Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA DE SEGOVIA

Grado en Ingeniería Informática de Servicios y Aplicaciones

Protocolos criptográficos y seguridad en la nube: estudio teórico-práctico

---

Alumna: Alicia Martínez Mendoza

Tutor: José Ignacio Farrán Martín

Índice general

Índice de figuras

Índice de tablas

Resumen

El almacenamiento en la nube es cada vez más utilizado en todo tipo de ámbitos. A pesar de ello, aún existe cierta desconfianza debido a la preocupación por la privacidad de los datos. Teniendo en cuenta que se almacenan grandes cantidades de información de todo tipo, es imprescindible disponer de las medidas de seguridad necesarias para poder protegerla, en especial si se trata de datos valiosos o sensibles.

Almacenar la información de forma segura es necesario, sin embargo, el problema aparece cuando se desean realizar operaciones sobre esta información al mismo tiempo que se preserva la privacidad. El principal objetivo es dar utilidad a los datos almacenados, sin que personas no autorizadas logren obtener información privada.

En este trabajo, nos centraremos en la computación multiparte segura, que propone una solución para poder trabajar y realizar operaciones con datos privados sin revelar más información de la necesaria.

Palabras clave: seguridad en la nube, computación multiparte segura, evaluación segura de funciones.

Parte I

Memoria del Proyecto

Capítulo 1

Introducción

Con el crecimiento del almacenamiento en la nube, aparecen nuevos desafíos. El principal reto es la seguridad. Además de la importancia de preservar la privacidad información almacenada, se necesita seguir protegiendo la información cuando se trabaja con ella. Para que los datos realmente resulten útiles, debemos poder realizar operaciones sobre ellos.

Para dar una solución a este problema, aparece la computación multiparte segura, que permite que varios usuarios trabajen con información compartida, pero sin revelar a los demás la información que posee cada uno.

A lo largo de este trabajo, veremos qué es la computación multiparte segura, y nos centraremos en varias operaciones básicas con diversas aplicaciones. Este es un trabajo de investigación, y las operaciones implementadas sirven para comprender y comprobar el funcionamiento de la teoría, no se trata de un proyecto de sotfware. Las operaciones que veremos se basan en la compartición de secretos, que también explicaremos, ya que es una herramienta muy útil para la computación multiparte.

1.1. Objetivos

A continuación, se listan los objetivos principales de este trabajo:

OB-1 Estudiar la bibliografía relacionada con la computación multiparte segura y los métodos de preservar la privacidad de la información a la hora de realizar operaciones sobre ella.

Dentro de este objetivo podemos ver varios subobjetivos, de modo que en relación a la computación multiparte debemos estudiar:

OB-1.1 Su origen.

OB-1.2 Consideraciones generales sobre la seguridad de los protocolos.

OB-1.3 Aplicaciones.

OB-2 Estudiar las principales herramientas o maneras de abordar la computación multiparte:

OB-2.1 Circuitos cifrados de Yao.

OB-2.2 Cifrado homomórfico.

OB-2.3 Esquema de Shamir para compartir secretos.

De los subobjetivos anteriores, nos centraremos en el OB-2.3, ya que se utilizará el esquema de Shamir para realizar las operaciones del objetivo OB-3.

OB-3 Estudiar varias operaciones realizadas utilizando computación multiparte, en concreto:

OB-3.1 Suma de secretos

OB-3.2 Combinación lineal de valores secretos

OB-3.3 Media de varios valores secretos

OB-3.4 Multiplicación de secretos

OB-3.5 Posición de un elemento dentro de un conjunto de valores secretos

OB-3.6 Mediana y cuartiles a partir de conjuntos de valores secretos

OB-4 Realizar una implementación de las operaciones anteriores utilizando el lenguaje Python. Esta implementación deberá mostrar el correcto funcionamiento de las operaciones.

1.2. Organización del documento

En primer lugar, el el capítulo 2, se expone la metodología de trabajo que se ha seguido en el desarrollo de este proyecto, mencionando también las herramientas utilizadas. El resto del documento describe los conceptos que nos interesan en este trabajo (listados en los objetivos anteriores) de la siguiente manera:

En los capítulos 3 y 4, se incluye una breve introducción al almacenamiento en la nube, junto a los conceptos de seguridad de la información y privacidad, y una introducción a la criptografía, explicando la criptografía simétrica y asimétrica.

A continuación, en el capítulo 5, se introduce el concepto de computación multiparte segura, indicando su origen y principales aplicaciones. En el capítulo 6 se explican los

principales conceptos de seguridad, y en el capítulo 7 se incluyen las principales formas de abordar la computación multiparte (de estas, la que más nos interesa es el esquema de Shamir).

Posteriormente, se incluyen explicaciones con ejemplos de las operaciones enumeradas en el apartado de objetivos (1.1). Estas operaciones se incluyen dentro del capítulo 8, y ha sido necesario comprender su funcionamiento para implementarlas en Python.

Para finalizar, se muestra el funcionamiento de los programas realizados en Python para las operaciones anteriores. Esta sección se encuentra en el anexo B. En el anexo A se describen los entregables del proyecto, formados por las carpetas con los programas escritos en Python.

Capítulo 2

Metodología de trabajo

En esta sección se especifica la metodología de trabajo que se ha seguido durante el desarrollo del proyecto, así como su ciclo de vida, la planificación temporal y las herramientas utilizadas.

Respecto a la metodología de trabajo, se ha seguido una metodología clásica. Se ha considerado la más adecuada ya que se deberán seguir etapas definidas durante el desarrollo del proyecto. Estas etapas se describen en el siguiente apartado (2.1).

En concreto, se ha optado por una metodología incremental, debido a la importancia de completar unas etapas antes de continuar con las siguientes, realizándolas de una manera escalonada. Por ejemplo, como veremos a continuación, es importante comprender el funcionamiento de una operación para poder realizar su implementación.

2.1. Ciclo de vida del proyecto

El desarrollo de este trabajo puede dividirse en las siguientes etapas, a lo largo de las cuales se va redactando el contenido de la memoria:

E1 Primera etapa, en la que se estudia acerca del tema general de computación multiparte.

Esta etapa consiste en el primer contacto con el concepto de computación multiparte, por lo que el objetivo principal se basa en leer acerca de este tema y conseguir una visión general sobre la que se profundizará a lo largo de las siguientes etapas.

E2 Segunda etapa, en la que se comienzan a estudiar más en profundidad algunos conceptos relacionados con la computación multiparte, en concreto, estudiamos lo

relativo a los objetivos OB-1 y OB-2. Aunque la memoria se irá redanctando a lo largo todas las etapas, esta segunda etapa será la que esté más vinculada con la realización de la misma. De esta manera, se van reflejando en la memoria los conocimientos adquiridos durante el desarrollo de esta etapa.

Además, hemos decidido centrarnos en varias operaciones sencillas que serán implementadas en Python. Para cada una de las operaciones elegidas, se realizan las etapas siguientes:

E3 Tercera etapa, en la que se estudia la operación elegida y se realizan ejemplos para comprender su funcionamiento.

Con el objetivo de realizar la implementación en Python de las operaciones, el primer paso consiste en comprender el funcionamiento de las mismas. Por ello, en la memoria se incluyen, además de las explicaciones correspondientes, ejemplos que muestran la forma de proceder al realizar estas operaciones sobre unos valores concretos.

E4 Cuarta etapa, en la que se realiza una implementación en Python de la operación elegida.

Esta implementación ayudará a mostrar el funcionamiento de la computación multiparte aplicada a operaciones concretas.

E5 Quinta etapa, en la que se realizan correcciones sobre la etapa anterior.

Cada implementación es revisada por el tutor para poder corregir los posibles errores o realizar mejoras sobre la implementación de las operaciones realizadas en Python.

2.2. Planificación temporal

Teniendo en cuenta los objetivos planteados para el proyecto, se ha establecido la división de las etapas del apartado anterior, que consituyen las tareas principales a realizar.

Puesto que en la segunda etapa incluimos la realización de la memoria, esta etapa implica un mayor esfuerzo en comparación con las demás. De hecho, esta etapa se realizará de forma paralela a las etapas 3, 4 y 5, que se realizan de forma iterativa para cada una de las operaciones elegidas. La realización de estas operaciones se ha dividido en tres bloques:

- **Bloque 1:** Suma, media y combinación lineal.
- **Bloque 2:** Mediana y cuartiles.
- **Bloque 3:** Multiplicación.

Se ha realizado la agrupación de las operaciones de esta manera por la relación que existe entre ellas. Por ejemplo, las operaciones del bloque 1 no dejan de ser el caso general de la operación de combinación lineal junto a dos casos particulares que son la suma y la media. Además, se deben realizar en primer lugar las operaciones del bloque 1, ya que estas se reutilizan en los siguientes bloques.

El orden de las etapas se planifica siguiendo el diagrama de la figura 2.1. En este diagrama se pretende ofrecer una visión general del orden de las etapas, donde es especialmente relevante el orden de las etapas 3, 4 y 5, como se ha mencionado anteriormente.

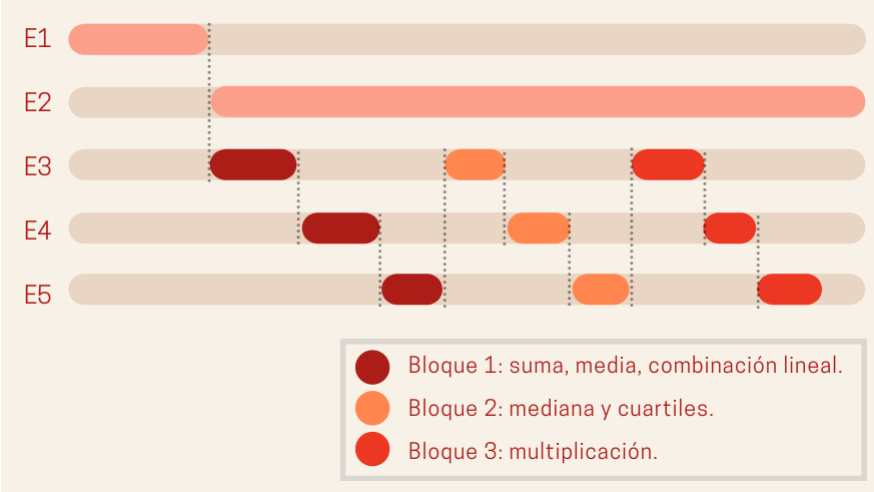

Figura 2.1: Visión general de la duración y el orden de las etapas del proyecto.

2.3. Herramientas utilizadas

Las herramientas que se han utilizado en la realización de este proyecto son las siguientes:

- **Spyder:** esta herramienta proporciona un entorno de desarrollo para trabajar con el lenguaje de programación **Python**. Además, Spyder se integra con paquetes como NumPy, SymPy o pandas, que utilizaremos a la hora de implementar en Python las operaciones elegidas.
- **Jupyter:** este entorno de desarrollo basado en la web se utilizará para obtener la salida de los programas realizados en Python. Ofrece la ventaja de ser multiplataforma, ya que se accede a él a través de un navegador web y, además, permite exportar la salida de un cuaderno Jupyter en formato LaTeX.
- **Overleaf:** se ha elegido este editor de LaTeX porque presenta dos ventajas notables: permite trabajar desde cualquier dispositivo, accediendo a él desde el navegador, y

ofrece almacenamiento en la nube. Esta herramienta se ha utilizado para crear este documento.

Capítulo 3

Almacenamiento en la nube

El almacenamiento en la nube proporciona una forma de acceder a la información desde cualquier lugar y utilizando cualquier dispositivo. En lugar de almacenar los archivos en un dispositivo local, estos se almacenan en un conjunto de servidores a los que se accede a través de Internet.

Además de la ventaja de poder acceder a los datos desde cualquier lugar, el almacenamiento en la nube facilita la realización de tareas colaborativas con la información almacenada. Tenemos los ejemplos de Google Drive o Dropbox, que ofrecen almacenamiento en la nube y, además, permiten que varios usuarios participen en la edición de un mismo documento.

La mayor preocupación que existe hoy en día respecto al almacenamiento en la nube es la privacidad de la información. Los archivos se almacenan en dispositivos de terceros en lugar de permanecer en un dispositivo local y es importante proteger la información, tanto a la hora de almacenarla como a la hora de trabajar con ella.

En este trabajo, nos centraremos en la forma de preservar la privacidad durante la realización de operaciones, utilizando información perteneciente a distintos usuarios o distintas fuentes de datos. También veremos una introducción a la criptografía para ver cómo se puede cifrar la información. No obstante, antes de tratar estos tema, describiremos los conceptos de seguridad y privacidad de la información.

3.1. Seguridad de la información

La seguridad de la información consiste en un conjunto de medidas que permiten proteger los datos de ataques tanto externos como internos, atendiendo a los 3 pilares de la información que se describen a continuación:

- **Confidencialidad:** El acceso a los datos debe estar limitado a los usuarios autorizados.
- **Integridad:** Cuando se recupera información almacenada, los datos recibidos deben ser correctos. Es necesario evitar que los datos sean modificados por usuarios no autorizados.
- **Disponibilidad:** Los datos deben poder recuperarse en cualquier momento en que los solicite un usuario autorizado.

3.2. Privacidad e información sensible

El concepto de **privacidad** ha sido definido de diversas maneras, pero nos quedaremos con la definición que hace Westin en [26], que da una idea bastante clara de lo que es la privacidad de la información:

> La privacidad es el derecho de los individuos, grupos o instituciones para determinar cuándo, cómo y hasta qué punto se comunica su información a los demás.
>
> (Westin, 1968: 7)

Otro concepto que resulta relevante en este proyecto es el de **información sensible**. Se considera información sensible a la **información personal identificable**, es decir: aquella información que sirve por sí sola para identificar a una persona. Por ejemplo, son datos sensibles el nombre, el domicilio, el número de teléfono, el email, o el número de cuenta bancaria.

Como podemos observar, la privacidad es un término muy amplio y difícil de cuantificar. El grado de privacidad establecido deberá ajustarse a cada caso concreto. Por ejemplo, no se protege de la misma manera un número de teléfono que un número de cuenta bancaria. Cuando es posible, se reduce el nivel de privacidad, normalmente con el objetivo de mejorar la eficiencia o de obtener un menor gasto económico.

Además, las capacidades de los adversarios crecen cada vez más, por lo que es necesario adaptarse a las nuevas tecnologías y a los nuevos tipos de ataques que van apareciendo.

Por estos motivos anteriores, y para conseguir una seguridad y privacidad adecuadas al contexto al que se van a aplicar, se pueden tener en cuenta varios factores:

- Se debe tener en cuenta el objetivo que se quiere conseguir al procesar los datos, estableciendo un alcance y unos límites.

- Se debe establecer quiénes son los *stakeholders* o partes interesadas.
- Se deben definir los roles y responsabilidades de cada usuario y cada parte del sistema.
- Se debe realizar una evaluación de los riesgos aplicables al caso concreto.

Capítulo 4

Criptografía

En este apartado, veremos algunos conceptos básicos de criptografía, para ver cómo se puede proteger la información.

A lo largo de la historia, se ha desarrollado el uso de mensajes cifrados con el objetivo de evitar que personas no autorizadas descubran el contenido de dicho mensaje. Aunque esto se ha utilizado principalmente en el ámbito militar, hoy en día tiene una gran importancia en el ámbito civil, y es especialmente relevante para el almacenamiento en la nube.

La ciencia encargada de estudiar la protección de información frente a usuarios no autorizados se denomina **criptología**, y se divide en el estudio de las siguientes disciplinas: criptografía y criptoanálisis. La **criptografía** es el estudio de técnicas de cifrado y descifrado de información. Es decir, se centra en los algoritmos, protocolos y sistemas que se utilizan para cifrar u ocultar información para hacerla ininteligible ante individuos no autorizados, y los métodos para descifrar y recuperar la información original. Por otro lado, el **criptoanálisis** estudia los métodos empleados para obtener la información original de un mensaje cifrado sin estar autorizado para ello, buscando un punto débil en la técnica de cifrado utilizada (ya sea tratando de obtener el mensaje original, o tratando de robar la clave que se ha utilizado en el cifrado).

4.1. Criptografía simétrica

La criptografía simétrica (o criptografía de clave privada) utiliza una única clave que se emplea tanto para cifrar como para descifrar. Dicha clave debe mantenerse en secreto, de lo contrario, cualquiera que tenga la clave podrá descifrar el mensaje cifrado.

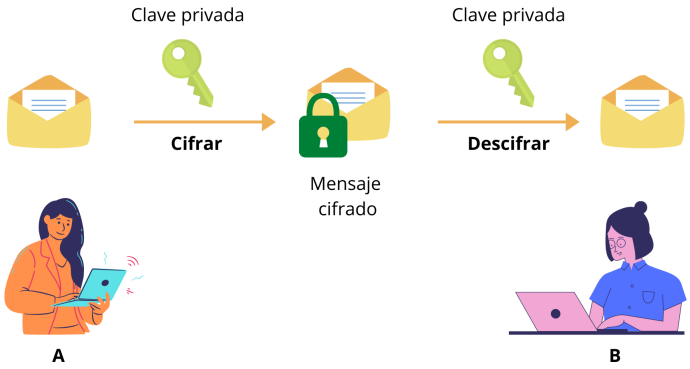

Figura 4.1: Criptografía simétrica.

En la figura 4.1, observamos a una persona A, que quiere enviar un mensaje a una persona B. Siguiendo un sistema de criptografía simétrica, se tiene una clave privada que A y B conocen. A cifra el mensaje con la clave y lo envía a B, que utiliza la misma clave para descifrarlo y obtener el mensaje original.

4.2. Criptografía asimétrica

En 1976, Diffie y Hellman introducen uno de los métodos criptográficos más relevantes: la **criptografía asimétrica** (o criptografía de clave pública). Se trata de una aportación de gran relevancia, ya que hasta entonces se utilizaban sistemas de clave privada, en los que tanto la persona que cifra el mensaje como el que la descifra deben conocer la clave privada. Este sistema tiene el riesgo de que la clave privada sea descubierta si se transmite a través de un canal inseguro.

