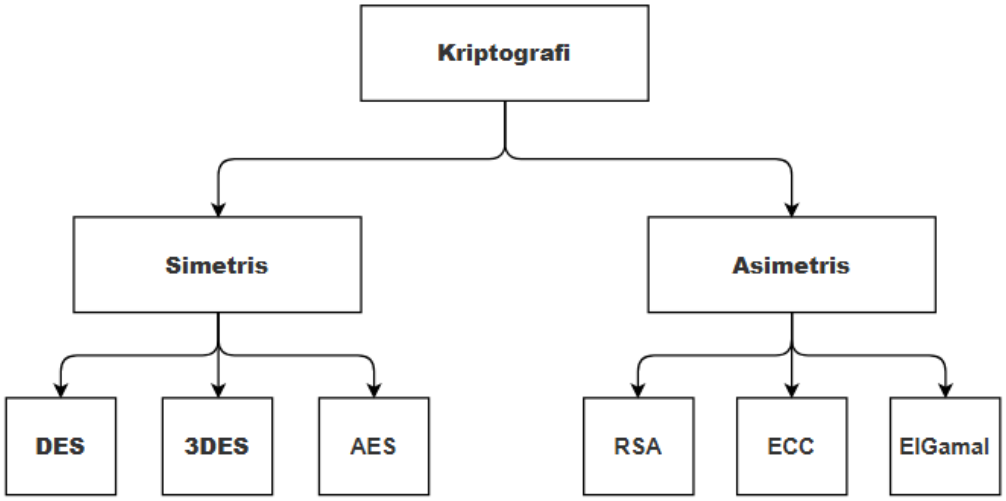
BAB II

LANDASAN TEORI

2.1 Kriptografi

Di dunia modern, keamanan merupakan istilah paling berharga dalam bidang sistem komunikasi. Keamanan berkaitan dengan berbagai teknologi dan metode, di mana salah satu teknologi paling aman adalah Kriptografi, di mana teks biasa diubah menjadi teks terenkripsi untuk mentransfer data kepada pengguna yang sah. Algoritma kriptografi dapat dibagi menjadi dua jenis berdasarkan jumlah kunci, yaitu Simetris dan Asimetris, di mana algoritma simetris menggunakan satu kunci tunggal dan algoritma asimetris menggunakan dua kunci yang berbeda (Anwar et al., 2019).



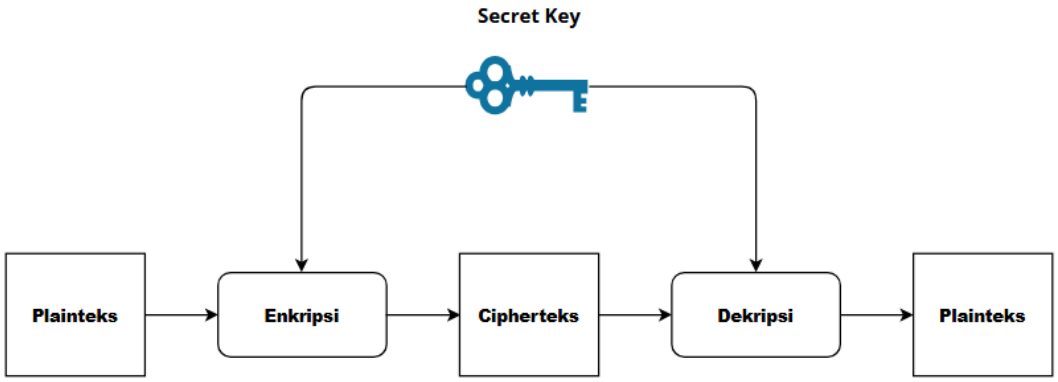
Gambar 2.1 Taxonomy Kriptografi (Maqsood et al., 2017)

Kriptografi muncul sebagai fondasi utama keamanan komputer di era digital, menyediakan keamanan esensial untuk data dalam dunia yang sangat bergantung pada sistem yang saling terhubung. Kriptografi merujuk pada seni dan ilmu membuat data tidak dapat dipahami oleh pihak yang tidak berwenang, namun pihak

yang berwenang dapat mengakses dan menggunakan informasi tersebut. Mereka yang asli menggunakan sandi (Käppler et al, 2022) untuk melindungi pesan penting. Namun, kriptografi modern telah menjadi sangat kompleks, penting untuk keamanan komunikasi, perlindungan data sensitif, dan autentikasi pengguna dan perangkat melalui internet dan jaringan digital lainnya. Pertumbuhan pesat internet, e-commerce, komputasi awan, dan Internet of Things (IoT), jumlah besar data pribadi, keuangan, dan pemerintah terus-menerus dikirimkan melalui jaringan global (Jagtap, 2023). Kriptografi memastikan informasi sensitif tetap rahasia dengan mencegah akses melalui teknik enkripsi. Selain itu, kriptografi menjaga integritas data dengan memastikan informasi tidak diubah selama transmisi atau penyimpanan, yang krusial untuk mengandalkan sistem digital. Selain itu, kriptografi berperan penting dalam pemantauan otentikasi dan penyangkalan, memungkinkan pengguna untuk memverifikasi keaslian kontak mereka dan tidak menyangkal keaslian tindakan mereka (Vaishnavi, 2021).

