ECE498AC Midterm

Instructor: Andrew Miller

October 15, 2019

Name _____ NetID _____

Instructions

Please either 1) typeset your answers in LaTeX and submit a PDF file on gradescope, 2) write answers by hand and upload a scanned PDF file on gradescope. We prefer to read succinct and precise answers. If you can be precise while being succinct with your answers, please try.

Due date: 11:59pm, Oct 22 2019.

Academic integrity reminder: We treat academic integrity very seriously. You are supposed to do this exam individually. You may refer to lecture notes, optional textbooks, or internet searches. However, you should not talk about the problems with peers or ask for help online.

How multiple choice is graded. Multiple choice questions may have multiple correct answers. If there are N correct answers, then each correct answer is worth $1/N$ of the total points for the question. You lose $1/N$ points for every wrong answer you circle. The total for a question cannot go negative. In code, `score = points * max(num_correct - num_wrong, 0) / N`

Clarity, succinctness, writing your name and Netid: [5 pts].

1 Indistinguishability [6 pts]

1. (3 pts) If $\{X_n\}_n$ is computationally indistinguishable from $\{Y_n\}_n$, and $\{Y_n\}_n$ is computationally indistinguishable from $\{Z_n\}_n$, then (select the one that is always correct)
   (a) $\{X_n\}_n$ is computationally indistinguishable from $\{Z_n\}_n$
   (b) $\{X_n\}_n$ can be efficiently distinguished from $\{Z_n\}_n$
   (c) Sometimes $\{X_n\}_n$ can be efficiently distinguished from $\{Z_n\}_n$, sometimes $\{X_n\}_n$ is computationally indistinguishable from $\{Z_n\}_n$
2. (3 pts) If $\{X_n\}_n$ can be efficiently distinguished from $\{Y_n\}_n$, and $\{Y_n\}_n$ can be efficiently distinguished from $\{Z_n\}_n$, then (select the one that is always correct)
   (a) $\{X_n\}_n$ is computationally indistinguishable from $\{Z_n\}_n$
   (b) $\{X_n\}_n$ can be efficiently distinguished from $\{Z_n\}_n$

(c) Sometimes $\{X_n\}_n$ can be efficiently distinguished from $\{Z_n\}_n$, sometimes $\{X_n\}_n$ is computationally indistinguishable from $\{Z_n\}_n$

2 Useful Asymptotical Notations [6 points]

(No need to prove throughout this question.)

2.1 [2 pts]

Among the following functions in n, please select all that are negligible functions in n:

a. $\frac{1}{2^{\sqrt{n}}}$ b. $\frac{1}{2^n}$ c. $n^{-\log n}$ d. $\frac{\log n}{n+1}$ e. $\frac{1}{n^{\log\log n}}$ f. $\frac{1}{n!}$ g. $\frac{1}{n^2 \log n}$

2.2 [2 pts]

Suppose that $f_1(n), f_2(n)$ are negligible functions in n. Let $g(n)$ denote some fixed positive polynomial in n. Also, $f_1(n), f_2(n), g(n)$ are positive for every $n \geq 1$. Which of the following *must* be negligible functions in n?

a. $f_1(n) + f_2(n)$ b. $f_1(n)g(n)$ c. $g(f_2(n))$ d. $g(n)^{-g(n)}$ e. $f_1(n)^{g(n)}$ f. $\sqrt[3]{f_1(n)}$

2.3 [2 pts]

Let $g_1(n), g_2(n)$ denote two fixed polynomials in n. Which of the following must be polynomial in n:

a. $g_1(n) + g_2(n)$ b. $g_1(n)g_2(n)$ c. $g_1(n)^{g_2(n)}$ d. $g_1(n) + 55$ e. $g_1(n) + 2^n$ f. $g_1(n)^{100}$

3 Pseudorandomness [25 pts]

3.1 Construction Implications [5 pts]

Recall that we discussed a construction for a Universal One Way Function (UOWF) - a concrete function family that is a One Way Function (OWF) if any OWF exists. We also discussed a Universal Hardcore Predicate (UHCP), which provides pseudorandom bits from any one way function. Let $\{\mathcal{G}_\lambda =< g_\lambda >\}_\lambda$ be a family of prime-order groups and generators (each $\mathcal{G}_\lambda$ has order p_λ, where p_λ is a λ-bit prime).

