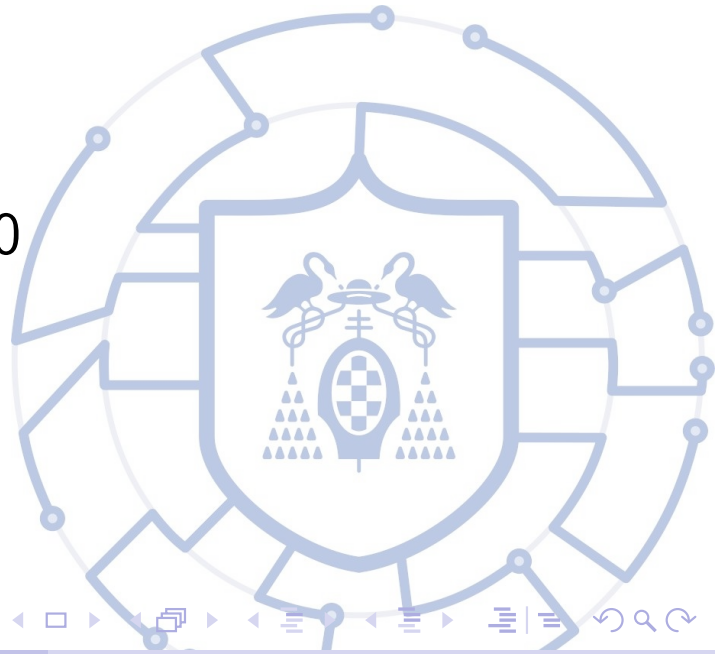
Criptografía homomórfica

Kevin van Liebergen Jorge Lafuente

Universidad de Alcalá

23 de diciembre de 2020



Introducción

- Un criptosistema es **homomórfico** si permite realizar una operación algebraica sobre un texto cifrado y se manifiesten en el texto plano tras descifrar
- Dada una función de cifrado $Hom(m, K)$ y otra de descifrado $Hom^{-1}(m, K)$, m texto plano, c texto cifrado y K elemento secreto.
- Dados dos mensajes en claro M_1 y M_2

