Sistem Kriptografi Kuantum

Perancangan dan Analisis Sistem Kriptografi Kuantum dalam Menghadapi *Cyber Attack Quantum*

Gabriel Parpunguan Halomoan Panjaitan (18221159) (*Author*)
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): gabrielpanjaitan46@gmail.com

***Abstract*— Dalam era digital yang semakin maju, ancaman siber menjadi semakin kompleks dengan munculnya teknologi komputer kuantum. Makalah ini membahas perancangan dan analisis sistem kriptografi kuantum dalam menghadapi serangan siber berbasis kuantum. Penelitian ini berfokus pada pengembangan protokol Distribusi Kunci Kuantum (Quantum Key Distribution - QKD) yang mampu memastikan keamanan komunikasi data melalui prinsip-prinsip mekanika kuantum. Melalui pendekatan teoritis dan eksperimental, makalah ini mengevaluasi efektivitas berbagai skema QKD, seperti BB84 dan E91, dalam menghadapi potensi ancaman dari komputer kuantum. Hasil penelitian menunjukkan bahwa sistem kriptografi kuantum dapat memberikan tingkat keamanan yang jauh lebih tinggi dibandingkan dengan sistem kriptografi klasik. Dengan kemampuan mendeteksi setiap upaya intersepsi, sistem ini mampu mempertahankan kerahasiaan dan integritas data secara optimal. Selain itu, analisis terhadap serangan siber berbasis kuantum menunjukkan bahwa dengan implementasi strategi mitigasi yang tepat, ancaman ini dapat diatasi secara efektif. Makalah ini juga membahas tantangan teknis dalam implementasi sistem kriptografi kuantum, termasuk kebutuhan infrastruktur dan biaya. Temuan ini memberikan wawasan penting bagi pengembang dan praktisi keamanan siber dalam merancang sistem keamanan yang tangguh terhadap serangan kuantum di masa depan. Dengan demikian, penelitian ini berkontribusi pada peningkatan keamanan informasi dan perlindungan terhadap ancaman siber yang berkembang pesat..**

I. PENDAHULUAN

Kemunculan komputasi kuantum merepresentasikan perubahan paradigma dalam keamanan infrastruktur. Kemampuan komputasi kuantum yang luar biasa mengancam metode kriptografi tradisional, seperti RSA dan ECC, yang selama ini melindungi integritas data. Ancaman ini semakin nyata seiring dengan kemampuan komputasi kuantum dalam menyelesaikan masalah kompleks seperti faktorisasi bilangan besar dan perhitungan logaritma diskret, yang berpotensi menghancurkan sistem enkripsi yang ada.

Penelitian ini mengeksplorasi secara mendalam ancaman yang timbul dari komputasi kuantum terhadap berbagai elemen infrastruktur digital, termasuk aplikasi, data, sistem operasi, dan jaringan. Terungkap bahwa sebelum beralih ke algoritma yang tahan kuantum, infrastruktur kriptografi rentan terhadap serangan yang dapat memecahkan algoritma seperti RSA, Diffie-Hellman, dan kriptografi kurva eliptik. Ancaman ini mencakup penyadapan data, pencurian identitas, penipuan keuangan, manipulasi data, dan spionase siber.

Meskipun organisasi mulai beralih ke algoritma tahan kuantum, mereka tetap rentan terhadap serangan seperti serangan penolakan layanan, serangan protokol kriptografi, rekayasa sosial, dan malware canggih yang sulit terdeteksi. Serangan-serangan ini dapat mengganggu infrastruktur kritis dan mengakses informasi sensitif.

Makalah ini menggunakan kriteria yang mapan dan pemetaan STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, dan Elevation of Privilege) untuk mengidentifikasi, mengevaluasi, dan memprioritaskan potensi ancaman terhadap aset-aset kritis. Selain itu, makalah ini menekankan pentingnya langkah-langkah tahan kuantum untuk melindungi dari serangan yang akan datang. Secara keseluruhan, penelitian ini memberikan wawasan penting tentang dampak komputasi kuantum pada keamanan infrastruktur, serta panduan praktis untuk pengembangan langkah-langkah pencegahan yang sesuai. Temuan ini merupakan langkah signifikan menuju peningkatan keamanan lingkungan jaringan di era komputasi kuantum.

II. PERKEMBANGAN KOMPUTASI KUANTUM

A. Tren Terkini dan Arah Masa Depan dalam Keamanan Tahan Kuantum

Kemajuan terbaru dalam komputasi kuantum menimbulkan kekhawatiran mengenai ancamannya terhadap sistem kriptografi konvensional. Beberapa peneliti menyoroti perlunya peta jalan yang tahan kuantum dan standar keamanan yang terukur. Kajian ini menekankan pentingnya peralihan sistematis ke PQC dan strategi yang matang untuk mencapai sistem perusahaan yang tahan kuantum.

B. Evolusi Komputasi dan Kriptografi Kuantum

Komputasi kuantum telah beralih dari eksplorasi teoretis ke aplikasi praktis dengan dampak signifikan pada kriptografi. Penelitian awal menunjukkan kerentanan protokol kriptografi

konvensional seperti RSA dan ECC terhadap serangan kuantum. Hal ini memicu pengembangan PQC, yang dirancang untuk aman terhadap ancaman komputasional klasik dan kuantum.

C. *Pengembangan dan Standarisasi Kriptografi Pasca-Kuantum*

Respon terhadap ancaman kuantum, penelitian dan standarisasi dalam PQC semakin cepat. Organisasi seperti NIST sedang mengevaluasi berbagai metode kriptografi yang tahan kuantum. Pengembangan PQC melibatkan berbagai pendekatan seperti kriptografi berbasis kisi, hash, kode, dan isogeni, masing-masing dengan kekuatan dan kasus penggunaan unik.

D. *Dampak Komputasi Kuantum pada Keamanan Infrastruktur*

Kemajuan pesat dalam komputasi kuantum menimbulkan tantangan signifikan terhadap infrastruktur digital, memerlukan pemahaman mendalam tentang potensi kerentanannya. Peningkatan teknologi komputasi kuantum oleh Google dan IBM menegaskan urgensi untuk menghadapi ancaman kuantum. Lindsay menganalisis dampak potensial komputasi kuantum pada protokol kriptografi, sementara Mangla et al.memfokuskan pada kerentanan jaringan 5G dan tantangan keamanan di masa depan untuk jaringan 6G.

E. *Mitigasi Ancaman Kuantum dalam Infrastruktur Digital*

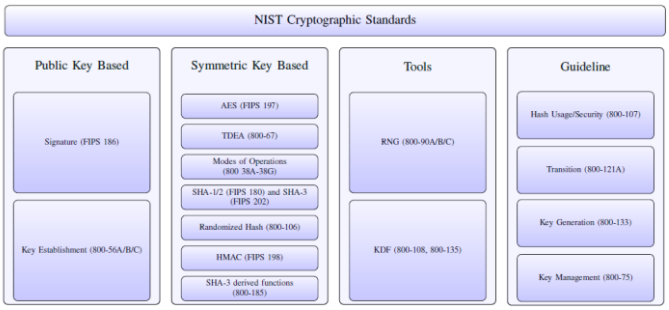
Berbagai perspektif muncul dalam menanggapi tantangan komputasi kuantum. Analisis Lindsay menyoroti interaksi kompleks antara infrastruktur teknologi dan institusi organisasi. Mangla et al. mempelajari kerentanan dalam jaringan 5G. Laporan Quantum Threat Timeline 2022 oleh Mosca dan Piani diharapkan memberikan penilaian kronologis tentang lanskap ancaman kuantum. Faruk et al. mengeksplorasi dualitas komputasi kuantum sebagai ancaman dan solusi di domain keamanan siber.

F. *Strategi Penyedia Layanan Cloud Utama dalam Mengamankan Infrastruktur dari Ancaman Kuantum*

Komputasi kuantum menawarkan peluang besar dan tantangan keamanan siber yang signifikan, memengaruhi infrastruktur cloud. Penyedia cloud utama seperti Amazon Web Services (AWS) , Microsoft Azure, dan Google Cloud Platform (GCP) sedang mengembangkan strategi untuk mengatasi tantangan ini. Mereka fokus pada kriptografi tahan kuantum (PQC), menggabungkan metode enkripsi saat ini dengan algoritma tahan kuantum, dan menekankan kelincahan kriptografi. Penyedia ini harus menyeimbangkan antara keamanan dan kinerja komputasi karena beberapa algoritma tahan kuantum kurang efisien. Upaya mereka mencakup semua lapisan infrastruktur, termasuk pengembangan aplikasi aman kuantum, enkripsi data, dan penerapan modul keamanan perangkat keras tahan kuantum. Kolaborasi dengan vendor perangkat keras, komunitas kriptografi, dan peneliti sangat penting, serta kepatuhan terhadap standar keamanan kuantum global seperti yang dipelopori oleh NIST.

III. Standar Kriptografi dan Komputasi Kuantum: Dampak Siber dan Analisis Resiko

Makalah ini memperkenalkan kerangka keamanan yang dirancang untuk menangani berbagai ancaman siber dari kemajuan komputasi kuantum, termasuk infrastruktur saat ini dan yang akan datang seperti lingkungan berbasis cloud. Kerangka ini berfokus pada sembilan elemen infrastruktur kritis: aplikasi, data, runtime, middleware, sistem operasi, virtualisasi, perangkat keras, penyimpanan, dan jaringan, untuk mengatasi kerentanan dan risiko terkait dalam menghadapi kemampuan komputasi kuantum. Penting untuk memahami ancaman serius komputasi kuantum terhadap sistem kriptografi yang ada, dan menggunakan metodologi penilaian risiko dari NIST, kerangka ini dirancang untuk melawan ancaman baru terhadap kriptografi tradisional dan pasca-kuantum. Dengan kedatangan komputer kuantum yang kuat, dampaknya pada sistem kriptografi publik dan simetris tidak dapat dihindari, dan bahkan metode kriptografi tahan kuantum yang sedang dipertimbangkan oleh NIST tidak sepenuhnya kebal. Makalah ini mengkaji risiko keamanan yang berkembang dari komputasi kuantum terhadap solusi kriptografi yang ada dan potensial.