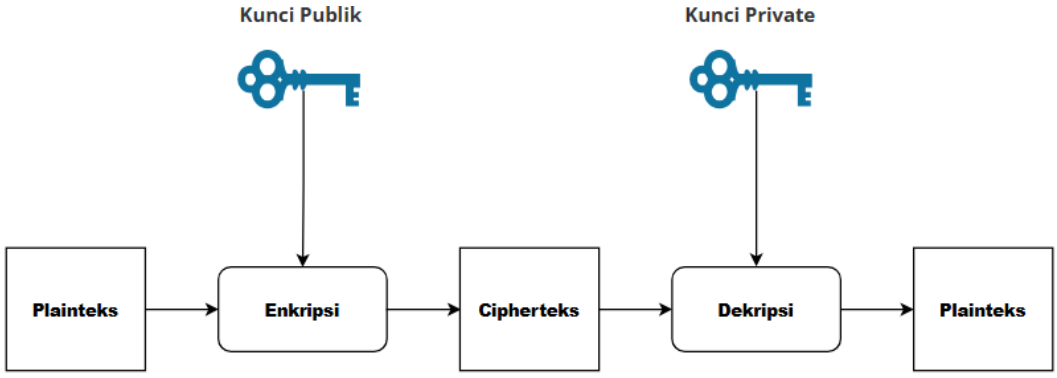
2.2 Kriptografi Simetris dan Asimetris

Algoritma kriptografi dapat dibagi menjadi dua jenis berdasarkan jumlah kunci, yaitu Simetris dan Asimetris, di mana algoritma simetris menggunakan satu kunci tunggal dan algoritma asimetris menggunakan dua kunci yang berbeda (Anwar et al., 2019). Dalam algoritma enkripsi simetris, pengirim data memproses teks asli (data asli) dan kunci enkripsi bersama dengan enkripsi khusus dan mengirimkannya. Hal ini memerlukan agar pihak yang mendekripsi harus mengetahui kunci rahasia tersebut sebelumnya. (Zhang, 2021).



Gambar 2.2 Kriptografi Simetris

Kriptografi kunci simetris menggunakan satu kunci rahasia yang sama untuk mengenkripsi dan mendekripsi pesan. Metode ini tetap menjadi tulang punggung enkripsi data berkat kecepatan dan efisiensinya, terutama untuk data dalam jumlah besar. Algoritma simetris secara umum dibagi menjadi dua kategori: algoritma blok (yang beroperasi pada blok bit berukuran tetap menggunakan putaran substitusi dan permutasi) dan algoritma aliran (yang menghasilkan aliran kunci untuk mengenkripsi data bit demi bit atau byte demi byte). Untuk mengenkripsi pesan dengan panjang sembarang secara aman, enkripsi blok digunakan bersama mode operasi yang menentukan cara blok dihubungkan, sementara enkripsi aliran secara inheren menghasilkan aliran kunci yang terus menerus (Shah et al., 2025).



