NEXO GLOBAL
Enlazando el futuro de los jóvenes Vallecaucanos

CRIPTOGRAFÍA HOMOMÓRFICA: ALGORITMO DE ENCRIPTACIÓN

THRESHOLD FULLY HOMOMORPHIC ENCRYPTION

Karol Stephany Insuasty Mejía

Estudiante del programa de Ingeniería Electrónica
Universidad del Valle, Cali-Colombia

Valle Invencible

GOBERNACIÓN VALLE DEL CAUCA

SGR Sistema General de Regalías

InfiValle

USC Universidad Santiago de Cali

El conocimiento es de todos

Minciencias

NEXO GLOBAL
Enlazando el futuro de los jóvenes Vallecaucanos

UNIVERSIDAD DE PURDUE

UNIVERSIDAD DEL VALLE

Ministerio de Ciencia Tecnología e Innovación
Gobernación del Valle del Cauca
Instituto Financiero para el Desarrollo del Valle del Cauca INFIVALLE
Universidad Santiago de Cali

Pasantía Internacional "Nexo Global Valle del Cauca"
Santiago de Cali, Colombia
18 de marzo de 2022

DISEÑO Y DIAGRAMACIÓN

Juan Diego Tovar Cardenas
librosusc@usc.edu.co

Valle Invencible | Gobernación Valle del Cauca | SGR Sistema General de Regalías | InfiValle | USC Universidad Santiago de Cali | El conocimiento es de todos | Minciencias

CRIPTOGRAFÍA HOMOMÓRFICA: ALGORITMO DE ENCRIPTACIÓN *THRESHOLD FULLY HOMOMORPHIC ENCRYPTION*

Homomorphic Cryptography: Threshold Fully Homomorphic Encryption Algorithm

Karol Stephany Insuasty Mejía
Estudiante del programa de Ingeniería Electrónica
Universidad del Valle, Cali-Colombia
karol.insuasty@correounivalle.edu.co

Resumen. En este paper se presentan los conceptos fundamentales teóricos y matemáticos de la criptografía homomórfica. Se realizan pruebas de escritorio para comprender el funcionamiento del algoritmo de encriptación llamado Threshold Fully Homomorphic Encryption. Finalmente, se presentan las metodologías de diseño hardware que se podrían utilizar para desarrollar un trabajo a futuro.

Palabras clave: Criptografía, Cifrado Homomórfico, Algoritmos.

Abstract. The theoretical and mathematical fundamental concepts of homomorphic cryptography is presented in this paper. Some proofs are implemented in order to know the algorithm's functionality named Threshold Fully Homomorphic Encryption. Finally, some methodologies are presented to design the hardware prototype.

Keywords: Cryptography, Homomorphic Encryption, Algorithms.

1. INTRODUCCIÓN

Hoy en día el uso de las nuevas tecnologías de la información y la comunicación son importantes en diversas esferas del diario vivir. Sin embargo, a partir de estos avances tecnológicos han surgido nuevos retos respecto a la seguridad de los mismos. Es por esto, que por la influencia del internet se genera el concepto de seguridad de la información que provee servicios necesarios como confidencialidad, disponibilidad e integridad a los usuarios [1].

De esta manera, la criptografía se define por [1] como una línea de las matemáticas que ofrece las herramientas necesarias en el mundo digital para prevenir problemas de seguridad en sistemas computarizados.

La criptografía moderna se clasifica en dos grandes grupos: criptografía de clave privada y criptografía de clave pública. En esta última se destacan ciertos algoritmos tales como DSA, curvas elípticas o RSA.

El algoritmo RSA es el primer esquema de encriptación de clave pública que posee la propiedad homomórfica, pero presenta una dificultad en el momento de rellenar un mensaje con bits aleatorios antes del cifrado para lograr la seguridad semántica [2]. Por esta razón, diferentes autores han propuesto nuevos esquemas de encriptación de clave pública con propiedades homomórficas.