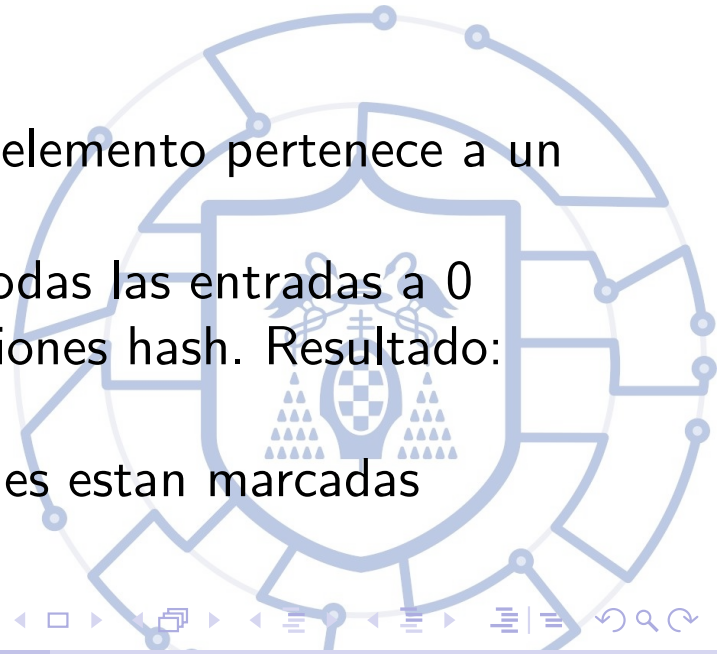
Cifrado Homomórfico

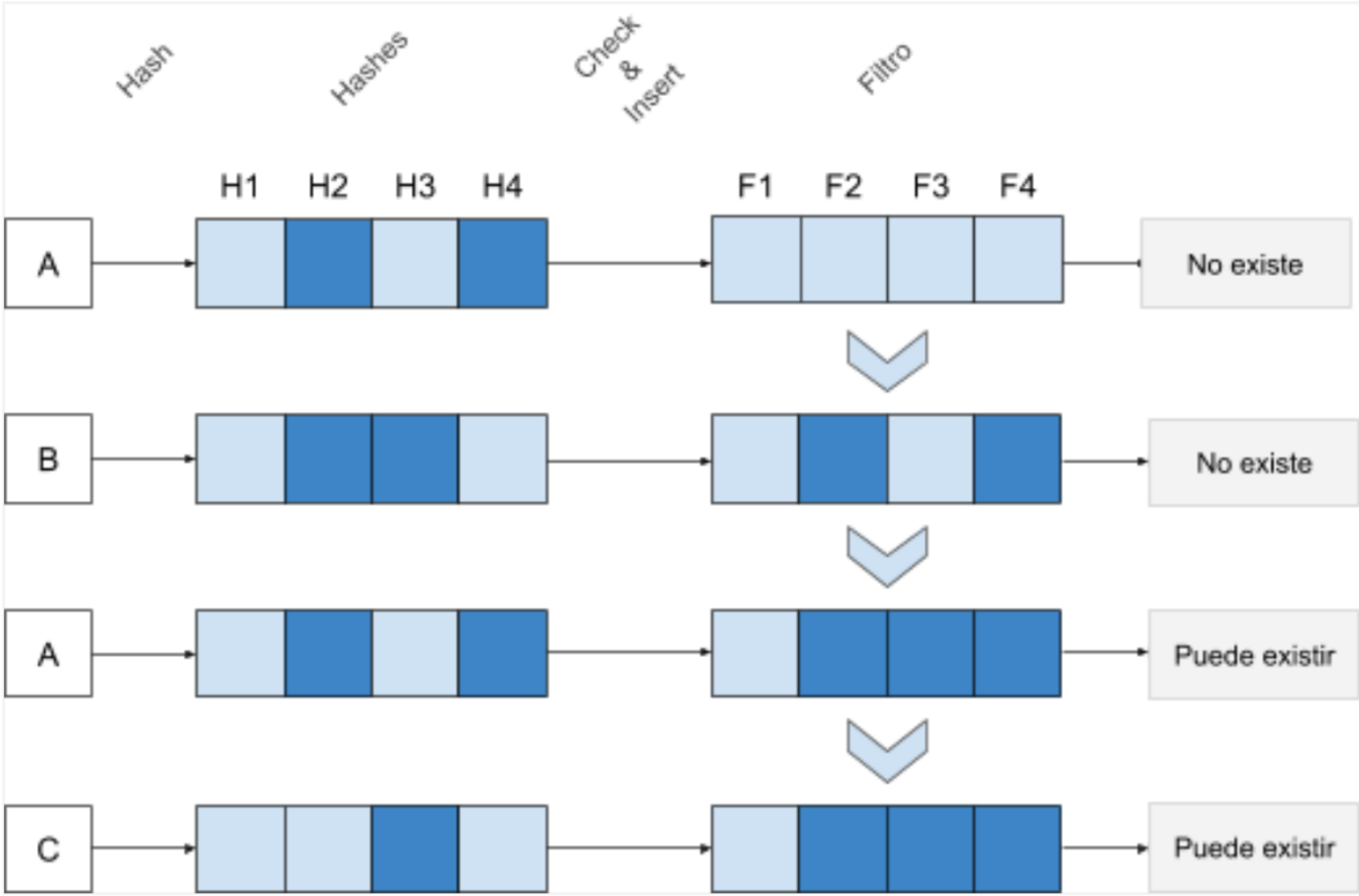
El sistema (Hom, Hom^{-1}), será un sistema homomórfico para la $+$ si

