Information Security

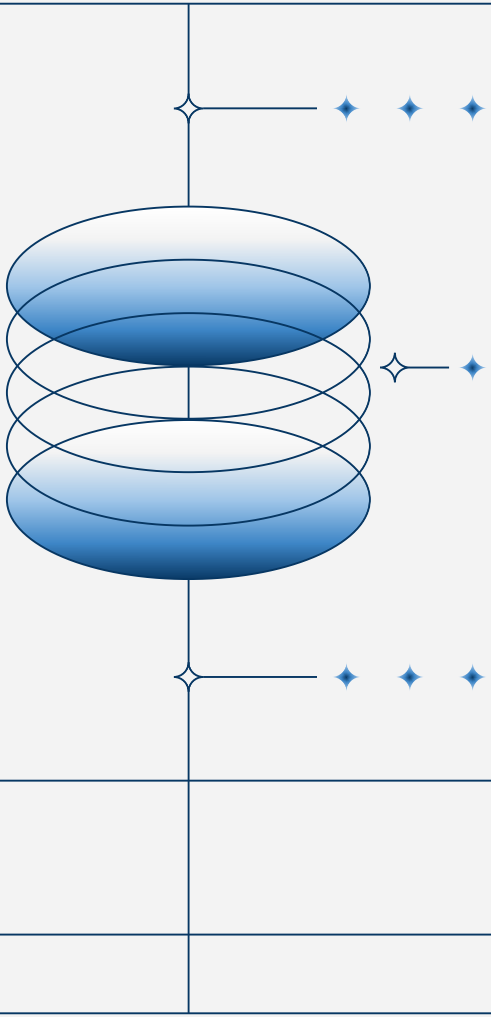
19102096 Donghwan Lee
19102127 Suho Lee

TABLE OF CONTENTS

01

Homomorphic Encryption

Concept, Progress, Advantage & Challenge



Introduction

Jewelry processing – Metaphor for Homomorphic Encryption

- **Homomorphic encryption** is a form of encryption in which, as the "homomorphic" part of the name implies, manipulating the encrypted data produces **the same sequence** as manipulating the original data.

- This characteristic is often expressed in the context of processing jewelry in a display case. The jeweler processes the jewelry at the owner's request, but he **never actually touches** the jewelry with his bare hands.

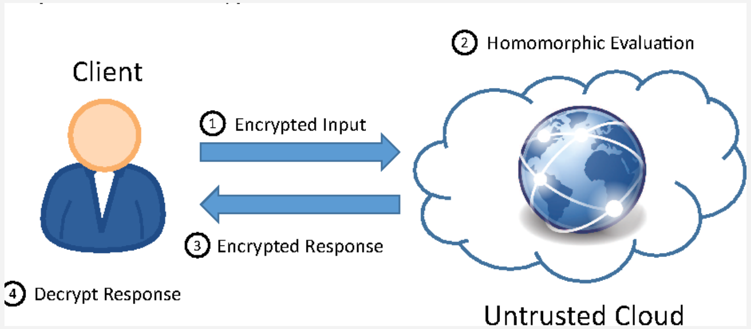
- Instead, he can process the jewelry **indirectly** through gloves that are connected to the display case. The Homomorphic Encryption **corresponds to** the display case surrounding the jewelry.

- **The only person** who can open the safe and access the gem itself is **the owner** of the safe key, and the jeweler cannot obtain the gem itself unless he somehow obtains the key.

Concept

Concept of Homomorphic Encryption

- Homomorphic Encryption allows **homomorphic Evaluation** of encrypted data.
- It is **almost impossible** for a company to know the actual content of the data, and this provides high security for the privacy of the user.
- **At the same time** providing the efficiency that allows the necessary operations to be carried out.



Flow of Data under the homomorphic encryption system

Progress

Type of Homomorphic Encryption

- **[Partially homomorphic encryption]** encompasses schemes that support the evaluation of circuits consisting of **only one type** of gate, e.g., addition or multiplication.