Gambar 3.1 Standard Kriptografi NIST

A. *Ancaman terhadap kriptografi asimetris*

Algoritma klasik yang mendukung komunikasi aman, seperti RSA dan ECC, bisa dipecahkan oleh komputer kuantum menggunakan algoritma Shor.

B. *Ancaman terhadap kriptografi simetris*

Sistem kriptografi simetris menghadapi risiko dari algoritma kuantum seperti algoritma Grover dan Brassard et al., yang dapat melemahkan keamanan mereka.

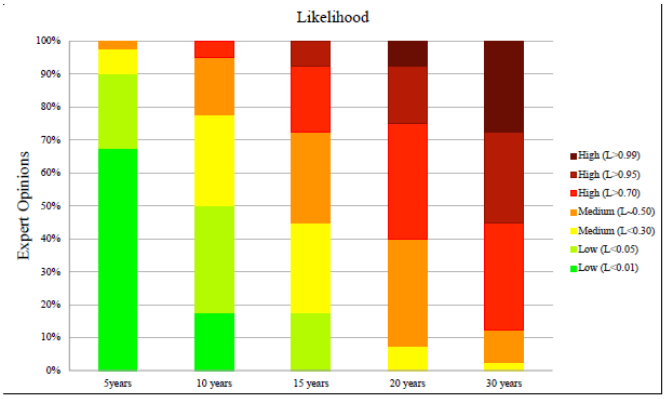
Crypto Type	Algorithms	Variants	Key Length (bits)	Classic Strength (bits)	Quantum Strength (bits)	Vulnerabilities	L	I	R	Recommended QC-Resistant Solutions
Asymetric	ECC [52]	ECC 256	256	128	0	Broken by Shor's Algorithm [19].	M	H	H	Algorithms presented in Table II.
		ECC 384	384	256	0		M	H	H	
		ECC 521	521	256	0		M	H	H	
	FFDHE [53]	DHE2048	2048	112	0	Broken by Shor's Algorithm [19].	M	H	H	
		DHE3072	3072	128	0		M	H	H	
	RSA [54]	RSA 1024	1024	80	0	Broken by Shor's Algorithm [19].	M	H	H	
		RSA 2048	2048	112	0		M	H	H	
		RSA 3072	3072	128	0		M	H	H	
Symmetric	AES [55]	AES 128	128	128	64	Weakened by Grover's Algorithm [56].	M	M	M	Larger key sizes are needed.
		AES 192	192	192	96		M	M	M	
		AES 256	256	256	128		M	L	L	
	SHA2 [57]	SHA 256	-	128	85	Weakened by Brassard et al.'s Algorithm [58].	M	M	M	Larger hash values are needed.
		SHA 384	-	192	128		L	L	L	
		SHA 512	-	256	170		M	L	L	
	SHA3 [57]	SHA3 256	-	128	85	Weakened by Brassard et al.'s Algorithm [58].	M	M	M	
		SHA3 384	-	192	128		M	L	L	
		SHA3 512	-	256	170		M	L	L	

Gambar 3.2 Classic Cryptographic Standards and Quantum Computing: Assessing Cyber Risks

C. *Mengukur Ancaman Kuantum*

Prediksi memperkirakan kemunculan komputer kuantum dalam 5 hingga 30 tahun ke depan, yang mampu memecahkan RSA-2048 dalam 24 jam. Prediksi para ahli digunakan untuk

menghitung kemungkinan ancaman ini selama berbagai rentang waktu (5, 10, 15, 20, dan 30 tahun).



Gambar 3.3 Cumulative Expert Opinions Related to Quantum Threat to Classic Cryptography

D. Penilaian Dampak Kuantum

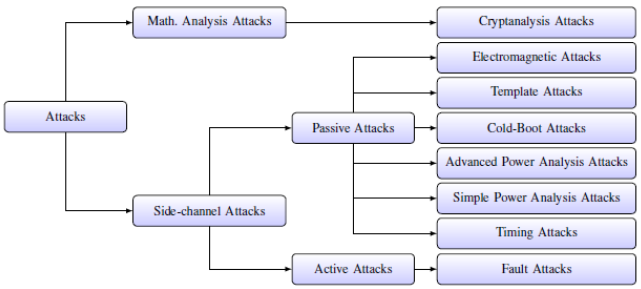
Dampak ancaman kuantum terhadap algoritma klasik dievaluasi berdasarkan kekuatan keamanan kuantumnya. Dampak tinggi jika keamanan di bawah 64 bit, rendah jika 128 bit atau lebih, dan menengah jika di antaranya.

E. Transisi ke Algoritma Kriptografi tahan kuantum

- Perlunya Transisi: Algoritma kriptografi kunci publik saat ini perlu diganti dengan algoritma kriptografi tahan kuantum.
- Inisiatif NIST: National Institute of Standards and Technology (NIST) meluncurkan inisiatif untuk menstandarisasi algoritma kriptografi tahan kuantum melalui kompetisi.
- Jenis Algoritma Pasca-Kuantum:
  - Berbasis kisi (lattice-based)
  - Berbasis kode (code-based)
  - Berbasis hash (hash-based)
  - Berbasis isogeny (isogeny-based)

F. Tantangan Algoritma Tahan Kuantum

Tantangan dapat berupa eksploitasi kebocoran informasi selama pelaksanaan algoritma kriptografi, seperti konsumsi daya, radiasi elektromagnetik, atau informasi waktu. Analisis kriptografi digunakan untuk memecahkan skema enkripsi atau tanda tangan dengan mengidentifikasi kelemahan struktural dalam algoritma. Beberapa algoritma tahan kuantum yang dipertimbangkan oleh NIST telah menjadi target serangan saluran samping dan analisis kriptografi.



Gambar 3.4 Taxonomy of Attacks for Quantum-Safe Cryptographic

IV. IMPLIKASI KEAMANAN SIBER KUANTUM KOMPUTASI PADA INFRASTRUKTUR DIGITAL DI ERA PRA-MIGRASI

A. Dampak Siber Komputasi Kuantum pada Fase Pra-Migrasi

Komputasi kuantum menghadirkan tantangan signifikan bagi sistem kriptografi tradisional. Banyak algoritma kriptografi saat ini, yang sangat penting untuk melindungi infrastruktur dan data rahasia, berisiko menjadi tidak efektif terhadap serangan komputer kuantum. Organisasi yang lambat mengadopsi metode kriptografi tahan kuantum berisiko terpapar berbagai ancaman yang didukung oleh teknologi kuantum, termasuk pelanggaran kriptografi, pencurian identitas, penipuan keuangan, dan manipulasi data. Quantum computing memiliki kemampuan untuk mendekripsi metode enkripsi standar seperti RSA, Diffie-Hellman, dan kriptografi kurva eliptik, yang dapat menyebabkan akses tidak sah ke informasi penting, termasuk komunikasi pribadi, kata sandi, dan transaksi keuangan. Eksploitasi tanda tangan digital oleh komputer kuantum juga dapat memfasilitasi pencurian identitas, memungkinkan akses tidak sah ke sistem keamanan tinggi atau data sensitif, yang dapat berdampak pada operasi pemerintah atau militer. Selain itu, kemampuan komputer kuantum untuk merusak keamanan kriptografi dalam transaksi keuangan dapat menyebabkan penggelapan dana, transfer tidak sah, atau manipulasi catatan keuangan. Komputasi kuantum juga mungkin memungkinkan perubahan dalam penyimpanan data digital, yang mengarah pada manipulasi catatan medis, laporan keuangan, atau basis data pemilu. Negara atau entitas kuat yang memanfaatkan kemampuan komputasi kuantum dapat terlibat dalam kegiatan spionase lanjutan, menargetkan data rahasia dan strategis seperti rahasia dagang industri atau informasi keamanan nasional. Untuk mengurangi kerentanan ini, organisasi disarankan untuk secara proaktif beralih ke teknologi kriptografi tahan kuantum. Selain peningkatan kriptografi ini, menerapkan kontrol akses yang kuat serta mekanisme deteksi dan respons ancaman lanjutan adalah strategi penting untuk memperkuat keamanan siber di era kuantum.

B. *Vektor Serangan Komputasi Kuantum pada Fase Pra-Migrasi*

Komputasi kuantum mempengaruhi sistem klasik dengan berbagai penggunaan potensial seperti rekayasa kuantum, kriptografi, pembelajaran mesin, dan kecerdasan buatan. Akibatnya, penyerang kuantum memiliki kemampuan untuk merusak beberapa algoritma enkripsi yang paling menonjol saat ini. Bagian ini mengeksplorasi tantangan keamanan yang ada dalam infrastruktur sebelum migrasi ke standar kriptografi pasca-kuantum dan menghadirkan ancaman, kerentanan, vektor serangan, dan jenis kriptografi dari sistem klasik yang ada untuk mengevaluasi risiko yang dapat disebabkan oleh penyerang kuantum. Klasifikasi ini membantu analis keamanan menemukan langkah mitigasi dan mengidentifikasi masalah keamanan dengan lebih cepat. Vektor serangan yang dapat disebabkan oleh mesin kuantum meliputi:

1. Penyuntikan Kode: Algoritma kuantum dapat mematahkan metode enkripsi saat ini, memungkinkan penyuntikan kode berbahaya ke dalam pembaruan perangkat lunak atau komunikasi yang dipercaya;
2. Eksploitasi Hypervisor: Penyerang kuantum dapat menyusup ke hypervisor dalam lingkungan virtualisasi, memungkinkan kontrol penuh atas kerangka kerja virtual;
3. Migrasi VM dan Risiko Kuantum: Enkripsi data selama migrasi VM terancam oleh kemampuan kuantum untuk mendekripsi informasi dengan cepat;
4. Eksploitasi Kernel OS: Algoritma kuantum dapat merusak integritas dan otentikasi kernel sistem operasi, memungkinkan pengambilalihan sistem total;
5. Eksploitasi Firmware: Penyerang kuantum dapat merusak pembaruan firmware, menimbulkan risiko spionase dan sabotase;
6. Pemulihan Kunci Kriptografi Simetris: Algoritma kuantum dapat secara signifikan mengurangi waktu brute-force kunci kriptografi simetris, mengancam kerahasiaan data;
7. Eksfiltrasi Data yang Tidak Sah: Algoritma kuantum dapat mendekripsi dan mengakses data rahasia dalam VM dan komunikasi terenkripsi;
8 Pelanggaran Enkripsi Server Penyimpanan/Disk: Penyerang kuantum dapat mendekripsi kunci penyimpanan, membahayakan kerahasiaan dan integritas data;
9. Kolisi Permintaan POST: Serangan kolisi yang diperkuat kuantum dapat menyebabkan server kelebihan beban dengan permintaan POST yang intensif sumber daya;
10. Eksploitasi Enkripsi yang Diperkuat Kuantum: Penyerang kuantum dapat mengenkripsi data dengan algoritma kompleks yang sulit didekripsi dengan metode konvensional;
11. Pelanggaran Titik Akses (AP): Jaringan Wi-Fi dengan protokol lama rentan terhadap serangan kuantum yang dapat mendekripsi komunikasi terenkripsi, memungkinkan akses tidak sah dan serangan berbasis jaringan.

