Understanding Information Entropy

Yi Mao

atsec information security corporation
9130 Jollyville Road, Suite 260
Austin, TX 78759
Tel: +1 512 615 7300
Fax: +1 512 615 7301
www.atsec.com

In discussions of random number generation (RNG), people often talk about the term "entropy" as if it was interchangeable with the term "random." For example:

- The random seed is taken from an entropy pool.
- Entropy bits are added to the pool from external sources such as mouse and keyboard activity, disk I/O operations, and specific interrupts.
- Cloned sibling Virtual Machines may have loads of entropy in each of their pools, but they are all the same entropy copied over from the same frozen state.
- A RNG seeded with insufficient entropy produces predictable keys.
- RNG failures are often rooted in bad entropy.
- Software systems face the security problem of lacking entropy.

Use of the term "entropy" in these examples suggests that entropy is a kind of bit-string that:

- has something to do with randomness
- can form a pool
- can be compared as same or different
- can be described as good or bad
- can be attributed as sufficient or insufficient

If this is meant to be a riddle, then what is the mysterious object (or concept) hidden behind the term "entropy"?

In his 1948 paper "A Mathematical Theory of Communication", Claude E. Shannon introduced a notion of quantifying the expected value of the information contained in a message, usually in units such as bits. This became the well-known *Shannon entropy*. Shannon entropy is a measure of the average information content one is missing when one does not know the value of the random variable.

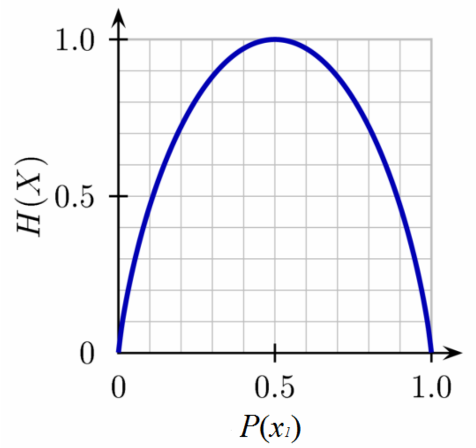
The core idea of entropy is that it is the expected **average** information content associated with a random **variable**. Mathematically, Shannon denoted the entropy *H* of a discrete random variable *X* with possible values $\{x_1, .., x_n\}$ and possibility $p(x_i)$ ($i=1,\ldots n$) for *X* taking value x_i as

$$H(X) = \sum_{i=1}^{n} p(x_i)\, I(x_i) = \sum_{i=1}^{n} p(x_i) \log_b \frac{1}{p(x_i)} = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i),$$

Where *b* is the base of the logarithm used and $I(x_i)$ denotes the information content of x_i. In particular, for *b*=2, the unit of entropy is a bit.

Consider tossing a coin with known probabilities of the coin landing with "heads" or "tails" showing. The entropy reaches the maximum value when the coin has equal probability ½ of landing with "heads" side up or "tails" side up (i.e. the coin is fair). In the case of fair coin, it is the most uncertain situation to predict the outcome of the next toss. Hence, the result of each toss of the coin delivers a full 1 bit of information.

Assuming that the probability of heads $p(x_1)$ = 1– the probability of tails $p(x_2)$, the correlation between the entropy $H(X)$ (i.e. the expected uncertainty) of a coin flip, measured in bits, and $p(x_1)$ can be illustrated by the graph below:



(i.e. the probability of x being heads)

In general, if a variable X has 2^n possible values and all of them have the equal possibility for being the real value of X, then the entropy $H(X)$ reaches its maximum n, meaning that the outcome of X being a certain value delivers the full n bits of information. It requires two necessary conditions for the entropy to maximally reach the full n bits of information:

1. There are 2^n possible values for X to take value from.
2. All possible values of X have the uniform probability distribution to be realized.

