PERBANDINGAN ALGORITMA RSA DAN MERKLE-HELLMAN DALAM RANCANG BANGUN APLIKASI PENYANDIAN PESAN TEKS

Disusun dan Diajukan untuk Memenuhi Persyaratan Ujian Akhir Memperoleh
Gelar Sarjana Komputer pada Fakultas Sains dan Teknologi
Universitas Pembangunan Panca Budi
Medan

SKRIPSI

OLEH:

NAMA	:	**KARISA ENDAH WIDIYASARI**
NPM	:	**1814370155**
PROGRAM STUDI	:	**SISTEM KOMPUTER**

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
2022

Halaman Pengesahan

PENGESAHAN TUGAS AKHIR

JUDUL : PERBANDINGAN ALGORITMA RSA DAN MERKLE-HELLMAN DALAM RANCANG BANGUN APLIKASI PENYANDIAN PESAN TEKS

NAMA : KARISA ENDAH WIDIYASARI
N.P.M : 1814370155
FAKULTAS : SAINS & TEKNOLOGI
PROGRAM STUDI : Sistem Komputer
TANGGAL KELULUSAN : 29 Oktober 2022

DIKETAHUI

DEKAN

Hamdani, ST., MT.

KETUA PROGRAM STUDI

Eko Hariyanto, S.Kom., M.Kom

DISETUJUI
KOMISI PEMBIMBING

PEMBIMBING I

Andysah Putera Utama Siahaan, S.Kom., M.Kom., Ph.D.

PEMBIMBING II

Chairul Rizal, S.Kom., M.M.S.I.

Note : dokumen ini telah diproses secara online melalui sistem informasi akademik UNPAB

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Karisa Endah Widiyasari

NPM : 1814370155

Prodi : Sistem Komputer

Judul Skripsi : Perbandingan Algoritma RSA dan Merkle-Hellman dalam Rancang Bangun Aplikasi Penyandian Pesan Teks

Dengan ini menyatakan bahwa :

1. Tugas Akhir/Skripsi saya bukan hasil plagiat.
2. Saya tidak akan menuntut perbaikan nilai indeks prestasi (IPK) setelah ujian siding meja hijau.
3. Skripsi saya dapat di publikasikan oleh pihak lembaga dan saya tidak akan menuntut akibat publikasi tersebut.

Demikian pernyataan ini saya buat dengan sebenar-benarnya, Terima kasih.

Medan, 01 Desember 2022

Yang membuat pernyataan

Karisa Endah Widiyasari

SURAT ORISINALITAS

Dengan ini menyatakan bahwa dalam skripsi ini tidak terdapat karya yang diajukan untuk memperoleh gelar kesarjanaan didalam perguruan tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis di acu dalam skripsi ini dan disebutkan dalam daftar pustaka.

Medan, 01 Desembeer 2022

Yang membuat pernyataan

Karisa Endah Widiyasari

ABSTRAK

KARISA ENDAH WIDIYASARI
Perbandingan Algoritma RSA Dan Merkle-Hellman Dalam Rancang Bangun Aplikasi Penyandian Pesan Teks
2022

Setiap data perlu diamankan dengan baik. Untuk itu diperlukan algoritma dalam melakukan pengamanan. Ada dua buah algoritma kriptografi modern yang akan dibahas pada penelitian ini yaitu algoritma RSA dan Merkle-Hellman. Penelitian ini akan mencoba membandingkan kecepatan dari kedua algoritma ini dalam memproses *plaintext* dan *ciphertext.* Hasil enkripsi dapat terlihat bahwa algoritma RSA memiliki kecepatan yang lebih baik dari Merkle-Hellman. Hal ini dapat dilihat dari hasil pengukuran dalam melakukan enkripsi dan dekripsi pesan. Pengukuran menunjukkan algoritma Merkle-Hellman membutuhkan waktu delapan kali lebih lama daripada algoritma RSA.

