

IF4020 Kriptografi

Enkripsi Homomorfik



Oleh: Rinaldi Munir

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
ITB

Homomorphic Encryption



Komputasi pada cipherteks

- Bagaimana mengubah cipherteks sehingga ketika didekripsi hasilnya sama seperti mengubah plainteksnya?
- Cara konvensional:
 - (i) dekripsi cipherteks menjadi plainteks,
 - (ii) ubah plainteks seperti yang diinginkan,
 - (iii) enkripsi kembali plainteks hasil perubahan menjadi cipherteks yang baru.

→ Tidak aman jika hasil Langkah (ii) berhasil disadap oleh pihak lawan

Contoh: kasus e-voting

Enkripsi Homomorfik (EH)

- *Homomorphic encryption*
- EH melakukan komputasi pada cipherteks tanpa harus mendekripsi cipherteks tersebut,
- namun hasil komputasi pada cipherteks, bila didekripsi, memberikan hasil yang tepat sama bila komputasi yang serupa dilakukan pada plainteksnya.
- Misalkan E menyatakan fungsi enkripsi. Fungsi enkripsi E dikatakan homomorfik jika diberikan $E(x)$ dan $E(y)$, maka orang dapat memperoleh $E(x \diamond y)$ tanpa mendekripsi x dan y untuk beberapa operasi \diamond (Rivest, 2002).

Homomorfik aditif vs Homomorfik multiplikatif

- Sebuah algoritma enkripsi homomorfik dikatakan aditif (*additive*) jika

$$E(x + y) = E(x) \otimes E(y)$$

E adalah fungsi enkripsi, x dan y adalah plainteks, dan \otimes menyatakan operasi yang bergantung pada *cipher* yang digunakan (operasi $+$, \times , atau operasi lainnya).

- Sebuah algoritma enkripsi homomorfik dikatakan multiplikatif jika

$$E(x \cdot y) = E(x) \otimes E(y)$$

E adalah fungsi enkripsi, x dan y adalah plainteks, dan \otimes menyatakan operasi yang bergantung pada *cipher* yang digunakan (operasi $+$, \times , atau operasi lainnya).

Jenis-jenis enkripsi homomorfik

- 1. Enkripsi homomorfik sebagian** (*partially homomorphic encryption*)
Enkripsi homomorfik sebagian hanya memungkinkan satu operasi aritmetika pada cipherteks, yaitu operasi penjumlahan saja atau operasi perkalian saja.
- 2. Enkripsi homomorfik penuh** (*fully homomorphic encryption*),
operasi penjumlahan dan operasi perkalian dapat dilakukan terhadap cipherteks

Enkripsi homomorfik sebagian

- Enkripsi homomorfik sebagian hanya memiliki satu operasi saja pada cipherteks, yaitu operasi penjumlahan saja atau operasi perkalian saja, namun tidak keduanya.
- Jadi, enkripsi homomorfik hanya memiliki satu sifat, yaitu sifat aditif atau sifat multiplikatif. Dua buah cipherteks hanya dapat dijumlahkan saja atau dikalikan saja.
- Tiga buah algoritma enkripsi homomorfik sebagian: (1) algoritma RSA, (2) algoritma ElGamal, dan (3) algoritma Paillier.

Enkripsi Homomorfik dengan Algoritma RSA

- Algoritma RSA bersifat multiplikatif, yaitu hasil kali dua buah cipherteks apabila didekripsi hasilnya sama dengan mengalikan kedua plainteksnya
- Algoritma RSA:
Enkripsi: $E(m) = c = m^e \bmod n$
Dekripsi: $D(c) = m = c^d \bmod n$
- Misalkan
plainteks: m_1 dan m_2 ,
cipherteks: c_1 dan c_2 ,
 $E(m_1) = c_1 = m_1^e \bmod n$
 $E(m_2) = c_2 = m_2^e \bmod n$

- Kalikan kedua cipherteks:

$$\begin{aligned} E(m_1) \cdot E(m_2) &= c_1 \cdot c_2 = (m_1^e \bmod n) \cdot (m_2^e \bmod n) \\ &= (m_1 \cdot m_2)^e \bmod n \end{aligned}$$

Sedangkan $E(m_1 \cdot m_2) = (m_1 \cdot m_2)^e \bmod n$.