C. *Tantangan Keamanan pada Setiap Lapisan Infrastruktur*

Berdasarkan vektor serangan yang diuraikan, berbagai ancaman di berbagai lapisan infrastruktur muncul, menyoroti perlunya penilaian ulang yang komprehensif. Ancaman-ancaman ini menantang kerangka keamanan tradisional dan memerlukan pergeseran paradigma dalam strategi pertahanan dan standar kriptografi. Setiap lapisan, dari penyimpanan data hingga transmisi jaringan, menghadapi kerentanan unik, yang penting dalam era komputasi kuantum yang sedang berkembang.

Pada Lapisan Aplikasi, komputasi kuantum memperkenalkan risiko signifikan terhadap keamanan data dan otentikasi pengguna, memerlukan adopsi algoritma tahan-kuantum dan peningkatan protokol keamanan. Lapisan Data menghadapi ancaman besar karena kemampuan komputasi kuantum untuk mendekripsi data dengan cepat, sehingga integrasi primitif kriptografi yang aman-kuantum sangat penting. Lapisan Waktu Nyata menghadapi risiko eksposur data sensitif dan kompromi lingkungan eksekusi, yang memerlukan standar enkripsi tahan-kuantum dan model enkripsi hibrida. Lapisan Middleware rentan terhadap serangan yang memanfaatkan teknik kriptanalisis kuantum, memerlukan penerapan tanda tangan berbasis hash stateful dan saluran terenkripsi yang aman-kuantum.

Pada Lapisan Sistem Operasi, implikasi keamanan mencakup potensi pelanggaran kriptografi tingkat kernel, yang memerlukan peningkatan perlindungan kernel dan adopsi skema tanda tangan yang aman-kuantum. Lapisan Virtualisasi menghadapi ancaman dari serangan kuantum yang menargetkan hypervisor dan akses tidak sah ke infrastruktur virtualisasi, yang memerlukan peningkatan algoritma kriptografi tahan-kuantum dan isolasi enforced-hardware. Lapisan Perangkat Keras terancam oleh algoritma kuantum canggih yang dapat merusak data terenkripsi, memerlukan proses boot yang aman dan peningkatan enkripsi dengan solusi yang aman-kuantum. Lapisan Penyimpanan menghadapi risiko eksposur data dan akses tidak sah, memerlukan teknik enkripsi tahan-kuantum dan audit keamanan reguler. Lapisan Jaringan menjadi medan pertempuran baru dalam keamanan siber, memerlukan pergeseran menuju standar kriptografi yang dapat menahan serangan kuantum dan penguatan pertahanan jaringan secara keseluruhan.

V. IMPLIKASI KEAMANAN SIBER KUANTUM KOMPUTASI PADA INFRASTRUKTUR DIGITAL DI ERA PASCA-MIGRASI

A. *Dampak Cyber Komputasi Kuantum dalam Fase Pasca-Migrasi*

Transisi ke algoritma kriptografi yang aman dari kuantum menandai langkah penting dalam melindungi organisasi dari kekuatan komputasi maju dari komputer kuantum. Namun,

transisi ini juga membawa tantangan keamanan cyber baru yang melampaui ancaman dekripsi semata. Salah satu tantangan utama adalah isu terkait dengan (a) peningkatan ukuran kunci dan lalu lintas jaringan, (b) kompleksitas implementasi, (c) overhead kinerja, dan (d) penyesuaian perangkat keamanan jaringan. Peningkatan ukuran kunci dan lalu lintas jaringan merupakan adopsi PQC umumnya menghasilkan kunci kriptografi dan teks terenkripsi yang lebih besar. Hal ini dapat menyebabkan lalu lintas jaringan yang lebih terfragmentasi, menimbulkan tantangan bagi sistem dengan kemampuan terbatas dalam mengelola dan merakit kembali data yang terfragmentasi. Kompleksitas implementasi dari PQC adalah integrasi PQC menambah lapisan kompleksitas pada sistem yang ada, berpotensi menciptakan kerentanan keamanan baru. Kerentanan ini khususnya terasa dalam manajemen paket jaringan yang terfragmentasi, membutuhkan langkah-langkah proaktif untuk pemeliharaan keamanan yang kuat di era kuantum. Algoritma PQC dapat menurunkan kinerja, terutama dalam lingkungan lalu lintas tinggi. Penurunan ini dapat membebani infrastruktur jaringan, meningkatkan kerentanan terhadap serangan yang memanfaatkan kerentanan terfragmentasi. Volume lalu lintas terenkripsi PQC yang semakin meningkat memerlukan kemajuan pada perangkat keamanan jaringan, termasuk firewall dan sistem deteksi intrusi, untuk efektif memproses dan memeriksa tipe lalu lintas baru ini. Selama fase transisi, mungkin ada kerentanan sementara yang dapat memperparah efek serangan berbasis fragmentasi saat ini dan potensial munculnya ancaman siber baru. Meskipun transisi ke Kriptografi Pasca-Kuantum (PQC) penting dalam mitigasi ancaman dari komputasi kuantum, itu memperkenalkan sejumlah tantangan keamanan cyber baru. Oleh karena itu, adopsi PQC adalah solusi penting namun tidak menyeluruh. Strategi keamanan komprehensif diperlukan, yang memperkuat sistem terhadap berbagai ancaman yang kompleks yang bisa muncul di lanskap pasca-kuantum. Bagian berikutnya akan mengeksplorasi berbagai vektor serangan yang mungkin muncul dalam infrastruktur yang beralih ke PQC.

B. Vektor Serangan Komputasi Kuantum dalam Fase Pasca-Migrasi

Sistem kriptografi pasca-kuantum menggunakan algoritma tahan kuantum, melindungi dari serangan baik oleh komputer kuantum maupun klasik. Meskipun demikian, penyerang kuantum terus berupaya untuk merusak kriptografi pasca-kuantum dengan mengidentifikasi dan mengeksploitasi kerentanan. Mereka terus menggunakan mesin kuantum untuk mengevaluasi titik lemah, menerapkan algoritma kuantum, serangan sisi (seperti yang dijelaskan dalam Tabel II), kriptoanalisis, penyisipan kode, dan lainnya. Bagian ini mengulas tantangan keamanan yang mungkin muncul dalam infrastruktur setelah beralih ke standar kriptografi PQ (dirangkum dalam Tabel IV) dan mengeksplorasi ancaman, kerentanan, vektor serangan, dan jenis kriptografi terkait. Informasi ini bertujuan untuk membantu analis keamanan dalam mengidentifikasi dan mengatasi masalah keamanan dengan cepat. Di bawah ini, kami membahas potensi vektor serangan yang berasal dari mesin kuantum dalam sistem berbasis kriptografi PQ.

Penyerang kuantum mampu merusak algoritma kriptografi dengan mengeksploitasi kelemahan implementasi. Serangan sisi mungkin berasal dari kolokasi mesin virtual (VM) atau host, infrastruktur jaringan bersama, dan skenario lain yang melibatkan sumber daya bersama. Kerentanan kunci termasuk Pemanfaatan Kernel OS, Pemanfaatan Hypervisor, Pemantauan VM, Pemeriksaan Komunikasi Inter-VM, serta Cross-VM, Cache Mikro-arsitektur, dan kesalahan. Kerentanan ini dapat menyebabkan pengungkapan informasi sensitif, pemalsuan data, dan tantangan dalam memastikan non-repudiasi.

Penyerang kuantum mungkin melakukan penyisipan kode, mengeksploitasi batas buffer memori untuk menyisipkan kode jahat atau rentan. Dalam serangan seperti itu, kode jahat atau rentan disisipkan ke dalam sistem dengan mengeksploitasi kerentanan ini, berpotensi memengaruhi Perangkat Lunak Aplikasi, Hypervisors, dan Firmware. Ancaman mungkin termasuk kode tidak aman dan Penyisipan Perinta, yang mengarah ke eksekusi Keyloggers, Virus, Cacing, Pintu Perangkap, Malware Tanpa Berkas, Trojan, Rootkit, Spyware, Crimeware, Pelarian VM, Pencurian Data dan Sabotase Sistem

Dalam konteks kriptografi pasca-kuantum dan kemajuan dalam komputasi kuantum, sistem file dan hard drive menghadapi risiko yang meningkat. Kerentanan ini sering berasal dari daemon jaringan, klien email, atau browser web, yang dapat menjadi jalur bagi penyerang kuantum. Dengan memanfaatkan kerentanan overflow buffer, penyerang dapat menyisipkan malware penghapus ke dalam sistem. Jenis malware ini sangat berbahaya, menargetkan data penting untuk dihapus atau rusak, yang dapat menyebabkan gangguan sistem yang parah dan tidak dapat diperbaiki.

C. Tantangan Keamanan di Setiap Lapisan Infrastruktur dalam Fase Pasca-Migrasi

Berbasis pada vektor serangan terperinci yang terkait dengan mesin kuantum dalam sistem yang telah bermigrasi ke kriptografi pasca-kuantum, kita dapat melihat serangkaian ancaman yang muncul yang memengaruhi beberapa lapisan infrastruktur. Lanskap baru ini membutuhkan penilaian ulang komprehensif terhadap strategi keamanan dan langkah-langkah di setiap lapisan berikut.