For each of the following statements, judge them as either (T: true/valid, F: false/invalid):

1. If P != NP, then OWFs exist.
2. If OWFs exist, then P != NP.
3. If OWFs exist, then the function family $f_\lambda(x) = g_\lambda^x$ is a OWF.
4. If P != NP, then applying UHCP to the UOWF results in a pseudorandom generator (PRG).
5. If OWFs exist, then applying UHCP to f_λ results in a PRG

3.2 Pseudorandom Functions [5 pts]

Let $F_k(\cdot)$ be any PRF, where key k is uniformly sampled from $\{0,1\}^\lambda$. Show whether or not the following are PRF constructions. To show it is a PRF, you must use a reduction. To show it is not, you must give a counterexample.

1. $F'_k(x) = F_k(x+1) \| F_k(x)$
2. $F'_k(x) = F_x(k)$
3. $F'_k(x) = F_{F_k(0^\lambda)}(x)$

3.3 Pseudorandom Generators [5 pts]

Let $G : \{0,1\}^\lambda \to \{0,1\}^\lambda$ be a PRG. Circle all PRGs below. (No need to prove) (For the purpose of this question, G counts as a PRG even though it is not "expanding")

a. $G'(s) = \text{if } \lambda > 1024 \text{ then } G(s) \text{ else } 0^\lambda$

b. $G'(s_1 \,||\, s_2) = G(s_1) \oplus G(s_2)$, where $|s_1| = |s_2| = \lambda$

c. $G'(s_1 \,||\, s_2) = G(s_1) \vee G(s_2)$, where $|s_1| = |s_2| = \lambda$

d. $G'(s_1 \,||\, s_2) = s_1 \,||\, G(s_2)$, where $|s_1| = |s_2| = \lambda$

e. $G'(s) = s \,||\, G(s)$

3.4 Random Number Generation [5 points]

Given an oracle method O that generates random $\lambda-$bit strings of the form $\{0,1\}^\lambda$, implement a method for generating uniform random **integers** from $[0, 100)$.

3.5 More Pseudorandom Functions [5 pts]

We used a "Special Encryption" scheme for the garbled circuits assignment. Let $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be a length-doubling PRF. The encryption scheme we used for messages $m \in \{0,1\}^\lambda$ was

- $\mathsf{Gen}(1^\lambda) : k \leftarrow \{0,1\}^\lambda$
- $\mathsf{Enc}(k, m)$: Sample $r \xleftarrow{\$} \{0,1\}^\lambda$. Then $C = (r, F(k,r) \oplus (m \| 0^\lambda))$.
- $\mathsf{Dec}(k, C)$: parse C as (r, c'). Compute $m' = F(k,r) \oplus c'$. Parse m' as $m_L \| m_R$. If $m_R = 0^\lambda$, then output m_L, otherwise output $\perp$, where $\perp$ indicates decryption failed due to an invalid ciphertext.

What's "special" about this scheme is that it's unlikely a ciphertext can be decrypted under two independently generated keys. More formally, for an adversary $\mathcal{A}$, define the distribution

$$D_{\mathcal{A}} = \left[\begin{array}{l} k_1 \leftarrow \mathsf{Gen}(1^\lambda) \\ k_2 \leftarrow \mathsf{Gen}(1^\lambda) \\ C \leftarrow \mathcal{A}^{\mathsf{Enc}(k_1,\cdot),\mathsf{Enc}(k_2,\cdot)}(1^\lambda) \end{array} : (k_1, k_2, C) \right],$$

and then the "Special Encryption" property can be written as

$$\forall \mathcal{A}, \Pr\left[(k_1, k_2, C) \leftarrow D_{\mathcal{A}} : \begin{array}{c} \mathsf{Dec}(k_1, C) \neq \perp \\ \wedge\ \mathsf{Dec}(k_2, C) \neq \perp \end{array} \right] = \mathsf{negl}(\lambda). \tag{1}$$

Prove that the "Special Encryption" property 1 above holds. Hint: Follow these suggested steps:

1. Define a similar distribution, $D'_{\mathcal{A}}$, which is like $D_{\mathcal{A}}$ except that the Enc, Dec oracles replace the PRF F with a truly random function.
2. Show that $D'_{\mathcal{A}}$ satisfies the property, using the fact that $\mathcal{A}$ makes a polynomial number of queries.
3. Show that $D'_{\mathcal{A}}$ is indistinguishable from $D_{\mathcal{A}}$, by using a reduction to the definition of PRF.

