Fortune Cookies as a Source of Entropy - A Qualitative Analysis

Abstract

Entropy, as used in computing, is a source of random information collected from various sources. This randomness is subject to various measurements of quality and is a measurable resource that can be depleted by using it up. Coming up with sources of high quality entropy under non-standard circumstances is a research question of interest within the security community. This paper explores an unconventional source of entropy; the usage of so-called "lucky numbers" on fortune cookies as a source of entropy. Studies are conducted on both the cost of this source compared to more conventional sources as well as the quality of random numbers generated.

1. Introduction

Entropy, as described in information theory, is the expected value of the information contained in a "message". A "message" is defined in this case to be any flow of information [**?**]. When applied specifically to computing, entropy is the random information gathered by the operating system and stored for later usage. The stored data is referred to as the "entropy pool". This random information is then taken from the pool for applications which require unpredictable streams of data. Common applications include cryptographic algorithms, video games, and animated screensavers.

Since entropy is required to be unpredictable, it cannot be re-used in applications. Such re-use would result in a cycle in which the information would predictably repeat with a known frequency. This would violate the intended "random" purpose of the data. As a result entropy is a limited resource on computing systems which must be filled up again with further random data. Sources of random data vary with systems, both in the sense of the operating system and the hardware platform that the operating system runs on. As an example, the Linux kernel takes timestamps of various events that are considered unpredictable [**?**]. These events are items that are not believed to happen on a regular basis, such as hardware interrupts from keyboard & mouse movement, network packets being received, low-level noise from enabled microphones, etc. By taking a timestamp in the microsecond range and adding each of those timestamps in, randomness can be derived. There are additional mathematics required, such as xor'ing the pool and hashing the received data to spread the entropy out evenly [**?**], but the basic idea is that unpredictable events form the basis of random numbers.

These sources of entropy fail under various circumstances however. If the rate of entropy usage exceeds the rate of entropy generation, issues have been known to occur [**?**]. These issues can result in the compromise of secure systems by allowing attackers to guess the secret random numbers used, thus breaking the encryption. These issues are more prevalent on embedded systems or virtual systems, where there is a lack of access to unpredictable events.

When there are no common sources of randomness available or those sources are unable to produce sufficient entropy dedicated hardware can be used to produce entropy. These devices use various methods to produce entropy. Some popular methods include measuring radioactive decay or examining quantum tunneling backwards via a diode [**?**]. By using these devices high quality entropy can be produced in order to keep the entropy pool full.

2. Background of Measuring Entropy in Systems

Measuring entropy in systems requires several different measurements to ensure that a sequence is truly random. There are various tests that have been devised to check the quality of a random number generator and we will use the tests provided by the "ent" open-source tool to perform a quality check on the numbers gathered [**?**].

The tests that the ent tool will check are as follows:

- Entropy: A measurement of the information density of the data. This could also be thought of as how compressible the data is.
- Chi-Square Test: This is a calculation of the chi-square distribution on the data and by interpreting the values output, once can come to a conclusion on how random the data is. This test is notably sensitive to issues with PRNG generators.
- Arithmetic Mean: The mean of the values input. If the mean is not near the center of the range then this is an indication that the values are not evenly distributed, but sway towards being high or low.
- Monte Carlo Value for Pi: This is a random test that, for truly random sequences, will converge on the value of pi, when given sufficient data.
- Serial Correlation Coefficient: This measures the extent to which a byte is based on the previous byte.

By interpreting the results of these measurements a determination of the quality of entropy can be determined. Each of the mea-

surements has a "high water mark" which can be derived by observing a truly random sequence. In the case of the ent measurements the random sequence is the decay of a radioactive isotope. By seeing how closely other random numbers match that, a determination can be made on the quality of the random number generator under test.

3. Conducting Measurements of Entropy in the "Lucky Numbers" of Fortune Cookies

The actual measurement of the numbers from fortune cookies required a crowdsourcing approach to obtain the numbers. Due to various budgetary and time constraints it proved infeasible to directly record a series of fortune cookie numbers. Instead a site was found that recorded lucky numbers from fortune cookies [**?**].

The site contained two categories of numbers: Pick Three, and Lucky Numbers. Combined, this led to several hundred entries. Pick Three was a series of three digit numbers used in various state lotteries. Lucky Numbers consisted of six numbers, each a maximum of two digits long, i.e. less than one hundred. There were 386 entries under the Pick Three category and 284 Lucky Numbers. This implies a total of 670 fortune cookies were used to obtain these results.

