Programa de Cifrado en Python con múltiples paradigmas matemáticos

C. Rico Echeverry, M. Currea

***Abstract*—A program written in python will be made, whose functions will allow multiple encryption. Implementing in the first instance an encryption algorithm based on RSA to which the Miller-Rabin Primality Test will be applied. With this particular prime generator, several primes of different lengths will be generated in order to explain and see the operation of encryption systems that make use of Fermat's little theorem, the Euclid algorithm that allows the use of modular arithmetic.**

Index Terms: RSA, Encriptacion, Primos, Euclides

I. Introduction

Corresponde indicar, a modo de introducción, que los problemas matemáticos más comunes, suelen tener que ver con formas en las que se desea proteger la información sensible, el cifrado, es por tanto, un estudio matemático, y muchas ramas de las llamadas matemáticas discretas, se ven aplicadas en su finalidad. El reto que se desea afrontar es el de implementar un código en Python, primero para generar números primos con el test de primalidad de Miller-Rabin.

Luego haciendo uso de varias ramas de las matemáticas como: la función phi de Euler, el algoritmo de Euclides, y la aritmética modular se calcularán claves privadas y públicas de un sistema RSA las cuales se usarán para encriptar y desencriptar mensajes sencillos con fines explicativos.

II. Estado del arte

Lo que se pretende hacer no es exactamente nuevo, o revolucionario, aunque si bastante útil, RSA ya cuenta con múltiples implementaciones en Python, y los otros principios matemáticos se han desarrollado a lo largo del curso de Matemáticas Discretas II - 2020-III, sin embargo el que múltiples paradigmas matemáticos se usen para tal finalidad es algo que se desea desarrollar.

A. Pure Python RSA implementation

Python-RSA es una implementación pura de Python RSA. Admite cifrado y descifrado, firma y verificación de firmas y generación de claves de acuerdo con PKCS # 1 versión 1.5. Puede usarse como una biblioteca de Python, así como en la línea de comandos. El código fue escrito principalmente por Sybren A. Stüvel.

.

B. PY Public-key cryptography

La criptografía asimétrica también llamada criptografía de clave pública (en inglés *public-key cryptography*) o criptografía de dos claves (en inglés *two-key cryptography*), es el método criptográfico que usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona que recibirá el mensaje. Una clave es pública y se puede entregar a cualquier persona, la otra clave es privada y el propietario debe guardarla de modo que nadie tenga acceso a ella. Disponible como librería en formato .py

C. Non-linear cryptography

A finales de la década de 1980, Meier y Staffelbach descubrieron por primera vez la importancia de las funciones altamente no lineales en criptografía desde el punto de vista de los ataques de correlación en cifrados de flujo, y más tarde por Nyberg a principios de la década de 1990 después de la introducción del método de criptoanálisis diferencial. Las funciones no lineales perfectas (PN) y no lineales casi perfectas (APN), que tienen las propiedades óptimas para ofrecer resistencia contra el criptoanálisis diferencial, han sido desde entonces objeto de intensos estudios por parte de muchos matemáticos. En este trabajo se analizan algunos de los resultados teóricos obtenidos sobre estas funciones en los últimos 25 años. Recordamos cómo los vínculos con otros conceptos matemáticos han acelerado la búsqueda de funciones PN y APN. Para ilustrar el uso de las funciones PN y APN en la práctica, discutimos ejemplos de cifrados y su resistencia a los ataques diferenciales. En particular, recordamos que en las aplicaciones criptográficas se suelen utilizar funciones subóptimas.

III. Marco Teorico

A. ¿Que es RSA?

RSA (Rivest – Shamir – Adleman) es un criptosistema de clave pública que se utiliza ampliamente para la transmisión segura de datos. También es uno de los más antiguos. El acrónimo RSA proviene de los apellidos de Ron Rivest, Adi Shamir y Leonard Adleman, quienes describieron públicamente el algoritmo en 1977. Un sistema equivalente fue desarrollado en secreto, en 1973 en GCHQ (la agencia británica de

inteligencia de señales), por el matemático inglés Clifford Cocks. Ese sistema fue desclasificado en 1997.

