Impact of Scrambling on Barcode Entropy

Marie Vans, Steven Simske, Margaret Sturgill, Jason Aronoff; Hewlett-Packard Labs; Fort Collins, CO, USA

Abstract

Security barcodes and other "actionable" barcodes have become commonplace as a consequence of the recent ubiquity of mobile phones equipped with high-quality cameras. In this paper, we provide methods for quantifying the entropy of the embedded barcode data, assuming methods other than the standards-specified error correcting code (ECC) approaches can be adopted. Entropy, which reduces the likelihood of a fraudulent agent being able to "guess" correct barcodes, is measured directly using a variety of novel algorithms and applied to large sets of barcodes. Our data, however, show that removing ECC provides the additional advantage of increasing entropy. Thus, all other settings (data payload size, printing technology, substrate used, etc.) being equal, eliminating the ECC will increase the security of the information content.

Introduction

Barcodes are not just for ringing up sales anymore. Many organizations (ScanBuy, Microsoft, etc.), standards bodies (OMA, GS1 Mobile Commerce, etc.) and consortiums have specified the data models for barcodes to be captured at point of sale, as a means to connect to a website, or for consumer capture of salient content about products and/or their surroundings [1].

Increasingly, the older 1D barcode standards (e.g. Code 39 and UPC) traditionally used at point of sale have been replaced and/or augmented with 2D or 3D barcodes. These high-density barcodes can be used for additional data carrying (e.g. mass serialization) or referencing (e.g. URL pointing) aims. Barcodes are typically encoded with error-correcting code (ECC), which is added to make them more robust to certain types of distortion and damage. However, the nature of the ECC added is derived from assumptions—such as Shannon entropy—more relevant to 1D barcodes or general information theory. As such, the use of ECC itself can be questioned—which opens the door to using barcodes as information carriers outside of the current barcode standards.

Our experience has shown that the most critical barcode distortions to address are (a) effect of the print-scan (PS) cycle, or "copying" cycle; (b) localized damage such as water damage and/or puncturing; and (c) blurring [4]. We have shown in previous work that it is advantageous to either (a) increase the size of the barcode modules themselves or (b) duplicate the barcode data itself within the barcode, in place of ECC [5]. The latter typically results in a barcode "unreadable" to the current standard for the symbology, and allows custom interpretation of familiar barcode representations.

In this paper, we describe an attempt to highlight the differential effects of scrambling methods on entropy by applying encryption methods to randomly generated strings, both with and without ECC. Increased entropy, which reduces the likelihood of a fraudulent agent being able to "guess" correct barcodes, is measured directly using a variety of novel algorithms and applied to large sets of barcodes.

We discuss the implications of these findings on overall security printing and forensic printing ecosystems. Since large organizations are responsible for much of the fraud—counterfeiting, factory overruns, diversion, smuggling, rebate fraud, etc.—that currently exceeds 8% of world trade [6], an effective security ecosystem is designed to decrease the initial time to discovery and enable efficient and accurate assessment of the size of the counterfeiters involved. The barcode scrambling approaches outlined herein are an important part of that ecosystem of combined security printing, investigation/evidence gathering and prosecution.

Entropy Measures

We used three entropy measures on a dataset of 672,000 2-dimensional barcodes, half incorporating Reed-Solomon ECC [2], with the other half having no ECC. We used entropy as a measure for the effect of ECC and scrambling on the 2D barcodes. Here, entropy represents signal randomness, i.e. how the bits are distributed in a signal. For example, Equation 1 presents what we name "Normalized Entropy", where N is the maximum number of run lengths (of 0's or 1's) and E(X) is $M * (½)^i$, which is the expected number of run lengths of each length i. x is the actual number found in each bin i and M is the total number of all run lengths.

