Lecture 15: Encrypting Long Messages
Recall

- Suppose \( f : \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n} \) is a one-way permutation (OWP)
- Then, we had see that the function
  \[ G : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^{2n+1} \]
  defined by
  \[ G(r, x) = (r, f(x), \langle r, x \rangle) \]
  is a one-bit extension PRG

- Let us represent \( f^i(x) \) as a short-hand for \( f(\cdots f(f(x))\cdots) \).
  \( f^0(x) \) shall represent \( x \).

- By iterating the construction, we observed that we can create a stream of pseudorandom bits by computing
  \[ b_i(r, x) = \langle r, f^i(x) \rangle \]
  (Note that, if we already have \( f^i(x) \) stored, then we can efficiently compute \( f^{i+1}(x) \) from it)

- So, the idea is to encrypt long messages where the \( i \)-th bit of the message is masked with the bit \( b_i(r, x) \)
Without loss of generality, we assume that our objective is to encrypt a stream of bits \((m_0, m_1, \ldots)\).

**Gen()**: Return \(sk = (r, x) \leftarrow \{0, 1\}^{2^n}\), where \(r, x \in \{0, 1\}^n\).

Alice and Bob, respectively, shall store their state variables: state\(_A\) and state\(_B\). Initially, we have state\(_A = state_B = x\).

**Enc\(_{sk, state_A}\)(\(m_i\))**: \(c_i = m_i \oplus \langle r, state_A \rangle\), and update state\(_A = f(state_A)\), where \(sk = (r, x)\).

**Dec\(_{sk, state_B}\)(\(\tilde{c}_i\))**: \(\tilde{m}_i = \tilde{c}_i \oplus \langle r, state_B \rangle\), and update state\(_B = f(state_B)\), where \(sk = (r, x)\).

Note that the \(i\)-th bit is encrypted with \(b_i(r, x)\) and is also decrypted with \(b_i(r, x)\). So, the correctness holds. This correctness guarantee holds as long as the order of the encryptions and the decryptions remain identical.

Note that each bit \(b_i(r, x)\) is uniform and independent of all previous bits (for computationally bounded adversaries). So, the scheme is secure against all computationally bounded adversaries.