En un criptosistema de clave pública, la clave de cifrado es pública y distinta de la clave de descifrado, que se mantiene secreta (privada). Un usuario de RSA crea y publica una clave pública basada en dos números primos grandes, junto con un valor auxiliar. Los números primos se mantienen en secreto. Cualquier persona puede cifrar los mensajes mediante la clave pública, pero solo puede decodificarlos alguien que conozca los números primos.

La seguridad de RSA se basa en la dificultad práctica de factorizar el producto de dos números primos grandes, el "problema de factorización". Romper el cifrado RSA se conoce como el "problema RSA". Si es tan difícil como el problema de la factorización es una cuestión abierta. No hay métodos publicados para anular el sistema si se utiliza una clave lo suficientemente grande.

RSA es un algoritmo relativamente lento. Debido a esto, no se usa comúnmente para cifrar directamente los datos del usuario. Más a menudo, RSA se utiliza para transmitir claves compartidas para criptografía de clave simétrica, que luego se utilizan para cifrado-descifrado masivo.

El algoritmo RSA implica cuatro pasos: generación de claves, distribución de claves, cifrado y descifrado.

Un principio básico detrás de RSA es la observación de que es práctico encontrar tres números enteros positivos muy grandes e, d y n, tales que la exponenciación modular para todos los enteros m se cumple ($0 \leq m < n$):

$$(m^e)^d \equiv m(mod\ n)$$

y que conociendo e y n, o incluso m, puede ser extremadamente difícil de encontrar d. La barra triple (≡) aquí denota congruencia modular. Además, para algunas operaciones es conveniente que se pueda cambiar el orden de las dos exponenciaciones y que esta relación también implique:

$$(m^d)^e \equiv m(mod\ n)$$

La intención es que los mensajes cifrados con la clave pública solo se puedan descifrar en un período de tiempo razonable utilizando la clave privada. La clave pública está representada por los números enteros n y e; y, la clave privada, por el entero d (aunque n también se usa durante el proceso de descifrado, por lo que también podría considerarse como parte de la clave privada).

B. Test de Primalidad Miller-Rabin

La prueba de primalidad de Miller-Rabin o la prueba de primalidad de Rabin-Miller es una prueba de primalidad: un algoritmo que determina si un número dado es probable que sea primo, similar a la prueba de primalidad de Fermat y la prueba de primalidad de Solovay-Strassen. Gary L. Miller lo descubrió en 1976; La versión de Miller de la prueba es determinista, pero su corrección se basa en la hipótesis ampliada de Riemann que no ha sido probada.

Michael O. Rabin lo modificó para obtener un algoritmo probabilístico incondicional en 1980. A menudo se dice que esta prueba fue descubierta por M.M. Artjuhov en 1967, pero esto es incorrecto: una lectura del artículo de Artjuhov (particularmente su Teorema E) muestra que descubrió el Solovay -Prueba de Strassen, no de Miller-Rabin.

Al igual que las pruebas de Fermat y Solovay-Strassen, la prueba de Miller-Rabin se basa en una igualdad o un conjunto de igualdades que son verdaderas para los valores primos, luego verifica si se cumplen o no para un número que queremos probar para determinar su primalidad.

Primero, un lema sobre raíces cuadradas de unidad en el campo finito $\mathbb{Z}/p\mathbb{Z}$, donde p es primo y $p > 2$. Ciertamente 1 y −1 siempre dan 1 cuando se eleva al cuadrado módulo p; estas son raíces cuadradas triviales de 1. No hay raíces cuadradas no-triviales de 1 módulo p (un caso especial del resultado de que, en un campo, un polinomio no tiene más ceros que su grado). Para mostrar esto, suponga que x es una raíz cuadrada de 1 módulo p. Entonces:

$$x^2 \equiv 1\ (mod\ p)$$
$$(x-1)(x+1) \equiv 0\ (mod\ p)$$

En otras palabras, el primo p divide el producto $(x - 1)\ (x + 1)$. Según el lema de Euclides, divide uno de los factores $x - 1$ o $x + 1$, lo que implica que x es congruente con 1 o con −1 módulo p.

Ahora, sea n primo y $n > 2$. De ello se deduce que $n - 1$ es par y podemos escribirlo como $2s \cdot d$, donde s y d son números enteros positivos y d es impar. Para cada a en $\left(\frac{\mathbb{Z}}{p\mathbb{Z}}\right)*$, ya sea:

$$a^d \equiv 1\ (mod\ n)$$
$$a^{2^r \cdot d} \equiv -1\ (mod\ n)$$

para algunos $0 \leq r \leq s - 1$. Para demostrar que uno de estos debe ser verdadero, recuerde el pequeño teorema de Fermat.

