Homework #2

Sam Quinn

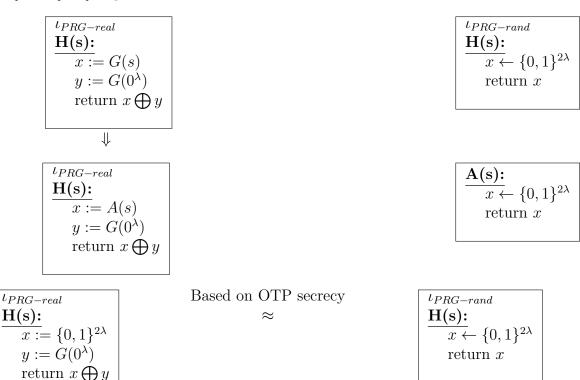
January 27, 2016

1

Which of the following are negligible functions? Justify your answers.

- $\sqrt{\frac{\lambda}{2^{\lambda}}} \to 0$ Is **negligible**, the denominator grows exponentially while the numerator grows linear making this equation approach 0.
- $\frac{1}{2^{log(\lambda^2)}} \to \infty$ Is **Not negligible**, as the values of λ continue to increase there are polynomial values that could grow faster in the numerator. Thus this equation approaches ∞ .
- $\frac{1}{\lambda^{log(\lambda)}} \to 0$ Is **negligible**, the denominator will grow faster than the numerator. As λ gets larger it is also raised to the log of itself which will eventually on the path to ∞ surpass any numerator making it approach 0.
- $\frac{1}{\lambda^2} \to \infty$ Is **not negligible**, because the denominator in this equation grows linearly there are numerators raised to a constant that could grow faster than the denominator. This equation approaches 0.
- $\frac{1}{2^{(\log \lambda)^2}} \to 0$ **Is negligible**, because the denominator will grow exponentially while the numerator will grow linear. This equation will approach 0.
- $\frac{1}{\sqrt{\lambda}} \to \infty$ This is **not negligible**, the numerator will grow much faster than the denominator which is a $\sqrt{\lambda}$. This equation approaches 0.
- $\frac{1}{2^{\sqrt{\lambda}}} \to 0$ Is **negligible**, even though the denominator has an exponent that is a $\sqrt{\lambda}$ it is still a positive exponent making the denominator grow faster than the numerator. This equation approaches 0.

Let $G: \{0,1\}^{\lambda} \to \{0,1\}^{2\lambda}$ be a length doubling PRG, and consider then algorithm $H: \{0,1\}^{\lambda} \to \{0,1\}^{3\lambda}$ given below:



In the first step we are able to detach the x assignment in to a separate function named A(). Because we assume that G() is a secure PRG that returns a uniformly random number every time we could make A() a substitute with this assumption we can eliminate s. We **cannot** get ys value from A() because if we pass the same value in each time the PRG will return the same random number. Because OTP security we can say that H() is secure based on the fact that we always will have a random x and when \bigoplus with the same number y it will provide total secrecy.

3

Show that H is not a secure PRG (even if G is). Describe a successful distinguisher for $\iota_{prq-real}^H$ and $\iota_{prq-rand}^H$. Explicitly compute its advantage.

```
\frac{\mathbf{H}(\mathbf{s}):}{x := G(s)}
return s \| x
```

```
\frac{\mathbf{H}(\mathbf{s}):}{x \leftarrow \{0,1\}^{3\lambda}} return x
```

```
Query(s):

x = \text{First } \lambda \text{ bits of } s

x' = \text{Last } 2\lambda \text{ bits of } s

z = G(x)

y = \text{Last } 2\lambda \text{ bits of output from } G(s)

return x' == y
```

To determine which world we are in we could create a distinguisher function named Query() above. The first thing we would have to do is get the full output from H() and pass that into Query(). Inside of Query() we extract the first λ bits off the front of s and assign it to s with the remaining going into s. S would pass s into the same S world. If the output from S is the same as S we have determined that we are in the $U_{PRG-real}$ world.

4

Suppose F is a secure PRF. Define the following function F' as:

$$F'(k, x || x') = F(k, x) | F(k, x \bigoplus x')$$

x and x' are each in bits long, where in is the input length of F. Show that F' is not a secure PRF (even if F is). Describe a distinguisher and compute its advantage.

Hint: Remember, you are not attacking F. In fact, F may be the best PRF in the world. You are attacking the faulty way in which F' uses F.

$$x = 1111$$
 $F'(k, x || x') = F(k, x) || F(k, x \bigoplus x')$
 $x' = 0000$ $F'(k, x || x') = 1111 || 1111 \bigoplus 0000$
 $x || x' = 11110000$ $F'(k, x || x') = 11111111$

We can exploit F' by taking advantage of the \bigoplus with a zero string. Every string \bigoplus with 0 returns itself. So now if we pass in a string where the second half of the string is all 0's then we can prove that we will get repeating results back consistently in this case all 1's.