to the fact that in each case, it takes about 0.2 seconds to build up enough variance to cross the threshold set by the SSFA.

3.2 Signal Strength Fourier Analysis (SSFA) Explanation

There are two phenomenon which contribute to creating those signature vertical bands in the STFT color-graphs, which represent the presence of higher magnitude of a large range of frequencies. The first is an interference which arises from the interleaving of two distinct distributions of data points which is discussed further later in the section. It is certain that the frequencies in the vertical band result from interference because when the STFT is performed on the victim and attacker traces separately as illustrated below, no higher level frequencies are present. Since the Fourier transform of a composite signal is the sum of the Fourier transforms of the contributing signals, it is clear from the color-graphs in Figures 20-23 that the higher frequencies must originate from interference between the two signals as opposed to from anything within the signals themselves.

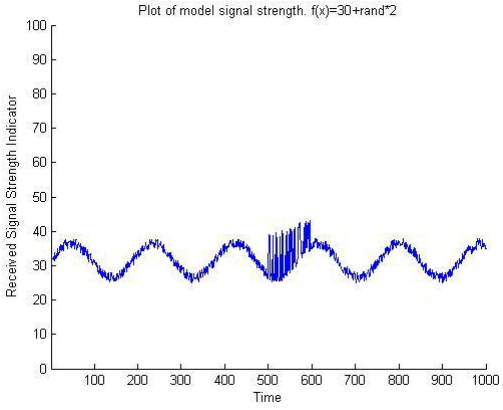

Figure 18. Plot of Combined Signal (Victim and Attacker)

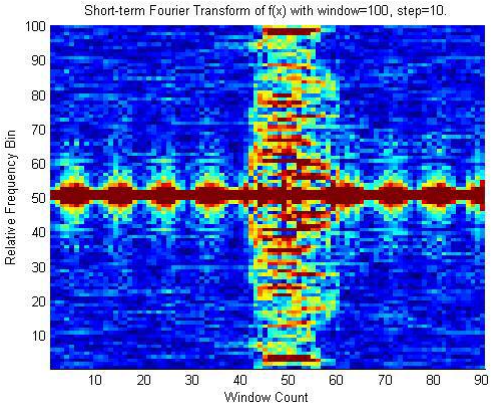

Figure 19. STFT of Combined Signal (Victim and Attacker)

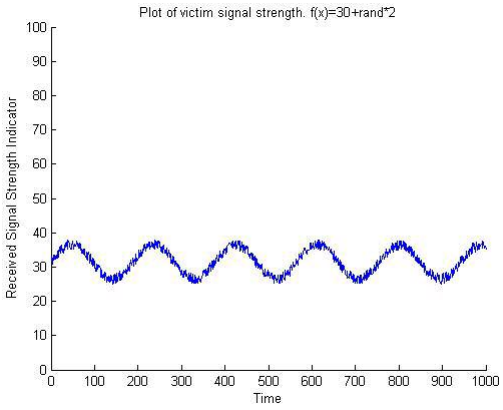

Figure 20. Plot of Base Signal (Victim Only)

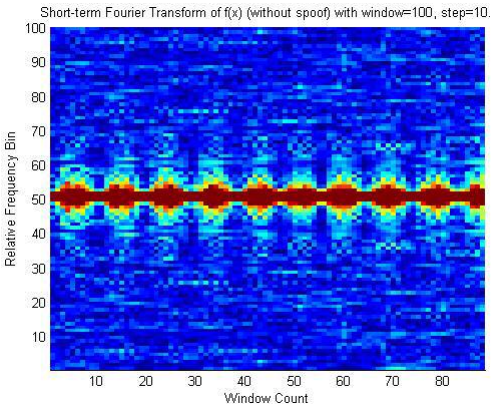

Figure 21. STFT of Base Signal (Victim Only)

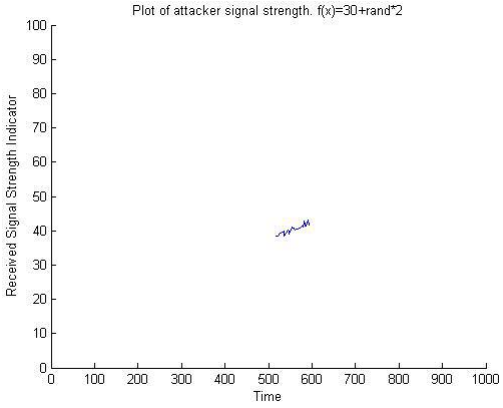

Figure 22. Plot of Spoof Signal (Attacker Only)

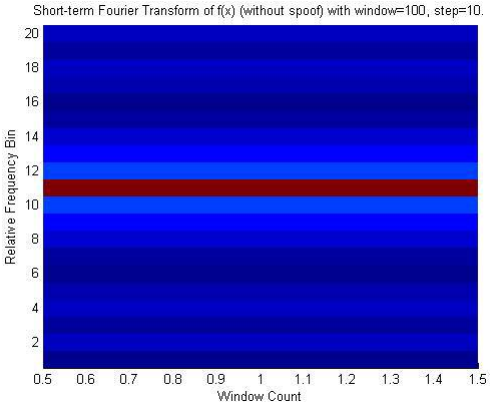

Figure 23. STFT of Spoof Signal (Attacker Only)

The second phenomenon which creates the vertical bands in the STFT color-graphs is the random intervals with which the interleaving takes place. If the interleaving of the two distinct distributions occurs with a regular interval such as every other or every third values, only specific frequencies are illuminated in the STFT. For example, if the Matlab code from the previous example is altered so that every second and third frame is from the spoofer, then only a specific frequency gets illuminated.

```
for i=1:1000
    data(i,1)=30+rand*3+5*sin(i/30);
    if (i>500 && i<600 && mod(i,3)~=1)
        data(i,1)=40+rand*2+2*sin((i)/30);
    end
end
```

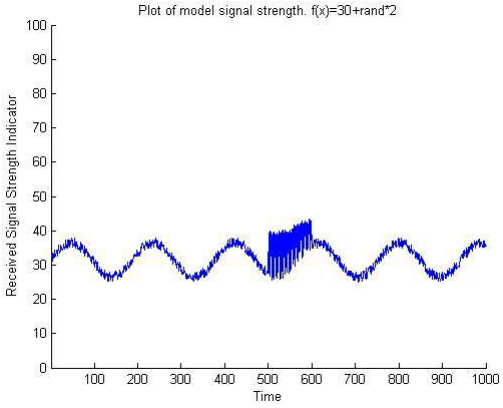

Figure 24. Plot of Base Signal with Natural Variation with Regularly Interleaved Spoof Frames

