Homework 7

Due Thursday, October 29

Written Part

6. Let's do a few checks from the implementation part.

   (a) Compute the Jacobi symbols $\left(\frac{8}{15}\right), \left(\frac{11}{15}\right), \left(\frac{12}{15}\right)$ by hand and confirm your solutions from 1(c) are correct.

   Proof.
$$\left(\frac{8}{15}\right) = \left(\frac{8}{3}\right)\left(\frac{8}{5}\right) = \left(\frac{2}{3}\right)\left(\frac{3}{5}\right) = (-1)(-1) = 1,$$
   where we verify directly that 2 is not a square mod 3 and 3 is not a square mod 5.
$$\left(\frac{11}{15}\right) = \left(\frac{11}{3}\right)\left(\frac{11}{5}\right) = \left(\frac{2}{3}\right)\left(\frac{1}{5}\right) = (-1)(1) = -1,$$
   where again 2 is not a square mod 3, but 1 is certainly a square mod 5.
$$\left(\frac{12}{15}\right) = \left(\frac{12}{3}\right)\left(\frac{12}{5}\right) = \left(\frac{0}{3}\right)\left(\frac{2}{5}\right) = (0)(-1) = 0.$$
   □

   (b) In the Goldwasser-Micali algorithm it was suggested that the random number be chose as greater than $\sqrt{N}$. Why?
   Suppose $r < \sqrt{N}$ and Bob wanted to send the bit 0. Then the ciphertext would be $c \equiv r^2 \mod N$, but the reduction of r^2 modulo N is r^2, which is easily verified as a square (since it is a square in $\mathbb{Z}$). Since the security of the algorithm depends on it being difficult to know whether c is a square or not, the security is compromised.

7. Let p be an odd prime and $g \in \mathbb{F}_p^*$ a primitive root. Fix any $h \in \mathbb{F}_p^*$. In this problem we study how to get information about $\log_g(h)$.

   (a) Describe how to easily tell $\log_g(h)$ is even or odd.
   By HW3 Problem 6 we know that $\log_g(h)$ is even if and only if h has a square root. This holds precisely when the Legendre symbol $\left(\frac{h}{p}\right) = 1$. But the Legendre symbol is easily computed as $h^{\frac{p-1}{2}} \mod p$.

   (b) We can write $\log_g a$ in binary:
$$\log_g a = \varepsilon_0 + \varepsilon_1 \cdot 2 + \varepsilon_2 \cdot 2^2 + \varepsilon_3 \cdot 2^3 + \cdots \qquad \varepsilon_i \in \{0,1\}.$$
   Explain why (a) means that we know ε_0. This property is summarized as saying that the *first bit* of the discrete log problem over $\mathbb{F}_p$ is insecure.
   We know that ε_0 is 0 if $\log_g a$ is even, and 1 otherwise. In part (a) we showed the parity of the log is easily computed, therefore so is ε_0.

(c) If $p-1$ is divisible by higher powers of 2, we can recover more bits! Factor $p-1 = 2^k m$. Describe an algorithm to compute the first k bits of $\log_g h$, that is, to recover $\varepsilon_0, \varepsilon_1, \cdots, \varepsilon_{k-1}$. You may assume that there is a fast algorithm to compute square roots modulo p (if $p \equiv 3 \mod 4$ we described such an algorithm is class, but there is a general fast algorithm which we may encounter in the coming weeks).
We should run through the following loop k times:

(1) Compute ε_0 using the method describd in part (a).

(2) If $\varepsilon_0 = 0$, set $a' = a$. Otherwise, set $a' = g^{-1}a$.

(3) Compute b such that $b^2 \equiv a \mod p$.

(4) Return to step (1) with a replaced by b.

Normally here one should give a proof of correctness, but because I didn't explicitly ask for it I will delay it until next week.

8. Let p be a prime number and $g \in \mathbb{F}_p^*$ a primitive root. Let i and j be integers such that $\gcd(j, p-1) = 1$. Let A be arbitrary. Set:

$$\begin{aligned} S_1 &\equiv g^i A^j \mod p \\ S_2 &\equiv -S_1 j^{-1} \mod p-1 \\ D &\equiv -S_1 i j^{-1} \mod p-1 \end{aligned}$$

(a) Show that the pair (S_1, S_2) is a valid Elgamal signature for the document D. In particular, this means Eve can produce valid Elgamal signatures.

Proof. We we run `elgamalVerify` we compute:

$$\begin{aligned} A^{S_1} S_1^{S_2} &\equiv A^{g^i A^j} (g^i A^j)^{-g^i A^j j^{-1}} \\ &\equiv A^{g^i A^j} A^{-g^i A^j} g^{-g^i A^j i j^{-1}} \\ &\equiv g^{-S_1 i j^{-1}} \mod p, \end{aligned}$$

which is precisely the value of $g^D \mod p$. □

(b) Explain why this doesn't mean that Eve can forge Sam's signature on a given document. What extra information would allow Eve to do this?
The document D depends on the choice of i and j. If one were to start for with D and try to reverse engineer i and j, one would have to solve when trying to find i and j giving S_1 (for example).