$$e_1 = \sum_{i=1}^{N} \log\left(\frac{E(X) + |E(X) - x|}{E(X)}\right) * E(X)$$

Equation 1 – Normalized Entropy

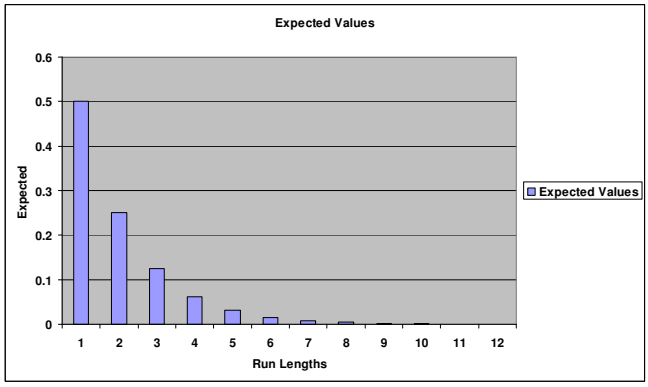

Figure 1: Expected bin % for Maximum Entropy

Figure 1 shows a graph of the expected bin percentages for maximum entropy of a signal. For example, if we have a string with 32 different runs of 1's and 0's, there should be 16 run

lengths of 1, 8 run lengths of size 2, 4 run lengths of size 3, and so on.

Figure 2 shows three characterizing e_1 values using the Normalized Entropy algorithm: the maximum entropy, low entropy wherein all the run lengths are approximately equal, and a minimum or no entropy when the string is essentially all 1's or 0's. As can be seen, higher entropy results in a lower value for e_1.

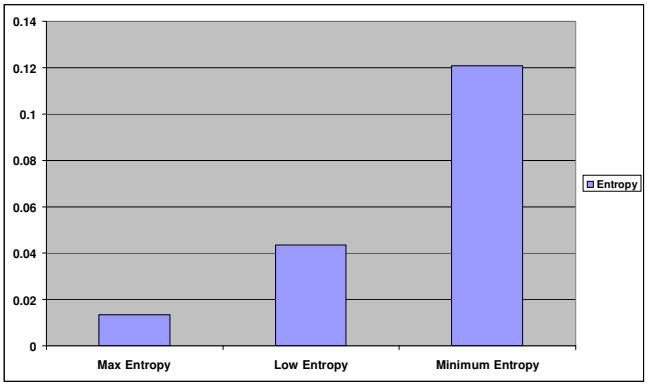

Figure 2: Sample e_1 Values for Max, Low, and Minimum Entropy

We used another entropy measure, based on Hamming Distance, shown in Equation 2 (e_2). In this case, N refers to the maximum Hamming Distance (HD) between two bytes and x refers to the normalized i HD of the actual strings. This HD is calculated on a moving window along a string in a forward direction. Figure 3 shows the general trend for this measure. For example, if the string contains a pattern of 1's and 0's such that the hamming distance is always the same (1100110011001100), entropy would be low. An additional HD measure (e_3) was also used, which is similar to Equation 2 except that the HD is calculated by moving the window in any direction.

$$e_2 = \frac{\sum_{i=1}^{N} \log_2 \left(\frac{1 + |x - 1.0|}{1.0} \right) * 1.0}{N - 1}$$

Equation 2 - Hamming Distance Entropy

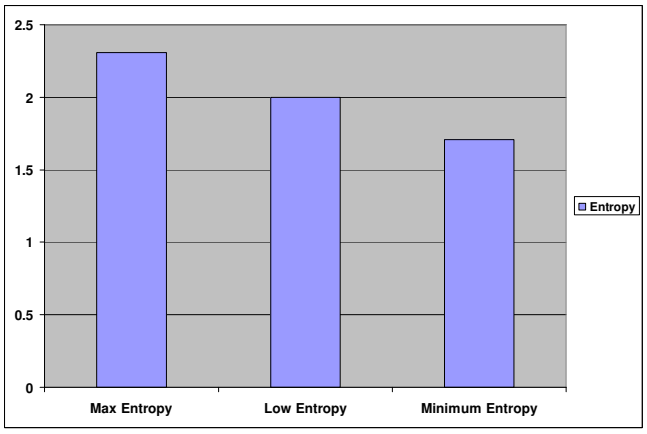

Figure 3: HD Entropy - Sample e2 Values for Max, Low, & Minimum Entropy

We used these measures because they are insensitive to string length, the expected values are easy to compute, and they converge quickly.

