Cryptography: HW2

Trevor Bramwell

Professor Mike Rosulek
January 23, 2015

1 Negligible Functions

Unless otherwise stated, all the following functions are negligible.

$$\frac{1}{2^{n/2}} = \frac{1}{2^{1/2}} * \frac{1}{2^n}$$

We know $\frac{1}{2^n}$ is negligible. Since this takes the form of $p(x)f(x)$ where $p(x)$ is a polynomial and $f(x)$ is negligible, $p(x)f(x)$ is negligible as well.

$$\frac{1}{2^{\log{(n^2)}}} < \frac{1}{2^{log(n)}} < \frac{1}{n^c}$$

Becuase $(\log n)$ grows at a quicker rate than the constant c, $\frac{1}{2^{\log{(n)}}}$ will asymptotically approach zero faster than $\frac{1}{n^c}$.

$$\frac{1}{n^{\log{(n)}}} < \frac{1}{n^c}$$

$$\frac{1}{2^{(\log n)^2}} < \frac{1}{2^{(\log n)}} < \frac{1}{n^c}$$

$$\frac{1}{(\log n)^2} > \frac{1}{\log n} > \frac{1}{n^c} \text{ - Not Negligible.}$$

$$\left(\frac{1}{n^{1/n}} = \frac{1}{n^{-n}} = n^n\right) > \frac{1}{n^c} \text{ - Not Negligible.}$$

$$\left(\frac{1}{\sqrt{n}} = \frac{1}{n^{1/2}}\right) >= \frac{1}{n^c} \text{ - Not Negligible.}$$

$$\frac{1}{2^{\sqrt{n}}} < \frac{1}{2^n} < \frac{1}{n^c}$$

$$\frac{1}{n^{\sqrt{n}}} < \frac{1}{n^{\log n}} < \frac{1}{n^c}$$

2 Distinguisher for the injective PRG: G.

a $\text{bias}(A, \mathcal{L}^G_{\textsf{prg-real}}, \mathcal{L}^G_{\textsf{prg-rand}}) = \frac{1}{2^{n+\ell}}$

This distinguisher is no better than a random guess, except that it randomly guesses all possible outputs.

b This does not contradict G being a PRG. The security property of $\mathcal{L}^G_{\textsf{prg-real}}$ and $\mathcal{L}^G_{\textsf{prg-rand}}$ still holds.

c When the bias of G is not injective, the bias has the possibility of increasing a non-negligible amount, thus rendering $\mathcal{L}_{\textsf{prg-real}}$ insecure.

3 Secure Length-Tripling PRG

(a)

$$A \diamond \boxed{\begin{array}{l} \underline{\text{QUERY}():} \\ \quad s \leftarrow \{0,1\}^{2n} \\ \quad x := \text{G}(s_{left}) \\ \quad y := \text{G}(s_{right}) \\ \quad \text{return } x \oplus y \end{array}}$$

(b)

$$A \diamond \boxed{\begin{array}{l} \underline{\text{QUERY}():} \\ \quad s_{left} \leftarrow \{0,1\}^{n} \\ \quad s_{right} \leftarrow \{0,1\}^{n} \\ \quad x := \text{G}(s_{left}) \\ \quad y := \text{G}(s_{right}) \\ \quad \text{return } x \oplus y \end{array}}$$

(c)

$$A \diamond \boxed{\begin{array}{l}\underline{\text{QUERY}():}\\ \quad s_{left} \leftarrow \{0,1\}^n \\ \quad x := \mathrm{G}(s_{left}) \\ \quad s_{right} \leftarrow \{0,1\}^n \\ \quad y := \mathrm{G}(s_{right}) \\ \quad \text{return } x \oplus y \end{array}}$$

(d)

$$A \diamond \boxed{\begin{array}{l}\underline{\text{QUERY}():}\\ \quad x := \text{QUERY'}() \\ \quad s_{right} \leftarrow \{0,1\}^n \\ \quad y := \mathrm{G}(s_{right}) \\ \quad \text{return } x \oplus y \end{array}} \diamond \boxed{\begin{array}{l}\underline{\text{QUERY'}():}\\ \quad k \leftarrow \{0,1\}^n \\ \quad \text{return } G(k) \end{array}}$$

