CAPITULO V

En el capítulo II se vieron los criptosistemas simétricos más importantes en el mercado, o sea, los que están en la norma. Entonces, queda pendiente presentar a los criptosistemas asimétricos.

Sin embargo, antes de entrar en materia es necesario mostrar algunos conceptos importantes.

**V.I. Procedimientos de un solo camino.**

Este concepto es toral en el desarrollo de los criptosistemas asimétricos o de llave pública. Veamos una situación práctica de esta característica. Suponga que se tiene dos números primos; 11 y 13, entonces, realizar la multiplicación de ambos se puede llevar a cabo sin dificultad, dando como resultado 143. Por otro lado, si nos proporcionan el número 143 y nos preguntan por los factores primos que dan el resultado anterior, la respuesta no es tan inmediata. Digamos que la factorización es más complicada.

Ahora bien, el problema anterior se llevar a un nivel más complejo, expliquémonos más claramente, suponga que los factores son números primos de cada uno. Entonces, el problema de factorización es mucho muy complicado de resolver con las herramientas que tenemos actualmente.

Por lo tanto, el camino de un solo sentido es aquel que se puede recorrer en una dirección con cierta facilidad; lo que en el caso anterior fue realizar el producto. Sin embargo, la factorización o el sentido opuesto es muy complicado realizarlo, y para factores con valores “grandes” es prácticamente imposible llevarlo a cabo, al menos con las herramientas actuales.

Otro caso que tiene esta característica es el siguiente: suponga que se tiene un elemento generador de un campo finito [1]. Más adelante se verá con detalle lo que significa un elemento generador. También, considere que se tiene un primo ; entonces, dado un entero positivo no es complicado encontrar a un que cumple con: mod. .

Pero por otro lado, si conocemos a; es prácticamente imposible dar con el valor de cuando los números enteros positivos son “grandes”, digamos que son del orden de .

Al problema anterior se le denomina como; el problema del logaritmo discreto [2]. De hecho, si el lector da con un procedimiento de un solo camino, podría ser la base para desarrollar un criptosistema asimétrico.

**V.2. Alta primalidad.**

En la anterior sección se habla de números primos de de manera muy suelta, sin embargo, surge la pregunta ¿cómo podemos obtener un primo de ese tamaño?

En realidad, el problema se resuelve de otro modo; digamos que tenemos un algoritmo con la siguiente característica: una vez que se corre el algoritmo con valores particulares el resultado tiene solamente dos opciones; es compuesto o no es compuesto (es primo). Si el resultado nos dice que el número en cuestión es compuesto, implica que el número es compuesto. Pero si el resultado arroja que no es compuesto, entonces, hay una probabilidad de riesgo, esto es, podría ser el caso de que sea compuesto.

El algoritmo que se describirá más adelante, tiene un error de . Seguramente el lector se dirá que es un error muy grande para inducir que se trata de un primo. Sin embargo, como fue señalado arriba el algoritmo se corre para valores particulares, en realidad es un solo valor particular cada vez; entonces, que tal si en cada turno que se corre el algoritmo se elige al azar un número particular diferente; lo anterior implica que los eventos asociados al correr dicho algoritmo con valores particulares diferentes, son independientes.

Pero si esto es así, implica que los errores se multiplican cuando se desea calcular el error global de una sucesión de eventos.

¡Ah!, pero si el error se calcula de esta manera, resulta ser que cuando se corre el algoritmo 100 veces el error total es ; esto último significa que éste se puede hacer tan pequeño como se quiera.

Resumiendo, se considera que un entero positivo tiene alta primalidad, cuando el algoritmo, que será descrito más tarde, cada vez que se corre para diferentes valores el resultado es: no es compuesto (primo). Si además, este algoritmo se corre con diferentes generadores de números aleatorios, y siempre se obtiene el mismo resultado, esto es, no es compuesto, entonces, podemos considerar para propósitos prácticos que el número propuesto es primo.

**V.2.1. Inversos multiplicativos**

Debido a que el primer criptosistema asimétrico que veremos es; RSA, y que éste utiliza la , phi de Euler. Por otro lado, sabemos que en los campos finitos donde es primo, todos los elementos tal que cumplen con la existencia del inverso multiplicativo; o sea, dado un que cumple con las condiciones anteriores, existe un con , de tal manera que ()() mod. 1.