Lapisan Aplikasi menghadapi tantangan kriptografi pasca-kuantum yang signifikan, terutama dalam manajemen kunci dan ciphertext. Untuk mengurangi risiko, praktik pengkodean yang aman sangat penting, termasuk pemeriksaan batas dan audit kode untuk mencegah kerentanan sisi saluran. Isolasi VM yang kuat dan kebijakan alokasi sumber daya yang ketat penting untuk melindungi dari eksploitasi sumber daya bersama. Penerapan bahasa pemrograman yang aman, menjaga perangkat lunak tetap terbaru, dan mendeploy sistem deteksi intrusi canggih kritis untuk mencegah manipulasi dan akses tidak sah.

Lapisan Data rentan terhadap ancaman pengungkapan informasi, terutama dari penyerang yang mampu

menggunakan teknik kriptoanalisis canggih untuk mengeksploitasi kerentanan seperti serangan sisi saluran dan overflow buffer. Isolasi VM yang ketat, penggunaan modul keamanan perangkat keras, dan audit keamanan reguler penting untuk mendeteksi dan memperbaiki kerentanan.

Pada lapisan runtime, penyerang dapat memanfaatkan kerentanan seperti akses memori out-of-bounds atau korupsi memori untuk memfasilitasi serangan seperti Return Oriented Programming (ROP) dan Jump-Oriented Programming (JOP). Penting untuk menerapkan pengelolaan patch reguler dan mekanisme perlindungan runtime untuk mengurangi risiko ini.

Lapisan Middleware rentan terhadap serangan DoS yang disebabkan oleh serangan kelebihan kapasitas, di mana sumber daya host dibanjiri oleh lalu lintas terenkripsi berat dari VM jahat. Protokol migrasi VM yang dioptimalkan juga diperlukan untuk menangani beban enkripsi pasca-kuantum dengan efisien dan menjaga kontinuitas layanan selama periode lalu lintas berat.

Lapisan Perangkat Keras rentan terhadap serangan sisi saluran yang rumit yang memanfaatkan kerentanan dalam komponen-komponen tertentu seperti akselerator kriptografi dan cache bersama dalam prosesor multi-core. Implementasi mekanisme keamanan firmware yang aman dan penerapan teknik mitigasi serangan sisi saluran diperlukan untuk melawan ancaman ini.

Integritas dan ketersediaan data dalam lapisan penyimpanan dapat terancam karena kerentanan overflow buffer yang dapat dieksploitasi oleh serangan wiper dan ransomware yang rumit. Perlindungan overflow buffer dan pembaruan perangkat lunak reguler diperlukan untuk melawan ancaman ini.

Lapisan jaringan rentan terhadap sejumlah kerentanan, termasuk dari analisis kriptografis, kebocoran sisi saluran, injeksi kesalahan, dan fragmentasi. Penting untuk menerapkan pembaruan algoritma reguler, enkripsi yang ditingkatkan, isolasi dan teknik pengelolaan sumber daya untuk melawan ancaman ini.

Dalam keseluruhan, penanganan tantangan keamanan pasca-migrasi memerlukan pendekatan yang berlapis dan tanggap, termasuk integrasi kriptografi tahan kuantum yang terus-menerus, manajemen kunci yang kuat, dan praktik arsitektur keamanan yang kokoh. Sinergi antara vendor, implementor, dan pengguna penting untuk mencapai tujuan ini.

VI. FRODO (KRIPTOGRAFI BERBASIS LATTICE)

Kemajuan terkini dalam kriptografi pasca-kuantum telah memicu minat luas dalam pengembangan skema kriptografi praktis yang mampu bertahan dari serangan kuantum. Minat ini mendorong badan-badan standar dan lembaga pemerintah, seperti NIST, NSA, dan proyek PQCRYPTO, untuk mengumumkan niat mereka untuk beralih ke standar kriptografi yang menawarkan ketahanan terhadap kuantum. Masalah kriptografi tradisional, seperti faktorisasi bilangan bulat dan logaritma diskret, rentan terhadap serangan kuantum dalam waktu polinomial, yang menyoroti urgensi untuk pendekatan baru. Kriptografi berbasis lattice muncul sebagai bidang yang menjanjikan, menawarkan primitif yang beragam dan potensi untuk solusi tahan kuantum yang praktis. Pertukaran kunci dan kerahasiaan ke depan diidentifikasi sebagai aspek penting dalam menghadapi ancaman kuantum, menekankan perlunya protokol yang aman dan efisien untuk melindungi lalu lintas internet. Dalam konteks ini, fokus pada pengembangan protokol pertukaran kunci yang tahan kuantum sejalan dengan prinsip kerahasiaan ke depan dan keamanan jangka panjang. Perbandingan antara lattice generik dan ideal menekankan pentingnya evaluasi yang ketat dan eksplorasi masalah matematis baru untuk keamanan pasca-kuantum. Pengenalan protokol inovatif, seperti "Frodo," berdasarkan pada masalah LWE asli, menunjukkan kelayakan pertukaran kunci berbasis lattice tanpa struktur tambahan cincin. Evaluasi kinerja protokol, termasuk mikrobenchmark dan skenario penerapan dunia nyata, mengilustrasikan kelayakan sebagai skema tahan kuantum yang praktis, menawarkan alternatif yang menjanjikan terhadap pendekatan berbasis lattice ideal.

Protokol pertukaran kunci dari LWE (Learning With Errors) adalah sebuah metode yang mengandalkan kesulitan dalam menyelesaikan permasalahan LWE untuk membentuk kunci rahasia yang digunakan dalam kriptografi pasca-kuantum. Dalam protokol ini, terdapat tiga parameter yang digunakan: modulus q, dimensi matriks n, dan distribusi kesalahan ψ. Protokol ini melibatkan dua entitas, Alice dan Bob, yang saling menghasilkan matriks A yang sama dan dikombinasikan dengan rahasia LWE untuk menghitung kunci publik mereka. Kunci bersama K dihasilkan dari kunci rahasia LWE mereka yang kemudian diekstraksi dalam bentuk bilangan biner. Protokol ini dapat diintegrasikan ke dalam protokol TLS (Transport Layer Security) dengan server berperan sebagai Alice dan klien berperan sebagai Bob. Penggunaan protokol ini dalam TLS memungkinkan pertukaran kunci yang aman antara server dan klien. Selain itu, protokol ini dapat diimplementasikan dalam \textit{hybrid ciphersuites}, di mana kriptografi berbasis lattice dipadukan dengan skema legacy seperti ECDH (Elliptic Curve Diffie-Hellman) untuk meningkatkan keamanan kriptografi. Protokol ini juga mencakup mekanisme rekonsiliasi yang memungkinkan kedua belah pihak untuk mencapai kesepakatan kunci yang eksak, meskipun terdapat kesalahan yang mungkin terjadi selama pertukaran kunci. Kesalahan dalam pertukaran kunci ini diakibatkan oleh distribusi kesalahan yang digunakan, yang dipilih sesuai dengan kebutuhan kriptografi yang diinginkan. Dalam implementasinya, terdapat empat distribusi kesalahan yang digunakan, masing-masing memiliki PDF (Probability Density Function) yang berbeda dan mewakili pendekatan yang berbeda terhadap distribusi Gaussian yang dibulatkan. Keempat distribusi ini dipilih berdasarkan pada kebutuhan tertentu dan mendekati distribusi Gaussian yang dibulatkan dengan varian yang diberikan. Distribusi kesalahan yang dipilih akan memengaruhi keamanan dan kinerja protokol pertukaran kunci LWE.

"Security Assessment and Parameter Selection" digunakan untuk memberikan estimasi keamanan yang

konservatif terhadap serangan klasik dan kuantum, serta menyarankan parameter-parameter yang sesuai untuk protokol yang dibahas sebelumnya. Metodologi ini didasarkan pada penelitian sebelumnya, dengan sedikit modifikasi yang memperhitungkan beberapa faktor penting. Penting untuk dicatat bahwa pendekatan yang digunakan di sini jauh lebih konservatif dibandingkan dengan yang umumnya digunakan dalam literatur, dengan tujuan utama bukan hanya untuk menunjukkan kelayakan, tetapi juga untuk memberikan jaminan keamanan jangka panjang. Penjelasan tentang metode analisis dan pilihan parameter, bersama dengan pertimbangan terkait, memberikan dasar yang kuat untuk memastikan keamanan sistem yang diusulkan. Selain itu, bagian ini menyajikan parameter-parameter yang direkomendasikan dengan detail dan menyajikan perbandingan dengan parameter-parameter lainnya yang mungkin ada dalam literatur, memberikan panduan yang jelas bagi pembuat kebijakan dan pengembang sistem untuk memilih parameter yang sesuai dengan kebutuhan keamanan mereka.

Implementasi Kodenya sebagai berikut

```
function initMatrixDefault(x, y) {
   return Array.from({ length: x }, () => new Array(y));
}

function initMatrixRandom(x, y, q) {
   return Array.from({ length: x }, () => Array.from({ length: y }, () => nextInt(q)));
}

function multiply(A, B) {
   const A_x = A.length;
   const A_y = A[0].length;
   const B_x = B.length;
   const B_y = B[0].length;
   if (B_x !== A_y) {
      alert("Matrix inner dimensions must agree");
      return;
   }
   return Array.from({ length: A_x }, (_, i) =>
      Array.from({ length: B_y }, (_, j) => {
         let sum = 0;
         for (let k = 0; k < A_y; k++) {
            sum += A[i][k] * B[k][j];
         }
         return sum;
      })
   );
}

function addMod(A, B, q) {
   checkDimensions(A, B);
   const A_x = A.length;
   const A_y = A[0].length;
   return Array.from({ length: A_x }, (_, i) =>
      Array.from({ length: A_y }, (_, j) => {
         let result = A[i][j] + B[i][j];
         if (result < 0) result += q;
         return result % q;
      })
   );
}

function nextInt(q) {
   return Math.floor(Math.random() * q);
}

function checkDimensions(A, B) {
   const A_x = A.length;
   const A_y = A[0].length;
   const B_x = B.length;
   const B_y = B[0].length;
   if (B_x !== A_x || B_y !== A_y) {
      alert("Matrix dimensions must agree");
      return;
   }
}

// Pengujian
const A = initMatrixRandom(2, 2, 5);
const B = initMatrixRandom(2, 2, 5);
const C = multiply(A, B);
console.log("Result of A * B:", C);
```

Penjelasan

- initMatrixDefault(x, y): Fungsi ini digunakan untuk membuat matriks dengan ukuran x baris dan y kolom, diisi dengan nilai-nilai default (biasanya nol). Fungsi ini menggunakan fungsi Array.from() untuk membuat array dengan panjang x, lalu menginisialisasi setiap elemen array tersebut menjadi array baru dengan panjang y. Hasil akhirnya adalah matriks dengan ukuran x x y.