- **[Somewhat homomorphic encryption]** schemes can evaluate two types of gates, but only for a **subset of circuits**.

- **[Leveled fully homomorphic encryption]** supports the evaluation of arbitrary circuits composed of multiple types of gates of bounded (**pre-determined**) depth.

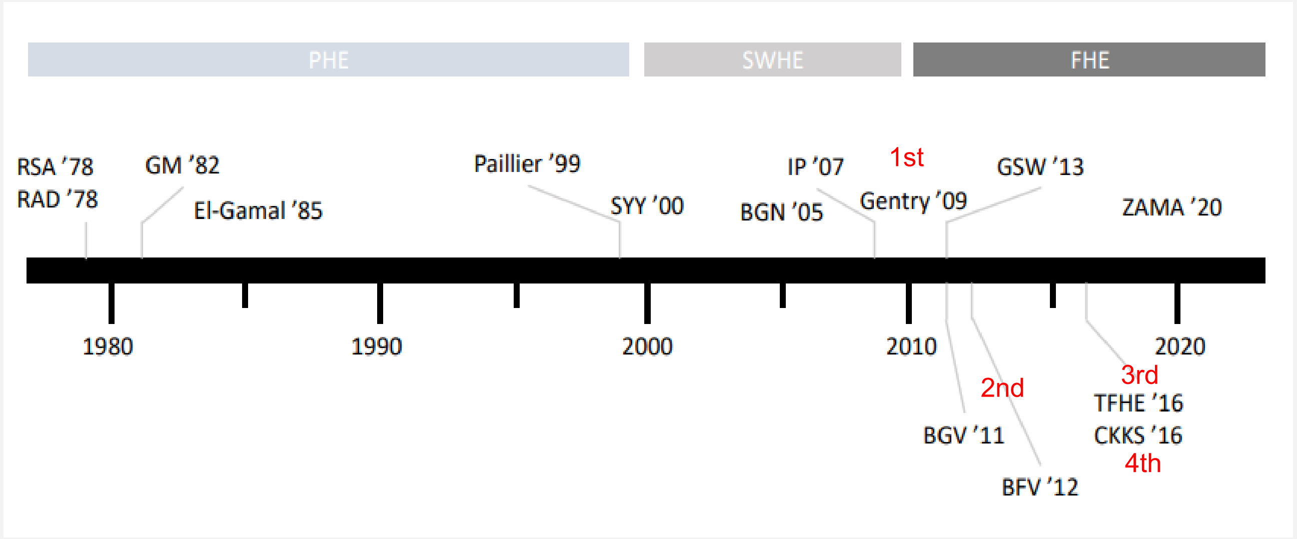
- **[Fully homomorphic encryption (FHE)]** allows the evaluation of arbitrary circuits composed of multiple types of gates of **unbounded depth** and is the strongest notion of homomorphic encryption.

Progress

History of Homomorphic Encryption

So far, homomorphic encryption has been divided into **four generations** based on three main type.

Here's a high-level view of how it all works



Scheme

Fully Homomorphic Schemes

[BFV]

• Message type : Integer number
• The noise value is added to **the lower part** of the encrypted message
→ **Easily invade** the plaintext value when the size of the noise value increases
• Perform additional operations to reduce the impact of noise to a ciphertext
(Reboot / Bootstrapping / Re-linearization)

[BGV]

• Message type : Integer number
• The noise value is added to **the upper part** of the encrypted message
→ **Do not easily invade** the value of original message even if the size of the noise value increases

Scheme

Fully Homomorphic Schemes (2)

[CKKS]

- Message type : Real number
- The noise value is added to **the lower part** of the encrypted message (same as BFV)
- Rescaling / Bootstrapping operation **exists.**

-> It reduces the noise in ciphertexts and enables more multiplications
-> It divides the ciphertext by the scaling factor and discards the last prime from the modulus

[TFHE]

- Message type : Binary number
- **Operate fast bootstrapping** & fast binary calculation of the ciphertext
- No SIMD (Simple Instruction Multiple Data) operation is applied.