$$Hom(M_1) + Hom(M_2) = Hom(M_1 + M_2)$$

Enclaves

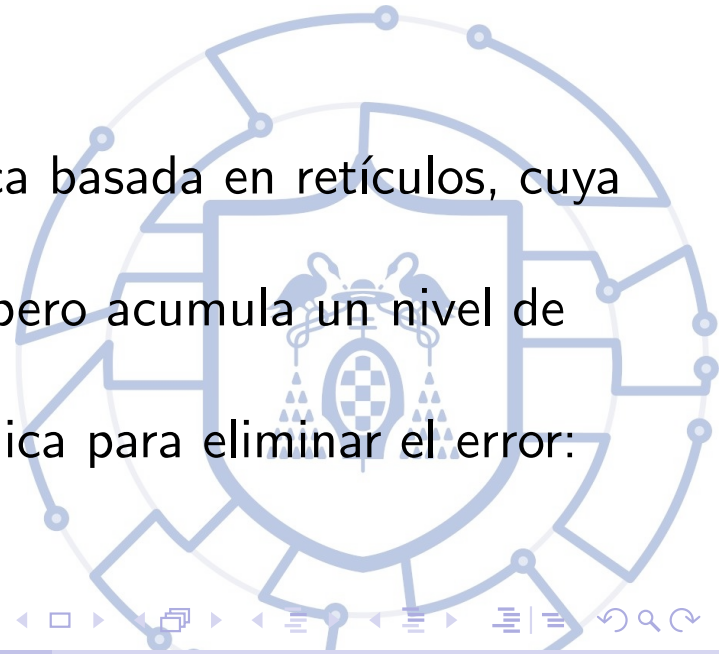
- Secure multi-party computation
  - Intersección de conjuntos privados (problema de los millonarios de Yao)
  - Comparación de hashes de las entradas de cada conjunto
  - Ejemplo: Dos agendas de contactos
    - Ordenación segura
    - Comparación segura
    - No aporta seguridad semántica
- Filtros de Bloom
  - Estructuras que permiten comprobar si un elemento pertenece a un conjunto sin revelarlo.
  - Vector de valores binario inicializado con todas las entradas a 0
  - Introducir valor: se procesa por varias funciones hash. Resultado: marcará una posición del vector
  - Comprobar valor: comprobar si las posiciones estan marcadas




Hash
Hashes
Check & Insert
Filtro
H1
H2
H3
H4
F1
F2
F3
F4
A
No existe
B
No existe
A
Puede existir
C
Puede existir

Generaciones

- **Objetivo:** crear un criptosistema que permita aplicar las operaciones suma y producto (puertas AND y XOR)
- Pre-HE
  - Em 1978, **Rivest, Adleman y Dertouzos** presentan un sistema criptografico que permite realizar operaciones sobre el texto cifrado
  - Criptosistema de **Paillier**, homomórfico respecto a la suma.
  - Homomorfismos no computables
- Primera generación [4]
  - En 2005, **Regev**, aproximación criptográfica basada en retículos, cuya funcion trampa esta basada en **LWE**.
  - Homomorfismo en la suma y el producto, pero acumula un nivel de error que corrompe el resultado
  - En 2009, **Craig Gentry**, presenta una técnica para eliminar el error: **Bootstraping**



Generaciones

- Segunda generación
  - Se desarrollan algoritmos de criptografia homomórfica válidos para la computación (Brakerski, Vaikuntanathan, Vercauteren y Fan)
  - Utilizan **grandes espacios numéricos** en comparación el tamaño del conjunto del error
  - Definiendo el límite de integridad de datos, se puede calcular el **número de operaciones** realizables
- Tercera generación
  - Gentry, Sahai y Waters diseñan un algoritmo capaz de computar el producto y la suma de una forma eficiente y aplicable a la computación arbitraria
  - Algoritmo de tipo **FHE**
  - A partir de GSW, los investigadores se han centrado en la **eficiencia** de las soluciones y la **usabilidad**

Innovación

- **Objetivo:** Construir sistemas aportando un grado mayor de **privacidad** y **confidencialidad**
- Aplicaciones:
  - **Sistema de voto electronico**
  - **Aplicaciones médicas**
    - Búsquedas en bases de datos médicos cifrados
    - Uso de aplicaciones móviles para la realización de dignosticos médicos
    - Aplicaciones en genomas
  - **Sistemas de control**
    - Redes eléctricas inteligentes
    - Monitorización de coches policías, ambulancias y bomberos desde la nube
  - **Análisis predictivo**

Homomorfismo en el logaritmo discreto

- Logaritmo discreto, complejidad computacional
- RSA homomórfico en el producto

Fórmula

$$c_1 = m_1^e \ mod \ n$$

$$c_2 = m_2^e \ mod \ n$$

$$c_3 = c_1 * c_2$$

$$c_3 = (m_1^e mod \ n) * (m_2^e \ mod \ n)$$

$$c_3 = (m_1 * m_2)^e \ mod \ n$$

Homomorfismo en el logaritmo discreto

Entonces

$$m_3 = c_3^d \ mod \ n$$

$$m_3 = ((m_1 * m_2)^e)^d \ mod \ n$$

$$m_3 = m_1 * m_2$$

Por lo tanto

$$desc(c_1 * c_2) = desc(c_1) * desc(c_2)$$

Homomorfismo en la asunción del residuo compuesto

- Dificultad encontrar residuos compuestos ($z = y^n \ mod \ n^2$)
- Criptosistema de Paillier homomórfico en la suma [3]

