Channel Sampling Strategies for Monitoring Wireless Networks

Udayan Deshpande, Tristan Henderson, David Kotz
Department of Computer Science,
Dartmouth College, Hanover, NH 03755
{udayan,tristan,dfk}@cs.dartmouth.edu

Abstract—
Monitoring the activity on an IEEE 802.11 network is useful for many applications, such as network management, optimizing deployment, or detecting network attacks. Deploying wireless sniffers to monitor every access point in an enterprise network, however, may be expensive or impractical. Moreover, some applications may require the deployment of multiple sniffers to monitor the numerous channels in an 802.11 network. In this paper, we explore sampling strategies for monitoring multiple channels in 802.11b/g networks. We describe a simple sampling strategy, where each channel is observed for an equal, predetermined length of time, and consider applications where such a strategy might be appropriate. We then introduce a sampling strategy that weights the time spent on each channel according to the number of frames observed on that channel, and compare the two strategies under experimental conditions.

I. Introduction

As IEEE 802.11 "Wi-Fi" networks become an integral part of many enterprises and organizations, it becomes increasingly important to monitor these networks for evidence of problems, attacks or breakdowns. With such networks being used for critical services such as VoWLAN (Voice over Wireless Local Area Networks), any such breakdown or attack could have a serious effect on an organization's productivity.

Most WLAN measurement studies, e.g., [5], [15], [10], have monitored the wired side of Access Points in an infrastructure WLAN using SNMP, syslog and packet sniffing. These techniques enable the monitoring of traffic that has been bridged from the wireless side to the wired side of a network. They do not, however, allow us to gather MAC layer data from the wireless side of the network, data that might contain valuable information regarding misbehavior or attacks. In particular, it is impossible to detect wireless (layer 2) attacks, or problems that only affect the wireless side of the network.

To effectively troubleshoot a wireless network, it is therefore necessary to monitor the wireless side of the network as well. This can be achieved by deploying wireless "sniffers": measurement devices with 802.11 radios placed in "monitor" mode. Such radios will record every 802.11 frame that they hear.

Monitoring all of the traffic on the wireless side of an enterprise WLAN, however, can be expensive and difficult to set up. Since each sniffer's radio has a limited range, one might require as many sniffers as access points in the network. Additional sniffers might be required if the purpose is to monitor areas that have bad coverage. The cost of this large number of sniffers might be prohibitive.

Even if sufficient sniffers are deployed, the spectrum used for 802.11 networks is divided into several channels. For instance, 802.11b/g divides the 2.4GHz spectrum into 14 channels spaced 5MHz apart.[1] Thus to monitor all of the potential 802.11b/g traffic in the area of a particular sniffer might require 14 different radios. Sniffers with multiple radios, such as the Porcupine from Indiana University [2] are large and may be difficult to deploy on a large scale. Moreover, storing or transmitting captured frames from 11 simultaneous channels may be problematic, with each sniffer capable of capturing almost 600Mbps of traffic (assuming 11 channels at maximum 802.11 link-layer rates).

Rather than deploy multiple-radio sniffers, it might be more cost effective to use single-radio sniffers that periodically sample each of the available channels. At any given time, there may be traffic on any of the 802.11b/g channels, and a radio could cycle through these channels, capturing some traffic on one channel before moving on to another.

The simplest way to sample the channels would be to spend an equal amount of time on every channel. If, however, there is no traffic on some channel, it would be wasteful to spend equal time on that channel. Furthermore, to spend time T/n, where T is total cycle time and n is number of channels, on channels that have a small share of traffic may also be wasteful. On the other hand, for particular applications it might be useful to spend time on underutilised channels, for instance for detecting rogue access points that may be "hiding" in lesser-used channels.

Some aspects of wireless monitoring have been studied in the literature (see Section VI). Yeo et al [16] describe an infrastructure for wireless monitoring. The authors identify three challenges of wireless monitoring as

[1] In the USA, the FCC only allows the use of channels 1 to 11, and we only consider those channels in this paper.

0-7803-9550-6/06/$20.00 © 2006 IEEE

Capacity of each Sniffer, Sniffer Placement and Data Collection. We would like to add *Sampling Strategies* as another challenge.