Scrambling Techniques

In order to test encryption methods, we ran several experiments using the three entropy measures on four scrambling algorithms. A test run consisted of 500 randomly generated strings, with an average length of 310 bits for each of the typical single block barcode symbol sizes of 12x12 up to 26x26 each with module sizes from 12 to 18 pixels. This generates 28,000 individual barcodes. Each test also has an associated scrambling algorithm and entropy measure. Each test was run twice; once using the maximum number of ECC bits allowable for the size and once using additional randomly generated data where the ECC bits would normally be inserted. A total of 672,000 barcodes were tested with half containing ECC bits and half without.

The four scrambling algorithms consisted of the following:

1. XOR: A randomly generated string of the same size as the entire string (message + ECC bits) and XOR'd with the input string.
2. Structural scramble: Divide the string matrix into equal sized structures (squares, rectangles, etc.). Swap bits within each structure so that the new structure is a mirror image of the original.
3. Even Check Bits: Add a check bit at the end of each row and column so that the total number of black modules is even.
4. Odd Check Bits: Add a check bit at the end of each row and column so that the total number of black modules is odd.

Tests

"Challenging" the entropy of the string set with another random string should result in different responses if the string is not as entropic as the challenge string. For example, Figure 4 demonstrates that when the completely random number is challenged there should be no difference in the entropy between the two randomly generated strings. However, when the string contains ECC, there may well be a detectable difference in the entropy between the string with ECC and the randomly generated challenge string. This is indicated by the oval surrounding the place within the string that contains ECC. One of the main objectives of these experiments is to determine whether that difference is detectable. If so, finding the best scrambling algorithm along with the most sensitive entropy measure to highlight differential effects leads us to a recommendation for adding security mechanisms to 2-dimensional barcodes.

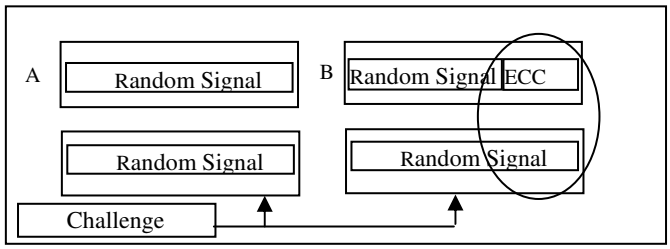

Figure 4: Effect of ECC on Challenge Result

Results and Discussion

Figure 5 shows the results for Normalized Entropy using all the scrambling techniques for ECC and NonECC strings. For each symbol size, the result is the percent change of entropy between the input and output strings. For example, results for the 12x12 symbol shows that the change in mean entropy for the string containing no ECC was very small. This makes sense because scrambling a fully random string should result in another random string. The entropy of 12x12 symbols with ECC, however, increased (by more than 5%) after scrambling. This is also logical as scrambling a string containing non-random bits should result in a more random string. These results look promising for developing a method for detecting attacks using the change in entropy after scrambling.

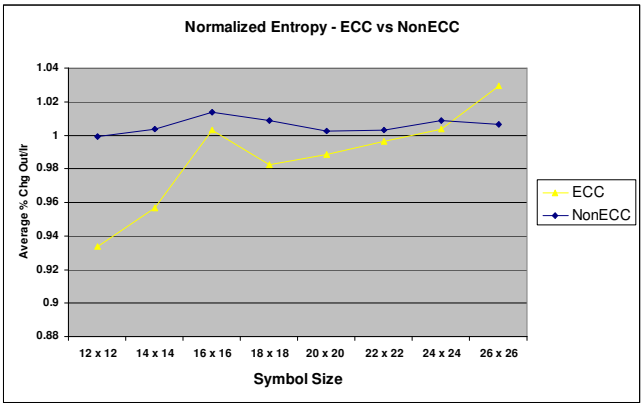

Figure 5: e_1 Normalized Entropy Results– ECC vs NonEcc

The next thing to look at is the population statistics. Figures 6 and 7 are the input and output mean e_1 values for ECC and non-ECC along with the standard error for the XOR scrambling algorithm. Unfortunately, these results are representative of the results for all the scrambling algorithms. The main conclusion to be drawn from these figures is that there is no way to distinguish between ECC and non-ECC strings by looking at difference in input or output means only.

