Pengamanan Model Machine Learning menggunakan Paillier Homomorphic Encryption

Implementasi dalam Email SPAM Detector

Ghebyon Tohada Nainggolan - 13520079(*Author*)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520079@std.stei.itb.ac.id ; ghebyon.tohada@gmail.com

***Abstract*— Keamanan data dan privasi telah menjadi isu yang krusial. Dalam konteks model machine learning, penting untuk menjaga kerahasiaan data pelatihan dan menjaga privasi data yang digunakan untuk membuat prediksi. Untuk mencapai tujuan ini, penulis mengusulkan penggunaan Paillier Homomorphic Encryption sebagai mekanisme keamanan untuk melindungi model machine learning. Paillier Homomorphic Encryption memungkinkan operasi matematika pada data terenkripsi. Makalah ini menyajikan implementasi pengamanan model machine learning menggunakan Paillier Homomorphic Encryption pada Email SPAM Detector.**

Kata Kunci—Paillier Homomorphic Encryption, machine learning, pengamanan data, privasi, email SPAM Detector

I. PENDAHULUAN

Dalam dunia yang semakin terhubung saat ini, pengiriman dan penerimaan email telah menjadi bagian penting dari komunikasi sehari-hari. Namun, masalah email SPAM yang tidak diinginkan tetap menjadi tantangan yang perlu diatasi. Salah satu pendekatan yang efektif dalam memerangi email SPAM adalah dengan menggunakan model machine learning untuk mengklasifikasikan email sebagai spam atau bukan spam.

Dalam contoh kasus ini, Alice, seorang pengguna email, telah melatih sebuah pengklasifikasi spam menggunakan regresi logistik pada kumpulan data email yang dimilikinya. Namun, Alice ingin menerapkan model yang telah ia pelajari kepada email pribadi Bob tanpa melibatkan pertukaran email atau mengungkapkan informasi sensitif seperti model yang dipelajari atau kumpulan data yang digunakan dalam pelatihan.

Untuk menjaga kerahasiaan data dan privasi, Alice menggunakan skema enkripsi homomorfik Paillier. Setelah proses pelatihan, Alice mengenkripsi model yang telah dibangun menggunakan kunci publik dari skema Paillier. Kunci publik dan model terenkripsi kemudian dikirim ke Bob. Bob menerapkan model terenkripsi tersebut pada data email pribadinya, menghasilkan skor terenkripsi untuk setiap email. Skor terenkripsi ini kemudian dikirim kembali kepada Alice. Dengan menggunakan kunci privat, Alice mendekripsi skor tersebut untuk mendapatkan prediksi apakah email tersebut termasuk spam atau bukan.

Dalam penelitian ini, fokus saya adalah pada penggunaan skema enkripsi homomorfik Paillier untuk menjaga kerahasiaan data dan privasi dalam pengklasifikasi email SPAM. Dengan menggunakan pendekatan ini, Alice dapat menerapkan model yang telah dipelajarinya pada email pribadi Bob tanpa harus mengorbankan kerahasiaan data atau memerlukan pertukaran email yang berisiko.

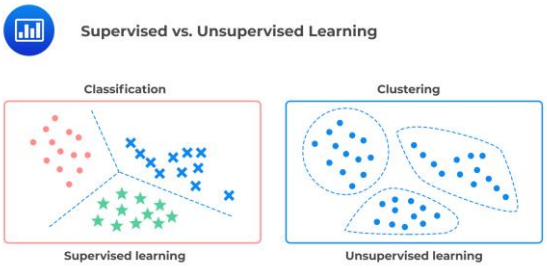
II. LANDASAN TEORI

A. *Machine Learning*

Machine learning adalah bidang ilmu komputer yang memberi komputer kemampuan untuk belajar tanpa diprogram secara eksplisit. Dengan kata lain, algoritma pembelajaran mesin dapat belajar dari data dan meningkatkan kinerjanya dari waktu ke waktu tanpa harus diberi tahu apa yang harus dilakukan.

Ada dua jenis utama *machine learning*: *supervised learning* dan *unsupervised learning*. Dalam *supervised learning*, algoritma diberikan sekumpulan data berlabel, artinya data tersebut telah diberi tag dengan keluaran yang benar. Algoritma kemudian belajar mengasosiasikan data input dengan output yang benar. Sebagai contoh, algoritma *supervised learing* dapat dilatih untuk mengklasifikasikan gambar kucing dan anjing dengan memberikan sekumpulan gambar yang telah diberi label sebagai kucing atau anjing.

