End Semester - Fall, 2024
M. Tech. Cryptology & Security - 2^{nd} Year
Advanced Cryptology

November 29, 2024

Maximum Marks: 50
Maximum Time: 3 hr
Open note / book exam

Instructions

- Maximum marks: 50; total marks provided in the paper: 65.
- Be short and precise. Partial marking will be provided for a partially correct answer/attempt. For example, for 5 mark question, the answer should ideally be 1/2 to 1 page long, and for 8 mark question it should be 1-2 pages. All parts of a question *must* be answered in the same place.
- You can use any book/notes during the exam, but *not internet*.
- The seating arrangements and other regulations circulated from Dean's office *must* be followed adequately.

Notations. The set of all integers are denoted by $\mathbb{Z}$. The ring of all integers modulo n is denoted by $\mathbb{Z}_n$. Below $\kappa \in \mathbb{N}$ denotes the security parameter throughout (for example, if 128 bit security is desired from the system, then κ is set to 128, this is often equal to the key-length). A uniform random sample from a domain D is denoted as $s \leftarrow_\$ D$. We assume $\langle g \rangle = \mathbb{G}$ to be a cyclic group of prime order p. For bilinear pairing $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, we assume each group $\langle g_i \rangle = \mathbb{G}_i$ is of order p.

1. Recall BLS multi-signatures, for two parties P_1 and P_2. P_i holds a key-pair $(\mathsf{sk}_i, \mathsf{vk}_i)$, where $\mathsf{vk}_i = g_2^{\mathsf{sk}_i}$. For a message m, P_i generates a partial signature $\sigma_i := H(m)^{\mathsf{sk}_i}$. The signatures are aggregated simply by computing the group product $\sigma = \sigma_1\sigma_2 = H(m)^{\mathsf{sk}}$, where $\mathsf{sk} := \mathsf{sk}_1 + \mathsf{sk}_2$. The aggregated signature is then verified by first computing $\mathsf{vk} = \mathsf{vk}_1\mathsf{vk}_2 = g_2^{\mathsf{sk}}$ and then checking $e(H(m), \mathsf{vk}) = e(\sigma, g_2)$.
   (a) Now consider a malicious party, that receives $(\sigma_1, \mathsf{vk}_1)$, computes a rogue key $\mathsf{vk}_2^* := g_2^\delta/\mathsf{vk}_1$, and signature $\sigma_2^* := \mathsf{H}(m)^\delta/\sigma_1$ for some arbitrarily chosen $\delta \in \mathbb{Z}_p$. What happens with the verification in this case? (answer briefly in 2-3 sentences)
   (b) Explain why the above is a problem? (answer briefly in 2-3 sentences)
   (c) Propose a way to fix the problem with arguments.

   (2+2+5)

2. A verifiable random function (VRF) is essentially a PRF with additional capabilities, such that the evaluation function can generate a proof which can be publicly verified with a public key. In particular, a VRF scheme consists of a triple of algorithms $(\mathsf{KGEN}, \mathsf{EVAL}, \mathsf{VER})$ with following syntax:
   - $\mathsf{KGEN}(1^\kappa) \to (\mathsf{sk}, \mathsf{vk})$ the key-generation outputs a secret-public key pair;

- $\mathsf{EVAL}(\mathsf{sk}, x) \to (y, \pi)$ the evaluation procedure uses the secret key to produce a pseudorandom output y and a proof π;
- $\mathsf{VER}(\mathsf{vk}, (x, y, \pi)) \to 1/0$ the verification algorithm checks whether the triple of input, output and proof are correctly generated with respect to a verification key.

(a) Construct a VRF from a BLS signature. Argue why the output is pseudorandom. (hint: use random oracles)

(b) Similarly, can you construct a VRF from any signature by above method? What additional property you need from a signature, so that it can be converted to a VRF?

(c) Like signatures, VRF can also be thresholdized. Describe how your BLS-based construction can be thresholdized.

(5+5+5)

3. (a) Describe a semi-honest secure two party computation for a single boolean AND gate using FHE and provide arguments for security using simulation technique.

   (b) If the FHE is instantiated with GSW scheme, what is the total communication (can be written in number of field/group elements for a specific field/group, e.g. $2n$ elements in $\mathbb{Z}_q$).

(8+8)

4. (a) Consider a two party setting, in that parties P_1 and P_2 would like to compute an addition over a field on their corresponding inputs x_1, x_2. They execute a protocol Π in which party P_1 sends over its input x_1 to P_2 and P_2 sends over x_2 to P_1. Then each party locally computes $y = x_1 + x_2$. Is this protocol secure or insecure in the semi-honest setting? Provide arguments in favor of your answer.

   (b) Now consider a three party setting (party P_i has input x_i for $i \in \{1, 2, 3\}$) where at most one party maybe (semi-honest) corrupt. Does the same protocol work? Explain.

(5+5)

5. Prove the following in bilinear pairing setting:

   (a) Consider Type-2 setting, where $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is an easy isomorphism. Then show that $\forall A, B \in \mathbb{G}_2$, the following holds: $e(\psi(A), B) = e(\psi(B), A)$.

   (b) In Type-2 setting, prove that DDH over $\mathbb{G}_2$ is easy.

   (c) In Type-3 setting, prove that: if BDDH holds then DDH over $\mathbb{G}_T$ also holds.

(5+5+5)