A lo largo de la presente investigación nos enfocamos en comprender a nivel teórico y aplicativo el algoritmo de encriptación *Threshold Fully Homomorphic Encryption,* con el fin de lograr proponer a nivel general un diseño hardware .

2. CONCEPTOS FUNDAMENTALES

2.1 Definición de Criptografía

La Criptografía es una palabra de origen griego en donde <<cripto>> se traduce como secreto y <<grafía>> como escritura. Por lo tanto, se podría definir como la ciencia que estudia todo lo relativo a la *escritura secreta*. [4]

Así mismo el autor Sanjuan, L en [4], define la criptografía como la disciplina que se encarga del diseño de diferentes procedimientos de cifrado de datos con el fin de enmascarar una determinada información de carácter confidencial. De esta forma, el esquema fundamental de un proceso criptográfico (cifrado/descifrado) se puede representar de la siguiente manera:

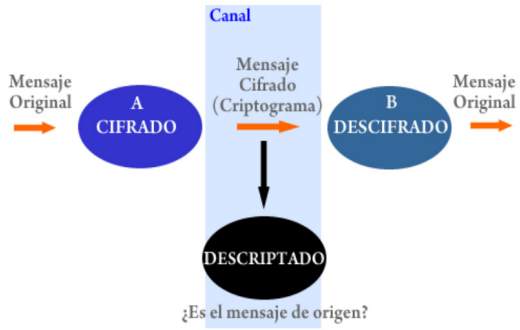

Figura 1. Representación del Proceso Criptográfico. Tomado de [4]

Como se observa en la Figura 1, A representa al emisor y B representa el receptor dentro del proceso del envío de un determinado mensaje. Al momento de enviar el mensaje, A realiza el proceso de *cifrado* o también llamado *encriptación de datos,* en el cual hace uso de un algoritmo para poder transformar la información inicial y que ésta sea ilegible para usuarios no autorizados. El mensaje una vez cifrado se denomina *criptograma* y es enviado mediante un canal público al receptor el cual realiza el proceso de *descifrado* o *desencriptación* del mensaje, es decir, transforma el criptograma en el mensaje original.

Cabe resaltar que la criptografía se relaciona directamente con la seguridad informática y busca cumplir con tres propósitos generales identificados dentro de la llamada *Tríada CIA (por sus siglas en inglés)* que trae a flote los conceptos de Confidencialidad, Integridad y Disponibilidad.

- Confidencialidad: Es equivalente a hablar de privacidad. Se procura salvaguardar la información para que no llegue a 'manos equivocadas' y que sólo aquellas personas que estén autorizadas puedan acceder de ella.
- Integridad: Implica mantener la consistencia, precisión y mantener la confiabilidad de los datos en todo momento. Es decir, se debe procurar que los datos del mensaje no sean modificados durante su envío y se hace necesario buscar alternativas para que personas externas no alteren los datos.
- Disponibilidad: Se refiere a que la información debe estar disponible siempre que se necesite acceder a ella. Y no se refiere sólo al software sino también al mantenimiento de hardware y las redes.

2.2 Tipos de criptografía

Dentro de la criptografía moderna, actualmente utilizada, se definen dos grandes grupos los cuales se describirán a continuación.

2.2.1 *Criptografía Simétrica o de Clave Secreta:* Se refiere a diferentes métodos que van a permitir tener una comunicación segura entre Emisor y Receptor haciendo uso de lo que se llamará Clave Simétrica o Única. Para que se ejecute correctamente el envío del mensaje dentro de este tipo de criptografía, en primera instancia el emisor (A) envía el mensaje al receptor (B), mensaje que se encripta con la clave privada de A; una vez el mensaje se ha enviado por el canal, el receptor (B) debe descifrar con su clave privada el criptograma y se obtiene el mensaje original [6]. Este proceso de encriptación se puede observar en la Figura 2.

