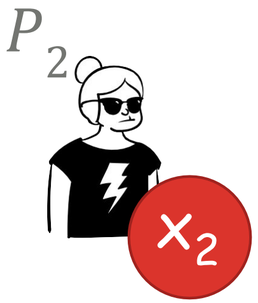
Technology Innovation Institute

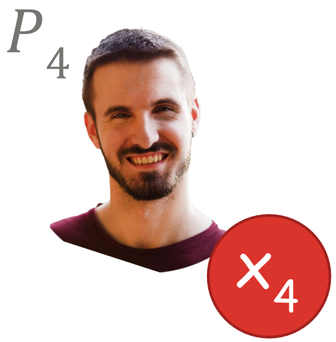
Multi-Party Computation (Cálculo Seguro Multipartito)

CatíoCrypto

Eduardo Soria Vázquez
eduardo.soria-vazquez@tii.ae

Introducción: Multi-Party Computation (MPC)





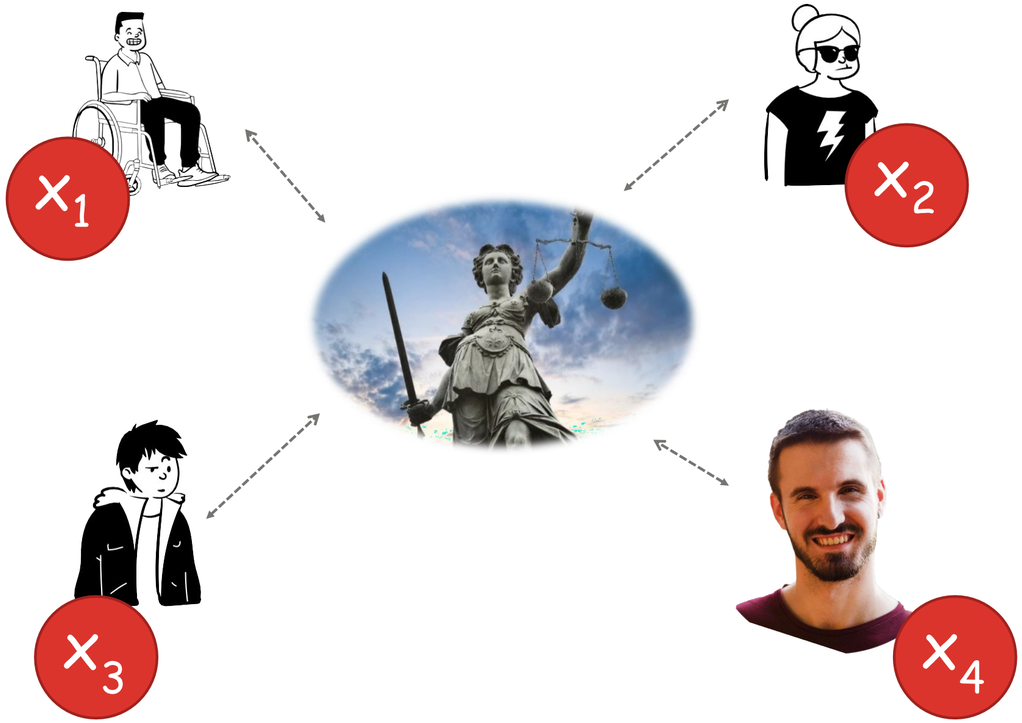
Un protocolo de MPC consta de:

- n entidades que “desconfían” las unas de las otras.
  - Cada entidad tiene un input privado.

- **Objetivo:** Calcular una función (pública) sobre el conjunto de sus datos privados. Con la garantía de:
  - No revelar nada más que el resultado de la función.
  - … y otras muchas.

$$y = f(x_1, x_2, x_3, x_4)$$

Multi-Party Computation vs Tercera Parte de Confianza



$$y = f(x_1, x_2, x_3, x_4)$$

Si hay una entidad en la cual TODOS confiamos, no hay necesidad de utilizar criptografía.

Confiamos en que esta entidad nos garantice que:

- No revela nuestros datos a nadie más.
- Realiza el cálculo de forma correcta.
- No es susceptible a ser coaccionada, ni hackeada.
- Nos devuelve el mismo resultado a todos.
- Nos devuelve siempre el resultado.

...

Problema con 3ª Parte Confianza:
Punto único de fallo.
Muchas veces, no existen.