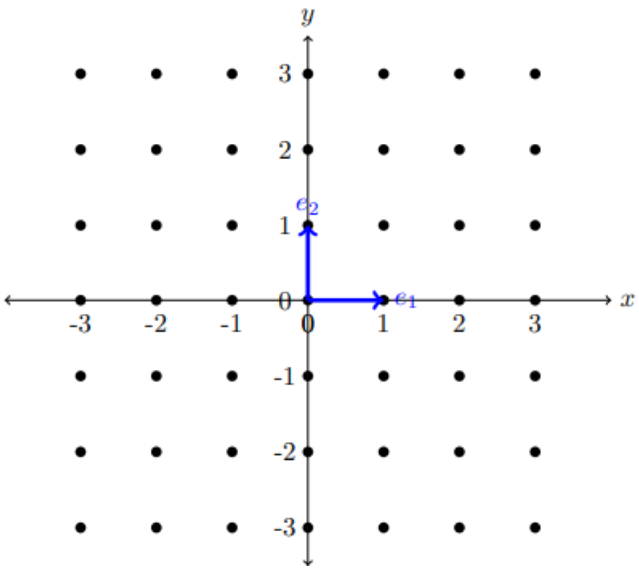
Gambar 2.3 Kriptografi Asimetris

Kriptografi kunci publik atau kriptografi asimetris adalah sistem kriptografi yang menggunakan pasangan kunci: kunci publik, yang dapat diketahui oleh orang lain, dan kunci pribadi, yang hanya pemiliknya yang boleh mengetahuinya. Sistem ini dirancang untuk mengatasi masalah yang tidak dapat diatasi oleh kriptografi simetris, seperti distribusi kunci, pengelolaan kunci, dan masalah tanda tangan digital. Secara umum, jenis enkripsi asimetris bergantung pada masalah komputasi yang sulit, dan pembangkitan kunci bergantung pada hal tersebut. Kriptografi asimetris telah diterapkan dalam komunikasi dan masalah perlindungan privasi karena menjamin privasi dan keamanan dengan otentikasi dan tanda tangan digital. Ini dapat dianggap sebagai teknik keamanan paling umum untuk mencegah kebocoran informasi pribadi pengguna dan memperkuat kehidupan sehari-hari (Shen et al., 2021).

2.3 Lattice Based Cryptography

Lattice Based Cryptography muncul sebagai kandidat utama untuk keamanan pasca-kuantum, didasarkan pada kesulitan matematis dari masalah kisi tertentu seperti *Learning With Error* (LWE) dan Masalah *Ring-LWE*. Keamanan dasar sistem-sistem ini bergantung pada kesulitan komputasi dalam menemukan vektor terpendek dalam kisi berdimensi tinggi, masalah yang tetap menantang bahkan bagi komputer kuantum (Peikert, 2016). Perkembangan terbaru telah memperkuat landasan teoretis kriptografi berbasis kisi, terutama dalam memahami perkiraan keamanan konkret dan asumsi kesulitan yang mendasari sistem-sistem ini (Adapa, 2025). Dua kelas utama algoritma berbasis kisi yang digunakan dalam kriptografi adalah NTRU dan LWE. Keamanan NTRU didasarkan pada kerumitan yang tidak

dapat dibuktikan dapat dikurangi menjadi pemecahan *Closest Vector Problem* (CVP) dalam kisi, sedangkan keamanan LWE bergantung pada pemecahan *Short Vector Problem* (SVP) yang dapat dibuktikan dapat dikurangi dalam kisi. Seperti yang dijelaskan (Meyer, 2025) sebuah kisi adalah subgrup diskrit dari R^n yang secara geometris tampak seperti himpunan titik periodik, seperti pada Gambar 2.5 dibawah ini.



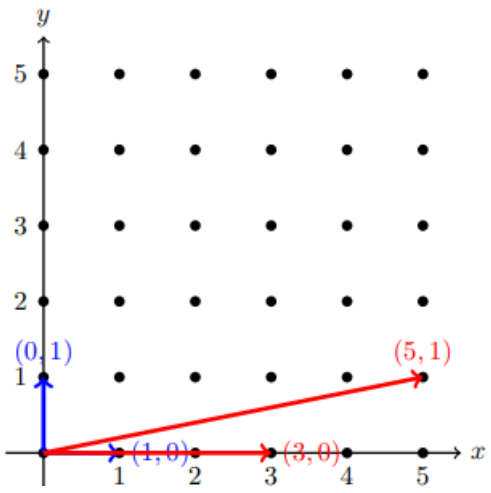
Gambar 2.4 Contoh Kisi Benar

Setiap kisi dihasilkan oleh suatu basis. Untuk himpunan vektor yang linier independent $B = \{b_0, \dots, b_{n-1}\}$, kisi yang dihasilkan B Adalah

$$\Lambda(B) = \{x_i \in \mathbb{Z} \mid \sum_{i=0}^{n-1} x_i \, . \, b_i\}$$

Dengan kata lain, kisi dihasilkan dengan mengambil kombinasi linier bulat dari vektor-vektor basis. Misalnya, kisi pada Gambar 2.5 dapat dihasilkan oleh basis B = {e1, e2} = {(1, 0),(0, 1)}. Basis dari sebuah kisi tidak unik, dan sebenarnya,

sebuah kisi memiliki jumlah basis tak terbatas. Perlu dicatat bahwa basis yang "baik" untuk sebuah kisi terdiri dari vektor-vektor pendek yang hampir ortogonal, sedangkan basis yang "buruk" terdiri dari vektor-vektor panjang dengan sudut kecil di antara mereka. Seperti yang ditunjukkan di bawah ini pada Gambar 2.6, basis B1 = {(1, 0),(0, 1)} adalah basis yang baik untuk kisi, sedangkan basis B2 = {(3, 0),(5, 1)} adalah basis yang buruk.



