1 Modern Cryptography

(Continued from previous notes.)

1.1 Elgamal Cryptosystem

The Elgamal cryptosystem is a public-key cryptosystem like RSA, named after the Egyptian cryptographer Taher Elgamal.

1.1.1 How Elgamal Works

The process begins with Bob choosing a public key. He picks a prime number p and a primitive root g of p. He chooses a random integer x with $0 \leq x < p-1$. This is his private key. He then computes $h = g^x \pmod{p}$ and his public key is the triple (p, g, h).

Suppose Alice wants to send Bob a message. She first encodes her message as an integer m between 0 and $p-1$ (e.g., by using the same "base 26" strategy that we employed for RSA.) Then, she chooses a random integer y between 0 and $p-1$ called the **ephemeral key**. Alice will have to choose a different ephemeral key for every message she sends, but Bob does not have to know the value of this key beforehand. Alice computes $s = h^y \pmod{p}$, $c_1 = g^y \pmod{p}$, and $c_2 = ms \pmod{p}$. Note that she can compute s and c_1 quickly using binary exponentation. The pair (c_1, c_2) is the ciphertext that she sends to Bob.

To decrypt the ciphertext (c_1, c_2), Bob first computes $c_1^x \pmod{p}$. Bob can do this quickly with binary exponentation. Notice that

$$c_1^x \equiv (g^y)^x = g^{xy} = (g^x)^y \equiv h^y \equiv s \pmod{p}.$$

In other words, Bob found the same value of s that Alice had, even though he does not know the value of the ephemeral key y. He then computes an inverse mod p of c_1^x using the extended Euclidian algorithm. From there, he computes

$$c_2(c_1^x)^{-1} \equiv c_2 s^{-1} \equiv (ms)s^{-1} \equiv m \cdot 1 = m \pmod{p},$$

thus allowing him to recover Alice's message m.

(Example.) Suppose Bob picks the prime $p = 4115549$ and $g = 2$ is his primitive root. He then picks a random integer $x = 2634326$. From there, he can compute

$$h = g^x \pmod{p},$$

getting $h = 1149114$. Thus, the triple $(4115549, 2, 1149114)$ is his public key. $x = 2634326$ must be kept secret.

Suppose Alice wants to send Bob the message `Hi Bob`. She begins by converting this message to the integer $m = 3340481$. Then, she chooses an ephemeral key $y = 2775147$. She keeps this value of y secret, and then computes

$$s = h^y \pmod{p} = 962840$$

using binary exponentation. Alice also keeps s a secret. She also computes

$$c_1 = g^y \pmod{p} = 621674$$

using binary exponentation. Finally, she computes

$$c_2 = ms \pmod{p} = 1911501.$$

From there, $(c_1, c_2) = (621674, 1911501)$ is the ciphertext she sends to Bob.

Bob receives the pair $(c_1, c_2) = (621674, 1911501)$. He computes

$$c_1^x \pmod{p} = 962840$$

using binary exponentation. This is the same value that Alice found for s. Then, he computes an inverse mod p and finds $s^{-1} \equiv 2329074 \pmod{p} = 4115549$. From there, he computes

$$c_2 s^{-1} \pmod{p} = 3340481,$$

and then converts this message back to the text `HIBOB`.

(Exercise.) Suppose Bob picks the prime $p = 29$ and the primitive root $g = 2$.

(a) Suppose Bob picks $x = 3$. What is his public key?

We compute

$$h = g^x \pmod{p} = 2^3 \pmod{29} = 8 \pmod{29}.$$

Therefore, Bob's public key is the triple $(p, g, h) = (29, 2, 8)$.

(b) Suppose Alice wants to send Bob the plaintext integer $m = 7$. What is the corresponding ciphertext pair?

Suppose Alice selects ephemeral key $y = 3$. Then, Alice can compute

$$s = h^y \pmod{p} = 8^3 \pmod{29} = 19 \pmod{29},$$

$$c_1 = g^y \pmod{p} = 2^3 \pmod{29} = 8 \pmod{29},$$

$$c_2 = ms \pmod{p} = 7(19) \pmod{29} = 17 \pmod{29}.$$

The pair, $(8, 17)$, is the ciphertext pair.

(c) Suppose Bob receives the ciphertext pair $(3, 9)$ from Alice. What is the plaintext integer m?