Cálculo Seguro Multipartito (MPC)

Multi-Party Computation

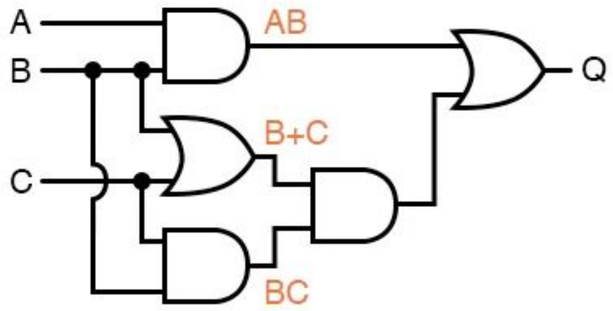
Altamente multidimensional (características de la red, del adversario, suposiciones computacionales…)

Métricas de eficiencia:

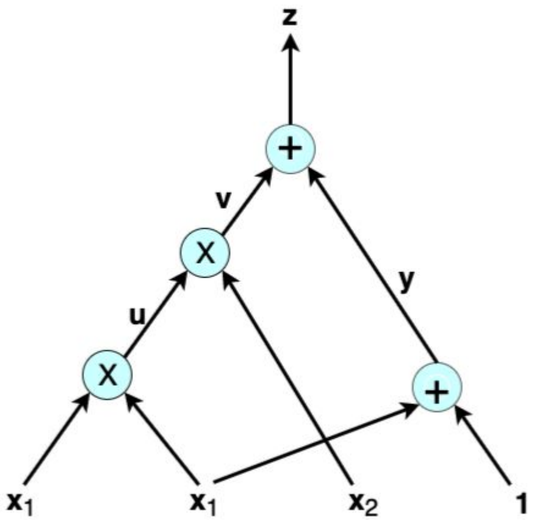
1. Cantidad de datos que se comunican (complejidad de comunciación).
2. Número de rondas de comunicación (complejidad de rondas).
3. Complejidad computacional.

La combinación de las distintas dimensiones determina la eficiencia que es posible (en ocasiones, tenemos cotas inferiores) o directamente si es posible o imposible hacer MPC con dicha selección.

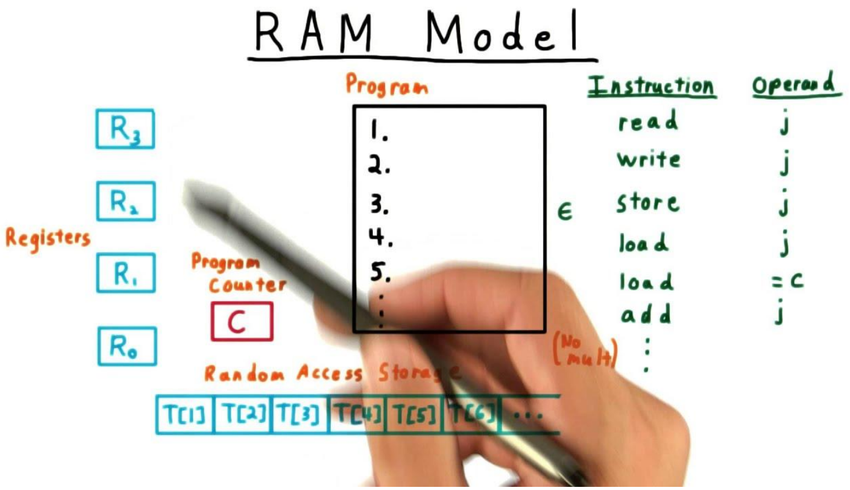
Dimensión: Modelo de computación



Circuitos Booleanos



Circuitos Aritméticos (sobre cuerpos o anillos finitos)



Memoria de Acceso Aleatorio (RAM)

Cualquier función o programa informático se puede expresar en cualquiera de los tres modelos

Dimensión: Fases en el protocolo

Paradigma “Offline-Online”

Existe una fase de preprocesado (“offline”) en la cual no es necesario que las entidades hayan decidido sus inputs para la función.

La fase “online” empieza cuando las entidades conocen sus inputs, y generalmente esta puede ejecutarse más rápido debido al trabajo Avanzado en el preprocesado.

Preprocesado dependiente de la función

La fase de preprocesado asume el conocimiento de la función que se calculará en la fase online.

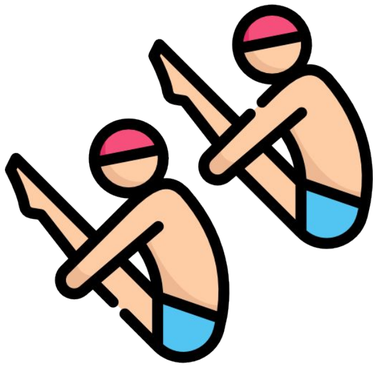
Preprocesado independiente de la función

La fase de preprocesado no asume el conocimiento de la función que se calculará en la fase online.

Sin preprocesado

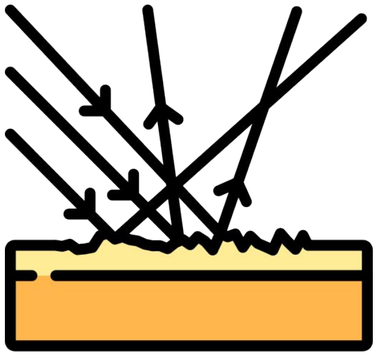
Existe una única fase, no se avanza trabajo antes de conocer los inputs.