- initMatrixRandom(x, y, q): Fungsi ini mirip dengan initMatrixDefault, tetapi kali ini nilai-nilai matriks diisi secara acak. Setiap elemen matriks dihasilkan dengan menggunakan fungsi nextInt(q), yang menghasilkan nilai acak antara 0 dan q-1, di mana q adalah parameter yang diberikan.

- multiply(A, B): Fungsi ini mengalikan dua matriks, A dan B. Pertama, fungsi memeriksa apakah dimensi dalam matriks tersebut cocok untuk perkalian matriks, yaitu jumlah kolom matriks A harus sama dengan jumlah baris matriks B. Jika dimensi tidak cocok, fungsi akan menampilkan pesan kesalahan dan mengembalikan undefined. Jika dimensi cocok, fungsi akan melakukan perkalian matriks sesuai aturan matematika standar. Hasilnya adalah matriks baru yang merupakan hasil perkalian matriks A dan B.

- addMod(A, B, q): Fungsi ini menambahkan dua matriks A dan B secara modulo q. Ini berarti setiap elemen hasil penjumlahan akan dihitung sebagai penjumlahan elemen-elemen yang sesuai dalam matriks A dan B, dan kemudian diambil modulo q. Ini sering digunakan dalam operasi kriptografi. Jika hasil penjumlahan negatif, maka akan ditambahkan q agar hasilnya tetap positif.

- nextInt(q): Fungsi ini menghasilkan nilai acak antara 0 dan q-1, di mana q adalah parameter yang diberikan.

- checkDimensions(A, B): Fungsi ini memeriksa apakah dimensi dua matriks A dan B sama. Jika dimensi tidak sama, fungsi akan menampilkan pesan kesalahan. Ini sering digunakan sebelum melakukan operasi seperti penambahan atau perkalian matriks untuk memastikan matriks memiliki dimensi yang sesuai.

```
var m = 8,
    n = 752,
    l = 8,
    a = 11,
    b = 4;
q = 32768,
    logq = 15;
sigma = 1.3229;

function alice0(l, n, q) {
    var amatrix = initMatrixRandom(n, n, q);
    var smatrix = ss; //n*l
    var ematrix = ee; //n*l
    var bmatrix = multiply(amatrix, smatrix);
    bmatrix = addMod(bmatrix, ematrix, q);

    am = amatrix;
    bm = bmatrix;
    sm = smatrix;
}

function bob(l, m, n, q) {
    var amatrix = am;
    var bmatrix = bm;

    var s1matrix = ss1; // Z^m*n
    var e1matrix = ee1; // Z^m*n
    var e2matrix = ee2; // Z^m*l

    var b1matrix = multiply(s1matrix, amatrix);
    var vmatrix = multiply(s1matrix, bmatrix);

    b1matrix = addMod(b1matrix, e1matrix, q);
    vmatrix = addMod(vmatrix, e2matrix, q);
    var cmatrix = new Array(m);
    var k1matrix = new Array(m);

    for (var i = 0; i < l; i++) {
        cmatrix[i] = vmatrix[i].slice();
        k1matrix[i] = vmatrix[i].slice();
    }

    for (var i = 0; i < m; i++) {
        for (var j = 0; j < l; j++) {
            cmatrix[i][j] = (cmatrix[i][j] >> 10) & 1;

            k1matrix[i][j] = (k1matrix[i][j] + 1024) % q;
            k1matrix[i][j] >>= 11;
        }
    }

    bb = b1matrix;
    cc = cmatrix;
}

function alice1(l, m, q) {
    var b1m = bb;
    var cm = cc;
    var smatrix = sm;

    var b1s = multiplyMod(b1m, smatrix, q);
    k2matrix = b1s;
    rec(k2matrix, m, l, 11, cm);
}
```

Pertama-tama, parameter-parameter seperti m, n, l, a, b, q, logq, dan sigma diinisialisasi. Langkah pertama dimulai oleh Alice melalui fungsi alice0(l, n, q), di mana dia menghasilkan matriks acak amatrix dengan ukuran n x n. Matriks ini kemudian dikalikan dengan matriks publik smatrix dan ditambahkan dengan matriks kesalahan ematrix, menghasilkan matriks bmatrix. Bob, melalui fungsi bob(l, m, n, q), melakukan serangkaian operasi serupa, menggunakan matriks publik dan matriks hasil dari langkah Alice untuk menghasilkan dua matriks kunci, k1matrix dan k2matrix. Terakhir, Alice menggunakan matriks kunci bb yang diperoleh dari Bob, bersama dengan matriks publik smatrix, untuk menghasilkan matriks b1s melalui perkalian modulo dalam fungsi alice1(l, m, q). Matriks k2matrix yang dihasilkan kemudian digunakan dalam operasi rec. Keseluruhan proses ini menggambarkan alur pertukaran kunci rahasia antara Alice dan Bob dalam protokol Frodo, di mana kunci rahasia dihasilkan dengan aman tanpa perlu dipertukarkan secara langsung.

performa pertukaran kunci berbasis LWE dievaluasi berdasarkan tiga karakteristik utama: kecepatan operasi kriptografi mandiri, kecepatan koneksi HTTPS, dan biaya komunikasi. Implementasi LWE yang digunakan ditulis dalam bahasa C dan diintegrasikan ke dalam OpenSSL v1.0.1f untuk

memungkinkan perbandingan dengan beberapa primitive post-kuantum lainnya.

Beberapa algoritma post-kuantum yang dievaluasi meliputi BCNS R-LWE key exchange, NewHope R-LWE key exchange, NTRU public key encryption key transport, dan SIDH (supersingular isogeny Diffie-Hellman) key exchange. Selain itu, implementasi OpenSSL dari ECDH dan RSA juga dimasukkan untuk perbandingan dengan pertukaran kunci non-post-kuantum yang banyak digunakan.

Evaluasi dilakukan dengan mengukur kecepatan operasi kriptografi mandiri menggunakan perintah openssl speed, serta performa koneksi HTTPS menggunakan web server Apache HTTP dengan prefork module untuk multi-threading. Pengujian dilakukan dengan memanfaatkan perangkat keras yang sama untuk semua pengukuran.

Hasil evaluasi menunjukkan bahwa pertukaran kunci berbasis LWE memiliki kinerja yang menjanjikan, terutama dalam kecepatan operasi kriptografi mandiri. Walaupun ada perbedaan signifikan dalam kinerja mikro antara LWE dan R-LWE (NewHope), gap ini menjadi lebih kecil ketika diukur dalam aplikasi yang menggunakan TLS. Selain itu, meskipun LWE memiliki dampak pada kinerja server, pengurangan throughput relatif kecil dibandingkan dengan R-LWE (NewHope), terutama saat ukuran halaman meningkat.

Secara keseluruhan, evaluasi menunjukkan bahwa meskipun LWE mempengaruhi kinerja TLS, dampaknya pada waktu koneksi dan ukuran handshake dapat diterima dalam banyak aplikasi, seperti penjelajahan web dan transfer data yang aman. Implementasi kriptografi berbasis LWE menawarkan alternatif yang menarik untuk mencapai keamanan pasca-kuantum tanpa tergantung pada struktur cincin, yang dapat mengurangi kemungkinan serangan spesifik pada struktur tersebut.

VII. McElice (Kriptografi kuantum berbasis kode)

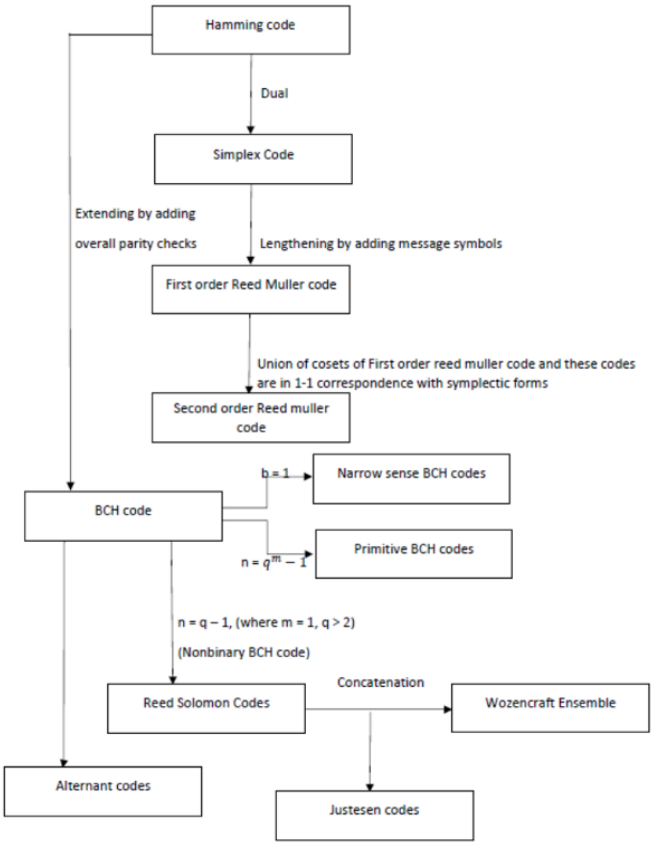
Kode linear adalah sebuah pendekatan dalam kriptografi yang mengadaptasi konsep dari teori pengkodean, yang awalnya digunakan dalam komunikasi digital untuk mengurangi kerugian data akibat kesalahan yang mungkin terjadi selama transmisi, seperti gangguan sinyal atau noise. Dalam konteks kriptografi, kode linear memungkinkan pengiriman pesan yang terenkripsi melalui media yang tidak aman dengan tingkat kesalahan yang dapat ditoleransi. Prinsip dasarnya adalah dengan mengkodekan data menggunakan matriks linear sehingga pesan yang terenkripsi dapat didekode dengan benar oleh penerima, bahkan jika terjadi kesalahan dalam proses transmisi.