Gambar 2.5 Contoh Kisi Salah

Akibatnya, NTRU mengalami kekurangan jaminan keamanan, tetapi dalam praktiknya memberikan fleksibilitas dan efisiensi yang lebih besar dalam implementasi. Sebaliknya, masalah LWE tahan terhadap serangan kuantum, namun sifatnya yang relatif tidak efisien mendorong peneliti untuk merancang formulasi yang lebih efisien (Nejatollahi et al., 2019).

2.4 Number Theoritic Transform (NTT)

Number Theoritic Transform (NTT) sering digunakan dalam skema kriptografi yang didasarkan pada kesulitan masalah Ring Learning With Errors (R-LWE) untuk melaksanakan perkalian polinomial modular secara efisien (Longa et

al., 2016). Dalam pengembangan Post Quantum Cryptography (PQC), NTT menjadi alat matematika yang kuat. Menghitung perkalian polinomial secara efektif menjadikannya komponen penting dalam kriptografi. Dalam kriptografi berbasis kisi, algoritma NTT sangat berguna, yang bergantung pada tingkat kesulitan masalah matematika tertentu untuk memastikan keamanan. Bidang ini semakin penting seiring dengan kemajuan teknologi komputasi kuantum dan metode enkripsi tradisional (Satriawan et al., 2023).

Contoh perhitungan NTT dari penelitian (Satriawan et al., 2024) seperti berikut, misalkan $G(x) = 1 + 2x + 3x^2 + 4x^3$ atau dalam notasi vektor $\boldsymbol{g} = [1,2,3,4]$. Dapat disimpulkan bahwa n = 4 dan kita bekerja para ring Z_{7681} dan ω adalah akar kesatuan primitif n. Bisa dihitung seperti:

$$\hat{g} \begin{bmatrix} \omega^{0x0} & \omega^{0x1} & \omega^{0x2} & \omega^{0x3} \\ \omega^{1x0} & \omega^{1x1} & \omega^{1x2} & \omega^{1x3} \\ \omega^{2x0} & \omega^{2x1} & \omega^{2x2} & \omega^{2x3} \\ \omega^{3x0} & \omega^{3x1} & \omega^{3x2} & \omega^{3x3} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Perhatikan bahwa pangkat ω adalah hasil dari perkalian antara nomor baris dan kolom. Karena ω adalah akar ke-n dari kesatuan, $\omega^k = \omega^{(k \bmod n)}$ untuk $k > n$.

$$\hat{g} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 \\ \omega^0 & \omega^2 & \omega^4 & \omega^6 \\ \omega^0 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{g} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 \\ \omega^0 & \omega^2 & \omega^0 & \omega^2 \\ \omega^0 & \omega^3 & \omega^2 & \omega^1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

Dari contoh, diperoleh salah satu akar ke-n dari kesatuan Z_{7681} adalah $\omega = 3383$. Apabila dimasukan kedalam matriks akan seperti:

$$\hat{g} = \begin{bmatrix} 3383^0 & 3383^0 & 3383^0 & 3383^0 \\ 3383^0 & 3383^1 & 3383^2 & 3383^3 \\ 3383^0 & 3383^2 & 3383^0 & 3383^2 \\ 3383^0 & 3383^3 & 3383^2 & 3383^1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{g} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 3383 & 7680 & 4298 \\ 1 & 7680 & 1 & 7680 \\ 1 & 4298 & 7680 & 3384 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\hat{g} = \begin{bmatrix} 10 \\ 913 \\ 7679 \\ 6764 \end{bmatrix}$$

Oleh karena itu, $NTT(g) = [10{,}912{,}7679{,}6764]$ dalam Z_{7681}.

2.5 Karatsuba

Algoritma Karatsuba adalah algoritma perkalian bilangan bulat yang cepat, ditemukan oleh Anatoly Karatsuba pada tahun 1960, dan dipublikasikan pada tahun 1962 (Delibasic et al., 2020). Ini adalah algoritma *divide-and-conquer* yang mengubah perkalian dua bilangan berdigit n menjadi tiga perkalian bilangan berdigit n/2, dan dengan mengulangi pengurangan ini, menjadi paling banyak. Contoh perhitungan dari algortima ini sebagai berikut,