- Hasil ini memperlihatkan bahwa algoritma RSA memiliki sifat multiplikatif, yaitu

$$E(m_1 \cdot m_2) = E(m_1) \cdot E(m_2)$$

- Sekarang, akan ditunjukkan bahwa hasil dekripsi terhadap $c_1 \cdot c_2$ sama dengan perkalian kedua plainteksnya, yaitu

$$D(c_1 \cdot c_2) = (m_1 \cdot m_2) \bmod n.$$

Bukti:

$$\begin{aligned} D(c_1 \cdot c_2) &= (c_1 \cdot c_2)^d \pmod n \\ &= ((m_1 \cdot m_2)^e)^d \pmod n \\ &= (m_1 \cdot m_2)^{ed} \pmod n \end{aligned}$$

Karena $ed = k\phi(n) + 1$ (lihat penurunan rumus RSA), maka

$$\begin{aligned} &= (m_1 \cdot m_2)^{k\phi(n) + 1} \pmod n \\ &= (m_1 \cdot m_2) (m_1 \cdot m_2)^{k\phi(n)} \pmod n \end{aligned}$$

Karena $a^{k\phi(n)} \equiv 1 \pmod n$, maka $(m_1 \cdot m_2)^{k\phi(n)} \equiv 1 \pmod n$, sehingga

$$\begin{aligned} &= (m_1 \cdot m_2) \cdot 1 \pmod n \\ &= (m_1 \cdot m_2) \pmod n \end{aligned}$$

Oleh karena itu, kita berhasil menunjukkan bahwa

$$D(c_1 \cdot c_2) = (m_1 \cdot m_2) \pmod n$$

Contoh 1: Misalkan kunci publik Alice adalah ($n = 3337$, $e = 79$) dan kunci privatnya $d = 1019$. Misalkan $m_1 = 2671$ dan $m_2 = 1800$.

$$E(m_1) = c_1 = m_1^e \bmod n = 2671^{79} \bmod 3337 = 2081$$

$$E(m_2) = c_2 = m_2^e \bmod n = 1800^{79} \bmod 3337 = 338$$

$$E(m_1) \cdot E(m_2) = c_1 \cdot c_2 = 2081 \cdot 338 = 703378$$

$$D(c_1 \cdot c_2) = (c_1 \cdot c_2)^d \bmod n = (703378)^{1019} \bmod 3337 = \mathbf{2520}$$

Hasil terakhir ini, 2520, sama dengan mengalikan 2671 dengan 1800 dalam modulus 3337, yaitu

$$m_1 \cdot m_2 = (2671 \cdot 1800) \bmod 3337 = 4807800 \bmod 3337 = \mathbf{2520}$$

Jadi, $D(c_1 \cdot c_2) = m_1 \cdot m_2 = 2520$, artinya hasil kali dua buah cipherteks apabila didekripsi hasilnya sama dengan mengalikan kedua plainteksnya.

Untuk menunjukkan RSA adalah enkripsi homomorfik yang bersifat multiplikatif, maka

$$E(m_1 \cdot m_2) = E(2520) = 2520^{79} \bmod 3337 = \mathbf{2608}$$

$$= E(m_1) \cdot E(m_2) = (2081 \cdot 338) \bmod 3337 = \mathbf{2608}$$

yang menunjukkan bahwa $E(m_1 \cdot m_2) = E(m_1) \cdot E(m_2)$

Enkripsi Homomorfik dengan Algoritma ElGamal

- Seperti RSA, algoritma ElGamal juga bersifat multiplikatif.