En la criptografía asimétrica, cada usuario posee una clave pública y una clave privada. La clave pública es conocida, de forma que alguien que quiera enviar información debe utilizar la clave pública del usuario receptor para cifrar la información. El usuario receptor recibe la información cifrada y utiliza su clave privada para descifrarla. Como no es necesario transmitir la clave privada, y el usuario receptor es el único que la posee, otros individuos no podrán descifrar el mensaje.

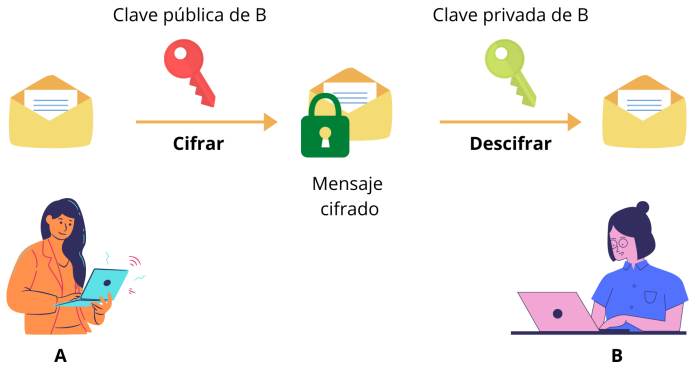

Figura 4.2: Criptografía asimétrica.

En la figura 4.2, observamos una persona A que envía un mensaje a una persona B utilizando un sistema de criptografía asimétrica. La persona B tiene una clave pública y una clave privada. Para enviarle un mensaje, A utiliza la clave pública de B para cifrar. Cuando B recibe el mensaje cifrado, utiliza su clave privada para descifrarlo y obtener el mensaje original.

La criptografía de clave pública no resulta muy eficiente a la hora de cifrar cantidades elevadas de información, debido a que la velocidad de cifrado es más lenta que la de los sistemas de clave privada. Sin embargo, se puede utilizar un sistema de clave pública para crear un canal de comunicación seguro para transmitir las claves privadas. De esta manera, los usuarios involucrados transmiten la clave del sistema de clave privada utilizando un sistema de clave pública y, posteriormente, cifrarán la información con un sistema de clave privada.

La seguridad de la criptografía de clave pública está basada en la dificultad de resolver ciertos problemas matemáticos. Un atacante que tenga un tiempo y unos recursos limitados no podrá conseguir el mensaje original. Este tipo de seguridad se llama **seguridad computacional**.

Esta seguridad se basa en recursos computacionales, por lo que teniendo unos recursos infinitos se llegaría a la solución. La idea es que con los recursos disponibles hoy en día, no se pueda llegar a la solución, o se necesite una cantidad de tiempo de cálculo demasiado elevada. Sin embargo, a medida que mejora la capacidad de cómputo de las máquinas utilizadas, estos sistemas criptográficos se vuelven más inseguros.

Frente a los criptosistemas de clave pública, los esquemas de compartición de secretos tienen la ventaja de presentar una **seguridad incondicional**, por lo que son seguros aunque el atacante tenga tiempo y recursos computacionales ilimitados. Como veremos más adelante, estos esquemas no se basan en ninguna hipótesis computacional, sino que se basan en restringir el acceso a la información a un subconjunto de los participantes.

Uno de los criptosistemas de clave pública más conocidos es el RSA, creado por Rivest, Shamir y Adleman. Además, es un de los primeros sistemas creados y se puede utilizar tanto para cifrar información como para realizar firmas digitales.

Capítulo 5

Computación Multiparte Segura

Sabemos que hoy en día se almacena información acerca de todo tipo de datos. Esta información resulta de mayor utilidad cuando podemos realizar operaciones sobre ella. Sin embargo, preservar la privacidad de esta información es más sencillo cuando únicamente se quiere almacenar que cuando se quiere trabajar con ella.

Por ejemplo, descifrar la información cada vez que se necesita realizar una operación sobre ella plantea un problema: si se procede de este modo, la entidad que realiza la operación está accediendo a datos privados no cifrados, poniendo en peligro la privacidad de la información.

A continuación, veremos que la computación multiparte segura propone una solución a este problema, permitiendo realizar este tipo de operaciones conjuntas sin que los datos dejen de ser privados. Hay que tener en cuenta que preservar la privacidad de los datos es considerado como un requerimiento imprescindible hoy en día, especialmente si se trata con datos valiosos o sensibles, como por ejemplo los datos relativos a la salud.

5.1. Origen y definición

El concepto de computación multiparte fue introducido por Andew Yao en 1982 [27]. La computación multiparte permite que varios participantes realicen unos cálculos a partir de valores secretos que posee cada uno de ellos, obteniendo el resultado deseado de forma que se revele la menor cantidad de información posible. Así, los participantes podrán realizar cálculos con sus datos, pero sin descubrir los valores secretos del resto de participantes.

- **Problema de los millonarios de Yao**

  Este problema fue planteado por Andew Yao para explicar el funcionamiento de la

computación multiparte, y consiste en lo siguiente:

En este problema, dos millonarios desean saber quién de los dos es más rico, sin revelar la riqueza que posee cada uno. Es decir, la única información que cada uno obtiene acerca del otro es si su riqueza es mayor o menor, pero no sabrán cuál es la cantidad que posee el otro.

- **Una solución al problema de los millonarios**

  A continuación, se explica una solución para el problema de los millonarios de Yao.

  Supongamos que tenemos un participante A que posee 3 millones ($x = 3$) y un participante B que posee 6 millones ($y = 6$).

  El objetivo es calcular cuál de los dos participantes es más rico sin revelar la cantidad que posee cada uno.

  En primer lugar, se deben representar los valores x e y en formato binario utilizando longitud n (en este caso $n = 3$).

  $$x = 3 = 011_2$$

  $$y = 6 = 110_2$$

  El siguiente paso, consiste en realizar los siguientes cifrados sobre x e y:

  - **0-encoding** o cifrado-0 de x es un conjunto $S_x^0 = \{x_n x_{n-1} ... x_{i+1} 1 \mid x_i = 0, \ \forall \ 1 \leq i \leq n\}$. En este cifrado, se invierte el bit menos significativo de todos los prefijos de x terminados en 0.
    Para $x = 011$, los prefijos son {0,01,011}, luego $S_x^0 = \{1\}$
    Para $y = 110$, los prefijos son {1,11,110}, luego $S_y^0 = \{111\}$
  - **1-encoding** o cifrado-1 de x es un conjunto $S_x^1 = \{x_n x_{n-1} ... x_{i+1} x_i \mid x_i = 1, \ \forall \ 1 \leq i \leq n\}$, formado por los prefijos de x terminados en 1.
    Para $x = 011$, los prefijos son {0,01,011}, luego $S_x^1 = \{01, 011\}$
    Para $y = 110$, los prefijos son = {1,11,110}, luego $S_y^1 = \{1, 11\}$

  Por último, se compara S_x^1 con S_y^0. Se tiene que $x > y$ si y solo si los dos conjuntos anteriores tienen algún elemento común (podemos ver la demostración en [12]), es decir, $S_x^1 \cap S_y^0 \neq \varnothing$. Por lo tanto, en este ejemplo se obtiene $x < y$ porque no hay ningún elemento común en estos conjuntos.

El problema de los millonarios se generaliza al problema GT ("*greater than*" o "mayor que"). Esta operación que permite comparar valores de forma segura tiene diversas aplicaciones, entre ellas la realización de subastas.

La computación multiparte (también llamada evaluación segura de funciones) es interesante cuando varios participantes, que pueden desconfiar unos de otros, desean realizar

una operación sobre sus datos conjuntos, y todos tienen interés en colaborar y en obtener los resultados de dicha operación.

Para definirla de manera más formal, la **computación multiparte** es "un modelo de computación distribuido en el que varios participantes calculan de forma colaborativa una función común sobre los valores de los demás, manteniendo la privacidad de sus propios valores secretos y obteniendo únicamente el resultado del cálculo". [23]

De este modo, este concepto resulta muy útil para realizar operaciones conjuntas cuando se quiere conservar la privacidad de los datos que van a ser empleados en los cálculos. Por ejemplo, permite realizar operaciones utilizando bases de datos distribuidas, y en general, operaciones con datos almacenados en la nube.

En la sección 5 de [13] y en la sección 1.3 de [9] se habla sobre varias aplicaciones de la computación multiparte segura, así como de ejemplos de despliegues de aplicaciones que se han llevado a cabo. A continuación, se exponen algunas de estas aplicaciones.

5.2. Aplicaciones de la computación multiparte

En esta sección, se mencionan algunas de las aplicaciones más interesantes de la computación multiparte segura:

- **Subastas electrónicas**

  En una subasta es importante tener en cuenta dos requisitos. En primer lugar, los valores establecidos por los postores y los vendedores deben mantenerse como valores privados (con posibilidad de revelar el valor de la oferta ganadora al final de la subasta). En segundo lugar, la subasta debe funcionar de manera que ningún participante pueda generar su valor a partir del de otro participante, obteniendo un valor ligeramente mayor y obteniendo así una ventaja. La computación multiparte se puede utilizar para cumplir estos requisitos.

- **Votación electrónica**

  La computación multiparte también se puede utilizar para votaciones, y es necesario que se cumplan los requisitos mencionados para las subastas: el voto de cada participante se mantiene secreto y ningún participante puede generar su voto en función de la elección de otro participante.

- **Gestión de claves**

  La gestión de claves es otra aplicación muy útil de la computación multiparte. Permite distribuir una clave en varios fragmentos, y que únicamente se pueda recuperar la clave si se juntan al menos n de los t fragmentos totales (ver esquemas umbral en 7.3). Esta distribución de la clave proporciona dos ventajas:

Por un lado, en el caso de que un atacante intente conseguir el valor de la clave, deberá conseguir al menos n de los fragmentos. Así, le será más difícil obtener la clave si está dividida en varios fragmentos que si estuviera almacenada en un solo fragmento (es decir, estando almacenada la clave en su forma original).

Por otro lado, si alguno de los fragmentos de la clave se pierde (por ejemplo, si se estropea el dispositivo en el que estaba almacenado), podrá recuperarse la clave a partir de los fragmentos restantes.

Varias plataformas que permiten gestión de claves son Sepior [18] y Unbound [24].

- **Machine learning**

  Tenemos el ejemplo de MiniONN [14] (2017), que plantea la posibilidad de transformar una red neuronal para convertirla en una red neuronal oculta. Esto se puede aplicar a modelos de machine learning en servicios en la nube, para poder preservar la privacidad de los datos y del modelo. Lo que se pretende conseguir es que el cliente mantenga sus datos privados y el servidor mantenga su modelo de machine learning privado.

- **Genómica (comparaciones de ADN)**

  Dado que los datos de ADN son datos sensibles en los que es muy importante preservar la privacidad, se ha propuesto aplicar la computación multiparte para poder realizar comparaciones de ADN con el objetivo de ayudar tanto en el diagnóstico de enfermedades como en la búsqueda de tratamientos. Podemos ver un ejemplo en [4] (2017).

- **Privacidad en bases de datos**

  Respecto a la realización de consultas en bases de datos, se consideran los siguientes escenarios: Por un lado, la realización de consultas a una base de datos sin que el propietario de la base de datos conozca la consulta que se ha realizado. Por otro lado, la realización de una consulta (por ejemplo, una consulta estadística) a una base de datos sin revelar valores individuales.

- **Análisis estadístico**

  Para realizar análisis protegiendo la privacidad de los datos, existen procesos como el k-anonimato o la seudonimización. Por un lado, **k-anonimato** consiste en generar una nueva versión de los datos de manera que no sea posible identificar a quién pertenecen. Esto se puede realizar de dos formas: mediante supresión (sustituyendo los valores de un atributo por un símbolo, por ejemplo, si tenemos una columna con los nombres de varias personas, sustituir los nombres por el símbolo "*"), o mediante generalización (sustituyendo los valores de un atributo por una categoría a la que pertenecen, por ejemplo, para un atributo "edad" se pueden establecer rangos de edades). Utilizando el k-anonimato, se consigue que para cada individuo del conjunto haya otros $k-1$ individuos que comparten el conjunto de atributos que podrían identificarlos.

Por otro lado, el prodedimiento de **seudonimización** consiste en sustituir los atributos de información personal por un seudónimo. De este modo, se tienen datos sin identificadores directos, y únicamente se podría identificar a un individuo a través de otra información adicional, que se encuentra almacenada por separado.

Con los dos métodos anteriores, el analista accede a una base de datos "anonimizada" (en la que no aparece información desde la que se pueda identificar a un individuo) en lugar de acceder a la base de datos original.

Uno de los posibles ataques que se realiza sobre los métodos anteriores para anular el anonimato establecido sobre la base de datos "anonimizada" es el ataque de vinculación de datos (o de re-identificación). Este ataque consiste en establecer una relación entre dos conjuntos de datos para los datos de un usuario, utilizando alguna información de enlace.

Sin embargo, la computación multiparte se considera una alternativa más deseable frente a los procedimientos anteriores, debido a que los resultados obtenidos son más imparciales y preserva la privacidad de forma más eficaz. Rmind [10] es un ejemplo para este tipo de aplicación. Se trata de una herramienta que realiza cálculos estadísticos sobre datos procedentes de distintas fuentes.

Además, estos análisis estadísticos se pueden aplicar a estudios para guiar decisiones de gobierno. Puesto que hoy en día hay una gran cantidad de datos que se guarda de manera digitalizada, podrían realizarse estudios sobre bases de datos gubernamentales, incluso de manera periódica, con el objetivo de ayudar al gobierno en la toma de decisiones. Sin embargo, algo que dificulta esta aplicación es la necesidad de proteger la información de las bases de datos utilizadas y la privacidad de los individuos cuya información ha sido recogida. Para dar solución a este problema, se puede realizar el análisis sobre estos datos utilizando la computación multiparte.

En [16] aparecen varias operaciones que se pueden realizar en un análisis estadístico preservando la privacidad, utilizando para ello la computación multiparte. Respecto a esta aplicación podemos mencionar Duality [6], que en 2020 presentó Duality SecurePlus Statistics [22], una solución que permite a varias organizaciones realizar análisis estadísticos utilizando distintas fuentes de datos, conservando su privacidad. En la figura 5.1 podemos observar un esquema que muestra funcionamiento.

Figura 5.1: Duality SecurePlus Statistics. [22]

5.3. Estado del arte

Ahora que conocemos las aplicaciones de la computación multiparte, veremos cuáles son algunas de las implementaciones que se han realizado en las últimas décadas.

- **Subasta de remolacha en Dinamarca (2008)** [17]

  Esta es la primera aplicación práctica de computación multiparte para un caso real a gran escala, y fue realizado en 2008. Antes de esta fecha, no se habían realizado implementaciones de este tipo, debido principalmente a que las soluciones que se podían crear a partir los primeros protocolos de computación multiparte resultaban muy ineficientes.

  Se trata de un ejemplo concreto de la aplicación de subasta electrónica, vista en el apartado anterior. Esta implementación está basada en el esquema de compartición de secretos de Shamir (que se explica en el apartado 7.3).

  En las subastas electrónicas, se pueden distinguir dos modelos principales: la subasta estándar en la que gana la oferta más alta, y la doble subasta. Este segundo modelo es el que se utiliza en esta aplicación, y sirve para casos en los que se quiere encontrar un precio justo para un producto a partir de su oferta y su demanda.

La necesidad de realizar esta aplicación aparece por el siguiente motivo. Los agricultores daneses que se dedican al cultivo de remolacha tienen un contrato de producción con la empresa Danisco, en el que se establece la cantidad de remolacha que se debe entregar y el precio al que se vende. Como Danisco es la única empresa en Dinamarca que se dedica a procesar remolacha, la falta de ayudas de la UE dedicadas a este sector provoca que aparezca la necesidad de modificar los contratos entre Danisco y los agricultores, para que la producción sea más rentable.

Esta modificación de los contratos de producción se realizó a nivel nacional utilizando la doble subasta. El objetivo de esta subasta consiste en encontrar el Precio de Compensación de Mercado (PCM), que depende de la oferta y la demanda del producto (en este caso, de remolacha). Durante el proceso de la subasta, tanto los vendedores como los compradores deben establecer los precios que consideran más adecuados, por los que están dispuestos a vender o comprar.

Por otro lado, en lo relativo a la seguridad, se consideró que la computación multiparte era la mejor opción. Previamente, se había planteado que Danisco realizara la subasta, pero los agricultores se podrían haber opuesto a ello ya que las pujas revelan información (acerca de su posición económica, su productividad, etc.). Y por otro lado, recurrir a una tercera empresa que realice la subasta y delegar en ella la responsabilidad de mantener la información privada es una opción que supone un gran gasto económico. Utilizar la computación multiparte supone una opción más barata y permite que ninguna de las partes tenga total acceso a la información en claro. Además, en esta aplicación se considera que el riesgo de que haya atacantes activos es muy baja, por lo que se establece una seguridad frente a adversarios semihonestos, lo que ayuda a mejorar la eficiencia (ver apartado 6.2 para ver los tipos de adversarios). Otro aspecto a tener en cuenta respecto a la seguridad, es que los servidores que se encargan de realizar los cálculos deben establecer pares de claves pública y privada para comunicarse con los clientes (es decir, con los participantes de la subasta).

De esta manera, al comenzar la subasta, los participantes (tanto compradores como vendedores) introducen sus valores. Estos valores se distribuyen utilizando compartición de secretos. Cuando el tiempo de la subasta ha terminado, se realizan los cálculos para obtener el PCM (estos cálculos son realizados por varios servidores). El tiempo de cálculo necesario es bastante elevado, en el caso de esta implementación fueron 30 minutos.