Kata Kunci: *keamanan, enkripsi, pesan, RSA, Merkle-Hellman*

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT, karena dengan berkat dan rahmat-Nya penulis masih diberikan kesempatan untuk menyelesaikan skripsi ini sebagaimana mestinya. Skripsi ini berjudul "**PERBANDINGAN ALGORITMA RSA DAN MERKLE-HELLMAN DALAM RANCANG BANGUN APLIKASI PENYANDIAN PESAN TEKS**". Dalam kesempatan ini, penulis mengucapkan rasa terima kasih yang tak terhingga kepada pihak-pihak yang telah membantu dalam penyelesaian skripsi ini. Penulis ingin mengucapkan terima kasih kepada :

1. Orang tua saya yang selalu memberikan semangat, dukungan dan motivasi dalam penyusunan skripsi ini.
2. Bapak Dr. H. Muhammad Isa Indrawan, S.E., M.M., selaku Rektor Universitas Pembangunanan Panca Budi Medan.
3. Bapak Hamdani, S.T., M.T., selaku Dekan Fakultas Sains dan Teknologi Universitas Pembangunan Panca Budi Medan.
4. Bapak Eko Hariyanto, S.Kom., M.Kom., selaku Ketua Program Studi Sistem Komputer Universitas Pembangunan Panca Budi Medan.
5. Bapak Andysah Putera Utama Siahaan, S.Kom., M.Kom., Ph.D., selaku Dosen Pembimbing I yang telah memberikan arahan dan membimbing dalam penyelesaian skripsi ini.
6. Bapak Chairul Rizal, S.Kom., M.M.S.I., selaku Dosen Pembimbing II yang telah memberikan ilmu pengetahuan, serta bimbingan dalam penyelesaian skripsi ini.
7. Dosen-dosen pada Program Studi Sistem Komputer Fakultas Sains dan Teknologi Universitas Pembangunan Panca Budi Medan.
8. Staff dan karyawan pada Universitas Pembangunan Panca Budi Medan.
9. Seluruh teman-teman penulis dari program studi Sistem Komputer, Fakultas Sains dan Teknologi, Universitas Pembangunan Panca Budi, Medan

Penulis juga menyadari bahwa penyusunan skripsi ini belum mendapatkan kesempurnaan dalam segi penulisan ataupun isi. Hal ini disebabkan pengetahuan penulis yang sangat terbatas. Penulis sangat mengharapkan adanya kritik dan saran dari pembaca untuk dapat memperbaiki isi skripsi.

Medan, 01 November 2022
Penulis

Karisa Endah Widiyasari
1814370155

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Keamanan komputer penting untuk dilakukan agar informasi yang akan digunakan menjadi aman dan tidak dapat disalahgunakan. Keamanan perlu diterapkan kepada pesan, terlebih-lebih pesan teks yang memiliki format *plaintext* atau format yang sama pada *American Standard Code for Information Interchange (ASCII)*, sehingga pesan ini dapat dibuka oleh editor apapun tanpa terkecuali. Pesan teks perlu diamankan dengan baik agar terhindar dari pencurian. Pesan teks ini biasanya pesan-pesan yang memiliki informasi singkat seperti pin *ATM*, *username*, *password*, dan informasi-informasi lainnya.

Kriptografi modern adalah salah satu metode yang dapat digunakan dalam mengamankan pesan. Ada beberapa algoritma yang termasuk dalam kriptografi modern yang dapat digunakan untuk mengamankan pesan teks yang pendek. Permasalahan yang terjadi adalah kecepatan suatu algoritma dalam melakukan proses enkripsi dan dekripsi pada pesan. Kekuatan kriptografi modern adalah terletak pada kunci dekripsi yang besar sehingga dapat mempengaruhi kecepatan kerja dari proses dekripsi.

Penulis menggunakan dua buah algoritma kriptografi modern yaitu *Rivest Shamir Adleman* (*RSA)* dan *Merkle-Hellman*. Kedua algoritma ini berjenis asimetris dimana kunci enkripsi dan dekripsi yang digunakan pada saat proses transformasi *plaintext* ke *ciphertext* dan sebaliknya adalah berbeda sehingga

membutuhkan perhitungan matematika yang rumit. Algoritma ini menggunakan *modulo power* yang biasanya menggunakan tipe bilangan berjenis *big integer*. *Big integer* merupakan bilangan bulat dengan kemampuan yang besar yang biasanya melebihi dari tipe data *integer* dan *long integer*. Semakin tinggi bilangan yang digunakan dalam proses enkripsi dan dekripsi, maka semakin aman data tersebut.

