Module Title: Introduction to Modern Cryptography
Exam Diet (Dec/April/Aug): May 2023
Brief notes on answers:

1. (a) (Shift cipher) By applying shift cipher, the relative shift between characters will still be preserved. Because the latter two options of plaintext have the same relative shift, by observing the ciphertext, the attacker can only choose the plaintext correctly with probability $\frac{1}{2}$ when the plaintext is either *vionlu* or *pcihfo*. When the plaintext is *actrpg*, the adversary can easily obtain it by observing the ciphertext.

   (b) (PRF and PRG)

   (i). F' is a pseudorandom function. We prove this by showing that if the adversary can distinguish F', then he can also distinguish F, or distinguish G. Suppose adversary $\mathcal{A}$ can distinguish F' with non-negligible probability, i.e.,

$$\left| \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F'(k,\cdot)}(1^n)=1] - \Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n)=1] \right| > \mathsf{negl}(n),$$

   We let D_1 simulate the experiment of F' for $\mathcal{A}$:

   - $\mathcal{A}(1^n)$ outputs x with length n.
   - Upon receiving x from $\mathcal{A}$, hand x to the challenger and parse y as $y_1 \dots y_{2n}$ (y is either sampled from the evaluation of G, or from U_{2n})
   - D_1 computes $F'(y_1 \dots y_n, x)$, and sends it to $\mathcal{A}$; D_1 outputs 1 iff. $\mathcal{A}$ outputs 1

   Let D_1^G denote the output of D_1 when the values D_1 receives are computed using a PRG G. We have $\Pr[D_1^G = 1] = \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F'(k,\cdot)}(1^n) = 1]$. Let $D_1^{U_{2n}}$ be the output of the distinguisher D_1 when the values it receives are sampled from U_{2n}, we have $\Pr[D_1^{U_{2n}} = 1] = \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F(k,\cdot)}(1^n) = 1]$ because now F' behaves exactly as F. Because G is a PRG, for any PPT distinguisher D_1, we have $|\Pr[D_1^G = 1] - \Pr[D_1^{U_{2n}} = 1]| \leq \mathsf{negl}(n)$, and therefore

$$\left| \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F'(k,\cdot)}(1^n)=1] - \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F(k,\cdot)}(1^n)=1] \right| \leq \mathsf{negl}(n). \quad (1)$$

   Since F is a secure PRF we learn that

$$\left| \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F(k,\cdot)}(1^n)=1] - \Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n)=1] \right| \leq \mathsf{negl}(n), \quad (2)$$

   Combining (1) and (2) together we get

$$\left| \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F'(k,\cdot)}(1^n)=1] - \Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n)=1] \right| \leq \mathsf{negl}(n),$$

   This contradicts our assumption, which means the $\mathcal{A}$ does not exist, hence F' is a pseudorandom function.

   (ii). F'' is not a pseudorandom function. To see this, suppose F^* is a pseudorandom function, we construct the following function F: when the key $k = 0^n$, output 0, otherwise it works the same as F^*. In this case, because $\Pr[k = 0^n] = \frac{1}{2^n}$, F is still a pseudorandom function. However, with the pseudorandom function F we defined, we formally define the following attacker $\mathcal{A}$ given 1^n and access to some function g:
   $\underline{\mathcal{A}^g(1^n):}$

- Query 0^n and receive $y = F''(k, 0^n)$.
- Output 1 if and only if $y = 0$.

As shown above, we have $\Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F''(k,\cdot)}(1^n) = 1] = 1$. But when g is a random function then y is independent, uniform string of length n, and so the probability that they are the same is exactly 2^{-n}. Thus, $\Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n) = 1] = 2^{-n}$, and the difference

$$\left| \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F''(k,\cdot)}(1^n) = 1] - \Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n) = 1] \right| = 1 - 2^{-n}$$

is not negligible.

(iii). F''' is not a pseudorandom function. To see this, note that the output of F''' is always 1^n. This is because, for i-th bit in $F(k, x)$, if it is 1 (0, resp.) then the i-th bit in $\overline{F(k, x)}$ is 0 (1, resp.), and $1 \oplus 0 = 0 \oplus 1 = 1$. Formally, we define the following attacker $\mathcal{A}$ given 1^n and access to some function g:

$\underline{\mathcal{A}^g(1^n):}$

- Query any input x and receive $y = F'''(k, x)$.
- Output 1 if and only if $y = 1^n$.

