Discovering an RC4 Anomaly through Visualization

Dino Schweitzer
Department of Computer Science
United States Air Force Academy CO 80840
dino.schweitzer@usafa.af.mil

Leemon Baird
Department of Computer Science
United States Air Force Academy CO 80840
leemon.baird@usafa.af.mil

ABSTRACT
Visualization can be an effective means for analyzing security data and teaching students different concepts about various security algorithms. At the Air Force Academy, interactive visualizations are used to teach ciphers to students in a cryptography course. In the course of preparing student visualizations about the RC4 cipher characteristics, an anomaly was discovered in the basic encryption algorithm. This paper describes the anomaly and the process of how it was discovered through visualization.

Categories and Subject Descriptors
E.3 [Data]: Data Encryption.

General Terms: Experimentation, Security.

Keywords
Visualization, Cryptography, Algorithm Analysis.

1. INTRODUCTION
Visualization is a powerful tool in many applications such as scientific study, simulations, data mining and analysis, business, and education. It is not a new concept, but a field with a rich history in the use of graphs and visual displays for better understandings of data [7]. The advent of increased computing power and advances in computer graphic algorithms led to the use of visualization for scientific computation [2]. Visualization in education has a similar rich background since the 1980's with applications such as algorithm animations, data structures and abstract concept representation, and virtual environments [8].

Visualization in information security has grown with the field and is a natural outgrowth of a data-rich envioronment. For example, the massive amounts of data associated with network traffic lends itself to a visual approach for data reduction and analysis for applications such as intrusion detection and routing investigations.

At the United States Air Force Academy, we are investigating the use of interactive visualizations in the education of information security. An interactive graphical tool for teaching security protocols have successfully been used to demonstrate concepts such as public key exchange [5].

VizSEC'06, November 3, 2006, Alexandria, Virginia, USA.

Similarly, in the Cryptography class we have developed and used a set of interactive visualization applets to demonstrate various historical and current cipher algorithms [6]. These visualizations demonstrate various attacks and weaknesses of different ciphers such as frequency analysis.

In addition to the interactive visualizations, we also create static visualizations for the classroom to illustrate different security concepts. For example, when demonstrating the concept of S-boxes in the DES encryption scheme, we created a visual image showing the mapping of all inputs to outputs as shown in Figure 1. The relative randomness of the 2-D mapping along with the frequencies by quadrant demonstrate the random characteristics of the S-box mappings.

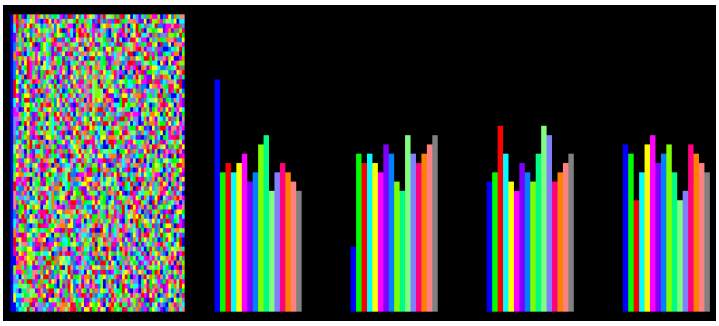

Figure 1. DES S-Box mappings for all inputs and by quadrant.

It was in the course of preparing these types of visualizations for explaining the RC4 encryption algorithm that led to the discovery of an interesting anomaly in the algorithm that was previously unknown to the security faculty.

2. RC4 ENCRYPTION ALGORITHM
RC4 is a stream cipher originally developed in 1987 by Ron Rivest of RSA Security [3]. Because of its simplicity and speed, it was used extensively for several applications and became the basis for the WEP standard used for wireless encryption. The basic concept is that a pseudo-random string of bytes is generated which is XOR'd with the plaintext to generate the ciphertext.

To generate the random bytes, a permutation of all possible combinations of bytes is stored in a 256 element array known as the S array. Initialization of the S array uses a key of arbitrary length to "shuffle" the locations using the simple algorithm shown:

```
for i from 0 to 255
        S[i] := i
j := 0
for i from 0 to 255
j := (j + S[i] +
  key[i mod keylength]) mod 256
swap(S[i],S[j])
```

The result of this initialization is an array with values 0-255 in an arbitrary order based on the key. Once the S array has been initialized, an equally simple algorithm is used to generate as many pseudo-random bytes as necessary to encode the message.

```
i := 0
j := 0
while GeneratingOutput:
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    swap(S[i],S[j])
    output S[(S[i] + S[j]) mod 256]
```

3. VISUALIZING RC4

As a demonstration of these algorithms and their operation, an interactive visualization applet was developed as shown in Figure 2 (a description of the tool can be found in [6]). The user can step through the algorithm and watch S-array locations get swapped according to the key. For demonstration purposes, the 256 element S-array was replaced with a 26 element array.