Penulis ingin membandingkan kedua algoritma tersebut untuk melihat proses mana yang lebih unggul pada saat digunakan pada pesan berjenis teks. Hasil pengukuran menggunakan kecepatan komputer pada spesifikasi yang sama sehingga dapat dilihat algoritma mana yang lebih baik dalam proses ini. Berdasarkan latar belakang yang sudah dijelaskan, maka penulis mengambil judul "**Perbandingan Algoritma RSA Dan Merkle-Hellman Dalam Rancang Bangun Aplikasi Penyandian Pesan Teks**".

1.2 Rumusan Masalah

Adapun rumusan masalah yang digunakan dalam penulisan skripsi ini adalah sebagai berikut:

1. Bagaimana membangun aplikasi kriptografi menggunakan algoritma RSA dan Merkle-Hellman?
2. Bagaimana mengukur kecepatan pada algoritma RSA dan Merkle-Hellman?
3. Bagaimana melakukan proses enkripsi dan dekripsi pada algoritma RSA dan Merkle-Hellman?

1.3 Batasan Masalah

Adapun batasan masalah yang digunakan dalam penulisan skripsi ini adalah sebagai berikut:

1. Data yang digunakan merupakan pesan teks singkat bertipe "*.txt"*.
2. Panjang pesan yang dapat diproses adalah 1024 karakter.
3. Bahasa pemrograman yang digunakan adalah Microsoft Visual Basic.NET.
4. Sistem yang dibangun berbasis *desktop*.

1.4 Tujuan Penelitian

Adapun tujuan penelitian yang digunakan dalam penulisan skripsi ini adalah sebagai berikut:

1. Untuk membangun aplikasi kriptografi menggunakan algoritma RSA dan Merkle-Hellman.
2. Untuk mengukur kecepatan pada algoritma RSA dan Merkle-Hellman?
3. Untuk melakukan proses enkripsi dan dekripsi pada algoritma RSA dan Merkle-Hellman.

1.5 Manfaat Penelitian

Adapun manfaat penelitian yang digunakan dalam penulisan skripsi ini adalah sebagai berikut:

1. Mengamankan pesan teks agar terhindar dari pencurian data.
2. Mendapatkan hasil kinerja algoritma *RSA* dan *Merkle-Hellman*.
3. Menambah ilmu pengetahuan kepada penulis.

BAB II

LANDASAN TEORI

2.1 Pesan

Pesan adalah dokumen perjalanan yang dikeluarkan oleh pemerintah suatu negara kepada warganya yang memverifikasi identitas dan kewarganegaraan pemegang untuk tujuan perjalanan internasional. Pesan adalah buklet kecil yang biasanya berisi nama pembawa, tempat lahir, tanggal lahir, tanggal penerbitan, tanggal kadaluwarsa, nomor pesan, foto dan tanda tangan. Ada beberapa jenis pesan tergantung dari status pengembannya di negara asalnya (Arton, 2021).

Pesan adalah dokumen perjalanan, biasanya dikeluarkan oleh pemerintah suatu negara kepada warganya, yang menyatakan identitas dan kewarganegaraan pemiliknya terutama untuk tujuan perjalanan internasional. Pesan standar mungkin berisi informasi seperti nama pemegang, tempat dan tanggal lahir, foto, tanda tangan, dan informasi identitas terkait lainnya.

Banyak negara telah mulai menerbitkan atau berencana menerbitkan pesan biometrik yang berisi microchip tertanam, menjadikannya dapat dibaca oleh mesin dan sulit dipalsukan. Pada Januari 2019, ada lebih dari 150 yurisdiksi yang menerbitkan e-pesan. Pesan non-biometrik yang dapat dibaca mesin yang diterbitkan sebelumnya biasanya tetap berlaku hingga tanggal kedaluwarsa masing-masing.

Seorang pemegang pesan biasanya berhak memasuki negara yang mengeluarkan pesan, meskipun beberapa orang yang berhak atas pesan mungkin bukan warga negara penuh dengan hak tinggal (misalnya warga negara Amerika atau warga negara Inggris). Pesan tidak dengan sendirinya menciptakan hak apa pun di negara yang dikunjungi atau mewajibkan negara penerbit dengan cara apa pun, seperti memberikan bantuan konsuler. Beberapa pesan membuktikan bahwa pemegang pesan memiliki status sebagai diplomat atau pejabat lainnya, yang berhak atas hak dan hak istimewa seperti kekebalan dari penangkapan atau penuntutan.