Tras una encuesta realizada a los agricultores, la mayoría afirmaron que (a pesar de no tener conocimiento acerca de los procesos que se realizan durante los cálculos) mantener la confidencialidad de las pujas era un factor importante para ellos, y estuvieron satisfechos con la confidencialidad conseguida.

- **Estudio sobre distintas fuentes de datos acerca de estudiantes en Estonia (2015)** [21]

  En este estudio, el objetivo consiste en observar si se produce una relación entre

los estudiantes dejan los estudios o los terminan más tarde de lo establecido y los estudiantes que están trabajando mientras estudian. Para ello, son necesarios los datos que confirman que un estudiante ha estado trabajando (historial laboral y de salarios, procedente de la Junta de Impuestos de Estonia) y los datos que confirman que ha estado estudiando (historial educativo, perteneciente al Ministerio de Educación), obtenidos de dos fuentes de datos distintas. Para permitir la colaboración de estos datos, se recurre a la computación multiparte, en concreto a la herramienta Sharemind [11].

Este ejemplo es un caso de estudio estadístico a gran escala utilizando datos gubernamentales, preservando su privacidad. Fue realizado en 2015, y su objetivo consiste en comprobar si hay una relación para estudiantes en Estonia entre trabajar durante los estudios universitarios y no graduarse a tiempo.

- **Publicidad: intersección de conjuntos para conversión de anuncios (2017)** [20]

  El problema de conversión de publicidad *online* a *offline* se puede resolver utilizando la computación multiparte.

  En este caso, para anuncios de Google, se quiere ver cuántas personas que han visto un anuncio han realizado una compra debido a la publicidad.

  Para ello, se calcula el tamaño de la intersección entre dos conjuntos privados. Por un lado, el proveedor de anuncios (Google), posee la información acerca de los usuarios que han visto un anuncio y, por otro lado, la empresa que vende el producto del anuncio posee la información de los clientes que lo han comprado y cuánto han gastado. Debido a la información con la que tratan, ninguno de ellos quiere revelar los datos que posee. Con la operación propuesta en [20], obtienen el tamaño de la intersección de sus conjuntos, es decir, el número de usuarios que han realizado una compra tras ver un anuncio.

  De esta manera, la empresa puede ver qué parte de sus beneficios se debe a la publicidad y decidir si es rentable seguir invirtiendo en ella, mientras que Google puede ver si los anuncios están teniendo efecto y hacen que las personas se decidan a realizar una compra.

- **Brecha salarial en Boston (2018)** [25]

  El objetivo es permitir que una tercera persona utilice sus recursos para realizar un estudio sobre las desigualdades salariales, manteniendo la privacidad de quienes han aportado sus datos para el estudio.

  Así, se puede estudiar si existe una desigualdad salarial en función de diferencias raciales o de género, o aplicar este estudio a un contexto concreto como las pequeñas empresas.

Capítulo 6

Seguridad

De forma general, se dice que un protocolo de computación multiparte es **seguro** cuando todos los participantes reciben resultados correctos y ninguno de los participantes puede obtener más información de la estrictamente necesaria (es decir, no se podrá obtener más información de la que se pueda deducir del propio resultado de las operaciones).

Lo que se quiere decir con que se pueda deducir información a partir del resultado es lo siguiente. Supongamos que dos participantes están realizando una operación de suma de secretos. En este caso, es fácil observar que si uno de los participantes tiene el valor secreto x y obtiene el valor z como resultado del cálculo, podrá deducir a partir de estos dos valores que el otro participante posee el valor secreto $y = z - x$. Sin embargo, para un caso con más participantes, no podrá obtener el valor secreto de cada uno de los demás participantes únicamente partiendo de su propio valor y del resultado. La cantidad de información que se pueda deducir de esta forma podrá ser aceptable dependiendo de la aplicación concreta, del tipo de información que se vaya a tratar o, como en este ejemplo, del número de participantes. Aparte de los prerrequisitos que se quieran considerar para realizar una operación, los participantes no deben ser capaces de obtener más información acerca de los secretos de los demás durante la realización del protocolo.

Los requisitos que se deben tener en cuenta en materia de seguridad para los protocolos de computación multiparte son los siguientes:

- **Privacidad:** únicamente se revela el resultado de la operación realizada, ningún participante debe obtener los valores privados de los demás.

- **Resultado correcto (*correctness*):** el resultado que obtienen los participantes debe ser correcto. El protocolo no debe poder ser modificado para devolver un resultado diferente al de evaluar la función con las entradas correspondientes a los participantes.

- **Independencia de las entradas:** un participante no podrá crear su valor de entrada a partir de los valores de los demás.
- **Imparcialidad:** si uno de los participantes recibe el resultado, todos los demás también deben recibirlo.
- **Garantía de recibir el resultado:** cada participante debe recibir el resultado de la operación.

Además de estos requisitos de seguridad que acabamos de describir, se debe utilizar un canal de comunicación seguro como prerrequisito para llevar a cabo un cálculo utilizando la computación multiparte. Por ejemplo, si se distribuyen los fragmentos de un secreto como veremos en el esquema de Shamir (7.3), estos fragmentos deberían enviarse cifrados.

6.1. Paradigma del mundo ideal y del mundo real

A la hora de estudiar un protocolo de computación multiparte y su seguridad, se consideran los paradigmas del mundo ideal y del mundo real, que funcionan como se describe a continuación:

- En el caso del **mundo ideal**, se tiene una tercera parte de confianza, que además será incorruptible. Los participantes envían sus valores a esta tercera parte, que se encarga de realizar los cálculos y devolver el resultado. Esta sería la forma intuitiva de resolver un problema. Sin embargo, esto presenta un inconveniente notable: para realizar las operaciones, esta tercera parte debe tener acceso a los datos en claro (sin cifrar).
- En el **modelo real**, no existe esta tercera parte. En su lugar, los participantes recurren a un protocolo de computación multiparte segura. De este modo, ninguno de los participantes y ninguna tercera parte tendrá acceso a todos los datos en claro. Un protocolo del mundo real debe simular un protocolo del mundo ideal.

Con estos dos paradigmas, se comprueba la seguridad de un protocolo. Para un adversario, un protocolo perteneciente al mundo real debe ser igual a uno perteneciente al mundo ideal. Es decir, un adversario no debe poder causar más daño ni obtener más información del protocolo real que si estuviera atacando al protocolo del mundo ideal.

Mientras que en el mundo ideal la visión que tiene un adversario del protocolo es únicamente su entrada privada y la salida o resultado de la operación, en el mundo real ve un registro de mensajes. Aún así, el mundo real debe ser una simulación del mundo

ideal para que el adversario no pueda obtener información sobre los datos privados de los participantes.

A modo de conclusión, tenemos que los protocolos de computación multiparte se pueden calcular de forma sencilla utilizando una tercera parte de confianza. El objetivo es realizar los cálculos y obtener el mismo resultado sin recurrir a esta tercera parte. De este modo, no se necesita que ningún participante tenga acceso a todos datos en claro.

6.2. Tipos de adversarios

Los participantes que atacan un protocolo de computación multiparte para obtener más información o modificar su funcionamiento a su favor se consideran participantes **deshonestos** o **corruptos**.

Dentro de estos participantes deshonestos o adversarios, se distinguen los siguientes tipos:

- **Semi-honestos:** también llamados **pasivos** u **honestos pero curiosos**.

  Estos son adversarios que siguen el protocolo, es decir, los mensajes que envía y sus resultados son los correctos. Sin embargo, intenta obtener más información de la que le corresponde, incumpliendo el requisito de privacidad. Tratan de obtener esta información interceptando los mensajes que se transmiten entre los participantes. Por ejemplo, en el caso de una subasta o de una votación, este tipo de adversario no tiene el objetivo de modificar el resultado final, sino de saber cuáles han sido los precios establecidos por los participantes, o qué ha votado cada participante.

- **Maliciosos:** también llamados **activos**.

  Estos adversarios se desvían de los pasos a seguir en el protocolo. Este tipo de adversario puede ser un programa que controla a varios de los participantes y determina su comportamiento durante la ejecución del protocolo. Además de hacer que se incumpla el requisito de privacidad, pueden atacar el resto de los requisitos. Pueden sustituir los valores de entrada por valores aleatorios, de modo que no se obtendrá un resultado final correcto (se incumple el requisito de resultado correcto). También pueden interrumpir la ejecución del protocolo, de manera que no se obtendrá un resultado de las operaciones (se incumple la garantía de recibir el resultado).

Una vez definidos los tipos de adversarios, veremos que los protocolos pueden presentar seguridad frente a uno o ambos tipos, e incluso frente a cierto porcentaje de adversarios. Como se indica en [5], considerando que un subconjunto $S \subset P$ (siento P el conjunto total de participantes), se puede tratar de participantes corruptos, se dice que un protocolo es

seguro contra un adversario **con umbral de corrupción t** si dicho protocolo es seguro contra todos los subconjuntos de tamaño como mucho t. De esta manera, muchos autores determinan la seguridad de sus protocolos diciendo que son seguros frente a cierto grado de corrupción (por ejemplo, seguros si 1/3 de los participantes son corruptos).

6.3. Tipos de seguridad

A partir de los tipos de adversarios anteriores, se distinguen dos tipos de seguridad:

- **Pasiva:** cuando la condición de seguridad definida en el párrafo anterior se mantiene en un escenario en el que todos los participantes siguen el protocolo.
- **Activa:** se da cuando la condición de seguridad del párrafo anterior se mantiene en el caso en que haya participantes que se desvían del protocolo.

De este modo, podemos tener dos tipos de seguridad para un protocolo de computación multiparte, dependiendo del tipo de adversarios frente a los que pueda seguir siendo seguro. Por ejemplo, en el caso que hemos visto de la subasta de remolacha en Dinamarca, se elige una seguridad pasiva ya que esto consigue aumentar la eficiencia del protocolo y el riesgo de que haya adversarios maliciosos se considera muy bajo.

Capítulo 7

Formas de abordar la computación multiparte

La computación multiparte ha sido abordada de distintas maneras, siendo las principales los circuitos cifrados de Yao, el cifrado homomórfico y los esquemas para compartir secretos. A continuación se describen estas tres herramientas, utilizadas para resolver los problemas de computación multiparte. La que más nos interesa es el esquema para compartir secretos (o esquema de Shamir), ya que será el que utilicemos a la hora de estudiar las operaciones a implementar en Python.

7.1. Circuitos cifrados de Yao

Este tipo de circuito cifrado (también llamado circuito confuso) fue introducido por Andrew Yao, quien también introdujo el concepto de computación multiparte. Los circuitos cifrados son una técnica criptográfica que permite que dos participantes evalúen una función de forma segura, donde dicha función se expresa en forma de circuito booleano y las entradas de los participantes se representan en forma de cadenas de bits.

Para ver su funcionamiento, en primer lugar hay que ver cómo funciona una puerta lógica en este tipo de circuitos.

Veamos un ejemplo con una puerta AND:

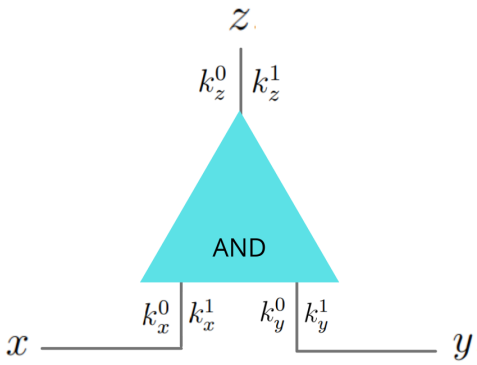

Figura 7.1: Puerta AND cifrada para un circuito de Yao.

Como se observa en la figura 7.1, hay dos cables de entrada x e y, y un cable de salida z. A cada cable se asocian dos claves, una de ellas tendrá valor 1 y otra 0. Para x, las claves asociadas son k_x^0 y k_x^1, mientras que y tiene las claves k_y^0 y k_y^1, y z tiene las claves k_z^0 y k_z^1.

Sabemos que la tabla de verdad para una puerta AND es la siguiente:

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

Tabla 7.1: Tabla de verdad de una puerta AND con entradas x, y y salida z.

A partir de la tabla anterior, realizamos una tabla llamada GCT (*Garbled Computation Table* o tabla cifrada). En esta tabla tendremos los valores de las claves asociadas a las entradas y a la salida, y en la última columna se tiene el cifrado que correspondería a la salida (siendo $E_k(a)$ el cifrado de a utilizando la clave k):

x	y	z	GCT
k_x^0	k_y^0	k_z^0	$E_{k_x^0}(E_{k_y^0}(k_z^0))$
k_x^0	k_y^1	k_z^0	$E_{k_x^0}(E_{k_y^1}(k_z^0))$
k_x^1	k_y^0	k_z^0	$E_{k_x^1}(E_{k_y^0}(k_z^0))$
k_x^1	k_y^1	k_z^1	$E_{k_x^1}(E_{k_y^1}(k_z^1))$

Tabla 7.2: Tabla GCT para la puerta AND.

Si tenemos dos valores de entrada a, $b \in \{1, 0\}$, solo se puede descifrar correctamente una de las filas de la tabla GCT, la que corresponda a: $E_{k_x^a}(E_{k_y^b}(k_z^{g(a,b)}))$, siendo g la función que se desea evaluar.

Teniendo en cuenta lo anterior, la forma de proceder para construir una puerta cifrada es la siguiente:

El participante A debe elegir dos claves aleatorias para cada cable como hemos visto anteriormente, de forma que se tendrán 6 claves en total. A continuación, crea la tabla GCT y aplica una permutación sobre sus filas. Esta permutación se realiza con la finalidad de evitar que la posición de las claves revele información acerca de los valores originales al que se asocian.

El participante A envía la última columna de la tabla GCT y la clave k_x^a (siendo a el valor de entrada de A, $a \in \{0, 1\}$).

El participante B no puede descifrar los valores de la tabla GCT únicamente con la clave k_x^a de A, necesita también la clave correspondiente a su valor. Este proceso presenta dos problemas: por un lado, si A envía las dos claves posibles para B, este último podrá descifrar más información de la necesaria, mientras que si B pide que A le envíe solo la clave correspondiente a su valor b, A conocerá el valor privado de B. Como queremos que los participantes obtengan la menor cantidad de información posible, para poder garantizad la privacidad de sus datos de entrada, la clave de B se transmite utilizando transferencia inconsciente (*oblivious transfer*). El protocolo de transferencia inconsciente parte de dos valores m_0, m_1 que posee el emisor y un bit $b \in \{0, 1\}$ que posee el receptor. Al final del protocolo, el receptor obtiene m_b, y el emisor no obtiene ninguna información. El participante B obtiene su clave k_y^b utilizando este protocolo y descifra los valores de la tabla GCT.

Sólo uno de los valores de la tabla GCT es el correcto. Para que B reconozca cuál es, normalmente se utiliza un bloque de ceros al cifrar, de manera que si B obtiene la clave seguida de un bloque de ceros al descifrar, sabrá que corresponde al valor correcto: $k_z^{g(a,b)}$.

Para un circuito cifrado completo, dicho circuito debe representar la función que los

participantes que quieren evaluar, y todas las puertas lógicas que lo forman deben estar cifradas de la forma que acabamos de ver.

Respecto a la seguridad de estos circuitos, son seguros frente a adversarios semihonestos (es decir, presentan seguridad pasiva). Una forma de modificarlos para que presenten seguridad activa, consiste en utilizar pruebas de conocimiento cero. Las pruebas de conocimiento cero permiten que un participante pruebe ante otro que una afirmación es cierta, de forma que cualquier resultado calculado a partir de la afirmación también será cierto. Este método mejora la seguridad, pero tiene un coste computacional mayor.

Para terminar, podemos mencionar que una de las aplicaciones más comunes de los circuitos de Yao es la minería de datos preservando la privacidad.

7.2. Cifrado homomórfico

Esta segunda forma de abordar la computación multiparte se basa en un modelo cliente-servidor y utiliza las propiedades homomórficas de los esquemas de cifrado, permitiendo realizar operaciones sobre los datos sin necesidad de descifrarlos.

Dentro del cifrado homomórfico se distinguen tres tipos: total, medio y parcial.

- El cifrado **parcialmente** homomórfico se puede utilizar para evaluar cualquier función utilizando operaciones aditivas o multiplicativas, pero no se pueden combinar ambos tipos de operaciones. En este sentido, es más restrictivo que los otros, y también es el que conlleva un menor coste computacional.
- En los esquemas homomórficos de tipo **medio**, se pueden utilizar tanto operaciones aditivas como multiplicativas, pero de manera restringida. Por ejemplo, se puede emplear este esquema evaluando un circuito de cierta profundidad. Por ello, este tipo de esquema también se denomina cifrado homomórfico **nivelado**.
- Por último, los esquemas de cifrado **totalmente** homomórfico permiten utilizar tanto operaciones aditivas como multiplicativas para evaluar cualquier función, y no presentan la limitación existente en los esquemas de tipo medio. Este tipo de cifrado es el que presenta un mayor coste computacional.

Los esquemas de Paillier, Goldwasser-Micali, y ElGamal son algunos ejemplos de los más conocidos, y son esquemas **parcialmente homomórficos**.

El esquema de Paillier permite realizar la suma de dos elementos cifrados, y también la multiplicación de un elemento cifrado y otro en texto plano. Sin embargo, no permite multiplicar dos elementos cifrados. Supongamos que tenemos los valores u_1, u_2 y que $E(x)$

es el cifrado de un elemento x utilizando el esquema de Paillier. Entonces, a partir de $E(u_1)$ y $E(u_2)$, es posible calcular $E(u_1 + u_2)$, pero no podremos calcular $E(u_1 \cdot u_2)$. Se podría calcular $E(u_1 \cdot u_2)$ a partir de $E(u_1)$ y u_2, pero es importante que nos fijemos en que u_2 no está cifrado.