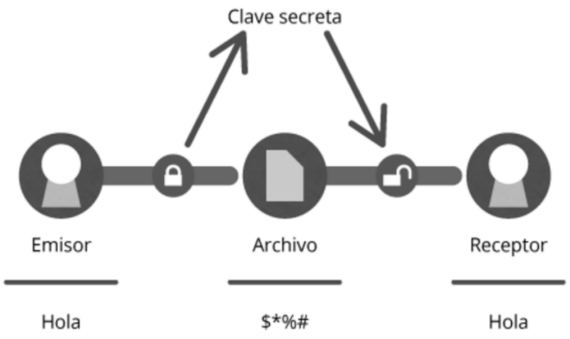

Figura 2. Proceso de Encriptación Clave Simétrica. Tomado de [5]

2.2.2 *Criptografía Asimétrica o de Clave Pública:* Es aquella que usa dos claves diferentes para cada usuario, una para cifrar denominada Clave Pública y la Clave Privada es la que se usa para descifrar el mensaje por el receptor. Estas dos claves tienen características matemáticas especiales ya que se generan siempre

por parejas y se relacionan entre sí. Durante el proceso de comunicación, se debe tener en cuenta que el emisor (A) dispone de su propia clave pública y privada, y debe conocer la clave pública del receptor. Una vez A envía el mensaje a B, cifrándolo con la clave pública de B, es B quien descifra el mensaje usando su propia clave privada y así poder obtener el mensaje original [2]. En la Figura 3 se representa gráficamente este proceso:

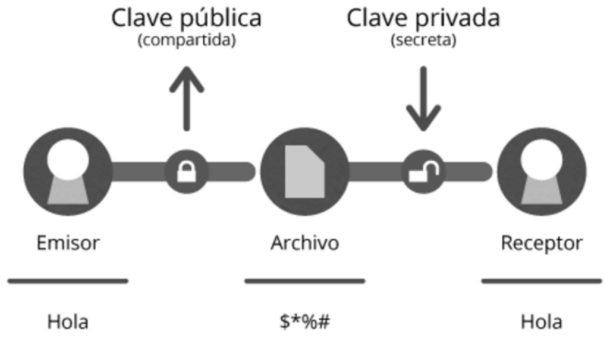

Figura 3. Proceso de Encriptación Clave Asimétrica. Tomado de [5]

3. DEFINICIÓN DE UN CRIPTOSISTEMA

En general, un criptosistema es un sistema criptográfico que provee un método para transformar un mensaje llamado *texto plano*, en otro mensaje llamado *texto cifrado*, usando solamente una clave secreta. Sí el criptosistema es seguro, el texto cifrado puede hacerse público de forma segura, y ninguna parte sin conocimiento de la clave secreta puede recuperar el texto plano.

Según Kevin,H [6] un criptosistema se define como:

1. P es un conjunto finito de posibles textos planos.
2. C es un conjunto finito de textos cifrados.
3. K es un conjunto finito de claves posibles.
4. Para cada K ∈ K, hay una regla de cifrado eK ∈ E y una regla de descifrado correspondiente dK ∈ D. Cada eK : P → C y dK: C → P son funciones tales que dK(eK(x)) = x para cada texto plano x ∈ P.

Si la regla de descifrado dK es la misma que la regla de cifrado de eK, o puede derivarse fácilmente de eK, entonces dicho criptosistema se denomina criptosistema de clave simétrica, ya que tanto el cifrado como el descifrado pueden realizarse utilizando únicamente el conocimiento de eK. En la configuración de **clave privada**, el valor de eK debe mantenerse en secreto, de lo contrario un adversario podría descifrar los mensajes. Si es computacionalmente imposible determinar dK a partir eK, entonces dicho criptosistema se denomina criptosistema de **clave pública**, ya que eK puede hacerse público de forma segura sin permitir que un adversario descifre los mensajes. Por computablemente inviable, se quiere decir que un adversario computacionalmente limitado tiene una posibilidad insignificante de tener éxito en el cálculo de dK dado eK, con respecto a algún parámetro de seguridad predefinido. En muchas configuraciones, se toma como la longitud de bits de los parámetros públicos del criptosistema.