4 Group Theory [18 pts]

4.1 Identifying Groups [4 pts]

Which of the following are groups? Circle them. (No need to prove.)

a. $(\{0,1\}^\lambda, ||)$, where $\{0,1\}^\lambda$ is the set of λ-bitstrings and $||$ denotes concatenation

b. $(\mathbb{R}^{\lambda\times\lambda}, \times)$, where $\mathbb{R}^{\lambda\times\lambda}$ is the set of all $\lambda \times \lambda$ matrices over real numbers, and the operation is matrix multiplication

c. $(\mathbb{Z} - \{0\}, \times)$, where $\mathbb{Z} - \{0\}$ is all the integers except for 0

d. $(A \times B, \star)$, where $(A, +)$ and $(B, \times)$ are groups, $A \times B$ is their cartesian product, and $(a_1, b_1) \star (a_2, b_2) = (a_1 + a_2, b_1 \times b_2)$

4.2 Cyclic Groups [8 pts]

a. Prove the following claim: All prime-order groups are commutative (meaning, $x \cdot y = y \cdot x$).

b. Consider the group $\mathcal{G} = \mathbb{Z}^*_{6q}$, where q is a prime number greater than 6. What is the order of this group? (Answer with a closed form solution in terms of q.)

4.3 Applications of Lagrange's Theorem [6 pts]

Recall that a crypto eggs are $X = g^x$ where $x \in \mathbb{Z}_p$, $g \in \mathcal{G}$, $|\mathcal{G}| = p$, and p is a large prime.

Alice posts on Piazza, "My Crypto Egg public key is special, it has a square root! My public key is X, and I found a group element Y such that $Y^2 = X$."

a. How special is Alice's egg? (e.g., She's right, some X have square roots; She's wrong, X can't have a square root; She's wrong, every X has a square root). Whichever you choose, show why.

b. If X has a square root, do you need the secret key x to find it? Explain why or why not.

5 Interactive Proofs [20 pts]

This question involves a variant of the Sigma protocols for Zero Knowledge Proofs discussed in class. You'll have to work through the protocol construction, definitions, and proofs.

Alice knows two secret keys $x \xleftarrow{\$} \mathbb{Z}_p, y \xleftarrow{\$} \mathbb{Z}_p$, such that her public keys are $X = g^x, Y = g^y$. (Assume that Alice posted X, Y publicly on Piazza).

Alice posts on Piazza, "My crypto egg secrets x and y are special. They lie on the elliptic curve $y^2 = x^3 + 7 \mathsf{mod} p$. How can Alice use a zero knowledge proof to prove this, without revealing any other information x and y?

Let $\mathcal{G}_\lambda$ be a family of groups in which the discrete log problem is hard, and the order of each group in the family is $|\mathcal{G}_\lambda| = p_\lambda = 2^{\mathsf{poly}(\lambda)}$.

5.1 [4 pts]

Illustrate an ideal functionality below that could carry out this protocol:

5.2 [4 pts]

Write the goal for the necessary ZK proof scheme using Camenisch-Stadler notation.

$$ZK\{(\_) : \_\}$$

Simplify or rearrange the C-S specification at this point to simplify the next steps.

Hint: It may help to create one or more ephemeral commitments.

5.3 [4 pts]

Finish constructing the protocol (P, V) below, where P is the prover and V is the verifier, such that $\left[P(1^\lambda, a) \leftrightarrow V(1^\lambda, A)\right]$ emulates the ideal functionality .

1. Round 1 (commit):
2. Round 2 (challenge): V sends the challenge c to P where

$$c \xleftarrow{\$} \mathbb{Z}_p \backslash \{0\}$$

3. Round 3 (response): What does P send to V in response?
4. Verification: What does V do with the response?

5.4 [4 pts]

Define the "Honest Verifier Zero Knowledge" (aka "Simulatable") property for this scheme, in terms of View_P and View_V. Be sure to explain what the views consist of.

Give a construction for the simulator and prove it satisfies the definition.

5.5 [4 pts]

Define the "Extractability" property for this scheme.

Give a construction for the extractor $\mathcal{E}$ and prove it extracts correctly with non-negligible probability.

6 Zero Knowledge Applications [18 = 10+8 points]

6.1 Zero knowledge specifications [10 points]

Let g be a generator in a prime-order group of size p with 256-bits. Give an informal procedure for proving the following statements under Honest Verifier Zero Knowledge. Answer by writing in Camenisch-Stadler notation, and explain how to derive any witnesses or ephemeral values needed. You can make use of range proofs that a committed value lies in the range $[0, 2^k)$ for some fixed k without further explanation.

1. [5 pts] How would you prove in ZK that $2^{64} \geq a \geq b \geq 0$ for a pederson commitment $A = g^a h^r$ and $B = g^b h^{r'}$?

2. [5pts] How could you prove in ZK that $a \neq b$ for a pederson commitment $A = g^a h^r$ and $B = g^b h^{r'}$?

