Ryan Young

CSE 3400

2/7/2020

Homework #2

1.) *Let* $G:\{0,1\}n \rightarrow \{0,1\}^{n+1}$ *be a secure PRG. Is* $G'(r\|s) = r\|(r \oplus G(s))$, *where* $r \in \{0,1\}^{n+1}$ *and* $s \in \{0,1\}^n$*, also a secure PRG? A formal proof is not needed. A few sentences of explanation should suffice. Hint: Look at the lecture notes for a similar example, however there it is presented as a formal proof.*

Yes, it will absolutely be secure. Since we know that PRG G is secure and we are XOR'ing that secure function with a random string, we know that the product will also be secure. Then finally we are concatenating the secure product of that XOR with that random string thus making the whole PRG G' secure.

2.) *Let F', F'':{0,1}*×D→R be two polynomial-time computable functions, and letFk1,k2(x) =F'k1(F''k2(x)). Assume F' is a secure PRF. Present a choice for F'' such that the resulting F is not a secure PRF. Hint: You have complete control over how you define F''.*

F" will be an unsecure pseudo random function with the same key as F'. So, when F" is broken the attacker will now have the key to F'. So now that we have the input to F' (this is the input of F" and the key to F") and the key to F' the function will now be unsecure.

3.) *Let f be a secure PRP from n-bit strings to n-bit strings. Show that* $g_{kL,kR}(m_L\|m_R) = f_{kL}(m_L)\|f_{kR}(m_R)$ *is not a secure PRP (over 2n-bit strings). That is, present code for an attacker to 'win' the PRP-distinguishability game against g.*

The attacker can input two queries, one with 0's^{2n} and one with $0^n||1^n$. From the output of the PRP the attacker can now tell whether or not it was the PRP g or it was a random permutation.

4.) *Explain why block ciphers fail the IND-CPA test. That is, present code for an attacker that can 'win; the IND-CPA game, thus showing that a block cipher is not IND-CPA secure form of encryption.*

Block ciphers fail the IND-CPA test. This is because a block cipher will output the same ciphertext when given a plain text string at different times. That is to say if an attacker inputs "Hello" into the block cipher two different times it will output the same ciphertext twice. The attacker could also input a string of all 0's and a string of random bits (0's and 1's) and see if there are any clear differences between the inputs.

5.) *Table 2.4 in the lecture notes specifies the encryption process for all the mode of operation. Provide the decryption process for each mode and show the correctness is satisfied. That is show* $m = D_k(E_k(m))$ *for every* $k, m \in \{0,1\}^*$.

For all encryption I will be using an xor for simplicity, my message will be 101 and the key will be 110.

ECB: $C_I = E_K(M_I)$ so, $M_I = D_K(C_I)$ if the encryption method is xoring the message with the key then this will produce 011 for a cipher text. The for decryption, xor the cipher text with the key will product 101 ie the same message as before encryption.

PBC: $C_I = M_I \oplus E_K(I)$ so, $M_I = C_I \oplus D_K(E_K(I))$ again here if the encryption method is xoring the input with the key then 001 xor 110 = 111. This is the xor'd with the message giving us 111 xor 101 = 010. To decrypt we would do this operation in return so 010 xor with 101 = 111. And then 111 xor with 001 gives us 110.
PBR: $C_I = R_I \,||\, M_I \oplus E_K(R_I)$ so, $M_I = D_K(E_K(R_I))\ R_I \oplus C_I$ (2nd half of the bits)
OFB: $r_0 \$\leftarrow\{0,1\}^n$, $r_i = E_K(r_i-1)$, $c_0 \leftarrow r_0$, $ci \leftarrow r_i \oplus m_{i.}$ $M_I = C_1 \oplus E_K(C_{I-1})$ To, decrypt, we used the known IV, key and encryption function. We $\oplus$ the ciphertext block 1 (message block 1 $\oplus$ with encryption (IV)) with the encryption of (IV). Then keep incrementing in the same way we do for encryption.
CFB: $c_0 \$\leftarrow\{0,1\}^n$, $C_I \leftarrow M_I \oplus E_K(c_I-1)$, $M_I = C_I \oplus P_I = C_I \oplus E_K(C_I-1)$
CBC: $C_0 \$\leftarrow\{0,1\}^n$, $C_I \leftarrow E_K(M_I \oplus C_I-1)$, CBCE k (m;IV) = c[0]||c[1]||...||c[n]|c[0]] = IV, (i > 0) c[i] = Ek(c[i−1]$\oplus$m[i]

6.) *Hackme Inc. proposed the following highly efficient block cipher process using two 64-bit keys k_1, k_2, for 64-bit blocks:*

$$EK_1,k_2(m) = (m \oplus k_1) + k_2(\mathrm{mod}2^{64})$$

a.) *Show that $E_{k1,k2}$ is an invertible permutation by presenting the inverse permutation $D_{k1,k2}$.*

$EK_1,k_2(m) = (m \oplus k_1) + k_2(\mathrm{mod}2^{64})$ so, $DK_1,K_2 = (C - k_2(\mathrm{mod}2^{64})) \oplus k_1$

b.) *Show that (E, D) satisfies the correctness property.*

This satisfies the correctness property. For example, take the k_1 = 010 and take m = 111, when xor'd together we would get 101. Now we would add whatever is in $k_2(\mathrm{mod}2^{64})$ but in this case those number do not matter because we can simply subtract them from the cipher text. So, the ciphertext in this example would be 101 + $k_2(\mathrm{mod}2^{64})$. To decrypt we would xor the ciphertext - $k_2(\mathrm{mod}2^{64})$ (101) with the k_1 key (010) and the message would be 111.