Por otro lado, es importante resaltar que existen ciertas propiedades relevantes dentro de los sistemas criptográficos, que si bien, no todos los algoritmos cumplen con todas ellas, ayudan a clasificar y determinar la relevancia de dichos algoritmos. A continuación, se presenta una breve descripción de dichas propiedades.

3.1 Propiedades de sistemas criptográficos

Las propiedades más comunes en los sistemas criptográficos son la indistinguibilidad y maleabilidad:

3.1.1 Indistinguibilidad: Esta propiedad se refiere a la imposibilidad del adversario de un criptosistema de distinguir pares de textos cifrados en función del mensaje que cifran. Esta propiedad se considera como un requisito básico para la mayoría de los criptosistemas de clave pública para poder demostrar que son 'seguros' frente a diferentes ataques como el ataque de texto plano elegido, ataque de texto cifrado elegido y ataque de texto cifrado adaptable. [7]

De igual manera, según [7] se considera que un criptosistema es seguro *en términos de indistinguibilidad* si ningún adversario es capaz de identificar la elección de un mensaje al azar con una probabilidad significativamente mejor que la de adivinar al azar (½). Es decir, si un adversario puede distinguir el texto cifrado con una probabilidad mayor a ½, entonces se considera que este adversario tiene una "ventaja" y el esquema no se considera seguro en términos de indistinguibilidad.

3.1.2 Maleabilidad: Se considera que un algoritmo cifrado es "maleable" si es posible transformar un texto cifrado en otro texto cifrado que se descifre en un texto plano relacionado [10]. Aunque se considere que la maleabilidad de un algoritmo limita la seguridad teórica de un criptosistema, según [6] no es necesariamente una propiedad no deseada. De esta forma se introduce una clase especial de criptosistemas que están diseñados para permitir cálculos sencillos sobre textos cifrados. Estos algoritmos se encuentran dentro de la categoría de los criptosistemas homomórficos y permiten tomar dos mensajes cifrados eK(m1) y eK(m2) y calcular eK(m1 + m2) sin necesidad de conocer la clave privada.

4. CRIPTOGRAFÍA HOMOMÓRFICA

En general, la mayoría de los criptosistemas están definidos sobre dos grupos algebraicos o anillos, . El primer grupo que está definido sobre un grupo soporta una única operación, mientras que los criptosistemas definidos sobre un anillo soportan naturalmente dos operaciones: suma y multiplicación [6].

De esta forma según Henry, K. J. en [6] un criptosistema homomórfico, es un criptosistema de uso general que a su vez tiene las llamadas 'capacidades de homomorfismo de privacidad'. A su vez, un homomorfismo de privacidad se define como: un homomorfismo φ de un sistema algebraico U qu desde un conjunto S, algunas operaciones f1, f2, ..., algunos predicados p1, p2, ... , y algunas constantes distinguidas s1, s2, ... , se convierte en un sistema algebraico C consistente en un conjunto S', algunas operaciones f1', f2', ...,algunos predicados p1', p2', ..., y algunas constantes distinguidas. Y se cumple:

1. $(\forall i)(a, b, c, \ldots) \in S\ 0\ [f\ 0\ i\ (a, b, \ldots) = c \Rightarrow fi(\varphi(a), \varphi(b), \ldots) = \varphi(c)]$
2. $(\forall i)(a, b, c, \ldots) \in S\ 0\ p\ 0\ (a, b, \ldots) \equiv p(\varphi(a), \varphi(b), \ldots)$
3. $(\forall i)\ \varphi(s\ 0\ i\ ) = si\ .$