Dalam pembelajaran *unsupervised learning*, algoritma tidak diberikan data berlabel apa pun. Alih-alih, algoritma diberikan sekumpulan data yang tidak berlabel dan harus belajar menemukan pola dalam data itu sendiri. Misalnya, algoritma *unsupervised learing* dapat digunakan untuk mengelompokkan kumpulan dokumen dengan menemukan kelompok dokumen yang mirip satu sama lain.



Gambar 1. *Supervised Learning* dan *Unsupervised Learning*

Sumber : https://analystprep.com/study-notes/wp-content/uploads/2021/03/Img_12.jpg

Algoritma *machine learning* dapat digunakan untuk memecahkan berbagai macam masalah. Beberapa aplikasi umum *machine learning* meliputi:

- Classification: Mengkategorikan data ke dalam kelompok yang berbeda
- *Regression*: Memprediksi nilai kontinu dari sekumpulan data masukan
- *Clustering*: Menemukan kelompok titik data yang serupa
- *Natural Language Processing*: Memahami dan memproses bahasa manusia
- *Computer Vision*: Mengidentifikasi dan memahami objek dalam gambar dan video

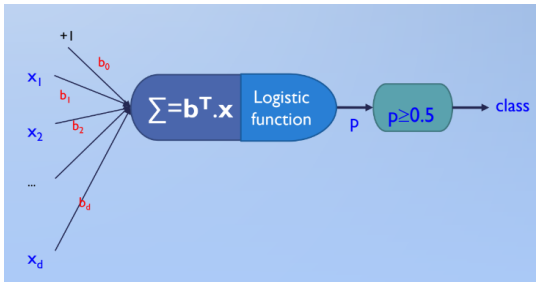
B. *Logistic Regression*

Logistic Regression adalah model statistik yang digunakan untuk memprediksi kemungkinan hasil biner, seperti apakah seorang pasien akan mengembangkan suatu penyakit atau tidak. Model didasarkan pada fungsi logistik, yaitu distribusi probabilitas yang digunakan untuk memodelkan probabilitas terjadinya suatu peristiwa.

Model *Logistic Regression* adalah model linier, artinya probabilitas hasil merupakan fungsi linier dari variabel prediktor. Variabel prediktor dapat berupa apa saja yang dianggap terkait dengan hasil, seperti usia, jenis kelamin, atau riwayat kesehatan.

Model *Logistic Regression* sesuai dengan data menggunakan prosedur estimasi kemungkinan maksimum. Prosedur estimasi kemungkinan maksimum menemukan nilai parameter model yang memaksimalkan kemungkinan data yang diamati.

Setelah model sesuai dengan data, model tersebut dapat digunakan untuk memprediksi probabilitas hasil untuk data baru. Probabilitas prediksi dihitung dengan memasukkan nilai variabel prediktor ke dalam persamaan model.



Gambar 2. Model *Logistic Regression*

Sumber : Slide Perkuliahan Machine Learning

Dalam implementasinya dikenal *Stochastic Gradient Ascent* untuk melatih model *Logistic Regression*. Berikut merupakan pseudocode untuk *training Logistic Regression*

```
//TRAINING DATA D={<X1,Y1>...<XN,YN>};
//              MAX-ITER T;
//              LEARNING RATE n
Intialize b
For t = 1, …, T:
    For each example <xi, yi>: #randomly chosen example
        pi=prediction for xi using the current coefficients b
        For each non-zero feature of xij: bj = bj + n(yi-pi)xij
Return b
```

Logistic Regression adalah *tool* yang dapat digunakan untuk memprediksi probabilitas hasil biner. Model ini mudah disesuaikan dan diinterpretasikan, dan relatif *robust* terhadap *assumptions violation*.

C. *Homomorphic Encryption*

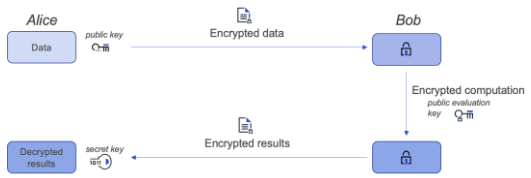
Enkripsi homomorfik adalah jenis enkripsi yang memungkinkan perhitungan dilakukan pada data terenkripsi tanpa mendekripsi data terlebih dahulu. Artinya, data terenkripsi dapat dibagikan dengan orang lain tanpa takut mereka dapat membaca data tersebut.

