UConn, School of Computing
Fall 2024
CSE 3400/CSE 5850: Introduction to Computer and Network Security / Introduction to Cybersecurity

Supplementary PRG Exercises

Full formal proofs
Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be a PRG. For each of the following constructions, prove formally whether it is a PRG or not.

1. $G_1(s||t) = \bar{t} \parallel G(s)$
2. $G_2(s||t) = \bar{s} \parallel G(s) \parallel t$
3. $G_4(s) = G(s) \parallel \texttt{lsb}(G(s))$ (where `lsb` of a string is the last bit of the string)

($\parallel$ is the concatenation operation and $\oplus$ is the bitwise XOR operation)

Solutions:

1. This PRG scheme is secure. Intuitively, it is simply a concatenation of a pseudorandom string (the output of $G(s)$) and a random string $\bar{t}$ (the negation of a random string is also a random string).

   We proceed by reduction to prove this (by contrapositive). We assume that G_1 is insecure, meaning that there exists a PPT attacker $\mathcal{A}'$ that breaks G_1 with non-negligible advantage. That is, $\mathcal{A}'$ has the following advantage in distinguishing G_1:

   $$\varepsilon^{PRG}_{\mathcal{A}',G_1}(n) = \Pr[\mathcal{A}'[(G_1(s||t)] - \Pr[\mathcal{A}'(r)]$$

   where r is a truly random string and $\varepsilon(n)$ is a non-negligible function.

   We show how to build another PPT distinguisher or attacker $\mathcal{A}$ against G. $\mathcal{A}$ can utilize $\mathcal{A}'$ as a subroutine in order to break PRG security of G. $\mathcal{A}$ has the following behavior:

   - $\mathcal{A}$ receives an input string (denoted w) and wants to determine if it is from G or a true random string.
   - $\mathcal{A}$ generates a string t and inverts it (making $\bar{t}$), then concatenates it with w to create $\bar{t}||w$
   - $\mathcal{A}$ passes $\bar{t}||w$ as a challenge to $\mathcal{A}'$
   - Then, $\mathcal{A}$ simply outputs what $\mathcal{A}'$ outputs:

   $$\mathcal{A}(w) = \begin{cases} 1 & \text{if } \mathcal{A}'(\bar{t} \parallel w) = 1 \\ 0 & \text{if } \mathcal{A}'(\bar{t} \parallel w) = 0 \end{cases}$$

   We can now illustrate $\mathcal{A}$'s advantage in distinguishing G, which is the same as the advantage of $\mathcal{A}'$ to distinguish *G'*:

   $$\varepsilon^{PRG}_{\mathcal{A},G}(n) = \Pr[\mathcal{A}(G(s)) = 1] - \Pr[\mathcal{A}(r) = 1]$$

Since $\mathcal{A}$ simply outputs whatever $\mathcal{A}'$ outputs, then its advantage is equals to $\mathcal{A}'$s advantage. Thus, we see that if G_1 is insecure, then G must be insecure as well. However, this contradicts the assumption that G is a secure PRG. Therefore, G_1 must be a secure PRG.

2. This PRG scheme is not secure. Intuitively, the seed of the PRG is given in the output. Knowing the seed of the PRG breaks its security. This is because the construction of the PRG is public (only the seed is secret). If the attacker knows the seed, it can compute G over the seed and check if the output equals to the challenge.

   In detail, we can build a PPT distinguisher $\mathcal{A}$ that has the following behavior:

   - On input the challenge string (let it be denoted as w which could be either $G_2(s\|t)$ or r) $\mathcal{A}$ extracts the first n bits of the output, then invert them to obtain a string call it x.
   - $\mathcal{A}$ computes $G(x)$ (recall construction of G is known).
   - After that $\mathcal{A}$ extracts the next $2n$ bits of w (which corresponds to $G(s)$), let's denotes these $2n$ bits as y. $\mathcal{A}$ outputs the following:
     - 1 if $x = y$ meaning that $\mathcal{A}$ thinks that w is the output of G_2.
     - 0 if $x \neq y$ meaning that $\mathcal{A}$ thinks that w is a true random string.

   In the case that w is the output of G_2, then $\mathcal{A}$ will always win ($\Pr[A(G_2(s \parallel t))] = 1)$).

   In the case of a truly random string r, this property is not guaranteed. The probability that the first n bits are the seed for G is 2^{-n}, which means the probability that $\mathcal{A}$ will mistakenly identifies r as the output of G_2 is $\Pr[A(r)] = \frac{1}{2^n}$).

   Thus, the advantage of A in the PRG security game is $1-2^{-n}$, which is non-negligible, so G_2 is not a secure PRG.

3. This PRG scheme is not secure. Intuitively, an attacker A can simply check the least significant two bits of the input string in the PRG security game. If they are identical, it outputs 1 (meaning that the input string is the output of G_4), else, it outputs 0 (meaning that A thinks that the input string is a true random one). The advantage of this attacker in winning the PRG security game is non-negligible.

   In detail, build a PPT distinguisher or attacker $\mathcal{A}$ that has the following behavior:

   - On input the challenge string (denote it as w) $\mathcal{A}$ extracts the last two bits of w.
   - If these two bit are identical, then A outputs 1, otherwise it outputs 0.

   Now we analyze the advantage of A. If w is the output of G_4, then all the time its last two bits will be identical. Thus, A will be able to identify G_4 with probability 1. So

   $$\Pr[\mathcal{A}(G_4(s))] = 1$$

   In the case of a truly random string r, this property is not guaranteed. The probability that the last two bits of r to be identical is when r happens to have these as 00 or 11.

This happens with probability $\frac{1}{2}$ (the probability to have a string with 00 is 0.25 and same to have it with 11 as the least two bits, the sum of both is 0.5).Thus:

$$\Pr_{r \xleftarrow{\$} \{0,1\}^{2n+1}} [\mathcal{A}(r) = 1] = \frac{1}{2}$$

Thus, the advantage for $\mathcal{A}$ to distinguish the output of G_4 from a true random string is given as follows:

$$\varepsilon^{PRG}_{\mathcal{A},G_4}(n) = \Pr[\mathcal{A}(G_4(s)) = 1] - \Pr_{r \xleftarrow{\$} \{0,1\}^{2n+1}} [\mathcal{A}(r) = 1] = 1 - 0.5 = 0.5$$

The value 0.5 is non-negligible, so G_4 is not a secure PRG.