MATH/CMSC 456 (Washington) Exam 2 April 29, 2010

Do each problem on a separate sheet of paper (so you need 5 sheets). Do Problem 1 on page 1, Problem 2 on page 2, etc. Do not staple. Put your name on each sheet.

1. (20 points = 5+5+10)

(a) Show that 3 is a primitive root mod 7.

(b) Show that 2 is not a primitive root mod 7.

(c) Suppose you know that $3^6 \equiv 2^2 \cdot 11 \pmod{137}$ and $3^{10} \equiv 2 \pmod{137}$. Find a value of x such that $3^x \equiv 11 \pmod{137}$.

2. (20 points = 15 + 5) Suppose p is a large prime, α is a primitive root, and $\beta \equiv \alpha^a \pmod{p}$. The numbers p, α, β are public. Peggy wants to prove to Victor that she knows a without revealing it. They do the following:

1. Peggy chooses random numbers $r_1, r_2, r_3 \pmod{p-1}$ with $r_1 + r_2 + r_3 \equiv a \pmod{p-1}$.
2. Peggy computes
$$h_1 \equiv \alpha^{r_1}, \quad h_2 \equiv \alpha^{r_2}, \quad h_3 \equiv \alpha^{r_3} \pmod{p}$$
and sends h_1, h_2, h_3 to Victor.
3. Victor chooses $i, j \in \{1, 2, 3\}$ and asks Peggy to send r_i and r_j.

(a) Give the remaining steps in the procedure. Victor wants to be at least 99% convinced that Peggy knows a.

(b) Give a reasonable method for Peggy to choose the three random numbers such that $r_1 + r_2 + r_3 \equiv a \pmod{p-1}$. (A method that doesn't work is "Choose three random numbers and see if their sum is a; if not, keep trying.")

3. (20 points) Nelson has a hash function H_1 that gives an output of 60 bits. Friends tell him that this is not a big enough output, so he takes a strong hash function H_2 with a 200-bit output and uses $H(x) = H_2(H_1(x))$ as his hash function. That is, he first hashes with his old hash function, then hashes the result with the strong hash function to get a 200-bit output, which he thinks is much better. The new hash function H can be computed quickly. Does it have preimage resistance, and does it have strong collision resistance? Explain your answers. (*Note:* Assume that computers can do up to $2^{50} \approx 10^{15}$ computations for this problem. Also, since it is essentially impossible to prove rigorously that most hash functions have preimage resistance or collision resistance, if your answer to either of these is "yes" then your explanation is really an explanation of why it is probably true.)

4. (20 points = 6+6+1+7) Recall the ElGamal Signature scheme: Alice chooses a large prime p, a primitive root α, and a secret integer a, and computes $\beta \equiv \alpha^a \pmod{p}$. She chooses a random k with $\gcd(k, p-1) = 1$ and computes

$$r \equiv \alpha^k \pmod{p}; \quad s \equiv k^{-1}(m - ar) \pmod{p-1}.$$

The signed message is (m, r, s). A signature (m, r, s) is valid if $\beta^r r^s \equiv \alpha^m \pmod{p}$.

(a) Suppose Alice chooses $k = 1$. Then $r \equiv \alpha \pmod{p}$, so it doesn't look like Alice sends something that isn't already known. Explain in detail, however, how Eve can break the system).

(b) (no relation to part (a)) Let $r \equiv \alpha\beta \pmod{p}$ and $m \equiv -r \pmod{p-1}$. What value of s is needed for a valid signed message (m, r, s)?

(c) Explain why it is unlikely that $\beta^r \equiv \beta^{\alpha\beta} \pmod{p}$ in part (b) (*Suggestion:* Now go back to your solution of (b) and replace all occurrences of $\beta^{\alpha\beta}$ with β^r).

(d) (no relation to parts (a), (c)) Suppose a hash function H is used, so $H(m)$ is signed and the verification equation for (m, r, s) becomes

$$\beta^r r^s \equiv \alpha^{H(m)} \pmod{p}.$$

Why is the method of part (b) unlikely to succeed in producing a valid signed message (m, r, s) ?

5. (20 points = 10 + 10)

(a) If there are 5 people in a room, what is the probability that no two of them have birthdays in the same month? (you may leave the answer as a product, or you may use an approximation for this product; assume that each person has probability 1/12 of being born in any given month).

(b) Let n be the product of two distinct large primes. Let $H(x) = x^2 \pmod{n}$. What properties of a hash function does H satisfy, and which properties does it not satisfy? Explain why. Do not worry about how fast H may be computed. It's fast, but probably not fast enough to be a good hash function.