Dimensión: Suposiciones sobre la red



Síncrona:

Las entidades están de acuerdo en una misma hora actual. Se establecen momentos límite para la recepción de mensajes. Existe la noción de “rondas” de comunicación.



Asíncrona:

Cada entidad tiene un reloj con una hora independientemente establecida. En la práctica, esto quiere decir que no se puede esperar que los mensajes se reciban en coordinación con los relojes internos de las otras entidades o p.ej. determinar exactamente cuánto tiempo esperar un mensaje.

Dimensión: Proporción de entidades corruptas

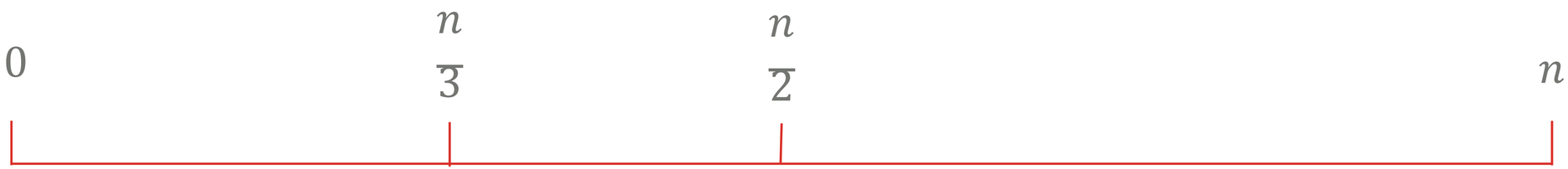
El adversario puede corromper (controlar) a lo sumo *t* entidades de las *n.* Tres escenarios principales

Mayoría honesta 2/3:

Al menos dos tercios de las entidades son honestas, es decir $t < n/3$

Mayoría deshonesta:

Al menos una entidad es honesta, es decir $t < n$.



Mayoría honesta:

Al menos una mayoría (estricta) de las entidades son honestas, i.e. $t < n/2$.

Dimensión: Capacidades del adversario

Seguridad pasiva:

Las entidades corruptas (controladas por el Adversario) siguen las instrucciones del protocolo. Sin embargo, poniendo en conjunto sus datos y los mensajes que reciben, tratan de deducir información más allá del output de la función.

Seguridad activa:

Las entidades corruptas (controladas por el Adversario) actúan de manera arbitraria. En particular, pueden mandar mensajes más tarde (o no enviarlos), modificar su contenido o ignorar por completo las instrucciones del protocolo.

Dimensión: Movilidad del adversario

Seguridad estática:

El Adversario especifica qué entidades van a ser corrompidas antes de comenzar la ejecución del protocolo de MPC.

Seguridad Adaptativa:

El Adversario puede decidir qué entidades corromper (controlar) de forma adaptativa durante la ejecución del protocolo, es decir en función de los mensajes que se intercambien y lo que vaya sucediendo.

I Dimensión: Capacidad computacional del Adversario

Seguridad Incondicional:

No se impone ninguna restricción en cuanto a la cantida de recursos computacionales del adversario. Por lo tanto, la seguridad se basa en estadística y herramientas de la teoría de la información. Disitinguimos dos escenarios:

- Si existe una pequeña probabilidad (conocida) de que el sistema sea inseguro, hablamos de seguridad ESTADÍSTICA (nos puede valer un protocolo que solo fracasa con probabilidad 2^{-80}, p.ej.).
- Si no hay probabilidad de error (como p.ej. con el One-Time Pad en sistemas de cifrado) la seguridad es PERFECTA.

No requiere problemas computacionalmente difíciles.

Resistente a ordenadores cuánticos

La seguridad incondicional es imposible cuando $t > n/2$

Seguridad computacional:

El Adversario es modelado como una máquina de Turing probabilística en tiempo polynomial (PPT) en vez de tener una capacidad ilimitada. La seguridad reposa sobre la hipótesis de que algún problema computacional específico es difícil de resolver en tiempo polinomial, p.ej. la descomposición en factores primos o el problema del logaritmo discreto.

Dimensión: Garantías sobre el output

Entrega garantizada

Las entidades honestas siempre reciben el output.

Justo

Si las entidades corruptas reciben el output, entonces también lo reciben las entidades honestas.

Seguridad con aborto

Las entidades corruptas pueden recibir el output mientras impiden que las entidades honestas lo hagan.

Existen mitigaciones para este modelo, como que p.ej. en tal caso, las entidades honestas puedan ponerse de acuerdo sobre la identidad de al menos a una de las entidades corruptas.