Enkripsi: $(a, b) = (g^k \bmod p, y^k m \bmod p)$

Dekripsi: $m = (b/a^x) \bmod p$

- Misalkan plainteks: m_1 dan m_2 , cipherteksnya: (a_1, b_1) dan (a_2, b_2)

$E(m_1) = (a_1, b_1) = (g^{k_1} \bmod p, y^{k_1} m_1 \bmod p)$

$E(m_2) = (a_2, b_2) = (g^{k_2} \bmod p, y^{k_2} m_2 \bmod p)$

- Kalikan cipherteks (a_1, b_1) dan (a_2, b_2) sebagai berikut:

$E(m_1) \cdot E(m_2) = (a_1, b_1) \cdot (a_2, b_2)$

$= (a_1 \cdot a_2, b_1 \cdot b_2)$

$= (g^{k_1} \cdot g^{k_2}, y^{k_1} m_1 \cdot y^{k_2} m_2) \bmod p$

$= (g^{k_1 + k_2}, y^{k_1 + k_2} \cdot m_1 \cdot m_2) \bmod p$

- Sedangkan

$$E(m_1 \cdot m_2) = (a, b) = (g^k, y^k \cdot m_1 \cdot m_2) \text{ mod } p, \text{ dalam hal ini } k = k_1 + k_2$$

- Hasil ini memperlihatkan algoritma ElGamal memiliki sifat multiplikatif, yaitu

$$E(m_1 \cdot m_2) = E(m_1) \cdot E(m_2)$$

- Sekarang, akan ditunjukkan bahwa hasil dekripsi terhadap hasil kali dua cipherteks sama dengan perkalian kedua plainteksnya:

$$D((a_1, b_1) \cdot (a_2, b_2)) = (m_1 \cdot m_2) \text{ mod } p$$

Bukti:

$$\begin{aligned} D((a_1, b_1) \cdot (a_2, b_2)) &= D(g^{k_1 + k_2}, m_1 \cdot m_2 y^{k_1 + k_2}) \\ &= \frac{m_1 \cdot m_2 \cdot y^{k_1 + k_2}}{(g^{k_1 + k_2})^x} \pmod p \end{aligned}$$

Mengingat $g^x \equiv y \pmod p$, maka

$$\begin{aligned} &= \frac{m_1 \cdot m_2 \cdot (g^x)^{k_1 + k_2}}{(g^{k_1 + k_2})^x} \pmod p \\ &= \frac{m_1 \cdot m_2 \cdot (g^{k_1 + k_2})^x}{(g^{k_1 + k_2})^x} \pmod p \\ &= m_1 \cdot m_2 \pmod p \end{aligned}$$

Oleh karena itu, kita berhasil menunjukkan bahwa

$$D((a_1, b_1) \cdot (a_2, b_2)) = (m_1 \cdot m_2) \pmod p$$

Enkripsi Homomorfik dengan Algoritma Paillier

- Algoritma Paillier termasuk ke dalam algoritma kriptografi kunci-publik. Algoritma ini dikembangkan oleh Pascal Paillier pada tahun 1999 (Paillier, 1999).
- Keamanan algoritma ini didasarkan pada sulitnya memecahkan persoalan residu ke- n (*composite residuosity problem*). Persoalan residu kelas ke- n adalah sebagai berikut:

Diberikan bilangan komposit n dan bilangan bulat z . Bilangan z dikatakan residu ke- n modulo n^2 jika terdapat sebuah nilai y sedemikian sehingga $z \equiv y^n \pmod{n^2}$.

- Contoh: $n = 8$ dan $z = 33$. Maka, carilah nilai y sedemikian sehingga $33 \equiv y^8 \pmod{64}$. Nilai y yang memenuhi adalah $y = 5$, sebab $33 \equiv 5^8 \pmod{64}$. Persoalan ini akan semakin sukar bila n dan z bernilai besar.