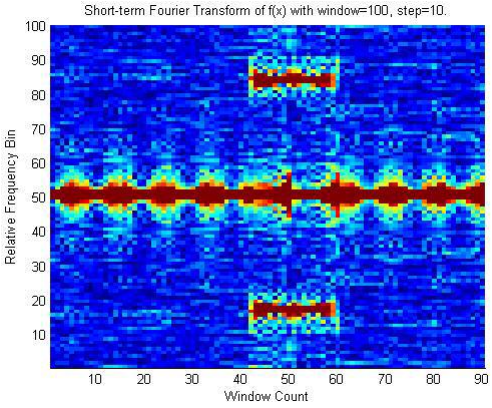

Figure 25. STFT of Base Signal with Natural Variation with Regularly Interleaved Spoof Frames

When the interleaving is randomly driven, it requires many more frequency coefficients (to include higher frequencies) to describe the signal observed in the STFT window. This dramatic increase in frequency coefficients serves to be the evidence used to detect spoofing behavior. This frequency variance is governed by a special case of Bernoulli distribution in the formula below:

$$\sigma^2 = (\theta_2 - \theta_1)^2 p(1-p)$$

where p is the probability of changing from one state to the other and θ_1 and θ_2 are the values of the two states. The variance reaches its maximum value for a set of models when the probably of changing a state is 0.5. The variance is also quadratically related to the difference of the models. The intuition from this formula is that two things are contributing to variance used by the STFT analysis technique. The first is the value of p, which in our application represents how even of a mix of frames from both the attacker and the victim exist in a given STFT window. If there are no spoofed frames in a

window, then the chance of going to the second state (governed by 1-p) is zero and which also reduces the variance of the observed signal to zero. The second is how different the two signal strengths are. If the observed signal from an attacker and victim can be modeled by the same strength, then the variance will go to zero. Conversely, the variance increases quadratically as the signal difference between the two states grows.

Therefore, it is the interference between the two distinct statistical distributions which generates the illumination of a higher frequency in the STFT combined with the random intervals of interleaving which spreads this illumination across many high and low frequencies creating the characteristic vertical band of a spoof in the STFT color-graph.

3.3 Chi-Square Signal Strength Analysis Technique

At Rutgers University's WINLAB, PhD candidate Qing Li and Prof Wade Trappe have designed a system that attempts to detect a spoof by using a chi-square test statistic performed on a sliding window of 250 signal strength values [17].

The chi-square approach builds a histogram for the "profile" of a device with 100 bins representing the RSSI values from 0 to 100. When a new window is evaluated, a count is totaled for each bin based on the observed RSSIs of the window and a chi square statistical test is performed. The calculation used this formal,

$$\chi^2 = \sum_{j=1}^{k} \frac{(O_j - E_j)^2}{E_j} \quad ,$$

where k is the number of bins (100 in our case), O is the observed count in a specific bin and E is the expected count in a specific bin based on the distribution profile of the device. The null hypothesis for this test statistic is that the population of the profile window and the test window are the same. The chi-square approach uses a probability of error threshold of $\alpha = 0.01$. If the chi-square value exceeds the critical value of 135.81 (99 degrees of freedom, $\alpha = 0.01$), then an alert has occurred.

To compare the efficacies of the two signal strength techniques, SSFA and the chi-square approach were run simultaneously against the model simulations preformed earlier on the SSFA. In these simulations, the first 250-frame window was used as the profile distribution and all future windows were compared against it. During simulations, I deliberately avoid placing spoof attacks during this first window.

Figure 26 illustrates the performance of techniques in the scenario with a spoof attack, occurring between t=5 and t=6, characterized by a base signal without variation. In it, both techniques are successful in detecting the spoof. Since it is a window-based detection scheme, the chi-square approach concludes that an attack as occurred between the times of 4.99 and 7.48 seconds, while SSFA concludes the spoof is occurring at 5.2 seconds. There is again clearly a slight delay in SSFA conclusion due to the need to accumulate enough evidence to cross the high-frequency variance threshold (as explained earlier in this chapter).

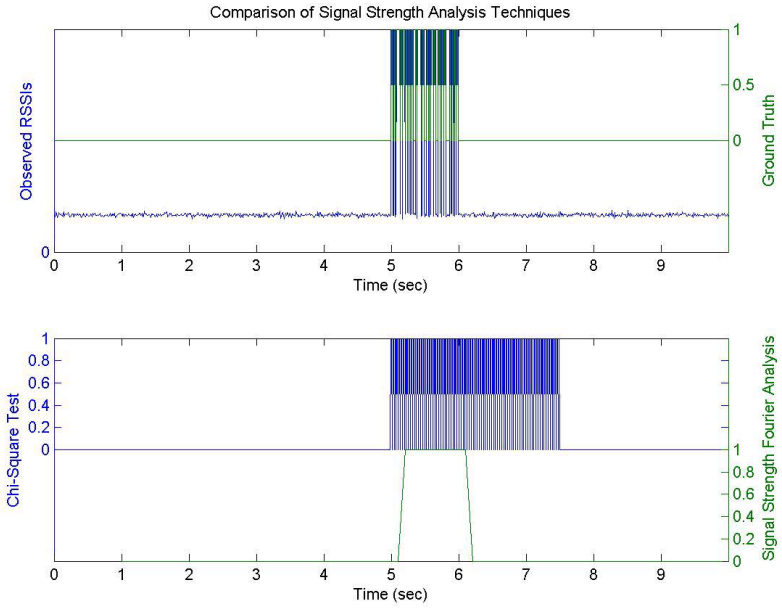

Figure 26. Comparison of Signal Strength Analysis Techniques Against Base Plot without Variation with Spoof

Figure 27 depicts the same spoof attack with a legitimate signal characterized by a base signal with sinusoidal fluctuation to represent gradual natural environmental variation. This scenario illustrates the weakness of the chi-square approach. The profile

generated for this device in the first window causes many false positives to occur for the chi-square approach while SSFA is again able accurately detect the presence of a spoof between 5 and 6 seconds. This scenario will become very important later in the experimentation chapter.