Untuk menghitung hasil perkalian 12345 dan 6789, di mana B = 10, pilih m = 3. Digunakan pergeseran kanan m kali untuk memecah operand masukan menggunakan basis hasil (B^m = 1000),

$$1234 = 12 \,.\, 1000 + 345$$

$$6789 = 6 \,.\, 1000 + 789$$

Hanya tiga perkalian, yang dilakukan pada bilangan bulat yang lebih kecil, digunakan untuk menghitung tiga hasil parsial:

$$z_2 = 12x6 = 72$$

$$z_0 = 345x789 = 272205$$

$$z_1 = (12x345)x(6+789) - z_2 - z_0 = 357x795 - 72 - 272205$$

$$= 283815 - 72 - 272205 = 11538$$

Maka didapatkan hasilnya dengan hanya menjumlahkan ketiga hasil parsial ini, yang telah disesuaikan posisinya (dan kemudian memperhitungkan sisa pembagian dengan mendekomposisi ketiga masukan ini ke dalam basis 1000, sama seperti pada operand masukan):

$$= z_2 \,.\, (B^m)^2 + z_1 \,.\, (B^m)^1 + z_0 \,.\, (B^m)^0$$

$$= 72 \,.\, 1000^2 + 11538 \,.\, 1000 + 272205 = 83810205$$

2.6 Kundu's Hybridized Karatsuba-NTT Algorithm

Pada penelitian (Kundu et al., 2022), memberikan contoh praktis yang menunjukan cara perhitungan rumus yang salah (1) dan rumus yang benar (2). Untuk melakukan nya, dipilih polinomial $f, g \in \mathbf{R}_{17} = \mathbf{Z}_{17}[x] < x^8 + 1 >$:

$$f = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7$$

$$g = 1 + x^3 + x^6 + x^7$$

dengan n = 8, q = 17, akar primitif ω = 2 mod 17, dan invers ω^{-1} = 9 mod 17. Lalu pisahkan f dan g menjadi 2 bagian (2-part karatsuba),

$$f = (1 + x + x^2 + x^3) + x^4(1 + x + x^2 + x^3)$$

$$g = (1 + 0 \,.\, x + 0 \,.\, x^2 + 1 \,.\, x^3) + x^4(0 + 0 \,.\, x + 1 \,.\, x^2 + 1 \,.\, x^3)$$

Ubah kedalam vektor,

$$f_0 = 1 + x + x^2 + x^3 \Longrightarrow F_0 = [1,1,1,1,0,0,0,0]$$

$$f_1 = 1 + x + x^2 + x^3 \Longrightarrow F_1 = [1,1,1,1,0,0,0,0]$$

$$g_0 = 1 + x^3 \Longrightarrow G_0 = [1,0,0,1,0,0,0,0]$$

$g_1 = x^2 + x^3 \Longrightarrow G_1 = [0,0,1,1,0,0,0,0]$

Dari definisi (2), memiliki $\widehat{F_0} = NTT(F_0) = \sum_{j=0}^{7} a_{\omega^{ij}} \pmod{17}$, $\forall i = 0, \ldots, 7$. Bagian ini akan menjelaskan secara eksplisit bagaimana NTT(F_0) dihitung, yang akan membantu untuk memahami perhitungan NTT untuk vektor lainnya. Pertama kita memiliki,

$(\widehat{F_0})_0 = a_0\omega^{0.0} += a_0\omega^{0.1} + \cdots += a_7\omega^{0.7} \pmod{17} = 4$ dan apabila di analogikan,

$(\widehat{F_0})_1 = 15 \pmod{17}$ $(\widehat{F_0})_2 = 0 \pmod{17}$

$(\widehat{F_0})_3 = 7 \pmod{17}$ $(\widehat{F_0})_4 = 7 \pmod{17}$

$(\widehat{F_0})_5 = 12 \pmod{17}$ $(\widehat{F_0})_6 = 0 \pmod{17}$

$(\widehat{F_0})_7 = 4 \pmod{17}$

Oleh karena itu kita memiliki,

$\widehat{F_0} = \widehat{F_1} = [4,15,0,7,0,12,4]$

Lalu, serupa dengan itu, kita menghitung,

$\widehat{G_0} = NTT(G_0) = [2,9,14,3,0,10,5,16]$

$\widehat{G_1} = NTT(G_1) = [2,12,12,15,0,13,3,11]$

Selanjutnya, hitung beberapa komponen menggunakan notasi dalam definisi 1 dan berguna untuk rumus (1) dan (2):

1. $\widehat{F_0} \circ_q \widehat{G_0} = [8,16,0,4,0,1,0,13]$
2. $\widehat{F_1} \circ_q \widehat{G_1} = [8,10,0,3,0,3,0,10]$
3. $\widehat{F_0} +_q \widehat{F_1} = [8,13,0,14,0,7,0,8]$
4. $\widehat{G_0} +_q \widehat{G_1} = [4,4,9,1,0,6,8,10]$