Clave pública (n, g), clave privada (l, u)

$$c(m) = g^m * r^n \ mod \ n^2$$

$$c_3 = c_1 * c_2$$

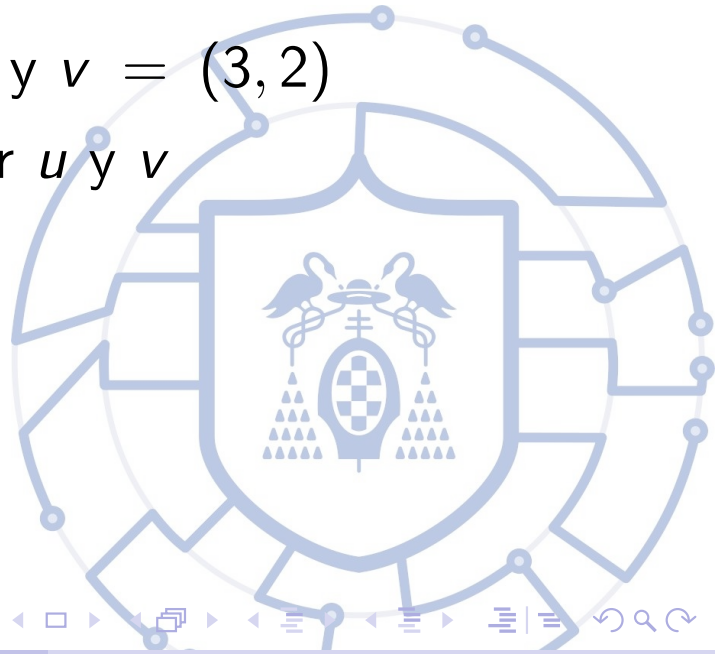
$$c_3 = (g^{m_1} * r_1^n \ mod \ n^2) * (g^{m_2} * r_2^n \ mod \ n^2)$$

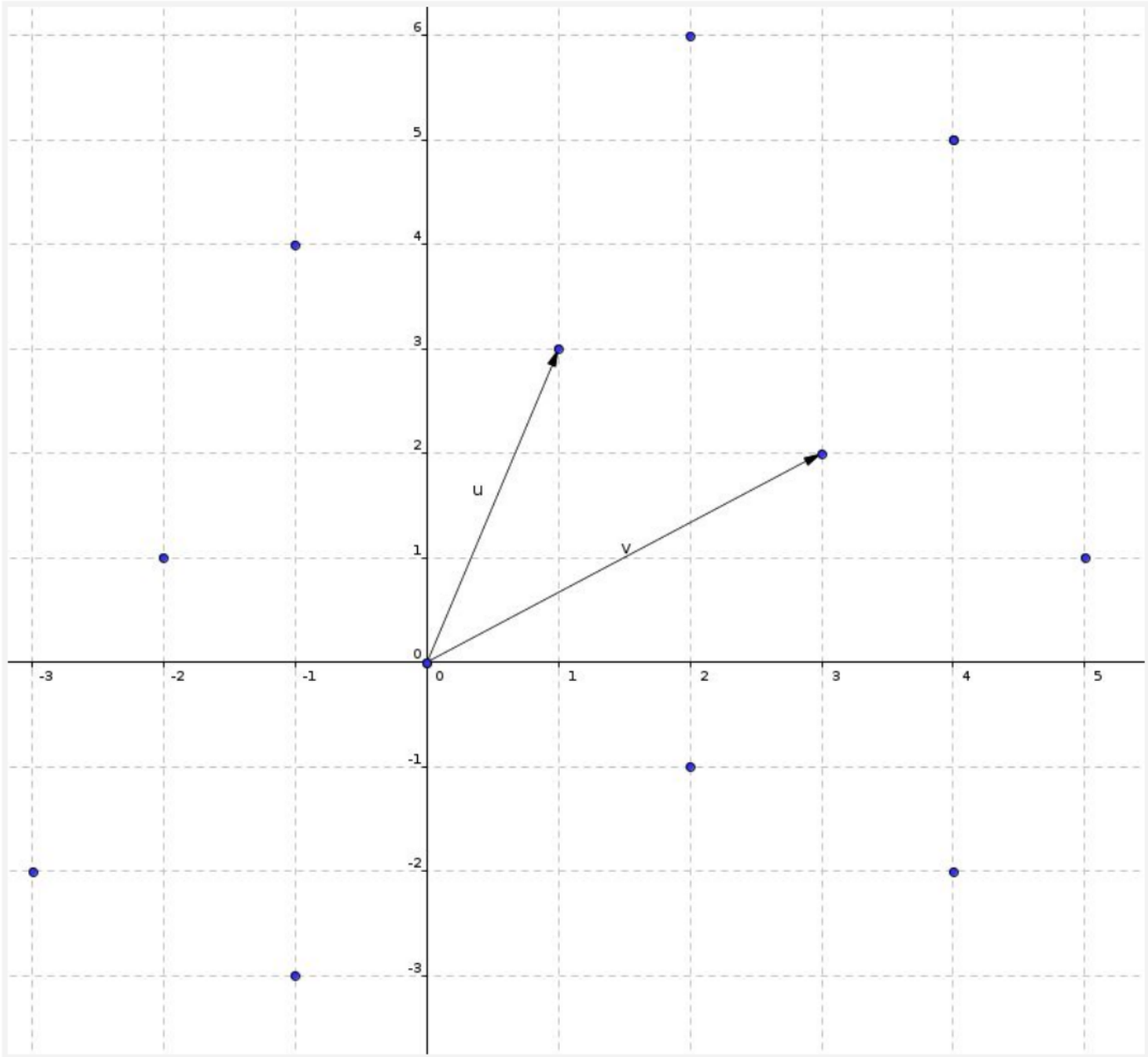
$$c_3 = (g^{m_1 + m_2}) * (r_1 * r_2)^n mod \ n^2$$

