CSC I4900 Midterm

Important: please answer **only 3** of the following questions. Answer the three you feel most confident about, and leave everything else blank. **Reminder:** You can use your computer, our lecture notes, our books (Nigel Smart, Abhi + Rafael) and chat with me on Twitch but please use **nothing else**. Do not look for answers online, and do not communicate with anyone but me during the exam. You'll have 150 minutes for the exam.

Name: _____

Problem	Score
1 (10 points)	
2 (10 points)	
3 (10 points)	
4 (10 points)	
5 (10 points)	
6 (10 points)	
7 (10 points)	
8 (10 points)	
9 (10 points)	
total: 30 points	

1. Prove that the DDH assumption is **false** for the group $G = \mathbb{Z}_p^\times$, where p is a prime. Take for granted some generator $\langle g \rangle = \mathbb{Z}_p^\times$.

2. Suppose that $\gamma \in \mathbb{Z}_p^\times$ generates a group G of prime order q, and say $q^2 \nmid (p-1)$. With such parameters, the DDH assumption is conjectured to hold. However, if we were to use this group for ElGamal encryption, the plaintext space would be $\langle \gamma \rangle$ and not all of $\mathbb{Z}_p^\times$. Show in fact that ElGamal is **not** IND-CPA secure if we use it to encrypt arbitrary messages in $\mathbb{Z}_p^\times$. (That is, if we are allowed to encrypt values that are not a power of γ.) *Hint:* multiplying $h \in \langle \gamma \rangle$ with $k \notin \langle \gamma \rangle$ results in $hk \notin \langle \gamma \rangle$, and if you think about the CRT, you will see that it is easy to check whether or not a value is in $\langle \gamma \rangle$. How could you modify the scheme to encrypt arbitrary bit strings (say of length $\ell < \log_2 q$) if given a hash function $H : \langle \gamma \rangle \longrightarrow \{0,1\}^\ell$? (You can model the hash function as a random oracle.)

3. Suppose that a prime number p is of the form $p = 2^k + 1$. Describe an efficient algorithm for computing discrete logarithms in $\mathbb{Z}_p$. That is, an efficient procedure for computing the exponent x when given the value $y = g^x \mod p$ where $\langle g \rangle = \mathbb{Z}_p^\times$. ("Efficient" means polynomial time in k.) *Hint:* you can learn the bits of the exponent with the help of the Legendre symbol $x^{(p-1)/2}$.

4. Demonstrate that public-key encryption which protects against a *computationally unbounded* adversary is not possible to achieve. Give as formal an argument as you can.

5. Describe a protocol for *re-sharing* a secret that has already been shared via polynomial interpolation (the Shamir scheme). The protocol should **not** at any point reconstruct the original secret, and the new shares should be distributed identically to "fresh" shares. You can assume the parties are "honest but curious" and will faithfully execute the protocol you specify, but might try to learn additional information from the communications.

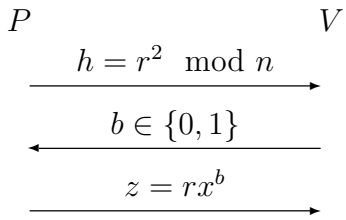
6. A t-out-of-n secret sharing scheme is information-theoretically secure if any subset of $\leq t$ shares reveals no information about the secret. More precisely, for any values $s_1, \ldots, s_t$ of a collection of t shares and for any probability distribution S on the space of secrets,
$$\Pr[S = s \mid s_1, \ldots, s_t] = \Pr[S = s].$$
Show that any information-theoretically secure secret sharing scheme which shares secrets from a finite set X using shares from another finite set Y must satisfy $|X| \leq |Y|$. *Hint:* it's a lot like the proof that you need to use long keys for information-theoretically secure encryption.

7. A function $f : X \longrightarrow Y$ is said to be *random-self-reducible* if an efficient procedure to compute f on *random* instances can be used to compute f on *any* instance. Put another way, the existence of an oracle $\mathcal{O}$ with the following property
$$\Pr_{x \xleftarrow{\$} X}[\mathcal{O}(x) = f(x)] \geq \frac{1}{\text{poly}(\ell)}$$

(where ℓ represents a security parameter), would imply the existence of an efficient[1] procedure (which can make calls to $\mathcal{O}$) for computing f on *any* input. Show how both inverting RSA (computing m from $x = m^e \mod n$) and the discrete log problem (computing a from $x = g^a \mod p$) are random-self-reducible. *Note:* the oracle $\mathcal{O}$ only "works" on uniformly random instances! You can't make any assumptions about what it does for specific or arbitrarily chosen inputs. You only know what it does on average.

8. A *block cipher* (like AES) applies a pseudorandom permutation to a message block based on a key k: $C_k : \{0,1\}^\ell \longrightarrow \{0,1\}^\ell$. To encrypt messages $m \in \{0,1\}^*$ which are longer than ℓ bits, one approach (called "electronic codebook mode") is to simply apply C_k to each ℓ-bit block of message m, concatenating the results to produce the ciphertext. Prove that this approach is **not** IND-CPA secure.

9. Let n be an RSA number ($n = pq$, where p, q are distinct ℓ-bit primes) and $y = x^2 \mod n$, for some $x \in \mathbb{Z}_n^\times$. Suppose that a prover P who knows x wishes to prove its knowledge of a square root x this to a verifier V that knows only y. Consider the following protocol:



Here, $r \in_R \mathbb{Z}_n^\times$ is chosen by the prover and $b \in_R \{0,1\}$ is chosen by the verifier. The above is repeated k times (with independent random choices for r, b in each round) and the verifier accepts the proof if and only if in each round $z^2 = hy^b$. Show that this protocol is zero knowledge for an honest verifier. Further, show that it is actually a proof of knowledge – if the prover could be rewound to answer different values of b with the same value of r, then the secret square root x could be extracted.

[1] By this, we mean that the algorithm always runs in polynomial time and succeeds in computing $f(x)$ with all but negligible probability (independent of the instance x!).