De este modo, y recordando que se utiliza el modelo cliente-servidor, sería posible que el cliente almacenase sus datos de forma remota (por ejemplo, a través de almacenamiento en la nube) y solicitase al servidor la realización de alguna operación con esos datos. El servidor realiza la operación sin desencriptarlos y devuelve el resultado al cliente. El cliente tiene la clave de descifrado y puede desencriptar el valor enviado por el servidor para poder hallar el resultado de la operación.

Los esquemas de Paillier y Goldwasser-Micali son aditivos, mientras que el ElGamal es multiplicativo. En el esquema de ElGamal, a partir de los valores cifrados $E(u_1)$ y $E(u_2)$, se puede obtener $E(u_1 \cdot u_2)$.

A continuación, se describe de manera formal lo que significa que esquema de cifrado homomórfico sea aditivo o multiplicativo:

Sea $\diamond$ una operación, u_1, u_2 dos elementos no cifrados, y $E(u_1)$, $E(u_2)$ los cifrados de los elementos anteriores, un esquema es homomórfico:

- aditivo $\Leftrightarrow E(u_1) \diamond E(u_2) = E(u_1 + u_2)$
- multiplicativo $\Leftrightarrow E(u_1) \diamond E(u_2) = E(u_1 \cdot u_2)$

En 2009, Gentry [8] presenta el primer esquema de cifrado **totalmente homomórfico**. Además, plantea dos aplicaciones principales para su esquema: la realización de consultas cifradas en motores de búsqueda y las búsquedas sobre datos cifrados. Este esquema tiene la ventaja de presentar una menor complejidad de comunicación frente a los circuitos cifrados de Yao. Sin embargo, la carga computacional es relativamente elevada, para lo que el autor propone utilizar este esquema de forma asíncrona, por ejemplo, aplicado al filtrado de spam de correos electrónicos cifrados.

Es interesante tener en cuenta que el principal caso de uso al que se aplica el cifrado homomórfico es el *cloud computing* o computación en la nube. De este modo, un usuario puede acceder a una mayor potencia computacional de la que tiene en su equipo, y deja que el servicio en la nube realice los cálculos (al usuario debe costarle menos encriptar los datos que realizar el cálculo, para que le salga rentable alquilar los servicios en la nube). Los servicios en la nube procesan los datos sin descifrarlos.

Otro caso de uso también bastante común es *machine learning* preservando la privacidad. Además de cifrar los valores privados, se puede cifrar la función evaluada, por lo que en este caso se pueden mantener privados tanto los datos como el modelo de *machine learning* utilizado.

7.3. Compartición de Secretos: Esquema de Shamir

Blakley [3] y Shamir [19] introdujeron el método de compartición de secretos, cada uno de forma independiente, en 1979. La idea básica consiste en dividir un valor secreto en n fragmentos, de forma que cada participante obtiene un fragmento, y sólo es posible recuperar el valor secreto si se juntan al menos t de los fragmentos (con $t \leq n$) para un umbral (t, n). Este será el método en el que nos centraremos a partir de ahora.

Por consiguiente, en esta sección se explica una forma de compartir secretos entre varios participantes o usuarios utilizando el esquema de Shamir. La mayoría de operaciones de computación multiparte que se describen más adelante están basadas en este esquema de compartición de secretos.

El objetivo principal es repartir un valor secreto entre varios participantes de manera que sólo si se pone de acuerdo un número mínimo de participantes se pueda recuperar el valor secreto. Entonces, si el número de participantes no alcanza el mínimo, será imposible que recuperen el secreto.

Esto se puede aplicar, por ejemplo, en la gestión de claves criptográficas, dividiéndola en varias partes que deben reunirse para recuperarla. Por un lado, este esquema permite que la clave se pueda recuperar si alguna de las partes se ha perdido, pero también dificulta la posibilidad de que la clave sea descubierta por usuarios no autorizados, ya que en lugar de estar almacenada en un único lugar, estará dividida en varias partes (cada una en un lugar distinto) y se necesitan varias partes para poder conseguirla.

Teniendo en cuenta los conceptos mencionados en el apartado de seguridad 6, los protocolos con una mayoría honesta de participantes (es decir, donde la mayoría de los participantes no son corruptos) suelen utilizar la compartición de secretos.

Esquema de Shamir

El esquema de Shamir proporciona una forma de distribuir un valor entre varias partes y mantener ese valor secreto hasta que todos los participantes se ponen de acuerdo y deciden recuperar dicho valor.

Según el funcionamiento del esquema de Shamir, se establece un número mínimo de participantes que se necesitan para recuperar el valor secreto. Supongamos que de un grupo de t participantes, se necesitan al menos n participantes para recuperar el valor (siendo $n \leq t$). Esto es lo que se llama un esquema de umbral (n, t). Con esta técnica, si se tienen menos de n participantes, es imposible recuperar el valor secreto.

Este tipo de esquema se puede aplicar a situaciones en las que se necesite llegar a un acuerdo entre varias partes para realizar alguna acción.

El esquema de Shamir funciona de la siguiente manera:

Supongamos un conjunto de t participantes: $P_1, P_2, ..., P_t$. Cada participante P_i tendrá una parte s_i de un valor secreto S, de forma que cada P_i conoce únicamente el s_i que le corresponde, pero no los valores s_j de los demás participantes (siendo $j \neq i$).

Sea n, $1 \leq n \leq t$, el mínimo número de participantes (o de fragmentos del secreto, si algún participante posee más de un fragmento) que se necesita para recuperar el valor secreto S. Se podrá obtener S a partir de n o más partes s_i del secreto, pero nunca a partir de menos de n partes.

De este modo, tenemos lo que se llama un **esquema de umbral (n,t)** para los valores $s_1, s_2, ..., s_t$.

Para calcular el valor secreto S, los participantes realizarán una interpolación polinómica. En el caso en que no haya suficientes participantes, estos obtendrán un sistema de ecuaciones indeterminado, por lo que no podrán llegar a la solución y serán incapaces de recuperar el valor secreto.

A continuación, se muestra un ejemplo utilizando el esquema de Shamir.

7.3.1. Ejemplo: Esquema de Shamir

Supongamos que se quiere compartir el secreto $S = 563$ y que el esquema umbral (n, t) que se quiere emplear es un esquema de umbral $(3, 5)$.

En primer lugar, se crea un polinomio de grado $n - 1 = 2$, de la forma:

$$p(x) = a + b \cdot x + c \cdot x^2$$

donde $a = S$ y los coeficientes b y c se generan de forma aleatoria.

Por ejemplo, consideremos el polinomio $p(x) = 563 + 32 \cdot x + 67 \cdot x^2$.

A continuación, cada participante P_i elige un punto x_i de forma aleatoria y obtiene el resultado de evaluar el polinomio $p(x)$ en ese punto.

Por ejemplo, consideremos que cada participante tiene los puntos siguientes (para cada P_i se tiene $(x_i, p(x_i))$:

$$P_1 \rightarrow (1, 662)$$

$$P_2 \rightarrow (2, 895)$$

$$P_3 \rightarrow (3, 1262)$$
$$P_4 \rightarrow (4, 1763)$$
$$P_5 \rightarrow (5, 2398)$$

Ahora, para poder recuperar el secreto, al menos 3 participantes deben ponerse de acuerdo (ya que el esquema es de umbral (3,5)). Por ejemplo, si los participantes P_1, P_2 y P_4 quieren recuperarlo, deben resolver un sistema de ecuaciones de la forma $m + n \cdot x + q \cdot x^2$, utilizando los puntos calculados anteriormente. Obtienen el sistema de ecuaciones siguiente:

$$\begin{cases} m + n \cdot x_1 + q \cdot {x_1}^2 = p(x_1) \\ m + n \cdot x_2 + q \cdot {x_2}^2 = p(x_2) \\ m + n \cdot x_4 + q \cdot {x_4}^2 = p(x_4) \end{cases}$$

Sustituimos por los valores correspondientes y se obtiene:

$$\begin{cases} m + n + q = 662 \\ m + 2 \cdot n + 4 \cdot q = 895 \\ m + 4 \cdot n + 16 \cdot q = 1763 \end{cases}$$

Observamos que si tuviéramos menos de 3 participantes, el sistema sería indeterminado y no podríamos hallar la solución.

Se resuelve el sistema y se obtiene la siguiente solución:

$$\begin{cases} m = 563 \\ n = 32 \\ q = 67 \end{cases}$$

Esta **solución** corresponde al polinomio $p(x) = 563 + 32 \cdot x + 67 \cdot x^2$.

El **valor del secreto** S que los participantes quieren obtener corresponde al valor del **término independiente**: $m = 563$.

7.3.2. Realizar operaciones utilizando el esquema de Shamir

Utilizando la compartición de secretos que nos facilita el esquema de Shamir, será posible realizar muchas de las operaciones que aparecen posteriormente. Siguiendo el modelo que aparece en [2], la realización de estos cálculos se puede dividir en los pasos siguientes:

Considerando un total de t participantes y un esquema de umbral (n, t),

1. Cada participante P_i distribuye entre los demás participantes las partes o fragmentos de su valor secreto s_i, utilizando un polinomio de grado $n - 1$.

2. Se realiza el cálculo de la operación deseada, $F(x_1, ..., x_t)$. Para ello, cada participante realiza dicha operación con los fragmentos de los secretos que le han enviado el resto de participantes, y el fragmento de su propio secreto.

3. A partir de los resultados obtenidos por los participantes en la operación anterior, se realiza una reconstrucción del resultado para obtener el valor deseado.

Capítulo 8

Computación Multiparte Segura: Operaciones

A continuación veremos algunas operaciones utilizando la computación multiparte. Tanto la suma como la multiplicación se consideran operaciones básicas, y a partir de ellas se pueden realizar otro tipo de operaciones más complejas.

Además, la combinación lineal se puede considerar como un caso general para la suma, y a partir de este caso general, se puede ir al caso particular del cálculo de la media. Este tipo de operaciones tienen la ventaja de presentar una menor complejidad frente a las demás, ya que cada participante necesita realizar una única operación, que es la combinación lineal de los fragmentos de los secretos.

Estas operaciones se realizan en módulo primo p, pero si se quieren evitar las reducciones modulares, bastaría con establecer un valor p lo suficientemente grande en relación a los valores de entrada.

Por otro lado, a partir del cálculo de un valor en una posición concreta, se pueden calcular datos estadísticos como la mediana y los cuartiles de un conjunto de datos, permitiendo que este conjunto total esté formado por varios subconjuntos privados pertenecientes a los distintos participantes.

A continuación se describen estas operaciones, incluyendo ejemplos de su funcionamiento. En el anexo A, se encuentra una explicación de la implementación en Python de estas operaciones, junto a ejemplos de ejecución de los programas.

8.1. Suma de secretos

Una de las operaciones que vamos a realizar es la suma de secretos. En la suma de secretos, cada participante tendrá un valor secreto, y el objetivo es calcular la suma de los secretos de varios participantes sin que ninguno de ellos pueda obtener información acerca de los secretos que poseen los demás.

Para ver cómo se realiza la suma de secretos, partimos de un número t de participantes: $P_1, P_2, ..., P_t$, cada participante P_i con un valor secreto s_i que los demás participantes $P_j, (j \neq i)$, no conocen. Supongamos que los participantes quieren calcular la suma de sus secretos $s_1 + s_2 + ... + s_t$ sin revelar su valor secreto a los demás.

Algoritmo 1 Suma de secretos

Entrada: Valores secretos $s_1, s_2, ..., s_t$. Cada valor s_i pertenece al participante P_i, $(1 \leq i \leq t)$.
Salida: Suma de los valores secretos: $s_1 + s_2 + ... + s_t$.
Procedimiento:

1. Cada participante P_i elige un valor aleatorio α_i, con la condición de que todos los α_i deben ser distintos (se puede elegir $\alpha_i = i$). Llamaremos a estos valores "segmentos", y serán valores públicos conocidos por todos los participantes.

2. Cada participante P_i crea un polinomio $p_i(x)$ para distribuir su secreto. El grado del polinomio debe ser $(t-1)$, el término independiente debe ser el valor del secreto s_i, y el resto de coeficientes son aleatorios.

3. Cada participante P_i evalúa su polinomio $p_i(x)$ en los puntos x = α_j para todo $j = 1, 2, ..., t; j \neq i$, y envía los resultados obtenidos al resto de los participantes, de modo que cada P_j recibe el valor $p_i(\alpha_j)$.

4. De acuerdo con el paso anterior, ahora cada participante P_j tiene los valores $p_i(\alpha_j$ para $i = 1, 2, ..., t; i \neq j$. Cada P_j calcula la suma de estos valores:

$$h(\alpha_j) = \sum_{i=1}^{t} p_i(\alpha_j)$$

5. Los participantes ponen en común los valores $h(\alpha_i)$ y plantean un sistema de ecuaciones como el del apartado acerca del esquema de Shamir. Debe ser un polinomio de grado $t-1$ que pase por los puntos $(\alpha_i, h(\alpha_i))$. Se resuelve el sistema y el valor que corresponda al término independiente es el valor de la suma de los secretos de los participantes.

8.1.1. Ejemplo: Suma de Secretos

Consideramos 3 participantes P_1, P_2, P_3, que poseen los valores secretos $s_1 = 2, s_2 = 4$ y $s_3 = 5$, respectivamente.

Por otro lado, los participantes P_1, P_2, P_3 tienen los segmentos $\alpha_1 = 1, \alpha_2 = 2, \alpha_3 = 3$, respectivamente.

El **objetivo** es calcular la suma de $s_1 + s_2$.

En primer lugar, P_1 y P_2 utilizan los polinomios siguientes para distribuir los secretos (suponemos un esquema de umbral (3,3)):

$p_1(x) = x^2 + x + a = x^2 + x + 2$

$p_2(x) = 2 \cdot x^2 - x + b = 2 \cdot x^2 - x + 4$

El participante P_1 utiliza su polinomio y los segmentos de los demás participantes (estos segmentos son valores conocidos por todos) para calcular los resultados siguientes:

$p_1(\alpha_1) = p_1(1) = 1^2 + 1 + 2 = 4 \quad \rightarrow \quad P_1$ se queda con este resultado

$p_1(\alpha_2) = p_1(2) = 2^2 + 2 + 2 = 8 \quad \rightarrow \quad P_1$ le envía el resultado a P_2

$p_1(\alpha_3) = p_1(3) = 3^2 + 3 + 2 = 14 \quad \rightarrow \quad P_1$ le envía el resultado a P_3

El participante P_2 utiliza su polinomio y los segmentos de los participantes para calcular los resultados siguientes:

$p_2(\alpha_1) = p_2(1) = 2 \cdot 1^2 - 1 + 4 = 5 \quad \rightarrow \quad P_2$ le envía el resultado a P_1

$p_2(\alpha_2) = p_2(2) = 2 \cdot 2^2 - 2 + 4 = 10 \quad \rightarrow \quad P_2$ se queda con este resultado

$p_2(\alpha_3) = p_2(3) = 2 \cdot 3^2 - 3 + 4 = 19 \quad \rightarrow \quad P_2$ le envía el resultado a P_3

Los participantes quieren calcular $a + b$. Para ello, cada participante P_i debe calcular $h(\alpha_i) = p_1(\alpha_i) + p_2(\alpha_i)$.

El participante P_1 calcula: $h(\alpha_1) = p_1(\alpha_1) + p_2(\alpha_1) = 4 + 5 = 9$

El participante P_2 calcula: $h(\alpha_2) = p_1(\alpha_2) + p_2(\alpha_2) = 8 + 10 = 18$

El participante P_3 calcula: $h(\alpha_3) = p_1(\alpha_3) + p_2(\alpha_3) = 14 + 19 = 33$

Ahora, deben poner en común los valores $h(\alpha_i)$ obtenidos al realizar el cálculo anterior y buscar un polinomio de la forma $m + n \cdot x + q \cdot x^2$ que pase por los puntos $(\alpha_i, h(\alpha_i))$.

Por lo tanto, deben resolver el sistema de ecuaciones siguiente:

$$\begin{cases} m + n + q = 9 \\ m + 2 \cdot n + 4 \cdot q = 18 \\ m + 3 \cdot n + 9 \cdot q = 33 \end{cases}$$

La solución del sistema es: $m = 6, n = 0, q = 3$. Por lo tanto el polinomio obtenido es $6 + 3 \cdot x^2$. La solución que quieren obtener los participantes es el término independiente de este polinomio, luego la solución de la suma $a + b$ es 6.

8.2. Combinación Lineal

El apartado anterior de la suma de secretos, es realmente un caso particular de combinación lineal en el que los coeficientes de la combinación valen 1 para todos los participantes.

A continuación veremos el caso general: la combinación lineal.

En el caso de la suma, el paso 4 del algoritmo 1 consiste en sumar los fragmentos de los valores secretos. Este es el único paso que cambia en la combinación lineal, ya que en lugar de la suma, se debe realizar la combinación lineal con los coeficientes deseados.

8.2.1. Ejemplo: Combinación lineal con 3 participantes

Como las operaciones se realizan de forma distribuida, en la figura 8.1 se muestra un diagrama para reflejar de forma visual qué valores se comparten y cuáles permanecen como valores secretos o privados.

En el esquema, se muestra el caso en el que 3 participantes quieren realizar el cálculo de la función $h(a, b, c) = x \cdot a + y \cdot b + z \cdot c$, siendo a, b, c los valores secretos de los participantes P_1, P_2, P_3, respectivamente.

Observamos que los valores de partida son los secretos a, b, c y los segmentos (que son valores públicos) $\alpha_1, \alpha_2, \alpha_3$. Aunque los segmentos son valores públicos, cada uno corresponde a un participante. Por otro lado, el paso final consiste en obtener el resultado de la operación $h(a, b, c)$.