Matriks-matriks yang digunakan dalam kriptografi berbasis kode meliputi matriks generator dan matriks pemeriksaan paritas. Matriks generator, yang direpresentasikan sebagai G, merupakan matriks $k \times n$di mana k adalah panjang pesan asli dan n adalah panjang pesan yang terenkripsi. Matriks ini digunakan untuk menghasilkan pesan terenkripsi dengan menambahkan redundansi pada pesan asli. Di sisi lain, matriks pemeriksaan paritas, direpresentasikan sebagai H, adalah matriks $(n-k) \times n$ yang memastikan kesalahan dalam pesan terdeteksi dan diperbaiki saat diterima oleh penerima.

Proses enkripsi melibatkan aplikasi fungsi linear pada blok pesan asli untuk menghasilkan pesan terenkripsi. Di sisi penerima, proses dekripsi dilakukan untuk mengembalikan pesan asli dari pesan terenkripsi. Teknik-teknik dekripsi yang umum digunakan termasuk Dekripsi Daftar, Dekripsi Jarak Minimum, dan Dekripsi Sindrom. Masing-masing dari teknik ini dirancang untuk memperbaiki kesalahan dan memulihkan pesan asli dari pesan yang terganggu.

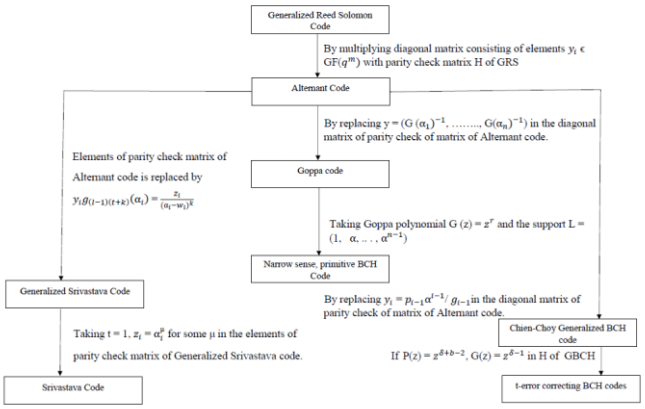
Secara keseluruhan, kriptografi berbasis kode linear memanfaatkan prinsip-prinsip teori pengkodean untuk menyediakan metode enkripsi dan dekripsi yang kuat dan andal, yang penting untuk melindungi keamanan komunikasi digital.

Kode kesalahan-koreksi telah menjadi fondasi yang penting dalam teori pengkodean, memiliki asal usulnya dalam kebutuhan untuk mengatasi kehilangan data yang disebabkan oleh gangguan dalam transmisi digital. Dua jenis utama dari kode kesalahan-koreksi adalah kode blok dan kode konvolusional. Dalam kode blok, data masukan dibagi menjadi blok-blok dengan panjang k digit, sementara dalam kode konvolusional, input dan outputnya adalah aliran kontinu digit. Operasi pada kode, seperti puncturing, extending, shortening, lengthening, expurgating, atau augmenting, digunakan untuk menyesuaikan panjang kode sesuai dengan batasan sistem. Kode kesalahan-koreksi linear memiliki properti khusus yang harus dipenuhi, tergantung pada metrik kode yang digunakan, seperti metrik Hamming, metrik Rank, dan metrik Lee. Berbagai jenis kode, seperti kode Gabidulin dan kode Gabidulin Low Rank, memanfaatkan metrik Rank untuk mendeteksi dan memperbaiki kesalahan rank, memperluas aplikasi kode kesalahan-koreksi dalam kriptografi dan komunikasi digital. Interaksi kompleks antara berbagai jenis kode mencerminkan kompleksitas dalam penggunaan kode kesalahan-koreksi dalam konteks kriptografi modern, di mana metode pengkodean dan dekoding kode digunakan dalam skema enkripsi dan tanda tangan digital untuk memastikan keamanan komunikasi digital.



Gambar 7. 1 Skema BCH code

McEliece dan Niederreiter adalah dua algoritma kriptografi awal yang dikembangkan dalam kriptografi berbasis kode. Algoritma McEliece pertama kali dibuat pada tahun 1978 menggunakan kode Goppa biner dengan parameter [n,k] = [1024, 524]. Meskipun beberapa varian McEliece yang menggunakan kode linear lainnya telah dikembangkan, varian-varian tersebut terbukti rentan terhadap serangan. Hanya McEliece yang menggunakan Kode Goppa Biner yang terbukti tahan terhadap serangan kuantum hingga saat ini, sehingga dipilih untuk putaran ketiga standardisasi oleh NIST. McEliece memiliki kompleksitas kuadrat dalam panjang blok dan tidak ada algoritma kuantum waktu polinomial yang ditemukan untuk mendekode kode blok linear umum. Algoritma ini juga mencakup elemen acak dalam setiap enkripsi melalui penggunaan vektor kesalahan yang dihasilkan secara acak, yang merupakan keunggulan utama McEliece. Namun, ukuran kunci yang besar menjadi keterbatasan utama algoritma ini. Algoritma kriptografi Niederreiter, yang dikembangkan pada tahun 1986, sangat mirip dengan McEliece tetapi mengenkripsi pesan menggunakan matriks pemeriksaan paritas, bukan matriks generator seperti dalam McEliece, dan menggunakan Kode Reed–Solomon tergeneralisasi.



Gambar 7.2 Skema Algoritma McElice

Skema tanda tangan berbasis kode linear telah dikembangkan berdasarkan pendekatan FDH (full domain hash) oleh Courtois-Finiasz-Sendrier (dikenal sebagai CFS), yang menggunakan Kode Goppa. Skema tanda tangan CFS yang dimodifikasi—mCFS—dikembangkan oleh Dallot. Skema tanda tangan berbasis Transformasi Fiat-Shamir pada skema identifikasi tanpa pengetahuan telah dikembangkan oleh Stern et al., Jain et al., dan Cayrel et al. Namun, tidak ada skema tanda tangan berbasis kode yang masuk dalam daftar pendek NIST untuk putaran ketiga standardisasi. Skema tanda tangan dengan nama RankSign dalam pengaturan metrik peringkat diusulkan, tetapi diserang dengan serangan pemulihan kunci struktural pada tahun 2018. Skema Tanda Tangan Berbasis Kode Acak (RaCoSS) diserahkan ke NIST, tetapi diserang dua hari setelah pengajuan dan meskipun telah diperbaiki, kembali diserang pada tahun yang sama. Masalah utama skema ini adalah berat dari tanda tangan yang valid besar. Pada tahun yang sama, Persichetti menyesuaikan skema Lyubashevsky dengan kode metrik Hamming kuasi-siklik acak, tetapi juga diserang dalam dua karya independen berikutnya. Pada tahun 2019, Anguil et al. mengusulkan skema tanda tangan dengan nama Durandal dalam konteks metrik peringkat, dengan keamanan berdasarkan masalah baru PSSI+. Pada tahun yang sama, skema tanda tangan Wave diusulkan, yang mengikuti kerangka hash-and-sign dengan keamanan berdasarkan asumsi baru bahwa kode generalized (U,U+V) independen dari kode linear acak. Meskipun terbukti bahwa tanda tangan keluarannya independen dari kunci rahasia, kesulitan dalam membedakan kode generalized (U,U+V) dari kode linear acak masih belum jelas. Kesimpulannya, pekerjaan yang ada pada skema tanda tangan berbasis kode yang aman membangun keamanannya pada asumsi-asumsi intractability yang belum matang.

	Stern	Jain et al.	Cayrel et al.
Keygen	0.0170 ms	0.0201 ms	0.339 ms
Sign	31.5 ms	16.5 ms	24.3 ms
Verify	2.27 ms	135 ms	9.81 ms
sk	1.24 bits	1536 bits	1840 bits
pk	512 bits	1024 bits	920 bits
System prams	65.5 kB	65.5 kB	229 kB
Signature	245 kB	263 kB	229 kB

Gambar 7.3 Skema perbandingan tanda tangan digital berbasis kode

Berikut implementasi kriptografi McElice.

```
import numpy as np

# Helper functions

def random_binary_matrix(rows, cols):
    return np.random.randint(2, size=(rows, cols))

def generate_goppa_code(n, k):
    # Simple example of generating a Goppa code
    return random_binary_matrix(k, n)

def invertible_matrix(k):
    while True:
        matrix = random_binary_matrix(k, k)
        if np.linalg.det(matrix) % 2 != 0:
            return matrix

def perm_matrix(n):
    perm = np.random.permutation(n)
    P = np.zeros((n, n), dtype=int)
    for i in range(n):
        P[i, perm[i]] = 1
    return P

def mod2_matrix_mult(A, B):
    return np.mod(np.dot(A, B), 2)

# Key generation

def mc_eliece_keygen(n, k):
    G = generate_goppa_code(n, k)
    S = invertible_matrix(k)
    P = perm_matrix(n)
    G_hat = mod2_matrix_mult(mod2_matrix_mult(S, G), P)
    return G_hat, S, P

# Encryption

def mc_eliece_encrypt(G_hat, message, t):
    n = G_hat.shape[1]
    error_vector = np.zeros(n, dtype=int)
    error_positions = np.random.choice(n, t, replace=False)
    error_vector[error_positions] = 1
    ciphertext = mod2_matrix_mult(message, G_hat) ^ error_vector
    return ciphertext, error_vector

# Decryption

def mc_eliece_decrypt(ciphertext, S, P, error_vector):
    S_inv = np.linalg.inv(S).astype(int) % 2
    P_inv = np.linalg.inv(P).astype(int) % 2
    decoded_message = mod2_matrix_mult(ciphertext ^ error_vector, P_inv)
    decoded_message = mod2_matrix_mult(decoded_message, S_inv)
    return decoded_message

# Example usage

n = 7
k = 4
t = 1

G_hat, S, P = mc_eliece_keygen(n, k)

message = np.array([1, 0, 1, 1])
ciphertext, error_vector = mc_eliece_encrypt(G_hat, message, t)

decrypted_message = mc_eliece_decrypt(ciphertext, S, P, error_vector)

print(f"Original message: {message}")
print(f"Encrypted message: {ciphertext}")
print(f"Decrypted message: {decrypted_message}")
```

Algoritma ini mencakup pembuatan kunci publik dan privat, serta enkripsi dan dekripsi pesan. Pada bagian awal, terdapat beberapa fungsi bantu seperti random_binary_matrix untuk membuat matriks biner acak, generate_goppa_code untuk menghasilkan kode Goppa sederhana, invertible_matrix untuk membuat matriks biner acak yang dapat dibalik, perm_matrix untuk membuat matriks permutasi, dan mod2_matrix_mult untuk melakukan perkalian matriks di bawah mod 2. Dalam pembuatan kunci (mc_eliece_keygen), matriks generator G untuk kode Goppa dibuat, bersama dengan matriks biner acak yang dapat dibalik S dan matriks permutasi P. Kunci publik G_hat kemudian dihitung sebagai G hat=S×G×P. Untuk enkripsi (mc_eliece_encrypt), pesan dienkripsi dengan mengalikan matriks generator G_hat dan menambahkan vektor error acak. Pada dekripsi (mc_eliece_decrypt), ciphertext dikoreksi dengan vektor error dan dikalikan dengan invers matriks permutasi P dan matriks S. Contoh penggunaan menunjukkan pembuatan kunci publik dan privat, enkripsi pesan, dan dekripsi pesan yang terenkripsi untuk memverifikasi integritas proses. Implementasi ini adalah representasi sederhana dari algoritma McEliece, yang dalam aplikasi nyata memerlukan kode Goppa yang lebih kompleks dan mekanisme yang lebih kuat untuk menangani berbagai aspek keamanan dan efisiensi.