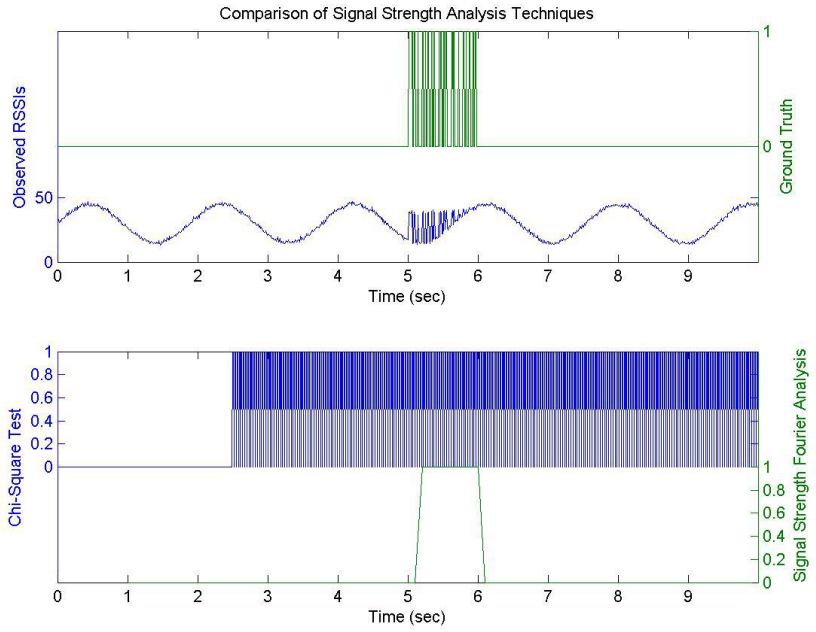

Figure 27. Comparison of Signal Strength Analysis Techniques Against Base Plot with Simulated Natural Variation with Spoof

Figure 28 depicts ARM behavior with a spoof. In this case, it appears that both techniques have positively detected the attack (SSFA at 5.3 seconds and chi-square between 4.99 and 7.48 seconds).

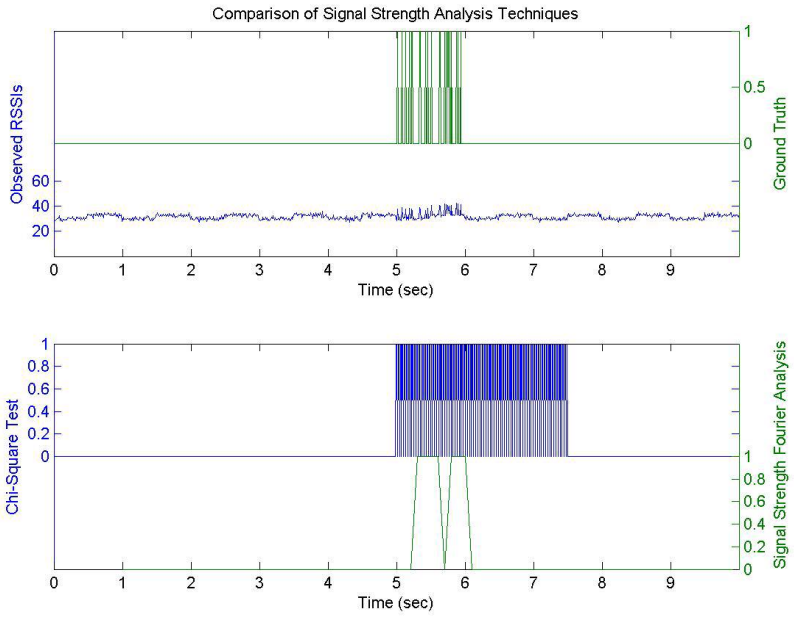

Figure 28. Comparison of Signal Strength Analysis Techniques Against Base Plot with Simulated ARM-like Variation with Spoof

The next scenario demonstrates when the chi-square approach clearly defeats the SSFA. This scenario involves a steady average legitimate signal with a relatively large amount of noise depicted in Figure 29 below. By increasing the level of noise by a factor of 3 over the previous iteration of a steady signal without variation, the chi-square approach still successfully picks out the window with the spoof in it, while SSFA suffers from numerous false positives.

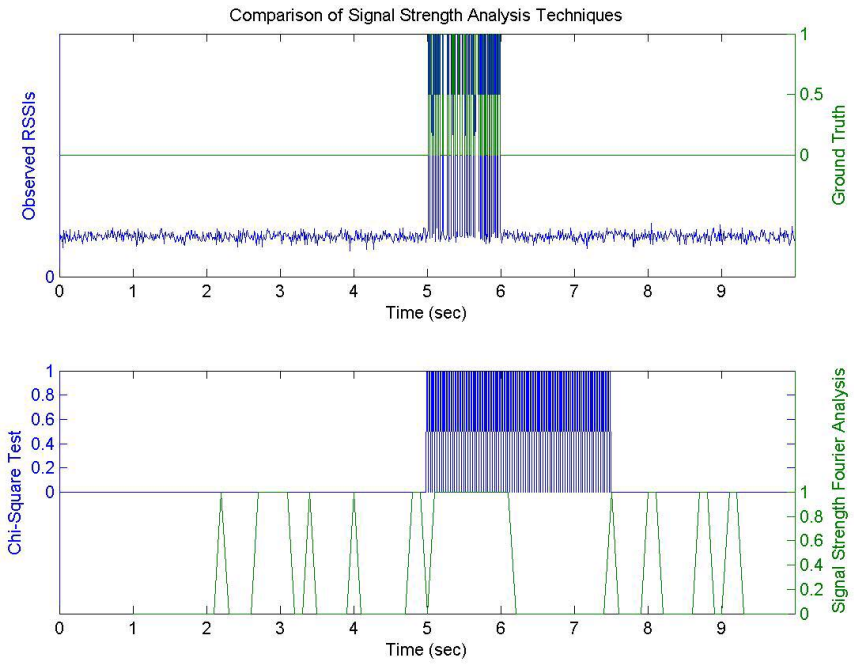

Figure 29. Comparison of Signal Strength Analysis Techniques Against Base Plot with Large Level of Gaussian Noise with Spoof

Finally the sensitivity of each signal strength technique was compared using the same Matlab code of a steady base signal without variation. In successive simulations the amount of spoofed frames was incrementally reduced to zero to determine at what point each technique could no longer detect the spoof. The simulations demonstrated that the chi-square approach appears to drop out at 22 spoofed frames in the 100 frame window, while SSFA could still successfully pick out a spoof with only about 4 spoofed frames as evidence in a 100 frame window.

3.4 Signal Strength Analysis Summary

As demonstrated by the discussion in the last section, the chi-square approach can withstand greater Gaussian noise than SSFA, if the mean of the legitimate signal does not fluctuate. However, if there is fluctuation, even very large fluctuation, SSFA can still accurately pick out the spoof attacks while the chi-square approach generates many false positives. Additionally, SSFA is significantly more sensitive to smaller amounts of spoofing evidence.

Chapter 4: Experimentation

4.1 Setup

In order to compare the performance of the proposed spoof detection techniques, the spoof attack traces from the dataset in [9] were used. The dataset was in the form of dozens of pcap capture files created using tcpdump. Each capture file contained approximately 20 spoof attacks on the four devices usually taking place during a period of about 10 minutes.