5. $(\widehat{F_0} +_q \widehat{F_1}) \circ_q (\widehat{G_0} +_q \widehat{G_1}) = [15,1,0,14,0,8,0,12]$
6. x^4 dapat dianggap sebagai vektor [0,0,0,0,1,0,0,0], sehingga NTT ([0,0,0,0,1,0,0,0]) = [1,16,1,16,1,16,1,16] (menjadi n = 8)
7. $\widehat{F_0} \circ_q \widehat{G_0} - \widehat{F_1} \circ_q \widehat{G_1} = [0,6,0,1,0,15,0,3]$
8. $(\widehat{F_0} +_q \widehat{F_1}) \circ_q (\widehat{G_0} +_q \widehat{G_1}) - \widehat{F_0} \circ_q \widehat{G_0} - \widehat{F_1} \circ_q \widehat{G_1} = [16,9,0,7,0,4,0,6]$

Sekarang bisa diperiksa dengan mudah bahwa rumus (1) memberikan,

$$h = NTT^{-1}([0,6,0,1,0,15,0,3] + [16,8,0,10,0,13,0,11])$$

$$= NTT^{-1}([16,14,0,11,0,11,0,14])$$

Dari definisi 3, kita memiliki $h = NTT^{-1}(H) = n^{-1} \sum_{j=0}^{7} a_j \omega^{-ij}$ (mod 17), $\forall i = 0, \ldots, 7$. Terutama,

$$h_0 = 15[H_0\omega^{-0.0} + H_1\omega^{-0.1} + \cdots + H_7\omega^{-0.7}] = 4 \text{ (mod 17)}$$

Dan, dianalogikan,

$h_1 = 4$ (mod 17) $\quad$ $h_2 = 2$ (mod 17)

$h_3 = 0$ (mod 17) $\quad$ $h_4 = 0$ (mod 17)

$h_5 = 0$ (mod 17) $\quad$ $h_6 = 2$ (mod 17)

$h_7 = 4$ (mod 17)

Diperoleh,

$$h = [4,4,2,0,0,0,2,4] \rightarrow 4 + 4^x + 2x^2 + 2x^6 + 4x^7$$

Formula (2) memberikan,

$$h = NTT^{-1}([0,6,0,1,0,15,0,3]) + x^4.NTT^{-1}([16,9,0,7,0,4,0,6])$$

$$= [1,1,0,0,16,16,0,0] + x^4.[1,1,2,4,3,3,2,0]$$

$$= [15,15,15,0,0,0,2,4]$$

$$\rightarrow 15 + 15x + 15x^2 + 2x^6 + 4x^7$$

Hasil yang terakhir adalah hasil yang benar untuk perkalian polinomial.

2.7 Nth Degree Truncated Polynomial Ring Units (NTRU)

NTRU adalah sistem kriptografi asimetris berbasis kisi yang menjadi finalis dalam proyek Standarisasi Kriptografi Pasca-Kuantum NIST. Keamanannya berdasarkan *Short Vector Problem*, yang menjadi sulit secara komputasi pada dimensi yang lebih tinggi. Akibatnya, NTRU tahan terhadap serangan dari komputer klasik dan kuantum, yang menjadikannya kandidat yang sangat baik untuk kriptografi pasca-kuantum (Meyer, 2025). Algoritma ini lebih cepat dan lebih ringkas dibandingkan dengan RSA yang digunakan secara luas. Terdapat beberapa redundansi dalam algoritma ini, yang dapat dihilangkan dengan mengorbankan ketepatan yang sempurna. Penelitian terhadap parameterisasi yang tepat dari NTRU telah berlangsung sejak tahun 1990-an, oleh karena itu menjadi salah satu kriptosistem yang terdokumentasi dengan lebih baik dan terpercaya di Round 3 (Bavdekar et al., 2023; Moody et al., 2020b).

Sistem enkripsi NTRU diimplementasikan dengan menggunakan polynomial dari ring $R = Z_q [x]/(x^n - 1)$, di mana q adalah pangkat dua. Dua polinomial f dan g dibangkitkan dengan koefisien dalam himpunan {-1,0,1}, dan $h = g \cdot f^{-1}$ dalam R. Kunci publik adalah h, sedangkan polinomial f dan g adalah kunci privat. Untuk mengenkripsi sebuah pesan acak yang seragam m diwakili oleh sebuah polinomial dalam R dengan koefisien {-1,0,1}, pengirim menghitung $c = 3hr + m$, di mana $r \in R$ adalah sebuah polinomial dengan koefisien yang dipilih secara acak dari himpunan {-1,0,1}. Untuk mendekripsi, pemegang kunci privat menghitung $e = c f \bmod q$ dan kemudian memulihkan pesan m dari $e \cdot f^{-1} \bmod 3$ (Alagic et al., 2022b).