Bob computes

$$c_1^x \pmod p = 3^3 \pmod{29} = 27 \pmod{29}.$$

This value is s; that is, $s = 27 \pmod{29}$. From there, we want to find the inverse of $c_1^x = 27$ mod 29. To do this, let's find Bezout's coefficient;

$$29 = 27q + r \implies 29 = 27(1) + 2 \implies 2 = 29 + 27(-1)$$

$$27 = 2q + r \implies 27 = 2(13) + 1 \implies 1 = 27 + 2(-13)$$

$$2 = 1q + r \implies 2 = 1(2) + 0.$$

From this, $\gcd(27, 29) = 1$ so we can find the Bezout coefficient.

$$\begin{aligned} 1 &= 27 + 2(-13) \\ &= 27 + (29 + 27(-1))(-13) \\ &= 27 + 29(-13) + 27(-1)(-13) \\ &= 27 + 29(-13) + 27(13) \\ &= 27(14) + 29(-13). \end{aligned}$$

From this, it follows that the Bezout coefficients are $x = 14$ and $y = -13$; more importantly, we find that the inverse of $c_1^x = 27$ mod 29 is $x = 14$. So,

$$c_2 s^{-1} \pmod p = 9(14) \pmod{29} = 10 \pmod{29},$$

so $m = 10$.

(Exercise.) If Bob wants to be able to receive messages with $r = 10$ characters, how large must he choose p to be? What if $r = 100$? $r = 1000$?

Assuming we choose to use the "base 26" strategy for encoding the message, the largest possible 10 character message would be `ZZZZZZZZZZ`. Here, `Z` corresponds to the number 25, so we can encode this message as follows:

$$\sum_{i=0}^{9} 25 \cdot 26^i = 26^{10} - 1.$$

Recall that the integer encoding of the message m must be between 0 and $p-1$, i.e., $0 \le m \le p-1$. So, $p > 26^{10} - 1 \implies p - 1 > 26^{10} - 2$. The same reasoning applies for $r = 100$ and $r = 1000$.

1.1.2 Why Elgamal is Probably Secure (For Now...)

There are at least two strategies Eve might employ to recover the plaintext m from the ciphertext (c_1, c_2).

- Eve can try to find Bob's decryption key x so she can follow Bob's decryption strategy but, in order to do this, she needs to find the discrete log base g of h mod p.
- Even can try to find Alice's ephemeral key y, but then she needs to find the discrete log base h of c_1 mod p.

In any case, Eve needs to find a discrete log base g mod p. So, the security of the Elgamal cryptosystem relies on the presumed difficulty of the following:

(Discrete Logarithm Problem.) Suppose you are given a prime p, a primitive root g mod p, and an integer a not divisible by p. Find the discrete log base g of a mod n. In other words, find the unique integer k such that $0 \le k \le p - 1$ such that $g^k \equiv a \pmod p$.

As p gets larger, the problem becomes difficult for classical computers. The naive method to solving this problem would be to try all possible values of k from 1 to $p-1$, but this is linear in p and exponential in the number of digits of p. Although there are faster algorithms out there, they are not faster by much[1].

1.2 Diffie-Hellman Key Exchange

The Diffie-Hellman key exchange is *not* quite a cryptosystem for exchanging messages, but rather it is a protocol that allows Alice and Bob to share a secret, but neither has full control over the content of the shared secret. The shared secret can be used as the key for a symmetric key cipher like a one-time pad.

The procedure is as follows:

- Alice and Bob publicly agree to fix a prime p and a primitive root g mod p.
- Alice then chooses a secret integer $0 \le a < p-1$ and sends Bob $x = g^a \pmod{p}$. She can compute this value quickly using binary exponentation.
- Bob similarly chooses a secret integer $0 \le b < p-1$ and sends Alice $y = g^b \pmod{p}$.
- Alice computes $s = y^a \pmod{p}$ and Bob computes $s = x^b \pmod{p}$.

The two values of s that Alice and Bob computes are the same, because

$$y^a \equiv (g^b)^a = g^{ab} = (g^a)^b \equiv x^b \pmod{p}.$$

Thus, Alice and Bob now share a secret, s. Neither of them have full control over the shared secret, so this cannot be regarded as Alice or Bob sending a message to the other.

(Exercise.) Suppose Alice and Bob agree to use $p = 11$ and $g = 2$. Alice chooses the integer $a = 3$. She receives the integer $y = 5$ from Bob. What is her shared secret s with Bob?

The shared secret is

$$s = y^a \pmod{p} = 5^3 \pmod{11} = 4.$$

[1] There are no known algorithm that accomplishes this task that is polynomial in the number of digits of p.