9. In this exercise we describe a potential security flaw in the Elgamal digital signature algorithm. Suppose that Samantha made the mistake of signing two documents D and D' using the same random value k.

(a) Explain how Eve can immediately recognize that Samantha has made this blunder.

Proof. An Elgamal encryption scheme fixes a prime p and primitive root g at the outset (in fact this is public information!). Then a signature consists of 2 peices (S_1, S_2), and the first $S_1 \equiv g^k \mod p$ only depends on k, and if the same k is used twice S_1 is the same each time. □

(b) Let the signature for D be $D^{sig} = (S_1, S_2)$ and the signature for D' be $D'^{sig} = (S_1', S_2')$. Explain how Eve can recover Samantha's secret signing key a.
We first see that $S_1 \equiv S_1' \equiv g^k \mod p$. Then we consider S_2 and S_2':

$$\begin{aligned} S_2 &\equiv (D - aS_1)k^{-1} \mod p-1 \\ S_2' &\equiv (D' - aS_1')k^{-1} \mod p-1. \end{aligned}$$

We first will first find k. We know the values of S_2, S_2', so we can subtract them, and because $S_1 \equiv S_1' \mod p$ we get the following congruence:

$$S_2 - S_2' \equiv (D - D')k^{-1} \mod p-1.$$

We also know the values of D and D' (these are the public documents), so that if $g = \gcd(D - D', p-1)$ is equal to 1, we could just divide and find k^{-1} (and therefore k). Unfortunately, this is not the case in general. Nevertheless, HW2 Problem 7 gave us methods to study solutions of linear equations modulo $p-1$. Let $s = S_2 - S_2'$ and $d = D - D'$. Then we are solving:

$$dx = s \mod p-1, \tag{1}$$

for x. We know k^{-1} is a solution, so that there are g many solutions to Equation 1 (by HW2 Problem 7). In fact, we showed in HW2 Problem 7 if a_0 is any solution to equation 1, the set of solutions is:

$$\left\{ a_0, a_0 + \frac{p-1}{g}, a_0 + 2\frac{p-1}{g}, \cdots, a_0 + (g-1)\frac{p-1}{g} \right\}.$$

We know that k^{-1} must be part of this list, so if we can find some a_0 solving this equation, we narrow our search considerably. To do this we use the extended Euclidean algorithm to find u, v such that $du + (p-1)v = g$. By HW2 Problem 7, the fact that Equation 1 has a solution means that $g|s$, so that $s/g = \ell \in \mathbb{Z}$. Multiplying the equation through by ℓ we get:

$$s = g\ell = du\ell + (p-1)v\ell \equiv d(u\ell) \mod p-1,$$

so that $a_0 = u\ell$ is a solution. Then one of $\{a_0, a_1, \cdots, a_{g-1}\}$ is k^{-1}, where $a_i = a_0 + i\frac{p-1}{g}$. To see which one it is, we compute

$$S_1^{a_i} = (g^k)^{a_i} = g^{a_i k} \mod p$$

for each i. If the output is congruent to g, then $g^{a_i k - 1} \equiv 1 \mod p$ so that the order of g (which is $p-1$) divides $a_i k - 1$. This implies that $a_i \equiv k^{-1} \mod p-1$, so that inverting this a_i recovers k.
This is a great start. Now that we know k we can try to recover a in a similar way. We will use the equation:

$$S_2 \equiv (D - aS_1)k^{-1} \mod p-1.$$

Multiplying through by k, subtracting D, and multiplying by -1 gives:

$$aS_1 = D - kS_2 \mod p-1 \tag{2}$$

As above, if $g' = \gcd(S_1, p-1)$ were equal to 1, then we could divide by S_1 and recover a. But of course this is not always true. We must run the same method as before, letting $d' = S_1$ and $s' = D - kS_2$, and searching for solutions to:

$$d'x = s' \mod p-1 \tag{3}$$

The process is identical. We first find a single solution using HW2 Problem 7 and the Euclidean algorithm to write $d'u' + (p-1)v' = g'$, multiplying through by ℓ' where where $g'/s' = \ell' \in \mathbb{Z}$, so that $x = a_0' = u'\ell'$ is a solution. Then we write the set of solutions $\{a_0', a_1', \cdots, a_{g-1}'\}$ where $a_i' = a_0' + i\frac{p-1}{g'}$. We know that a is a solution to equation 3, so that it must be equal to one of the a_i'. To find which one we compute $g^{a_i'} \mod p$ for each i, and see which one is equal to the public verification key $A \equiv g^a \mod p$. Since g is a primitive root, if $g^{a_i'} \equiv g^a \mod p$, we know $a_i' \equiv a \mod p-1$, and so we have extracted Sam's private signing key.

A few comments. First, in general the gcd of 2 numbers much smaller than the two numbers themselves, so reducing our search for k (respectively a) to just $\gcd(d, p-1)$ (resp. $\gcd(d', p-1)$) many candidates is quite a speed up. Second, each time we found our list of candidates ofor k (resp. a) we ran essentially the same process, so this would be a good pace to have a helper function.