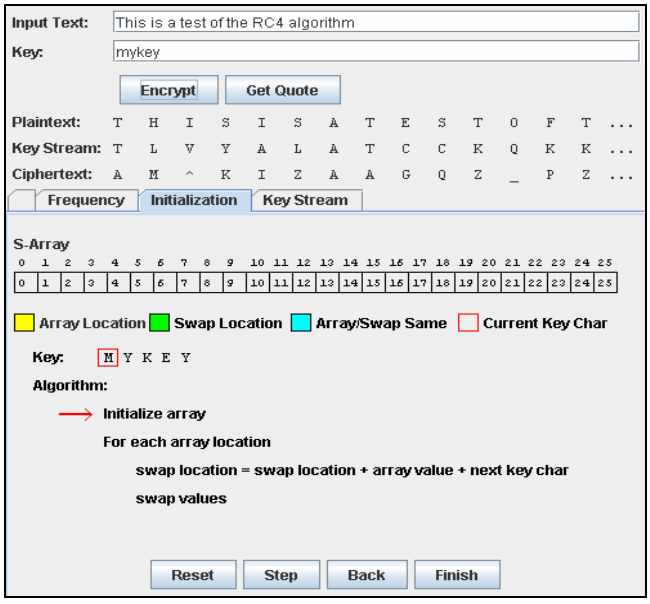

Figure 2. RC4 visualization applet.

In addition to the interactive visualization, we wanted to demonstrate a known weakness of the RC4 algorithm when similar keys are used in the generation of the S array. Two keys with the first *n*-bytes the same will have related outputs for the first few bytes, because there will be correlations between the first *n* positions of the S array. To illustrate this, we generated S-arrays using random keys with the first 4 or 8 bytes the same. We calculated the average difference of the resulting S-array locations and color coded them as shown in Figure 3 (the 256 elements are shown in a 16 by 16 grid). As is visually obvious from the image, the first *n* bytes varied by a much different amount than the remaining bytes of the array (first 4 were the same on the left, first 8 on the right).

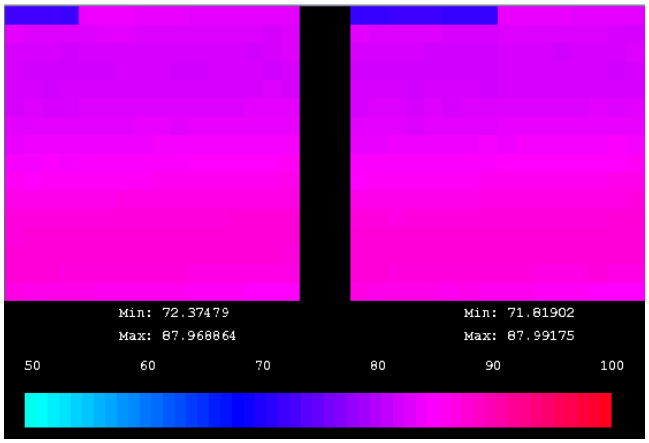

Figure 3. Difference in S-arrays with the first bytes of the key the same.

In addition to the anticipated visual result, an unexpected consequence of this visualization was the pattern appearing in the remaining bytes of the S-array. While a difference was expected in the first *n* bytes, it was expected that the remaining bytes would follow a fairly simple monotonic function. Instead, there was a large amount of structure visible. This visual anomaly led to deeper analysis of the algorithm.

To further investigate the randomness of the initialization shuffle, a second visualization was created counting the number of times each value appeared in each location of the S-array as shown in Figure 4. It should be noted that this analysis is for the general application of the algorithm, and not the specific weak case illustrated above. The X axis represents all possible values (0-255) and the Y axis is all array locations. Each point in the graph is the average number of times each value occurs in each location for 400,000 generations with random keys. The line graph on the right is a cross section clearly showing the discontinuity that occurs at X=Y. Thus, the "random" shuffle algorithm used in the RC4 S-array initialization was far from random! Initial search of the cryptography literature did not reveal this anomaly as common knowledge.

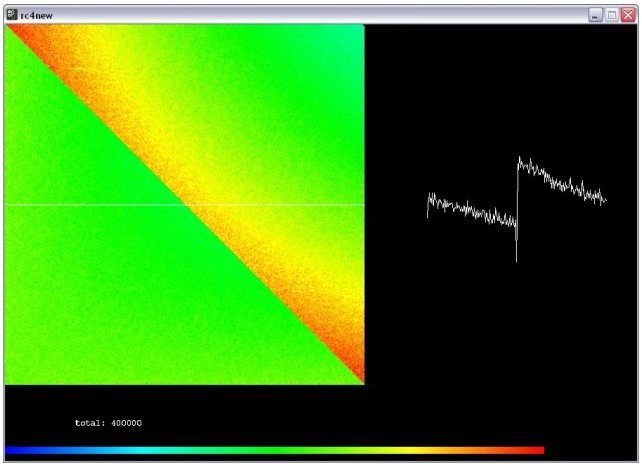

Figure 4. Probability graph of given value at given location in S-array.