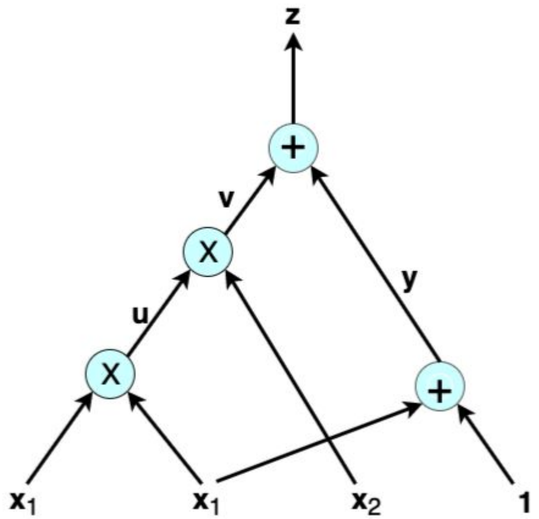
MPC: Posibilidades e imposibilidades

Tipo de Seguridad	Umbral t	Seguridad			Garantías sobre Output		
		Perfecta	Estadística	Compu-tacional	Entrega Garantizada	Justo	con Aborto
Pasiva	$< n$	✗	✗	✓	✓	✓	✓
	$< n/2$	✓	✓	✓	✓	✓	✓
	$< n/3$	✓	✓	✓	✓	✓	✓
Activa	$< n$	✗	✗	✓	✗	✗	✓
	$< n/2$	✗	✓	✓	✓	✓	✓
	$< n/3$	✓	✓	✓	✓	✓	✓

- Los protocolos con mayoría deshonesta requieren de hipótesis computationales. En presencia de adversarios activos, no pueden alcanzar más que seguridad con aborto.
- Los protocolos con mayoría honesta pueden basar su seguridad en teoría de la información y pueden lograr la entrega garantizada de output.
- La seguridad perfecta requiere que el adversario corrompa $< \boldsymbol{n/2}$ (resp. $< \boldsymbol{n/3}$) entidades si el adversario es pasivo (resp. activo).

MPC a partir de compartición de secretos en circuitos aritméticos

¿Cómo calculamos una función?



1. Cada entidad comparte su(s) input(s) secretos correspondientes.

2. Sucesivamente calculamos los valores intermedios y outputs del circuito, en forma de secreto compartido:

    1. Las operaciones lineales no requieren de interacción si el esquema de compartición de secretos es lineal.

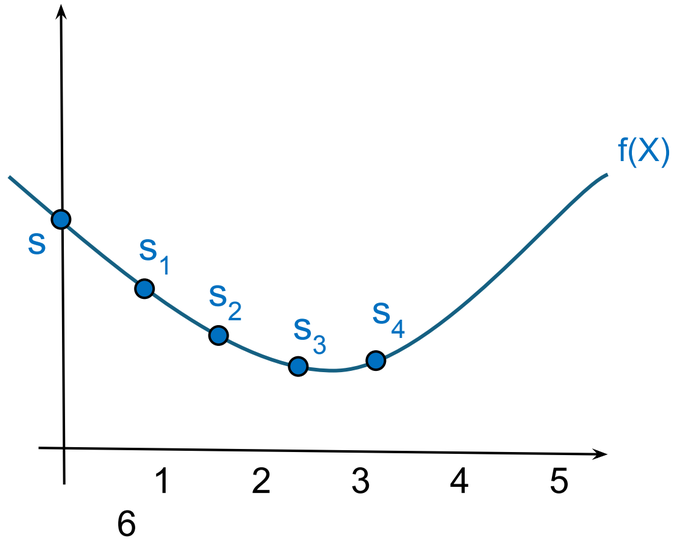
    2. Productos u operaciones no lineales requieren interacción (casi siempre).

3. Reconstruimos los outputs a las entidades que les correspondan dichos resultados.

Compartición de secretos de Shamir: Multiplicación

Shamir

1. **Fase de compartición:**
   I. $Compartidor$ escoge de forma unif. aleatoria $f \in \mathbb{F}_p[X]$ de grado t tal que $f(0) = s$.
   II. $Compartidor$ a cada P_i su parte $f(i) = s_i$.
2. **Fase de reconstrucción:**
   a. Cada P_i envía s_i a la entidad que quieren que reconstruya el secreto.
   b. Dicha entidad obtiene $f \in \mathbb{F}_p[X]$ mediante interpolación y recupera el secreto evaluando $f(0) = s$.



Sean $[x]_t$ e $[y]_t$ dos secretos compartidos mediante Shamir para umbral t.

Si cada P_i calcula localmente $x_i \cdot y_i$, las entidades obtienen **de forma segura** el secreto compartido $[x \cdot y]_{2t}$.

Pista: Isomorfismo de espacios vectoriales de ayer

MPC en el Estilo de BGW, seguridad pasiva

¿En qué dimensiones nos movemos?

