HomeWork2

Jonnhy

September 2024

Problem 1: PRF Constructions

Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a secure pseudorandom function (PRF). We are asked to determine if the following constructions are secure PRFs. In all cases, k is a long random secret key.

1. $F'_k(x) = F_k(\overline{x}) \| F_k(x)$

Analysis: In this construction, $F'_k(x)$ concatenates the PRF applied to the bitwise negation $\overline{x}$ with the PRF applied to x. While $F_k(x)$ and $F_k(\overline{x})$ are individually secure as PRFs, the deterministic relationship between x and $\overline{x}$ creates a vulnerability.

Since $\overline{x}$ is deterministically related to x, an attacker could exploit this relationship. By observing $F'_k(x)$, which includes both $F_k(x)$ and $F_k(\overline{x})$, the attacker can distinguish this construction from a random function, making the construction insecure.

Conclusion: *Not secure.* The deterministic relationship between x and $\overline{x}$ allows an attacker to break the pseudorandomness of the function by observing correlated outputs.

2. $F''_k(x) = F_{k_1}(x) \oplus F_{k_2}(x) \| x$

Analysis: This construction XORs the outputs of two PRFs, $F_{k_1}(x)$ and $F_{k_2}(x)$, and concatenates the result with x. The XOR of two independent PRFs should still behave like a pseudorandom function, provided that k_1 and k_2 are independently chosen.

However, the inclusion of x as part of the output breaks the pseudorandomness. An attacker can trivially distinguish this construction from a random function because the output contains the input x, which should not happen in a truly random function.

Conclusion: *Not secure.* The inclusion of x in the output allows an attacker to easily distinguish this construction from a random function.

3. $F'''_k(x) = lsb(F_{k_1}(x)) \| F_{k_2}(x)$

Analysis: In this construction, the least significant bit (LSB) of $F_{k_1}(x)$ is concatenated with the output of $F_{k_2}(x)$. While $F_{k_2}(x)$ is still a secure PRF, truncating $F_{k_1}(x)$ to only its LSB could lead to a loss of entropy and security.

The LSB may not be sufficiently random, and it could potentially introduce detectable patterns that an attacker could exploit. Extracting only a single bit from the PRF output reduces the security of the construction, making it distinguishable from a PRF that outputs full-length pseudorandom values.

Decryption Algorithm: Given the ciphertext $(RH(y), c)$, decrypt by computing:

$$m = G(RH(y)) \oplus c$$

since $c = G(RH(y)) \oplus m$.

Security Against CPA: This scheme uses a secure PRF F to generate the value $y = F_k(r)$ and a secure PRG G to produce the keystream $G(RH(y))$ for one-time pad encryption. As long as r is chosen uniformly at random, the PRF and PRG ensure that the keystream is unpredictable and independent of the message, providing CPA security. The attacker advantage in distinguishing the ciphertext from random is negligible due to the security of the PRG and PRF.

Conclusion: *Secure.* The construction provides CPA security as the keystream used in the encryption is pseudorandom and independent of the message.

2. Encryption Scheme 2:

Given message $m \in \{0,1\}^n$, choose a random string $r \in \{0,1\}^n$ and encrypt m as follows:

$$E_k(m) = (r, F_k(F_k(r)) \oplus m)$$

Decryption Algorithm: Given the ciphertext (r, c), decrypt by computing:

$$m = F_k(F_k(r)) \oplus c$$

since $c = F_k(F_k(r)) \oplus m$.

Security Against CPA: The scheme encrypts m using a pseudorandom keystream $F_k(F_k(r))$. The randomness of r ensures that for each encryption, a fresh and independent pseudorandom value is used for the keystream, making the ciphertext indistinguishable from random to a CPA attacker. Since F is a secure PRF, the double application of F on r does not reduce the security, and the attacker cannot gain any advantage in distinguishing the ciphertext.

Conclusion: *Secure.* The use of a random r and the security of the PRF ensure that the scheme is CPA-secure.

3. Encryption Scheme 3:

Given message $m \in \{0,1\}^{3n}$, parse m as $m = m_1 \| m_2 \| m_3$ where $|m_1| = |m_2| = |m_3| = n$. Choose random strings $r \in \{0,1\}^n$ and $r' \in \{0,1\}^n$, and encrypt m as follows:

$$E_k(m) = (r, r', F_k(1^n) \oplus m_1, F_k(r) \oplus m_2, F_k(r') \oplus m_3)$$

Decryption Algorithm: Given the ciphertext (r, r', c_1, c_2, c_3), decrypt by computing:

$$m_1 = F_k(1^n) \oplus c_1, \quad m_2 = F_k(r) \oplus c_2, \quad m_3 = F_k(r') \oplus c_3$$

since $c_1 = F_k(1^n) \oplus m_1$, $c_2 = F_k(r) \oplus m_2$, and $c_3 = F_k(r') \oplus m_3$.

Security Against CPA: The encryption of each block (m_1, m_2, and m_3) uses a fresh pseudorandom keystream derived from the PRF F_k. The randomness of r and r' ensures that different keystreams are used for each message block, making the ciphertext indistinguishable from random for a CPA attacker. Since F_k is a secure PRF, the attacker cannot distinguish the keystreams from random values.

Conclusion: *Secure.* The construction ensures CPA security due to the use of fresh pseudorandom values for each block of the message, and the randomness of r and r'.

Problem 3: OTP and Feistel Network

1. Alice's Claim on OTP

Alice claims that the One-Time Pad (OTP) is a deterministic encryption scheme and, as such, cannot be secure against a chosen plaintext attack (CPA) because there is no randomness generation in OTP.

Analysis: Alice's claim is *incorrect.* OTP is actually **not** a deterministic encryption scheme. The security of OTP relies on the randomness of the key. For each message m, the key k is chosen uniformly at random and is as long as the message itself. The ciphertext $c = m \oplus k$ is different for every new randomly chosen key. Thus, OTP is perfectly secure as long as the key is used only once.

In the context of CPA security, OTP *does* provide security because the ciphertext does not reveal any information about the plaintext, provided that the key is random and used only once. However, if the key is reused (deterministically), the encryption scheme becomes insecure. The OTP scheme is secure against CPA, as long as the key is used only once and is truly random.

Conclusion: Alice's claim is false. OTP is not deterministic and is secure against CPA, given that the key is random, unique, and used only once.

2. Decrypting the Feistel Network

The Feistel network shown in Slide 10 of Lecture 4 defines the encryption process using a Feistel structure. Given the input $g_k(m)$ (which is the ciphertext) and assuming the same network structure, we can decrypt and recover the original message m.

A Feistel network is invertible by nature because of the way the rounds are structured. Given an input (L_0, R_0), the encryption process for n rounds is:

$$L_{i+1} = R_i, \quad R_{i+1} = L_i \oplus F_k(R_i)$$

for $i = 0, 1, \ldots, n-1$.

Decryption Process: To decrypt and recover the original message m, we use the same Feistel structure but reverse the process:

$$R_i = L_{i+1}, \quad L_i = R_{i+1} \oplus F_k(L_{i+1})$$

Starting with the final values (L_n, R_n), we can iteratively apply the reverse operations of the Feistel network to recover (L_0, R_0), which is the original input message m.

Thus, the Feistel network is invertible, and we can decrypt the ciphertext and recover the original message using the same key and reversing the round functions.

Conclusion: Yes, we can decrypt and recover m using the same Feistel network structure by reversing the encryption process.