Abstract—Low power devices such as common wireless router platforms are not capable of performing reliable full packet capture due to resource constraints. In order for such devices to be used to perform link-level measurement on IEEE 802.11 networks, a packet sampling technique is required in order to reliably capture a representative sample of frames. The traditional Berkeley Packet Filter mechanism found in UNIX-like operating systems does not directly support packet sampling as it provides no way of generating pseudo-random numbers and does not allow a filter program to keep state between invocations. This paper explores the use of the IEEE 802.11 Frame Check Sequence as a source of pseudo-random numbers for use when deciding whether to sample a packet. This theory is tested by analysing the distribution of Frame Check Sequences from a large, real world capture. Finally, a BPF program fragment is presented which can be used to efficiently select packets for sampling.

I. INTRODUCTION

IEEE 802.11 [1] has become a popular technology choice for building wide-area wireless networks cheaply and easily. As such, IEEE 802.11 is being used in far more scenarios than it was originally intended. IEEE 802.11 is now being used as the link-layer technology to power a range of networks, from community-driven remote rural networks to urban mesh networks. The size and complexity of such networks is increasing, however the state of wireless link-layer measurement has not improved significantly with these changes.

Accurate link-layer measurement of wireless wide-area networks in a distributed fashion is important. The nature of the shared wireless medium means that a wide-area network can suffer from interference from itself as logically separate links share the same physical medium. As networks get larger and more complex, this effect increases. Performing link-layer measurement at each wireless router would allow for analysis of this effect as well as other interesting measurement opportunities.

Low-power devices used in wide-area wireless community networks are generally unable to process full packet capture at line rates. The overhead involved in copying each packet from kernel- to user-space overwhelms the devices, leading to inaccurate and incomplete captures. Similarly, passive “vicinity-sniffing” [14] gives an inaccurate view of the channel conditions at each actual receiver. Projects such as WiFidelity [11] have attempted to quantify these effects by analysing existing wireless traces and determining the completeness or fidelity of the trace.

Traditionally the overhead involved in copying frames from kernel- to user-space has been mitigated by employing kernel-side filtering to drop packets that are not interesting. The Berkeley Packet Filter (BPF) [7] provides a mechanism for such kernel-side filtering, however it is generally used to filter packets based on fields within the packet, such as source address or destination address. For example, it is easy with BPF to filter all packets destined to a certain host or port.

Generally if a random sample of packets was required, a sample could be taken by randomly selecting packets after they had been passed from kernel- to user-space, for example, by using the standard C library’s rand() function (or similar). This method fails to work on hosts where the overhead of passing all frames from kernel- to user-space is significant, such as on commodity wireless routers. Instead, it would be more appropriate to randomly sample packets before they are passed to user-space. Using the existing BPF mechanism seems to be the obvious way to achieve this.
However, BPF does not provide support for statistical packet sampling. It does not provide a pseudo-random number generator nor does it allow state to be kept between invocations of each filter program. All filtering decisions must be made based on the contents of the packet being inspected.

In their 2006 paper, Gonzales and Paxon [4] discuss additions to the BPF interpreter that provide access to pseudo-random number sources for use in packet sampling. While their approach is valid and useful it requires changes to the host system’s kernel which may be seen as too intrusive by many operators. This paper describes a method of performing pseudo-random packet sampling that does not require changes to the host system and so can be used in the general case. The advantage of Gonzales’ method is that it provides true pseudo-random sampling without any dependence on packet fields being present.

On the wired side there is much existing literature surrounding packet sampling techniques. One of the more interesting techniques is hash-based sampling [8], [5]. This method takes a hash of certain header fields which are invariant at each hop and uses this hash to make sampling decisions. This allows routers to capture a sub-set of the packets on a network and additionally provides the ability to perform trajectory sampling [3], that is, to capture the same frame at each hop through the network.

Hash-based sampling is designed for high-powered wired routers which have ample processing power. In order to passively monitor an IEEE 802.11 wireless network interface without adversely affecting the performance of the host, a technique is needed to allow for cheap, stateless statistical sampling of packets. It should be noted that this technique is specifically not intended to allow full packet capture on low-power devices. Instead it is intended to allow passive monitoring of a network by taking a reliable sample of the frames on the air.