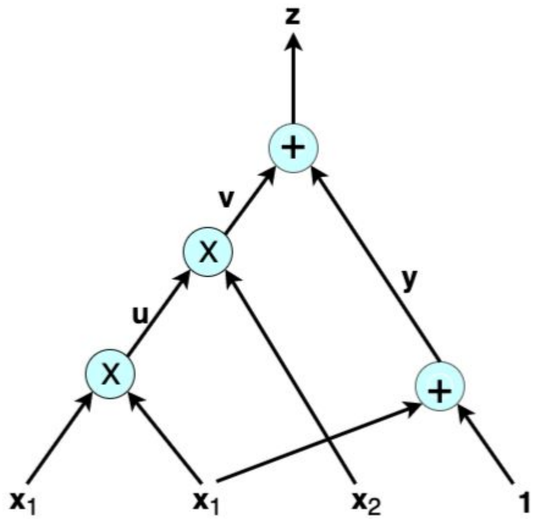
1. Modelo de computación: Circuitos aritméticos
2. Red: Síncrona
3. Sin preprocesado
4. Adversario:
    1. Pasivo
    2. Estático
    3. Seguridad Perfecta
    4. Umbral de corrupción: t $< n/2$

Herramientas: Esencialmente el protocolo [BGW88]

1. Idea general para protocolos basados en compartición de secretos de hace dos diapositivas.
2. Compartición de secretos de Shamir

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. “Completenes
theorems for noncryptographic fault-tolerant distributed computation”.

MPC en el Estilo de BGW, seguridad pasiva



1. Cada entidad comparte su(s) input(s) secretos correspondientes.

2. Sucesivamente calculamos los valores intermedios y outputs del circuito, en forma de secreto compartido:

    1. Las operaciones lineales no requieren de interacción si el esquema de compartición de secretos es lineal.

    2. Productos u operaciones no lineales requieren interacción (casi siempre).

3. Reconstruimos los outputs a las entidades que les correspondan dichos resultados.

MPC en el Estilo de BGW, seguridad pasiva

Protocolo de multiplicación

Sean $[x]_t$ e $[y]_t$ dos valores a multiplicar.

1. Las entidades calculan de forma local $[z]_{2t} = [x \cdot y]_{2t}$. Denotemos a las partes del secreto como $(z_1, \dots, z_n)$.
2. Cada entidad P_i usa el esquema de Shamir para compartir su parte $[z_i]_t$ con el resto de entidades.
3. De forma local, las entidades calculan $[x \cdot y]_t = \sum_{i=1}^{2t+1} \lambda_i \cdot [z_i]_t$.

Los valores λ_i son los coeficientes de interpolación de Lagrange.
¡Puedes deducirlos a partir del isomorfismo entre polinomios y su evaluación!

1. ¿Qué pasaría si no impusieramos $t < n/2$?
2. Este protocolo, ¿es seguro contra un adversario activo?

MPC en el Estilo de BGW, seguridad activa

¿En qué dimensiones nos movemos?

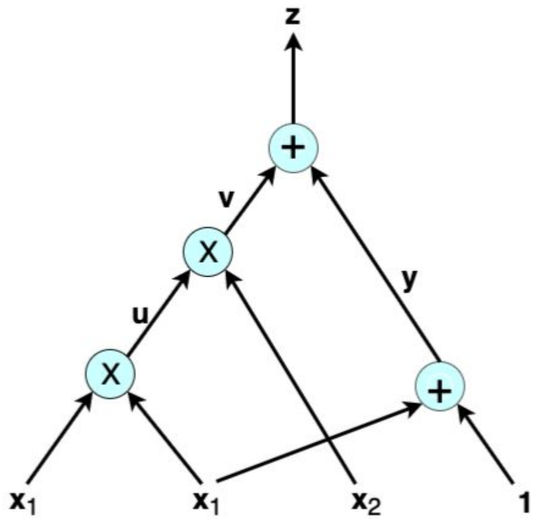
1. Modelo de computación: Circuitos aritméticos
2. Red: Síncrona
3. Preprocesado (“independiente de f”).
4. Adversario:
    1. Activo
    2. Estático
    3. Seguridad Perfecta
    4. Umbral de corrupción: $t < n/3$
    5. Seguridad con aborto (PODRÍA SER MEJOR)

Herramientas: Una simplificación del protocolo [BH08]

1. Idea general para protocolos basados en compartición de secretos.
2. Compartición de secretos de Shamir
3. Pares de reducción de grado

[BH08] Z. Beerliová-Trubíniová and M. Hirt. “Perfectly-secure MPC with linear communication complexity”. In *Theory of Cryptography Conference*, pages 213–230. Springer, 2008.