Banyak negara biasanya mengizinkan masuknya pemegang pesan dari negara lain, terkadang membutuhkan visa juga untuk diperoleh, tetapi ini bukan hak otomatis. Banyak kondisi tambahan lainnya, seperti tidak mungkin menjadi tuntutan publik karena alasan keuangan atau lainnya, dan pemegang yang tidak dihukum karena kejahatan, dapat berlaku. Jika suatu negara tidak mengakui negara lain, atau sedang berselisih dengannya, negara tersebut dapat melarang penggunaan pesan mereka untuk perjalanan ke negara lain tersebut, atau mungkin melarang masuknya pemegang pesan negara lain tersebut, dan terkadang kepada orang lain yang memiliki, untuk Misalnya, mengunjungi negara lain. Beberapa individu dikenakan sanksi yang melarang mereka masuk ke negara tertentu.

Beberapa negara dan organisasi internasional mengeluarkan dokumen perjalanan yang bukan merupakan pesan standar, tetapi memungkinkan pemegangnya untuk melakukan perjalanan internasional ke negara-negara yang mengakui dokumen tersebut. Misalnya, orang tanpa kewarganegaraan biasanya

tidak diberi pesan nasional, tetapi mungkin bisa mendapatkan dokumen perjalanan pengungsi atau "pesan Nansen" sebelumnya yang memungkinkan mereka untuk melakukan perjalanan ke negara-negara yang mengakui dokumen tersebut, dan terkadang untuk kembali ke negara penerbit. .

Pesan dapat diminta dalam keadaan lain untuk mengonfirmasi identifikasi seperti check in di hotel atau saat menukar uang ke mata uang lokal. Pesan dan dokumen perjalanan lainnya memiliki tanggal kedaluwarsa, setelah itu tidak lagi dikenali, tetapi disarankan agar pesan berlaku setidaknya selama enam bulan karena banyak maskapai penerbangan menolak naik penumpang yang pesannya memiliki tanggal kedaluwarsa lebih pendek, bahkan jika negara tujuan mungkin tidak memiliki persyaratan seperti itu.

2.1.1 Jenis Pesan

Ada beberapa jenis pesan yang diterbitkan oleh setiap negara berdasarkan kategori dan fungsinya, antara lain:

1 Pesan Diplomatik

Pesan diplomatik diberikan kepada diplomat yang melakukan perjalanan bisnis resmi dan mewakili negara asal mereka di luar negeri. Diplomat ditunjuk oleh pemerintah untuk menjalankan bisnis resmi di luar negeri dan memelihara hubungan politik, ekonomi, dan sosial dengan negara lain. Pesan mereka biasanya memberi mereka hak dan kekebalan tertentu, seperti pembebasan dari tuntutan hukum dan pajak di negara tuan rumah.

2 Pesan darurat

Pesan darurat atau pesan sementara, dikeluarkan jika Anda kehilangan pesan atau dicuri dan Anda tidak punya waktu untuk mengajukan yang baru. Misalnya, jika Anda meninggalkan pesan di pesawat karena kecelakaan dan karena itu tidak dapat naik penerbangan lanjutan, Anda dapat menghubungi kedutaan dan diberikan pesan darurat satu arah yang akan mengantarkan Anda pulang.

3 Pesan Resmi

Pesan resmi, atau pesan dinas, adalah jenis pesan yang dikeluarkan untuk pegawai pemerintah. Tujuannya adalah agar pejabat di negara tujuan mengetahui bahwa pembawa tersebut memasuki negara tersebut untuk urusan resmi, mewakili negara mereka dalam kapasitas resmi. Pemegang pesan resmi biasanya tidak diberikan hak khusus

2.1.2 Mengapa Data Pesan Penting Diamankan

Pesan merupakan dokumen penting yang harus dijaga kerahasiaannya. Pada pesan, ada beberapa informasi dan kode yang tidak boleh diketahui oleh orang lain. Pesan dapat ditiru dan disalahgunakan oleh orang yang tidak bertanggung jawab. Keamanan pesan merupakan tanggung jawab dari pemegang pesan terlebih-lebih jika pesan telah dipindai dan dibagikan ke orang lain untuk keperluan administrasi.