In this paper, we describe a sampling method whose parameters can be adjusted to meet the needs of various applications. This paper is laid out as follows. In Section II we describe different metrics for evaluating the performance of a sampling strategy. Section III outlines some applications that require different sampling strategies. Section IV defines one specific strategy, and some experiments using this strategy are presented in Section V. Section VI discusses related work, and Section VII lists future extensions to this work.

II. Metrics

To determine a sampling strategy, the relative *importance* of a particular channel in relation to other channels needs to be quantified. The importance of a channel may vary with time and depends on the goals of the applications. This concept is similar to the concept of bandwidth demand and there are techniques that fairly allocate bandwidth among flows [7]. The importance of a channel may also be thought of as the "monitoring demand" from that channel. Since this demand may not be known in advance, we may have to estimate the monitoring demand on the various channels using historical data.

This "demand" metric may vary from application to application. In the simplest case, the demand may be shared across all channels, and so an appropriate sampling strategy would be to spend equal time on each channel. In this case, we are sampling the amount of time spent on each channel, and using this to determine the amount of time to be spent on each channel in the next round (although in this case the intervals are equal). We refer to this as an **Equal/Time/Time** sampling strategy. In another example, the metric may be the Frame Count on the channel (an **Equal/Frame Count/Frame Count** strategy).

We may also use this metric to determine a sampling strategy that divides time among the channels in the same proportion as the observed relative frame counts on different channels. We refer to this as a **Proportional/Time/Frame Count** sampling strategy, as we are using the Frame Count to determine the amount of Time on each channel. To know the frame count, we must sample the channel first (in some simple and fair manner). As the traffic volumes on various channels will vary with time, we must periodically update relative frame count proportions that are used to determine the sampling strategy.

Another metric, also time varying, may be the number of clients on a particular channel (a **Proportional/Time/Client Count** sampling strategy). Other metrics include the number of attacks on some channels, number of legitimate access points, or the proportion of vulnerable or important traffic (such as VoIP) in relation to total traffic on some channels. Another sampling strategy could be to remain on a particular channel for as much time as is required to capture n frames, or to observe m clients. These techniques may be useful in "characterization" applications.

III. Channel Sampling Applications

There are many applications of wireless monitoring and more specifically, channel sampling. Monitoring may be used for enabling security, management, characterization, and deployment of 802.11b/g wireless networks. Different applications may require different sampling strategies; we now discuss some of these.

A. Security

The shared nature of wireless networks introduces many security risks and potential attacks, such as denial of service, man-in-the-middle frame injection attacks, or eavesdropping (see the literature for discussion of several 802.11 attacks [6], [9]). One method for detecting these attacks is to capture all frames and recognize particular attack signatures [8]. Another mechanism is to detect the effect of the attack on the network, such as looking for clients that obtain unfair bandwidth allocation [12].

These detection methods may benefit from different kinds of sampling strategies. A signature-based detection technique that depends on capturing every frame of an attack may require continuous monitoring of a particular channel. On the other hand, some attacks, such as those that require flooding, may still be detected even if a channel is sampled.

Some attackers may wish to attack channels that have maximum traffic volume, while others may be motivated to attack channels with high numbers of clients. Sampling strategies that give higher weight to such channels may be of particular use.

It is important to note that if an attacker knows the monitoring system's sampling strategy, the system is vulnerable to the attacker modifying his or her behavior to evade observation. For example, if the attacker knows that the sampling strategy being used is any kind of Proportional/Time Counting sampling strategy, the attacker may introduce traffic on some channels and attack the remaining channels. To circumvent this attack the sampling strategy should involve some random behavior.

B. Management

Problems in wireless networks do not result solely from attacks. There might be RF holes, connectivity problems or authentication problems that a network administrator needs to discover as soon as they occur. Adya et al discusses a management infrastructure that depends on wireless monitoring by clients [4]. A smart sampling strategy

could be used to augment the management techniques described in that paper.

Network managers may wish to monitor the channels to determine QoS being provided to VoIP calls. The sampling metric could be based on the proportion of VoIP traffic.

