Final Exam

1 Problem 1 (80 points)

(a) Let $F : \{0,1\}^n \to \{0,1\}^n$ be a one-way *permutation* - that is, F is a one-way function that is also a permutation. Explain why $F(F(x))$ is also a one-way permutation

(b) Give an example of a one-way function $F : \{0,1\}^n \to \{0,1\}^n$ such that $F(F(x))$ is not a one-way function. You may assume a one-way function F' on $\{0,1\}^{n/2} \to \{0,1\}^{n/2}$

(c) Let N be the product of two large primes such that 3 is relatively prime to $\phi(N)$. The equation $x^4 = x \bmod N$ has as solutions $x = 0$ and $x = 1$. Explain why finding *any* other solution is as hard as factoring N.

(d) Let N be the product of two large primes such that $3 \times 5 \times 7 = 105$ is relatively prime to $\phi(N)$. Show that if you can compute 6th roots mod N, then you can also compute both 5th roots and 7th roots mod N. Explain why the converse may not be true (namely, that being able to compute both 5th roots and 7th roots mod N does not necessarily give you the ability to compute 6th roots)

(e) Let q be a large prime such that $p = (q-1)/2$ is also prime, and let g be a generator of $\mathbb{Z}_q^*$. Consider the one-way function $F : \{0, \ldots, q-2\} \to \mathbb{Z}_q^*$ given by $F(x) = g^x \bmod q$. Consider the function $h : \{0, \ldots, q-2\} \to \{0,1\}$ given by $h(x) = x \bmod 2$. Explain why h is *not* a hardcore bit for F.

(f) Suppose you are given a pseudorandom function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. Construct a commitment scheme for single bit messages. You do not need to formally prove security, but you should explain intuitively why your scheme is secure

(g) Your friend claims to have an algorithm that can solve all of NP in polynomial time. They want to sell you the algorithm. Since you are suspicious that the algorithm exists, you ask to see a zero-knowledge proof that their algorithm does as claimed (let's assume that the existence of the algorithm can be phrased as an NP statement, and therefore proved in zero knowledge). To your surprise, your friend says "Ah, I already proved it to Bob, and here's the transcript of our proof", and hands you a transcript of a proof, which passes verification.

Explain why you should not trust your friend.

(h) Suppose you have two public-key encryption schemes $(\mathsf{Gen}_0, \mathsf{Enc}_0, \mathsf{Dec}_0), (\mathsf{Gen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$. You are pretty sure at least one of them is CCA-secure, but suspect that one may be insecure; you don't know which. A friend suggests encrypting m by nesting the encryptions: $\mathsf{Enc}_0(\mathsf{pk}_0, \mathsf{Enc}_1(\mathsf{pk}_1, m))$. In the CPA setting, this will give a CPA secure scheme provided at least one of the two schemes is CPA secure (this was a solution to one of your homework exercises).

Explain why this nested scheme may not work in the CCA setting. That is, if one of the two schemes is CCA secure, the nested scheme may not be.

2 Problem 2 (30 points)

Two standards committees propose to save bandwidth by combining compression (such as the Lempel-Ziv algorithm used in the zip and gzip programs) with encryption. Both committees plan on using a variable length secret key encryption scheme, such as CBC-mode.

(a) One committee proposes to compress messages before encrypting them. Explain why this is a bad idea.

(b) The other committee proposes to compress ciphertexts after encryption. Explain why this is a bad idea.

Over the years many problems have surfaced when combining encryption and compression. The CRIME and BREACH attacks are good representative examples.

3 Problem 3 (30 points)

Let $G : \{0,1\}^s \to \{0,1\}^n$ be a PRG with $s < n$. We can use G to get a stream cipher $\mathsf{Enc}(k, m) = G(k) \oplus m$. We saw in class that if the PRG is a secure pseudorandom generator, then Enc has ciphertext indistinguishability (in the one-time setting)

Prove the converse: if $\mathsf{Enc}(k, m) = G(k) \oplus m$ has ciphertext indistinguishability, prove that G *must* be a pseudorandom generator.

4 Problem 4 (40 points)

Consider the following variant of the 3DES construction that uses only two keys: for a block cipher (E, D) with key space $\mathcal{K}$, define E' as $E'((k_1, k_2), m) := E(k_1, E(k_2, E(k_1, m)))$.

