Códigos y Criptografía - Curso 2018-2019

Práctica 5: Intercambio de claves.

Protocolo de Diffie y Hellman

- Usaremos la función de las prácticas anteriores **letranumero(texto)**.
- Como en esta práctica, y en posteriores, se le va a pedir a Matlab que haga operaciones con cifras muy elevadas, debemos tener cuidado de trabajar con enteros cuando sea necesario para que no aproxime, uint64. Asimismo, siempre que las operaciones sean modulares debemos hacerlas modulares de modo individual para que no se sature.

1.- Función pote=potencia (c, d, n)

Función que usa el algoritmo de potenciación rápida para calcular las potencias modulares.

Entradas:

c: base de la potencia. Un número natural.

d: exponente de la potencia. Un número natural.

n: módulo. Un número natural.

Salida: *pote* es la potencia c^d módulo *n*.

Ejemplo

```
>> pote = potencia(34237778,38472317, 101010331)

pote = 25000315
```

2.- Función generador = genera_0 (g, p)

Función que indica si el número natural g es generador del grupo multiplicativo determinado por el primo p. Para ello utilizaremos la definición de generador.

Entradas:

g: número natural de Z_p, a decidir si es generador.

p: número primo que determina el grupo multiplicativo.

Salida:

generador=0: en caso de que no sea generador o las entradas no sean correctas.

generador=g: en caso de que sea generador.

También nos debe mostrar el tiempo que tarda en decidir si el elemento es generador.

Ejemplo

>> generador = genera_0(7,100003)

Elapsed time is 50.897518 seconds.

generador =7

>> generador = genera_0(18,100003)

Elapsed time is 50.555716 seconds.

generador = 0

3.- Función generador = genera (g, p)

Función que indica si el número natural g es generador del grupo multiplicativo determinado por el primo p. Para ello utilizaremos la definición de generador.

Entradas:

g: número natural de Z_p, a decidir si es generador.

p: número primo que determina el grupo multiplicativo.

Salida:

generador=0: en caso de que no sea generador o las entradas no sean correctas.

generador=g: en caso de que sea generador.

También nos debe mostrar el tiempo que tarda en decidir si el elemento es generador.

Ejemplo

>> generador = genera(15,1234542)

el número p debe de ser primo y g menor que p y mayor que cero

Elapsed time is 0.000416 seconds.

generador = 0

>> generador = genera(12,1234547)

g no es generador

Elapsed time is 0.009601 seconds.

generador =0

>> generador = genera(15,1234547)

Elapsed time is 0.007678 seconds.

generador = 15

4.- Crear un programa para el intercambio de claves de Diffie y Hellman

Se deben proporcionar los datos comunes a A y a B: el valor del primo y el valor del generador, el programa debe comprobar que se cumplen estas condiciones.

El programa generará tanto para A como para B sus respectivos números aleatorios, nos indicará cuáles son las potencias que se envían, y como tanto A como B obtienen la clave común a partir de los valores obtenidos por el contrario.

Ejemplo

Nos ponemos de acuerdo en g= 33 y p=1999

A genera un número aleatorio aa=557 entre 2 y 1997.
A envía a B: pote_a=mod(g^aa,p)= mod(33 ^557,1999)=1185

B genera un número aleatorio bb=1093 entre 2 y 1997.
B envía a A: pote_b=mod(g^bb,p)=mod(33 ^1093,1999)=1448

A y B calculan la clave con la que van a cifrar sus mensajes
la clave que obtiene A es clave= potencia(pote_b,aa,p)= potencia(1448,557,1999)=946
la clave que obtiene B es clave= potencia(pote_a,bb,p)=potencia(1185,1093,1999)=946

5.- Crear un programa que muestre la vulnerabilidad del intercambio de claves

Se deben proporcionar los datos comunes a A y a B: el valor del primo y el valor del generador, el programa debe comprobar que se cumplen estas condiciones.

El programa generará tanto para A como para B sus respectivos números aleatorios, nos indicará cuáles son las potencias que envían e intercepta C. Generará para el supuesto espía C otro número aleatorio, y nos mostrará cómo A y C obtienen una clave común (aquí A piensa que C es B) y , y como B y C obtienen la otra clave común (aquí B pensará que C es A).

Ejemplo

Nos ponemos de acuerdo en g= 33 y p=1999

A genera un número aleatorio aa=1599, entre 2 y 1997

A envía a B: pote_a=mod(g^aa,p)= mod(33 ^1599,1999)=789

EL INTRUSO CAPTURA pote_a =789, y lo guarda

EL INTRUSO GENERA cc=284, entre 2 y 1997

EL INTRUSO ENVIA a B: pote_a=mod(g^cc,p)= mod(33 ^284,1999)=225

(B piensa que le llega de A)

B genera un número aleatorio bb=843, entre 2 y 1997

B envía a A: pote_a=mod(g^bb,p)=mod(33 ^843,1999)=1874

EL INTRUSO CAPTURA pote_b =1874, y lo guarda

EL INTRUSO ENVIA a A el mismo dato que le ha enviado a B: pote_c= 225

A piensa que le llega de B

A y B calculan sus claves con los datos que tienen (observa que son distintas pero A y B no lo saben

A cifrará con potencia(pote_c,aa,p)=204

B cifrará con potencia(pote_c,bb,p)=738

EL INTRUSO CALCULA TAMBIEN LAS CLAVES

sabe que A cifrará con potencia(pote_c,aa,p)=204

sabe que B cifrará con potencia(pote_c,bb,p)=738