$$desc(c_1, c_2) = desc(c_1) + desc(c_2)$$

Cifrado basado en retículos [2]

- Función trampa $\rightarrow$ Clase NP (no resoluble en tiempo polinómico)
- Retículos: Estructuras algebraicas similares a espacios vectoriales, generados con una base
- Base formada por los elementos $u = (1, 3)$ y $v = (3, 2)$
- Posible formar $w = (4, 5)$. Resultado sumar u y v




u
v

- Dos problemas más representativos
  - SVP (Shortest Vector Problem)
    - Encontrar vector cercano al origen del espacio (i.e. punto que equivale a 0) dada una base larga
  - CVP (Closest Vector Problem)
    - Combinación de elementos de una base larga que resulte en el punto del espacio más cercano a un punto dado

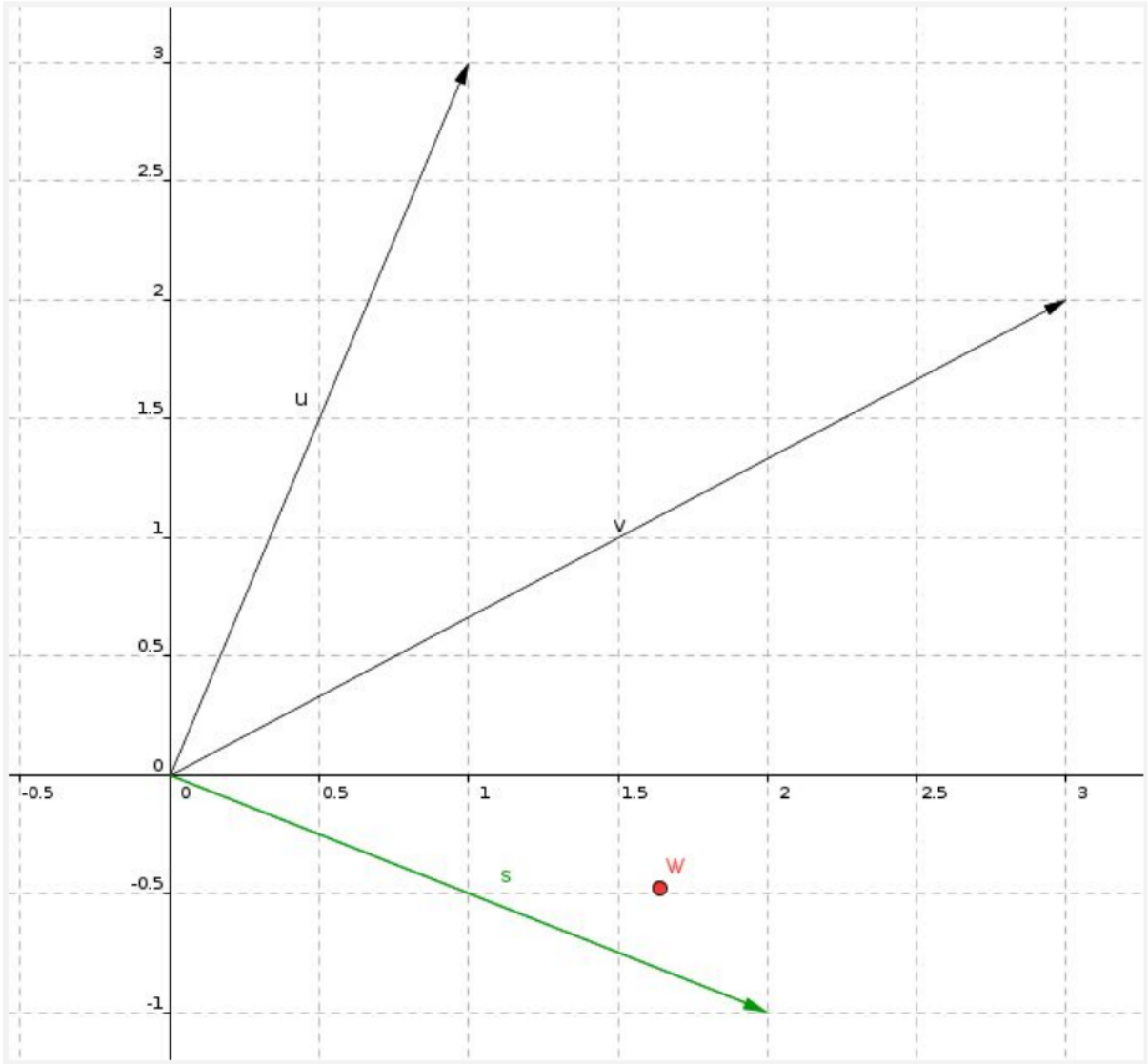
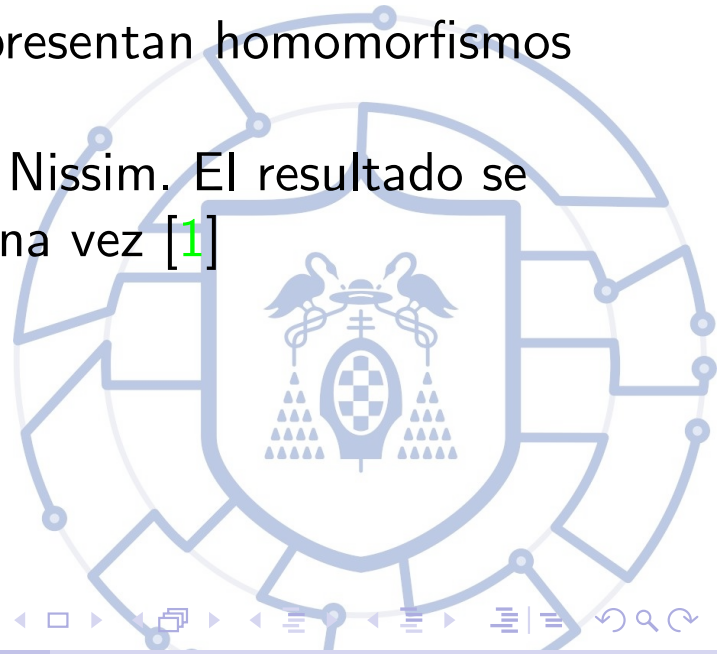


Figura: Ejemplo de CVP (combinación de u y v más cerca al punto w)

- LWE (Learn With Errors)
  - Problema plantea dificultad de determinar qué variables se han introducido en una función polinómica si se introduce un error aleatorio
  - Método de Regev
- Bootstrapping
  - Técnica que trata de eliminar el problema del ruido introduciendo la operación *Recrypt*
  - Utiliza claves especiales para eliminar el ruido cuando se opera
  - Nuevo cálculo $\rightarrow$ Gran coste de eficiencia