2.1.3 Data kualitatif

Tidak seperti data kuantitatif, yang berkaitan dengan angka dan angka, data kualitatif lebih bersifat deskriptif daripada numerik. Data kualitatif biasanya tidak mudah diukur secara kuantitatif dan dapat diperoleh melalui observasi atau survei terbuka atau pertanyaan wawancara. Penelitian kualitatif kemungkinan besar akan memberikan jawaban atas pertanyaan seperti "mengapa?" dan bagaimana?". Seperti disebutkan, metode pengumpulan data kualitatif kemungkinan besar terdiri dari pertanyaan terbuka dan jawaban deskriptif dan sedikit atau tidak ada nilai numerik. Data kualitatif adalah cara terbaik untuk mendapatkan wawasan tentang pemikiran dan perilaku audiens (mungkin yang seseorang identifikasi menggunakan penelitian kuantitatif, tetapi tidak dapat menganalisis secara lebih rinci).

Data yang diperoleh dengan menggunakan metode pengumpulan data kualitatif dapat digunakan untuk menemukan ide-ide baru, peluang, dan masalah, menguji nilai dan keakuratannya, merumuskan prediksi, mengeksplorasi bidang tertentu secara lebih rinci, dan menjelaskan angka-angka yang diperoleh dengan menggunakan teknik pengumpulan data kuantitatif. Karena metode pengumpulan data kuantitatif biasanya tidak melibatkan angka dan perhitungan matematis tetapi lebih mementingkan kata-kata, bunyi, pikiran, perasaan, dan data yang tidak terukur lainnya, data kualitatif sering dianggap lebih subyektif, tetapi pada saat yang sama memungkinkan pemahaman yang lebih dalam. Beberapa teknik pengumpulan data kualitatif yang paling umum termasuk survei terbuka dan kuesioner, wawancara, kelompok fokus, observasi, studi kasus, dan sebagainya (Jovancic, 2019).

2.2 Keamanan Data

Teknologi digital sekarang hanya bagian dari kehidupan. Dari belanja online hingga perbankan bersih dan bisnis hingga infrastruktur pemerintah, teknologi digital memainkan peran penting. Terlepas dari berbagai keuntungan digitalisasi, serangan dunia maya adalah titik hitam. Dalam beberapa tahun terakhir, kami telah menyaksikan banyak serangan dunia maya tingkat tinggi. Bahkan, kita dapat mengatakan bahwa jumlah serangan siber telah meningkat pesat dalam beberapa tahun terakhir (Barot, 2018).

2.2.1 Pengertian Keamanan Data

Sederhananya, keamanan data adalah praktik pengamanan data seseorang. Ini juga dikenal sebagai keamanan informasi, Keamanan TI, atau keamanan informasi elektronik. Data dapat diamankan menggunakan berbagai teknologi perangkat keras dan perangkat lunak. Beberapa alat umum adalah *antivirus*, enkripsi, *firewall*, otentikasi dua faktor, tambalan perangkat lunak, pembaruan, dll.

2.2.2 Pentingnya Keamanan Data

Banyak orang memiliki kesalahpahaman umum bahwa hanya organisasi besar, pemerintah, dan bisnis yang menjadi target pelaku *cyber*. Ya, ini tidak benar. Keamanan data tidak hanya penting untuk bisnis atau pemerintah. Komputer, tablet, dan perangkat seluler seseorang bisa menjadi target selanjutnya. Biasanya, pengguna biasa menjadi sasaran penyerang karena informasi sensitif mereka, seperti detail kartu kredit, detail perbankan, kata sandi, dll (Rao & Selvamani, 2015).

Keamanan dunia maya harus menyeluruh dan mulus untuk semua orang - apakah seseorang seorang individu atau bisnis. Menurut perkiraan oleh Pusat Studi Strategis dan Internasional, kejahatan dunia maya merugikan ekonomi global lebih dari 400 miliar USD per tahun. Tidak perlu dikatakan, pelanggaran data dan serangan *cyber* akan meningkat pada waktunya karena jaringan komputer berkembang - serangan *cyber* semakin besar dan semakin baik setiap hari.

