Fundamentals of Cryptography: Final

Wednesday Jan 10, 2-4PM

Problem 1 (1pt) p is a *safe prime* if ____ fill the blank ____.

Problem 2 (4pt) State, to the best of your knowledge, the relations between the following cryptographic assumptions. Draw an arrow from assumption A to assumption B if assumption A implies assumption B. Note that the relation is transitive, so if you draw an arrow from A to B and an arrow from B to C, there is no need to draw a third arrow from A to C.

- The existence of OWFs.
- The existence of OWPs.
- The existence of constant-round key exchange protocols.
- The existence of (CPA-secure) public-key encryption schemes.
- The existence of digital signature schemes.

Problem 3 (2pt) State the difference between zero-knowledge proof (ZKP) and zero-knowledge proof of knowledge (ZKPoK). In either proof system, we assume both the prover and the verifier are efficient, the honest prover is given the instance and a witness. Compared with ZKP, ZKPoK satisfies an additional property: ____ fill the blank ____.

Problem 4 (3pt) Garbled Circuits You should state how to garble a boolean circuit. The solution is not unique.

Given the circuit C, for each wire $i \in [n]$, the garbling algorithm generates two random $L_{i,0}, L_{i,1}$ as follows: ____ fill the blank ____. Output $L_{1,0}, L_{1,1}, \ldots, L_{n_{\text{in}},0}, L_{n_{\text{in}},1}$ as the input labels. And output the garbled circuit $\tilde{C}$ as follows

- For each $i \in \{n_{\text{in}}+1, \ldots, n\}$, generate and output a table as follows: fill the blank. Say the gate function is $g : \{0,1\} \times \{0,1\} \to \{0,1\}$, and the gate takes wires j_1, j_2 as inputs.
- For each output wire $i \in \{n - n_{\text{out}} + 1, \ldots, n\}$, output ____ fill the blank ____.

Formalization of a circuit (for problem 4). A circuit has n wires, including n_{in} input wires, n_{out} output wires and $n - n_{\text{in}} - n_{\text{out}}$ intermediate wires. W.l.o.g., the wires are indexed by $1, \ldots, n$. Given the input x, the value of the i-th wire, denoted by v_i, and the output of the circuit are determined as follows. For a boolean circuit, the input is in $\{0,1\}^{n_{\text{in}}}$. For an arithmetic circuit over $\mathcal{R}$, the input is in $\mathcal{R} \in \{0,1\}^{n_{\text{in}}}$.

- For each $i \le n_{\text{in}}$, the i-th wire is the i-th input wire, so $v_i = x_i$.
- For each $i > n_{\text{in}}$, the i-th wire is the output of a gate. Say the gate function is g, and the gate takes wires $j_1, \ldots, j_t$ as inputs ($j_1 \le \cdots \le j_t < i$). Then $v_i = g(v_{j_1}, \ldots, v_{j_t})$.
- The output of the circuit is $(v_{n-n_{\text{out}}+1}, \ldots, v_n)$.

Problem 5 (5pt + bonus) Paillier Encryption Revisit In Pset 9, we analyze the Paillier encryption scheme. The public key is $N = pq$, where p, q are two random distinct λ-bit safe primes. Given the public key, the message space is $\mathbb{Z}_N$. The encryption algorithm is $\mathsf{Enc}(pk = N, m \in \mathbb{Z}_N) \to h \cdot (1+N)^m$, where h is a random "hard subgroup" element.

The security of Paillier encryption is based on the decisional composite residuosity (DCR) assumption, which says a random element in the "hard subgroup" is indistinguishable from a random element in $\mathbb{QR}_{N^2}$ (or $\mathbb{Z}_{N^2}$, depending on how the hard subgroup is defined).

In this problem, we consider a generalization of Paillier encryption. The message space is $\mathbb{Z}_{N^d}$, where d is a given constant.

Part A. State the generalized Paillier encryption scheme. How to encrypt and decrypt?

$\mathsf{Gen}(1^\lambda)$ samples safe primes $p = 2p'+1, q = 2q'+1$, lets $N = pq$.
Output $pk = N$, $sk = p'q'$.

$\mathsf{Enc}(pk, m)$, for $m \in \mathbb{Z}_{N^d}$, fill the blank

$\mathsf{Dec}(sk, c)$, for $c \in \mathbb{Z}_{N^{d+1}}$, fill the blank

Part B. Prove the new public-key encryption scheme is CPA-secure. State the necessary assumption.

Part C (bonus). Let $d = 2$. Is the new assumption stronger or weaker or equivalent to the DCR assumption used by standard Paillier? Prove your statement.

Problem 6 (5pt) Construct an "identity-based signature" scheme $(\mathsf{Gen}, \mathsf{Ext}, \mathsf{Sign}, \mathsf{Verify})$. The syntax is

- The key generation algorithm $\mathsf{Gen}(1^\lambda)$ samples a public key pk and a master secret key msk.
- Given the master key and an ID, the key extraction algorithm $\mathsf{Ext}(msk, ID)$ returns an ID-associated key sk_{ID}.
- Given an ID, an associated key, and a message, the signing algorithm $\mathsf{Sign}(ID, sk_{ID}, m)$ outputs a signature σ.
- Given the public key, an ID, a message, and a signature, the verification algorithm $\mathsf{Verify}(pk, ID, m, \sigma)$ accepts or rejects.