MPC en el Estilo de BGW, seguridad activa



1. Cada entidad comparte su(s) input(s) secretos correspondientes.

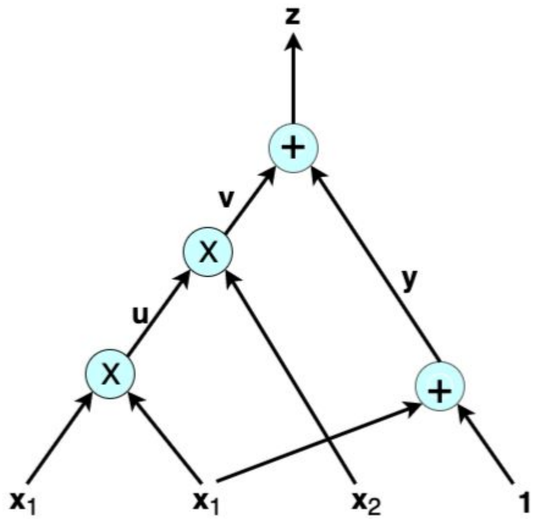
2. Sucesivamente calculamos los valores intermedios y outputs del circuito, en forma de secreto compartido:

    1. Las operaciones lineales no requieren de interacción si el esquema de compartición de secretos es lineal.

    2. Productos u operaciones no lineales requieren interacción (casi siempre).

3. Reconstruimos los outputs a las entidades que les correspondan dichos resultados.

MPC en el Estilo de BGW, seguridad activa



1. Cada entidad comparte su(s) input(s) secretos correspondientes.

2. Sucesivamente calculamos los valores intermedios y outputs del circuito, en forma de secreto compartido:

    1. Las operaciones lineales no requieren de interacción si el esquema de compartición de secretos es lineal.

    2. Productos u operaciones no lineales requieren interacción (casi siempre).

3. Reconstruimos los outputs a las entidades que les correspondan dichos resultados.

MPC en el Estilo de BGW, seguridad activa

Protocolo de repartición de inputs

Preprocesado: Un secreto **correctamente** compartido $[r]_t$, donde r es uniformemente aleatorio y únicamente conocido por P_i.

Inputs: x, el input secreto de P_i que nos queremos asegurar de que se comparte **correctamente**.

Outputs: $[x]_t$

1. P_i hace broadcast del valor e $= x - r$.
2. De forma local, las entidades calculan $[x]_t = e + [r]_t$.

Un canal de broadcast se asegura de que todas las entidades reciben el mismo mensaje.

MPC en el Estilo de BGW, seguridad activa

Protocolo de multiplicación

Preprocesado: Un par de reducción de grado $([r]_t, [r]_{2t})$, donde r es uniformemente aleatorio.
Inputs: $[x]_t$ e $[y]_t$ dos valores a multiplicar.
Outputs: $[x \cdot y]_t$

1. Las entidades calculan de forma local $[x \cdot y]_{2t}$.
2. También de forma local, calculan $[z]_{2t} = [x \cdot y]_{2t} - [r]_{2t}$.
3. Las entidades reconstruyen públicamente z, aplicando *detección* de errores*.
4. De forma local, las entidades calculan $[x \cdot y]_t = z + [r]_t$.

Podemos ver $[z]_d$ como una palabra en un Código de Reed-Solomon [n,d+1,n-d].
Por lo tanto, cuando t $< n/3$podemos detector errores en el caso d = 2t y corregir errores en el caso d = t.

MPC en el Estilo de BGW, seguridad activa

Preprocessing double-sharings with active security

Output: A set of double sharings $\{(\llbracket r_i \rrbracket_t, \llbracket r_i \rrbracket_{2t})\}_{i=2t+1}^{n}$

Protocol: The parties proceed as follows

1. Each party P_i samples $s_i \in_R \mathbb{F}$ and secret-shares it using degree-t and degree-$2t$ polynomials. The parties obtain $\llbracket s_i \rrbracket_t$ and $\llbracket s_i \rrbracket_{2t}$, but observe that corrupt parties may distribute shares *inconsistently*.
2. The parties compute locally the following shares:

$$\begin{pmatrix} \llbracket r_1 \rrbracket_t \\ \llbracket r_2 \rrbracket_t \\ \vdots \\ \llbracket r_n \rrbracket_t \end{pmatrix} = \boldsymbol{M} \cdot \begin{pmatrix} \llbracket s_1 \rrbracket_t \\ \llbracket s_2 \rrbracket_t \\ \vdots \\ \llbracket s_n \rrbracket_t \end{pmatrix}, \qquad \begin{pmatrix} \llbracket r_1 \rrbracket_{2t} \\ \llbracket r_2 \rrbracket_{2t} \\ \vdots \\ \llbracket r_n \rrbracket_{2t} \end{pmatrix} = \boldsymbol{M} \cdot \begin{pmatrix} \llbracket s_1 \rrbracket_{2t} \\ \llbracket s_2 \rrbracket_{2t} \\ \vdots \\ \llbracket s_n \rrbracket_{2t} \end{pmatrix}.$$

3. For each $i \in [2t]$, all the parties send their shares of $\llbracket r_i \rrbracket_t$ and $\llbracket r_i \rrbracket_{2t}$ to P_i.
4. Upon receiving these shares, each P_i for $i \in [2t]$ checks that the received sharings of $\llbracket r_i \rrbracket_t$ and $\llbracket r_i \rrbracket_{2t}$ are t and $2t$-consistent, respectively. If any of the sharings is not consistent, or if both are but the reconstructed value is not equal in both cases, P_i sends abort to all parties and halts.
5. If no party sends an abort message in the previous step, then the parties output the double-sharings $(\llbracket r_i \rrbracket_t, \llbracket r_i \rrbracket_{2t})$ for $i \in \{2t+1, \ldots, n\}$.