Ada dua jenis utama enkripsi homomorfik: enkripsi homomorfik aditif dan enkripsi homomorfik multiplikatif. Enkripsi homomorfik aditif memungkinkan operasi penambahan dilakukan pada data terenkripsi, sedangkan enkripsi homomorfik multiplikatif memungkinkan operasi perkalian dilakukan pada data terenkripsi.

Enkripsi homomorfik dapat digunakan untuk berbagai aplikasi, termasuk:

- Berbagi data dengan aman: Enkripsi homomorfik dapat digunakan untuk berbagi data secara aman dengan orang lain tanpa takut mereka dapat membaca data tersebut. Ini dapat berguna untuk berbagi data sensitif, seperti rekam medis atau informasi keuangan.
- Audit: Enkripsi homomorfik dapat digunakan untuk mengaudit data tanpa harus mendekripsi data terlebih dahulu. Hal ini dapat berguna untuk memastikan bahwa data tersebut akurat dan belum dirusak.

- Komputasi yang menjaga privasi: Enkripsi homomorfik dapat digunakan untuk melakukan komputasi pada data tanpa mengungkapkan data yang mendasarinya. Ini berguna untuk melindungi privasi individu, seperti saat melakukan penelitian medis atau analisis keuangan.



Gambar 3. Skema Homomorphic Encryption

Sumber : https://dualitytech.com/how-to-use-homomorphic-encryption-in-the-real-world/

D. *Paillier Homomorphic Encryption*

Enkripsi homomorfik Paillier adalah enkripsi homomorfik berjenis homomorfik aditif, yang berarti bahwa jumlah dari dua nilai terenkripsi dapat dihitung tanpa mendekripsi salah satu nilai. Properti ini membuat enkripsi homomorfik Paillier berguna untuk berbagai aplikasi, seperti komputasi multipartai yang aman dan pemungutan suara elektronik.

Enkripsi homomorfik Paillier didasarkan pada kesulitan masalah faktorisasi bilangan bulat. Kunci publik dari sistem kriptografi Paillier adalah produk dari dua bilangan prima besar, sedangkan kunci privat adalah nilai dari salah satu bilangan prima. Untuk mengenkripsi nilai, itu dikalikan dengan kunci publik. Untuk mendekripsi nilai terenkripsi, itu dikalikan dengan kunci pribadi dan kemudian dibagi dengan produk dari dua bilangan prima besar.

Properti tambahan dari enkripsi homomorfik Paillier dapat digunakan untuk menghitung jumlah dari dua nilai terenkripsi. Untuk melakukan ini, dua nilai terenkripsi dikalikan bersama dan kemudian hasil kali dibagi dengan hasil kali dari dua bilangan prima besar. Hasilnya adalah jumlah terenkripsi dari dua nilai asli.

Berikut merupakan skema algoritma enkripsi homomorfik Paillier :

Key Generation:

1. $Take\ two\ large\ prime\ number\ p\ and\ q, randomly, Confirm\ that\ gcd(pq, (p-1)(q-1))\ is$ 1.

   $if\ not\ start\ again$
2. $Compute\ n - pq.$
3. $Define\ fuction\ L(x) = (x-1)/n$
4. $Compute\ \lambda\ as\ lcm(p-1, q-1)$
5. $Pick\ a\ random\ integer\ g\ in\ the\ set\ \mathbb{Z}n$
6. $Calculate\ \mu = (L(g\text{^}(\lambda)\ mod\ n\text{^}(2)))\text{^}-1$
7. $The\ public\ key\ is\ (n, g)\ for\ encryption$
8. $The\ private\ key\ is\ \lambda\ (for\ decryption)$

Encryption:

1. $let\ m\ be\ a\ message\ to\ be\ encrypted\ where\ 0 \le m \le n$
2. $Select\ random\ r\ where\ 0 < r < n\ and\ r \in \mathbb{Z}n$
3. $Compute\ ciphertext\ as: c = g\text{^}(m).\ r\text{^}(n)\ mod\ n\text{^}(2)$

Decryption

1. $let\ c\ be\ the\ ciphertext\ to\ decrypt, where\ c \in \mathbb{Z}n2$
2. $Compute\ the\ plain\ text\ message\ as$