2.8 State Of The Art

Berdasarkan rumusan masalah dan tujuan penelitian yang telah dibuat, maka dilakukan penyusunan *literature review* dari penelitian sebelumnya yang berkaitan dengan *quantum cryptography*. Beberapa penelitian sebelumnya yang dilakukan adalah, sebagai beri

Tabel 2.1 *State Of The Art*

No	Judul Penelitian	Metode	Kelebihan	Kekurangan	Gap Penelitian
1	*Preprocess-then-NTT Technique and Its Applications to Kyber and Newhope* (Zhou et al., 2018)	*Preprocess-then-NTT* (pNTT)	Menguran gi syarat modulus q menjadi tidak lagi harus memenuhi $q \equiv 1 \pmod{2n}$ dan cukup $q \equiv 1 \pmod{n}$ pada *1-round*	Kompleksitas waktu masih tinggi, pada 1PtNTT memiliki kompleksitas sekitar 7/6 kali dan 2PtNTT sekitar 5/4 kali dari NTT biasa.	Hanya menurunkan nilai modulus q tanpa memperbaiki struktur perhitungan perkalian polinomial sehingga masih memerlukan pendekatan yang bisa

			PtNTT atau $q \equiv 1$ (mod n/2) pada *2-round PtNTT.*		meningkatkan kompleksitas waktu.
No	**Judul Penelitian**	**Metode**	**Kelebihan**	**Kekurangan**	**Gap Penelitian**
2	*When NTT Meets Karatsuba: Preprocess-then-NTT Technique Revisited* (Zhu et al., 2019)	*Number Theoritic Transform* (NTT) + Karatsuba	Modulus q berkurang lebih jauh menjadi $n/2^{\alpha-1}\|(q-1)$, serta waktu satu kali komputasi lebih hemat daripada NTT biasa.	Overhead konstanta tetap lebih tinggi dari NTT biasa meski secara asimtotik sama.	Tidak diintegrasikan pada algoritma kriptografi sehingga belum dapat diketahui dampak dari nilai parameternya.
3	*Public Key Compression and Fast Polynomial Multiplication for NTRU using the Corrected Hybridized NTT-Karatsuba Method* (Kundu et al., 2022)	*Number Theoritic Transform* (NTT) + Karatsuba	Mengimplementasikan metode perhitugan perkalian polinomial terhadap algoritma kriptografi, serta mampu menurunkan nilai modulus q secara signifikan.	Hanya sebatas adaptasi perhitungan matematis.	Belum dilakukan evaluasi secara empiris.

Tabel 2.1 menunjukkan penelitian terdahulu yang berkaitan dengan penelitian ini. Penelitian (Zhou et al., 2018) menggunakan pendekatan *Preprocess-then-NTT* (PtNTT) berhasil mengurangi syarat modulus q menjadi tidak lagi harus memenuhi

$q \equiv 1 \pmod{2n}$ dan cukup $q \equiv 1 \pmod{n}$ pada *1-round PtNTT* atau $q \equiv 1 \pmod{n/2}$ pada *2-round PtNTT.* Akan tetapi, kompleksitas waktu masih tinggi, pada 1PtNTT memiliki kompleksitas sekitar 7/6 kali dan 2PtNTT sekitar 5/4 kali dari NTT biasa. Penelitian (Zhu et al., 2019) melalui pendekatan hibrid NTT-Karatsuba berhasil menjadikan modulus q berkurang lebih jauh menjadi $n/2^{\alpha-1}|(q-1)$, serta waktu satu kali komputasi lebih hemat daripada NTT biasa. Tapi, dalam prosesnya terjadi overhead konstanta tetap lebih tinggi dari NTT biasa meski secara asimtotik sama. Hal tersebut memengaruhi terhadap kompleksitas waktu komputasi yang awalnya $O(3n\log n)$ membesar menjadi $O(5n\log n)$. Selanjutnya, (Kundu et al., 2022) melakukan perbaikan terhadap rumus perhitungan NTT-Karatsuba yang dibuat oleh (Zhou et al., 2018) dan mengadaptasikan rumus matematisnya terhadap algoritma NTRU. Selain melakukan perbaikan rumus matematis, penuruan nilai modulus q turut dilakukan dan berhasil menurunkan nilai parameter n=1024, nilai q berhasil diturunkan dari 1.061.093.377 menjadi 83969, dan untuk n=2048 diturunkan dari $2^{57} + 25 \cdot 2^{13} + 1$ menjadi 166657. Sehingga kesimpulan dari peneliti terdahulu secara keseluruhan berhasil menurunkan nilai modulus q melalui pendekatan nya masing-masing, dan terdapat beberapa perbedaan seperti (Zhou et al., 2018) yang sudah mengimlementasikan rumus perhitungannya terhadap algoritma Kyber dan Newhope untuk melakukan evaluasi dari dampak penurunan nilai modulus q, sedangkan (Zhu et al., 2019) hanya mengevaluasi hasil dari rumus perkalian polinomial untuk melihat dampak pada waktu satu kali komputasinya saja, dan (Kundu et al., 2022) hanya mengoreksi pendekatan yang dibuat oleh (Zhu et al., 2019).