Show that this block cipher can be defeated by a meet-in-the-middle style attack using $O(|\mathcal{K}|)$ evaluations of E and D and using $O(|\mathcal{K}|)$ queries to the block cipher challenger.

5 Problem 5 (40 points)

Let $\mathbb{G}$ be a cyclic group of prime order p with generator g. For an $n \times m$ matrix $A \in \mathbb{Z}_p^{n \times m}$, let g^A be the the $n \times m$ matrix whose i, j entry is $g^{A_{i,j}}$. In other words, g^A is discrete exponentiation, applied component-wise over the elements of A.

Let $D_0(n, m)$ be the distribution of g^A for a uniformly random matrix A. Let $D_1(n, m)$ be the distribution over g^A, where A is chosen at random subject to the restriction that A is a matrix of rank 1. Here, we will consider consider the problem of distinguishing $D_0(n, m)$ from $D_1(n, m)$.

(a) Explain how DDH is essentially identical to this problem in the case $n = m = 2$ (remember that we defined DDH as distinguishing (h, h^a, h^b, h^{ab}) from (h, h^a, h^b, h^c) for a random generator h and random $a, b, c \in \mathbb{Z}_p$)

(b) Prove that, if the DDH assumption holds on $\mathbb{G}$, then $D_0(n, m)$ is computationally indistinguishable from $D_1(n, m)$ for any constant n, m. That is, show how to take any distinguisher for $D_0(n, m), D_1(n, m)$ and obtain a DDH adversary.

Hint: The following fact may be useful (you can take it as given; you don't need to prove it). Fix a matrix A of rank r. Then choose random full-rank $B \in \mathbb{Z}_p^{n \times n}$ and full-rank $C \in \mathbb{Z}_p^{m \times m}$. Then the matrix $B \cdot A \cdot C$ is a truly random matrix subjected to the restriction that the rank is r.

Another fact that may be useful (and again you don't need to prove): If E is chosen uniformly at random in $\mathbb{Z}_p^{k \times \ell}$, then it will be full rank (namely, the rank is $\min(k, \ell)$) with probability $1 - O(1/p)$. Since p is huge, this probability is overwhelming.

6 Problem 6 (40 points)

Show that the following problems are equivalent in a cyclic group $\mathbb{G}$ of prime order p:

(1) Given g, g^a, g^b for a random generator g, and random scalars $a, b \in \mathbb{Z}_p$, compute g^{ab}. This is just the computational Diffie-Hellman problem.

(2) Given g, g^a for a random generator g and random scalar $a \in \mathbb{Z}_p$, compute g^{a^2}

(3) Given g, g^a for a random generator g and a random scalar $a \in \mathbb{Z}_p \setminus \{0\}$, compute $g^{1/a}$.

That is, if you have an adversary A for any one of the problems that runs in time t and succeeds with probability ϵ, you can construct an adversary A' for either of the other problems that runs in time $\mathsf{poly}(t, 1/\epsilon)$ and succeeds with probability $\mathsf{poly}(1/t, \epsilon)$.

Notice that you can prove this fact with as few as three such reductions, say by showing that an adversary for (1) can be used to construct an adversary for (2), an adversary for (2) can be used to construct an adversary for (3), and finally that an adversary for (3) can be used to construct an adversary for (1). Of course, you can use any set of reductions you want, and you are allowed to use more that 3 if that is more convenient. If you are unable to fully prove equivalence between all three problems, show as many directions as you can for partial credit.

7 Problem 7 (40 points)

Here, we will see how to shorten the public keys for signature schemes. Let $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a many-time secure signature scheme, with public keys in some set $\mathcal{X}$. Let $H : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a collision resistant hash function. Show how to construct a many-time secure signature scheme $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ with public keys in $\mathcal{K} \times \mathcal{Y}$. Note that $\mathcal{K} \times \mathcal{Y}$ will in general be much smaller than $\mathcal{X}$. Prove the security of your scheme.

Hint: Gen' *will work as follows. To generate a public and secret key, run* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}()$ *and* $k \leftarrow \mathcal{K}$*, and then output* $\mathsf{sk}' = \mathsf{sk}$ *and* $\mathsf{pk}' = (k, H(k, \mathsf{pk}))$*. Your job is to describe how to sign and verify, and to prove security.*