Library

Homomorphic Libraries

Library name	Developer	Schemes	Description
Helib	IBM	BGV/CKKS	BGV scheme with the GHS optimization
SEAL	Microsoft	BGV/CKKS/BFV	
Lattigo	EPFL-LDS	BGV/CKKS/BFV	Implementation in Go
HEaaN	CryptoLab	CKKS	
OpenFHE	Duality Tech.	BGV/CKKS/BFV/TFHE/FHEW	Succeeded from PALISADE
TFHE -rs	Zama	TFHE	Supporting boolean , integer operation

Advantage & Challenge

Advantages of Homomorphic Encryption

- Before homomorphic encryption, encrypted data could be used to protect information, but the data **must be decrypted first** in order to process it.
- This meant that users' **information had to be revealed**, potentially putting them at **risk**.
- However, this problem **can be solved** with the introduction of homomorphic encryption. As mentioned earlier, homomorphic encryption guarantees that operations can be performed on encrypted data, so users can request the necessary data processing **without exposing** their privacy.
- Furthermore, because homomorphic ciphers are based on the difficulty of solving **lattice-based problems**, they are **resistant** to decryption by **quantum computers**, which is recognized as a major risk for most ciphers in use today.

Advantage & Challenge

Challenges of Homomorphic Encryption

- There is **still a problem** that homomorphic encryption must overcome **- performance**.
- Because homomorphic encryption requires so much computation during the encryption process, it's difficult to embed the system with the power of a typical computer.
- Many researchers are working to speed up the process, but it's still **not fast enough** for commercialization.
- Until this limitation is overcome, homomorphic encryption is **just a beautiful formula** that **can't be used in practice.**
- We need to quickly develop ways to improve their performance.

02

Use Cases

Cloud Computing, IOT, Medical, Finance

Usecase

CASE 01– Secure Cloud Computing

1. Introduction

- Cloud computing offers numerous benefits such as cost reduction and resource maintenance simplicity. However, security remains a primary concern in cloud environments.

- Encrypting data before sending it to the cloud is essential. However, decryption is required at each operation, necessitating the client to provide their private key to the cloud provider for data decryption before executing necessary calculations.

Usecase

CASE 01– Secure Cloud Computing

2. Operation

- Fully Homomorphic Encryption (FHE) represents a significant advancement in Cloud Computing security.
- It enables the outsourcing of calculations on confidential data to the Cloud server while retaining the secret key.
- With FHE, the Cloud server can perform computations on encrypted data and return the decrypted results, allowing for secure data processing in the cloud without compromising confidentiality.