Our hypothesis is that the IEEE 802.11 Frame Check Sequence (FCS) field can be treated as a pseudo-random variable suitable for use in packet sampling decisions. This paper shows that given a set of constraints, the FCS can indeed be used for packet sampling. In comparison to hash-based sampling, using the FCS has the advantage that it requires no extra computation of the hash - it is pre-computed by the wireless hardware. On the other hand, the FCS covers the entire frame, not just invariant fields. This means that it cannot be used for trajectory sampling.

First, a real-world passive wireless trace is captured and the distribution of received FCSs is analysed to determine whether the variable follows a uniform distribution. This ensures that each element from the distribution has the same probability of selection and allows us to easily identify any outliers.

Once it has been determined that the variable is indeed uniform, the random number generator must be checked for other properties. In particular it must be shown that the source does not generate cyclical patterns of numbers which could result in a variable with uniform distribution but which has a predictable bit stream. The National Institute of Standards and Technology (NIST) provides a statistical test suite for the evaluation of new random number generators, which is described by Soto [12]. The NIST Statistical Test Suite (STS) is used in this paper to evaluate the quality of the FCS as a random number generator.

Section II discusses the data set used to perform analysis of the characteristics of the random variable under investigation. Section III discusses the results of the statistical analysis performed on the data set. Section IV provides a sample BPF fragment for use in packet sampling and Section V concludes the paper, discussing advantages and disadvantages of using the FCS as a pseudo-random number.

II. DATA COLLECTION

Several passive wireless traces were captured over a period from 6–12 March 2008. Using a Soekris net5501 single-board computer with an Atheros 5004-based wireless miniPCI Network Interface Card set to “monitor” mode, some 40 million frames were captured complete with link-level IEEE 802.11 and Radiotap [10] monitoring headers. The captured frames include the IEEE 802.11 Frame Check Sequence as the last four bytes of the payload. The Radiotap header provides per-frame signal level, noise level, bit-rate and flags to indicate various properties of the frame. The MadWiFi driver [6] was set to capture all frames, including those that failed their Frame Check Sequence check in hardware.

The captures were taken at a busy site which houses end-points of several 802.11b/g links. A 10 dBi omnidirectional antenna was used to passively capture frames in the vicinity.

This provided the capture point with a wide variety of traffic to capture to allow for a representative sample of traffic to use during analysis. The capture point was set to capture 7.5 GB of full frames (due to storage limits on the capture point), which were then transferred to a central location. The following analysis is performed...
on three subsets of the data in the most complete trace. In total the trace captured 38,656,691 frames. The passed data set contains frames that passed their FCS (22,993,726 frames). Of the remaining frames (those that failed their FCS), 11,828,110 had no layer-2 data which is discussed in Section III-C and 3,356 frames were less than 4 bytes long. With these frames removed, the failed data set contains 3,831,449 frames and the all data set contains 26,825,225 frames.

The full trace used in this paper is available and is described more fully at [13]. It was anonymised using Crypto-PAN prefix-preserving IP anonymisation and the user payload was zeroed in each packet, with the last four bytes of each packet (the 802.11 Frame Check Sequence) left intact. The dataset is available to researchers by contacting the author.

### III. Statistical Analysis and Results

This section discusses the details of each of the statistical tests performed to validate the randomness of the FCS variable. First, the variable is tested for uniformity. Once this is established, more rigorous tests are performed on the FCS sample to determine whether the FCS can be treated as a suitable random number generator.

#### A. Tests for Uniformity

The Kolmogorov-Smirnov (KS) test [2] as implemented by the R statistical environment [9] was used to test the data for uniformity by performing a single-sample goodness-of-fit test. A goodness-of-fit test is a hypothesis test that either accepts or rejects the null hypothesis, \( H_0 \). In this instance, \( H_0 \) is that the measured distribution of the FCS variable follows the uniform probability distribution with parameters \( \min = 0, \max = 2^{32} - 1 \). A \( p \)-value less than 0.01 is considered to be strong evidence against the null hypothesis that all 32-bit FCS strings are equally probable and \( p \)-values greater than 0.1 are considered to give no evidence against this null hypothesis. Results of the KS tests can be seen in Figure 1.

The KS test results for the passed, failed and all data sets indicate that none can be considered uniform. A \( p \)-value of \( 2.2 \times 10^{-16} \) is as close to zero as we are likely to get and constitutes a strong rejection of \( H_0 \). Investigation of the frequency of unique FCSs showed that several FCSs had dis-proportionately high frequencies. These correspond to 802.11 “Control” frames. Control frames include link-layer acknowledgement and CTS/RTS frames. Control frames do not contain fields such as a sequence number which can change on a per packet basis. This means that for each sender/receiver pair, all of the Control frames have static content and hence a static FCS. These Control frames were removed from the traces by filtering all frames less than or equal to 20 bytes in length.