Once the numbers were dumped from the site they required some massaging to put them in a usable format. In order to maximize the entropy by removing repeating elements all non-numerical characters were stripped and the numbers from the site were treated as a single continuous number. No sorting was performed on the numbers beyond the order they were found on the site. Duplicate detection was performed and it was noted that there were no duplicate entries found. This resulted in a 4441 digit long number. However, the ent tool reads in numbers a byte at a time. In order to properly account for that the data was converted to base-255 and resulted in a 1844 character long file. Upon running the "ent" tool the following analysis was found:

- Entropy = 7.915610 bits per byte.
- Optimum compression would reduce the size of this 1844 byte file by 1 percent.
- Chi square distribution for 1844 samples is 210.39, and randomly would exceed this value 98.10 percent of the times.
- Arithmetic mean value of data bytes is 127.1350 (127.5 = random).
- Monte Carlo value for Pi is 3.231270358 (error 2.85 percent).
- Serial correlation coefficient is -0.022281 (totally uncorrelated = 0.0).

These values are not very good. The entropy is very high, nearly at the maximum of eight bits per byte. However the Chi square distribution reveals that the numbers are not actually very random and could be predictable. The numbers generated by using fortune cookies are probably not suitable for cryptographic applications unless one was entering the numbers into a headless server or other device unable to obtain better entropy.

By looking on Amazon, the best deal for fortune cookies was a 350-count box for $24.99 [**?**]. This would imply that each cookie costs seven cents and implies that obtaining an equivalent amount of cookies as would be needed to produce the entropy file would cost $47.84. As an interesting side-note a 1024-bit TLS connection with an algorithm that implements Perfect Forward Secrecy requires about 635 bytes of data [**?**]. Using Pick Three as a high bound of cookies required and Lucky Numbers as a low bound one can obtain that between $3.78 to $15.11 worth of fortune cookies (at the bulk 350 cookie rate) would be required to obtain sufficient entropy to complete the above SSL handshake. Since prices go down as more cookies are purchased at once, someone making use of this method would be encouraged to stock up on cookies to obtain better pricing on their entropy.

4. Entropy Analysis of Actual Lotto Numbers

The entropy analysis of fortune cookies proved disappointing since the numbers failed the tests for truly random numbers. As a benchmark for quality random numbers, the next step is to analyze the randomness of actual lottery numbers. For purposes of this paper, the California lotteries were used. They have the advantage of providing easy to download records of numbers drawn [**?**]. The Powerball and Mega Millions winning draws, sorted in descending order from the most recent date going backwards were used to generate the numbers.

Once the numbers were downloaded they were modified to be in the same format as the other numbers, that is a 4441 digit long number converted to base-255. No sorting was used. Any numbers from the draw that were over the 4441 digit length were discarded. The results were as follows:

- Entropy = 7.904676 bits per byte.
- Optimum compression would reduce the size of this 1844 byte file by 1 percent.
- Chi square distribution for 1844 samples is 240.37, and randomly would exceed this value 73.59 percent of the times.
- Arithmetic mean value of data bytes is 129.1562 (127.5 = random).
- Monte Carlo value for Pi is 3.166123779 (error 0.78 percent).
- Serial correlation coefficient is 0.014775 (totally uncorrelated = 0.0).

These results are much better than before. The most important test here is the Chi-square distribution, which indicates that the values tested appear to not be predictable. This means that they could be used in applications requiring truly random numbers.

4.1 Limits of Using Lotto Numbers - amount of useful entropy that can be gained

One should note that the downsides of using drawn lotto numbers is two-fold. The first is that there is only a limited number of lotto numbers that can be drawn. This puts a hard upper limit on the amount of entropy that can be obtained from this source.

The other issue is that if the source of the random numbers, i.e. which lotto is being used, is discovered, then this system will become entirely predictable. The author suggests the following unpredictable perturbation to this system:

1. Obtain a round globe.
2. Create a list of lotteries in which the user has access to the recorded drawn values.
3. Mark off equal sized stripes along the globe, each stripe coorresponding to a different lottery.
4. The user, sitting by the globe, closes their eyes.
5. The user spins the globe, and with their eyes still closed, pokes the globe with their index finger until the globe comes to a halt.
6. Opening their eyes, the user takes the results of that lottery for the entropy input.

This method is believed to add enough unpredictability to the results to make it safer to use this system of entropy generation, in so much as this system can be used. By adding the lotteries of as

many places as possible and keeping the list secret, the ability to use this method is improved.