To process these data files a custom C program which relies on the libpcap utilities was written to process wireless networking frames [18]. This program, *wifi_parser*, enables the user to either read from a static pcap capture file or live data directly from a wireless interface. The user can additionally select which fields from either the 802.11 header, or the Prism (physical layer) header, which is added to each frame by a wireless card containing a Prism chipset. The output of this program is a single line of user-defined output for each frame processed. It was necessary to write a custom program because the two major text-based wireless networking parsers, Tcpdump and Tethereal, do not allow easy isolation of many header fields [16] [24]. The

wifi_parser help page below further describes some of the functionality built into this tool.

```
[root@sudi-dhcp-139 devel]# ./wifi_parser -h

Usage: ./wifi_parser

  options:
   -c count       number of packets to count before exiting
   -i interface   wireless network interface (default: ath0)
   -r filename    read input from filename (supercedes -i)
   -f filter      input capture filter
   -s selection   header selector
   -p prism       source contains prism header
   -h             this help

      header selector options:
      Prism:
      h  host time
      m  mac time
      c  channel
      i  rssi
      n  signal quality
      g  signal
      n  noise
      r  rate
      x  istx
      802.11:
      t  timestamp
      l  framelen
      u  duration
      f  frametype
      y  subtype
      d  destination
      s  source
      b  bssid
      q  sequence numbers
      a  flags
         0:0:0:0:0:0:0:0
         | | | | | | | \_Order
         | | | | | | \___WEP
         | | | | | \_____More Data
         | | | | \_______Pwr Mgt
         | | | \_________Retry
         | | \___________More Frag
         | \_____________From DS
         \_______________To DS
```

Below is example output from wifi_parser:

```
[root@sudi-dhcp-139 testing]# ./wifi_parser -p -r mapreceiver100000.cap -s yaibstq -c 100
1142390417.842684  15 0_0_0_0_0_0_0_0    Beacon bssid: 000b868139c8 s: 000b868139c8 seq: 2811
1142390418.190113  23 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 526
1142390418.191141  26 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 527
1142390418.192145  25 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 528
1142390418.193152  26 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 529
1142390418.194214  24 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 530
1142390418.195346  24 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 531
1142390418.196372  26 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 532
```

```
1142390418.198596  24 0_0_0_0_0_0_0_0    Beacon bssid: 000b8680ff68 s: 000b8680ff68 seq: 533
1142390418.199593  25 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 534
1142390418.200652  26 0_1_0_0_0_0_0_0      Data bssid: 000b8680ff68 s: 0009e9b7400a seq: 535
```

After parsing the capture files with wifi_parser, a Bash shell script was used to sort the arrival times, RSSIs and sequence numbers into separate files by source address. Each of these files was then individually analyzed using the various aforementioned spoof detection techniques.

4.2 Results

To present the results of the experiments, a detailed discussion of the results of the first experiment will be followed by a summary of the results for all 12 experiments. Finally, the experimentation summary will include discussion of several notable cases of interest.

Detailed Results of Experiment 1

Device 000b868139c8
- *Experiencing five spoofs (at t = 144.2063, t = 198.0876, t = 288.3498, t = 310.1737, t= 532.9158):*

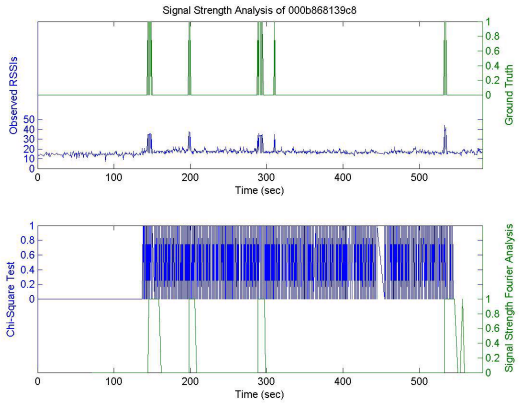

Figure 30. Comparison of Signal Strength Analysis Techniques for Device 000b868139c8
(Above graph is plot of RSSIs, below graph is comparison of Chi-square technique (top) and SSFA (bottom))

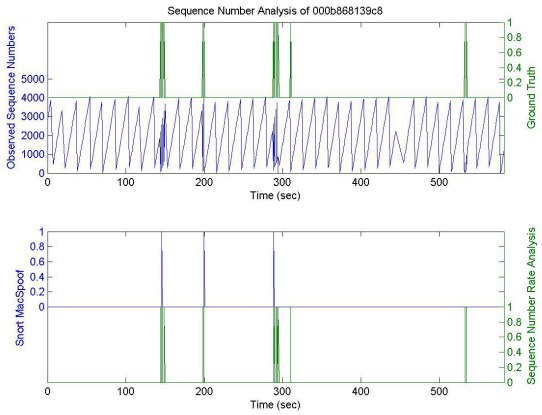

Figure 31. Comparison of Sequence Number Analysis Techniques for Device 000b868139c8
(Above graph is plot of sequence numbers, below graph is comparison of Snort-wireless MacSpoof (top) and SNRA (bottom))

	# True Positives	# False Negatives	#False Positives
SSFA	4 / 5	1 / 5	0 / 238 steps
Chi-square	5 / 5	0 / 5	3 / 8 windows
SNRA	5 / 5	0 / 5	0 / 2381 frames
MacSpoof	3 / 5	2 / 5	0 / 2381 frames

Explanation:

In this first example, SSFA is able to pick out four of the five attacks in Figure 30 while the chi-square approach correctly identifies the windows of all five of the attacks, but produces three false positives of the remaining four 250 frame windows. This result is to be expected due to the shift in the mean of the signal at around t = 120 seconds. If the experiment were to run longer continuing same signal strength mean that existed for the

last 80% of the experiment, there would be more false positives. The space on the end where chi-square doesn't produce a false positive is simply due to the fact that window had not accumulated 250 frames yet.

In Figure 31, both sequence analysis techniques are very precise in their conclusions, however MacSpoof misses two smaller attacks which the SNRA performs without flaw. It is important to reiterate here that these techniques work well in these experiments because the attacks that were used were susceptible to sequence number analysis.

