Problem 1 Merkle hash trees.
Merkle suggested a parallelizable method for constructing hash functions out of compression functions. Let \( f \) be a compression function that takes two 512 bit blocks and outputs one 512 bit block. To hash a message \( M \) one uses the following tree construction:

![Diagram of Merkle hash tree]

Prove that if one can find a collision for the resulting hash function then one can find collisions for the compression function.

Problem 2 In this problem we explore the different ways of constructing a MAC out of a non-keyed hash function. Let \( h : \{0,1\}^* \rightarrow \{0,1\}^b \) be a hash function constructed by iterating a collision resistant compression function using the Merkle-Damgård construction.

1. Show that defining \( MAC_k(M) = h(k \parallel M) \) results in an insecure MAC. That is, show that given a valid text/MAC pair \((M, H)\) one can efficiently construct another valid text/MAC pair \((M', H')\) without knowing the key \( k \).

2. Recall that in the Merkle-Damgård iterated construction one uses a fixed Initial Value IV as the initial chaining variable. Show that setting the IV to be the secret key \( k \) results in an insecure MAC.

---

Assignment #3

Due: Monday, February 28th, 2000.
3. Consider the MAC defined by \( MAC_k(M) = h(M \parallel k) \). Show that in expected time \( O(2^{b/2}) \) it is possible to construct two messages \( M \) and \( M' \) such that given \( MAC_k(M) \) it is possible to construct \( MAC_k(M') \) without knowing the key \( k \).

4. Give a short high level argument to show why the envelope method for constructing a MAC out of a hash function produces a secure MAC.

**Problem 3** Rabin suggested a signature scheme very similar to RSA signatures. In its simplest form, the public key is a product of two large primes \( N = pq \) and the private key is \( p \) and \( q \). The signature \( S \) of a message \( M \in \mathbb{Z}_N \) is the square root of \( M \) modulo \( N \). For simplicity, assume that the messages \( M \) being signed are always quadratic residues modulo \( N \). To verify the signature, simply check that \( S^2 = M \mod N \). Note that we did not include any hashing of \( M \) prior to signing. Show that a chosen message attack on the scheme can result in a total break. More precisely, if an attacker can get Alice to sign messages chosen by the attacker then the attacker can factor \( N \).

**Hint:** recall that a quadratic residue modulo \( N = pq \) has four square roots in \( \mathbb{Z}_N \). First show that there are two square roots of \( M \) that enable the attacker to factor \( N \) (use the fact that gcd’s are easy to compute). Then show how using a chosen message attack the attacker can get a hold of such a pair of square roots. Note that proper hashing prior to signing prevents this attacks.

**Problem 4** Suppose Alice and Bob share a secret key \( k \). A simple proposal for a MAC algorithm on fixed length messages is as follows: given a message \( M \) do: (1) compute 128 different parity bits of \( M \) (i.e. compute the parity of 128 different subsets of the bits of \( M \)), and (2) DES encrypt the resulting 128-bit checksum using \( k \). Naively, one could argue that without knowing \( k \) an attacker cannot compute the MAC of a message \( M \). Show that this proposal is flawed. Note that the algorithm for computing the 128-bit checksum is public.

**Hint:** show that an attacker can carry out an existential forgery given one valid message/MAC pair. Use linear algebra modulo 2.

**Extra Credit** Recall that in the ElGamal signature scheme, a signature is of the form \((a, b)\) where \( b \in \mathbb{Z}_q \) and \( a \) is an integer. In lecture, we glossed over the fact that \( a \) is required to be less than \( p \). Show that without this restriction, one can forge signatures for any message.

**Hint:** ElGamal says to find \((a, b)\) such that \( g^a a^b = g^M \). First show how to find \((b, c, d)\) such that \( g^d e^b = g^M \).