La definición de un criptosistema homomórfico es el conjunto de posibles textos planos P y el conjunto de textos cifrados posibles C; son ambos grupos tales que para cualquier K K y cualquiera de los dos textos cifrados c1= eK(m1), c2= eK(m2), y se mantiene la siguiente condición:

$$dK(c1 \cdot c2) = m1 \cdot m2$$

De igual forma, se respetan las condiciones algebraicas para las respectivas operaciones entre los textos cifrados:

1. $dK(c1 + c2) = m1 + m2$
2. $dK(c1 \cdot c2) = m1 \cdot m2$

Para resumir, el principal objetivo de la criptografía homomórfica se basa en poder realizar operaciones directamente sobre los datos cifrados, sin necesidad de descifrarlos antes, con la ventaja de poder disminuir la probabilidad de robo de los datos y poder tener un control más estricto sobre la disponibilidad de la información y así mismo lograr confidencialidad en el proceso.

A continuación se presentan en la Figura 4 y Figura 5 los esquemas de cifrado tanto no homomórfico como homomórfico, donde se puede observar claramente cómo se realizan operaciones de datos sin cifrar en este tipo de criptografía.

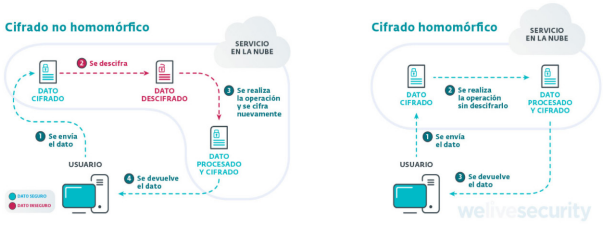

Figura 4. Esquema de Cifrado no Homomórfico Tomado de [9]

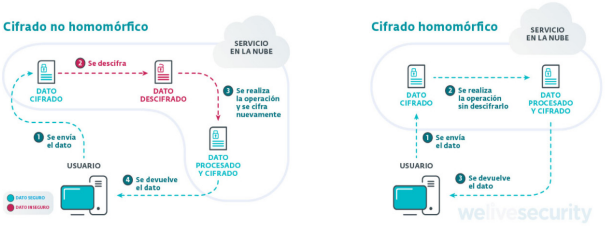

Figura 5. Esquema de Cifrado Homomórfico. Tomado de [9]