Sin embargo, que podemos decir del inverso multiplicativo, cuando el número de elementos del conjunto de residuos , es un entero positivo compuesto.

La condición necesaria y suficiente para que un elemento del conjunto tenga inverso multiplicativo es la siguiente: el máximo común divisor de , debe ser 1, lo cual se denota como sigue: mcd (,) = 1.

Ahora bien, conocer el número de elementos que tienen inverso multiplicativo cuando el módulo es compuesto, es algo que Euler resolvió. Por cierto, a este número se le llama la phi de Euler, y el procedimiento para calcularlo es el siguiente:

Cualquier entero positivo se puede expresar como el producto de sus factores primos, lo cual se expresa como: , donde los son factores primos, y los exponentes pueden ser uno o no. Cuando es dos, tres o más se dice que el factor tiene multiplicidad dos, tres etc.

Si denotamos a la phi de Euler para un entero positivo como;, entonces, . Queda claro que cuando , se sigue que .

Otra cuestión por resolver es: cómo calcular el inverso multiplicativo cuando éste existe. En este orden de ideas, primero mostraremos el algoritmo de Euclides [3].

**V.3. Algoritmo de Euclides.**

De hecho, el algoritmo de Euclides nos da el máximo común divisor de dos números; esto es, mcd (,). Dicho algoritmo procede como sigue:

Digamos que se dan dos enteros positivos, , y sin pérdida de generalidad suponga que . Entonces, se lleva a cabo el siguiente proceso:

,

.

(5.3.1)

En la expresión 5.3.1 las constantes son el cociente y las el residuo.

De esta última igualdad, se concluye que divide a . Sin embargo, si esto es así, entonces divide a . Si continuamos razonando de esta manera, llegamos a la conclusión que divide a y . Por lo tanto es un divisor común de y .

Ahora bien, para demostrar que es el máximo común divisor se procede de la siguiente manera: suponga que hay un divisor común a y que es mayor a , denotémoslo como ; pero si divide a y implica que también divide a , sin embargo, si divide a también divide a . Si continuamos con este razonamiento es simple concluir que divide a .

Pero si esto es así, implica que hay una contradicción, ya que un entero positivo no puede ser mayor a otro y además dividirlo.

Por lo tanto, es el máximo común divisor. De aquí, nosotros podemos averiguar si un entero positivo tiene o no inverso multiplicativo, ya que, siguiendo el algoritmo de Euclides solamente se debe verificar que el residuo es 1.

Queda por resolver, cómo se encuentra el inverso multiplicativo una vez que sabemos que existe. En este sentido, a continuación se define un algoritmo que nos ayudará a encontrar este inverso.

Suponga que se definen a; de acuerdo con el siguiente procedimiento:

(5.3.2)

Las constantes del algoritmo 5.3.2 corresponden a las del algoritmo de Euclides. A continuación, se enuncia y demuestra un teorema importante en teoría de números

Teorema 5.3.1.

La relación es verdadera para ; considerando que las se calculan de acuerdo con el algoritmo (5.3.2) y las corresponden al algoritmo de Euclides.

La demostración se llevará a cabo de acuerdo con el método inductivo. Se aclara que el paso base consta a su vez de dos pasos, ya que el algoritmo (5.3.2) empieza con dos valores iniciales.

Entonces, de manera simple se puede saber que cuando , , y que . Recuerde que .

Lo anterior es cierto, por lo que la relación se cumple para el paso base.

Hipótesis inductiva. Suponga que se cumple para los valores de .

Paso inductivo. Con base en la hipótesis anterior, la idea es demostrar que la relación sigue siendo válida para . De hecho, es lo que haremos abajo.

Usando la hipótesis de inducción tenemos que:

(5.3.3) y que,

(5.3.4) .

Por otro lado, de acuerdo con el algoritmo de Euclides el residuo se puede expresar de la forma siguiente:

(5.3.5)

Ahora bien, substituyendo a las expresiones (5.3.3) y (5.3.4) en (5.2.5) obtenemos que;

(5.3.6) .

Extrayendo como factor común a de la expresión (5.3.6), y además, teniendo en cuenta a (5.3.2) obtenemos el siguiente resultado:

.