If a variable X has k possible values where $2^{n-1}<k<2^n$ for some n, the uniformly distributed probabilities of its possible values can still maximizes the entropy $H(X)$, but $H(X)$ will be less than n bits of information. The unevenly distributed probabilities of the possible values will further reduce the entropy $H(X)$.

In a reference book on cryptography titled “Fundamentals of Cryptology,” Professor Henk van Tilborg points out that one can give the following interpretations to the entropy $H(X)$ of a random variable X:

- The expected amount of information that a realization of X gives
- Our uncertainty about X
- The expected number of bits needed to describe an outcome of X

With these interpretations in mind, one can expect the entropy function $H(X)$ to have the following properties:

1. Adding another possible value to X but with probability 0 of realization does not affect the uncertainty about X.
2. The order of the stated possible values of X does not change the entropy of X.
3. The uncertainty of X is maximized if all of its possible values have equal probability of realization.

4. The expected number of bits necessary to describe an outcome from variable X increases proportionally to the number of possible outcomes of *X* with the uniformly distributed probability of realization.

Many entropy measures defined in information theory satisfy the above stated properties. Shannon entropy is just one of them. NIST SP 800-90 uses a very conservative measure *min-entropy* (i.e. $H_{min}(X)$) that is defined as

$$H_{min}(X) = -lg_2(P_{min}),$$

where P_{min} is the maximum probability among the probabilities of all possible outcomes. Min-entropy is a more conservative estimate of entropy than Shannon entropy, since min-entropy is always less than Shannon entropy.

Regardless of how the mathematical formula defining information entropy varies from one to another, the common feature of the mathematical definitions of information entropy is always about an attribute of a variable whose possible values fall within a given set with an associated probability for each possible value. So, there are two key components involved in the definition of information entropy: one is a given set of all possible values, and the other is the set of the associated probabilities. Entropy is a property directly related to a set as opposed to elements within the set. There is a fundamental difference between a set and the elements within. Below is a story to illustrate this difference.

A very hungry caterpillar finally felt full after she ate a slice of watermelon, a cupcake, a sausage, a piece of berry-cake and a lollipop. However, when she reflected on the cause (i.e. food) and effect (i.e. being full), she concluded that it was the lollipop that made her full and the other food she ate before the lollipop was completely a waste. From that point on, she decided to eat only one lollipop for each meal.

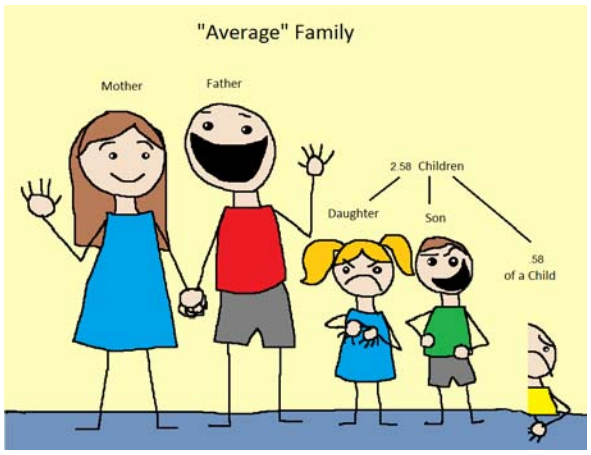
We know that being full is a cumulative result of eating the entire set of food illustrated in the picture above instead of a result tied up to a single item of food – the lollipop.

From the mathematical perspective, it is crystal clear that information entropy is a notion of an **average** information content carried in one realization of a possible outcome of a variable. However, most causal discussions of entropy (as exemplified at the beginning of this article), suggests that entropy bits are so real that one can actually point them out from this 32-bit string "10011000111011010001010100110110".

This is the puzzling, confusing, and mismatched idea that I will clarify below.

The description of the .58 child in an average family given in the children's book "The Phantom Tollbooth" by Norton Juster and Jules Feiffer paints a vivid picture of how a child might perceive this notion of "average" and explains the mismatch between the formal mathematical definition of information entropy and casual discussions of entropy.