   $m = L(c\text{^}(\lambda)\ mod\ n\text{^}(2)).\ \mu\ mod\ n$

III. Analisis Masalah dan Rancangan Solusi

A. *Masalah*

Dalam contoh kasus ini, Alice, seorang pengguna email, telah melatih sebuah pengklasifikasi spam menggunakan *Logistic Regression* pada kumpulan data email yang dimilikinya. Namun, Alice ingin menerapkan model yang telah ia pelajari kepada email pribadi Bob tanpa :

- Meminta Bob untuk mengirim email yang dimilikinya karena email-email tersebut bersifat *private* untuk Bob
- Membocorkan informasi tentang model dan kumpulan data yang dimiliki Alice

Fokus saya pada penelitian ini adalah penerapan enkripsi homomorfiknya. Hal ini pula yang menjadi konsiderasi penggunaan model *Logistic Regression* agar perhitungan yang digunakan bisa diterapkan menggunakan *Paillier Homomorphic Encryption* dan juga tidak ada masalah pada penerapan model tersebut pada data yang digunakan.

B. *Rancangan Solusi*

Skema/algoritma yang diimplementasikan dalam makalah ini adalah sebagai berikut :

1) Pengumpulan dan *Preprocessing* Data:
   a) Alice mengunduh data email yang dia miliki
   b) Bob mengunduh data email yang dia miliki
   c) Masing-masing Alice dan Bob melakukan CountVectorizer pada email sehingga didapat jumlah kemunculan kata-kata atau token dalam setiap dokumennya. (Proses ini terjadi pada *preprocess_data()*)

```
download_data()
X, y, X_test, y_test = preprocess_data()
```

Gambar 4. Pengumpulan dan *Preprocessing* Data

Sumber : Dokumen Penulis

Dalam hal ini X dan y merupakan data yang dimiliki Alice dan akan dijadikan sebagai data train Alice. X_test merupakan data yang dimiliki oleh Bob. Serta

y_test akan digunakan sebagai pembanding hasil klasifikasi

2) Pelatihan Model *Logistic Regression*:
   a) Model Logistic Regression dilatih pada data train di sisi Alice menggunakan library scikit-learn.
   b) Model Logistic Regression akan digunakan untuk mengklasifikasikan email menjadi spam atau bukan spam.

```
print("Alice: Learning spam classifier")
alice.fit(X, y)
```

Gambar 5. Pelatihan Model *Logistic Regression*

Sumber : Dokumen Penulis

3) Pembuatan Kunci Publik dan Privat Paillier:
   a) Kunci publik dan privat Paillier digenerate menggunakan skema kriptografi Paillier.
   b) Skema Paillier digunakan untuk melakukan enkripsi model dan skor prediksi.

```
alice.generate_paillier_keypair(n_length=1024)
```

Gambar 6. Pembuatan Kunci Publik dan Privat Paillier

Sumber : Dokumen Penulis

4) Enkripsi Model dan Pengiriman ke Bob
   a) Model Logistic Regression dienkripsi menggunakan kunci publik Paillier.
   b) Kunci publik dan model terenkripsi dikirim ke Bob untuk digunakan dalam pengklasifikasian email.

```
print("Alice: Encrypting classifier")
encrypted_weights, encrypted_intercept = alice.encrypt_weights()
bob = Bob(alice.pubkey)
bob.set_weights(encrypted_weights, encrypted_intercept)
```

Gambar 7. Enkripsi Model dan Pengiriman ke Bob

Sumber : Dokumen Penulis

5) Pengklasifikasian Email oleh Bob
   a) Bob menerapkan model terenkripsi pada data email pribadinya
   b) Bob memperoleh skor terenkripsi untuk setiap email.
   c) Bob mengirim skor terenkripsi kepada Alice

```
print("Bob: Scoring with encrypted classifier")
encrypted_scores = bob.encrypted_evaluate(X_test)
```

Gambar 8. Pengklasifikasian Email oleh Bob dan Pengiriman Skor Terenkripsi ke Alice