The KS-test results for the data sets with Control frames removed show that the FCSs from the passed data set can be considered uniform. However, the FCSs from the failed data set, and therefore the all data set, cannot. The non-uniformity of the failed data set will be discussed later.

The results from the KS-tests indicate that in order to use the FCS as a pseudo-random number, Control frames must be excluded. Practically this means that packet sampling based on the FCS will fail to sample link-layer acknowledgement frames and CTS/RTS frame exchanges. This may or may not be acceptable depending on the application. However, much information can still be gleaned from passive link-level capture even with Control frames removed, so the analysis of the data set with Control frames removed will continue.

We have shown that after removing Control frames from the data set, the FCS passes the KS test for coming from a uniform distribution. However, this test examines only one direction of departure from a uniform distribution. In particular it is possible that FCS pattern probabilities may be affected by a knowledge of earlier FCS values. To guard against this and other forms of departure from randomness further tests for randomness must be performed.

#### B. Tests for Randomness

The following tests are based on the NIST Statistical Test Suite (STS) for testing random number generators (RNGs). These tests treat the data as a number of samples from a proposed RNG and perform several statistical tests to determine whether the samples came from a “good” RNG. By performing each test a number of times and observing the distribution of resulting \( p \)-values a determination as to whether the RNG is “good” can be made.

The NIST STS tests were run over the three data sets, all, passed and failed. The FCS of each packet was extracted into a binary file which was used as the input to the test suite in order to simulate the output of a random number generator. These data sets were filtered to not contain packets shorter than 21 bytes in order to remove Control frames as discussed in the previous sections.
Each test in the test suite was run with a number of 1,000,000 bit samples from each data set. The all data set contained enough data for 500 samples. As the passed and failed data sets are subsets of the all data set, they contained enough data for 400 and 100 samples respectively. Each test that was run produced a p-value for each sample. In order for a sample to pass the test, the p-value must be $\geq \alpha$ which we take as 0.05 in the following. For each test, some proportion of samples will pass and some will fail. In order to determine whether a data set is random the proportion of samples that pass should lie within an acceptance interval. If the proportion of samples fall outside this acceptance interval then there is evidence that the sample came from a poor RNG.

The proportion of samples that passed each test are plotted in Figure 2 for each data set. The range of acceptable proportions is determined using the acceptance interval defined by:

$$1 - \alpha \pm 3\sqrt{\frac{\alpha(1 - \alpha)}{m}}$$

Figure 2 shows that for the passed data set, almost all of the tests had a proportion of passed samples within the specified acceptance interval (at $\alpha = 0.05$). This strongly indicates that the FCSs of the packets that passed their FCS can be treated as coming from a “good” RNG. When observing the proportions of samples that passed from the failed data set, most of the proportions failed to fall within the acceptance interval. In fact, most of the tests showed that only a very small proportion of samples passed, strongly indicating non-randomness.

The next section will go into more detail regarding the non-randomness of the failed data set. Finally, the two data sets were combined into the all data set, which shows a much higher spread of results. The results from the all data set can be correlated to the proportion of frames in the capture that failed their FCS. For example, in an environment where very few FCS errors are seen, the FCS would be close to random. However, as the packet error rate increases, the randomness of the FCS will decrease.

This section has presented results from running the
NIST Statistical Test Suite over the collected data. We have found that when the data set is restricted to frames that pass their FCS, the FCSs can be treated as coming from a “good” random number generator and hence can be used for random packet sampling. Frames that fail their FCS show a distinct non-randomness which the next section will investigate.

C. Analysis of the failed data set

Section III-B showed that the failed data set contained frames whose FCS could not be assumed to be random. This section explores the data in this data set and attempts to explain why this data may not be random.

The failed data set contained 15,662,965 frames in total. 95% of the failed frames were from the 1, 2, and 11 Mbit rates. Interestingly, the passed data set is spread fairly evenly across the 1, 2, 6, 24, 48 and 54 Mbit rates which indicates either that the lower rates are more susceptible to FCS failures (due to the longer time spent on the channel) or that in times of error, higher rates are less likely to be decoded by the receiver and so are less likely to appear in the trace at all. Alternatively, it may indicate that in times of high bit-error rate, the rate selection algorithm is preferring low bit rates. Of the failed frames, approximately 70% did not contain any layer 2 payload. This may indicate that the PLCP was decoded correctly but the receiver failed to decode the payload of the packet. These frames were discarded due to the rule that frames shorter than 21 bytes must be ignored.