Otro aspecto que podemos observar en el diagrama son los mensajes que se envían los participantes. En primer lugar, deben enviar los segmentos que pertenecen a cada uno (α_i), ya que todos deben conocerlos. Después, deben enviar los valores obtenidos al evaluar los polinomios distribuidores (es decir, los fragmentos de los secretos: $\beta_{i,j}$). Por último, se deben enviar los resultados parciales de la combinación lineal (μ_i).

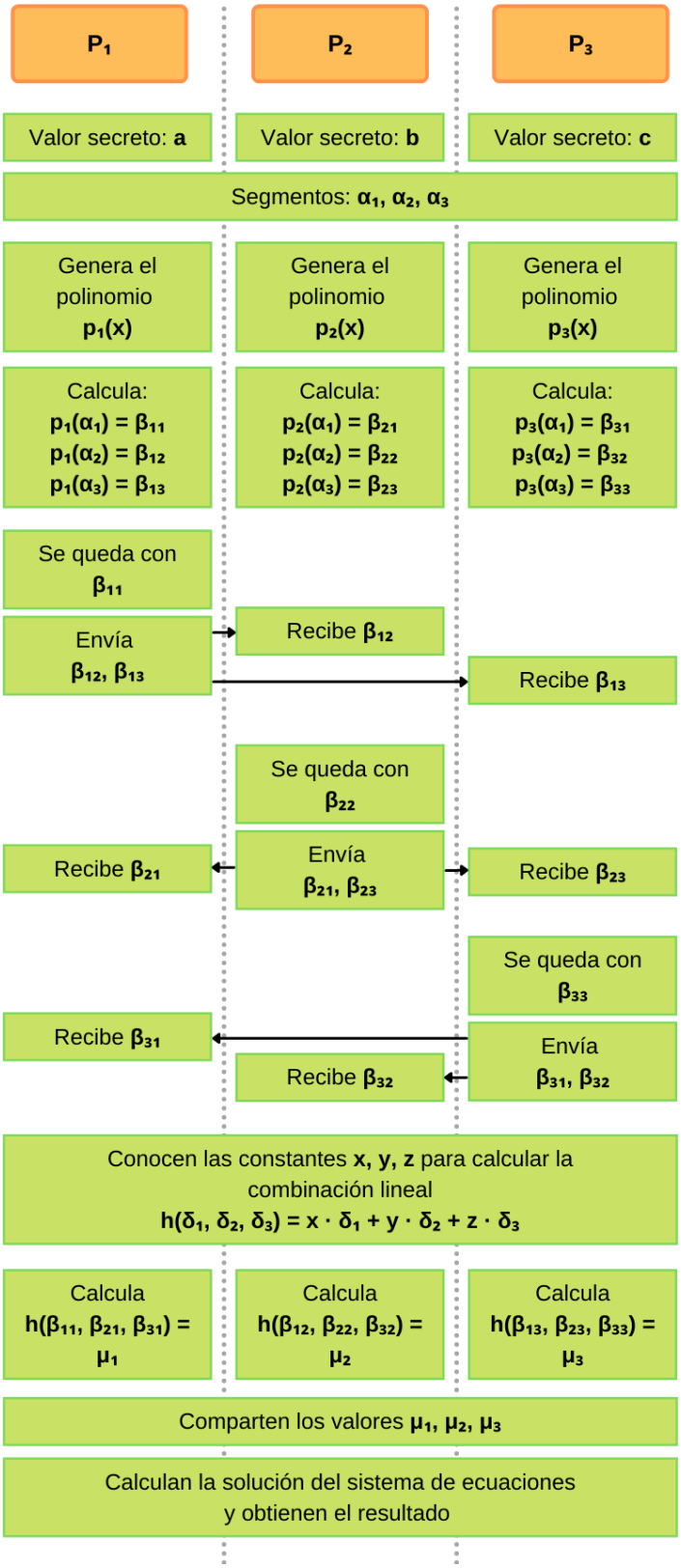

Figura 8.1: Combinación lineal con 3 participantes

8.3. Cálculo de la media

Teniendo en cuenta el apartado anterior, que explica la forma de realizar una combinación lineal, el cálculo de la media es muy sencillo. Simplemente se necesita tener en cuenta que los coeficientes de esta combinación lineal deben ser $1/n$ para cada valor secreto, siendo n el número total de participantes. Es decir, la función a calcular es:

$$h(\alpha_1, \alpha_2, ..., \alpha_n) = \frac{\alpha_1 + \alpha_2 + ... + \alpha_n}{n} = \frac{1}{n} \cdot \alpha_1 + \frac{1}{n} \cdot \alpha_2 + ... + \frac{1}{n} \cdot \alpha_n$$

siendo $\alpha_1, \alpha_2, ..., \alpha_n$ los valores secretos correspondientes a los participantes $P_1, P_2, ..., P_n$, respectivamente.

8.3.1. Ejemplo: Cálculo de la media

Supongamos que tres personas quieren averiguar cuál es su sueldo medio sin revelar cuánto cobra cada una.

Para realizar este cálculo de forma segura, cada una de estas personas, que denominaremos participantes, debe repartir a todos los demás un fragmento de su sueldo, siendo su sueldo el valor secreto.

Esto se puede realizar mediante un esquema de Shamir de umbral (3,3).

Supongamos que los participantes tienen los sueldos siguientes: $s_1 = 1500$, $s_2 = 2000$, $s_3 = 1000$.

Cada participante tendrá, además, un valor que hemos denominado segmento: $\alpha_1 = 1$, $\alpha_2 = 2$, $\alpha_3 = 3$.

A continuación, cada participante debe elegir un polinomio que utilizará para distribuir su valor secreto. Los polinomios se eligen al azar, pero deben ser de grado 2 y el término independiente debe ser el valor del secreto. El resto de los coeficientes son aleatorios.

Supongamos que los polinomios elegidos son los siguientes:

$p_1(x) = 1500 + x + x^2$ será el polinomio para distribuir el secreto del participante P_1

$p_2(x) = 2000 - x + x^2$ será el polinomio para distribuir el secreto del participante P_2

$p_3(x) = 1000 + x - x^2$ será el polinomio para distribuir el secreto del participante P_3

Los segmentos α_i son conocidos, así que cada participante utiliza su polinomio con los segmentos de los demás participantes para distribuir su secreto. De este modo, cada

participante conoce únicamente su propio polinomio, y envía a los demás el resultado de evaluar el polinomio en el segmento correspondiente.

El participante P_1 calcula los valores:

$p_1(\alpha_1) = p_1(1) = 1500 + 1 + 1^2 = 1502$

$p_1(\alpha_2) = p_1(2) = 1500 + 2 + 2^2 = 1506 \rightarrow$ envía el resultado al participante P_2

$p_1(\alpha_3) = p_1(3) = 1500 + 3 + 3^2 = 1512 \rightarrow$ envía el resultado al participante P_3

El participante P_2 calcula los valores:

$p_2(\alpha_1) = p_2(1) = 2000 - 1 + 1^2 = 2000 \rightarrow$ envía el resultado al participante P_1

$p_2(\alpha_2) = p_2(2) = 2000 - 2 + 2^2 = 2002$

$p_2(\alpha_3) = p_2(3) = 2000 - 3 + 3^2 = 2006 \rightarrow$ envía el resultado al participante P_3

El participante P_3 calcula los valores:

$p_3(\alpha_1) = p_3(1) = 1000 + 1 - 1^2 = 1000 \rightarrow$ envía el resultado al participante P_1

$p_3(\alpha_2) = p_3(2) = 1000 + 2 - 2^2 = 998 \rightarrow$ envía el resultado al participante P_2

$p_3(\alpha_3) = p_3(3) = 1000 + 3 - 3^2 = 994$

Ahora, cada participante debe calcular la función de la media (es el cálculo al que se quiere llegar) con los valores que ha obtenido.

El participante P_1 calcula: $media(\alpha_1) = \frac{p_1(\alpha_1)+p_2(\alpha_1)+p_3(\alpha_1)}{3} = \frac{1502+2000+1000}{3} = 1500 + \frac{2}{3}$

El participante P_2 calcula: $media(\alpha_2) = \frac{p_1(\alpha_2)+p_2(\alpha_2)+p_3(\alpha_2)}{3} = \frac{1506+2002+998}{3} = 1502$

El participante P_3 calcula: $media(\alpha_3) = \frac{p_1(\alpha_3)+p_2(\alpha_3)+p_3(\alpha_3)}{3} = \frac{1512+2006+994}{3} = 1504$

Con estos últimos valores que han obtenido al calcular la media, deben hallar un polinomio de la forma $a + b \cdot x + c \cdot x^2$ que pase por los puntos:

$$(\alpha_1, media(\alpha_1)), (\alpha_2, media(\alpha_2)), (\alpha_3, media(\alpha_3))$$

.

Por lo tanto, deben resolver el sistema de ecuaciones siguiente:

$$\begin{cases} a + b \cdot 1 + c \cdot 1 = 1500 + \frac{2}{3} & \rightarrow \quad 3 \cdot a + 3 \cdot b + 3 \cdot c = 4502 \\ a + 2 \cdot b + 4 \cdot c = 1502 \\ a + 3 \cdot b + 9 \cdot c = 1504 \end{cases}$$

La solución es: $a = 1500, b = \frac{1}{3}, c = \frac{1}{3}$. Por lo tanto, el polinomio obtenido es: $1500 + \frac{x}{3} + \frac{x^2}{3}$. Y la solución de la media que quieren obtener los participantes es el término independiente del polinomio, luego la media de los sueldos de los participantes es $a = 1500$.

8.4. Multiplicación de secretos

El caso de la multiplicación de secretos es similar al de la suma visto anteriormente, con la diferencia de que la multiplicación necesita un paso adicional que llamamos paso de reducción de grado. Esto se debe a lo siguiente:

En la suma, cada participante realiza con los fragmentos de los secretos la operación de la suma. Por ejemplo, supongamos que los participantes quieren calcular la suma de los secretos s_1 y s_2. Supongamos que $p_1(x)$ y $p_2(x)$ son los polinomios distribuidores de los secretos. Entonces, cada participante P_i calcula la suma $h(i) = p_1(i) + p_2(i)$. Observamos que tanto $p_1(i)$ como $p_2(i)$ son polinomios de grado n (siendo $n + 1$ el número de participantes que se necesitará para recuperar los secretos, suponiendo un esquema umbral $(n + 1, t)$), y al realizar la suma de los polinomios (es decir, al calcular $h(i)$) se obtiene de nuevo un polinomio de grado n. Por lo tanto, obtenemos que $h(x)$ es un polinomio de grado n cuyo término independiente es $s_1 + s_2$.

Sin embargo, no sucede lo mismo en la multiplicación. En este caso, cada participante debe calcular $h(i) = p_1(i) \cdot p_2(i)$. Mientras que el término independiente es $s_1 \cdot s_2$, el grado del polinomio es $2n$. Si $t < 2n + 1$, no habrá puntos suficientes para realizar la interpolación que se necesita para que los participantes recuperen el valor deseado.

Para dar una solución, seguiremos el método propuesto en [7], que aporta tanto el paso de reducción de grado como el de preservación de aleatoriedad del polinomio.

Supongamos que tenemos un esquema umbral $(n + 1, t)$, y que se tienen dos valores secretos s_a y s_b. Los participantes quieren hallar $s_a \cdot s_b$. Para ello, en primer lugar distribuyen los secretos utilizando los polinomios de grado n siguientes:

$$p_a(x) = s_a + u_1 \cdot x + u_2 \cdot x^2 + ... + u_n \cdot x^n$$
$$p_b(x) = s_b + w_1 \cdot x + w_2 \cdot x^2 + ... + w_n \cdot x^n$$

Recordemos que estos polinomios distribuidores tienen como término independiente el secreto que se quiere distribuir, y los demás coeficientes son aleatorios.

Cada participante P_i recibe los fragmentos $p_a(i)$ y $p_b(i)$ y calcula $h(i) = p_a(i) \cdot p_b(i) = h_i$.

Observemos que el polinomio $h(x) = p_a(x) \cdot p_b(x)$ es de la siguiente forma:

$$h(i) = s_a \cdot s_b + z_1 \cdot x + z_2 \cdot x^2 + ... + z_{2n} \cdot x^{2n}$$

donde los coeficientes $z_1, ..., z_{2n}$ dependen de los coeficientes $u_1, ..., u_n$, $w_1, ..., w_n$ de los polinomios p_a y p_b (mientras que el objetivo hubiera sido obtener un polinomio de grado n con el mismo término independiente $s_a \cdot s_b$, pero con coeficientes aleatorios).

Con los valores h_i obtenidos al realizar la multiplicación, cada participante genera un polinomio distribuidor para repartir fragmentos de su valor h_i. De este modo, cada participante genera un polinomio de la forma:

$$g_i(x) = h_i + y_1 \cdot x + y_2 \cdot x^2 + ... + y_n \cdot x^n$$

y cada participante P_j recibe los fragmentos $g_i(j)$ para $i = 1, ..., n + 1$.

A continuación, cada participante P_j calcula:

$$H(j) = \sum_{i=1}^{2n+1} \lambda_i \cdot h_i(j)$$

donde λ_i es la primera fila de la inversa de la matriz de Vandermonde.

Por último se realiza la recuperación del valor deseado (o reconstrucción del resultado) a partir de los valores $H(i)$, en lugar de hacerlo a partir de los valores $h(i)$ como se hubiera hecho en el caso de la suma.

8.4.1. Ejemplo: Multiplicación de secretos

Supongamos que tenemos un esquema de umbral $(2, 3)$, y dos valores secretos $a = 2$ y $b = 3$.

En primer lugar, se realiza la distribución de los valores secretos. Para ello se utilizarán los polinomios distribuidores siguientes:

$p_a(x) = x + 2$

$p_b(x) = 3 \cdot x + 3$

Como en los ejemplos anteriores, supondremos que los segmentos son $\alpha_i = i$ para cada participante P_i, $i = 1, 2, 3$. Teniendo esto en cuenta, veamos cuáles son los fragmentos obtenidos por cada participante al distribuir los secretos a y b.

Fragmentos obtenidos por el participante P_1:

$p_a(1) = 1 + 2 = 3$

$p_b(1) = 3 \cdot 1 + 3 = 6$

Fragmentos obtenidos por el participante P_2:

$p_a(2) = 2 + 2 = 4$

$p_b(2) = 3 \cdot 2 + 3 = 9$

Fragmentos obtenidos por el participante P_3:

$p_a(3) = 3 + 2 = 5$

$p_b(3) = 3 \cdot 3 + 3 = 12$

A continuación, calculan $h(i) = p_a(i) \cdot p_b(i)$

El participante P_1 calcula: $h(1) = 3 \cdot 6 = 18$

El participante P_2 calcula: $h(2) = 4 \cdot 9 = 36$

El participante P_3 calcula: $h(3) = 5 \cdot 12 = 60$

Los valores obtenidos también deben ser compartidos de forma secreta. Se utilizan los polinomios distribuidores siguientes:

$g_1(x) = x + 18$

$g_2(x) = 3 \cdot x + 36$

$g_3(x) = -2 \cdot x + 60$

El participante P_1 obtiene los fragmentos siguientes:

$g_1(1) = 1 + 18 = 19$

$g_2(1) = 3 + 36 = 39$

$g_3(1) = -2 + 60 = 58$

El participante P_2 obtiene los fragmentos siguientes:

$g_1(2) = 2 + 18 = 20$

$g_2(2) = 3 \cdot 2 + 36 = 42$

$g_3(2) = -2 \cdot 2 + 60 = 56$

El participante P_3 obtiene los fragmentos siguientes:

$g_1(3) = 3 + 18 = 21$

$g_2(3) = 3 \cdot 3 + 36 = 45$

$g_3(3) = -2 \cdot 3 + 60 = 54$

La matriz de Vandermonde que necesitaremos en este caso y inversa son:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \xrightarrow{(-1)} \begin{pmatrix} 3 & -3 & 1 \\ -5/2 & 4 & -3/2 \\ 1/2 & -1 & 1/2 \end{pmatrix}$$

Utilizando la primera fila de la matriz de Vandermonde inversa, se calcula $H(i)$:

$H(1) = 3 \cdot 19 - 3 \cdot 39 + 1 \cdot 58 = -2$

$H(2) = 3 \cdot 20 - 3 \cdot 42 + 1 \cdot 56 = -10$

$H(3) = 3 \cdot 21 - 3 \cdot 45 + 1 \cdot 54 = -18$

Por último, se recupera el resultado, para lo que simplemente se resuelve el sistema de ecuaciones siguiente:

$$\begin{cases} m + n + q = -2 \\ m + 2 \cdot n + 4 \cdot q = -10 \\ m + 3 \cdot n + 9 \cdot q = -18 \end{cases}$$

Y obtenemos que la solución es $m = 6$, $n = -8$, $q = 0$, por lo tanto, el valor del resultado deseado es 6 (y comprobamos que la multiplicación de los secretos es $2 \cdot 3 = 6$).

Como el esquema es de umbral $(2, 3)$, también podríamos recuperar el resultado con 2 participantes. Para ello, supongamos que los participantes P_1 y P_2 combinan sus fragmentos de la multiplicación para recuperar el resultado. Para ello deben resolver el sistema siguiente:

$$\begin{cases} m + n = -2 \\ m + 2 \cdot n = -10 \end{cases}$$

La solución del sistema es $m = 6, n = -8$, y el resultado final de la multiplicación de secretos es 6, como se quería.

8.5. Cálculo de la mediana y cuartiles

El objetivo de esta operación es calcular el k-ésimo elemento de dos o más conjuntos de datos confidenciales.

Las aplicaciones de esta operación suelen pertenecer al análisis estadístico. Por ejemplo, calcular la mediana de los salarios de los trabajadores de dos empresas competidoras, sin revelar los salarios a la otra empresa.

La mediana representa el punto medio del conjunto de datos. Es decir, la mitad de los valores del conjunto estarán por debajo de la mediana, y la otra mitad, por encima. La evaluación de la mediana se puede combinar con la de la media para interpretar la distribución de los datos. Si la distribución es simétrica, la media y la mediana tendrán valores similares.