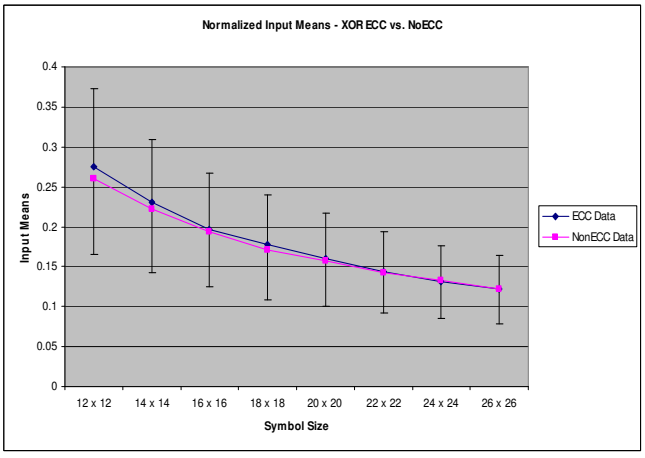

Figure 6:e_1 Input Means Compare -- ECC vs. NonECC

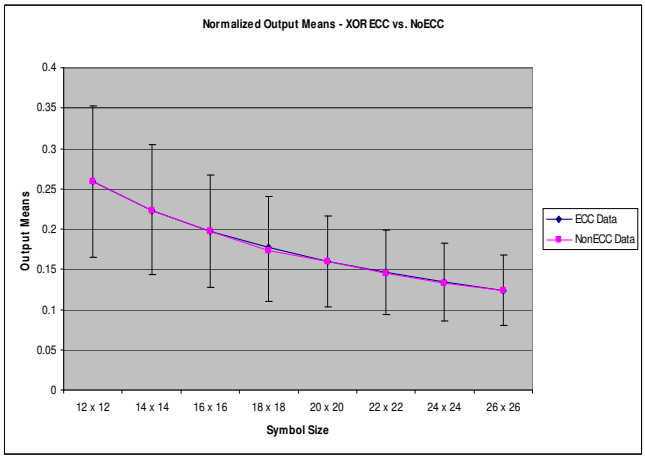

Figure 7: e_1 Output Means Compare - ECC vs. NonECC

In Figure 8, the data for Figures 6 and 7 are combined to show the input and output Normalized Entropy (e_1) means for ECC and non-ECC using the XOR scrambling algorithm. This figure highlights the difficulty in finding differences using population statistics.