Usecase

CASE 01– Secure Cloud Computing

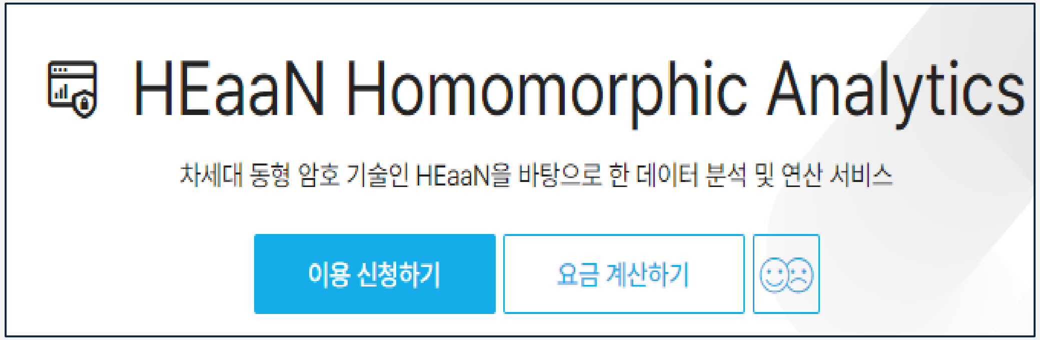
3. Example - Naver Cloud Platform

- CryptoLab's HEaaN, available on Naver's cloud platform, is the pioneering homomorphic encryption algorithm capable of computing real numbers.
- It ensures top-tier security while enabling limitless computation and analysis through data encryption.
- HEaaN maintains data integrity and is applicable across regulated industries like finance, healthcare, and the public sector.
- The Private Instance feature offers an autonomous analytics environment, allowing users to write data analytics codes using Jupyter Notebook and instantly view results.

Usecase

CASE 01– Secure Cloud Computing

4. Detail - Naver Cloud Platform



HEaaN Homomorphic Analytics

차세대 동형 암호 기술인 HEaaN을 바탕으로 한 데이터 분석 및 연산 서비스

- It generates a key from the console to be used for encryption. Uploaded data can then be encrypted in the cloud using the encryption key you generated, or uploaded to the cloud after pre -encryption locally.

- It uses uploaded and encrypted data to create projects and tasks and perform operations.

Usecase

Case 02 – FOG Computing for IOT

- FOG COMPUTING, initially proposed by CISCO to support large -scale scalable Internet of Things (IoT) deployment, has seen adoption in 5G/6G cellular networks under the name "edge computing."

- This concept offers data operations, storage, and application services similar to the cloud but provides users with services that mimic the cloud experience without relying on a central server.

- One critical aspect driving the need for fog computing is the demand for homomorphic encryption.

- By enabling preprocessing operations close to devices, fog computing reduces both bandwidth consumption and latency for IoT applications. Encrypted data facilitates preprocessing, making homomorphic encryption crucial in addressing privacy concerns.

Usecase

Case 02 – FOG Computing for IOT

- **Use homomorphic password**

  <Smart City Scenario>

  Smart City initiatives encompass a diverse array of applications, including intelligent transportation, efficient resource allocation (such as lighting, water, and waste management), safety and security measures, and environmental monitoring.

  These applications rely on extensive IoT deployments comprising small sensors that continuously transmit data to smart city data collectors.

  Within this framework, the fog layer plays a crucial role in preprocessing data at intermediate gateways. Additionally, to safeguard citizens' privacy, homomorphic encryption (HE) can be implemented as a protective measure.

Usecase

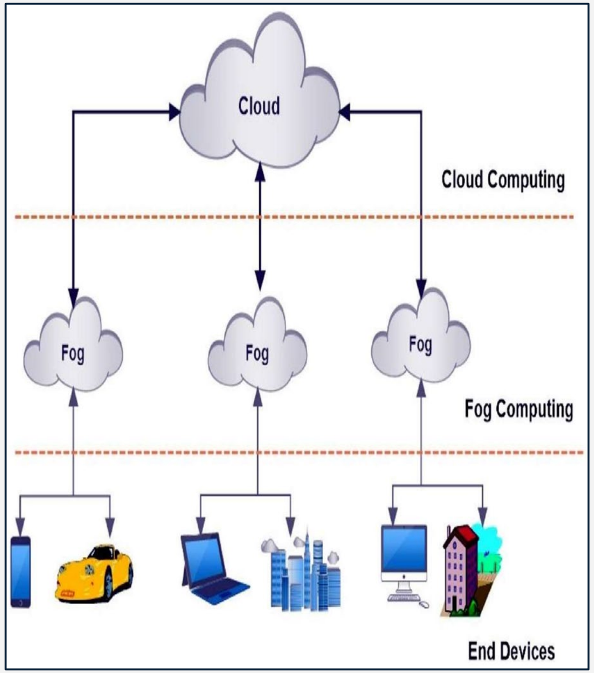
Case 02 – FOG Computing for IOT

- **To be specific,**

  In the context of smart city infrastructure, sensors play a pivotal role by encrypting data with the public key of the data collector before undergoing processing at the fog layer.

