An examination of the DRBG algorithms in this document reveals a common feature; each of them takes a value (referred to as a *seed*) and applies an algorithm to produce a potentially large number of pseudo-random bits. The most important feature of the interaction between seed and algorithm is that if an adversary doesn't know the seed then he can't tell the difference between the pseudo-random bits and a stream of truly random bits. Conversely, if he knows (or can guess) the seed then, since the algorithm is deterministic, he can easily distinguish the output from random. Thus the security of the RBG output is directly related to the adversary's inability to guess the seed. The *entropy source* is the critical component of an RBG which provides un-guessable values for the deterministic part of the RBG to use as seeds for the random bit generation process.

Every entropy source must include some process which is unpredictable. An intuitive (though impractical) example is tossing a coin and recording the sequence of heads and tails. More likely it will be an electronic process such as a noisy diode, which receives a constant input voltage level and outputs a continuous, normally distributed analog voltage level. Other possibilities include thermal noise or radioactive decay, measured by appropriate instruments. The unpredictability could involve human interaction with an otherwise deterministic system, such as the sampling of a high-speed counter whenever a human operator presses a key on a keyboard. In any case, there must be something happening which is unpredictable to an adversary. It could be fundamentally unpredictable (like when the next particle is detected by a Geiger counter), or it could be practically unpredictable (the adversary won't know the exact value of the high-speed counter, if he isn't close enough to the human pressing the key).

Since the most important thing is that the values be unpredictable, there must be a way to measure how unpredictable they are. The general term for this measure of uncertainty is *entropy*. There is a whole family of entropy measures; this document uses a very conservative one, namely *min-entropy*. Suppose that the unpredictable process produces one of n possible outputs or sequences of outputs at each time interval, with the i^{th} possible outcome having probability p_i. The min-entropy of the outputs is $-\log_2(\max\{p_i\})$. This represents the best case work for an adversary who is trying to guess an output from the entropy source.

One useful fact about min-entropy is that if two samples are independent then the entropy of their concatenation is the sum of their entropy. This makes sense; if the samples are independent, then guessing one sample provides no information for guessing the other one. An entropy source will likely combine several samples from the underlying unpredictable process to produce an output.

Once sufficient samples have been gathered, they generally need to be converted to bits (e.g. an analog voltage will be mapped to some digital value). Some entropy sources will perform further processing, perhaps guaranteeing unbiased output. An entropy source may even process the bits to the point where the output bitstring will have full entropy; i.e. the entropy of the bitstring will be (nearly) the same as its length. These are choices for the entropy source designer to make. The only thing this document assumes is that the entropy source produces a bitstring with a conservative estimate of its min-entropy.

It's worth reiterating that the entropy produced by the entropy source is always relative to the adversary and his ability to observe/predict its behavior. If the adversary has no uncertainty about the output, then the entropy produced is zero (and so is the security of the relying application). If the adversary is in a position to observe the outputs, the entropy is zero. And if the adversary has more knowledge of the behavior of the entropy source than he's given credit for (in fact, he might have a better model for its behavior than the designer), or if he can manipulate it so that the outputs are more predictable for him, then the entropy may be greatly reduced. This discussion motivates the need for a well-defined security boundary for the entropy source. It's important to know the adversary's limits of knowledge of the entropy source to get an accurate entropy estimate.