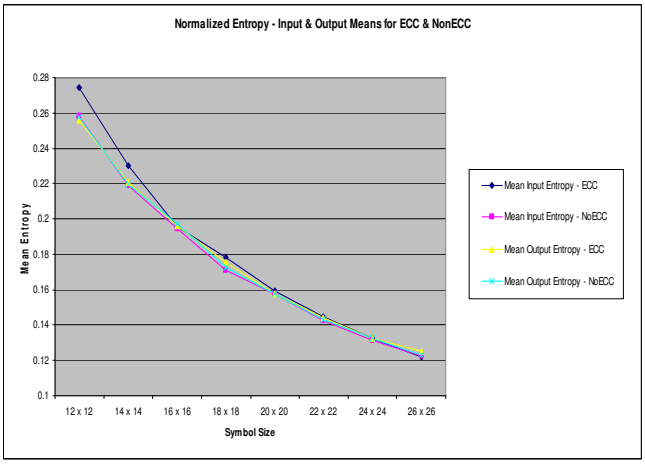

Figure 8:e_1 Mean Entropy Comparison -- ECC vs NonECC

Results are more pronounced between ECC and non-ECC using the Hamming Distance measures (e_2 and e_3) than for the Normalized Entropy measure. Figure 9 shows the first HD measure for ECC and non-ECC signals. Here, there is a detectable difference between the signals containing ECC and those that do not. Recall from Figure 3 that as a signal becomes less random, this entropy measure decreases. Figure 9 demonstrates that the change in entropy after scrambling results in higher entropy (less randomness) for both the ECC and the non-ECC strings. For the majority of the symbol sizes, e_2 output values are lower than input values. Figure 10 contains the results for the 2nd HD entropy measure (e_3). These results are very close to those of the 1st HD measure. Again, the answers seem logical. The ECC strings start out with more structure than the non-ECC string and become more random after scrambling.

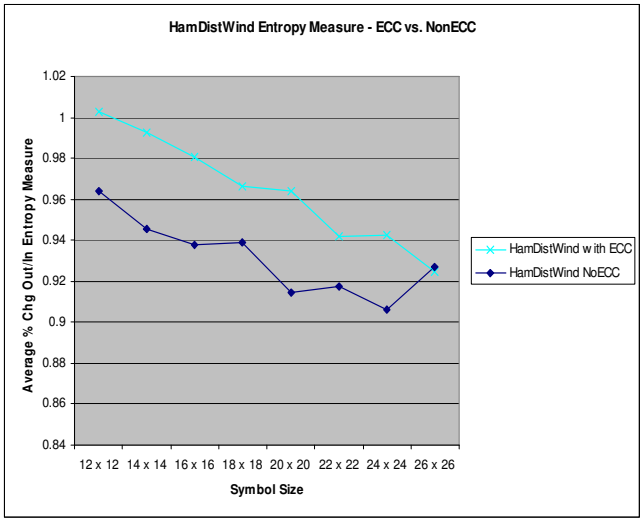

Figure 9: e_2 HD Entropy Measures ECC vs. NonECC

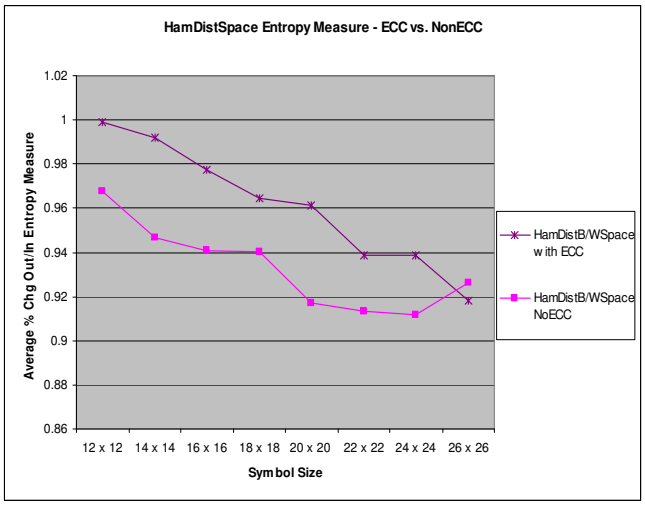

Figure 10: 2nd HD Entropy Measure (e_3) ECC vs. Non-ECC

If we look at individual results, any of the scrambling algorithms along with the Hamming Distance measures give us typical results in terms of separation between ECC and non-ECC strings. As an example, Figures 11-15 show the population statistics for the input and output means when using the XOR scrambling algorithm. The other scrambling algorithms show similar results. The data points contain only half the error bar in order to show the magnitude of the standard error. Obviously these two populations overlap and cannot be distinguished with any reasonable level of statistical confidence. In general, while the change in entropy after scrambling of the non-ECC strings is detectable, the population statistics (Figures 12-15) show that detecting the difference between ECC and non-ECC signals using population means is not easy, and is possibly impossible (and certainly impractical) using these methods.

