Security of PNGs and semantic security of stream ciphers

In these notes I will just look in a little more detail at the security material on pseudo random generators (PRGs) and the semantic security of the associated stream ciphers covered in "Stream Ciphers 4" in Week 1 of the Stanford cryptography course.

Pseudo Random Generators

For our purposes, a PRG is a function $G : \mathcal{K} \to \mathbf{2}^n$ where the key space $\mathcal{K}$ is generally $\mathbf{2}^s$ for some s which is much smaller than n. The goal in having a PRG is that its output should, in some sense, look like it's coming from $\mathbf{2}^n$ by choosing elements uniformly at random.

Under the assumption that $\mathcal{K}$ is much smaller in size than $\mathbf{2}^n$ this is of course impossible since the image of G is a tiny part of the set $\mathbf{2}^n$. In particular, suppose that we presented an adversary with a string and asked them "I generated this either uniformly at random in $\mathbf{2}^n$ or, by choosing $k \in \mathcal{K}$ and then giving you $G(k)$, which was it?" If they were able to identify the elements of the image of G then they could guess correctly with probability 1 in the case where we had used G and with probability $1 - |\mathcal{K}|/2^n$ if we hadn't simply by guessing G if the string was in the image, and random if it wasn't.

So in particular if it's easy to identify the image of G then a PRG is fundamentally broken. But this "identification" is a computational task – and if that task is difficult it's reasonable to say that G might still be secure. But asking for the identification of the range of G is too weak a criterion for (in)security. For instance if we had some algorithm that guessed whether a string was in the image of G then even if it was correct only $3/4$ of the time our opponent would have a significant advantage. Let's try to make this idea precise.

Statistical Tests and Advantage

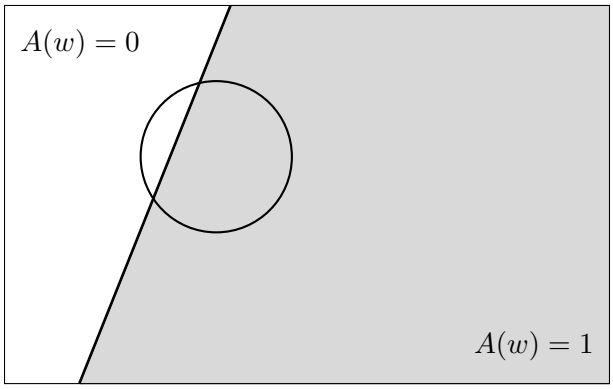
A statistical test A against a PRG G is a formalisation of the protocol sketched above. Namely it is an *efficient* algorithm which runs on input from $\mathbf{2}^n$ that outputs 0 or 1 (which could be interpreted as "not random" and "random" but are really just arbitrary labels). We are interested in the question of whether A performs differently on pseudo random input versus truly random input. So we define the *advantage* of A to be:

$$\mathrm{Adv}(A, G) = \left|\Pr_{k \leftarrow \mathbf{U}(\mathcal{K})}\left[A(G(k)) = 0\right] - \Pr_{r \leftarrow \mathbf{U}(\mathbf{2}^n)}\left[A(r) = 0\right]\right|.$$

where $k \leftarrow \mathbf{U}(\mathcal{K})$ means "k chosen uniformly at random from $\mathcal{K}$ and similarly $r \leftarrow \mathbf{U}(\mathbf{2}^n)$ means "r chosen uniformly at random from $\mathbf{2}^n$."

Note that we only use one of the outcomes in defining the advantage – but this doesn't matter since probabilities sum to one so any difference between G and "random" on value 0 is exactly balanced by the difference between G and "random" on value 1.

In other words, the advantage of A over G is the difference in the probability that A finds pseudo random strings look random, from the probability that it finds truly random strings look random (note that there is no actual requirement that A do a good job of identifying either type!) If the PRG is doing what it's supposed to then it should not be possible to find a statistical test with "non-negligible" advantage over it. This idea is illustrated below:



The large outer rectangle represents all of $\mathbf{2}^n$. The test is some algorithm A which returns either 1 or 0. The region where it returns 1 is represented in gray. Though we interpret this as A saying "this is random" note there's no requirement that this actually be the case. The circle represents the image of some pseudo random generator G, i.e., all the strings of the form $G(k)$ for some k in $\mathbf{2}^s$. The *advantage* of this test A against this PRG is the absolute value of the difference between the proportion of gray area to total area in the circle (the probability that "A thinks an output from the PRG is random"), and the proportion of gray area to total area in the rectangle – assuming of course that the areas faithfully represent the number of strings they contain. If this advantage is 0 or very close to it then A cannot be used to guess effectively whether strings are output from the PRG or from a truly random selection. If this advantage is significantly different from 0 then A provides evidence for (or against) a string being an output from the PRG.