Por otro lado, tenemos los cuartiles, que dividen el conjunto de datos en cuatro partes. El primer cuartil, indica que un tercio de los datos está por debajo de su valor, mientras que tres cuartos se sitúan por encima. El segundo cuartil es equivalente a la mediana, y el tercer cuartil indica que una tercera parte de los datos se sitúan por encima de su valor, mientras que el resto (tres cuartas partes) se sitúa por debajo.

La definición del k-ésimo elemento es la siguiente: "Para un conjunto ordenado S, el k-ésimo elemento es el valor $x \in S$ que está situado en la posición k cuando el conjunto S está ordenado ascendentemente." [15]

En nuestro caso, el conjunto S estará formado por varios subconjuntos, cada uno perteneciente a un participante. Se quiere mantener privada la información de los subconjuntos, de manera que ningún participante pueda saber cuáles son los valores de los subconjuntos de los demás. De esta manera, si los participanes $P_1, P_2, ..., P_t$ poseen los conjuntos $C_1, C_2, ..., C_t$, respectivamente, el objetivo es calcular el k-ésimo elemento de $C_1 \cup C_2 \cup ... \cup C_t$.

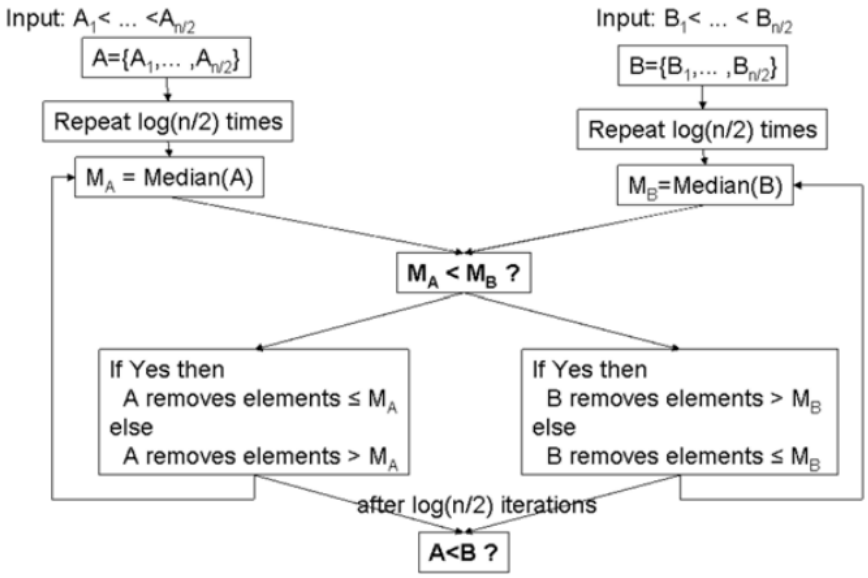

Figura 8.2: Cálculo de la mediana para dos participantes. [15]

En la figura 8.2, se muestra el funcionamiento del cálculo de la mediana para dos participantes, propuesto en [15]. Cada participante tiene como dato de entrada el conjunto ordenado (ascendentemente) de sus valores. Cada uno calcula su valor para la mediana, M_A para el participante A y M_B para el participante B. Comparan estos valores. Si $M_A < M_B$, A elimina de su conjunto los valores menores o iguales que M_A y B elimina de su conjunto los valores menores que M_B. En caso contrario, si $M_A \geq M_B$, A elimina de su conjunto los valores mayores que M_A y B elimina de su conjunto los valores menores o iguales que M_B. Vuelven a realizar la comparación de la media de los nuevos conjuntos, y repiten el proceso hasta que se tenga un único elemento en el conjunto. Para el caso de la mediana, se deben realizar $log(n/2)$ iteraciones.

Este método resulta bastante intuitivo, y eficiente debido a que el número de iteraciones que necesita realizar es de tamaño logarítmico.

Para el caso multipartito, los autores de [15] proponen un método (también aplicable al caso bipartito) que presenta un número de iteraciones logarítmico en M, siendo $M = \beta - \alpha + 1$ y $[\alpha, \beta]$ el rango de valores con los que se trabaja. Este rango es conocido por todos y se establece antes de comenzar el protocolo.

Este método se basa en los pasos siguientes: en primer lugar, se sugiere un valor para el k-ésimo elemento, en segundo lugar, se realiza un cálculo seguro en el que cada participante determina el número de valores menores que este valor propuesto para el k-ésimo elemento, y por último, se actualiza el valor propuesto.

Además, los autores proponen un método seguro frente a adversarios semi-honestos,

junto a la manera de modificarlo para hacerlo seguro frente a adversarios maliciosos. Esta modificación consiste en realizar verificaciones para comprobar que los valores de entrada de los participantes son coherentes con los datos que habían introducido anteriormente y con los pasos realizados.

Utilizaremos el protocolo FIND-RANKED-ELEMENT-MULTIPARTY de [15], que nos permite calcular el elemento de un conjunto de datos de tamaño n que se encuentra en la posición k, $(1 \leq k \leq n)$.

De esta manera, podremos calcular la mediana, que se encuentra en la posición $\lceil n/2 \rceil$, si n es impar, y haciendo la media del valor en la posición $n/2$ y el valor siguiente $(n/2)+1$ si n es par.

Para calcular los cuartiles, se procede del mismo modo que para hallar la mediana, pero teniendo en cuenta que el primer cuartil estaría en la posición $1/4$ y el tercer cuartil en la posición $3/4$.

Capítulo 9

Conclusiones

En este trabajo hemos visto una forma de realizar operaciones con valores privados pertenecientes a distintos participantes manteniendo la confidencialidad de estos valores, lo que se denomina computación multiparte segura.

Además, hemos visto que hay diversas formas de abordar la computación multiparte: los circuitos cifrados de Yao, el cifrado homomórfico y los esquemas para compartir secretos (basados en el esquema de Shamir). Estas tres herramientas proveen de diversas formas de resolver problemas, e independientemente de cuál se utilice, el objetivo es mantener la privacidad de los datos mientras se establece un equilibrio entre la seguridad requerida y la complejidad, adaptándose a cada caso concreto.

En nuestro caso, nos hemos centrado en el esquema de Shamir para estudiar varias operaciones básicas (como la suma o la multiplicación de valores secretos) que tienen diversas aplicaciones, por ejemplo en subastas o votaciones.

La computación multiparte debe mejorar sobre todo en términos de eficiencia. Sin embargo, es una opción que ofrece seguridad y tiene diversas utilidades. Además, a partir de las operaciones básicas se pueden realizar operaciones más complejas. También tiene la ventaja de presentar seguridad incondicional (o teórica), en lugar de seguridad computacional (o práctica), por lo que en un futuro esta tecnología podrá obtener mayor relevancia, dentro del ámbito de la seguridad de la información y, en concreto, a la hora de operar con información privada.

9.1. Líneas de trabajo futuro

Respecto a los contenidos planteados en este proyecto, se sugieren algunas ideas para posibles ampliaciones futuras:

- Las operaciones realizadas se pueden mejorar implementando medidas de seguridad contra adversarios maliciosos. Como ya hemos visto, los participantes maliciosos no siguen las normas del protocolo y pueden introducir datos erróneos, lo que provoca que el resto de participantes no obtenga un resultado correcto para la operación que se desea calcular. Para dar una solución a este problema, se pueden realizar validaciones sobre las entradas de los participantes y realizar un mayor control sobre los datos introducidos.
- Se puede ampliar el proyecto incluyendo implementaciones de más operaciones. Teniendo la suma y la multiplicación de secretos, se puede optar por otra de las operaciones básicas de la computación multiparte: la comparación de dos elementos. A partir de ella se podría realizar la ordenación de un conjunto de elementos.
- Ya que en este trabajo nos hemos centrado en el esquema de Shamir, una posible línea de trabajo futuro consistiría en profundizar en los circuitos cifrados de Yao o en el cifrado homomórfico.

Parte II

Bibliografía

Bibliografía

[1] Anaconda. *https://www.anaconda.com/*. (Visitado 03-05-2021).

[2] Judith Bar-Ilan y Donald Beaver. "Non-cryptographic fault-tolerant computing in constant number of rounds of interaction". En: *Proceedings of the eighth annual ACM Symposium on Principles of distributed computing (pp. 201-209)* (1989).

[3] George Robert Blakley. "Safeguarding cryptographic keys". En: *International Workshop on Managing Requirements Knowledge, (pp. 313)*. IEEE Computer Society. (1979).

[4] Karthik A. Jagadeesh, David J. Wu, Johannes A. Birgmeier, Dan Boneh y Gill Bejerano. "Deriving genomic diagnoses without revealing patient genomes". En: *Science, American Association for the Advancement of Science, (pp. 692-695)* (2017).

[5] Ivan Damgård y Jesper Buus Nielsen. "Scalable and unconditionally secure multiparty computation". En: *Annual International Cryptology Conference, (pp. 572-590). Springer.* (2007).

[6] Duality. *https://dualitytech.com/*. (Visitado 01-06-2021).

[7] Rosario Gennaro, Michael O Rabin y Tal Rabin. "Simplified VSS and fast-track multiparty computations with applications to threshold cryptography". En: *Proceedings of the seventeenth annual ACM symposium on Principles of distributed computing, (pp. 101-111).* (1998).

[8] Craig Gentry. *A fully homomorphic encryption scheme*. Stanford university, (2009).

[9] David Evans, Vladimir Kolesnikov y Mike Rosulek. "A pragmatic introduction to secure multi-party computation". En: *Foundations and Trends in Privacy and Security 2.2-3* (2017).

[10] Dan Bogdanov, Liina Kamm, Sven Laur, y Ville Sokk. "Rmind: a tool for cryptographically secure statistical analysis". En: *IEEE Transactions on Dependable and Secure Computing (pp. 481-495).* 15.3 (2016).

[11] Dan Bogdanov, Sven Laur, y Jan Willemson. "Sharemind: A framework for fast privacy-preserving computations". En: *European Symposium on Research in Computer Security, (pp. 192-206)* (2008).

[12] Hsiao-Ying Lin y Wen-Guey Tzeng. "An efficient solution to the millionaires' problem based on homomorphic encryption". En: *International Conference on Applied Cryptography and Network Security, (pp. 456-466).* Springer. (2005).

[13] Yehuda Lindell. "Secure Multiparty Computation (MPC)". En: *IACR Cryptol. ePrint Arch.* (2020).

[14] Jian Liu, Mika Juuti, Yao Lu y Nadarajah Asokan. "Oblivious Neural Network Predictions via MiniONN transformations". En: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, (pp. 619-631)* (2017).

[15] Gagan Aggarwal, Nina Mishra y Benny Pinkas. "Secure computation of the median (and other elements of specified ranks)". En: *Journal of cryptology, 23(3), 373-401.* (2010).

[16] Liina Kamm, Dan Bogdanov, Alisa Pankova, y Riivo Talviste. "Statistical Analysis Methods Using Secure Multiparty Computation". En: *Applications of Secure Multiparty Computation, (pp.58)* (2015).

[17] Peter Bogetoft, Dan Lund Christensen, Ivan Damgård, Martin Geisler, Thomas Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael Schwartzbach, y Tomas Toft. "Secure multiparty computation goes live". En: *International Conference on Financial Cryptography and Data Security, (pp. 325-343)* (2008).

[18] Sepior. *https://sepior.com/*. (Visitado 01-06-2021).

[19] Adi Shamir. "How to Share a Secret". En: *Massachusetts Institute of Technology* (1979).

[20] Mihaela Ion, Ben Kreuter, Erhan Nergiz, Sarvar Patel, Shobhit Saxena, Karn Seth, David Shanahan y Moti Yung. "Private Intersection-Sum Protocol with Applications to Attributing Aggregate Ad Conversions". En: *IACR Cryptol. ePrint Arch.* (2017).

[21] Dan Bogdanov y el equipo de Sharemind. "Smarter decisions with no privacy breaches - practical secure computation for governments and companies". En: *https://rwc.iacr.org/2015/Slides/RWC-2015-Bogdanov-final.pdf* (2015). (Visitado 02-06-2021).

[22] Duality SecurePlus Statistics. *https://dualitytech.com/duality-secureplus-statistics/*. (Visitado 01-06-2021).

[23] Riivo Talviste y col. "Applying secure multi-party computation in practice". Tesis doct. PhD Thesis, University of Tartu, (2016).

[24] Unbound. *https://www.unboundsecurity.com/*. (Visitado 01-06-2021).

[25] Andrei Lapets, Frederick Jansen, Kinan Dak Albab, Rawane Issa, Lucy Qin, Mayank Varia y Azer Bestavros. "Accessible privacy-preserving web-based data analysis for assessing and addressing economic inequalities". En: *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies* (2018).

[26] Alan F. Westin. “Privacy and freedom”. En: *Washington and Lee Law Review* 25.1 (1968).

[27] Andrew C. Yao. “Protocols for Secure Computations”. En: *23rd annual symposium on foundations of computer science (sfcs 1982) (pp. 160-164). IEEE.* (1982).

Parte III

Apéndices

Apéndice A

Manual de Usuario

En esta sección, se describen en primer lugar los entregables del proyecto y a continuación su funcionamiento, para que pueda servir a modo de manual de usuario. Estos entregables son los programas en Python realizados para implementar las operaciones descritas en este documento. Más adelante, en la sección B, se encuentran las pruebas de ejecución correspondientes a los programas realizados en Python.

A.1. Entregables del proyecto

En los entregables adjuntos a este trabajo se encuentran dos carpetas: "operaciones" y "operaciones_distribuidas". Para su ejecución, se debe ejecutar el menú correspondiente a cada carpeta, que permite elegir la operación deseada. Los contenidos de las carpetas son los archivos Python siguientes:

- **operaciones:** contiene las operaciones explicadas anteriormente. Observamos que las operaciones se realizan teniendo un único programa en ejecución todos los valores. El objetivo es ver el funcionamiento de los protocolos de computación multiparte de una forma sencilla.
  - **menu_operaciones.py:** menú principal que perite elegir la operación a realizar.
  - **op_problema_generico.py:** realiza la operación de suma, media y combinación lineal.
  - **op_multiplicacion_secretos.py:** realiza la operación de multiplicación de secretos.
  - **op_problema_mediana_cuartiles.py:** permite calcular la mediana, primer y tercer cuartiles, y el valor de un elemento en otra posición cualquiera.

- **operaciones_distribuidas:** contiene las mismas operaciones que en el caso anterior, pero se ejecutan de forma distribuida. Es decir, cada programa en ejecución sería equivalente a un participante. No hay un único programa en ejecución que vea todos los datos en claro.

  - **menu_operaciones_distribuidas.py:** menú principal que permite elegir la operación a realizar.
  - **op_generico_distribuido.py:** permite realizar la suma, media y combinación lineal.
  - **op_multiplicacion_distribuido.py:** realiza la multiplicación de valores secretos.
  - **op_mediana_cuartiles_distribuido.py:** permite calcular la mediana, primer y tercer cuartil, y el valor de una posición cualquiera. Requiere una cantidad elevada de intercambio de mensajes entre los participantes, por lo que requiere un mayor tiempo de ejecución en comparación con el resto de operaciones.

A.2. Manual de uso

- **Instalación**

  Para ejecutar los archivos es necesario instalar Python. Se recomienda instalar el entorno Anaconda, ya que ha sido el utilizado en la realización de este trabajo por incluir tanto Spyder como Jupyter. Este entorno se puede instalar desde la web de Anaconda [1], y se encuentra disponible para los sistemas operativos Windows, macOS y Linux.

  También se deben tener descargadas las carpetas mencionadas en el apartado A.1.

- **Ejecución de los programas**

  Los programas se pueden ejecutar desde Spyder, desde Jupyter o desde la terminal de comandos.

  El código que se debe ejecutar es el menú de operaciones. Desde este menú, se accederá al resto de operaciones de la carpeta correspondiente.

  - Desde **Spyder**: abrir el archivo correspondiente al menú de operaciones en Spyder y ejecutarlo (`Run file (F5)`) en la consola de Python.
  - Desde **Jupyter**: lanzamos JupyterLab desde Anaconda y subimos los archivos de Python mencionados en el apartado A.1. A continuación, creamos un notebook de Jupyter y escribimos y ejecutamos las líneas de código siguientes (indicando el menú de operaciones que queramos ejecutar):

```
[1]: import menu_operaciones
[2]: menu_operaciones.main()
```

- Desde la **terminal de comandos:** abrimos la terminal y ejecutamos el comando "`python [ruta del archivo a ejecutar]`". Por ejemplo:

  ```
  python menu_operaciones.py
  ```

Al elegir la opción deseada, se deben seguir los pasos indicados durante la ejecución. Estos pasos consistirán en introducir por teclado los datos indicados.

- **Descripción de los pasos a seguir durante las operaciones**

  Aquí se describen los pasos a seguir en cada operación:

  - **Media, suma y combinación lineal**
    1. Se pide la ruta del fichero CSV del que se leerán los datos. A continuación, se mostrarán las columnas del fichero CSV y se debe introducir el nombre de una de las columnas. Se recomienda que el fichero CSV tenga los nombres de las columnas escritos en la primera fila. Cada dato de la columna elegida corresponderá a un valor secreto de un participante. De este modo, se obtiene también el número de participantes.
    2. Se pide un número máximo de cifras que tendrá el número primo p. Las operaciones se realizarán módulo p.
    3. Se muestra un menú de operaciones para elegir media, suma o combinación lineal. Para la media y para la suma, simplemente se elige la opción correspondiente y se obtiene el resultado. Para la combinación lineal, es necesario volver a elegir un fichero CSV y una columna, cuyos valores serán los coeficientes de la combinación lineal.
  - **Multiplicación**
    1. Se pide el número de participantes totales (t).
    2. Se pide el número mínimo de participantes necesarios para recuperar el resultado (n). Debe cumplirse $2 \cdot (n-1)+1 < t$. El grado de los polinomios distribuidores será $n-1$.
    3. Se pide un número máximo de cifras que tendrá el número primo p. Las operaciones se realizarán módulo p.
    4. Se pide introducir los dos valores secretos a multiplicar. Deben introducirse separados por una coma, por ejemplo: `5, 2`.
    5. Se obtiene el resultado.
  - **Mediana y cuartiles**
    1. Se pide la ruta del fichero CSV del que se leerán los datos. En este caso, cada columna corresponderá a un conjunto de valores secretos pertenecientes a un participante.