  This encrypted data can only be decrypted by the data collector using a secret key, ensuring that data exposure at fog nodes is prevented.

  Moreover, this approach offers privacy protection for IoT nodes concerning data collectors, as data samples are aggregated, thus preserving anonymity and confidentiality.

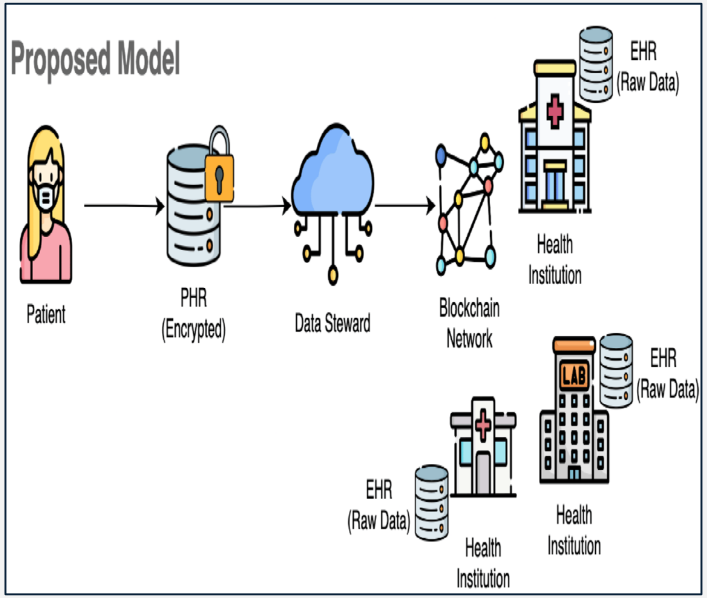


Usecase

Case 03 – Medical

- Homomorphic encryption, particularly the HEAAN (Homomorphic Encryption for Arithmetic of Approximate Numbers) scheme, presents a transformative solution for maintaining patient privacy while enabling computation on sensitive medical data.

- In this sector, we explore HEAAN's application in the medical sector, focusing on its ability to perform secure computations on encrypted data without decryption. We discuss its potential to revolutionize healthcare data privacy and analysis, its technical advantages, and its implications for medical research and personalized treatment.



Usecase

Case 03 – Medical

Behavioral analytics security technology minimizes risk of privacy breaches

- Efficient and robust cloud -based services are gaining popularity in healthcare as providers struggle to manually monitor numerous screens. However, privacy concerns regarding sharing sensitive personal data with cloud service providers pose a significant obstacle. Consequently, existing healthcare home monitoring services are restricted to monitoring limited areas.

- A cloud -based security algorithm addresses these concerns by encrypting body feature information, allowing it to infer basic daily movements or behaviors like falling. The encrypted data is transmitted to the cloud, where AI models analyze it to identify movements and falls. The results, also encrypted, are then forwarded to healthcare providers who can decrypt them, enabling immediate intervention if suspicious activities, such as falls, are detected.

- This technology enables the provision of reliable smart home monitoring services without compromising personal information privacy. Its potential applications extend across various healthcare service fields in the foreseeable future.

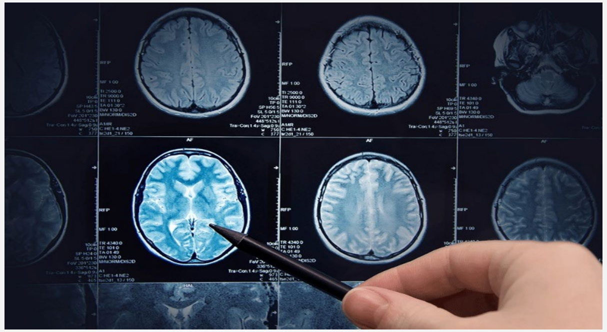
Usecase

Case 03 – Medical

- **Medical Imaging Analysis Processing**

  Emerging research and services focus on processing medical imaging to aid in disease diagnosis. However, medical imaging images contain highly sensitive patient medical information, making their utilization in their original state risky and potentially problematic for data analysis.