4. MATHEMATICAL ANALYSIS

The next step was to find an explicit formula for the function plotted in Figure 4. This function is P(*x*,*y*), the probability that the number starting in position *x* will end up in position *y* at the end of the initialization. Let *n* be the number of positions in the array (*n*=256 for RC4), and number the positions from 0 to *n*-1. Look at the number that starts in position *x*. Since the counter *i* always increases, there are only three paths that this number can take to eventually move it into position *y* at the end. Let *m* be the highest position index that the number ever reaches in its travels. Let *k* be how many times the number moves (including swapping with itself). The three paths the number can follow are:

Case 1: $m = y \geq x$	The number jumps around zero or more times until $i=y$, at which point it moves to *y* (or stays there if it was already there) and never moves again.
Case 2: $m > y \geq x$	The number moves to increasingly higher positions one or more times (each time *i* catches up to it), then the last time *i* reaches it, it moves down to *y*.
Case 3: $m \geq x > y$	The same as Case 2, except it might never move higher.

Because these cover all possible cases and are mutually exclusive, P(*x*,*y*) is simply the sum of the three probabilities of each of these occurring. The probabilities for each case are easily derived by considering all possible values for *m* and *k* and summing the probabilities over all paths. The probabilities for each of the three cases correspond to the three terms on the first three lines of the following (where $a=1/n$ and $b=1-a$). The final expression is Mathematica's simplification of the summations.

$$\mathrm{P}(x,y) = ab^{n-1-y} + \begin{cases} \sum_{m=y+1}^{n-1} \sum_{k=2}^{m-x+1} \binom{m-x-1}{k-2} a^k b^{n-k} & \text{if } y \geq x \\ \sum_{k=1}^{n-x} \binom{n-1-x}{k-1} a^k b^{n-k} & \text{if } y < x \end{cases}$$

$$= ab^{n-1-y} + ab^x - \begin{cases} ab^{n-1-y+x} & \text{if } y \geq x \\ 0 & \text{if } y < x \end{cases}$$

This matches the empirical results when the key is 256 random bytes. This is a significant result that apparently has never been published in a paper, though it appeared in a master's thesis [1]. The thesis gave a far more complex, indirect derivation (almost 3 pages of dense equations), so the simpler, direct derivation here is a useful contribution. This would never have happened without the visualization raising the issue. This shows the power of visualization in cryptography and mathematics.

5. CONCLUSION

RC4 is no longer considered by cryptographers a strong encryption algorithm and WEP is not used by people with serious concerns about their wireless security. The particular weakness in the initialization algorithm demonstrated is a significant result, yet apparently has not been published outside a thesis. It even has applications beyond cryptography, because it will arise any time someone tries to shuffle an array using a naive algorithm (swapping with a random position) rather than the standard algorithm (swapping with a random position that is not lower).

The key finding here is not the anomaly itself, or the mathematics behind it, but rather the use of visualization in accidental discovery. The presence of visual patterns in the image was obvious to the eye while totally hidden when looking at the data. Further visualization was used to investigate and understand the anomaly. The fact that it was discovered while trying to illustrate a separate concept gives credence to the idea of visual exploration of data. It is this ability to take advantage of the strong pattern recognition of the human visual system that gives visualization its power. This power will be a huge benefit to the study of information security as new techniques are developed and experimented with.

6. REFERENCES

[1] Mantin, Itsak, Analysis of the Stream Cipher RC4, Master's Thesis, Weizmann Institute of Science, Rehovot, Israel, November 2001.

[2] McCormick, B. H. Visualization in scientific computing. *SIGBIO Newsl.* 10, 1 (Mar. 1988), 15-21.

[3] R. L. Rivest. The RC4 encryption algorithm. RSA Data Security, Inc., Mar. 12, 1992.

[4] Schneier B., "Section 17.1 RC4,"*Applied Cryptography, Second Edition*, John Wiley & Sons, 1996.

[5] Schweitzer D., Baird L., Collins M., Brown W., Sherman M., GRASP: A visualization tool for teaching security protocols, Proceedings of the 10th Colloquium for Information Systems Security Education, June 2006.

[6] Schweitzer D., Baird L., The design and use of interactive visualization applets for teaching ciphers, Proceedings of the 7th IEEE Workshop on Information Assurance, June 2006

[7] Tufte E., The visual display of quantitative information, Graphics Press, Cheshire, CT, 1986.

[8] Wilson J., Aiken, R., and Katz, I. 1996. Review of animation systems for algorithm understanding. In *Proceedings of the 1st Conference on integrating Technology into Computer Science Education* (Barcelona, Spain, June 02 - 06, 1996). ITiCSE '96, pp. 75-77.