Entonces, si y se tiene en cuenta el resultado anterior se llega a la siguiente conclusión: . Lo anterior nos dice que, cuando en el algoritmo de Euclides, entonces es el inverso multiplicativo de módulo .

Los autores consideran conveniente presentar un ejemplo, para ilustrar los conceptos mostrados anteriormente.

Ejemplo 5.3.1.- Suponga que y se desea averiguar si los enteros positivos 88 y 92 tienen inversos multiplicativos módulo 143.

En el primer caso se puede comprobar que el mcd (143, 88) = 11. Se sigue que no tiene inverso.

Para el segundo entero no es complicado comprobar que mcd (143, 92) = 1; y que si seguimos el algoritmo de la expresión (5.3.2) nos podemos percatar que 1 y . Por lo tanto, el inverso multiplicativo de 92 módulo 143 es 14, ya que (9214) mod. 143 1.

**V.4. Algoritmo de Miller-Rabin.**

Como fue citado anteriormente la forma de proponer primos es mediante un algoritmo. El nombre de este algoritmo es “The Miller-Rabin primality test”. Este algoritmo tiene un error de cuando el resultado que arroja es: “primo” o no compuesto [4].

Pero cuando el resultado del algoritmo es: compuesto; entonces sin lugar a dudas es compuesto.

Se inicia con la propuesta de un número de 200 dígitos que termine en 1, 3, 7 o 9; llamémoslo . Una vez realizado esto, se realizan los siguientes pasos:

Paso 1. Lleve a cabo el siguiente cálculo: , tal que es impar.

Paso 2. Elija al azar un entero positivo tal que .

Paso 3. Realice el siguiente cálculo: .

Paso 4. If la respuesta “ es primo” y termina, de otra manera continua.

Paso 5. For

If , entonces “ es primo”; de otra manera:

do do

Paso 6. Si finalmente en ninguno de los pasos anteriores se establece que “ es primo”; entonces, el resultado es “ es compuesto”.

A continuación, se desarrolla un ejemplo para valores pequeños, con la intención de aclarar el algoritmo escrito anteriormente.

Ejemplo 5.4.1. Suponga que deseamos averiguar la primalidad de

Paso 1. Teniendo en cuenta que se sigue que ; luego entonces . Entonces, y .

Paso 2. Se elige un entero positivo que cumpla con: . Suponga que

Paso 3. Se realiza el cálculo mod. 143 56.

Paso 4. Como en este caso 56 1, se continua con el proceso.

Paso 5. En este paso la instrucción “For” se realiza solamente una vez, ya que . Sin embargo, como 56 es diferente a -1 o equivalentemente, diferente de 142. Por otro lado, como el proceso se realiza solo una vez, después del cálculo de la instrucción “For” ya no realiza un segundo ciclo.

Paso 6. Como en ninguno de los pasos anteriores se dijo que 143 es primo; entonces, se concluye que es compuesto.

Ahora bien, corramos el algoritmo para otro valor. Considere que . Entonces, los pasos del algoritmo se describen a continuación:

Paso 1. y , por lo que , de donde y .

Paso 2. Suponga que se elige .

Paso 3. Se lleva a cabo el cálculo mod. 127 1.

Paso 4. Como en este caso la respuesta es: “ es primo”.

Se le recuerda al lector que, del resultado anterior **no** se deduce inmediatamente que 127 sea primo, ya que hay un error de 0.25 de que el resultado sea falso. No obstante, el lector puede correr el algoritmo para diferentes valores de y percatarse que el resultado es el mismo.

Claro está, sabemos que 127 es primo. Sin embargo, el algoritmo descrito nos es de gran utilidad para enteros de tamaño .

**V.5. Exponenciación.**

Seguramente nos dimos cuenta que al ejecutar el algoritmo de la sección anterior, surge la dificultad de realizar la operación de exponenciación. Sobre todo, si tanto la base como el exponente son enteros de 200 dígitos.

De hecho, no hay una computadora actualmente que realice la operación modular para enteros del tamaño citado.

Entonces, es claro que debemos recurrir a otro tipo de procedimiento para realizar la operación de exponenciación.