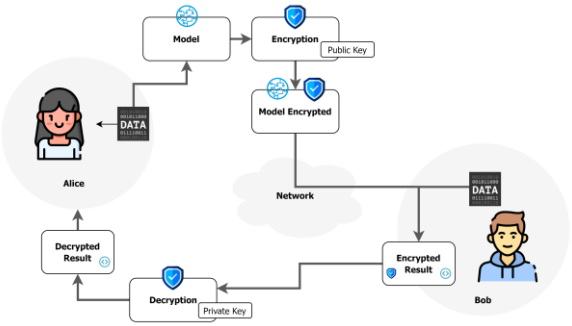
Sumber : Dokumen Penulis

6) Dekripsi Skor oleh Alice
   a) Alice menggunakan kunci privat Paillier untuk mendekripsi skor terenkripsi yang diterima dari Bob.
   b) Hasil dekripsi adalah prediksi skor spam vs. bukan spam.

```
print("Alice: Decrypting Bob's scores")
scores = alice.decrypt_scores(encrypted_scores)
```

Gambar 9. Dekripsi Skor oleh Alice

Sumber : Dokumen Penulis



Gambar 10. Skema Keseluruhan

Sumber : Dokumen Penulis

IV. IMPLEMENTASI

A. *Implementasi Sisi Alice*

```
class Alice:
    def __init__(self):
        self.model = LogisticRegression()

    def generate_paillier_keypair(self, n_length):
        self.pubkey, self.privkey = \
            paillier.generate_paillier_keypair(n_length=n_length)

    def fit(self, X, y):
        self.model = self.model.fit(X, y)

    def predict(self, X):
        return self.model.predict(X)

    def encrypt_weights(self):
        coef = self.model.coef_[0, :]
        encrypted_weights = [self.pubkey.encrypt(coef[i])
                             for i in range(coef.shape[0])]
        encrypted_intercept = self.pubkey.encrypt(self.model.intercept_[0])
        return encrypted_weights, encrypted_intercept

    def decrypt_scores(self, encrypted_scores):
        return [self.privkey.decrypt(s) for s in encrypted_scores]
```

Gambar 11. Implementasi Sisi Alice

Sumber : Dokumen Penulis

B. Implementasi Sisi Bob

```python
class Bob:
    def __init__(self, pubkey):
        self.pubkey = pubkey

    def set_weights(self, weights, intercept):
        self.weights = weights
        self.intercept = intercept

    def encrypted_score(self, x):
        score = self.intercept
        _, idx = x.nonzero()
        for i in idx:
            score += x[0, i] * self.weights[i]
        return score

    def encrypted_evaluate(self, X):
        return [self.encrypted_score(X[i, :]) for i in range(X.shape[0])]
```

Gambar 12. Implementasi Sisi Bob

Sumber : Dokumen Penulis

C. Simulasi

```python
if __name__ == '__main__':

    download_data()
    X, y, X_test, y_test = preprocess_data()

    print("Alice: Generating paillier keypair")
    alice = Alice()
    alice.generate_paillier_keypair(n_length=1024)

    print("Alice: Learning spam classifier")
    with timer() as t:
        alice.fit(X, y)

    print("Classify with model in the clear -- "
          "what Alice would get having Bob's data locally")
    with timer() as t:
        error = np.mean(alice.predict(X_test) != y_test)
    print("Error {:.3f}".format(error))

    print("Alice: Encrypting classifier")
    with timer() as t:
        encrypted_weights, encrypted_intercept = alice.encrypt_weights()

    print("Bob: Scoring with encrypted classifier")
    bob = Bob(alice.pubkey)
    bob.set_weights(encrypted_weights, encrypted_intercept)
    with timer() as t:
        encrypted_scores = bob.encrypted_evaluate(X_test)

    print("Alice: Decrypting Bob's scores")
    with timer() as t:
        scores = alice.decrypt_scores(encrypted_scores)
    error = np.mean(np.sign(scores) != y_test)
    print("Error {:.3f} -- this is not known to Alice, who does not possess "
          "the ground truth labels".format(error))
```

Gambar 13. Simulasi Main

Sumber : Dokumen Penulis

V. Hasil dan Analisis

```
Downloading 1/2: https://cloudstor.aarnet.edu.au/plus/index.php/s/RpHZ57z2E3BTiSQ/download
Downloading 2/2: https://cloudstor.aarnet.edu.au/plus/index.php/s/QVD4Xk5Cz3UVYLp/download
Importing dataset from disk...
Vocabulary size: 7994
Labels in trainset are 0.25 spam : 0.75 ham
Alice: Generating paillier keypair
Alice: Learning spam classifier
[elapsed time: 0.10 s]
Classify with model in the clear -- what Alice would get having Bob's data locally
[elapsed time: 0.00 s]
Error 0.040
Alice: Encrypting classifier
[elapsed time: 143.65 s]
Bob: Scoring with encrypted classifier
[elapsed time: 159.71 s]
Alice: Decrypting Bob's scores
[elapsed time: 47.36 s]
Error 0.040 -- this is not known to Alice, who does not possess the ground truth labels
```