As shown above, we have $\Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F'''(k,\cdot)}(1^n) = 1] = 1$. But when g is a random function then y is independent, uniform string of length n, and so the probability that they are the same is exactly 2^{-n}. Thus, $\Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n) = 1] = 2^{-n}$, and the difference

$$\left| \Pr_{k\leftarrow\{0,1\}^n}[\mathcal{A}^{F'''(k,\cdot)}(1^n) = 1] - \Pr_{f\leftarrow\mathsf{Func}}[\mathcal{A}^{f(\cdot)}(1^n) = 1] \right| = 1 - 2^{-n}$$

is not negligible.

(c) (MAC) Π is not a strong MAC. Now consider a secure MAC scheme $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$ and a new scheme $\Pi' = (\mathsf{Gen}', \mathsf{Mac}', \mathsf{Vrfy}')$ that works as follows:

$$\mathsf{Gen}'(\kappa) : k \xleftarrow{\$} \mathsf{Gen}(\kappa).$$
$$\mathsf{Mac}'(k, m) : R \xleftarrow{\$} \{0,1\}^n, t \leftarrow \mathsf{Mac}_k(m) \,\|\, R.$$
$$\mathsf{Vrfy}'(k, m, t) : t = (t', R), \text{ output 1 if } t' = \mathsf{Mac}_k(m) \text{ and 0 otherwise.}$$

Firstly, we need to prove Π' is secure under Mac-forge. We prove it through a reduction to the security of Π. Assuming the existence of the adversary $\mathcal{A}$ that can forge the MAC in the experiment Mac-forge for Π', we can construct the following adversary $\mathcal{A}'$:

- Query the MAC oracle with message m, receive back the tag t, sample a randomness R from uniform distribution, compute $t' = t \,\|\, R$, and send it to $\mathcal{A}$
- Output (m, t) received from $\mathcal{A}$ after querying the oracle

Then we can prove that Π' is not secure under Mac-strongForge. We can construct the following adversary $\mathcal{A}_{\texttt{strongForge}}$:

- Query the MAC oracle with message m, receive back the tag t.

- Parse t as $t_0 \| R$ sample a randomness R' from uniform distribution, compute $t' = t_0 \parallel R'$
- Output (m, t')

In this case, $\mathcal{A}_{\texttt{strongForge}}$ outputs (m, t) such that $\mathsf{Vrfy}_k(m, t) = 1$ and $(m, t) \notin \mathcal{M}$. It means Π' is not secure under Mac-strongForge.

2. (a) (Collision Resistance)

   (i). H' is collision resistant. Suppose towards a contradiction, H' is not collision resistant. I.e., it is feasible to find $x \neq x'$ such that $H_1(x) \parallel H_2(x) = H_1(x') \parallel H_2(x')$. This implies a collision in both H_1, H_2, contradicting the fact that either one of them is collision resistant.

   (ii). H'' is not collision resistant. Consider x, x' such that they are different only in the last bit. I.e., $x_i = x'_i$ for all $1 \leq i \leq n-1$. We have $H_1(x_1 x_2 \dots x_{n-1} 0) = H_1(x'_1 x'_2 \dots x'_{n-1} 0)$ and $H_2(x_1 x_2 \dots x_{n-1} 1) = H_2(x'_1 x'_2 \dots x'_{n-1} 1)$ thus $H''(x) = H''(x')$.

3. (a) (El Gamal encryption) We prove the security of the new scheme through a reduction to the security of the El Gamal encryption. Formally, assuming the existence of the adversary $\mathcal{A}$ that breaks the security of our new scheme with non-negligible probability, then we can construct the following adversary $\mathcal{A}'$ that contradicts the security of the El Gamal encryption scheme:

   - Use $\mathcal{A}$ to generate 2 plaintexts (m_0, m_1), and send it to the challenger.
   - Receive $(ct_1 = pk^{r_1} \cdot m_b, ct_2 = g^{r_1})$ from the challenger, sample r_2 from $\mathcal{Z}_p$, and compute $ct'_1 = ct_1 \cdot pk^{r_2}, ct'_2 = ct_2, ct_3 = g^{r_2}$
   - Send (ct_1, ct_2, ct_3) to $\mathcal{A}$, and output whatever $\mathcal{A}$ outputs.

   In this case, because $\Pr[b = b']$ is 1/2+non-negligible, we construct an adversary that can win the security game of El Gamal encryption with non-negligible advantage, which contradicts the security of the El Gamal encryption.

   (b) (One-time-secure signature) This scheme is not two-time-secure. The adversary can query two messages $m_0 = 000\dots0$ and $m_1 = 111\dots1$. From the first message $\mathcal{A}$ learns s_i^0 for all $1 \leq i \leq n$; similarly, from the second message $\mathcal{A}$ learns s_i^1 for all $1 \leq i \leq n$. I.e., $\mathcal{A}$ knows the entire SK. Then, for any message m other than m_0, m_1, the adversary can generate $\sigma(m)$ by indexing each bit to SK.