El algoritmo que lleva a cabo esta operación se le llama: “Square-and-Multiply” [5]. Este algoritmo procede como sigue:

Sabemos que un entero puede tener diferentes representaciones, dependiendo de la base con que se trabaje. Digamos, que nosotros estamos acostumbrados a la base 10; pero en general esto no es así.

De hecho, este es el caso en que vamos a recurrir a una base binaria. En otras palabras, expresaremos al exponente de forma binaria, el cual se denota como . Lo anterior se representa en 5.5.1.

(5.5.1) , donde toma los valores 0 o 1.

El valor en la expresión (5.5.1) es la longitud de la cadena, de ceros y unos, que representa el entero .

El algoritmo de la exponenciación se presenta abajo, sin embargo, se hacen las siguientes aclaraciones con relación a la notación: representa el entero que se eleva a una potencia y los son los coeficientes de la representación del entero base dos. La variable es auxiliar.

Paso 1.

Paso 2. For hasta 0

mod. .

do do

If , entonces mod. .

Paso 3. Escriba .

Hagamos un ejemplo con valores particulares.

Ejemplo 5.5.1. Considere que y . Además, que el módulo . No es complicado ver que el exponente en binario es: .

En la tabla 5.5.1 se presenta el procedimiento para obtener a mod. 13211.

Tabla 5.5.1. Resultados parciales de la operación exponenciación.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 12 | 1 | 1 | 8312 |
| 11 | 0 | 9025 |  |
| 10 | 0 | 4810 |  |
| 9 | 0 | 3639 |  |
| 8 | 0 | 4899 |  |
| 7 | 1 |  | 3742 |
| 6 | 1 |  | 5638 |
| 5 | 1 |  | 13210 |
| 4 | 0 | 1 |  |
| 3 | 0 | 1 |  |
| 2 | 0 | 1 |  |
| 1 | 0 | 1 |  |
| 0 | 1 |  | 8312 |

Como puede observarse, si desarrollamos el software para realizar la operación de exponenciación sin tener “overflow”; implicaría que programa debe realizar al menos el cuadrado del dato, esto es, el programa debe ejecutar la operación de mod. 13211. Claro está, lo anterior no es tan obvio si el entero positivo es del tamaño .

**V.6. Teorema Chino del residuo.**

Este teorema es importante en teoría de números y también en Criptografía. De hecho, se lleva cabo un ataque al Criptosistema RSA haciendo uso del Teorema Chino del residuo. Además, se pueden construir permutaciones utilizando esta herramienta.

Este Teorema ofrece una solución única a un sistema de congruencias; esto es, se plantea el siguiente problema:

Sean una colección de enteros positivos que son mutuamente primos relativos; en otras palabras, los factores primos de cada uno de ellos son diferentes entre sí. Lo anterior se expresa como sigue: dados con implica que son primos relativos. Además, considere que se tiene las siguientes constantes enteras: .

De acuerdo con esta información se expresa el siguiente sistema de congruencias:

5.6.1

Para construir la solución única del sistema 5.6.1, primero se define la función ; con .

Esta función se escribe como sigue:

5.6.2.

La función 5.6.2 es uno a uno o inyectiva [6]. Por otro lado, los autores consideran conveniente ilustrar este asunto con un ejemplo.

Ejemplo 5.6.1. Suponga que , se sigue que . Entonces todos los valores de son:

.

Es sencillo darse cuenta que en este ejemplo el valor de , lo cual coincide con el número de valores de .

Es también simple observar en este ejemplo, que todos los puntos de la forma: son diferentes.

Como se vio en el Capítulo II, los criptosistemas simétricos utilizan permutaciones en el proceso de cifrado, ya sea como una permutación aplicada a un bloque de bits o en la generación de cajas.

Por otro lado, se mencionó en el inicio de esta sección que se pueden construir permutaciones usando la función . Ilustraremos este aspecto mediante un ejemplo.

Ejemplo 5.5.2. Suponga que se tiene un sistema de dos módulos, esto es, . En este caso y los puntos son de la forma: . Además, considere que se desea construir una permutación en un arreglo de 12 elementos.

Ahora bien, cuando se escribe un número en el sistema decimal, se sabe que el número más a la derecha representa las unidades, el que le sigue a la izquierda las decenas y así sucesivamente.