4. ALGORITMO SOFTWARE

El algoritmo que se evaluó fue el llamado *Threshold Fully Homomorphic Encryption* en su versión 1.1. Seguidamente se muestra el algoritmo utilizado.

```
from sage.all import load, sqrt, RR, ZZ, pi, oo

load('https://bitbucket.org/malb/lwe-estimator/raw/HEAD/estimator.py')

n = 1024              # ciphertext dimension (also, key entropy)

sd = 2**(-25)         # noise standard deviation

alpha = sqrt(2*pi)*sd    # estimator defines noise rate = sqrt(2pi).stdev

q = 2**32             # for compatibility only

m = oo                # the attacker can use as many samples he wishes

secret_distribution = (0,1)

success_probability = 0.99


# Chosen cost model

# BKZ cost models: CLASSICAL - 0.292*beta + 16.4 + log(8*d,2) - primal

# i.e. BKZ.sieve =  lambda beta, d, B: ZZ(2)**RR(0.292*beta + 16.4 + log(8*d,2))
```

```
print("CLASSICAL PRIMAL")

print(primal_usvp(n, alpha, q, secret_distribution=secret_distribution, m=m, success_probability=success_probability, reduction_cost_model=BKZ.sieve))

# BKZ cost models: CLASSICAL - 0.292*beta + 16.4 + log(8*d,2) - dual

# i.e. BKZ.sieve =  lambda beta, d, B: ZZ(2)**RR(0.292*beta + 16.4 + log(8*d,2))

print("CLASSICAL DUAL")

print(dual_scale(n, alpha, q, secret_distribution=secret_distribution, m=m, success_probability=success_probability, reduction_cost_model=BKZ.sieve))


# For more conservative parameters, both classical and quantum

# BKZ cost models: CLASSICAL - 0.292 beta - primal

reduction_cost_model =  lambda beta, d, B: ZZ(2)**RR(0.292*beta)

print("CLASSICAL PRIMAL (conservative)")

print(primal_usvp(n, alpha, q, secret_distribution=secret_distribution, m=m, success_probability=success_probability, reduction_cost_model=reduction_cost_model))

# BKZ cost models: CLASSICAL - 0.292 beta - dual

print("CLASSICAL DUAL (conservative)")

print(dual_scale(n, alpha, q, secret_distribution=secret_distribution, m=m, success_probability=success_probability, reduction_cost_model=reduction_cost_model))

# BKZ cost models: QUANTUM - 0.265 beta - primal

reduction_cost_model =  lambda beta, d, B: ZZ(2)**RR(0.265*beta)

print("QUANTUM PRIMAL (conservative)")

print(primal_usvp(n, alpha, q, secret_distribution=secret_distribution, m=m, success_probability=success_probability, reduction_cost_model=reduction_cost_model))

# BKZ cost models: QUANTUM - 0.265 beta - dual

print("QUANTUM DUAL (conservative)")

print(dual_scale(n, alpha, q, secret_distribution=secret_distribution, m=m, success_probability=success_probability, reduction_cost_model=reduction_cost_model))
```

El algoritmo anteriormente presentado describe el proceso de encriptación y desencriptación que se tomó de una librería de GitHub llamada Fast Fully Homomorphic Encryption (TFHE). El algoritmo también realiza la evaluación homomórfica de las 10 compuertas binarias (and, or, xor, nand, nor, etc..), así como la negación y la compuerta mux.

Para ello fue necesario crear primeramente una Máquina Virtual, debido a que el algoritmo anterior está diseñado para Linux. Con la máquina virtual y teniendo Linux instalado se descargó un ambiente de programación donde usamos un compilador de C++11 con el que se realizaron las diferentes pruebas de escritorio. A continuación, se presenta una imagen del ambiente en la máquina virtual´.

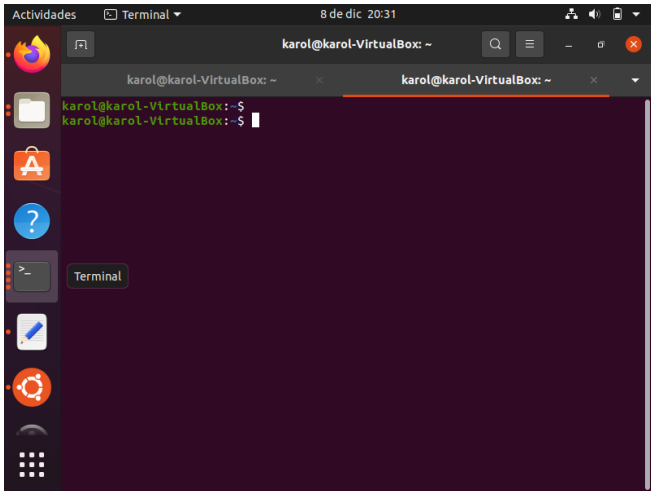

Figura 6. Máquina virtual implementada

El algoritmo usado genera un conjunto de claves secretas y un conjunto de claves en la nube. El conjunto de claves secretas es privado y proporciona capacidades de cifrado y descifrado. Por otra parte, el conjunto de claves en la nube se puede exportar a la nube, y permite operar sobre datos cifrados.

Al correr el algoritmo se obtiene como respuesta de la evaluación de las compuertas, en donde cada compuerta binaria tarda aproximadamente unos 13 milisegundos en evaluarse. De igual forma, el parámetro predeterminado alcanza una seguridad criptográfica de 110 bits, basada en 'ideal lattices'.