2.2.3 Dampak Penyerangan Terhadap Data

Kejadian penyalahgunaan data tidak ingin menakut-nakuti seseorang atau apa pun, tetapi ada banyak cara di mana seseorang dapat terpengaruh. Cara-cara ini termasuk serangan *phishing*, serangan *malware*, serangan *ransomware*, serangan *man-in-the-middle*, dll. Ingat, kesadaran seseorang adalah keamanan seseorang. Di sini, saya membagikan praktik penting yang perlu seseorang mulai hari ini untuk melindungi diri dari peretas:

1 Jangan pernah mengklik spam, phishing, atau email yang mencurigakan. Verifikasi atau periksa email atau tautan dengan cermat sebelum membuka lampiran apa pun.

2 Jika sesuatu tampak terlalu bagus untuk menjadi kenyataan, mungkin itu benar. Jangan menjadi korban penawaran, seperti "iPhone X hanya dengan $ 10" atau "Selamat! seseorang memenangkan mobil. Buka lampiran untuk mengklaim sekarang."

3 Jangan pernah mengunduh perangkat lunak atau aplikasi yang tidak tepercaya atau bajakan.

4 Jangan mengunduh perangkat lunak keamanan palsu.

5 Gunakan antivirus dan / atau firewall

6 Jangan melakukan transaksi online jika situs web tidak diamankan. Periksa HTTPS atau bilah alamat hijau sebelum melakukan pembayaran atau mengetikkan detail sensitif apa pun

7 Gunakan otentikasi dua faktor.

8 Jangan membagikan informasi pribadi atau sensitif seseorang kepada orang asing.

2.2.4 Hubungan Keamanan Data Dengan Bisnis

Informasi dan data dalam bisnis seseorang adalah aset bisnis yang berharga. Ini bisa menjadi kunci pertumbuhan dan kesuksesan. Keamanan data seseorang, oleh karena itu, harus menjadi prioritas dalam bisnis seseorang. Itu perlu dilindungi dari akses tidak sah untuk mencegahnya dirusak, dihancurkan atau diungkapkan kepada orang lain. Keamanan dapat dilanggar dalam sejumlah cara, misalnya oleh kegagalan sistem, pencurian, penggunaan yang tidak tepat, akses tidak sah atau virus komputer. Setiap kali seseorang terlibat dalam apa pun yang melibatkan Internet, keamanan data seseorang berisiko. Praktik kerja modern seperti kerja jarak jauh, perangkat IT portabel dan Wi-Fi semuanya meningkatkan ancaman terhadap keamanan data. Bahkan jika seseorang bekerja sendirian dari perangkat berbasis meja tunggal seseorang masih berisiko.

Efek dari pelanggaran keamanan data bisa menjadi bencana besar. Tidak hanya dalam hal gangguan pada operasi bisnis seseorang, tetapi juga potensi

kerusakan jangka panjang pada reputasi seseorang. seseorang mungkin telah menghabiskan beberapa tahun membangun merek dan reputasi seseorang untuk dihancurkan hanya dalam beberapa jam. Ada banyak cara untuk memastikan keamanan data - mulai dari pendidikan staf seseorang hingga solusi perangkat lunak dan perangkat keras. Tidak ada metode tunggal yang berdiri sendiri akan menawarkan solusi keamanan data yang lengkap sehingga penting untuk memahami di mana kerentanan seseorang dan melindungi diri seseorang sendiri.

Keamanan data adalah seperangkat alat dan teknologi yang melindungi data dari kehancuran, modifikasi, atau pengungkapan yang disengaja atau tidak disengaja. Keamanan data dapat diterapkan dengan menggunakan berbagai teknik dan teknologi, termasuk kontrol administratif, keamanan fisik, kontrol logis, organisasi, dan teknik perlindungan lainnya yang membatasi akses ke pengguna atau proses yang tidak sah atau berbahaya (Rao & Selvamani, 2015).