En este orden de ideas, representa las unidades y las decenas. Entonces, se pueden escribir los 15 puntos en ese orden; en otras palabras, el primer elemento es y el último de ellos es .

El conteo de los 15 elementos se muestra en la Tabla 5.5.1.

Tabla. 5.5.1. Conteo natural de los puntos.

|  |  |  |  |
| --- | --- | --- | --- |
| Conteo | 0 | 1 | 2 |
| Punto | (0,0) | (0,1) | (0,2) |
| Conteo | 3 | 4 | 5 |
| Punto | (1,0) | (1,1) | (1,2) |
| Conteo | 6 | 7 | 8 |
| Punto | (2,0) | (2,1) | (2,2) |
| Conteo | 9 | 10 | 11 |
| Punto | (3,0) | (3,1) | (3,2) |
| Conteo | 12 | 13 | 14 |
| Punto | (4,0) | (4,1) | (4,2) |

La imagen de la función para los 15 valores de son los siguientes:

.

Por otra parte, como se señaló arriba se desea construir una permutación de en un arreglo de 12 elementos. Por lo tanto, el orden natural de un arreglo de 12 elementos es: 0 1 2 3 4 5 6 7 8 9 10 11.

Teniendo en cuenta esta información, se pueden asociar los puntos de la imagen de con los valores del conteo de la Tabla 5.5.1.

Esto es, el punto de la imagen de tiene asociado el número 0 en la Tabla 5.51. Del mismo modo, el punto tiene asociado el número 4, y el punto el número 8. En este orden de ideas, el último punto de la imagen de es , y a este punto le corresponde el número 14 de acuerdo con la Tabla 5.5.1.

Se propone eliminar del arreglo a los puntos asociados con y . Lo anterior, porque estos puntos están asociados con el orden natural, en otras palabras, son el primero y el último número. Posteriormente, se inicia del número 13 al 1. Esto es, empezamos con 13 el cual tiene asociado el par ordenado (4,1) de acuerdo con la Tabla 5.5.1. Sin embargo, (4,1) tiene asociado el número 4 de acuerdo con la función Del mismo modo, 12 tiene asociado el par ordenado (4,0); y éste a su vez le corresponde el número 9 de acuerdo con la función

Entonces, el arreglo de números es el siguiente: 4 9 8 13 3 2 7 12 11 1 6 5 10.

Sin embargo, deseamos iniciar desde 0; por lo tanto, se debe restar 1 a cada uno de los elementos del arreglo anterior.

El arreglo resultante es: 3 8 7 12 2 1 6 11 10 0 5 4 9. Ahora bien, como lo que se solicita es una permutación sobre un arreglo de 12 elementos, entonces, se debe eliminar el número 12 de la cadena anterior. Por lo que finalmente obtenemos el siguiente resultado:

3 8 7 2 1 6 11 10 0 5 4 9.

Claramente, la cadena de arriba es una permutación de 12 elementos. Algunas reflexiones al respecto.

1. Se eliminan siempre el primero y último elemento, ya que estos están en orden.
2. Se eligen de tal forma que el producto de ellos sea mayor a la longitud del arreglo de elementos que se desea permutar. En general se busca que el producto no sea mucho mayor al tamaño del arreglo.
3. Los elementos que sobren se eliminan.

**V.7. El Teorema de Lagrange y el pequeño Fermat.**

Hay un resultado en Algebra Abstracta que es importante para el criptosistema RSA. Dicho resultado es el Teorema de LaGrange [7].

Se hacen algunas consideraciones previas. Suponga que se tiene un conjunto finito de elementos que define un grupo . En este conjunto está definida la operación de multiplicación. De aquí, se dice que un elemento tiene “orden” si . Considerando que es el entero positivo más pequeño que hace cumplir la anterior equivalencia [8]. Con esta información se enuncia el Teorema 5.6.1 conocido como el teorema de Lagrange.

Teorema 5.7.1. Si es un grupo multiplicativo de orden y . Entonces, el orden de divide a .

En otras palabras, divide o también se escribe como: .

Como una consecuencia del Teorema 5.7.1 se escribe el siguiente corolario:

Corolario 5.7.1. Si con , entonces

Recordemos, que es el número de números que tienen inverso multiplicativo módulo .