5. METODOLOGÍAS DE DISEÑO HARDWARE

Dentro de las metodologías más comunes para desarrollar un diseño hardware se pueden mencionar dos grandes grupos, el primero agrupa los dos tipos de diseño hardware más comunes en

cuanto a su estructura y el segundo grupo menciona la agrupación dependiendo de la descripción del algoritmo. A continuación se describe brevemente cada una de ellas.

4.1 Diseño hardware dependiendo de su estructura

4.1.1 Bottom-Up: La metodología Bottom-Up no implica una estructuración jerárquica de los elementos del sistema. Simplemente reúne componentes de bajo nivel para formar el diseño global.

Según Pardo, F. y Boluda J. en [13] se debe iniciar un diseño Bottom-Up realizando una descripción con esquemas de los componentes del circuito. Estos componentes se construyen normalmente a partir de otros que pertenecen a una biblioteca que contiene componentes básicos y representan unidades funcionales con significado propio dentro del diseño. Estas unidades son denominadas primitivas, ya que no es necesario disponer de elementos de más bajo nivel para el diseño que se desea realizar. La Figura 7 representa el diagrama de diseño para la metodología Bottom-up.

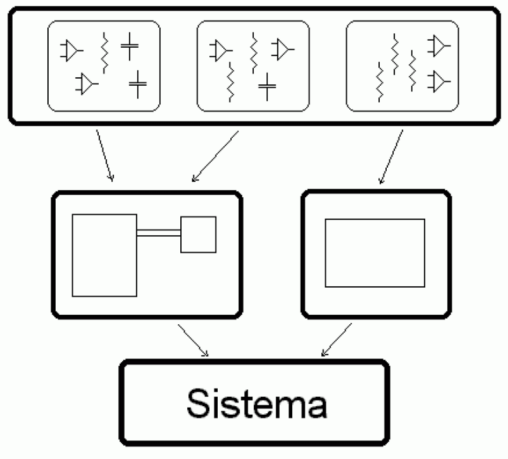

Figura 7. Diagrama de diseño para la metodología hardware Bottom-Up. Tomado de [12]

4.1.2 Top-Down: El diseño Top-Down consiste en capturar una idea con un alto nivel de abstracción, implementarla partiendo de la misma, e incrementar el nivel de detalle según sea necesario. El sistema inicial se va subdividiendo en módulos, estableciendo una jerarquía. Cada módulo se subdivide cuantas veces sea necesario hasta llegar a los componentes primarios del diseño como muestra el esquema de la Figura 8:

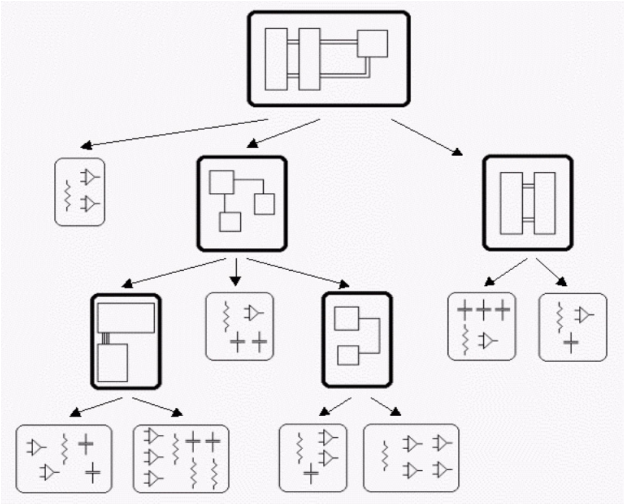

Figura 8. Diagrama de diseño para la metodología hardware Top-Down. Tomado de [12]

4.2 Diseño hardware dependiendo de su descripción

4.2.1 Comportamental : Según [12] el diseño comportamental como su nombre lo indica, se encarga de describir el comportamiento del circuito, sin poner énfasis en su arquitectura. Dicha descripción se realiza mediante un lenguaje de hardware específico. No se especifican señales ni elementos de bajo nivel.