Además, se realiza una suma de secretos tomando como valores secretos los tamaños de los conjuntos. De este modo, se calcula el número total de elementos.

2. Se muestra un menú de operaciones, donde se debe elegir entre la mediana, primer cuartil, tercer cuartil, u otra posición. Para los tres primeros, simplemente se introduce el número correspondiente a la opción deseada y se obtiene el resultado. Para la cuarta opción, también se pide introducir el número de la posición que se desea calcular.

- **Media, suma y combinación lineal (operaciones distribuidas)**

1. Se pide introducir el número de participantes que realizarán la operación.
2. Se pide introducir un ID de usuario. El ID debe ser un número natural que será usado como segmento durante los cálculos. Además, este ID será conocido por el resto de participantes. Este ID debe generarse de forma incremental de modo que se tenga participante 1 con ID = 1, participante 2 con ID = 2, etc.
3. El participante con ID = 1 debe introducir el número máximo de cifras que tendrá el número primo p. Las operaciones se realizarán en módulo p. El resto de participantes deben introducir el número p que reciban del participante 1.
4. Cada participante introduce su valor secreto.
5. Se generan los polinomios distribuidores de los secretos y se generan los valores a enviar al resto de participantes. Cada uno debe introducir los fragmentos recibidos en el orden correspondiente.
6. Se muestra un menú de operaciones. Se debe seleccionar la media, suma o combinación lineal. Al elegir una de las operaciones, se genera un resultado parcial a partir de los fragmentos recibidos en el paso anterior. Además, para la combinación lineal se pide introducir los coeficientes separados por comas. Para terminar, deben enviar el resultado parcial a los demás participantes y así recuperarán el resultado final.

- **Multiplicación (operaciones distribuidas)**

1. Se pide introducir el número de participantes que realizarán la operación (t).
2. Se pide el número mínimo de participantes necesarios para recuperar el resultado (n). Debe cumplirse $2 \cdot (n-1)+1 < t$. El grado de los polinomios distribuidores será $n-1$.
3. Se pide introducir un ID de usuario. El ID debe ser un número natural que será usado como segmento durante los cálculos. Además, este ID será conocido por el resto de participantes. Este ID debe generarse de forma incremental de modo que se tenga participante 1 con ID = 1, participante 2 con ID = 2, etc.

4. El participante con ID = 1 debe introducir el número máximo de cifras que tendrá el número primo p. Las operaciones se realizarán en módulo p. El resto de participantes deben introducir el número p que reciban del participante 1.
5. Los participantes 1 y 2 deben introducir los valores secretos que se van a multiplicar. A partir de estos valores, se generan los polinomios distribuidores y se generan los fragmentos a distribuir entre los participantes.
6. Cada participante introduce los fragmentos recibidos en el orden correspondiente.
7. Se genera un resultado parcial que debe enviarse a los demás participantes. Cada uno introduce los resultados parciales recibidos en el orden correspondiente y se obtiene el resultado final.

- **Mediana y cuartiles (operaciones distribuidas)**
  1. Se pide introducir el número de participantes que realizarán la operación.
  2. El participante con ID = 1 debe introducir el número máximo de cifras que tendrá el número primo p. Las operaciones se realizarán en módulo p. El resto de participantes deben introducir el número p que reciban del participante 1.
  3. El participante con ID = 1 debe introducir el número máximo de cifras que tendrá el número primo p. Las operaciones se realizarán en módulo p. El resto de participantes deben introducir el número p que reciban del participante 1.
  4. Se pide introducir la ruta del fichero CSV del que se leerán los datos, y una columna perteneciente al fichero. Los datos de dicha columna formarán el conjunto de valores secretos del participante.
  5. Se realiza una suma de secretos a partir del número de valores que posee cada participante, así que deben enviar los fragmentos generados a los demás participantes, y deben introducir los fragmentos recibidos en el orden correspondiente. A continuación, envían e introducen los resultados parciales y consiguen el número total de elementos con los que se va a operar.
  6. Se muestra un menú de operaciones. Se debe elegir se debe elegir entre la mediana, primer cuartil, tercer cuartil, u otra posición. Se debe introducir la opción deseada. Además, en la cuarta opción, se debe introducir el la posición que se desea calcular. A continuación, se realizan varias sumas de secretos, por lo que los participantes deben enviar e introducir los valores recibidos hasta que se llegue al resultado final.

Apéndice B

Pruebas de ejecución

Se han realizado implementaciones en Python de las operaciones mencionadas en este trabajo. Se trata de operaciones básicas que se pueden aplicar a una gran diversidad de contextos. Por ejemplo, en la operación del cálculo de la mediana y los cuartiles, se emplea la operación de la suma de secretos para obtener el número total de elementos con los que se trabaja.

Como hemos visto en el apartado anterior, tenemos dos carpetas de operaciones. En una de las carpetas, las operaciones se realizan de manera distribuida, como ocurriría en un caso real. En el caso de las operaciones que se realizan de forma distribuida, el tiempo de ejecución es mayor, sobre todo debido al tiempo empleado en las comunicaciones entre los participantes.

A continuación, se muestran las pruebas de ejecución, correspondientes a los archivos Python descritos en el apartado A.1.

B.1. Media, suma y combinación lineal

Esta primera funcionalidad permite realizar la media, suma u otra combinación lineal de los valores de los participantes.

Los valores de los participantes se obtienen de un fichero en formato CSV, de manera que cada valor corresponda a un participante. El primer paso es elegir el fichero del que se leerán los datos. También se permite elegir una columna del fichero, de modo que el número de participantes será el mismo que la longitud de la columna.

A continuación, se pide el número de cifras que tendrá el número primo p que se va a utilizar (las operaciones se realizarán modulo p). Este primo p debe ser mayor que el

número de participantes. Si se quiere evitar que se realicen reducciones modulares, se debe establecer un número primo p lo suficientemente grande.

Se muestra un submenú con las operaciones suma, media y combinación lineal.

Para la realización de la operación, se establecen unos números que llamamos segmentos, y son los valores en los que se evaluarán los polinomios distribuidores. Cada participante debe tener un segmento distinto, en este caso se generan de forma incremental. Estos segmentos son valores públicos, es decir, todos saben qué segmento corresponde a cada participante.

Cada participante genera su polinomio distribuidor, con término independiente su valor secreto, y el resto de coeficientes aleatorios. Posteriormente, utiliza este polinomio para distribuir su valor secreto. Para ello, evalúa el polinomio en los puntos de los segmentos y envía los fragmentos obtenidos al resto de participantes.

Una vez distribuidos los fragmentos, se realiza la operación deseada. Cada participante realiza la operación con los fragmentos que le han enviado los demás, más el de su propio secreto.

Con los resultados obtenidos por los participantes, se resuelve el sistema de ecuaciones para obtener el resultado final.

Las operaciones a realizar funcionan de la siguiente manera: todas se plantean con una combinación lineal. Para la media, todos los coeficientes son $1/n$, siendo n el número de participantes. Para la suma, todos los coeficientes son 1. Por último, para la combinación lineal, se permite introducir los coeficientes por teclado.

B.1.1. Pruebas de ejecución: media, suma y combinación lineal

A continuación se muestra la salida obtenida en Jupyter al ejecutar estas operaciones:

```
************************
   MENÚ PRINCIPAL
************************

     1 - Media, suma y combinación lineal de valores secretos
     2 - Multiplicación de secretos
     3 - Mediana y cuartiles
     0 - Salir
```

```
Introduzca el número de la opción que desea realizar:

 1


Introduzca la ruta del fichero del que se leerán los datos:

 C:/Users/Alicia/Desktop/datos_generico.csv



Las columnas existentes son: ['edad' 'coef']

Introduzca el nombre de la columna de la que se leerán los datos:

 edad


secretos
[1, 1, 1, 2, 1, 1, 1]

El número de participantes es: 7


Introduzca el número de cifras que tendrá en número primo p:

 5


p = 10427


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:
```

```
 1


MEDIA

RESULTADO 1.1429


   ************************
      MENÚ DE OPERACIONES
   ************************

         1 - Media
         2 - Suma
         3 - Otra combinación lineal (requiere introducir los coeficientes)
         0 - Salir

Introduzca el número de la opción que desea realizar:

 2


SUMA

RESULTADO 8.0


   ************************
      MENÚ DE OPERACIONES
   ************************

         1 - Media
         2 - Suma
         3 - Otra combinación lineal (requiere introducir los coeficientes)
         0 - Salir

Introduzca el número de la opción que desea realizar:

 3


Introduzca la ruta del fichero del que se leerán los coeficientes:
```

```
 C:/Users/Alicia/Desktop/datos_generico.csv


Las columnas existentes son: ['edad' 'coef']

Introduzca el nombre de la columna de la que se leerán los datos:

 coef


coeficientes
[2, 1, 2, 3, 1, 2, 4]

COMBINACIÓN LINEAL

RESULTADO 18.0


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 3


Introduzca la ruta del fichero del que se leerán los coeficientes:

 C:/Users/Alicia/Desktop/datos_generico.csv


Las columnas existentes son: ['edad' 'coef']

Introduzca el nombre de la columna de la que se leerán los datos:
```

```
 edad


coeficientes
[1, 1, 1, 2, 1, 1, 1]

COMBINACIÓN LINEAL

RESULTADO 10.0
```

Comprobemos que los resultados obtenidos son correctos. Los valores leídos del archivo (que serán los datos privados de los participantes) son los siguientes: [1,1,1,2,1,1,1], por lo tanto, se tienen 7 participantes. El número primo que se utilizará es $p = 10427$, que es lo suficientemente grande para evitar reducciones modulares con los datos de los participantes que se tienen en este caso.

Al elegir en el submenú la opción de la media, se obtiene el valor 1.1429. Comprobemos que es el valor correcto:

$$\frac{s_1 + s_2 + s_3 + s_4 + s_5 + s_6 + s_7}{numParticipantes} = \frac{1+1+1+2+1+1+1}{7} = \frac{8}{7} = 1'1429$$

Al elegir la segunda opción del menú, obtenemos la suma de todos los valores secretos de los participantes. En la ejecución se obtiene que la suma es 8, comprobemos que el resultado obtenido es correcto:

$$1 + 1 + 1 + 2 + 1 + 1 + 1 = 8$$

La tercera opción permite realizar otra combinación lineal, para lo que hay que elegir los coeficientes. Estos se obtienen de un fichero, del mismo modo que se han obtenido los valores de los participantes. Para la primera combinación lineal, los coeficientes utilizados son [2,1,2,3,1,2,4], por lo tanto, la operación realizada es la siguiente:

$$2 \cdot 1 + 1 \cdot 1 + 2 \cdot 1 + 3 \cdot 2 + 1 \cdot 1 + 2 \cdot 1 + 4 \cdot 1 = 18$$

Para la segunda combinación lineal, los coeficientes utilizados son [1,1,1,2,1,1,1], luego la combinación realizada es la siguiente:

$$1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 + 2 \cdot 2 + 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 = 10$$

Observamos que ambos valores coinciden con los obtenidos en la ejecución del programa.

B.2. Multiplicación

Esta solución permite multiplicar dos valores secretos. Permite introducir tanto el número de participantes total (t) como el umbral con el que se quiere trabajar, con la restricción de que si el umbral es $(n+1, t)$, debe cumplirse $2 * n + 1 \leq t$.

B.2.1. Pruebas de ejecución: Multiplicación

A continuación, se muestran dos ejemplos de ejecución de la multiplicación de secretos:

```
**************************
   MENÚ PRINCIPAL
**************************

      1 - Media, suma y combinación lineal de valores secretos
      2 - Multiplicación de secretos
      3 - Mediana y cuartiles
      0 - Salir

Introduzca el número de la opción que desea realizar:

 2

Introduzca el número de participantes:

 3

Introduzca el número de participantes necesarios para recuperar el resultado:

 2

gradoPol = 1

Introduzca el número de cifras que tendrá en número primo p:

 5
```

```
p = 14621

Introduzca los dos valores secretos separados por comas

 4,3


RESULTADO: 12.0
```

Aquí termina el primer ejemplo. En este caso, se tienen 3 participantes en total, y se necesitan al menos 2 participantes para recuperar el resultado, es decir, trabajamos con un esquema de umbral (2,3). Los polinomios distribuidores serán de grado 1. El número primo que se utiliza es $p = 14621$, lo suficientemente grande para evitar reducciones modulares. Con los valores secretos 4 y 3, se obtiene que la multiplicación es 12, luego el resultado es correcto.

Veamos otro ejemplo:

```
   ************************
      MENÚ PRINCIPAL
   ************************

        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 2

Introduzca el número de participantes:

 7

Introduzca el número de participantes necesarios para recuperar el resultado:

 4

gradoPol = 3
```

```
Introduzca el número de cifras que tendrá en número primo p:

 5

p = 51199

Introduzca los dos valores secretos separados por comas

 3,5


RESULTADO: 15.0
```

En este caso, hay 7 participantes en total, luego los secretos se distribuirán el 7 fragmentos. Se necesitan al menos 4 fragmentos para recuperar el resultado, así que se trata de un esquema de umbral (4,7). De nuevo, se trabaja con un número primo elevado $p = 51199$, en comparación con los valores secretos 3 y 5. Se obtiene que el resultado de multiplicar estos valores es 15.

B.3. Mediana y cuartiles

En esta sección proporcionamos una solución para que varios participantes, cada uno con un conjunto de datos, puedan calcular la mediana, cuartiles y otra posición cualquiera.

Además, para calcular el número total de elementos, se hace una llamada a una función del programa del apartado B.1. De este modo, se realiza una suma de secretos donde los valores secretos son los números de elementos que hay en cada conjunto.

Los valores secretos de los participantes se leen de un fichero CSV, donde cada columna de datos corresponde a un participante. De este modo, cada participante tiene un conjunto de datos. El número de participantes es igual al número de columnas.

B.3.1. Pruebas de ejecución: mediana y cuartiles

A continuación, se muestra un ejemplo de ejecución de esta operación:

```
************************
   MENÚ PRINCIPAL
```

```
   ************************

        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:


 3



Introduzca la ruta del fichero del que se leerán los datos:


 C:/Users/Alicia/Desktop/datos_mediana.csv



datos columna Part_A
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

datos columna Part_B
[20, 21, 25, 24, 25, 45, 45, 65, 56, 55, 36, 23, 36, 34]

datos columna Part_C
[20, 40, 50, 10, 22, 23, 14, 52, 55, 99, 8, 95, 5, 5]

conjuntos [[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], [20, 21, 25, 24, 25,
45, 45, 65, 56, 55, 36, 23, 36, 34], [20, 40, 50, 10, 22, 23, 14, 52, 55, 99, 8,
95, 5, 5]]

El número de participantes es: 3

Conjuntos ordenados de los participantes [[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,
12, 13, 14], [20, 21, 23, 24, 25, 25, 34, 36, 36, 45, 45, 55, 56, 65], [5, 5, 8,
10, 14, 20, 22, 23, 40, 50, 52, 55, 95, 99]]

Longitudes de los conjuntos [14, 14, 14]

SUMA

El número total de elementos es: 42
```

```
   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


MEDIANA (SEGUNDO CUARTIL)

RESULTADO: 20.5


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 2


PRIMER CUARTIL

RESULTADO: 8
```

```
   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 3


TERCER CUARTIL

RESULTADO: 40


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 4

Introduzca la posición que desea calcular:

 4


POSICIÓN k = 4
```

```
RESULTADO: 4


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 4

Introduzca la posición que desea calcular:

 2


POSICIÓN k = 2

RESULTADO: 2


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 4
```

```
Introduzca la posición que desea calcular:

 40


POSICIÓN k = 40

RESULTADO: 65
```

Como ya hemos mencionado, los datos se leen de un fichero CSV. Se leen los datos de las columnas y estos serán los conjuntos de datos privados de los participantes. En este caso hay 3 columnas, así que tenemos 3 participantes. Se calcula el número total de elementos utilizando la suma de secretos. El valor de la mediana que se ha calculado es 20'5. Como tenemos un número par de valores (42 en este caso) la mediana es la media de los valores en la posición 20 y 21 (los valores centrales: $42/2 = 21$). En este caso, los valores en las posiciones 20 y 21 (considerando la unión de los tres conjuntos) son 20 y 21, luego la mediana es $(20 + 21)/2 = 20{,}5$.

Para calcular el primer cuartil, debemos hallar el valor en la posición $42/4$, en este caso, la posición 10 corresponde al valor 8.

Para calcular el tercer cuartil, debemos hallar el valor en la posición $3 \cdot 42/4$. En este caso, la posición 31 corresponde al valor 40.

Por último, se calculan los valores correspondientes a las posiciones 2, 4 y 40. Se obtienen los valores 2, 4 y 65, respectivamente.

B.4. Cálculos distribuidos: Media, suma y combinación lineal

En esta operación, se permite introducir el número de participantes que van a colaborar y permite que cada participante introduzca su valor secreto.

