Práctica 7
Facultad de Ciencias
Criptografía y Seguridad

José de Jesús Galaviz Casas
galaviz@ciencias.unam.mx

Edgar Omar Arroyo Munguía
omar.am@ciencias.unam.mx

Luis Fernando Yang Fong Baeza
fernandofong@ciencias.unam.mx

29 de Enero 2020

1. Criptografía de Curvas Elípticas

Como hemos visto en clase, el problema del logaritmo discreto sobre curvas elípticas, es muy útil, tiene muchas vertientes y nos ayuda para implementar criptosistemas como Diffie-Hellman, recordando que el problema del logaritmo discreto sobre curvas elípticas, consiste en dados un par de puntos P, Q sobre una curva $E(\mathbb{F})$, encontrar un $s \in \mathbb{F}$, tal que $Q = sP$, sin embargo, Diffie-Hellman no es el único criptosistema implementable sobre curvas elípticas, en ésta práctica se aprenderá de uno más conocido como ECIES.

De igual manera, las curvas elípticas así como pueden ser usadas para encriptar, pueden de igual manera ser usadas para ataques criptográficos, en particular sobre RSA, sin embargo, no lo suficiente como para comprometer la seguridad del algoritmo. Éste método es conocido como ECM (Elliptic Curve Method) y sirve para factorizar un número cualquiera.

2. ECIES

El criptosistema ECIES (Elliptic Curve Integrated Encryption Scheme), que también es un criptosistema de llave pública.

Supongamos como siempre que Alice, quiere enviar un mensaje $M = m_1 m_2 ... m_n$ a Bob, sin que Carl se entere del mensaje, entonces, Bob tiene que establecer sus parámetros de llave pública, para esto, Bob debe escoger una curva elíptica $E(\mathbb{F}_q)$ tal que sea difícil resolver el problema del logaritmo discreto para esa curva, además escoge un punto A obre la curva, usualmente de orden grande N. Posteriormente, escoge un entero secreto y calcula $B = sA$, de manera que Bob procede a publicar su llave pública como (q, E, N, A, B) y su llave privada es s, sin embargo, este algoritmo también necesita dos funciones *hash* seguras o criptográficas, H_1 y H_2. (Ej. SHA256, SHA1, SHA2...) y una función simétrica de encripción, digamos E_k, como puede ser un desplazamiento de césar o un cifrado afín (depende de la llave k), todos éstos parámetros son públicos.

Como Alice quiere enviar el mensaje M a Bob, entonces debe de proceder como sigue.

1. Escoger un entero aleatorio $1 \leq k \leq N - 1$
2. Calcular a $R = kA$ y $Z = kB$

3. Escribe la salida de $H_1(R, Z)$ como $k_1||k_2$, esto es simplemente k_1 y k_2 concatenados y cada uno con su longitud específica.

4. Calcular $C = E_{k_1}(m_i)$ y a $t = H_2(C, k_2)$

5. Manda a bob (R, C, t).

En cuanto a Bob, para su desencripción, simplemente ejecuta los siguientes pasos:

1. Calcula a $Z = sR$, usando su llave secreta s.

2. Calcula posteriormente a $H_1(R, Z)$ y escribe su salida como $k_1||k_2$

3. Calcula ahora a $H_2(C, k_2)$, si eso no es exactamente igual a t, entonces la integridad del mensaje se vió comprometida y Bob deja de escuchar.

4. Calcula a $m_i = D_{k_1}(C)$ donde D_{k_1} es la función de desencripción para E_{k_1}

3. ECIES Simplificado

Sin embargo para ésta práctica, ocuparemos el algoritmo de ECIES Simplificado, que sigue este mismo esquema pero ocupa una técnica conocida como punto de compresión y puntos de descompresión, de manera que si tenemos una curva elíptica $E(\mathbb{F}_q)$ y tenemos un punto $(x, y) \in E$, entonces, $PuntoDeCompresion(x, y) = (x, y \ mod \ 2)$, en cuanto a su operación inversa, es un poco más compleja.

Sea $(x, i) \in \mathbb{Z}_q \times \mathbb{Z}_2$, entonces procedemos como sigue:

1. Calcular $z = x^3 + Ax + B \ mod \ q$.

2. Si $\left(\frac{z}{q}\right) \neq 1$, entonces hay un error, en otro caso, ir al siguiente paso.

3. $y = \sqrt{z} \ mod \ q$.

