Andrew Johnson

CS 427 Crypto

Homework #2

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

We see that the denominator of the final fraction in our limit will head towards infinity faster than the numerator. This causes the limit to look like,. Thus this function is negligible□

We see that if we use a polynomial with a power greater than zero then we get something that is not equal to zero, thus making this equation non-negligible□

Since c < ∞, and log(λ) →∞ as λ→∞, then our final equation gives . This makes the given equation negligible□

We can see that there is no option of this function being able to go to 0. So this function is non-negligible□

If we look at as λ→∞, we see that it goes towards -∞. This is why the limit goes to 0. Proving that this equation is negligible­□

We see that if c > 0, then our final function doesn’t go to 0. Thus this function is non-negligible□

Just like in one of the previous functions. There is no way for this function to go to 0, so this function is non-negligible□

We look at λ→∞ and notice that → -∞. This makes the final equation always 0, thus making our given function negligible□

|  |
| --- |
| Query():  S ← {0,1}λ |
| return H(s) |

|  |
| --- |
| Query(): |
| S ← {0,1}λ |
| x := G(S)  Y := G(0λ) |
| Return |

1. We take that we know G is a secure PRG.

≡

|  |
| --- |
| Query(): |
| S ← {0,1}λ |
| x := G(S) |
| Y := G(0λ) |
| Return |

|  |
| --- |
| Query(): |
| x := Query’() |
| Y := G(0λ)  Return |

|  |
| --- |
| LGPRG-Real |
| Query’():  S ← {0,1}λ |
| return G(S) |

For the first movement we fill in details of H(s).

≡ ◊

|  |
| --- |
| Query(): |
| x := Query’() |
| Y := G(0λ)  Return |

|  |
| --- |
| LGPRG-Real |
| Query’():  S ← {0,1}λ |
| return G(S) |

Factor out the terms of LGPRG-Real

|  |
| --- |
| LGPRG-Rand |
| Query’():  z ← {0,1}2λ |
| return z |

◊ ≡

|  |
| --- |
| Query(): |
| x := Query’() |
| Y := G(0λ)  Return |

|  |
| --- |
| Query(): |
| x ← {0,1}2λ |
| Y := G(0λ)  Return |

Since G is a secure PRG we can replace LGPRG-Real with LGPRG-Rand

|  |
| --- |
| LGPRG-Rand |
| Query’():  z ← {0,1}2λ |
| return z |

◊ ≡

Now we inline our Query’

|  |
| --- |
| Query(): |
| x ← {0,1}2λ “k” |
| Y := G(0λ) “m”  Return |

|  |
| --- |
| Query(): |
| c ← {0,1}2λ |
| Return c |

We can’t do the same thing to y since it doesn’t pass a parameter that is uniformly chosen. Although it now resembles OTP security of 2λ-bits, where x is our uniformly chosen 2λ-bit code and y will be our plaintext of 2λ-bits, since G(0λ) returns 2λ-bits. Thus we can say

≡

This finishes our proof of security saying that

|  |
| --- |
| Query(): |
| S ← {0,1}λ |
| x := G(S)  Y := G(0λ) |
| Return |

|  |
| --- |
| Query(): |
| c ← {0,1}2λ |
| Return c |

≡

LH(s)­PRG-Real­ LH(s)­PRG-Rand

1. Assuming LH(s)­PRG-Real­ and LH(s)­PRG-Rand look like the libraries below, respectively,

|  |
| --- |
| H(s): |
| X := G(s) |
| Return s||x |

|  |
| --- |
| H(s): |
| c ← {0,1}3λ |
| Return c |

All we need to break which of these libraries we are in is to pass 0λ through H and you should expect a string starting with λ 0’s and then the rest of the 3λ-bit string. If you don’t then you know you are in the RAND library. With our input there is a for REAL and for the RAND

So the adversary just has to compare the first λ-bits of c, and return s == cFirst λ.

1. How we can break the function F’. If we input a string so x’ is the string of all zeros then the real library will always have an output of F(k,x)||F(k,x). Where the random library would just output any random 2n bit string. So REAL has probability of , and where Rand would have a probability of . Which are different so F’ is insecure.

So the Adversary would call the function with x as any n-bit string and x’ be the n-bit string of 0’s. Then have it return whether c == x||x.