2.9 Matriks Penelitian

Untuk memperjelas posisi penelitian ini terhadap penelitian-penelitian sebelumnya, disusun matriks penelitian yang membandingkan fokus, pendekatan, dan keluaran dari masing-masing penelitian. Matriks ini berfungsi untuk mengidentifikasi kesenjangan (*research gap*) yang masih ada, sehingga dapat terlihat kontribusi baru yang diberikan oleh penelitian ini.

Tabel 2.2 Matriks Penelitian

No	Penelitian	Metode	Integrasi ke Algoritma PQC	Menghitung Waktu Operasi Polinomial	Menghitung Waktu Keygen, Enkripsi dan Dekripsi	Menghitung Besaran Kunci dan Ciphertext	Menghitung Keperluan Bandwith	Menganalisis Keamanan Latiice	Menganalisis Kegagalan Dekripsi
1	(Zhou et al., 2018)	Preprocess-then-NTT (pNTT)	✓	✓	✓	✓	-	-	-
2	(Zhu et al., 2019)	NTT, IpNTT, PtNTT, Karatsuba	-	✓	-	-	-	-	-

No	Penelitian	Metode	Integrasi ke Algoritma PQC	Menghitung Waktu Operasi Polinomial	Menghitung Waktu Keygen, Enkripsi dan Dekripsi	Menghitung Besaran Kunci dan Ciphertext	Menghitung Keperluan Bandwith	Menganalisis Keamanan Lattice	Menganalisis Kegagalan Dekripsi
3	(Kundu et al., 2022)	NTT-Karatsuba (Tanpa Implementasi)	✓	-	-	-	-	-	-
4	Penelitian ini	NTT-Karatsuba (Implementasi)	✓	✓	✓	✓	✓	✓	✓

Berdasarkan Tabel 2.3, terlihat bahwa penelitian terdahulu masih memiliki keterbatasan dalam cakupan pengujian. (Zhou et al., 2018) hanya menitikberatkan pada penerapan teknik Preprocess-then-NTT (pNTT) untuk mempercepat perkalian polinomial tanpa membahas aspek keamanan parameter yang digunakan. (Zhu et al., 2019) kemudian memperkenalkan pendekatan Number Theoretic Transform (NTT), Improved preprocess then-NTT (IPtNTT), dan Preprocess then-NTT (PtNTT) yang terbukti meningkatkan efisiensi komputasi, namun juga terbatas hanya pada pengukuran waktu operasi perkalian polinomial dan tidak mengintegrasikan nya kedalam algoritma PQC. Selanjutnya, Kundu et al. (2022) mengusulkan kompresi parameter untuk mempercepat operasi NTT–Karatsuba, tetapi

penelitian tersebut hanya menyajikan analisis teoritis tanpa implementasi nyata dan tidak membahas performa key generation, enkripsi, dekripsi, maupun evaluasi keamanan parameter.

Sementara itu, penelitian ini meneliti hal yang tidak dilakukan peneliti sebelumnya dengan mengimplementasikan pendekatan NTT–Karatsuba berbasis parameter hasil kompresi penelitian (Kundu et al., 2022), dan mengintegrasikannya kedalam proses key generation, enkripsi, dan dekripsi, serta mengungkap apakah ada kegagalan dekripsi pada algoritma NTRU. Penelitian ini juga menghitung waktu operasi polinomial, menampilkan ukuran kunci publik, dan ciphertext yang dihasilkan sebelum dan sesudah kompresi, serta menghitung besaran bandwith yang digunakan untuk ukuran ciphertext dari sebelum dan sesudah parameter q dikompresi. Lebih lanjut, tingkat bit keamanan dari parameter hasil kompresi juga dianalisis menggunakan *Lattice Estimator* untuk memastikan bahwa pengurangan ukuran parameter tidak menurunkan tingkat bit keamanannya. Maka demikian, penelitian ini tidak hanya menawarkan peningkatan pada waktu proses komputasi, tetapi juga memberikan evaluasi menyeluruh terhadap aspek ukuran, penggunaan sumber daya, dan nilai bit keamanan dari algoritma NTRU berbasis parameter hasil kompresi.