Pascal Paillier

<https://www.researchgate.net/profile/Pascal-Paillier>

https://twitter.com/pascal_paillier

<https://raais.co/speakers-2020-pascal-paillier-zama>

Prosedur Membangkitkan Pasangan Kunci

- Misalkan Alice ingin membangkitkan pasangan kunci privat dan kunci publiknya. Prosedur yang dilakukan Alice adalah sebagai berikut:
 1. Alice memilih dua buah bilangan prima sembarang, p dan q yang memenuhi syarat $\text{PBB}(pq, (p-1)(q-1)) = 1$. PBB = pembagi bersama terbesar = *greatest common divisor* = gcd .
 2. Alice menghitung $n = p \cdot q$ dan $\lambda = \text{KPK}(p-1, q-1)$. KPK = kelipatan persekutuan terkecil atau $\text{lcm} = \text{lowest common multiple}$.
 3. Pilih sembarang bilangan bulat g , dengan $g < n$.
 4. Hitung $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$, dengan fungsi $L(x) = (x-1)/n$.
- Hasil dari prosedur di atas:
 - Kunci publik Alice adalah pasangan (g, n) .
 - Kunci privat Alice adalah (λ, μ) .

Contoh 2: Alice menghitung kunci privat dan kunci publiknya sebagai berikut:

a) Misalkan $p = 7$ dan $q = 11$. Keduanya memenuhi syarat $PBB(pq, (p - 1)(q - 1)) = PBB(77, 60) = 1$.

b) Hitung $n = p \cdot q = 77$ dan $\lambda = \text{KPK}(p - 1, q - 1) = \text{KPK}(6, 10) = 30$.

c) Pilih g secara acak, $g < n^2$, misalkan $g = 5652$

d) Hitung $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$, dengan fungsi $L(x) = (x - 1)/n$.

$$L(g^\lambda \bmod n^2) = L(5652^{30} \bmod 77^2) = L(5652^{30} \bmod 5929) = L(3928)$$

$$L(3928) = (3928 - 1)/77 = 3927/77 = 51$$

$$\mu = 51^{-1} \pmod{77} = 74$$

• Diperoleh kunci publik dan kunci privat Alice adalah sebagai berikut:

Kunci publik: $(g, n) = (5652, 77)$

Kunci privat: $(\lambda, \mu) = (30, 74)$

Prosedur Enkripsi

1. Misalkan m adalah pesan yang akan dienkripsi, dengan syarat $0 \leq m < n$.
2. Pilih bilangan bulat acak r dengan syarat $0 \leq r < n$ dan $\text{PBB}(r, n) = 1$.
3. Hitung cipherteks dari m dengan persamaan $c = g^m \cdot r^n \bmod n^2$. Dalam hal ini, c adalah residu ke- n dalam modulus n^2 , dilambangkan dengan $[c]_g$.

Prosedur Dekripsi

1. Misalkan c cipherteks yang akan didekripsi.
2. Hitung plainteks dari c dengan persamaan $m = L(c^\lambda \bmod n^2) \cdot \mu \bmod n$ atau dengan persamaan lain yaitu

$$m = \frac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n^2$$

Contoh 3: Misalkan pesan yang dienkripsi adalah $m = 42$. Pilih $r = 23$, karena $0 \leq r < n$ dan $\text{PBB}(r, n) = 1$.

a) Enkripsi: cipherteks dihitung menggunakan kunci publik $(g, n) = (5652, 77)$ dan dengan persamaan:

$$\begin{aligned}c &= g^m \cdot r^n \cdot \text{mod } n^2 \\ &= 5652^{42} \cdot 23^{77} \text{ mod } 77^2 \\ &= 5652^{42} \cdot 23^{77} \text{ mod } 5929 \\ &= 5652^{42} \text{ mod } 5929 \cdot 23^{77} \text{ mod } 5929 \\ &= 4019 \cdot 606 \text{ mod } 5929 \\ &= 4624\end{aligned}$$

- Jadi, cipherteks untuk pesan $m = 42$ adalah $c = 4624$.