MPC en el Estilo de BGW, seguridad activa

Preprocessing double-sharings with active security

Output: A set of double sharings $\{(\llbracket r_i \rrbracket_t, \llbracket r_i \rrbracket_{2t})\}_{i=2t+1}^{n}$

Protocol: The parties proceed as follows

1. Each party P_i samples $s_i \in_R \mathbb{F}$ and secret-shares it using degree-t and degree-$2t$ polynomials. The parties obtain $\llbracket s_i \rrbracket_t$ and $\llbracket s_i \rrbracket_{2t}$, but observe that corrupt parties may distribute shares *inconsistently*.
2. The parties compute locally the following shares:

$$\begin{pmatrix} \llbracket r_1 \rrbracket_t \\ \llbracket r_2 \rrbracket_t \\ \vdots \\ \llbracket r_n \rrbracket_t \end{pmatrix} = \boldsymbol{M} \cdot \begin{pmatrix} \llbracket s_1 \rrbracket_t \\ \llbracket s_2 \rrbracket_t \\ \vdots \\ \llbracket s_n \rrbracket_t \end{pmatrix}, \qquad \begin{pmatrix} \llbracket r_1 \rrbracket_{2t} \\ \llbracket r_2 \rrbracket_{2t} \\ \vdots \\ \llbracket r_n \rrbracket_{2t} \end{pmatrix} = \boldsymbol{M} \cdot \begin{pmatrix} \llbracket s_1 \rrbracket_{2t} \\ \llbracket s_2 \rrbracket_{2t} \\ \vdots \\ \llbracket s_n \rrbracket_{2t} \end{pmatrix}.$$

3. For each $i \in [2t]$, all the parties send their shares of $\llbracket r_i \rrbracket_t$ and $\llbracket r_i \rrbracket_{2t}$ to P_i.
4. Upon receiving these shares, each P_i for $i \in [2t]$ checks that the received sharings of $\llbracket r_i \rrbracket_t$ and $\llbracket r_i \rrbracket_{2t}$ are t and $2t$-consistent, respectively. If any of the sharings is not consistent, or if both are but the reconstructed value is not equal in both cases, P_i sends abort to all parties and halts.
5. If no party sends an abort message in the previous step, then the parties output the double-sharings $(\llbracket r_i \rrbracket_t, \llbracket r_i \rrbracket_{2t})$ for $i \in \{2t+1, \ldots, n\}$.

To analyze the protocol let us assume without loss of generality that the corrupt parties are $P_1, \ldots, P_t$. We claim that if there are no abort messages, then the following holds:

1. For each $i \in \{2t+1, \ldots, n\}$ the sharings $\llbracket r_i \rrbracket_t$ and $\llbracket r_i \rrbracket_{2t}$ held by the honest parties are t and $2t$-consistent, respectively, and their underlying secrets match.

2. For each $i \in \{2t+1, \ldots, n\}$, the secret r_i looks uniformly random and unknown to the adversary.

Pares de reducción de grado

Reescribamos $M = \begin{pmatrix} A & B & C \\ D & E & F \\ G & H & I \end{pmatrix}$