VIII. SPHINCS+(KRIPTOGRAFI KUANTUM BERBASIS HASH)

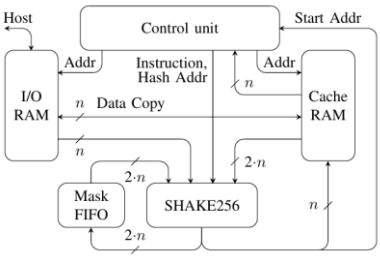
SPHINCS+ adalah skema tanda tangan umum yang dapat digabungkan dengan fungsi hash apa pun. Beberapa parameter memungkinkan penyesuaian antara usaha komputasi, ukuran tanda tangan, dan margin keamanan. Penulis menyediakan enam set parameter untuk keamanan dan kinerja, dengan opsi 'sederhana' dan 'kuat', serta tiga pilihan fungsi hash, menghasilkan 36 varian SPHINCS+. Semua

varian ini memiliki ukuran kunci kecil dan ukuran tanda tangan sedang (8 hingga 50 kbytes). Skema tanda tangan berbasis hash biasanya dapat dibuktikan aman selama fungsi hash yang mendasarinya dianggap aman. SPHINCS+ juga merupakan skema tanda tangan tanpa status, menjadikannya kandidat pengganti yang menjanjikan untuk skema tanda tangan yang banyak digunakan saat ini.

Kelemahan dari SPHINCS+ adalah waktu penandatanganannya. Laporan penulis menunjukkan latensi mulai dari 6,5 milidetik (sebagian dipercepat oleh perangkat keras) hingga beberapa detik pada prosesor 3,5GHz. Untuk meningkatkan throughput algoritma kriptografi, sering digunakan koprosesor khusus. SPHINCS-256, pendahulu SPHINCS+, adalah satu-satunya skema tanda tangan berbasis hash tanpa status yang diketahui memiliki akselerator berbasis perangkat keras. Alternatif lain untuk mempercepat penandatanganan SPHINCS-256 adalah menggunakan GPU yang kuat dan menghitung banyak tanda tangan secara paralel. Ada juga publikasi yang menyajikan koprosesor untuk skema tanda tangan berbasis hash dengan status, XMSS.

SPHINCS+ menggabungkan FORS dan pohon Merkle fraktal besar. Kunci privat SPHINCS+ adalah seed yang digunakan untuk menghasilkan semua nilai privat, dan kunci publiknya adalah simpul akar dari sub-pohon WOTS+ tertinggi. Tanda tangan dimulai dengan memilih alamat awal pseudo-acak untuk memilih pasangan kunci FORS yang menandatangani digest pesan. Kunci publik dari pasangan kunci FORS dipakai untuk menandatangani pohon Merkle fraktal. SPHINCS+ menggunakan tiga fungsi hash: SHA-256, SHAKE256, dan Haraka, dengan variasi parameter dan versi robust serta simple.

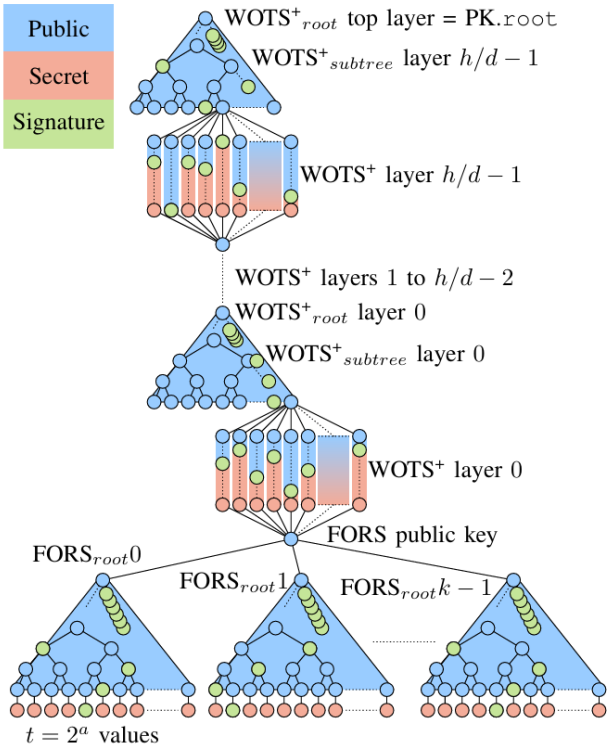
Kelebihan SPHINCS+ adalah sifatnya yang stateless, namun memerlukan usaha pemrosesan yang lebih besar dan ukuran tanda tangan yang lebih besar. Parameter dan fungsi hash yang berbeda memberikan pilihan dalam hal upaya komputasi, ukuran tanda tangan, dan margin keamanan.



Gambar 8.1 Arsitektur SPHINC+

Arsitektur SPHINCS+ meliputi:

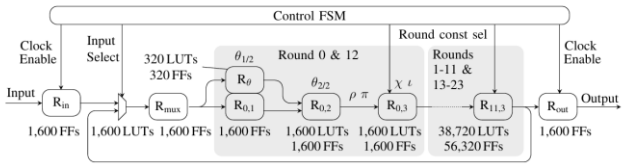
- Unit kontrol sebagai state machine yang menghasilkan instruksi internal.
- SHAKE256 sebagai kalkulator nilai hash pipelined.
- I/O RAM untuk memori kunci dan tanda tangan.
- Cache RAM untuk hasil sementara.
- Mask FIFO untuk buffer masker (hanya untuk SPHINCS+-robust).



Gambar 8.2 Skema implementasi SPHINCS+

SHAKE256 merupakan bagian dari standar NIST SHA-3 dan berfungsi sebagai fungsi keluaran yang dapat diperluas (XOF) dengan panjang keluaran arbitrer. Inti dari semua fungsi SHA-3 adalah permutasi KECCAK 1600-bit yang dipanggil 24 kali selama eksekusi SHAKE256. Berbagai implementasi SHA-3 berbasis FPGA memiliki tujuan desain yang berbeda, dengan perbedaan utama pada throughput yang ditargetkan dan jumlah sumber daya yang diperlukan. Implementasi ini dikategorikan ke dalam beberapa kelompok: Basic, Folded, Pipelined, dan Unrolled. Untuk performa optimal dari inti SPHINCS+, struktur Unrolled dipilih, karena memiliki throughput tertinggi. Komputasi putaran KECCAK diimplementasikan dalam struktur pipeline yang dijalankan dua kali per siklus hash, mengurangi kedalaman pipeline dari 24 menjadi 12 putaran dan menghemat hampir 50% sumber daya FPGA.

Pipeline KECCAK beroperasi pada kecepatan clock ganda dibandingkan dengan bagian inti lainnya, memungkinkan pemrosesan satu input dan output hash setiap siklus clock. Untuk mencapai hal ini, frekuensi clock pipeline KECCAK dipilih sebagai kelipatan integer (faktor 2) dari frekuensi clock utama. Inti SHAKE256 dibangun di sekitar pipeline KECCAK dan mengandung logika untuk format input serta menangani kasus ketika permutasi KECCAK dipanggil beberapa kali. Ini terjadi saat input atau output lebih besar dari laju SHAKE 1088 bit (136 byte), seperti pada saat penandatanganan dan verifikasi SPHINCS+ ketika 67 nilai kunci publik WOTS+ dikompresi menjadi satu daun pohon.



Gambar 8.3 Skema Keccak Pipeline

Implementasi dalam bentuk kode sebagai berikut:

```
def shake256(data, length):
    shake = hashlib.shake_256()
    shake.update(data)
    return shake.digest(length)

def generate_wots_keypair(seed):
    sk = shake256(seed, 32)
    pk = shake256(sk, 32)
    return sk, pk

def generate_tree_leaf(seed, address):
    return shake256(seed + address, 32)

def build_merkle_tree(leaves):
    tree = [leaves]
    while len(tree[-1]) > 1:
        level = []
        for i in range(0, len(tree[-1]), 2):
            combined = tree[-1][i]
            if i + 1 < len(tree[-1]):
                combined += tree[-1][i + 1]
            level.append(shake256(combined, 32))
        tree.append(level)
    return tree

def sign_message(message, seed):
    sk, pk = generate_wots_keypair(seed)
    leaves = [generate_tree_leaf(seed, i.to_bytes(4, 'little'))
for i in range(16)]
    tree = build_merkle_tree(leaves)
    message_hash = shake256(message, 32)
    signature = shake256(sk + message_hash, 32)
    return signature, tree[-1][0]

def verify_signature(message, signature, pk, merkle_root):
    message_hash = shake256(message, 32)
    expected_signature = shake256(pk + message_hash, 32)
    return signature == expected_signature and
shake256(signature, 32) == merkle_root
```

Kode ini mengimplementasikan beberapa fungsi dasar untuk skema tanda tangan digital SPHINCS+ menggunakan fungsi hash SHAKE256 dari library hashlib. Fungsi shake256 menerima data dan panjang output, mengembalikan nilai hash dengan panjang yang ditentukan menggunakan SHAKE256. Fungsi generate_wots_keypair menghasilkan pasangan kunci untuk WOTS+ (Winternitz One-Time Signature) dari seed yang diberikan, dimana kunci pribadi (sk) dan kunci publik (pk) masing-masing dihasilkan dengan meng-hash seed dan kunci pribadi menggunakan SHAKE256. Fungsi generate_tree_leaf menghasilkan daun pohon Merkle dari gabungan seed dan alamat. Selanjutnya, fungsi build_merkle_tree membangun pohon Merkle dari daftar daun, dengan menggabungkan dan meng-hash pasangan daun secara berulang hingga tersisa satu root. Fungsi sign_message menandatangani pesan dengan terlebih dahulu menghasilkan pasangan kunci WOTS+ dan daun pohon Merkle, kemudian membangun pohon Merkle dan menghasilkan tanda tangan dengan meng-hash gabungan kunci pribadi dan hash pesan. Fungsi ini mengembalikan tanda tangan dan root pohon Merkle. Terakhir, fungsi verify_signature memverifikasi tanda tangan dengan membandingkan tanda tangan yang diberikan dengan yang diharapkan, serta memastikan bahwa hash dari tanda tangan sesuai dengan root pohon Merkle yang diberikan.