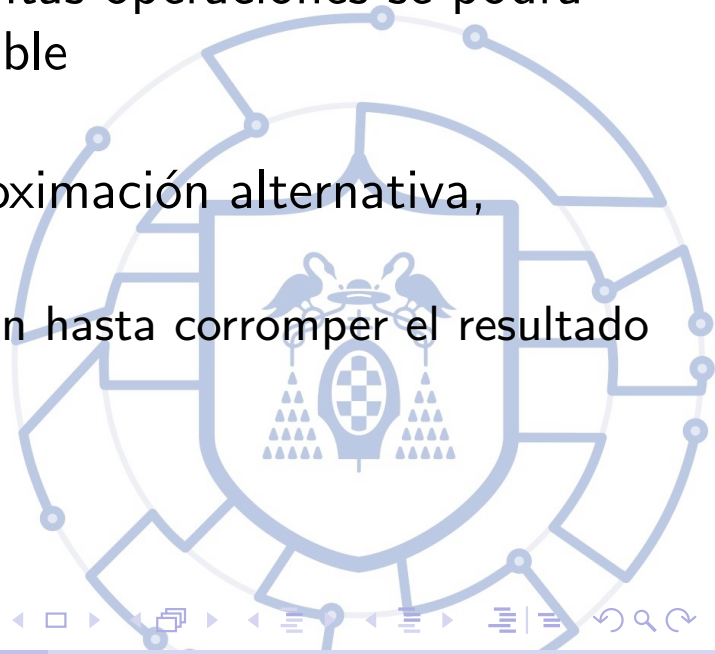
Algoritmos

- Se dividen en base a la capacidad que tengan para la computación, relación con las generaciones que hemos visto
- Partially Homomorphic Encryption
  - Por casualidad o de manera intencionada presentan homomorfismos
  - RSA, sistema de Paillier
  - Sistema implementado por Bonneh, Goh y Nissim. El resultado se corrompe si el producto se repite más de una vez [1]



Somewhat Homomorphic Encryption

- Somewhat Homomorphic Encryption
  - Realizan cualquier operación lógica pero limitan la capacidad de cómputo
  - Sistemas basados en control de ruido. Cuantas operaciones se podrá realizar antes de que este error sea inasumible
  - Sistemas de aproximación y truncado. Aproximación alternativa, elimina los valores menos significativos
    - El ruido se reduce pero se pierde precisión hasta corromper el resultado



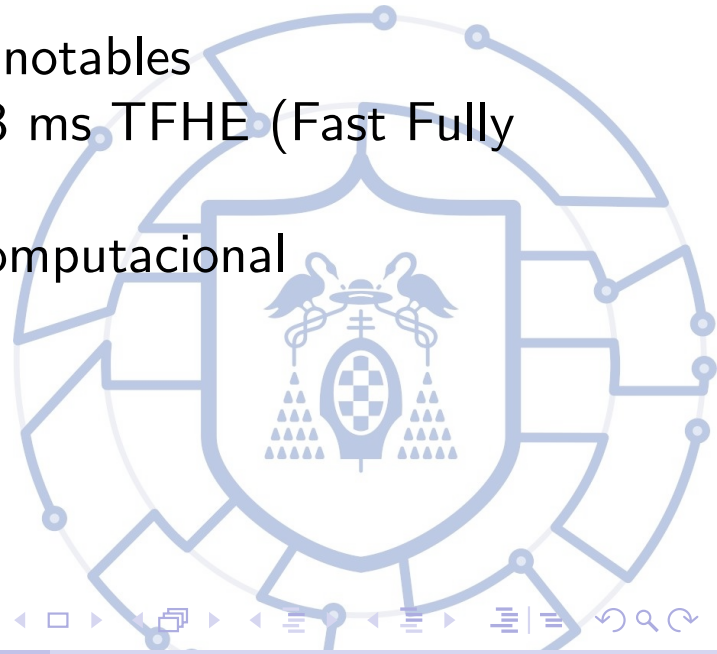
Fully Homomorphic Encryption

- Fully Homomorphic Encryption
  - Reducción de complejidad a operaciones aritméticas
  - TFHE (Fast Fully Homomorphic Encryption) Capacidad de evaluar una operación lógica en 13 milisegundos (frente a los 6 minutos que podía tardar bootstraping)
    - Lejos de las 1,417e9 operaciones de coma flotante que podría realizar un procesador actual en el mismo tiempo

Retos de la criptografía homomórfica

- ¿Por qué no hemos convertido nuestros sistemas para que funcionen con criptografía homomórfica?