C. Characterization

Characterizing wireless networks is useful for improving wireless protocols, simulating wireless networks, developing new mobile applications and so forth. Accurate characterization of wireless networks relies on accurate measurement, and so sampling strategies may prove useful. Sampling can increase the geographic channel coverage of sniffers, and reduce the expense of deploying multiple sniffers, one for each channel.

Enterprise wireless networks are becoming increasingly sophisticated, with access points that can automatically alter their power levels and channels to increase coverage and reduce interference. Characterizing such networks can be difficult with statically-configured sniffers. A network measurement infrastructure should adjust dynamically so as to continue to collect relevant data from the network, and an appropriate sampling strategy might be one that focuses on APs' currently-assigned channels.

D. Deployment

Wireless monitoring is useful in the deployment stage of a wireless network, in the "site survey", where AP density requirements are determined. But monitoring is also useful in the post-deployment phase, where sanity checks may need to be performed. For instance, it may be necessary to determine that the correct number of APs have been deployed, or that their channel assignment is correct. Channel-sampling sniffers could be useful for this purpose.

Post-deployment measurement could also be used for capacity planning, e.g., for noticing hotspots that indicate a need to deploy additional APs. In such a case a Proportional/Time/Client Count channel strategy might be appropriate.

IV. Implementation of a sampling strategy

Any sampling strategy needs to be dynamic to meet the needs of the changing wireless network environment. There may be situations that require closer monitoring on some channels than others. For example, after detecting an attacker on a particular channel, that particular channel might need to be monitored more closely.

We propose a strategy for determining sampling time intervals based on Frame Counts, that is, a Proportional/Time/Frame Count sampling strategy. Let n be the number of channels and T be the time for one complete sampling cycle, where a cycle is a period of time in which we visit each channel once. Let $f_{i,j}$ be the number of frames captured on channel i in the cycle j and $t_{i,j}$ be the time spent on channel i in cycle j.

Assume that the first cycle is an equal-time cycle with the radio spending $t_{i,1} = T/n$ time on each channel. At the end of the first cycle, the $\{f_{i,1}\}$ contains the number of frames captured on each channel in the first cycle.

The time to be spent on channel i in the next cycle $j+1$ is based on the proportions observed during cycle j:

$$t_{i,j+1} = \frac{(f_{i,j}/t_{i,j})}{\sum_{i=1}^{n}(f_{i,j}/t_{i,j})} \times T$$

This equation calculates the proportion of frames per unit time that are captured on channel i in iteration j. That ratio is multiplied by the total time T so that the time $t_{i,j+1}$ spent on channel i in the next iteration is in the same proportion.

We call the period that the sniffer spends on one channel an *interval*. It is important to not allow any channel's interval to become too small to capture a frame because then $f_{i,j} = 0$ and all future intervals for channel i will be zero. In our implementation we limit the minimum interval size M. If the proportion allocated to a particular channel is less than the minimum interval, we spend more than time T for the next cycle.

The cycle time T is an important feature. If T is short, correspondingly, the intervals will be small. Each change in channel introduces some overhead with respect to time. If intervals are small the channel switching overhead is significant, but the method reacts quickly to changes in traffic load. If, however, the cycle time is 10sec, the phenomenon might have ended before the correct channel is sampled again.

Therefore, longer cycle times are useful if efficiency is important for an application. Shorter cycle times are recommended, however, if quick reactivity is important.

V. Experimental Results

We have implemented the strategy described above to validate our hypothesis that through the use of smart sampling strategies we can obtain *better* data by some measure. The measure or metric is dependent on the application. In this section, we look at two applications. First, we describe a characterization application in which a higher number of frames captured is *better* than a fewer number of frames. Second, an Intrusion Detection System (IDS) application where a greater number of intrusion alerts is better.

A. Characterization experiment

A.1 Testbed

Our testbed consists of two Intel x86 sniffers with Atheros-based wireless cards. The sniffers ran Linux (Fedora Core 4 with kernel 2.6.14 and the MadWiFi driver)

for our experiments. The two sniffers were placed 90 cm apart in a research lab that had a crowded radio environment with several 802.11 experiments in progress at any given time.