Hay otro teorema―el pequeño Fermat― que también es importante por sus aplicaciones. Dicho resultado establece lo siguiente:

Teorema 5.7.2. Dado un número primo y un entero positivo , entonces, se cumple que . También, de manera equivalente se cumple que .

En el caso del teorema 5.7.2 el módulo es primo y por lo tanto, hay inversos, de hecho, todos los enteros que cumplen con , tienen inverso. Veamos un ejemplo donde se calculen todas las exponenciaciones del número ; tomando como módulo al primer primo de dos dígitos, en otras palabras, .

Ejemplo 5.7.1. Dado el entero calcule todas exponenciaciones que van de 1 hasta 10.

.

Claramente, el último valor del ejemplo anterior confirma la afirmación del Teorema 5.7.1. Además, el número 6 genera todos los residuos del primo 11, excepto el cero.

**V.8. Criptosistema RSA.**

Una vez que se vieron los conceptos de las secciones anteriores, estamos listos para entender al criptosistema RSA.

Se inicia con dos números primos y a partir de ellos se calcula el entero . De acuerdo con Euler, el número de números menores a que tienen inverso es . Es simple darse cuenta que es un entero par; por lo tanto no todos los números menores a diferentes de cero tienen inverso. Sin embargo, el criptosistema RSA propone buscar un entero tal que mcd . En realidad, hay un gran número de enteros que cumplen con la condición anterior, sin embargo, se debe aclarar que el entero **no** debe ser par. El algoritmo de Euclides es la herramienta que se utiliza para localizar al entero .

Una vez que se encuentra al entero , usando el algoritmo para el cálculo del inverso que se vio en la sección V.3, se obtiene el entero tal que

En el esquema RSA, al entero se le llama llave privada y al número llave pública. El conjunto de residuos y de la misma manera el conjunto de residuos . En este sentido, y el texto plano con .

Se aclara que, los textos planos son enteros positivos y no letras de algún alfabeto. De hecho, aunque es posible cifrar mensajes utilizando algún alfabeto en la práctica no se usa RSA de esta manera.

En realidad, RSA se emplea para distribuir la llave de un criptosistema simétrico. Queda claro que, la llave de un criptosistema simétrico es una cadena de ceros y unos, la cual a su vez se le puede asociar un entero positivo.

Ahora bien, con esta información el procedimiento de encriptación se define como sigue:

Dados el cifrado de un texto plano es .

En la expresión anterior la variable es el texto encriptado.

Por otro lado, dado el texto encriptado ; éste se descifra de la siguiente manera:

La forma de mostrar que es sencilla, a continuación se demuestra el punto:

, pero por otra parte, debido a que son inversos multiplicativos módulo .

Entonces, , sin embargo, de la expresión anterior se sigue que: ; ya que, de acuerdo con el Corolario 5.8.1 .

Entonces, de manera esquemática el criptosistema RSA se expresa como sigue:

Los conjuntos de texto plano y cifrado son: .

El conjunto de las llaves está dado por: .

Resumiendo. Dada un elemento , las funciones de cifrado y descifrado son las siguientes:

5.8.1. . Función de cifrado

5.8.2. . Función de descifrado.

Se aclara que los valores son públicos y los enteros positivos son privados.

Un ejemplo es muy conveniente para ilustrar el proceso de encriptación y descifrado del criptosistema RSA.

Ejemplo 5.8.1.- Suponga que ; se sigue que y . Además, considere que y que

Utilizando el algoritmo de la subsección V.3 se obtiene el inverso multiplicativo módulo 17388. Este inverso es: , el lector puede comprobarlo realizando las operaciones directamente.

Entonces, el cifrado de se lleva a cabo de acuerdo con: .

Por otro lado, el descifrado es: como nosotros lo esperábamos.

Algunas reflexiones. Los ataques a este criptosistema se basan en la factorización del entero [9]. Queda claro que si se logra factorizar , como es pública y el inverso es único, entonces se puede conocer , y con ello el texto plano .

Cuando se propone la llave pública y a partir de ésta se desea encontrar la llave privada, se señala que no siempre el primer valor de tendrá inverso multiplicativo módulo .