Gambar 14. Hasil Run Program

Sumber : Dokumen Penulis

Hasil menunjukkan bahwa nilai error adalah 0,040 yang persis sama. Ini menunjukkan keberhasilan implementasi pengamanan model machine learning menggunakan Paillier Homomorphic Encryption. Selain menjaga keamanan model, data yang dimiliki oleh Bob juga tetap aman karena tidak ada pengiriman data dari Bob ke Alice. Komputasi dilakukan secara lokal oleh Bob.

Ada hal lain yang perlu diperhatikan, yaitu waktu klasifikasi antara model linear regression yang sudah dienkripsi dan model linear regression biasa. Menggunakan mesin yang sama, waktu klasifikasi pada model linear regression yang sudah dienkripsi dengan Paillier Homomorphic Encryption adalah sekitar 159,71 detik, sedangkan pada model linear regression biasa waktu klasifikasinya bahkan tidak mencapai 0,00 detik. Ini merupakan fakta yang benar. Oleh karena itu, tidak banyak mesin atau layanan komputasi Cloud yang mampu menyelesaikan pelatihan model machine learning dengan model terenkripsi.

VI. Kesimpulan

Enkripsi homomorfik adalah *tool* yang dapat digunakan untuk melindungi privasi data. Melalui pendekatan yang berbeda seperti pada kasus utama penelitian ini, enkripsi homomorfik dapat digunakan untuk menjaga keamanan model *machine learning* untuk mendeteksi spam email, serta menjaga keamanan data email pihak lain.

Namun, penting untuk dicatat bahwa enkripsi homomorfik tidaklah sempurna. Masih ada beberapa batasan untuk enkripsi homomorfik, seperti fakta bahwa mungkin mahal secara komputasi untuk melakukan komputasi pada data terenkripsi. Batasan dalam enkripsi homomorfik:

- Kompleksitas komputasi: Enkripsi homomorfik mahal secara komputasi. Ini berarti lambat untuk melakukan perhitungan pada data terenkripsi.
- Keamanan: Enkripsi homomorfik tidak sepenuhnya aman. Masih ada beberapa serangan yang dapat digunakan untuk memecahkan enkripsi homomorphic.
- Ketersediaan: Enkripsi homomorfik belum tersedia secara luas. Hanya ada beberapa sistem enkripsi homomorfik yang tersedia, dan belum digunakan secara luas.

Source Code Link at DeepNote

https://deepnote.com/workspace/makalah-kripto-4029764f-f5b7-49d9-8b49-c24b7de80016/project/PHE-in-SPAM-Email-Clsf-13072d18-e29d-4898-8c20-c27b5c82f62c/notebook/test-ebcc4809f1c44fcdaf961d0f20b3bc5b

Acknowledgment

Penulis mengucapkan terima kasih kepada Bapak Rinaldi Munir, dosen utama IF4020 Kriptografi yang telah mengajar dan membimbing penulis dalam mempelajari materi kriptografi sehingga penulis dapat menyelesaikan makalah ini. Penulis juga berterima kasih kepada keluarga dan pihak-pihak yang

telah memberikan dukungan berupa waktu dan sumber daya untuk menyelesaikan makalah ini.

REFERENCES

[1] R. Munir, "Enkripsi Homomorfik," ITB Informatika, 2022-2023. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/35-Enkripsi-homomorfik-2023.pdf

[2] A. Trask, "Homomorphic Surveillance," June 5, 2017. [Online]. Available: https://iamtrask.github.io/2017/06/05/homomorphic-surveillance/

[3] Ogunseyi TB, Bo T, editors. Fast Decryption Algorithm for Paillier Homomorphic Cryptosystem. 2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS); 2020: IEEE.

[4] Duality Technologies, "How to Use Homomorphic Encryption in the Real World," [Online]. Available: https://dualitytech.com/how-to-use-homomorphic-encryption-in-the-real-world/.