Además, en lugar de generar los segmentos de forma iterativa como se hacía en el apartado B.1, cada participante debe introducir su propio segmento. En este caso, suponemos que cada participante P_i tiene el segmento i, y debe existir un participante P_1 que actuará como "distribuidor", en el sentido de que será el encargado de generar el número primo p. El resto de operaciones las realizará cada participante. El cálculo del resultado final lo podría realizar también el participante con el rol de distribuidor y después enviarlo a

los demás participantes, pero en este caso cada participante calcula el resultado final (ya que tiene todos los datos necesarios para ello). De este modo, evitamos que el distribuidor se quede con el resultado sin enviárselo a los demás, y se garantiza que todos reciben el resultado.

A continuación, se muestran los pasos que se realizan al ejecutar el programa y las pruebas de ejecución correspondientes a cada paso:

1. Se solicita el número de participantes que van a colaborar, y se pide que cada participante introduzca su "ID " de participante. Este ID se utilizará como segmento para evaluar los polinomios distribuidores, por lo que serán números conocidos por todos. Se supondrá que el participante P_i tendrá el segmento i (recordemos que en B.1 estos segmentos se generaban de forma incremental. Si los fragmentos tuvieran valores distintos, deberían introducirse por teclado).

2. Los segmentos anteriores también se utilizan para identificar al participante P_1 que se encarga de generar el número primo p que se utilizará a lo largo del protocolo. Debe compartir este número con los demás participantes, que lo introducen por teclado.

3. Cada participante introduce su valor secreto y genera el polinomio de grado $n-1$ que utilizará para distribuir su secreto (se utilizará un esquema umbral n, n donde $n = numParticipantes$).

4. Cada participante evalúa el polinomio que ha generado en los valores correspondientes a los segmentos de todos los participantes (recordemos que los segmentos son valores públicos conocidos por todos).

5. Cada participante envía los fragmentos de su secreto a los participantes correspondientes.

6. Una vez que los participantes tienen un fragmento de cada valor secreto, los utilizan para realizar la operación elegida, con lo que cada uno obtendrá un resultado parcial.

7. Haciendo públicos los resultados parciales, recuperan el resultado final.

A continuación, se muestra un ejemplo de esta operación distribuida, indicando la parte correspondiente a cada participante. Al realizar los cálculos de forma distribuida podemos ver el paso de mensajes que tiene lugar entre los participantes.

Para este ejemplo tenemos 3 participantes:

- **Participante 1**

```
   ************************
      MENÚ PRINCIPAL
   ************************

        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


Introduzca el número de participantes:

 3


El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 1


PARTICIPANTE 1


Introduzca el número de cifras que tendrá en número primo p:

 4


p 4211

Introduzca su valor secreto:

 2


polinomio [2, 2629, 2295]
```

```
Envíe el valor 715 al participante 1

Envíe el valor 1807 al participante 2

Envíe el valor 3278 al participante 3

Introduzca el fragmento recibido del participante 1

 715


Introduzca el fragmento recibido del participante 2

 3502


Introduzca el fragmento recibido del participante 3

 3393


fragmentos [715, 3502, 3393]


   *************************
      MENÚ DE OPERACIONES
   *************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


MEDIA

Su resultado parcial para la operación es: 2536.666667

Introduzca el resultado parcial correspondiente al participante 1
```

```
 2536.666667

Introduzca el resultado parcial correspondiente al participante 2

 1964.666667

Introduzca el resultado parcial correspondiente al participante 3

 2498.333333

RESULTADO 3.3333000000002357


   *************************
      MENÚ DE OPERACIONES
   *************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 2


SUMA

Su resultado parcial para la operación es: 3399

Introduzca el resultado parcial correspondiente al participante 1

 3399

Introduzca el resultado parcial correspondiente al participante 2

 1683

Introduzca el resultado parcial correspondiente al participante 3

 3284
```

```
RESULTADO 10.0


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 3

Introduzca 3 coeficientes separados por comas

 2,3,1


COMBINACIÓN LINEAL

Su resultado parcial para la operación es: 2696

Introduzca el resultado parcial correspondiente al participante 1

 2696

Introduzca el resultado parcial correspondiente al participante 2

 3774

Introduzca el resultado parcial correspondiente al participante 3

 3252

RESULTADO 18.0
```

- **Participante 2**

```
   ************************
      MENÚ PRINCIPAL
```

```
   ************************

        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


Introduzca el número de participantes:

 3

El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 2


PARTICIPANTE 2

Introduzca el número primo que se utilizará en los cálculos:

 4211


Introduzca su valor secreto:

 3


polinomio [3, 612, 2887]

Envíe el valor 3502 al participante 1

Envíe el valor 142 al participante 2

Envíe el valor 2556 al participante 3
```

```
Introduzca el fragmento recibido del participante 1

 1807

Introduzca el fragmento recibido del participante 2

 142

Introduzca el fragmento recibido del participante 3

 3945

fragmentos [1807, 142, 3945]


   *************************
      MENÚ DE OPERACIONES
   *************************

         1 - Media
         2 - Suma
         3 - Otra combinación lineal (requiere introducir los coeficientes)
         0 - Salir

Introduzca el número de la opción que desea realizar:


 1


MEDIA

Su resultado parcial para la operación es: 1964.666667

Introduzca el resultado parcial correspondiente al participante 1

 2536.666667


Introduzca el resultado parcial correspondiente al participante 2

 1964.666667
```

```
Introduzca el resultado parcial correspondiente al participante 3

 2498.333333

RESULTADO 3.3333000000002357


   *************************
      MENÚ DE OPERACIONES
   *************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 2


SUMA

Su resultado parcial para la operación es: 1683

Introduzca el resultado parcial correspondiente al participante 1

 3399


Introduzca el resultado parcial correspondiente al participante 2

 1683

Introduzca el resultado parcial correspondiente al participante 3

 3284

RESULTADO 10.0


   *************************
      MENÚ DE OPERACIONES
```

```
   ************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 3

Introduzca 3 coeficientes separados por comas

 2,3,1


COMBINACIÓN LINEAL

Su resultado parcial para la operación es: 3774

Introduzca el resultado parcial correspondiente al participante 1

 2696

Introduzca el resultado parcial correspondiente al participante 2

 3774


Introduzca el resultado parcial correspondiente al participante 3

 3252

RESULTADO 18.0
```

- **Participante 3**

```
   ************************
      MENÚ PRINCIPAL
   ************************
```

```
        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


Introduzca el número de participantes:

 3


El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 3


PARTICIPANTE 3

Introduzca el número primo que se utilizará en los cálculos:

 4211

Introduzca su valor secreto:
 5

polinomio [5, 595, 2793]

Envíe el valor 3393 al participante 1

Envíe el valor 3945 al participante 2

Envíe el valor 1661 al participante 3

Introduzca el fragmento recibido del participante 1

 3278
```

```
Introduzca el fragmento recibido del participante 2

 2556


Introduzca el fragmento recibido del participante 3

 1661


fragmentos [3278, 2556, 1661]


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


MEDIA

Su resultado parcial para la operación es: 2498.333333

Introduzca el resultado parcial correspondiente al participante 1

 2536.666667


Introduzca el resultado parcial correspondiente al participante 2

 1964.666667


Introduzca el resultado parcial correspondiente al participante 3
```

```
 2498.333333

RESULTADO 3.3333000000002357


   *************************
      MENÚ DE OPERACIONES
   *************************

         1 - Media
         2 - Suma
         3 - Otra combinación lineal (requiere introducir los coeficientes)
         0 - Salir

Introduzca el número de la opción que desea realizar:

 2


SUMA

Su resultado parcial para la operación es: 3284

Introduzca el resultado parcial correspondiente al participante 1
 3399

Introduzca el resultado parcial correspondiente al participante 2

 1683


Introduzca el resultado parcial correspondiente al participante 3

 3284

RESULTADO 10.0


   *************************
      MENÚ DE OPERACIONES
   *************************
```

```
        1 - Media
        2 - Suma
        3 - Otra combinación lineal (requiere introducir los coeficientes)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 3

Introduzca 3 coeficientes separados por comas

 2,3,1


COMBINACIÓN LINEAL

Su resultado parcial para la operación es: 3252

Introduzca el resultado parcial correspondiente al participante 1

 2696


Introduzca el resultado parcial correspondiente al participante 2

 3774


Introduzca el resultado parcial correspondiente al participante 3

 3252


RESULTADO 18.0
```

B.5. Cálculos distribuidos: Multiplicación

- **Participante 1**

```
**************************
   MENÚ PRINCIPAL
**************************
```

```
        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 2


Introduzca el número de participantes:

 3


Introduzca el número de participantes necesarios para recuperar el resultado:

 2

gradoPol = 1

El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 1

PARTICIPANTE 1


Introduzca el número de cifras que tendrá en número primo p:

 4

p 3491

Introduzca el número secreto que desea multiplicar:

 3

Envíe el valor 382 al participante 1
```

```
Envíe el valor 761 al participante 2

Envíe el valor 1140 al participante 3

Introduzca el fragmento recibido del participante 1

 382

Introduzca el fragmento recibido del participante 2

 847

fragmentos [382, 847]

Envíe el valor 2886 al participante 1

Envíe el valor 3390 al participante 2

Envíe el valor 403 al participante 3

Introduzca el fragmento recibido del participante 1

 2886

Introduzca el fragmento recibido del participante 2

 658

Introduzca el fragmento recibido del participante 3

 1998

fragmentos_g [2886, 658, 1998]

Su resultado parcial es: H = 8682.0

Introduzca el resultado parcial correspondiente al participante 1

 8682

Introduzca el resultado parcial correspondiente al participante 2
```

```
 3388

Introduzca el resultado parcial correspondiente al participante 3

 -5397

RESULTADO: 12.0
```

- **Participante 2**

```
  ************************
     MENÚ PRINCIPAL
  ************************

       1 - Media, suma y combinación lineal de valores secretos
       2 - Multiplicación de secretos
       3 - Mediana y cuartiles
       0 - Salir

Introduzca el número de la opción que desea realizar:

 2

Introduzca el número de participantes:

 3


Introduzca el número de participantes necesarios para recuperar el resultado:

 2

gradoPol = 1

El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 2


PARTICIPANTE 2
```

```
Introduzca el número primo que se utilizará en los cálculos:
 3491
Introduzca el número secreto que desea multiplicar:
 4

Envíe el valor 847 al participante 1
Envíe el valor 1690 al participante 2
Envíe el valor 2533 al participante 3
Introduzca el fragmento recibido del participante 1
 761

Introduzca el fragmento recibido del participante 2
 1690

fragmentos [761, 1690]
Envíe el valor 658 al participante 1
Envíe el valor 3405 al participante 2
Envíe el valor 2661 al participante 3
Introduzca el fragmento recibido del participante 1
 3390
Introduzca el fragmento recibido del participante 2
 3405
```

```
Introduzca el fragmento recibido del participante 3

 3433


fragmentos_g [3390, 3405, 3433]

Su resultado parcial es: H = 3388.0

Introduzca el resultado parcial correspondiente al participante 1

 8682


Introduzca el resultado parcial correspondiente al participante 2

 3388


Introduzca el resultado parcial correspondiente al participante 3

 -5397

RESULTADO: 12.0
```

- **Participante 3**

```
   ************************
      MENÚ PRINCIPAL
   ************************

        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 2


Introduzca el número de participantes:
```

```
 3


Introduzca el número de participantes necesarios para recuperar el resultado:

 2

gradoPol = 1

El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 3


PARTICIPANTE 3

Introduzca el número primo que se utilizará en los cálculos:

 3491


Introduzca el fragmento recibido del participante 1

 1140

Introduzca el fragmento recibido del participante 2

 2533


fragmentos [1140, 2533]

Envíe el valor 1998 al participante 1

Envíe el valor 3433 al participante 2

Envíe el valor 1377 al participante 3

Introduzca el fragmento recibido del participante 1
```

```
 403

Introduzca el fragmento recibido del participante 2

 2661

Introduzca el fragmento recibido del participante 3

 1377

fragmentos_g [403, 2661, 1377]

Su resultado parcial es: H = -5397.0

Introduzca el resultado parcial correspondiente al participante 1

 8682

Introduzca el resultado parcial correspondiente al participante 2

 3388

Introduzca el resultado parcial correspondiente al participante 3

 -5397

RESULTADO: 12.0
```

B.6. Cálculos distribuidos: Mediana y cuartiles

Para esta operación se realiza un ejemplo únicamente con dos participantes y se omite una parte del paso de mensajes, ya que las operaciones que se realizan durante el protocolo con sumas de secretos y necesitan que los participantes intercambién fragmentos hasta llegar a la solución.

- **Participante 1**

```
   ************************
      MENÚ PRINCIPAL
   ************************

         1 - Media, suma y combinación lineal de valores secretos
         2 - Multiplicación de secretos
         3 - Mediana y cuartiles
         0 - Salir

Introduzca el número de la opción que desea realizar:

 3

Introduzca el número de participantes:

 2


El ID debe ser un número natural que será usado como segmento durante los
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 1

PARTICIPANTE 1


Introduzca el número de cifras que tendrá en número primo p:

 4

p 5923

Introduzca la ruta del fichero del que se leerán los datos:

 C:/Users/Alicia/Desktop/datos_mediana_distrib.csv


Las columnas existentes son: ['Part_A' 'Part_B']

Introduzca el nombre de la columna de la que se leerán los datos:
```

```
 Part_A


secretos
[1, 2, 4]

El número de elementos que posee es: 3

Secretos ordenados: [1, 2, 4]

A continuación, se calculará el número total de elementos de forma segura

Envíe el valor 3116 al participante 1

Envíe el valor 306 al participante 2

Introduzca el fragmento recibido del participante 1

 3116


Introduzca el fragmento recibido del participante 2


fragmentos [3116, 3046]

SUMA

Su resultado parcial para la operación es: 239

Introduzca el resultado parcial correspondiente al participante 1

 239

Introduzca el resultado parcial correspondiente al participante 2

 472


El número total de elementos es: 6.0
```

```
   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


Envíe el valor 588 al participante 1

Envíe el valor 1173 al participante 2

Introduzca el fragmento recibido del participante 1

 588


Introduzca el fragmento recibido del participante 2

 5040


fragmentos [588, 5040]

SUMA

Su resultado parcial para la operación es: 5628

Introduzca el resultado parcial correspondiente al participante 1

 5628


Introduzca el resultado parcial correspondiente al participante 2

 5327
```

```
Envíe el valor 2680 al participante 1

Envíe el valor 5360 al participante 2

Introduzca el fragmento recibido del participante 1

 2680

Introduzca el fragmento recibido del participante 2

 2232


fragmentos [2680, 2232]

SUMA

Su resultado parcial para la operación es: 4912

Introduzca el resultado parcial correspondiente al participante 1

 4912

Introduzca el resultado parcial correspondiente al participante 2

 3901

Envíe el valor 3260 al participante 1

Envíe el valor 594 al participante 2

[...]

    Envíe el valor 2569 al participante 1

Envíe el valor 5138 al participante 2

Introduzca el fragmento recibido del participante 1

 2569
```

```
Introduzca el fragmento recibido del participante 2

 2603

fragmentos [2569, 2603]

SUMA

Su resultado parcial para la operación es: 5172

Introduzca el resultado parcial correspondiente al participante 1

 5172

Introduzca el resultado parcial correspondiente al participante 2

 4419

MEDIANA (SEGUNDO CUARTIL)

RESULTADO: 11.5
```

- **Participante 2**

```
   ************************
      MENÚ PRINCIPAL
   ************************

        1 - Media, suma y combinación lineal de valores secretos
        2 - Multiplicación de secretos
        3 - Mediana y cuartiles
        0 - Salir

Introduzca el número de la opción que desea realizar:

 3

Introduzca el número de participantes:

 2


El ID debe ser un número natural que será usado como segmento durante los
```

```
cálculos.
Además, este ID será conocido por el resto de participantes.
Introduzca su ID:

 2


PARTICIPANTE 2

Introduzca el número primo que se utilizará en los cálculos:

 5923


Introduzca la ruta del fichero del que se leerán los datos:

 C:/Users/Alicia/Desktop/datos_mediana_distrib.csv


Las columnas existentes son: ['Part_A' 'Part_B']

Introduzca el nombre de la columna de la que se leerán los datos:

 Part_B


secretos
[20, 21, 22]

El número de elementos que posee es: 3

Secretos ordenados: [20, 21, 22]

A continuación, se calculará el número total de elementos de forma segura

Envíe el valor 3046 al participante 1

Envíe el valor 166 al participante 2

Introduzca el fragmento recibido del participante 1

 306
```

```
Introduzca el fragmento recibido del participante 2

 166


fragmentos [306, 166]

SUMA

Su resultado parcial para la operación es: 472

Introduzca el resultado parcial correspondiente al participante 1

 239


Introduzca el resultado parcial correspondiente al participante 2

 472


El número total de elementos es: 6.0


   ************************
      MENÚ DE OPERACIONES
   ************************

        1 - Mediana (segundo cuartil)
        2 - Primer cuartil
        3 - Tercer cuartil
        4 - Otra posición (introducir por teclado)
        0 - Salir

Introduzca el número de la opción que desea realizar:

 1


Envíe el valor 5040 al participante 1

Envíe el valor 4154 al participante 2
```

```
Introduzca el fragmento recibido del participante 1
 1173

Introduzca el fragmento recibido del participante 2
 4154

fragmentos [1173, 4154]
SUMA
Su resultado parcial para la operación es: 5327
Introduzca el resultado parcial correspondiente al participante 1
 5628

Introduzca el resultado parcial correspondiente al participante 2
 5327
Envíe el valor 2232 al participante 1
Envíe el valor 4464 al participante 2
[...]

    Envíe el valor 5204 al participante 2
Introduzca el fragmento recibido del participante 1
 5138

Introduzca el fragmento recibido del participante 2
 5204
```

```
fragmentos [5138, 5204]

SUMA

Su resultado parcial para la operación es: 4419

Introduzca el resultado parcial correspondiente al participante 1

 5172


Introduzca el resultado parcial correspondiente al participante 2

 4419

MEDIANA (SEGUNDO CUARTIL)

RESULTADO: 11.5
```