4. Si $y \equiv i \ mod \ 2$, entonces regresamos a (x, y)

5. En otro caso, regresamos a $(x, q - y)$

Como podemos observar, este par de funciones, ya aseguran la integridad que nos pide ECIES, puesto que si al momento de descomprimir el punto que recibimos del mensaje, entonces rechazamos y dejamos de escuchar el mensaje, en otro caso seguimos traduciendo el mensaje, de manera que el algoritmo de encripción quedaría de la siguiente manera:

- Escoger un número aleatorio k, $1 \leq k \leq N - 1$
- Calcular a $R = kA$ y $Z = kB$.
- Calcular el punto de compresión de kA y calcular a $xx_0 \ mod \ q$ donde $kB = (x_0, y_0)$, donde necesariamente $x_0 \not\equiv 0 \ mod q$.
- Enviar a Bob $PuntoDeCompresión(kA), xx_0 \ mod \ q)$.

Observemos que estamos enviando una tupla, cuya primer entrada es otra tupla, de manera que el mensaje recibido por parte de Bob es una lista de elementos de $(\mathbb{Z}_q \times \mathbb{Z}_2) \times \mathbb{Z}_q$.

De manera que el algoritmo de desencripción es el siguiente, supongamos que recibimos una lista de elementos (y_1, y_2), entonces:

- Calcular el punto de descompresión de y_1, lo que nos tiene que dar un número que sea residuo quadrático $mod\ q$ y congruente con la segunda entrada de $y_1\ mod\ 2$, esto nos da un punto sobre la curva elíptica, digamos $D(x, y)$
- Posteriormente, tenemos que calcular $sD = (x_0, y_0)$.
- Calculamos ahora a $y_2(x_0)^{-1}\ mod\ q$ que nos da el texto plano como deseábamos.

4. ECM

La idea de ECM surge a partir de la naturaleza de sumar u puntos o multiplicarlos por un escalar, ¿Qué pasaría si en un dado punto, el inverso multiplicativo que se necesita para realizar la operación, no existe?

Este método fue utilizado por Lenstra, el cual tiene como entrada un número n que se asegura que es compuesto (que no es primo), tomar una curva elíptica en $\mathbb{Z}_n$, un punto en ella y sumarlo con él mismo un número k de veces hasta que eventualmente no tenga un inverso multiplicativo, cuidando de que no sea el mismo número ni 1, puesto que si no no sirve.

Sin embargo, ¿puede ser cualquier curva? La respuesta es no, tiene que cumplir que su determinante sea primo relativo con n, recordando que el determinante de una curva elíptica $E(\mathbb{Z}_n) := y^2 \equiv x^3 + Ax + B\ (mod\ n)$, se calcula como $4A^3 + 27B^2$. *Hint: Pensar en cómo despejando B de la ecuación original, se puede obtener una curva y un punto más fácil.*

De manera que ahora sí, ya existe una curva y un punto de una curva $mod\ n$ y hay que ir sumando ese punto con sigo mismo hasta que ya no tenga un inverso multiplicativo, de manera que necesariamente implica que un factor de n, va a ser calculado en el proceso de sumarlo una vez más, la tarea de la práctica es resolver esto, encontrar en qué punto se encuentra el factor de n, sin embargo, como este no es un método determinista, puede ser que el factor resultante de esta operación sea trivial (1 o n) en cuyo caso, no sirve y se tiene que repetir el proceso desde obtener una curva válida.

5. Implementación

Deberán de programar curvas elípticas primero con las dos operaciones definidas y con ayuda de sus funciones auxiliares anteriores (Algoritmo Extendido de Euclides, primos relativos, test de primalidad...), para implementar el comportamiento correcto de una curva elíptica antes de programar ECIES y deberán asegurarse de que pasen las pruebas unitarias correspondientes para curvas elípticas. Se usará a `None` para determinar a $\mathcal{O}$, el punto infinito.

Una vez acreditadas las pruebas unitarias, deberán implementar ECIES Simplificado que como siempre, es una clase de Python que tiene tres métodos, el constructor, la función de cifrado y la de decifrado, en el constructor deberán de crear la llave, la curva elíptica y todo lo necesario para el algoritmo. Deben de tomar a $n = 8231$, puesto que es el primo más cercano al tamaño de Unicode y representar cada caracter con `ord`.

Por último, para ECM, tendrán un único método que se llama `lenstra(n)`, donde n es el número a factorizar, pueden suponer que siempre es de la forma $n = pq$, de manera que deben obtener a p y a q y regresarlos en forma de tupla.