5. Cost Analysis compared with Hardware based RNGs

To better understand the financial cost associated with fortune cookie random numbers vs. more traditional hardware based sources a survey of hardware based random number generators was conducted.

First off, several Intel processor chips have random number generation built in [**?**]. These provide, via an assembler instruction, on demand random number generation. Since a system can either be assumed to have this available or not have it available at the time of inspection, and it is not believed that this feature would lead to a purchasing decision, the cost of this is treated as free.

The next category of products is consumer grade USB keys for random number generation. These products come from a variety of places that would be considered suspect due to the lack of standards surrounding them. Some of the products seem to fail their own testing for quality random numbers [**?**]. Others appeared to appeal to American patriotism and increasing fear in consumers to appeal to sales without describing the product in detail [**?**]. In most cases, "ent" was used as the basis for measuring the quality of output of these devices. This is particularly worrisome since "ent" is not intended to be an in-depth measuring tool nor do any of the products surveyed appear to follow advice from NIST and other standards bodies about the proper generation and testing of random numbers. In general, these devices seem to cost between $20 - $50 dollars. The economic appeal of these devices is that they are not consumed upon usage, like a fortune cookies numbers are. Instead they can produce more value over time by continuing to fill up the entropy pool. A $50 model producing 350 kilobits / second of entropy will break even with the 350-count of fortune cookies sampled above in a few seconds and be increasingly cost-effective from there on out.

The final category of products is commercial grade random number generators. Finding detailed sales information for these was difficult, but one representative product was found [**?**]. A USB enabled Random Number Generator, certified against real standards, would cost approximately $1055 and produces 4 Mbps of random data. Almost as soon as it is enabled it will surpass the quantity of random numbers produced by fortune cookies, though it's value will require it to be used for a long time, otherwise cheaper solutions may be better.

6. Conclusion

In conclusion, relying on manually entered sources of entropy is a dangerous, time-consuming, and expensive proposition. The quality of the random numbers generated is not consistently high, and requires verification to ensure that it is high. Even if regulated lotteries are used, there are concerns over the lottery sequence being discovered and limits on the amount of entropy that can be produced. Entering the numbers also takes a significant amount of time as well as money to gather the data needed.

It is recommended that anyone needing to generate more random numbers than their system can support on it's own look into using a hardware based rng over using more dubious solutions.

References

[1] Hazewinkel, Michiel, ed. (2001), "Entropy", Encyclopedia of Mathematics, Springer ISBN 978-1-55608-010-4

[2] Linux man page - random, URL: https://linux.die.net/man/4/random, Accessed: 2017-01-09

[3] Linux Kernel Code, File: /drivers/char/random.c, Linux Kernel 3.4.43

[4] Nadia Heninger and Zakir Durumeric and Eric Wustrow and J. Alex Halderman Mining Your {P}s and {Q}s: {D}etection of Widespread Weak Keys in Network Devices, Proceedings of the 21st USENIX Security Symposium, Aug 2012

[5] Wikipedia, Comparison of hardware random number generators, URL: https://en.wikipedia.org/wiki/Comparison_of_hardware_random_number_generators, Accessed: 2017-01-09

[6] John Walker, Linux man page - ent, URL: http://www.fourmilab.ch/random/, Accessed: 2017-01-11

[7] Evan Islam, Fortune Cookie Message, URL: http://www.fortunecookiemessage.com/lotto_numbers.php, Accessed: 2017-01-11

[8] Amazon Inc., Box of Fortune Cookies, URL, https://www.amazon.com/Golden-Bowl-Fortune-Cookies-350-Count/dp/B000I07O10/?th=1, Accessed: 2017-01-11

[9] Understanding and Managing Entropy Usage, Bruce Potter, Sasha Wood, URL: https://www.blackhat.com/docs/us-15/materials/us-15-Potter-Understanding-And-Managing-Entropy-Usage.pdf, Accessed: 2017-01-11

[10] California Lottery Winning Numbers, URL: http://www.calottery.com/win/winning-numbers, Accessed: 2017-01-11

[11] Intel, Intel Digital Random Numbe Generator (DRNG), Revision 2.0, May 15, 2014

[12] OneRng, Theory Of Operation, URL: http://moonbaseotago.com/onerng/theory.html, Accessed: 2017-01-30

[13] TrueRNG, Home Page, URL: http://ubld.it/products/truerng-hardware-random-number-generator/, Accessed: 2017-01-30

[14] IDQ Quantis RNG, Product Listing, URL: http://www.idquantique.com/random-number-generation/quantis-random-number-generator/, Accessed: 2017-01-30