Device 000b8680ff68:

- Experiencing six spoofs (at t = 62.7232, t = 125.2542, t = 166.0152, t = 212.6034, t= 259.2065, t = 461.5391):

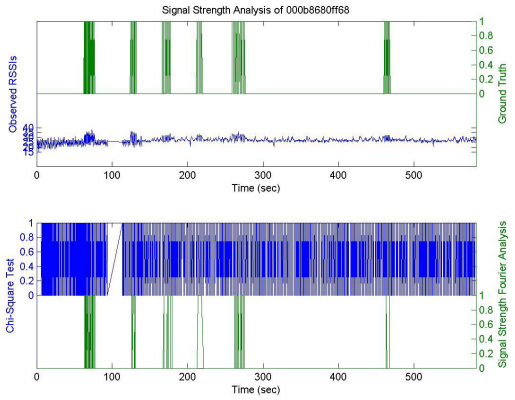

Figure 32. Comparison of Signal Strength Analysis Techniques for Device 000b8680ff68
(Above graph is plot of RSSIs, below graph is comparison of Chi-square technique (top) and SSFA (bottom))

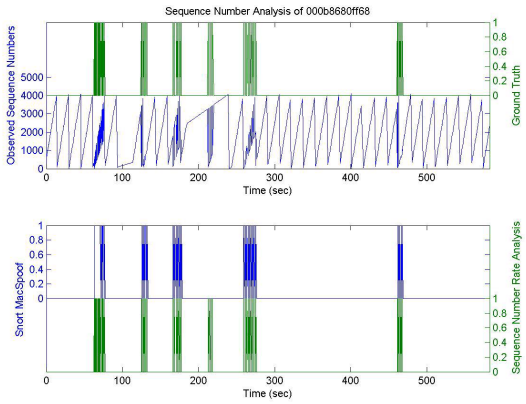

Figure 33. Comparison of Sequence Number Analysis Techniques for Device 000b8680ff68
(Above graph is plot of sequence numbers, below graph is comparison of Snort-wireless MacSpoof (top) and SNRA (bottom))

	# True Positives	# False Negatives	#False Positives
SSFA	6 / 6	0 / 6	0 / 1202 steps
Chi-square	6 / 6	0 / 6	41 / 47 windows
SNRA	6 / 6	0 / 6	0 / 12021 frames
MacSpoof	5 / 6	1 / 6	0 / 12021 frames

Explanation:

Figure 32 depicts an example of spoof attacks with only minor signal strength difference from the base signal. Like the previous example, due to the calibration drift of the legitimate device, the chi-square technique generates false positives for every window after the initial reference window. Meanwhile, the SSFA is able to successfully pick out each attack perfectly.

In Figure 33, the performance of MacSpoof and Sequence Number Rate Analysis is close, however there is one attack that MacSpoof misses.

Device 000b868138b8

- *Experiencing three spoofs (at t = 0.039752, t = 46.4842, t = 580.9178):*

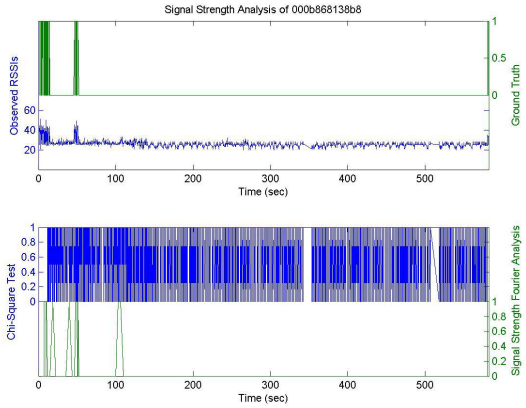

Figure 34. Comparison of Signal Strength Analysis Techniques for Device 000b868138b8
(Above graph is plot of RSSIs, below graph is comparison of Chi-square technique (top) and SSFA (bottom))

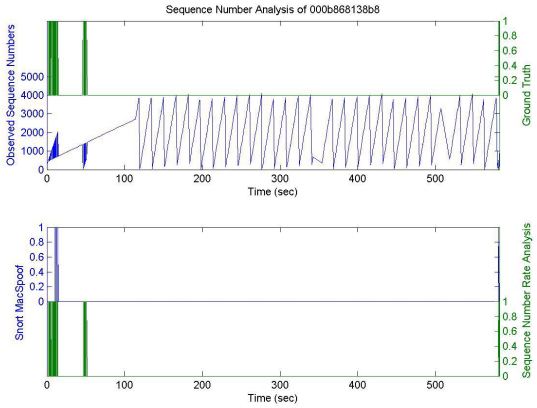

Figure 35. Comparison of Sequence Number Analysis Techniques for Device 000b868138b8
(Above graph is plot of sequence numbers, below graph is comparison of Snort-wireless MacSpoof (top) and SNRA (bottom))

	# True Positives	# False Negatives	#False Positives
SSFA	3 / 3	0 / 3	1 / 927 steps
Chi-square	3 / 3	0 / 3	33 / 36 windows
SNRA	3 / 3	0 / 3	0 / 9278 frames
MacSpoof	2 / 3	1 / 3	0 / 9278 frames

Explanation :

Figure 34 illustrates a slew of false positives for chi-square technique versus one false positive for SSFA. Note there is an attack beginning at the very end of the experiment window. In Figure 35, SNRA picks out every attack, while MacSpoof is only about to see two of the three.

Device 000b8680e4e8:

- Experiencing five spoofs (at t = 237.365044, t = 368.419378, t = 414.982608, t = 486.31211, t = 544.536608):

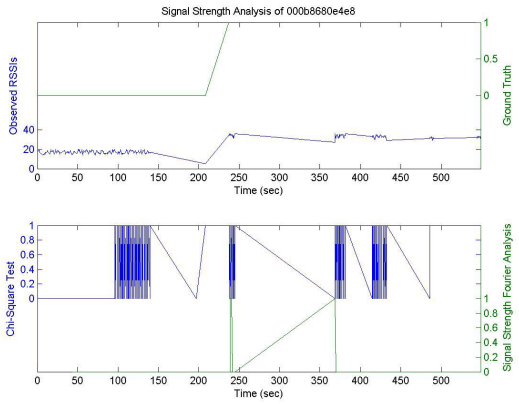

Figure 36. Comparison of Signal Strength Analysis Techniques for Device 000b8680e4e8
(Above graph is plot of RSSIs, below graph is comparison of Chi-square technique (top) and SSFA (bottom))

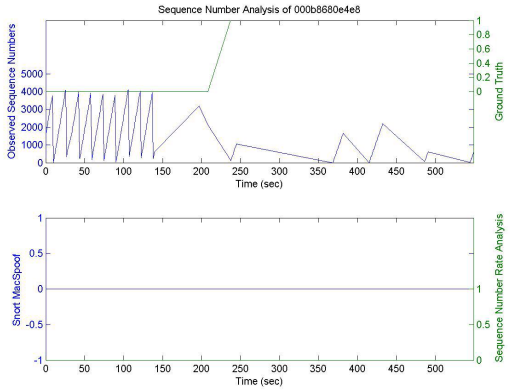

Figure 37. Comparison of Sequence Number Analysis Techniques for Device 000b8680e4e8
(Above graph is plot of sequence numbers, below graph is comparison of Snort-wireless MacSpoof (top) and SNRA (bottom))

	# True Positives	# False Negatives	#False Positives
SSFA	2 / 5	3 / 5	0 / 84 steps
Chi-square	4 / 5	1 / 5	1 / 3 windows
SNRA	0 / 5	5 / 5	0 / 840 frames
MacSpoof	0 / 5	5 / 5	0 / 840 frames

Explanation:

The behavior in this experiment depicts an example of the hidden node phenomenon. After about 140 seconds, the legitimate device is no longer heard for the remainder of the experiment. Another node which was closer was most likely drowning out this hidden node. In any event, the results are worth studying because this behavior will always be a challenge to any sensor-based system.