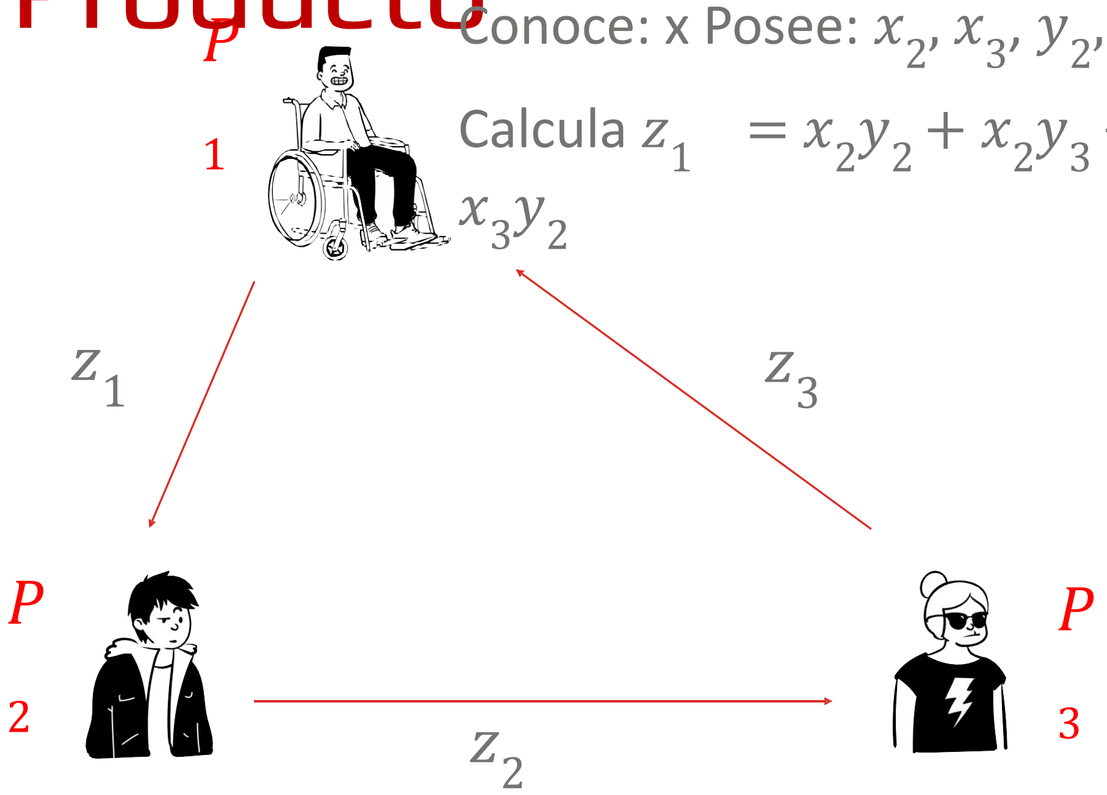
$$\begin{pmatrix} [\![r_{t+1}]\!]_d \\ [\![r_{t+2}]\!]_d \\ \vdots \\ [\![r_{2t}]\!]_d \end{pmatrix} = \boldsymbol{D} \cdot \begin{pmatrix} [\![s_1]\!]_d \\ [\![s_2]\!]_d \\ \vdots \\ [\![s_t]\!]_d \end{pmatrix} + \boldsymbol{E} \cdot \begin{pmatrix} [\![s_{t+1}]\!]_d \\ [\![s_{t+2}]\!]_d \\ \vdots \\ [\![s_{2t}]\!]_d \end{pmatrix} + \boldsymbol{F} \cdot \begin{pmatrix} [\![s_{2t+1}]\!]_d \\ [\![s_{2t+2}]\!]_d \\ \vdots \\ [\![s_n]\!]_d \end{pmatrix}.$$

Ya que M es una matriz hiperinvertible, podemos despejar:

$$\begin{pmatrix} [\![s_1]\!]_d \\ [\![s_2]\!]_d \\ \vdots \\ [\![s_t]\!]_d \end{pmatrix} = \boldsymbol{D}^{-1} \cdot \begin{pmatrix} [\![r_{t+1}]\!]_d \\ [\![r_{t+2}]\!]_d \\ \vdots \\ [\![r_{2t}]\!]_d \end{pmatrix} - \boldsymbol{D}^{-1}\boldsymbol{E} \cdot \begin{pmatrix} [\![s_{t+1}]\!]_d \\ [\![s_{t+2}]\!]_d \\ \vdots \\ [\![s_{2t}]\!]_d \end{pmatrix} - \boldsymbol{D}^{-1}\boldsymbol{F} \cdot \begin{pmatrix} [\![s_{2t+1}]\!]_d \\ [\![s_{2t+2}]\!]_d \\ \vdots \\ [\![s_n]\!]_d \end{pmatrix}.$$

Ejercicio RSS

Compartición de secretos replicada: Producto



P_1 tiene un secreto x y P_2 tiene un secreto y y P_3 tiene un secreto t. Usando compartición de secretos replicada, quieren realizar un protocolo de MPC para f(x,y,t) = x·y·t siguiendo los siguientes pasos:

1. P_1 (resp. P_2) comparte x (resp. y).
2. Multiplican como en la imagen de al lado.
3. [...]

Conoce: y Posee: x_1, x_3, y_1, y_3

Calcula $z_2 = x_3y_3 + x_1y_3 + x_3y_1$?

¡INSEGURO! No mantiene la privacidad siquiera contra un adversario pasivo, ¿por qué?

Conoce: x_2, x_1, y_2, y_1

Calcula $z_3 = x_1y_1 + x_1y_2 + x_2y_1$

Pista: P_2 es quien puede aprender de más.

Referencias y lecturas

Referencias y lecturas adicionales

[BGW88] Ben-Or, Michael, Shafi Goldwasser, and Avi Wigderson. "Completeness theorems for non-cryptographic fault-tolerant distributed computation." *Proceedings of the twentieth annual ACM symposium on Theory of computing*. 1988.