IX. SIDH (KRIPTOGRAFI KUANTUM BERBASIS ISOGENY)

SIDH, atau Supersingular Isogeny Diffie-Hellman, adalah sebuah protokol pertukaran kunci kriptografi kuantum yang didasarkan pada isogeni kurva eliptik supersingular. Protokol ini dikembangkan sebagai respons terhadap kemampuan komputer kuantum untuk menyelesaikan algoritma faktorisasi Shor, yang dapat digunakan untuk merusak kunci-kunci kriptografi klasik yang umum digunakan saat ini.

Berikut adalah penjelasan mengenai beberapa aspek utama dari SIDH:

- Basis Matematika: SIDH menggunakan matematika isogeni untuk menghasilkan kunci rahasia dan berkomunikasi antara dua entitas. Ini melibatkan kurva eliptik supersingular, yang memiliki struktur matematika tertentu yang membuatnya cocok untuk kriptografi kuantum.
- Key Generation: Pada tahap ini, pasangan kunci rahasia dan kunci publik dibangkitkan. Kunci rahasia terdiri dari titik-titik yang dipilih secara acak pada kurva eliptik supersingular, sedangkan kunci publik terdiri dari titik-titik yang dihasilkan dari isogeni yang dihitung berdasarkan kunci rahasia.
- Key Exchange: Protokol pertukaran kunci dilakukan dengan dua entitas yang ingin berkomunikasi secara rahasia. Mereka menggunakan kunci-kunci publik yang dihasilkan pada tahap sebelumnya untuk berbagi informasi rahasia dan mendapatkan kunci rahasia bersama.
- Keamanan: SIDH dirancang untuk menawarkan keamanan terhadap serangan kriptografi kuantum, terutama serangan yang berbasis pada algoritma

Shor. Namun, keamanan SIDH masih menjadi subjek penelitian aktif, karena terus berkembangnya pemahaman tentang kemampuan komputer kuantum.

- Efisiensi: Efisiensi SIDH dalam hal penggunaan sumber daya komputasi menjadi faktor penting, terutama dalam implementasi perangkat keras dan perangkat lunak. Banyak penelitian berfokus pada meningkatkan efisiensi SIDH agar dapat diterapkan dalam skenario yang lebih luas.
- SIDH adalah salah satu dari beberapa protokol kriptografi kuantum yang sedang dikembangkan untuk menghadapi tantangan dari kemajuan komputer kuantum. Dengan mengandalkan sifat-sifat unik dari kurva eliptik supersingular dan isogeni, SIDH menawarkan pendekatan yang menarik untuk pertukaran kunci aman dalam era komputasi kuantum.

SIDH mirip dengan protokol Diffie-Hellman klasik dalam beberapa hal:

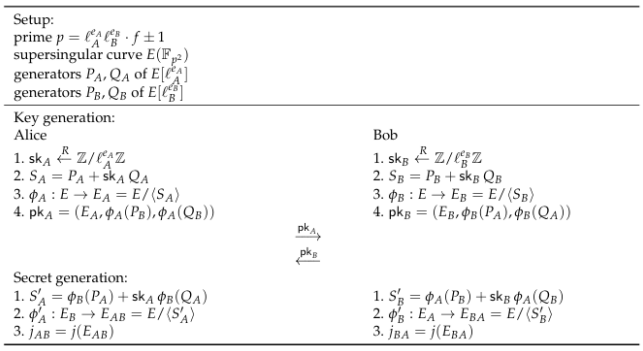
1. Inisialisasi Key: Dua pihak, Alice dan Bob, secara independen menghasilkan kunci rahasia mereka sendiri, mA dan skB, serta kunci publik mereka, pkA dan pkB. Hal ini mirip dengan cara pada protokol Diffie-Hellman klasik, di mana setiap pihak memiliki kunci privat dan publik mereka sendiri.

2. Perhitungan Isogeni Rahasia: Setelah mendapatkan kunci publik lawan, Alice dan Bob menghitung isogeni rahasia mereka sendiri, yaitu isogeni dari kurva eliptik awal ke kurva eliptik yang telah dimodifikasi dengan menggunakan kunci rahasia mereka. Ini memungkinkan mereka untuk memperoleh subgrup rahasia yang berbeda pada kurva eliptik lawan. Proses ini dapat dibandingkan dengan perhitungan nilai bersama pada protokol Diffie-Hellman klasik.

3. Perhitungan Kunci Bersama: Dengan menggunakan subgrup rahasia yang telah dihitung, setiap pihak kemudian menghitung isogeni rahasia lainnya, yang akhirnya menghasilkan kurva eliptik yang sama di kedua sisi. Nilai j-invariant dari kurva ini kemudian dijadikan sebagai kunci bersama. Proses ini analog dengan cara nilai bersama dihasilkan pada protokol Diffie-Hellman klasik.

4. Pergerakan di Graf Isogeni: Selama perhitungan, baik Alice maupun Bob bergerak di dalam graf isogeni, mulai dari kurva eliptik awal hingga mencapai kurva yang sama pada akhirnya. Ini mencerminkan perjalanan nilai bersama di grafik grup pada protokol Diffie-Hellman klasik.

Meskipun ada perbedaan detail dalam mekanisme dan matematika di balik SIDH dan Diffie-Hellman klasik, kedua protokol tersebut memiliki banyak kesamaan dalam prinsip dasar pertukaran kunci rahasia. Ini memungkinkan SIDH untuk diimplementasikan dengan cara yang serupa dengan protokol Diffie-Hellman klasik, tetapi dengan keamanan tambahan terhadap serangan kriptografi kuantum.



Gambar 9.1 Skema SIDH

Implementasi kodenya sebagai berikut

```
import random

class Entity:
   def __init__(self, name, params, get_other, isogeny_graph_walk, E):
      self.name = name
      self.P = params[name][0]
      self.Q = params[name][1]
      self.l = params[name][2]
      self.e = params[name][3]
      self.sk = random.randrange(self.l ** self.e)
      self.S = self.P + self.sk * self.Q
      # assert self.l ** self.e == self.S.order()
      self.pk = self.gen_pub_key(get_other(self.name))

   def gen_pub_key(self, other):
      return isogeny_graph_walk(E, self.S, self.l, self.e, other[0], other[1])

   def gen_shared_key(self, peer):
      S = peer.pk[1] + self.sk * peer.pk[2]
      shared_curve, _, _ = isogeny_graph_walk(peer.pk[0], S, self.l, self.e)
      return shared_curve.j_invariant()
```

Kelas Entity dalam kode Python di atas mewakili entitas dalam protokol SIDH. Saat objek Entity dibuat, konstruktor __init__ dijalankan, menerima beberapa parameter termasuk name (nama entitas), params (parameter yang disediakan dalam bentuk kamus), get_other (fungsi untuk mendapatkan kunci publik entitas lain),

isogeny_graph_walk (fungsi untuk melakukan langkah-langkah dalam grafik isogeni), dan E (kurva eliptik yang digunakan). Dalam konstruktor ini, atribut-atribut entitas seperti P, Q, l, e, sk, S, dan pk diinisialisasi berdasarkan nilai dari parameter. Metode gen_pub_key digunakan untuk menghasilkan kunci publik entitas dengan memanggil fungsi isogeny_graph_walk dengan parameter yang sesuai. Metode gen_shared_key digunakan untuk menghasilkan kunci bersama dengan entitas lain, di mana nilai S dihitung berdasarkan kunci publik entitas lain dan kunci rahasia sendiri, kemudian memanggil fungsi isogeny_graph_walk untuk menghasilkan kunci bersama, yang kemudian dikembalikan sebagai nilai kunci bersama. Dengan demikian, kelas Entity memberikan struktur untuk pembangkitan kunci rahasia dan kunci bersama dalam protokol SIDH antara dua entitas.

REFERENSI

[1] Yaser Baseri, Vikas Chouhan, Ali Ghorbani (2024) Cybersecurity in the Quantum Era: Assessing the Impact of Quantum Computing on Infrastructure

[2] E. S. Malygina, A. V. Kutsenko, A. Novoselov, N. S. Kolesnikov, A. O. Bakharev, I. S. Khilchuk, A. S. Shaporenko, N. N. Tokareva (2023). Post-Quantum Cryptosystems: Open Problems and Solutions. Lattice-Based Cryptosystems

[3] Bartosz Drzazga, Łukasz Krzywiecki (2022) Review of Chosen Isogeny-Based Cryptographic Schemes

[4] Chithralekha Balamurugan 1, Kalpana Singh 2, ,Ganeshvani Ganesan 1, Muttukrishnan Rajarajan (2021) Post-Quantum and Code-Based Cryptography—Some Prospective Research Directions

[5] Dorian Amiet, Lukas Leuenberger, Andreas Curigery and Paul Zbinden (2020) FPGA-based SPHINCS+ Implementations: Mind the Glitch