A.2 Experiment

We sniffed the 11 legal channels using the Equal/Time/Time sampling strategy on one sniffer. On another sniffer, we simultaneously sniffed using the Proportional/Time/Frame Count sampling strategy. We experimented with various cycle times $T = \{1.1 \text{ sec}, 2.2 \text{ sec}, \ldots, 22 \text{ sec}\}$. Each of these experiments was run for 10 minutes. The whole process was then repeated, swapping the strategy assignment to sniffer to discount for radio-propagation differences.

A.3 Results

We plot the distribution of time on each channel and the distribution of the count of frames captured on each channel.

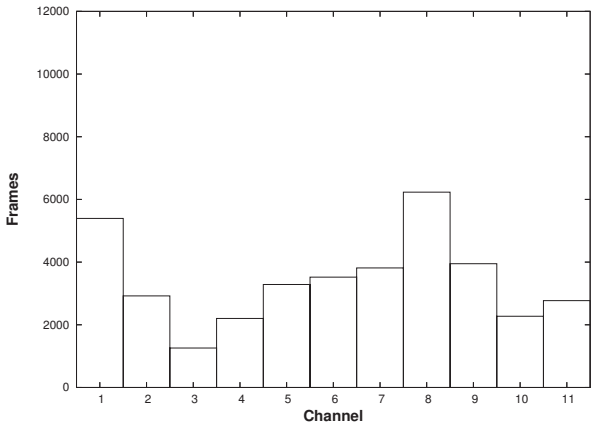

Fig. 1. Equal/Time/Time sampling strategy (T = 11 sec).

Figure 1 shows the number of frames captured on each channel using the Equal/Time/Time sampling strategy. We observe that the Frame Counts varied across channels. This example is one of 20 runs, all of which were similar except for one case described below.

Figure 2 shows that the time spent on each channel was close to being equal. The slight differences in the heights of the columns are due to a lag between our measurement and the time the channel was actually changed.

Figure 3 shows the traffic captured using the Proportional/Time/Frame Count sampling strategy. Note that the peaks (highest frame counts) are higher than in the case of the Equal/Time/Time sampling strategy because the more active channels are allocated more time in the Proportional/Time/Frame Count strategy.

In one of the runs (Figure 5), we observed a huge spike on channel 8. On further investigation we determined that the data was HTTP traffic from one specific client. Using the Equal/Time/Time sampling strategy, about 19 megabytes of traffic was captured in the run that lasted 10 minutes—about twice the volume as in other runs. Simultaneously, however, the sniffer running the Proportional/Time/Frame Count sampling strategy (Figure 7) captured nearly 80 megabytes of data. This volume was approximately 8 times the volume collected by other runs of the Proportional/Time/Frame Count strategy. This observation indicates that the Proportional/Time/Frame Count sampling strategy is indeed successful in maximizing the data capture. The Proportional/Time/Frame Count sampling strategy captured 6 times as many frames on channel 8 than the Equal/Time/Time strategy.

Figure 8 shows that the Proportional/Time/Frame Count sampling strategy adapted the time spent on each channel to the traffic abnormalities while in contrast the Equal/Time/Time strategy maintained equal proportions (Figure 6).

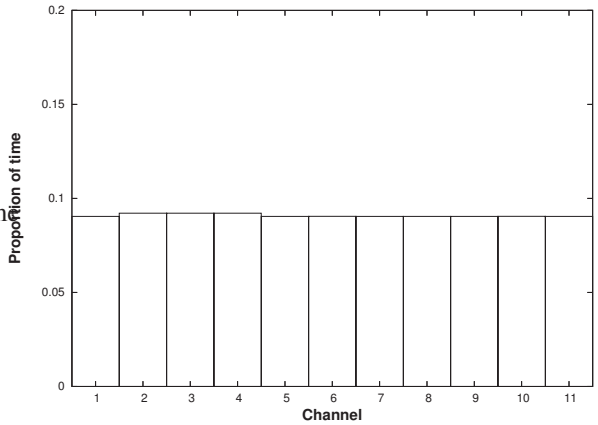

Fig. 2. Confirming equal time spent on each channel ($T = 11sec$).