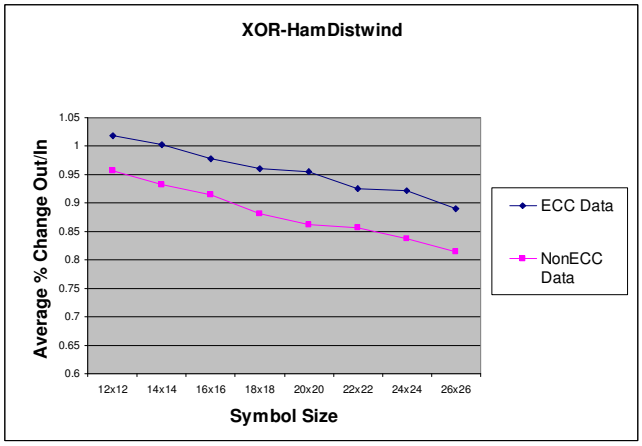

Figure 11: XOR Scrambling with HD % Change

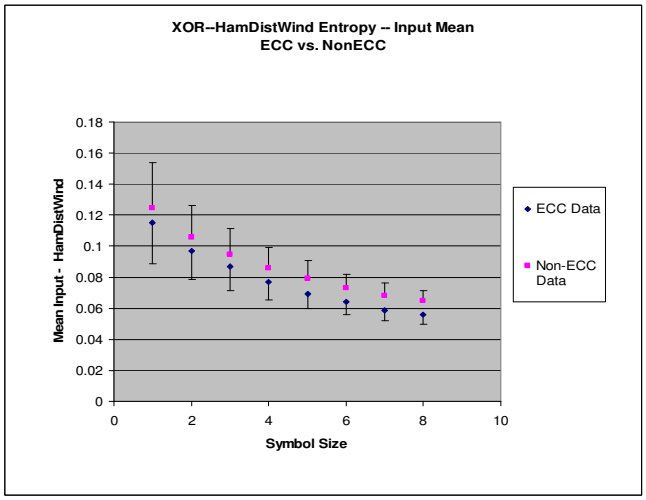

Figure 12: XOR Scrambling with HD - Input Mean

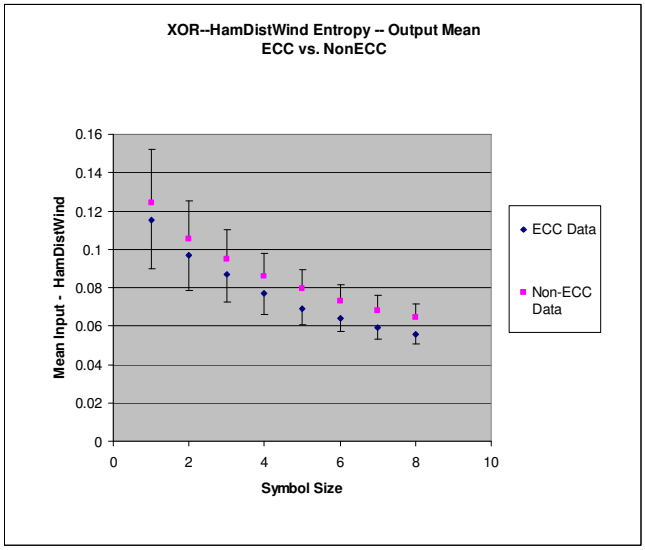

Figure 13: XOR Scrambling - Output Mean

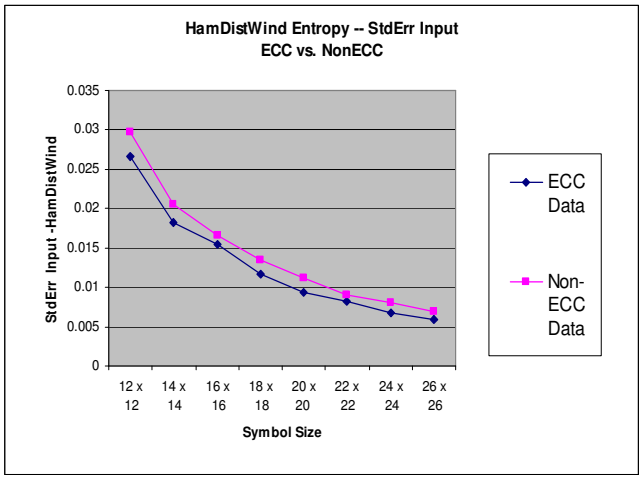

Figure 14: XOR Scrambling - StdErr Input

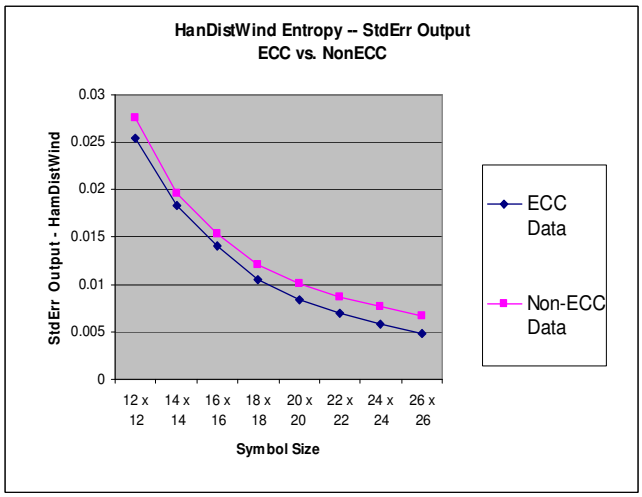

Figure 15: XOR Scrambling - StdErr Output

Conclusions

We have presented three entropy-based methods for determining the degree of randomness in a signal and the affect of scrambling on the outcome of these methods. In our case, we used two-dimensional barcodes because of the ubiquity of these symbols in supply chains of virtually every manufacturing sector. As the incidents of counterfeiting continue to raise, security at each node within the supply chain becomes more critical. The Data Matrix standard [2] does not take this type of security into consideration, as the ECC within the signal has structure and is therefore vulnerable to attacks. We have shown that our entropy measures and the appropriate "attack" can detect the difference between a truly random signal and a signal that contains structure. This can be used to discover whether ECC has been used on a set of materials, and if so potential vulnerabilities of the security data. The methods described here can also be implemented to determine whether data is encrypted, since proper encryption should also work to maximize entropy. As shown, it is possible to interrogate the entropy of the comprised signal and compare it to the original entropy values.