In Figure 37, both sequence number techniques failed to detect anything simply because too much time had expired between legitimate frames and attacker frames. The signal strength techniques fared better because time is not part of the analysis. The first

spoof frame which is transmitted almost 90 seconds after the last legitimate frame appears to each window count-based technique as a frame which immediately follows the last legitimate frame.

The chi-square technique predictably detects a statistical aberration for every frame window it analyzes after the reference window. It is correct on most of the attacks but simply because all of the remaining windows only contain attack frames. It misses the final attack because it has simply not gathered enough data to make a determination about the final window.

A key ingredient to the success of both of the sequence number analysis techniques and SSFA is that both the attacker and the target must be transmitting frames at about the same time. It is the interleaving that creates the gaps in sequence number analysis and higher frequencies in the STFT.

Summary of Experiment 1

19 Spoof Attacks	# TP	TP Rate	# FN	FN Rate	#FP	FP Rate
SSFA	15	79%	4	21%	1 / 2,452	0%
Chi-square	18	79%	1	5%	77 / 94	82%
SNRA	14	74%	5	26%	0 / 24,520	0%
MacSpoof	10	53%	9	47%	0 / 24,520	0%

Summary of Results 1-12

143 Spoof Attacks	# TP	TP Rate	# FN	FN Rate	#FP	FP Rate
SSFA	115	80.42%	28	19.58%	22 / 42,756	0.05%
Chi-square	140	98.59%	3	1.41%	1418 / 1561	90.84%
SNRA	122	85.31%	21	14.69%	1 / 427,752	0.0002%
MacSpoof	118	82.52%	25	17.48%	92 / 427,752	0.02%

The table above summarized the detection or True Positive (TP) rate, missed detection or False Negative (FN) rate, and False Positive (FP) rate. Between the two signal strength analysis techniques, the chi-square technique has a significantly higher detection rate, however, its FP rate is so high it renders it useless. Generally, four out of five spoofs are correctly detected using the SSFA.

The performance of the sequence number analysis techniques is much closer, however SNRA performs slightly better in every category.

One device all of the techniques had trouble with was 000b8680e4e8 simply because its signal strength was so low it was very hard for the sensor to hear and when the sensor cannot hear the legitimate signal it makes it very difficult to detect a spoof. In fact, the vast majority of all of the missed detections in all of the experiments were from this device. By removing this hard-to-hear device from the performance statistics as illustrated in the table below, the detection rates improve dramatically while the false positive rates remain the same.

114 Spoof Attacks	# TP	TP Rate	# FN	FN Rate	#FP	FP Rate
SSFA	107	93.86%	7	6.14%	21 / 41,819	0.05%
Chi-square	113	99.12%	1	0.88%	1418 / 1556	91.13%
SNRA	111	97.37%	3	2.63%	1 / 418,334	0.0002%
MacSpoof	108	94.74%	6	5.26%	85 / 418,334	0.02%

The chi-square technique performed poorly in these experiments. In the adjusted performance statistics the SSFA achieved a 93.86% detection rate however still suffered from an expected false positive occurring once every 1991 window steps or about 19910 frames. The Sequence Number Rate Analysis Technique did exceptionally well with an adjusted detection rate of 97.37% and a false positive only occurring every 418,334 frames.

Some experiments contained traces of attacks which demonstrated some items of interest which are worth drawing attention to. The most striking of these are the examples of where SSFA correctly picks out spoofs that are not apparent to the human eye, such as the trace of device 000b8680ff68 in experiment #4 depicted in Figure 38.

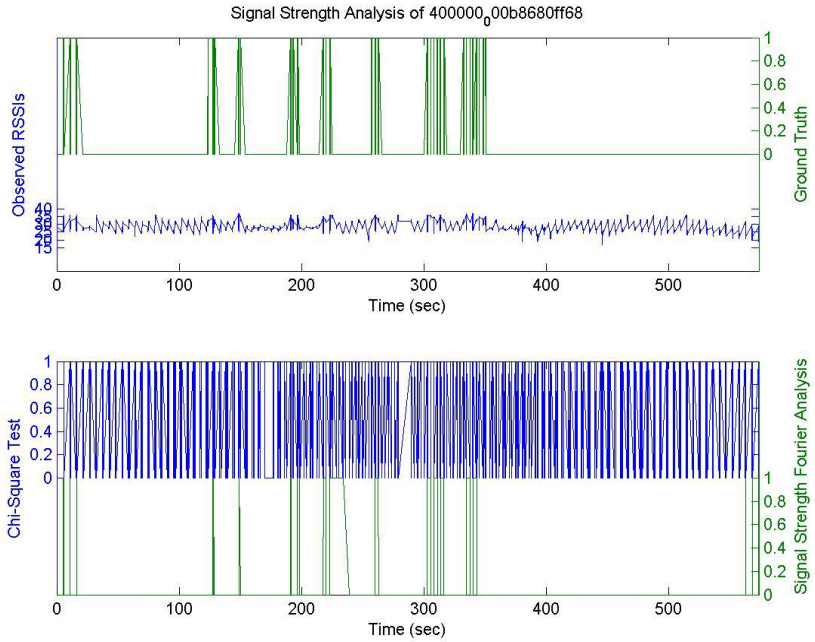

Figure 38. Comparison of Signal Strength Analysis Techniques on Plot with Imperceptible Spoofs, Experiment #4, 000b8680ff68

Figure 39 depicts another remarkably successful spoof detection using SSFA. This time, again, it was of the device 000b8680ff68, but in experiment #8.