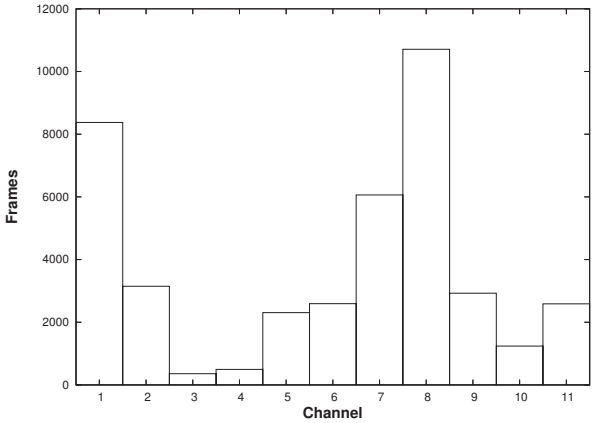

Fig. 3. Traffic Proportional/Time/Frame Count sampling strategy ($T = 11sec$)

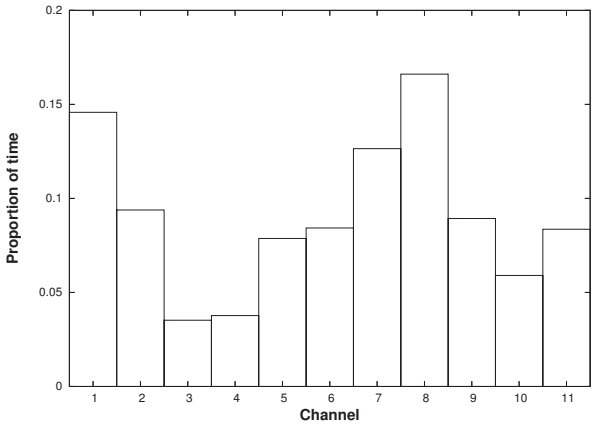

Fig. 4. For Proportional/Time/Frame Count strategy, time spent on each channel is proportional to traffic observed ($T = 11sec$)

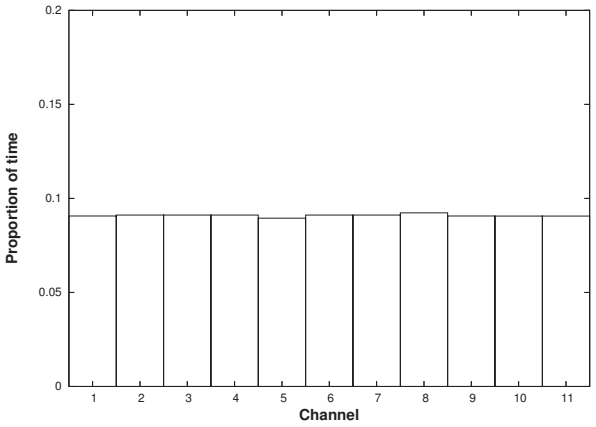

Fig. 6. Equal/Time/Time strategy: Time spent on each channel was approximately the same ($T = 3.3sec$)

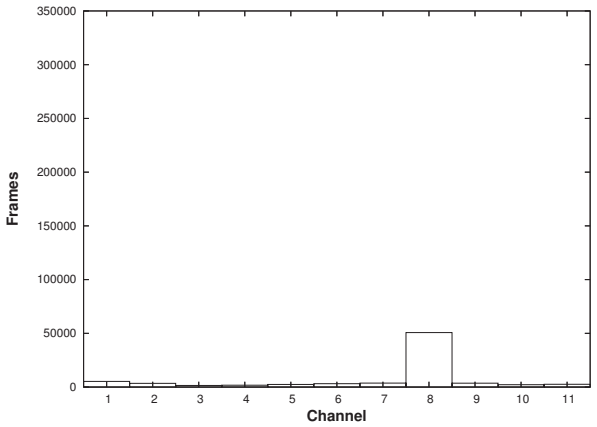

Fig. 5. Equal/Time/Time strategy: Huge traffic spike on channel 8 (T = 3.3 sec.) Note that the scale of the Y-axis is different from the previous plots. Compare this figure to Figure 7.