Acknowledgements

The authors gratefully acknowledge our collaborators, including Guy Adams and Pipo Caban.

References

[1] http://www.gs1.org/productssolutions/mobile/, http://www.epcglobalinc.org/home, http://www.microsoft.com/tag/content/overview/, and http://www.openmobilealliance.org/, last accessed on 25 June 2010.

[2] International Standard ISO/IEC 16022, "Information Technology—Automatic Identification and Data Capture Techniques—Data Matrix Bar Code Symbology Specification," 2nd ed., 15 Sept. (2006).

[3] Shannon, Claude E.: Prediction and entropy of printed English, *The Bell System Technical Journal*, 30:50-64, January 1951.

[4] S.J. Simske, M. Sturgill, and J.S. Aronoff, "Effect of Copying and Restoration on Color Barcode Payload Density", DocEng '09: Proc. 9th ACM symposium on Document engineering, pp. 127-130, 2009.

[5] M.Vans, S. J. Simske, and J. S. Aronoff, "Barcode Structural Pre-Compensation Optimization", NIP25, p. 167-169, September 20, 2009.

[6] World Economic Forum, Jan. 2009.

Author Biography

Marie Vans currently works at HP Labs in Fort Collins, Colorado in the HP Labs Security Printing and Imaging Group. Her main interests include Document Workflows, Statistical Language Processing, and Library Information Science. She holds a Ph.D. in computer science from Colorado State University. She previously worked at HP Labs in Haifa, Israel on image processing and analysis.