Therefore, we say that G is secure if there is no efficient statistical test A whose advantage over G is non-negligible. Notice that the word "efficient" here is critical – otherwise we could simply use brute force – list all the strings $G(k)$ and check whether the given string is among them. Are there any secure PRGs? Well, we don't know for sure because if we did we would have resolved the $P = NP$ problem. The idea is very simple - the PRG G is an algorithm in P. So the test "Output 0 if for some $k \in \mathcal{K}$ the string you are testing is $G(k)$" is in NP. This test certainly has non-negligible advantage (it's the test of the previous section) so it had better not be efficient – but if $P = NP$ then it could be converted into an efficient test. So the existence of a secure PRG requires

$P \neq NP$.

An important theorem that we won't prove (one direction is easy, the other not so much) is that security in the sense above is equivalent to unpredictability (where by unpredictability we mean that there is no bit i whose value in $G(k)$ can be efficiently predicted from the preceding bits.)

Semantic Security of Stream Ciphers

Since stream ciphers use a particular key only once, when we are considering the security of these we can assume that the attacker has access to only one ciphertext. What then should we mean when we say that such a cipher is secure? The idea is to return to Shannon's definition of perfect secrecy that, for any two messages the distribution of possible ciphertexts for the two messages be identical over the random choice of key. However, this is achievable only if the key length is at least as great as the message length. Instead we demand that these two distributions be "computationally indistinguishable" at least for messages which the attacker can exhibit.

I tend to think of this as requiring that an adversary not be able to tell the difference between a message from Alice to Bob that says "Buy" versus one that says "Sell" – this is not quite the same thing, but pretty nearly.

So now we have an encoding algorithm E. The adversary chooses two messages m_0 and m_1 of equal length. We choose a random key k as usual and return either $E(k, m_0)$ or $E(k, m_1)$. The adversary now uses an efficient algorithm on $E(k, m_x)$ to guess whether $x = 0$ or $x = 1$. Suppose they guess $x = 1$ – their "advantage" is the difference in the probability that this happened according to whether we chose m_0 or m_1 (in absolute value, since if they are consistently wrong, they can take advantage of that too!)

So let W_0 be the event "the algorithm guesses $x = 1$ when in fact $x = 0$" and W_1 be the event "the algorithm guesses $x = 1$ when in fact $x = 1$". These events are relative to the random choice of k alone, and we define:

$$\mathrm{Adv}(A, E) = |\Pr(W_0) - \Pr(W_1)| .$$

We say that E is *semantically secure* if there is no efficient adversary having non-negligible advantage.

Secure PRGs give Semantically Secure Stream Ciphers

It seems natural to believe that if we build a stream cipher out of a secure PRG then it should be semantically secure. The idea is to show that the advantage of the adversary in any efficient semantic security game is bounded by the sum of the advantages of two efficient adversaries in the PRG security game (in fact these two adversaries are actually the same, but that's not relevant). So, if the PRG is secure, those two advantages are negligible, and hence so is the advantage of the semantic security adversary.

Consider a semantic security adversary A and let W_0 and W_1 be as above. We allow the encoder to cheat and encode using a true one time pad. We let R_0 be the event "the algorithm guesses $x = 1$ when in fact $x = 0$ and a one time pad was used" and R_1 be the event "the algorithm guesses $x = 1$ when in fact $x = 1$ and a one time pad was used". Note that the output from a one time pad is a truly random string (every output is equally likely whether $x = 0$ or $x = 1$) so it must be the case that $\Pr(R_0) = \Pr(R_1)$.

The triangle inequality implies:

$$\begin{aligned} |\Pr(W_0) - \Pr(W_1)| &\leq |\Pr(W_0) - \Pr(R_0)| + |\Pr(R_0) - \Pr(R_1)| + |\Pr(R_1) - \Pr(W_1)| \\ &= |\Pr(W_0) - \Pr(R_0)| + |\Pr(R_1) - \Pr(W_1)| \,. \end{aligned}$$

But each of the two remaining terms is the advantage of a statistical test against the PRG. Namely, $\Pr(\mathrm{W}_0)$ is the probability of the event that an efficient algorithm outputs 1 when given $G(k) \oplus m_0$ for some randomly chosen key k whereas $\Pr(\mathrm{R}_0)$ is the probability of the event that the same efficient algorithm outputs 1 when given $r \oplus m_0$ for some truly random r.

So, each of these two terms is negligible, and hence the stream cipher is secure.

The other direction (if the PRG is insecure then so is the stream cipher) is easier - the semantic security adversary can just supply $m_0 = 0$ and m_1 a truly random string. They can then apply their efficient test with non-negligible advantage against the value returned by the challenger (which is a random $G(k)$ if $x = 0$ and a truly random string if $x = 1$), and their advantage is exactly the same as that in the PRG security game.