(e)

$$A \diamond \boxed{\begin{array}{l}\underline{\text{QUERY}():}\\ \quad x := \text{QUERY'}() \\ \quad s_{right} \leftarrow \{0,1\}^n \\ \quad y := \mathrm{G}(s_{right}) \\ \quad \text{return } x \oplus y \end{array}} \diamond \boxed{\begin{array}{l}\underline{\text{QUERY'}():}\\ \quad k \leftarrow \{0,1\}^{3n} \\ \quad \text{return } k \end{array}}$$

(f)

$$A \diamond \boxed{\begin{array}{l}\underline{\text{QUERY}():}\\ \quad x \leftarrow \{0,1\}^{3n} \\ \quad s_{right} \leftarrow \{0,1\}^n \\ \quad y := \mathrm{G}(s_{right}) \\ \quad \text{return } x \oplus y \end{array}}$$

(g)

$$A \diamond \boxed{\begin{array}{l}\underline{\text{QUERY}():}\\ \quad x \leftarrow \{0,1\}^{3n} \\ \quad y := \text{QUERY'}() \\ \quad \text{return } x \oplus y \end{array}} \diamond \boxed{\begin{array}{l}\underline{\text{QUERY'}():}\\ \quad k \leftarrow \{0,1\}^n \\ \quad \text{return } G(k) \end{array}}$$

(h)

$$A \diamond \boxed{\begin{array}{l} \underline{\text{QUERY}():} \\ \quad x \leftarrow \{0,1\}^{3n} \\ \quad y := \text{QUERY'}() \\ \quad \text{return } x \oplus y \end{array}} \diamond \boxed{\begin{array}{l} \underline{\text{QUERY'}():} \\ \quad k \leftarrow \{0,1\}^{3n} \\ \quad \text{return } k \end{array}}$$

(i)

$$A \diamond \boxed{\begin{array}{l} \underline{\text{QUERY}():} \\ \quad x \leftarrow \{0,1\}^{3n} \\ \quad y \leftarrow \{0,1\}^{3n} \\ \quad \text{return } x \oplus y \end{array}}$$

(j)

$$A \diamond \boxed{\begin{array}{l} \underline{\text{QUERY}():} \\ \quad z \leftarrow \{0,1\}^{3n} \\ \quad \text{return } z \end{array}}$$

Justification of steps:

(a) This is just $A \diamond \mathcal{L}^{G}_{\textsf{prg-real}}$ with the deatils of H filled in.

(a)⇒(b) We have split s into it's $left$ and $right$ parts. No effect on the program.

(c)⇒(d) Factored out s_{left} and first call to G in it's own subroutine $QUERY'()$. No effect on the program.

(e) We swap the call to the length-tripling PRG G. Assuming G is a secure PRG, we can replace it's call with the generation and return of $3n$ random-bits.

(e)⇒(f) Inlining of $QUERY'()$. No effect on the program.

(g)⇒(i) The process of (d) through (f) is repeated for y.

(j) Because we know that XOR produces a random distribution at uniform, it is simple to state that this process is the same as generating $3n$ random-bits $\square$

4 Length-Quadrupaling PRG Distinguisher

$$\boxed{\begin{array}{l} \underline{\mathrm{A}():} \\ \quad s \leftarrow \{0,1\}^n \\ \quad x := \mathrm{H}(s) \\ \quad \text{return } x_{left} = x_{right} \end{array}}$$

To show that H is **not** a PRG the above distinguisher is given. Because H uses G multiple times with the same s, x and y equal. Thus the bias for A is:

$$\begin{aligned} &\text{bias}(A, \mathcal{L}^H_{\textsf{prg-real}}, \mathcal{L}^H_{\textsf{prg-rand}}) = \\ &\qquad |Pr[A \diamond \mathcal{L}^H_{\textsf{prg-real}} \text{ outputs } 1] - Pr[A \diamond \mathcal{L}^H_{\textsf{prg-rand}} \text{ outputs } 1]| = \\ &\qquad |1 - 0| = \\ &\qquad 1 \end{aligned}$$