Of the remaining failed frames, there were 3,370,114 unique FCSs with 0.7% of FCSs making up nearly 4% of the received frames. By far the most common FCS was 0xFFFFFFFF (4,448 occurrences) followed by 0x00000000 (979 occurrences). Almost all of the highly occurring FCSs are from frames that have been truncated, hence the final four bytes of the frame are not the FCS, but a selection of 8 contiguous bytes from the payload. It appears that the primary cause of the non-randomness of the failed data set is truncation.

We hypothesise that in traces that do not display such a high level of truncation but instead show random bit-errors, a packet sampling technique based on FCS may still be valid. However, developing an FCS-based packet sampling technique in the presence of frame truncation would be impossible. Unfortunately there is no simple way to detect if an IEEE 802.11 frame has been truncated due to the lack of a length field in the MAC header. Further research is necessary to determine if an FCS-based packet sampling technique could be used to sample frames that fail their FCS but do not suffer from truncation.

Fig. 3. A BPF fragment to filter packets based on the FCS

IV. BPF Fragment

This section presents a sample BPF fragment that can be used to perform stateless random packet sampling for wireless networks. The program in Figure 4 filters packets shorter than 21 bytes and then compares the last four bytes of the packet (which it assumes is the FCS) with a constant $FCS_T = n \times 0.01 \times UINT_{32}_{MAX}$ where $n$ is the percentage of packets to be sampled. It assumes a link type of IEEE802_11 however the program can be easily altered to take into account extra meta-data headers such as Radiotap by altering the initial offset. The constant $MINLENGTH$ is set to 21 and ensures that any frame less than this length fails the filter. The ld instruction automatically fails the filter if it attempts to access past the end of the frame. This achieves the goal of filtering Control frames as well as frames that have no layer 2 content in a single BPF instruction.

A. Validation of the BPF Fragment

This section presents a short empirical validation of the effectiveness of the BPF fragment presented in section IV. The passed data set with Control frames removed was used as a source trace over which two sampling methods were applied. Simple random sampling using the standard C library’s rand(3) function is compared to the FCS sampling method presented in this paper.

The modified passed data set contained 14,562,399 frames. Both sampling methods were set to sample 10% of frames. The random sampling method sampled 1,455,398 frames and the FCS sampling method sampled 1,456,604. We then compared the packet size distributions of the two sampled traces using the Kolmogorov-Smirnov test. The packet size distribution of each of the sampled traces were compared to the packet size distribution of the source trace. Both sampling methods

```
ld [MINLENGTH]
ld #len
sub #4
tax
ldw [X]
jgt FCS_T, L1, L2
L1: ret #0
L2: ret #TRUE
```
achieved p-values greater than 0.9, a clear indication that the null-hypothesis - that the two samples were drawn from the same distribution - holds.

V. CONCLUSIONS

Current techniques for measuring wireless networks are either inaccurate or too resource intensive. They simply do not scale to allow for monitoring at each router on the network in a way that is cheap or reliable. A technique for cheap, accurate and stateless statistical packet sampling has been presented as a method for enabling distributed link-layer measurement and monitoring of wide-area wireless networks.

We have presented a study of the randomness of IEEE 802.11 Frame Check Sequence field over a large real-world trace and concluded that the FCS can be treated as random when Control frames are removed from the sample set. Additionally it was found that the non-randomness of the failed data set was caused by frame truncation which is an artifact of the environment in which the data set was captured. This indicates that there is further research to be done into determining when the FCS can be used as a random number when truncation is not present.

Finally, a BPF fragment to implement FCS-based stateless packet sampling was presented and a short validation of its usefulness was carried out.

Given the level of frame truncation present in the failed data set, it is hypothesised that it is the truncation causing the FCSs to appear non-random. More work is needed to study the effect of different type of FCS failure (such as bit-errors) to see if the FCS can be treated as random in this case.

In conclusion, given the above conditions, the IEEE 802.11 Frame Check Sequence field appears to be a valid source of random numbers for use in packet sampling in wireless networks. This allows low-end devices to perform accurate stateless packet sampling cheaply in kernel-space via the Berkeley Packet Filter system. This technique can now be used to enable distributed link-layer measurement of wide-area wireless networks.

REFERENCES