The story from the book roughly goes like this: The character named Milo met the .58 child who is one half of a small child being divided neatly from top to bottom and asked sympathetically, "What is the rest of your family like?" "Oh, we're just the average family," the child said thoughtfully; "mother, father, and 2.58 children – and, as I explained, I'm the .58."



"It must be rather odd being part of a person," Milo remarked. "Not at all," said the child. "Every average family has 2.58 children, so I always have someone to play with. Besides, each family also has average of 1.3 automobiles, and since I'm the only one who can drive three tenths of a car, I get to use it all the time."

"But averages aren't real," objected Milo; "they're just imaginary."

I hope the above story helps everyone understand that entropy bits are not real bits in a bit-string, and they do not form a pool called an "entropy pool." Entropy bits are just like the .58 child in an average family. They are not real. One cannot point to them in the same way as he points to a real child in any real family. The number of children varies for an average American family, or for an average Indian family, or for an average Chinese family. The notion of average is tied up to a set (e.g. the set of all American families, the set of all Indian families, the set of all Chinese families, and so on) based upon which the average is calculated. As such, information entropy, being the average information content, is also tied up to the underlying set (which consists of all possible values of the variable whose entropy is in question).

With this clarification in mind, the misperception that the entropy of a bit-string pool gets reduced after a certain value from the pool is observed should vanish. Consider a sequence of coin tossing events again. Observing a “heads”, does not reduce the uncertainty of what the outcome will be for the next toss (provided that the coin is fair).

By contrast, the size of the pool and the probability of occurrences for each bit-string in the pool has a significant impact on the resulting entropy. As a matter of fact, they are the two determining factors in the mathematical formula to calculate the information entropy. For example, consider a set of key-stroke timings with a millisecond incremental timer. Assume that normally a key stoke takes 0.5 second plus/minus 0.1 second. Then there are 200 possible values to measure the time that a key stroke takes. Assuming that all of these 200 possible values have the identical probability of being the real timing of a key stroke, then it amounts to 7-8 bits of entropy for the set of possible timings for a key stroke. If the timer has a higher resolution and provides a microsecond increment, then there will be 200,000 possible timings for a key stroke. Still keeping the assumption that all possible values have the equal chance to be the real timing for a key stroke, and then the resulting entropy of that much enlarged set of possible timings will be 17-18 bits.

In theory, the probability distribution is given prior to the calculation of the entropy. In reality, the probability of each outcome is obtained through the statistical means. That all possible outcomes have an identical probability for realization is a big assumption. This assumption is often checked against the statistical test suite specified in NIST SP 800-22 (http://csrc.nist.gov/publications/nistpubs/800-22-rev1a/SP800-22rev1a.pdf). A sequence of outcomes that passes this NIST test suite statistically confirms that the outcomes are indeed evenly distributed, with a more or less identical probability of occurrences. The larger the sequence of outcomes that pass the test suite; the higher the assurance that each outcome has the same probability of occurrences is. The commonly used entropy estimate tool ENT (http://www.fourmilab.ch/random/) calculates the entropy value based on the probability distribution that is also gathered by the statistical means over a collection of bytes. Again, the more bytes that are run through ENT, the more accurate the entropy estimate will be. Due to the nature of the statistical approach, there is no guarantee that the sequence of the outcomes is truly random even though it may have passed the test suite. The statistical test suite should be used in conjunction with an analytic approach that provides the rationale for the randomness of outcomes.

In conclusion, the term “entropy” in information technology literature is defined as a mathematical measure of randomness, but it’s also often used interchangeably with the word “random,” leaving out the idea of it being a numerical measure of randomness. If this is just used as a kind of shorthand for convenience’s sake, that’s fine. What’s important is that deep down to the core, we do understand what information entropy really is, the determining factors of entropy, how to analyze and justify an entropy source, and how to assess the quality of an entropy input. A good understanding of all these questions and the ability to answer them correctly are crucial in assessing the quality of a random number generator.