6.2 How NOT to Prove Yourself [8 pts]

Bob needs Alice to generate some Diffie Hellman triples, A, B, C where $\log_g A \cdot \log_g B = \log_g$ (in other words, $A = g^a, B = g^b, C = g^{ab}$ for some a and b). To make sure Alice generates valid triples, Bob expects her to provide a zero knowledge proof that they're constructed correctly. Bob writes down the following Camenisch-Stadler notation for his goal

$$ZK\{(a, b) : A = g^a, B = g^b, C = A^b\}$$

and then proceeds to provide a protocol based on Fiat-Shamir:

- $\mathsf{Prove}(a, b)$:
  1. $k_1 \xleftarrow{\$} \mathbb{Z}_p$
  2. $k_2 \xleftarrow{\$} \mathbb{Z}_p$
  3. $K_A := g_1^k$
  4. $K_B := g_2^k$
  5. $K_C := A_2^k$
  6. $c := \mathcal{H}(g^a \| g^b \| K_A \| K_B \| K_C)$
  7. $s_1 := k_1 + ca$
  8. $s_2 := k_2 + cb$
- $\mathsf{Verify}(A, B, C, K_A, K_B, K_C, s)$:
  1. Recompute $c := \mathcal{H}(A \| B \| K_A \| K_B \| K_C)$
  2. Check if $K_A A^c = g^{s_1}$
  3. Check if $K_B B^c = g^{s_2}$
  4. Check if $K_C C^c = A^{s_2}$

Whoops! That's broken. Show how Alice can generate an invalid triple (A, B, C) such that $\log_g A \cdot \log_g B \neq \log_g C$ but that passes the Verify test anyway.

This problem is inspired by the paper "How Not To Prove It," which you are welcome to use as a reference. (Since the reference uses slightly different variable names, the challenge of this problem is in translating all the notation correctly.)

7 Reductions and Hard Problems [3pts]

Which of these problems reduces to the others?

- (Computational Diffie Hellman): $\left[a \xleftarrow{\$} \mathbb{Z}_p, b \xleftarrow{\$} \mathbb{Z}_p; \mathcal{A}(1^\lambda, g^a, g^b) = g^{ab}\right]$
- (Discrete Logarithm): $\left[X \xleftarrow{\$} \mathcal{G}, x \leftarrow \mathcal{A}(1^\lambda, X) : g^x = X\right]$
- (Decisional Diffie Hellman): Distinguish

  $$\{a \xleftarrow{\$} \mathbb{Z}_p, b \xleftarrow{\$} \mathbb{Z}_p : (g^a, g^b, g^{ab})\}$$

  from

  $$\{a \xleftarrow{\$} \mathbb{Z}_p, b \xleftarrow{\$} \mathbb{Z}_p, r \xleftarrow{\$} \mathbb{Z}_p : (g^a, g^b, g^r)\}$$

(Hint: you should prove two reductions)

8 Design Questions [10 pts]

Piazza is great for asking questions anonymously, but the anonymity has a limit — you have to be logged in. Sanket, the TA for this course, is dissatisifed with Piazza's trust model, and wants to develop a fancy new cryptography tool for providing "Truly Anonymous Questions." The idea is, each student would start off with a credit of 3 tokens, which they can use to request advice anonymously. You can assume that students can post to a public bulletin board anonymously, but Sanket only wants to answer $3N$ total questions where N is the number of students actually in the course.

Help Sanket by designing a protocol to provide the following properties:

- Students that follow the protocol receive 3 tokens from Sanket.
- Each token can be claimed once and only once (so each student can ask up to 3 anonymous questions).
- The TA as well as the students learn the total number of questions asked, but no other information about how many questions (if any) each student asked.

You may use any of the cryptographic primitives we have discussed so far in the semester (e.g., zero-knowledge proofs, key exchange, symmetric/asymmetric encryption, digital signatures, commitments, oblivious transfer,

garbled circuits). You may assume that the TA Sanket will behave semi-honestly (i.e., will follow the protocol, but will try to learn as much information as he can from his view). You may use any high-level pseudocode inside your Camenisch-Stadler notation you like, e.g. $ZK\{(k, x) : C = \mathsf{Enc}_k(x)\}$ could be a proof that an encryption was formed correctly). To specify the precise security goals of the protocol, please give both an ideal functionality as well as attack-game definitions.

9 Bonus [2 pts]

Write a short cryptography joke :)

Note: for full points, the joke must be both original and funny.

Ineligible jokes include:

- any pun about "salt"
- "the lecture on zero knowledge was so good, I didn't learn anything!"