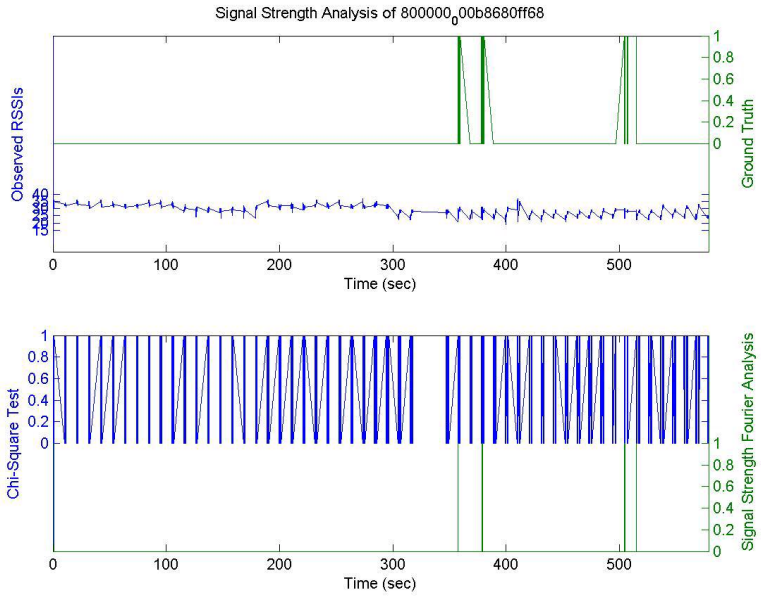

Figure 39. Comparison of Signal Strength Analysis Techniques on Plot with Imperceptible Spoofs, Experiment 8, 000b8680ff68

The analysis of 000b868139c8 in experiment #7 depicted in Figure 40 illustrates the breakdown of MacSpoof when dealing with a trace that is experiencing large amounts of natural loss.

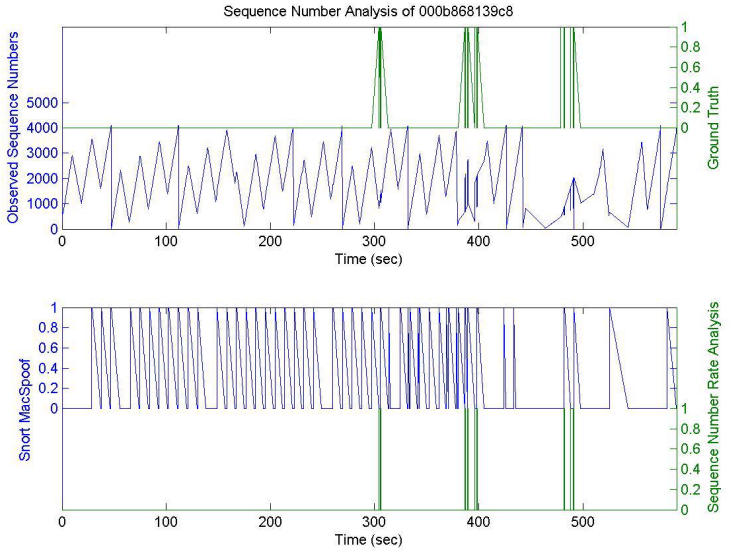

Figure 40. Comparison of Sequence Number Analysis Techniques, Experiment #7 000b868139c8

Figure 41 illustrates one of the few examples where the noise level was too high for STFT to work. The one spoof attack is picked out at the beginning; however, it is followed by quite a few false positives.

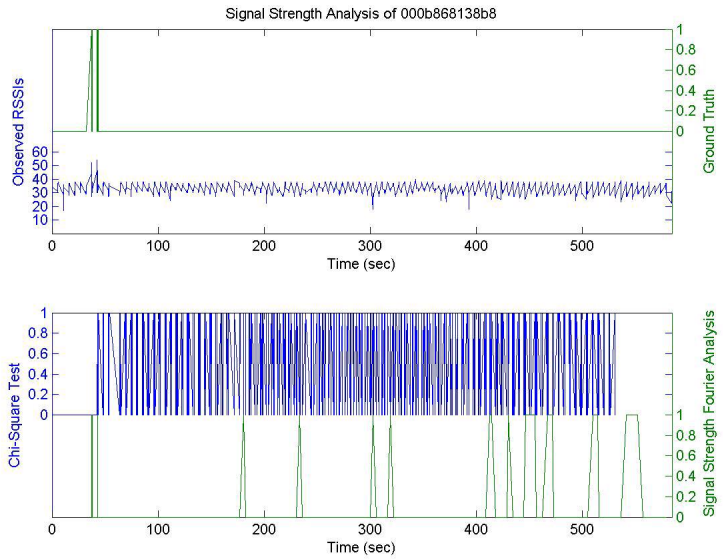

Figure 41. Comparison of Signal Strength Analysis Techniques, Experiment #4, 000b868138b8

Aside from the infrequent missed detection or false positive, SNRA rarely produced many errors.

Finally, in a trace without any attacks, conducted outside the above experiments, was the capture of Automated Power Management behavior in the wild. As can been seen in Figure 42, SSFA handles the abrupt changes in signal mean well without throwing a single false positive.

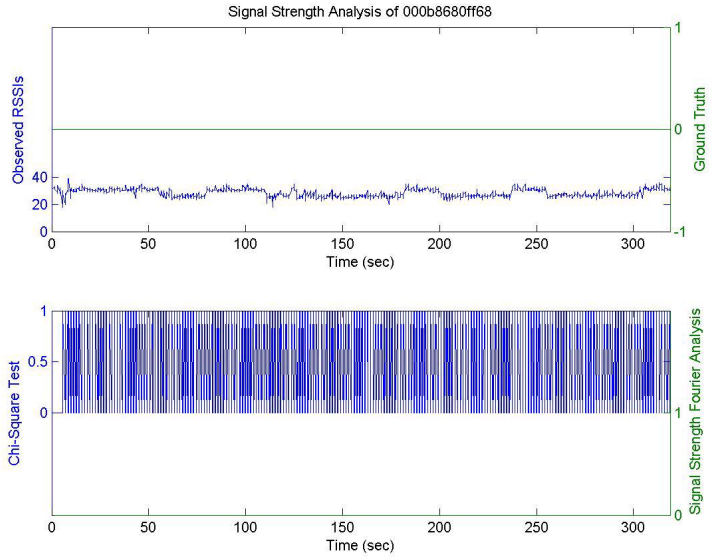

Figure 42. Comparison of Signal Strength Analysis Techniques Against a Base Signal Experiencing a Large Amount of Automated Power Management.

4.3 Overall Experimentation Summary

SSFA for using signal strength to detect spoofs convincingly outperformed a straight statistical approach as demonstrated by the chi-square approach. The SNRA also outperformed the open-source networking security benchmark, Snort-wireless's MacSpoof. In some instances, SSFA could correctly identify a spoof attack where human inspection would have failed. This is a very remarkable accomplishment.

Chapter 5: Conclusions

5.1 Summary

The paper has demonstrated both in simulation and in experimentation that the Sequence Number Rate Analysis (SNRA) technique performed better in detection and false positive rates than Snort-wireless's MacSpoof. SNRA accomplished this feat by calculating a frame transmission rate using the difference of sequence numbers divided by the difference of arrival times of consecutive frames. When this rate surpasses a theoretical frame transmission rate for 802.11b, SNRA concludes a spoof has occurred. The experimental results of the previous chapter confirm the superiority of the proposed technique over MacSpoof in most realistic situations.