b) Dekripsi: plainteks dihitung dengan kunci privat $(\lambda, \mu) = (30, 74)$ dan dengan persamaan:

$$\begin{aligned} m &= L(c^\lambda \bmod n^2) \cdot \mu \bmod n \\ &= L(4624^{30} \bmod 77^2) \cdot 74 \bmod 77 \\ &= L(4624^{30} \bmod 5929) \cdot 74 \bmod 77 \\ &= L(4624^{30} \bmod 5929) \cdot 74 \bmod 77 \\ &= L(4852) \cdot 74 \bmod 77 \\ &= (4852 - 1)/77 \cdot 74 \bmod 77 \\ &= 63 \cdot 74 \bmod 77 \\ &= 42 \end{aligned}$$

Sifat Homomorfik di dalam Algoritma Paillier

- Algoritma Paillier merupakan algoritma enkripsi homomorfik yang bersifat aditif, yaitu jika dua buah cipherteks dikalikan maka hasil dekripsinya sama dengan penjumlahan kedua plainteksnya.
- Misalkan dua buah plainteks adalah m_1 dan m_2 , hasil enkripsinya masing-masing adalah c_1 dan c_2 , yang dalam hal ini,

$$E(m_1, r_1) = c_1 = g^{m_1} \cdot r_1^{n^2} \text{ mod } n^2 \quad (1)$$

$$E(m_2, r_2) = c_2 = g^{m_2} \cdot r_2^{n^2} \text{ mod } n^2 \quad (2)$$

Kalikan cipherteks c_1 dan c_2 sebagai berikut:

$$\begin{aligned} E(m_1, r_1) \cdot E(m_2, r_2) &= c_1 \cdot c_2 = g^{m_1} \cdot r_1^{n^2} \cdot g^{m_2} \cdot r_2^{n^2} \text{ mod } n^2 \\ &= g^{m_1 + m_2} (r_1 \cdot r_2)^{n^2} \text{ mod } n^2 \end{aligned} \quad (3)$$

- Sekarang, akan ditunjukkan bahwa hasil dekripsi terhadap $c_1 \cdot c_2$ sama dengan penjumlahan kedua plainteksnya, yaitu

$$D(c_1 \cdot c_2) = m_1 + m_2 \pmod n \quad (4)$$

Bukti:

Tinjau $(1 + n) \in Z_n^*$. (Z_n^* = himpunan bilangan bulat tak-negatif = $\{1, 2, \dots, n^2\}$)

Perpangkatan $(1 + n)$ di dalam Z_n^* adalah sebagai berikut:

$$(1 + n)^2 = 1 + 2n + n^2 \equiv 1 + 2n \pmod{n^2}$$

$$(1 + n)^3 = 1 + 3n + 3n^2 + n^3 \equiv 1 + 3n \pmod{n^2}$$

$$(1 + n)^q = 1 + qn + [\text{suku-suku berderajat lebih tinggi}] \equiv 1 + qn \pmod{n^2} \quad (5)$$

Dekripsi $c_1 \cdot c_2$ adalah

$$D(c_1 \cdot c_2) = L((c_1 \cdot c_2)^\lambda \pmod{n^2}) \cdot \mu \pmod n$$

Berdasarkan persamaan (3),

$$(c_1 \cdot c_2)^\lambda \equiv g^{\lambda(m_1 + m_2)} \cdot (r_1 \cdot r_2)^{\lambda n} \pmod{n^2} \equiv g^{\lambda(m_1 + m_2)} \pmod{n^2} \quad (6)$$

menurut Teorema Carmichael (O'Keefe, 2008).

Kemudian, $g^\lambda \equiv (1 + n)^{\lambda[g]_{(1+n)}} \pmod{n^2}$. Substitusikan ini ke dalam (6), diperoleh

$$(c_1 \cdot c_2)^\lambda \equiv (1 + n)^{\lambda[g]_{(1+n)} \cdot (m_1 + m_2)} \pmod{n^2} \quad (7)$$

Menurut (5),

$$(1 + n)^{\lambda[g]_{(1+n)} \cdot (m_1 + m_2)} \equiv 1 + (m_1 + m_2) \cdot \lambda[g]_{(1+n)} \cdot n \pmod{n^2}$$

Jadi,

$$L((c_1 \cdot c_2)^\lambda \pmod{n^2}) \equiv (m_1 + m_2) \cdot \lambda[g]_{(1+n)} \cdot \pmod{n} \quad (8)$$

sedangkan μ adalah balikan dari $\lambda[g]_{(1+n)}$, atau $\mu = (\lambda[g]_{(1+n)})^{-1}$.