En este orden de ideas, debido a que es un número par se debe proponer un número impar, sin embargo, esto no garantiza que tenga inverso multiplicativo, pero si este es el caso, se elige el entero más dos; esto es, . Si aún éste no tiene inverso, se propone a: . En la práctica el entero que si tiene inverso multiplicativo módulo no está muy lejos de .

La administración de la comunicación segura es de tipo jerárquico [9], en otras palabras, las personas que intervienen en el esquema no conocen a ; excepto la persona de mayor jerarquía. Las demás personas, solamente conocen su clave privada, a y las claves públicas de las otras personas que intervienen en la comunicación segura.

El remitente usa la llave pública del destinatario para enviar el mensaje. El destinatario utiliza su clave privada para descifrarlo.

Los mensajes son encriptados con un sistema simétrico, porque son seguros y rápidos. Sin embargo, la llave del sistema simétrico tiene asociado un entero positivo y éste es el que se cifra para enviárselo al destinatario [10].

Entonces, en estricto un esquema de comunicación segura debe incluir un sistema simétrico y un asimétrico.

Ejercicios

1.- Construya un número de 11 dígitos que tenga terminación en 1, 3 7 o 9. Aplique el algoritmo de primalidad hasta 10 veces en caso de que el resultado sea es primo. Si el resultado indica que es compuesto, entonces proponga otro número con las características anteriores.

2.- Dado el módulo , encuentre los inversos multiplicativos en caso de que existan de los siguientes números: 11007, 13422 y 15873.

3.- Realice los siguientes cálculos de exponenciación:,

4.- Desarrolle un programa para realizar la operación de exponenciación con números de 50 dígitos y presente un ejemplo.

5.- Usando el Teorema Chino del residuo, encuentre una permutación de 16 elementos diferentes. Por otro lado, una vez construida la permutación encuentre su inversa.

6.- Dados los valores de en un esquema RSA, encuentre los siguientes elementos: .

7.- Utilizando los valores de del problema 6, proponga un valor privado y un valor público ; además, considere el caso hipotético de que la función Has Sha de un mensaje es 53. Con esta información, lleve a cabo la firma digital del mensaje.

Bibliografía

[1] Gutiérrez J, Tena J, “Protocolos criptográficos y seguridad en redes”, Universidad de Cantabria, ISBN 84-8102-345-0, 2003.

[2] Hecht Juan Pedro, “A Zero―Knowledge Authentication Protocol using Non Commutative Groups”, Universidad de Buenos Aires, doi: 10.13140/RG.2.1.3734.7364, 2011.

[3] Fenghua Wang, Zhitao Huang, Yiyu Zhou, “A Method for Blind Recognition of Convolution Code Based on Euclidean Algorithm”, [Wireless Communications, Networking and Mobile Computing](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4339774), doi: [10.1109/WICOM.2007.358](https://doi.org/10.1109/WICOM.2007.358), 2007.

[4] Arnault F., “Rabin-Miller primality test: composite numbers which pass it”, Mathematics of Computation, doi: 10.1090/S0025-5718-1995-1260124-2, 1995.

[5] Rebeiro Chester, Mukhopadhyay Debdeep, Bhattacharya Sarani, “Timing Channels in Cryptography: A Micro-Architectural Perspective”, Springer, doi: 10.1007/978-3-319-12370-7, 2015.

[6] Rosen K., “Discrete Mathematics and its Applications”, Mc Graw Hill, fifth edition, ISBN 0-07-242434-6, 2003.

[7] Herstein I. N., “Álgebra Abstracta”, Grupo Editorial iberoamericano, ISBN 02-353820-1, 1988.

[8] Gallian J., “Contemporary abstract algebra”, seventh edition, Brooks/Cole, 2011.

[9] Douglas R. Stinson, “CRYPTOGRAPHY: Theory and practice”, Chapman & Hall/ CRC Press, 2006.

[10] Víctor Manuel Silva-García, Rolando Flores-Carapia, Carlos Rentería Márquez, Benjamín Luna-Benoso, Cesar Antonio Jiménez-Vázquez, Cipher image damage and decisions in real time, Journal of Electronic Imaging, SPIE, doi: 10.1117/1.JEI.24.1.013012, 2015.