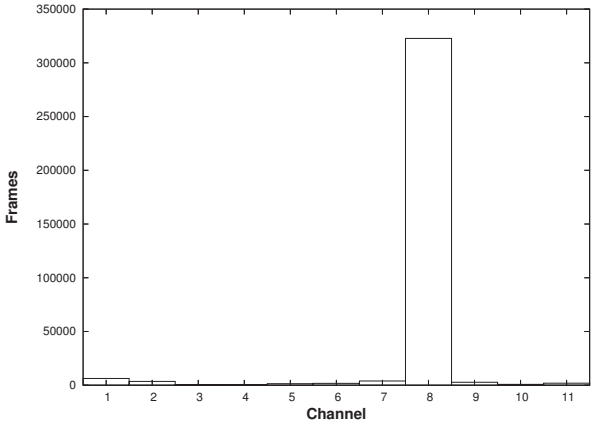

Fig. 7. The Proportional/Time/Frame Count sampling strategy accentuates the spike in traffic volume ($T = 3.3sec$)

B. IDS Experiment

As described in section III-A, one use of sampling is for detecting security breaches in wireless networks. In the case of denial-of-service attacks, a malicious attacker may be motivated to disrupt as much traffic as possible. To achieve this goal, the attacker will need to monitor the channels to determine which channel has the highest load. Once this channel is determined, the attack can be launched on that channel.

To determine the effectiveness of our system in such a scenario, we mimicked this attack strategy. We periodically measured the traffic on each channel and launched an attack on the channel with the highest number of frames. Each attack was of random length (between 0 and 12 seconds) with intervals of random length (between 5 and 12 seconds) between each attack.

We used a Linux laptop as the attacker's machine, running the Auditor distribution of Linux. We ran the deauthentication attack *file2air*, which spoofs the MAC address of an AP and sends flood of deauthenticaton messages to a victim so as to deny service. We configured *file2air* to send a flood of deauthentication frames with interframe intervals of 1 millisecond.

We sampled the 11 802.11b/g channels using the Proportional/Time/Frame Count and the Equal/Time/Time sampling strategies. To detect the attack, we ran the popular IDS tool *snort-wireless* [3] on the traces captured using the two sampling strategies.

We observe that snort-wireless detects a greater number of abnormal sequence number gaps (which indicate MAC spoofing) in the traces collected from the Proportional/Time/Frame Count (Figure 9) strategy than the Equal/Time/Time sampling strategy.[2] Snort-wireless also generates more alerts (Figure 10) in the Proportional-Time Frame Count trace. This indicates that the Proportional/Time/Frame Count strategy captured more attack instances than the Equal/Time/Time sampling strategy. A

[2]Note that snort-wireless only flags a deauth attack alert on spoofed deauthentication frames, rather than legitimate deauthentications.

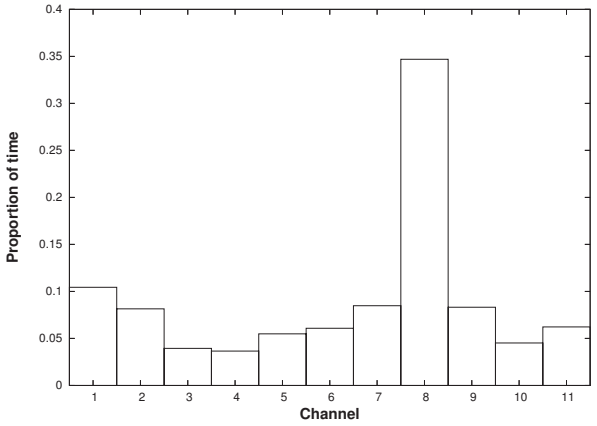

Fig. 8. Time spent on channel 8 was much more than the other channels as the sampling strategy adjusted ($T = 3.3sec$). Note that the scale of the Y-axis for this plot is different from previous time plots.