4.2.2 Estructural : El diseño estructural es aquel que permite enumerar los componentes de un circuito junto con sus interconexiones. Este tipo de diseño se realiza usando esquemas que describen gráficamente los componentes del circuito, o mediante el uso de un lenguaje, en donde simplemente se enumeran todos los elementos [12].

6. REFLEXIÓN

La realización de la presente pasantía de investigación aportó en distintos ámbitos académicos de la investigadora. Puesto que, permitió una comprensión global del tema relacionado con criptografía homomórfica. Este tipo de criptografía permite realizar operaciones directamente sobre los datos cifrados, sin necesidad de descifrarlos antes, con la ventaja de poder disminuir la probabilidad de robo de los datos y poder tener un control más estricto sobre la disponibilidad de la información y así mismo lograr confidencialidad en el proceso. Y existen dos propiedades básicas que rigen su funcionamiento los cuales son indistinguibilidad y maleabilidad.

7. CONCLUSIÓN

A partir de las pruebas realizadas del algoritmo *Threshold Fully Homomorphic Encryption* se logró una correcta comprensión del proceso de encriptación y desencriptación de datos usando las propiedades de la Criptografía Homomórfica. Se pudo de igual manera, presentar dos posibles soluciones para como trabajo futuro poder realizar el esquemático hardware del algoritmo que permite los dos procesos.

7. REFERENCIAS

[1] Marrero Travieso, Y. (2003). La Criptografía como elemento de la seguridad informática. Acimed, 11(6), 0-0.

[2] Morán Torres, O. C., Poveda Moreno, R. A., & Quintero Cadena, L. B. "Criptografía moderna:(criptosistemas de clave pública y privada)" (Doctoral dissertation),2003

[3] Díaz, A. M. P. (1995). ¿ Qué es la criptografía?. *Buran*, 6-8.

[4] Sanjuan, L. (2012). Criptografía I.

[5] Página Web: https://www.researchgate.net/figure/Figura-41-Criptografia-de-clave-privada-o-simetrica-imagen-superior-y-criptografia-de_fig12_314733146

[6] Henry, K. J. (2008). *The theory and applications of homomorphic cryptography* (Master's thesis, University of Waterloo).

[7] Página Web: https://hmong.es/wiki/Ciphertext_indistinguishability

[8] Página Web: https://hmong.es/wiki/Malleability_(cryptography)

[9] Página Web: https://www.welivesecurity.com/la-es/2019/09/04/criptografia-homomorfica-paradigma-cifrado-cercano/

[10] Boneh, D., Gennaro, R., Goldfeder, S., Jain, A., Kim, S., Rasmussen, P. M., & Sahai, A. (2018, August). Threshold cryptosystems from threshold fully homomorphic encryption. In *Annual International Cryptology Conference* (pp. 565-596). Springer, Cham.

[11] Chillotti, I., Gama, N., Georgieva, M., & Izabachène, M. (2020). TFHE: fast fully homomorphic encryption over the torus. *Journal of Cryptology*, *33*(1), 34-91.

[12] Schweers, R. J. (2002). *Descripción en VHDL de arquitecturas para implementar el algoritmo CORDIC* (Doctoral dissertation, Universidad Nacional de la Plata).

[13] Pardo, F. y Boluda J., "VHDL Lenguaje para síntesis y modelado de circuitos," Alfaomega Grupo Editor, 2000.

Cita recomendada

Insuasty Mejía KS. Criptografía Homomórfica: Algoritmo de encriptación Threshold Fully Homomorphic Encryption. Nexo Global. Artículos de reflexión. 2022; p. 1-9.