  To address this issue, homomorphic ciphers can be employed to allow the utilization of medical images without the risk of decryption. This approach enhances all aspects of patient care, including efficiency and the accuracy of record updates, thereby improving patient outcomes.



Usecase

Case 04 – Finance

Analyze credit data

- Homomorphic encryption enables combining and analyzing data without decryption, reducing the risk of leakage compared to plaintext analysis. By entrusting the decryption key to a trusted third party, concerns regarding personal information theft and leakage are significantly diminished.

- Linking credit scoring with homomorphic encryption ensures secure processing and storage of credit information. With this technology, credit bureaus or financial institutions can calculate credit scores without direct access to the encrypted data.

- Cryptolab , a specialist in homomorphic encryption, applies HEaaN's technology to analyze national pension data from approximately 2.34 million individuals and credit data from KCB, showcasing the practical application of homomorphic encryption in data analysis..

03

Our Service

[Finance] Secure credit scoring system

Secure credit scoring system

Service Overview

- A person's credit information is stored in homomorphic encryption, and a bank or financial institution can perform to calculate a credit score on this encrypted data.
- This way , your credit information **is not exposed to the outside world**, but the financial institution can still make a credit assessment based on the information they need.

Our Credit Score Formula

How do We set Credit Score Formula?

- This score can be used by lenders to determine whether to issue you a credit card or not, or whether you're a good or bad credit risk .
- However, the formula for calculating credit scores from "KCB" and "Nice" is not publicly available, so we don't know the formula .
- Therefore...

Replaced with a credit score calculation formula *we built ourselves*

Credit scoring methods & criteria

How do we set Credit Scoring Formula?

- A financial institution wants to build predictive and classification models using direct homomorphic encrypted data to determine customers instead of inaccurate credit scores .

- Formula

Credit Score = (w1×Income) + (w2×Age) + (w3×Number of children) + (w4×Whether or not you own a car) + (w5×Real estate) + (w6×Sector of income) + (w7×Education level) + (w8×Marital status) + (w9×How you live) + (w10×Whether you have a personal phone) + (w11×Whether you have a work phone) + (w12×Do you have a home phone) + (w13×Do you have email) + (w14×Occupation) + (w15×Number of family members)

Credit scoring methods & criteria

How do we get weight(w)?

1. Correlations between variables

2. Feature importance

⇒ Since they both result in a numerical value in decimal form, combine them to calculate the weight

And Finally…

use each individual's credit score on a 0-100 scale

2 things to address?

1. Distribution of values, e.g., categorical vs. integer data? How can feature importance be applied at once for values that vary in density (e.g., annual salary can range from 0 to 100 million, so if feature is high, categorical will measure low because it has a finite number of values)?

2. How to order the values in categorical data by what criteria?

Analytics process

After Calculating the Credit Score

1. classify whether the applicant's credit status is 'good' or 'bad' using customer history data **(credit_record.csv)**

2. Build a machine learning prediction model for credit card issuance using the applicant's customer information **(application_record.csv)** according to the above credit status classification .

=> Data obtained from Kaggle.Dataset

In the end

3 Processes

1. Calculate an individual's credit score with the formula we created
2. classify whether the applicant's credit status is 'good' or 'bad' using customer history data **(credit_record.csv)**
3. Build a machine learning prediction model for credit card issuance using the applicant's customer information **(application_record.csv)** according to the above credit status classification.

OUR TEAM

Donghwan Lee

ITM 19102096

Suho Lee

ITM 19102127

THANKS!