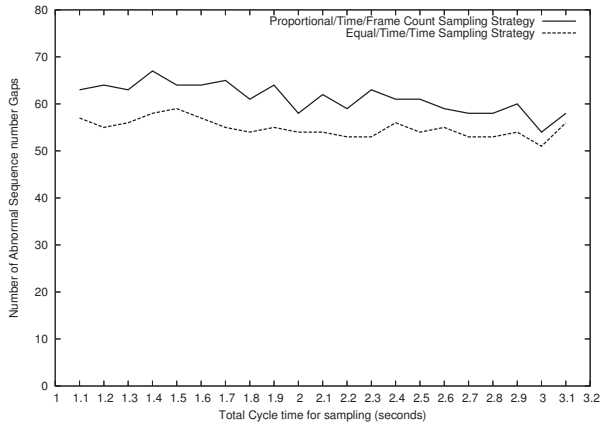

Fig. 9. Number of abnormal sequence number gaps detected by snort-wireless. snort-wireless detects more abnormal sequence number gaps in the trace captured by the Proportional/Time/Frame Count sampling strategy than the Equal/Time/Time strategy.

paired t-test indicates that the two strategies perform differently at the 1% level.

Figures 9 and 10 show the outcome of our experiments, with total cycle times varying from 1.1 seconds to 3.1 seconds, and a minimum interval time of 0.5 seconds in each run. Although there is no trend over the increasing cycle times, snort-wireless consistently detects a greater number of abnormal sequence number gaps and flags more alerts in the the Proportional/Time/Frame Count sampling strategy than the Equal/Time/Time strategy. Our expectation that more attacks would be detected using smart sampling was therefore correct.

VI. Related Work

We mention above some of the many wireless network characterization studies [5], [15], [10] that use wired-side methods to monitor wireless networks. Wireless-side characterization studies are less common, and are

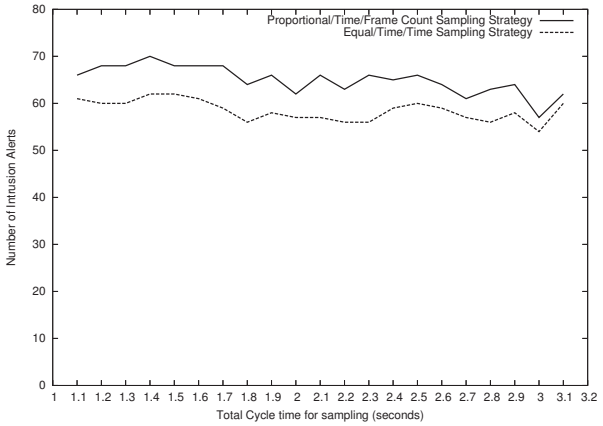

Fig. 10. Number of Alerts flagged by snort-wireless. snort-wireless consistently flags more alerts in the trace captured using the Proportional/Time/Frame Count sampling strategy than the Equal/Time/Time strategy

typically concerned with only measuring the channels on which APs are assigned [11], [14], [17].

We are unaware of any work that looks at channel-sampling or channel-hopping strategies. The popular "war driving" tool, Kismet [1], includes configuration options for channel-hopping sequences and channel-dwell times (channel intervals). The interval times are configured statically, however.

VII. Conclusions and Future work

In this paper we have discussed the motivation for smart channel-sampling strategies, and experimented with a Proportional/Time/Frame Count sampling strategy that spends more time on channels with higher observed frame rates.

The ideas presented in this paper are still at a preliminary stage. There are many sampling strategies that remain to be explored. In the near term, we intend to integrate the channel sampling system into a monitoring system for an infrastructure wireless network. We plan to implement more sampling strategies and explore their usefulness. We are also considering active sampling strategies that dynamically alter their parameters (such as cycle time) depending on the observed traffic, and introducing randomization in channel-hopping sequences and intervals in order to detect an elusive attacker hiding in lesser utilized channels.

A property of 802.11 networks we intend to exploit is *cross-channel interference*. Due to channel overlap, even if a radio is on a particular channel, signals from adjacent channels can be observed [13]. Sampling can be optimized by using these leakage signals from other channels. Ideally, we will not need to hop to some channels nearly as often.