Maka, dengan mengalikan (8) dengan μ diperoleh

$$\begin{aligned} D(c_1 \cdot c_2) &= L((c_1 \cdot c_2)^\lambda \bmod n^2) \cdot \mu \bmod n \\ &= (m_1 + m_2) \cdot \lambda[g]_{(1+n)} \cdot (\lambda[g]_{(1+n)})^{-1} \bmod n \\ &= (m_1 + m_2) \bmod n \end{aligned}$$

Jadi, kita berhasil menunjukkan bahwa

$$D(c_1 \cdot c_2) = (m_1 + m_2) \bmod n.$$

Contoh 4: Dari Contoh 3 telah diperoleh hasil enkripsi $m_1 = 42$ dengan $r_1 = 23$ adalah cipherteks $c_1 = 4624$. Misalkan $m_2 = 29$ dan $r_2 = 30$ dan setelah dihitung cipherteksnya adalah $c_2 = 1539$.

Perkalian kedua cipherteks adalah

$$c_1 \cdot c_2 = 4624 \cdot 1539 = 7116336 \text{ mod } 5929 = 1536$$

Dekripsi $c_1 \cdot c_2$ adalah

$$\begin{aligned} D(c_1 \cdot c_2) &= L((c_1 \cdot c_2)^\lambda \text{ mod } n^2) \cdot \mu \text{ mod } n \\ &= L(1536^{30} \text{ mod } 77^2) \cdot 74 \text{ mod } 77 \\ &= L(1536^{30} \text{ mod } 5929) \cdot 74 \text{ mod } 77 \\ &= L(1536^{30} \text{ mod } 5929) \cdot 74 \text{ mod } 77 \\ &= L(155) \cdot 74 \text{ mod } 77 \\ &= (155 - 1)/77 \cdot 74 \text{ mod } 77 \\ &= 2 \cdot 74 \text{ mod } 77 \\ &= 148 \text{ mod } 77 \\ &= 71 \end{aligned}$$

Hasil dekripsi di atas sama dengan penjumlahan dua buah plainteks dalam modulus 77, yaitu

$$(m_1 + m_2) \text{ mod } n = (42 + 29) \text{ mod } 77 = 71 \text{ mod } 77 = 71$$

Jadi, $D(c_1 \cdot c_2) = (m_1 + m_2) \text{ mod } n = 71$.

Penggunaan Enkripsi Homomorfik Algoritma Paillier di dalam *E-voting*

- Misalkan terdapat 3 orang kandidat: X, Y, dan Z, dan tiga orang pemilih: V_1, V_2, V_3 .

	X	Y	Z
V_1	1	0	0
V_2	0	1	0
V_3	1	0	0
Total	2	1	0

Plainteks

	X	Y	Z
V_1	C_{1X}	C_{1Y}	C_{1Z}
V_2	C_{2X}	C_{2Y}	C_{2Z}
V_3	C_{3X}	C_{3Y}	C_{3Z}
Total	C_X	C_Y	C_Z

Cipherteks

Gambar 18.1 Tabel pemungutan suara di dalam *e-voting* (Rivest, 2002)

- C_{1X}, C_{1Y} , dan lain-lain menyatakan suara yang diberikan oleh pemilih. Suara ini dienkripsi dengan menggunakan kunci publik panitia pemilihan. Kunci privat hanya diketahui oleh panitia pemilihan.