C. Pequeño Teorema de Fermat

Pierre de Fermat fue un abogado francés en el Parlamento de Toulouse, Francia, y un matemático a quien se le atribuye el mérito de los primeros desarrollos que llevaron al cálculo infinitesimal, incluida su técnica de adecuación.

El pequeño teorema de Fermat establece que si p es un número primo, entonces para cualquier número entero a, el número $a^p - a$ es un múltiplo entero de p. En la notación de la aritmética modular, esto se expresa como:

$$a^p \equiv a \ (mod\ p)$$

Por ejemplo, si $a = 2$ y $p = 7$, entonces $2^7 = 128$, y $128 - 2 = 126 = 7 \times 18$ que es un múltiplo entero de 7. Si a no es divisible por p, el pequeño teorema de Fermat es equivalente a la afirmación de que $a^{p-1} - 1$ es un múltiplo entero de p, o en símbolos:

$$a^{p-1} \equiv 1 \ (mod\ p)$$

Por ejemplo, si $a = 2$ y $p = 7$, entonces $2^6 = 64$ y $64 - 1 = 63 = 7 \times 9$ es, por tanto, un múltiplo de 7. El pequeño teorema de Fermat es la base de la prueba de primalidad de Fermat y es uno de los resultados fundamentales de la teoría elemental de números. El teorema lleva el nombre de Pierre de Fermat, quien lo declaró en 1640. Se llama el "pequeño teorema" para distinguirlo del último teorema de Fermat.

*D. **Función φ de Euler***

Es una función importante en teoría de números. Si n es un número entero positivo, entonces $\varphi(n)$ se define como el número de enteros positivos menores o iguales a n y coprimos con n, es decir, formalmente se puede definir como:

$$\varphi(n) = |\{m \in \mathbb{N} | \mathrm{m} \leq \mathrm{n} \ \wedge \ \mathrm{MCD}(\mathrm{m}, \mathrm{n}) = 1\}|$$

donde $|\cdot|$ significa la cardinalidad del conjunto descrito.

La función φ es importante principalmente porque proporciona el tamaño del grupo multiplicativo de enteros módulo n. Más precisamente, $\varphi(n)$ es el orden del grupo de unidades del anillo $\mathbb{Z}/n\mathbb{Z}$. En efecto, junto con el teorema de Lagrange de los posibles tamaños de subgrupos de un grupo, proporciona una demostración del teorema de Euler que dice que $a^{\varphi(n)} \equiv 1 \ (mod\ n)$ para todo a coprimo con n. La función φ juega también un papel clave en la definición del sistema de cifrado RSA.

*E. **Algoritmo de Euclides***

En matemáticas, el algoritmo de Euclides, o algoritmo de Euclides, es un método eficiente para calcular el máximo común divisor (MCD) de dos enteros (números), el número más grande que los divide a ambos sin un resto.

Lleva el nombre del antiguo matemático griego Euclides, quien lo describió por primera vez en sus *Elementos* (c. 300 a. C.). Es un ejemplo de un algoritmo, un procedimiento paso a paso para realizar un cálculo de acuerdo con reglas bien definidas, y es uno de los algoritmos más antiguos de uso común.

Se puede usar para reducir fracciones a su forma más simple y es parte de muchos otros cálculos criptográficos y teóricos de números.

El algoritmo euclidiano procede en una serie de pasos de modo que la salida de cada paso se utiliza como entrada para el siguiente. Sea k un número entero que cuenta los pasos del algoritmo, comenzando con cero. Así, el paso inicial corresponde a $k = 0$, el siguiente paso corresponde a $k = 1$, y así sucesivamente.

