Homework 6: Public key, RSA/Dlog and lattices

Total of 120 points

1. (KL Ex 8.10, 15 points) Prove that for every \( x \in \{0, \ldots, m-1\} \) (even if \( x \) is not in \( \mathbb{Z}_m^* \)) if \( ed = 1 \pmod{\varphi(m)} \) then \((x^e)^d = x \pmod{m}\).

2. (KL Ex 8.20, 25 points) Let \( m,e \) be as in the RSA problem, let \( y \in \mathbb{Z}_m^* \), and let \( f_0 \) be the RSA function \( f_0(x) = x^e \) and \( f_1 \) be its “shifted by \( y \)” variant \( f_1(x) = y \cdot x^e \).
   
   a. (10 points) Prove that given two inputs \( x \neq x' \in \mathbb{Z}_m^* \) such that \( f_0(x) = f_1(x') \), one can find \( y^{1/e} \pmod{m} \).
   
   b. (15 points) Conclude that \( \ell = 10 \log m \), if we pick \( m,e,y \) as above and let \( H_{m,e,y}(z_1, \ldots, z_\ell) \) be defined as \( f_{z_1}(f_{z_2}(\cdots(f_{z_\ell}(1))\cdots)) \) then this collection is a collision resistant hash family \( (0,1)^\ell \) to \( \mathbb{Z}_m^* \) if the RSA function is hard to invert. That is, if there is an algorithm that given a random hash function \( H \) from this collection finds \( z \neq z' \in \{0,1\}^\ell \) such that \( H(z) = H(z') \) then there is an algorithm to invert the RSA function.

3. (One time signatures, 25 points) As I mentioned it is in fact possible to get digital signatures based on only private key cryptography. In this exercise we will show a baby version of this. We say that a signature scheme \((G,S,V)\) is a one time signature scheme if it satisfies the security definition of digital signatures (with a public verification key) with the restriction that the adversary is only allowed to make a single query \( m \) to the signing oracle, and needs to output a signature on a message \( m' \neq m \).

Let \( PRG : \{0,1\}^n \rightarrow \{0,1\}^{2n} \) be a pseudorandom generator. Prove that the following scheme is a secure one-time signature scheme for messages of length \( \ell \):

- **Key generation:** Pick \( 2\ell \) independent random strings in \( \{0,1\}^n \) which we’ll denote by \( x_1^0, \ldots, x_\ell^0, x_1^1, \ldots, x_\ell^1 \). The secret signing key is the tuple \((x_b^i)_{b \in \{0,1\}, i \in [\ell]} \) while the public verification key is the tuple \((y_b^i)_{b \in \{0,1\}, i \in [\ell]} \) where \( y_b^i = PRG(x_b^i) \)

- **Signing:** To sign a message \( m \in \{0,1\}^\ell \), output the \( \ell \)-tuple \((x_1^m, \ldots, x_\ell^m) \).
• **Verification:** To verify a message \( m \) w.r.t. signature \( (x'_1, \ldots, x'_t) \) and public key \( (y'_i)_{i \in \{0,1\}, i \in [t]} \), check that \( PRG(x'_i) = y'_i \) for all \( i \in [t] \).

4. (30 points) Consider the following variant of the DSA signature scheme:

• **Key generation:** Let \( G \) be a cyclic group. Pick generator \( g \) for \( G \) and \( a \in \{0, \ldots, |G| - 1\} \) and let \( h = g^a \). Pick \( H : \{0, 1\}^t \times \{0, \ldots, |G| - 1\} \rightarrow \{0, \ldots, |G| - 1\} \) and \( F : G \rightarrow \{0, \ldots, |G| - 1\} \) to be some functions that we consider as random oracles. The public key is \( (g, h) \) (as well as the functions \( H, F \)) and secret key is \( a \).

• **Signature:** To sign a message \( m \), pick \( b \) at random, let \( f = g^b \), let \( c = F(f) \) and \( d = H(m, c) \) and then let \( s = b^{-1}[d + a \cdot c] \) where all computation is done modulo \( |G| \). The signature is \( (f, s) \).

• **Verification:** To verify a signature \( (f, s) \) on a message \( m \), compute \( c = F(f) \) and \( d = H(m, c) \) and then check that \( s \neq 0 \) and \( f^s = g^d h^c \).

a. (20 points) Prove that this is a secure one-time signature scheme in the random oracle model, assuming the difficulty of the discrete logarithm problem in \( G \). See footnote for hint

b. (10 points) Prove that this is a secure (many times) signature scheme in the random oracle model, assuming the difficulty of the discrete logarithm problem in \( G \).

5. (25 points) Prove that under the LWE assumption, the following variant of our lattice based encryption scheme is secure: (you can use the assumption of security of the scheme presented in class if it helps.)

• **Parameters:** Let \( \delta(n) = 1/n^4 \) and let \( q = \text{poly}(n) \) be a prime such that LWE holds w.r.t. \( q, \delta \). We let \( m = n^2 \log q \). *(Same as before)*

• **Key generation:** Pick \( x \in \mathbb{Z}_q^n \). The private key is \( x \) and the public key is \( (A, y) \) with \( y = Ax + e \) with \( e \) a \( \delta \)-noise vector and \( A \) a random \( m \times n \) matrix. *(Same as before)*

• **Encrypt:** To encrypt \( b \in \{0, 1\} \) given the key \( (A, y) \), pick \( w \in \{0, 1\}^m \) and output \( 2w^\top A, 2(w, y) + b \) (all modulo \( q \) of course). The difference is that instead of adding either 0 or 2, we add either 0 or 1, but multiply this by 2 so the result would be even or odd as needed.

• **Decrypt:** To decrypt \( (a, \sigma) \), output 0 iff \( |\langle a, x \rangle - \sigma| \) is even. (Instead of asking this to be smaller than \( q/10 \).)

---

1You need to design a reduction that takes \( h = g^a \) and returns \( a \) using “in its belly” an adversary for the signature scheme. You can use \( h \) as the public key. The scheme ensures that to produce a valid signature the adversary will first need to ask \( F \) on the query \( f \), and then ask \( H \) on the query \( m, F(f) \). The idea is that once an adversary makes a query \( f \) to the oracle \( F \), then they have “committed” to the value \( b \) such that \( g^b = f \) even if they didn’t disclose it. Now, if they are able to successfully sign the message \( m \) with decent probability over the output of \( H(m, c) \) then we’ll be able to find two different responses \( d \neq d' \) for which they can sign successfully. This will yield two linearly independent equations on the two unknowns \( b \) and \( a \).