- Dengan algoritma Paillier, 0 yang sama tidak selalu dienkripsi menjadi cipherteks yang sama (tergantung nilai r yang digunakan, sebab r selalu berbeda setiap mengenkripsi plainteks).
- Dengan demikian, pihak lawan tidak dapat menyimpulkan apakah dua orang pemilih memberikan suara yang sama dengan melihat apakah cipherteksnya sama. Dengan demikian kerahasiaan pemilih terjamin.
- Untuk alasan ini, maka RSA tidak cocok untuk aplikasi *e-voting* (Rivest, 2012), sebab plainteks yang sama selalu dienkripsi menjadi cipherteks yang sama (ingat $c = m^e \bmod n$, karena kunci publik selalu sama untuk mengenkripsi setiap suara, maka angka 0 dan 1 selalu dienkripsi menjadi cipherteks yang sama pula).

- Untuk menghitung total suara setiap kandidat, maka semua cipherteks pada setiap kolom dikalikan:

$$C_X = C_{1X} \cdot C_{2X} \cdot C_{3X}$$

$$C_Y = C_{1Y} \cdot C_{2Y} \cdot C_{3Y}$$

$$C_Z = C_{1Z} \cdot C_{2Z} \cdot C_{3Z}$$

	X	Y	Z
V_1	C_{1X}	C_{1Y}	C_{1Z}
V_2	C_{2X}	C_{2Y}	C_{2Z}
V_3	C_{3X}	C_{3Y}	C_{3Z}
Total	C_X	C_Y	C_Z

- Menurut sifat homomorfik pada algoritma Paillier, perkalian cipherteks sama dengan menjumlahkan plainteksnya:

$$C_X = M_{1X} + M_{2X} + M_{3X}$$

$$C_Y = M_{1Y} + M_{2Y} + M_{3Y}$$

$$C_Z = M_{1Z} + M_{2Z} + M_{3Z}$$

	X	Y	Z
V_1	1	0	0
V_2	0	1	0
V_3	1	0	0
Total	2	1	0

M_{ij} menyatakan plainteks pada baris i dan kolom j .

- Dengan melakukan komputasi hanya pada cipherteks maka suara yang diberikan oleh pemilih dapat terjamin kerahasiaannya.
- Begitu pula total suara sementara seorang kandidat juga tetap rahasia sehingga pemilih lain maupun panitia pemilihan tidak mengetahui hasil sementara ini.

Enkripsi Homomorfik Penuh

- Enkripsi homomorfik penuh (*fully homomorphic encryption* atau FHE) artinya enkripsi yang memiliki sifat aditif dan multiplikatif sekaligus.
- Hasil penjumlahan atau perkalian dua buah cipherteks bila didekripsi hasilnya sama dengan penjumlahan atau perkalian plainteksnya.
- Algoritma kriptografi kunci-publik yang telah kita bicarakan sejauh ini (RSA, ElGamal, dan Paillier) tidak memenuhi kriteria enkripsi homomorfik penuh.

- Selama 30 tahun setelah Rivest, Adleman, dan Dertouzos menyajikan tantangan konstruksi skema FHE, namun tantangan ini tetap tidak terpecahkan (Wu, 2015).
- Tahun 2005 Boneh, Goh, dan Nissim mengembangkan skema FHE, namun skema tersebut hanya mendukung satu operasi perkalian saja dan banyak operasi penjumlahan cipherteks.
- Barulah pada tahun 2009 konstruksi skema FHE yang mendukung banyak operasi perkalian dan penjumlahan dikembangkan oleh Gentry (2009) dengan menggunakan kriptografi berbasis teori *lattice* (di luar pembahasan kuliah ini)

- Di dalam skema FHE yang diusulkan oleh Gentry diimplementasikan sebagai sirkuit Boolean.
- Operasi perkalian dan penjumlahan dinyatakan sebagai operasi AND dan XOR dalam string biner. Sirkuit Boolean adalah rangkaian logika yang disusun oleh gerbang-gerbang logika (AND, OR, NOT, XOT, dan lain-lain). Sembarang sirkuit Boolean dapat ditulis dengan XOR dan AND saja.
- Skema FHE ini menjadi universal karena sekali kita dapat mengevaluasi secara homomorfik XOR dan AND, kita dapat mengevaluasi sembarang sirkuit Boolean untuk fungsi apapun (Coron, 2015).