Additionally, the paper has demonstrated that using the Short-Term Fourier Transform, the Signal Strength Fourier Analysis (SSFA) could detect spoofs of stationary devices using signal strength as evidence with greater accuracy and far fewer false positives than another research project. This technique proved that it was able to survive the calibration drift and automated power management that pose the greatest challenges to other detection schemes based on signal strength.

5.2 Future Work

A possible extension of this work would be to extend the SSFA to survive mobile transmitting devices or environments with large moving objects with RF reflective surfaces. One thing not covered in detail in this paper was that it is key to SSFA that the legitimate signal source is stationary. This not a far-fetched assumption to make because, more often than not, the victim of the spoofing behavior is an access point which is stationary. The reason moving objects are a problem is because when a device moves through a 3-dimensional space the many paths of emissions going from the source to the sensor can change abruptly due to something such as a reflecting surface. These abrupt changes from multipath also cause vertical bands in the STFT colorgraph and are very difficult to distinguish from authentic attacks.

Additionally, another area of future work would be implementing the hash function for retransmitted frames as described in Chapter 3. If a table of hash values for every sequence number was maintained for each hardware address, then when a frame was sent with its retransmit bit set, a hash value comparison could determine it the frame truly was a legitimate retransmission or a frame attempting to evade a sequence number analysis technique.

References

1. P. Abry and D. Veitch, "Wavelet analysis of long-range-dependent traffic", IEEE Trans. on Information Theory 44, 1, 2—15, 1998.

2. AirDefense, Inc. AirDefense. http://www.airdefense.net

3. AirMagnet INC. AirMagnet. http://www.airmagnet.com/

4. A. Arbaugh, et al. "Your 802.11 Network has No Clothes," In First IEEE International Conference on Wireless LANs and Home Networks, December, 2001.

5. Aruba Networks, "Dartmouth Goes to Aruba to Build Nation's Largest University Wi-Fi Network," http://www.arubanetworks.com/solutions/case-studies/dartmouth.php

6. P. Bahl and V. N. Padmanabhan. "RADAR: An in-building RFbased user location and tracking system," In Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), volume 2, pages 775–784, Tel Aviv, Israel, Mar. 2000.

7. N. Borisov, et al., "Intercepting Mobile Communications: The Insecurity of 802.11," Seventh Annual International Conference on Mobile Computing and Networking, July 2001.

8. A.A. Cardenas, S.Radosavac and J.S. Baras, "Detection and Prevention of MAC Layer Misbehavior in Ad Hoc Networks," in SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks, Washington DC, 2004.

9. U. Deshpande, T. Henderson and D. Kotz. "Channel Sampling Strategies for Monitoring Wireless Networks," In Proceedings of the Second International Workshop On Wireless Network Measurement (WiNMee). IEEE Computer Society Press. April 2006.

10. J. Edney and W. Arbaugh, "Real 802.11 Security----Wi-Fi Protected Access and 802.11i," Addison Wesley, July, 2003.

11. S. Felis , J. Quittek and L. Eggert, "Measurement-Based Wireless LAN Troubleshooting," Proceedings of First Workshop on Wireless Network Measurements , 3 April 2005, Riva del Garda, Trentino, Italy.

12. M. Gast, 802.11 Wireless Networks, O'Reilly & Associates, Inc. (2002).

13. A. Haeberlen, et al., "Practical robust localization over large-scale 802.11 wireless

networks", MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking, September 2004.

14. P. Huang , A. Feldmann , W. Willinger, A non-instrusive, wavelet-based approach to detecting network performance problems, Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, November, 2001.

15. IEEE 802.11-1999 standard specification.

16. V. Jacobson, C. Leres and S. McCanne. tcpdump. http://www.tcpdump.org/

17. Q. Li, W. Trappe. "Light-weight, Relationship-based MAC Layer Defenses for Wireless Networks," WINLAB Research Review, Rutgers University, December 6 2005.

18. libpcap utility. http://sourceforge.net/projects/libpcap

19. W. Leland , M. Taqqu , W. Willinger , D. Wilson, "On the self-similar nature of Ethernet traffic," Conference proceedings on Communications architectures, protocols and applications, p.183-193, September, 1993.

20. Y. Lim, T. Schmoyer, J. Levin and H.L. Owen. Wireless Intrusion Detection and Response. Proc. IEEE Workshop on Information Assurance, United States Military Academy, West Point, NY, USA, June 18–20, 2003, pp.68–75.

21. N. Lomb, "Least-squares frequency analysis of unequally spaced data," Astrophysics and Space Science 39, 447—462, 1976.

22. M. Lynn and R. Baird. "Advanced 802.11 Attack." Black Hat Briefings, July 2002.

23. A. Oppenheim, and R. Schafer, Discrete-Time Signal Processing, Upper Saddle River, NJ: Prentice-Hall, 1999, pp 468-471.

24. A. Orebaugh, G. Morris, E. Warnicke and G. Ramirez. Ethereal Packet Sniffing. Syngress Publishing, February 2004.

25. C. Partridge, et al. "Using signal processing to analyze wireless data traffic", International Conference on Mobile Computing and Networking, 2002.

26. M. Raya, J. Hubaux, and I. Aad, 2004. "DOMINO: a system to detect greedy behavior in IEEE 802.11 hotspots," In Proceedings of the 2nd international Conference on Mobile Systems, Applications, and Services (Boston, MA, USA, June 06 - 09, 2004). MobiSys '04. ACM Press, New York, NY, 84-97.

27. S. Qian and D. Chen, Joint Time-Frequency Analysis: Method and Application, Prentice Hall, 1996.

28. R. Rivest, "The MD5 message-digest algorithm," IETF Network Working Group, RFC 1321, April 1992.

29. Y. Rong, S.K. Lee, and H.A. Choi, "Detecting stations cheating on backoff rules in 802.11 networks using sequential analysis," Proceedings from IEEE INFOCOM 2006.

30. T. Roos, P. Myllymaki, H. Tirri, P. Misikangas, and J. Sievanan. "A probabilistic approach to WLAN user location estimation," International Journal of Wireless Information Networks, 9(3), July 2002.

31. Snort-wireless 802.11 intrusion detection system. http://snortwireless.org/

32. P.D. Welch, "The use of fast fourier transform for estimation of power spectra: A method based on time averaging over short, modified periodograms," IEEE Trans. on Audio Electroacoustics AU-15, 70—73, 1967.

33. J. Wright. Detecting Wireless LAN MAC Address Spoofing. White Paper, January 2003.

34. J. Wright. "Layer 2 Analysis of WLAN Discovery Applications for Intrusion Detection", http://home.jwu.edu/jwright/papers/l2-wlanids.pdf, 2002.