The scheme is secure, if no p.p.t. adversary can win the following unforgeability game with non-negligible probability.

- The challenger samples $(pk, msk) \leftarrow \mathsf{Gen}(1^\lambda)$, and sends pk to the adversary.
- The adversary are allowed to make the following two types of queries
  - The adversary chooses an ID, and receives the associated key sk_{ID}.
  - The adversary chooses an ID and a message, and receives the corresponding signature.

- After polynomially many queries, the adversary outputs $(\hat{ID}, \hat{m}, \hat{\sigma})$.
- The adversary wins if 1) the adversary never queried the key associated with $\hat{ID}$, 2) the adversary never queried the signature of message $\hat{m}$ under $\hat{ID}$, and 3) $\mathsf{Verify}(pk, \hat{ID}, \hat{m}, \hat{\sigma})$ accepts.

State your construction, and prove its security. The construction can be based on any cryptographic assumption mentioned in class.

Problem 7 (5pt) Consider the following commitment scheme that commits one-bit messages. The scheme is defined in the CRS model. The common reference string is a random λ-bit safe prime p and a generator $g \in \mathbb{Z}_p^*$. ((p, g) can also be determined by a common random string.)

- To commit a bit $m \in \{0, 1\}$, sample random integers x, y from a sufficiently large domain such that $g^x \neq g^{\pm 1}$ and $g^y \neq 1$, send (g^x, g^y, g^{xy}) as the commitment if $m = 0$, send (g^x, g^{xy}, g^y) as the commitment if $m = 1$.
- Open the commitment by sending (m, x, y).

Part A. Is the commitment scheme (computationally) hiding under proper computational assumption? Explain your answer.

Part B. Is the commitment scheme (statistically) binding? Explain your answer.

Part C. How to fix the problems in part A and/or part B so that the scheme becomes (computationally) hiding and (statistically) binding?

Problem 8 (5pt) In this problem, $\mathbb{F}$ denotes a finite field that may depend on the security parameter.

Oblivious linear function evaluation (OLE) is a special case of 2-party computation. The receiver's input is a value $x \in \mathbb{F}$, the sender's input is $(a, b) \in \mathbb{F}^2$, the receiver's output is $ax + b$. The sender's input can be viewed as a linear function $z \mapsto az + b$, and the receiver's output is the evaluation of the linear function on receiver's input.

OLE can be generalized to "oblivious polynomial evaluation (OPE)". The receiver's input is a value $x \in \mathbb{F}$, the sender's input is $(a_0, a_1, \ldots, a_d) \in \mathbb{F}^{d+1}$ where d is a public constant, the receiver's output is $\sum_{i=0}^{d} a_i x^i$.

Part A. Show that OLE implies OPE in the semi-honest setting. That is, given a semi-honest OLE protocol, construct a semi-honest OPE protocol. Also prove the security of your protocol.

Part B. Show that OLE implies OPE in the malicious setting. For simplicity, we consider full security (the statement holds under other malicious security notation as well). That is, given a (computationally) fully secure OLE protocol, construct a (computationally) fully secure OPE protocol. Explain in high level why your protocol is secure.

You can rely on the fact that fully secure OLE protocol implies fully secure VOLE protocol. VOLE is short for vector OLE, the receiver's input is $x \in \mathbb{F}$, the sender's input consists of two vectors $\mathbf{a}, \mathbf{b}$, the receiver's output is $\mathbf{a}x + \mathbf{b}$.

The construction should be black-box. For example, it should not relied on generic zero-knowledge proof.

2PC security definition (for problem 8). Consider a 2PC problem, two parties are Alice and Bob, their inputs are x and y, their outputs are $f_A(x, y)$ and $f_B(x, y)$.

A 2-party computation protocol is semi-honest secure, if there are p.p.t. simulators $\mathcal{S}_A, \mathcal{S}_B$, such that for any x, y, Alice's view in $\langle \mathcal{A}(x), \mathcal{B}(y) \rangle$ can be simulated by $\mathcal{S}_A(x, f_A(x, y))$, Bob's view can be simulated by $\mathcal{S}_B(y, f_B(x, y))$.

A 2-party computation protocol is fully secure, if it is fully secure when Alice or Bob is maliciously corrupted. We focus on the case if Alice is corrupted, the case of corrupted Bob is symmetric. The protocol is fully secure against maliciously corrupted Alice, if for any p.p.t. environment $\m\mathcal{E}$, for any p.p.t. adversary $\mathcal{A}$, there is a p.p.t. simulator $\mathcal{S}$, such that the following two distributions are indistinguishable:

($\mathcal{E}$'s view, $\mathcal{A}$'s view, Bob's output) in the real world

- $\mathcal{E}$ sends input y to Bob.
- $\mathcal{E}, \mathcal{A}$ interact during the protocol.
- $\mathcal{A}$ and Bob run the protocol, during which $\mathcal{A}$ may arbitrarily deviate from the protocol.

($\mathcal{E}$'s view, $\mathcal{S}$'s output, Bob's output) in the ideal world

- $\mathcal{E}$ sends input y to Bob.
- $\mathcal{E}, \mathcal{A}$ interact during the protocol.
- $\mathcal{A}$ chooses an input x. $\mathcal{A}$ and Bob send x, y respectively to a trusted party. The trusted party sends $f_A(x, y), f_B(x, y)$ to $\mathcal{A}$, Bob respectively.