Cada paso comienza con dos residuos no negativos r_{k-1} y r_{k-2} Dado que el algoritmo asegura que los residuos disminuyan constantemente con cada paso, r_{k-1} es menor que su predecesor r_{k-2} . El objetivo del k-ésimo paso es encontrar un cociente q_k y un resto r_k que satisfaga la ecuación:

$$r_{k-2} = q_k r_{k-1} + r_k$$

y que tienen $0 \leq rk < r_{k-1}$. En otras palabras, los múltiplos del número menor r_{k-1} se restan del número mayor r_{k-2} hasta que el resto r_k sea menor que r_{k-1}. En el paso inicial ($k = 0$), los restos r_{-2} y r_{-1} son iguales a a y b, los números para los que se busca el MCD. En el siguiente paso ($k = 1$), los restos son iguales a b y el resto r_0 del paso inicial, y así sucesivamente. Por tanto, el algoritmo se puede escribir como una secuencia de ecuaciones:

$$\begin{aligned} a &= q_0 b + r_0 \\ b &= q_1 r_0 + r_1 \\ r_0 &= q_2 r_1 + r_2 \\ r_1 &= q_3 r_2 + r_3 \\ &\vdots \end{aligned}$$

Si a es menor que b, el primer paso del algoritmo intercambia los números. Por ejemplo, si $a < b$, el cociente inicial q_0 es igual a cero y el resto r_0 es a. Por tanto, r_k es más pequeño que su predecesor r_{k-1} para todo $k \geq 0$.

Dado que los restos disminuyen con cada paso, pero nunca son negativos, un resto r_N debe ser eventualmente igual a cero, momento en el que el algoritmo se detiene. El resto final distinto de cero r_{N-1} es el máximo común divisor de a y b. El número N no puede ser infinito porque solo hay un número finito de enteros no negativos entre el resto inicial r_0 y cero

IV. Resultados

Los resultados fueron almacenados en el arreglo y se pueden visualizar en el notebook

```
(p,q,s,clave privada)=( 17 , 419 , 6688 , 1215 )
(p,q,s,clave privada)=( 631 , 929 , 584640 , 179761 )
(p,q,s,clave privada)=( 809 , 229 , 184224 , 80119 )
Clave publica=( 7123 , 4255 )
     Clave publica=( 586199 , 344401 )
    Clave publica=( 185261 , 111943 )
Mensaje= 6000    Mensaje Encriptado= 2787    Mensaje Desencriptado= 6000
        Mensaje= 6000    Mensaje Encriptado= 501966    Mensaje Desencriptado= 6000
       Mensaje= 6000    Mensaje Encriptado= 139151    Mensaje Desencriptado= 6000
```

Los escenarios visualizados son encriptando el mismo mensaje, sin embargo al utilizar la librería random, todos los valores de la clave pública y clave privada cambian y por lo tanto el desarrollo de los algoritmos y la encriptación.

V. Conclusiones

1. El proyecto se presenta como un ejemplo académico con el fin de ilustrar el funcionamiento del algoritmo de encriptación RSA sin embargo puede ser quebrantado “fácilmente” o en tiempos prudentes.

2. Al aumentar el valor de los números primos p y q, mayor a los 200 dígitos, la factorización de n se vuelve imposible con la computación que utilizamos hoy en día.

3. El mismo mensaje con distintas claves se encriptará de maneras distintas.

4. El algoritmo RSA tiene un límite de encriptamiento que corresponde a el valor de n, todo mensaje que supere el valor de n no se encriptará correctamente.

Referencias

Example:

[1] *Lovász, L.; Pelikán, J.; Vesztergombi, K. (2003). Discrete Mathematics: Elementary and Beyond. New York: Springer-Verlag. pp. 100–101. ISBN 0-387-95584-4.*

[2] Trappe, Wade; Washington, Lawrence C. (2002), Introduction to Cryptography with Coding Theory, Prentice-Hall, p. 78.

[3] Musical toothbrush with adjustable neck and mirror, by L.M.R. Brooks. (1992, May 19). *Patent D 326 189*
[Online]. Available: NEXIS Library: LEXPAT File: DESIGN

[4] https://en.wikipedia.org/wiki/Fermat%27s_little_theorem

[5] *Miller, Gary L. (1976), "Riemann's Hypothesis and Tests for Primality", Journal of Computer and System Sciences, **13** (3): 300–317.*

[6] *Rabin, Michael O. (1980), "Probabilistic algorithm for testing primality", Journal of Number Theory, **12** (1): 128–138.*

[7] *https://en.wikipedia.org/wiki/Miller%E2%80%93Rabin_primality_test*