VIII. Acknowledgements

This research program is a part of the Institute for Security Technology Studies, supported under Award number NBCH2050002 from the U.S. Department of Homeland Security, Science and Technology Directorate. Points of view in this document are those of the author(s) and do not necessarily represent the official position of the U.S. Department of Homeland Security or the Science and Technology Directorate.

Thanks to Doug Madory, Bennet Vance, and the anonymous reviewers for useful comments.

References

[1] Kismet 802.11 network detector. http://www.kismetwireless.net/.

[2] Porcupine sixteen-radio high performance wireless device. http://porcupine.iu.edu/.

[3] Snort-wireless 802.11 intrusion detection system. http://snort-wireless.org/.

[4] A. Adya, P. Bahl, R. Chandra, and L. Qiu. Architecture and techniques for diagnosing faults in IEEE 802.11 infrastructure networks. In *Proceedings of ACM MobiCom 2004*, pages 30–44, Philadelphia, PA, Sept. 2004.

[5] A. Balachandran, G. M. Voelker, P. Bahl, and P. V. Rangan. Characterizing user behavior and network performance in a public wireless LAN. In *Proceedings of ACM SIGMETRICS 2002*, pages 195–205, Marina Del Rey, CA, June 2002. ACM Press.

[6] J. Bellardo and S. Savage. 802.11 denial-of-service attacks: Real vulnerabilities and practical solutions. In *Proceedings of the Twelfth USENIX Security Symposium*, pages 15–28, Washington, DC, USA, Aug. 2003. USENIX Association.

[7] D. M. Chiu. Some observations on fairness of bandwidth sharing. In *Fifth IEEE Symposium on Computers and Communications (ISCC)*, pages 125–131, Antibes-Juan les Pins, France, July 2000.

[8] F. Guo and T. cker Chiueh. Sequence number-based MAC address spoof detection. In *Proceedings of the 8th International Symposium on Recent Advances in Intrusion Detection*, pages 309–329, Seattle, WA, USA, Sept. 2005.

[9] C. He and J. C. Mitchell. Security analysis and improvements for IEEE 802.11i. In *Proceedings of the 12th Annual Network and Distributed System Security Symposium*, pages 90–110, San Diego, CA, USA, Feb. 2005.

[10] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. In *Proceedings of ACM MobiCom 2004*, pages 187–201. ACM Press, Sept. 2004.

[11] A. P. Jardosh, K. N. Ramachandran, K. C. Almeroth, and E. M. Belding-Royer. Understanding congestion in IEEE 802.11b wireless networks. In *Proceedings of the 2005 Internet Measurement Conference*, pages 279–292, Berkeley, CA, USA, Oct. 2005.

[12] P. Kyasanur and N. H. Vaidya. Detection and handling of MAC layer misbehavior in wireless networks. In *Proceedings of the International Conference on Dependable Systems and Networks*, San Francisco, CA, USA, June 2003.

[13] A. Mishra, E. Rozner, S. Banerjee, and W. Arbaugh. Exploiting partially overlapping channels in wireless networks: Turning a peril into an advantage. In *Proceedings of the 2005 Internet Measurement Conference*, pages 311–316, Berkeley, CA, USA, Oct. 2005.

[14] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorian. Measurement-based characterization of 802.11 in a hotspot setting. In *Proceedings of the ACM SIGCOMM 2005 W-WIND Workshop*, pages 5–10, Philadelphia, PA, USA, Aug. 2005.

[15] D. Tang and M. Baker. Analysis of a local-area wireless network. In *Proceedings of ACM MobiCom 2000*, pages 1–10, Boston, MA, Aug. 2000. ACM Press.

[16] J. Yeo, M. Youssef, and A. Agrawala. A framework for wireless LAN monitoring and its applications. In *Proceedings of the Third ACM Workshop on Wireless Security (WiSe'04)*, pages 70–79, Philadelphia, PA, Oct. 2004.

[17] J. Yeo, M. Youssef, T. Henderson, and A. Agrawala. An accurate technique for measuring the wireless side of wireless networks. In *Proceedings of the International Workshop on Wireless Traffic Measurements and Modeling*, pages 13–18, Seattle, WA, USA, June 2005.