- Eficiencia
  - Diferencias de eficiencia entre SHE y FHE notables
  - 6 minutos bootstraping, 0,5 seg FHEW, 13 ms TFHE (Fast Fully Homomorphic Encryption)
  - Problema viene dado por la complejidad computacional

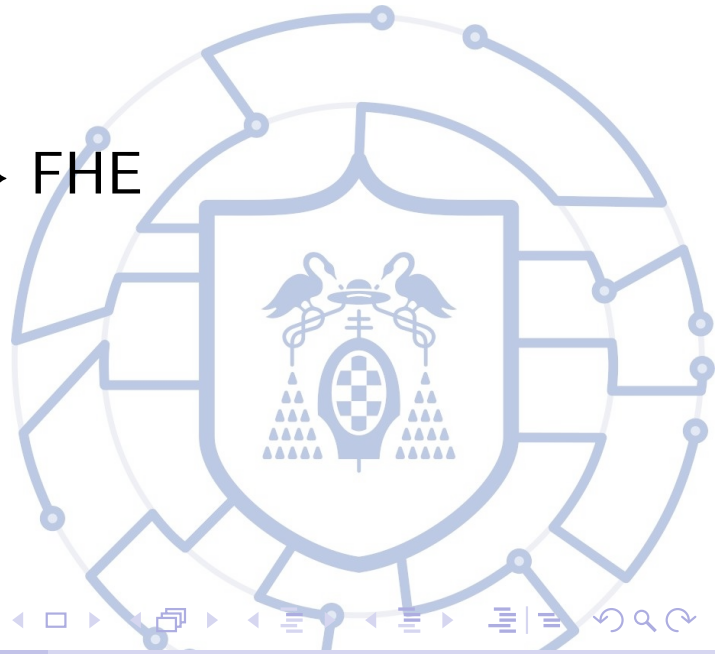


Retos de la criptografía homomórfica

- Usabilidad
  - Sistemas demasiados complejos para ofrecerlos como solución
  - El objetivo es mejorar la seguridad del sistema
    - Mala implementación $\rightarrow$ Grave riesgo
  - Falta de estandarización $\rightarrow$ Mina la posible inversión en ello
- Seguridad
  - Los algoritmos son robustos
    - Fácil realizar una mala implementación (algo común)
  - Sistemas mucho más complejos
    - Menos escrutados $\rightarrow$ Más fácil que puedan plantear fallas

Conclusiones

- Continuo desarrollo
- Salto cualitativo importante (esquemas más eficientes)
- Carencia de madurez para sustituirlo en algunos sistemas
  - Servir como complemento estratégico
- SHE para operaciones con cierta eficiencia
- Operaciones con profundidad operacional $\rightarrow$ FHE
- Valoración de eficacia, sin olvidar seguridad



Bibliografía I

Dan Boneh, Eu-Jin Goh y Kobbi Nissim. "Evaluating 2-DNF Formulas on Ciphertexts". En: *Theory of Cryptography*. Ed. por Joe Kilian. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, págs. 325-341. ISBN: 978-3-540-30576-7.

Craig Gentry. "Fully Homomorphic Encryption Using Ideal Lattices". En: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. STOC '09. Bethesda, MD, USA: Association for Computing Machinery, 2009, págs. 169-178. ISBN: 9781605585062. DOI: 10.1145/1536414.1536440. URL: https://doi.org/10.1145/1536414.1536440.

Bibliografía II

Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". En: *Advances in Cryptology — EUROCRYPT '99*. Ed. por Jacques Stern. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, págs. 223-238. ISBN: 978-3-540-48910-8.

Oded Regev. "New Lattice-Based Cryptographic Constructions". En: *J. ACM* 51.6 (nov. de 2004), págs. 899-942. ISSN: 0004-5411. DOI: 10.1145/1039488.1039490. URL: https://doi.org/10.1145/1039488.1039490.

¡Gracias!
¿Preguntas?