Documento Informe Caso 3

Tabla de contenidos

1. Descripción de la Organización de Archivos en el ZIP

1.1 Estructura de carpetas y archivos

- **src**: Contiene el código fuente del proyecto.
  - **utilidades**: Carpeta dentro de src con utilidades adicionales (cifrado y descifrado).

- **keys**: Carpeta que contiene los archivos de llaves del servidor.

1.2 Descripción de los archivos clave

- *llavePrivada.ser*: Archivo con la llave privada del servidor.
- *llavePublic.ser*: Archivo con la llave pública del servidor.

- *Cliente.java*: Implementación del cliente que consulta el estado de los paquetes.
- *HiloCliente.java*: Clase para manejar el procesamiento concurrente de las consultas de clientes.
- *Paquete.java*: Clase que representa la entidad de los paquetes. - *Servidor.java*: Implementación del servidor que responde a las consultas.
- *Usuario.java*: Clase que representa a los usuarios que consultan los paquetes.
- *CifradoHelper.java*: Clase auxiliar para gestionar el cifrado y descifrado.

2. Instrucciones para Correr el Servidor y el Cliente

2.1 Instrucciones para iniciar el servidor

i. Compilar los archivos

ii. Generar las llaves asimétricas

- Ejecuta el archivo Servidor.java y selecciona la opción **1** en el menú para generar las llaves RSA necesarias. Esto generará dos archivos: publicKey.ser y privateKey.ser.

- Se generan las llaves RSA que pueden ser usadas para autenticación o cifrado en futuras conexiones.

iii. Iniciar el servidor

- Selecciona la opción **2** en el menú para cargar los datos predefinidos (32 usuarios y sus paquetes) e iniciar el servidor en el puerto 12345.
- El servidor queda esperando conexiones de clientes en ese puerto.

2.2 Instrucciones para iniciar el cliente

Para ejecutar el cliente, se ingresa dos argumentos: idUsuario e idPaquete. Por ejemplo: *java src.Cliente user1 pkg1*

El cliente realiza los siguientes pasos:

1. **Genera un par de claves Diffie-Hellman (DH)** y envía su clave pública al servidor para establecer una clave secreta compartida.
2. **Recibe la clave pública del servidor** y genera una clave AES a partir de la clave compartida DH.
3. **Envia el idUsuario y el idPaquete** al servidor para consultar el estado del paquete.
4. **Recibe la respuesta cifrada** del servidor, la descifra usando la clave AES compartida, y muestra el estado del paquete en la consola.

Para probar la concurrencia con un número específico de clientes, se pueden lanzar múltiples instancias del cliente en paralelo. No hay un límite explícito en el código, se podría controlar la cantidad de clientes concurrentes que se desea ejecutar lanzando múltiples hilos de clientes al mismo tiempo desde una aplicación o script de pruebas, o ejecutando varias veces el programa `Cliente`.

Si se quiere implementar un límite en el número de clientes concurrentes, sería necesario modificar la clase `Servidor` para que controle el número de hilos que se pueden ejecutar a la vez.

3. Tablas de Datos

- 3.1 Formato y descripción de los datos recolectados
    - Escenario 1. Un servidor y un cliente iterativos. El cliente genera 32 consultas.

Consulta	Tiempo para generar G, P y Gx (ns)	Tiempo para responder el reto (ns)	Tiempo para verificar la consulta (ns)
1	222840300	37029200	386100
2	3640000	4345700	305400
3	3413800	6686480	1232200

4	4055500	6390800	204700
5	9221300	7540000	191000
6	3597900	4406800	287200
7	4992700	4782900	272700
8	5045600	4245700	217000
9	3849900	3741600	119800
10	6229600	3685000	175100
11	3541600	2901200	131100
12	2548800	3184000	212600
13	2772800	2536200	177000
14	2485000	2442800	157200
15	3252400	2652300	210800
16	4382400	4708200	235000
17	1984100	2798200	151000
18	1953900	10410600	93500
19	1829100	1730600	186400
20	1981900	3505600	1089800
21	1843100	3223000	117600
22	1416000	1631700	88700
23	2842000	2352500	103200
24	1223800	1399100	73200
25	2800600	1610000	109300
26	1347100	1296200	66800
27	2072300	1787800	126400
28	1271500	1939900	80700
29	1626600	1601800	295400
30	1472400	1544000	81700
31	1534900	2281000	90000
32	1480100	2251800	192300

- Escenario 2. Un servidor y un cliente que implementen delegados.
  - 4 delegados

Tiempo en responder el reto (ns)	Tiempo en generar G,P,Gx (ns)	Tiempo en verificar la consulta (ns)
143927700	1042000	19810800
146072600	961200	20353900
145582800	1954200	20577600
143740500	2218300	22679900

  - 8 delegados

Tiempo en responder el reto (ns)	Tiempo en generar G,P,Gx (ns)	Tiempo en verificar la consulta (ns)
181086000	470717400	84895500
187054300	461730200	98814300
107620300	474304400	97885400
263512800	473268300	97149500
218447700	474612300	112290500
234033300	466671900	110501400
209300300	463362900	99721800
204859900	565192500	105207300

- 32 delegados

Tiempo en responder el reto (ns)	Tiempo en generar G,P,Gx (ns)	Tiempo en verificar la consulta (ns)
20595300	199978900	224400
5978900	5578100	159700
4904500	8412400	263300
22177200	17495700	227600
5608600	8984000	117900
5462300	4252900	510900
4187900	6891200	200000
4018000	2651400	170100
9902700	4366800	186300
4946600	3123100	163200
10439900	4415100	157700
2951300	7926900	289200
5937000	3104500	139300
6531300	2370500	158500
3928800	4436000	154600

4242400	4543300	193200
6531300	2370500	158500
3928800	4436000	154600
3928800	4543300	

4. Graficas.

5. Respuesta preguntas planteadas y conclusiones

5.1 Identifique la velocidad de su procesador, y estime cuántas operaciones de cifrado puede realizar su máquina por segundo, en el caso evaluado de cifrado simétrico y cifrado asimétrico. Escriba todos sus cálculos.

a. Cálculos para Cifrado Simétrico (AES)

Para AES, estimamos que un procesador de 1 GHz realiza aproximadamente **50 millones de operaciones por segundo**. Así que, para 4.4 GHz:

AES por segundo=4.4GHz×50millones=220millones de operaciones AES por segundo

b. Cálculos para Cifrado Asimétrico (RSA)

Para RSA de 1024 bits, estimamos que un procesador de 1 GHz realiza aproximadamente **100,000 operaciones RSA por segundo**. Para 4.4 GHz:

RSA por segundo=4.4GHz×100,000=440,000operaciones RSA por segundo

.