Keamanan data mengacu pada langkah-langkah privasi digital pelindung yang diterapkan untuk mencegah akses tidak sah ke komputer, *database*, dan situs *web*. Keamanan data juga melindungi data dari korupsi. Keamanan data adalah aspek penting dari TI untuk organisasi dari berbagai ukuran dan tipe. Keamanan data juga dikenal sebagai keamanan informasi atau keamanan komputer.

Contoh teknologi keamanan data termasuk backup, masking data dan penghapusan data. Ukuran teknologi keamanan data utama adalah enkripsi, di mana data digital, perangkat lunak / perangkat keras, dan *hard drive* dienkripsi dan karenanya tidak dapat dibaca oleh pengguna dan peretas yang tidak sah. Salah satu metode yang paling umum dijumpai dalam mempraktikkan keamanan data adalah

penggunaan otentikasi. Dengan otentikasi, pengguna harus memberikan kata sandi, kode, data biometrik, atau bentuk data lainnya untuk memverifikasi identitas sebelum akses ke sistem atau data diberikan. Keamanan data juga sangat penting untuk catatan perawatan kesehatan, sehingga pendukung kesehatan dan praktisi medis di AS dan negara-negara lain berupaya menerapkan privasi rekam medis elektronik dengan menciptakan kesadaran tentang hak-hak pasien terkait dengan pelepasan data ke laboratorium, dokter, rumah sakit dan fasilitas medis lainnya.

2.2.5 Solusi Keamanan Data

Data membutuhkan enkripsi dalam mengamankan informasi yang ada dalam data tersebut. Dengan enkripsi data canggih, *tokenization*, dan manajemen utama untuk melindungi data di seluruh aplikasi, transaksi, penyimpanan, dan *platform big data*, Teknik ini menyederhanakan perlindungan data sensitif bahkan dalam kasus penggunaan yang paling kompleks sekalipun. Beberapa model keamanan data antara lain:

1. Keamanan akses *cloud* - Platform perlindungan yang memungkinkan seseorang untuk pindah ke *cloud* dengan aman sambil melindungi data dalam aplikasi *cloud*.
2. Enkripsi data - Solusi keamanan data-sentris dan tokenisasi yang melindungi data di lingkungan perusahaan, *cloud*, seluler, dan data besar.
3. Modul keamanan perangkat keras - Modul keamanan perangkat keras yang menjaga data keuangan dan memenuhi persyaratan keamanan dan kepatuhan industri.

4. Manajemen kunci - Solusi yang melindungi data dan memungkinkan kepatuhan regulasi industri.
5. *Enterprise Data Protection* - Solusi yang menyediakan pendekatan *data-centric end-to-end* untuk perlindungan data perusahaan.
6. Keamanan Pembayaran - Solusi menyediakan enkripsi dan tokenisasi point-to-point lengkap untuk transaksi pembayaran ritel, memungkinkan pengurangan lingkup PCI.
7. Big Data, Hadoop, dan perlindungan data IofT - Solusi yang melindungi data sensitif di Danau Data - termasuk Hadoop, Teradata, Micro Focus Vertica, dan platform Big Data lainnya.
8. Keamanan Aplikasi Seluler - Melindungi data sensitif di aplikasi seluler asli sembari menjaga data dari ujung ke ujung.
9. Keamanan Peramban Web - Melindungi data sensitif yang diambil di peramban, dari titik pelanggan memasukkan pemegang kartu atau data pribadi dan menjaganya agar tetap terlindungi melalui ekosistem ke tujuan tuan rumah tepercaya.
10. eMail Security - Solusi yang menyediakan enkripsi ujung ke ujung untuk email dan olahpesan seluler, menjaga informasi pribadi dan informasi kesehatan pribadi tetap aman dan pribadi.

2.2.6 Kerahasiaan

Kerahasiaan mengacu pada melindungi informasi agar tidak diakses oleh pihak yang tidak berwenang. Dengan kata lain, hanya orang yang diberi wewenang

untuk melakukannya yang dapat memperoleh akses ke data sensitif. Bayangkan catatan bank harus dapat diakses, tentu saja, dan karyawan di bank yang membantu dalam menjalankan transaksi harus dapat mengaksesnya, tetapi tidak ada orang lain yang seharusnya. Kegagalan untuk menjaga kerahasiaan berarti bahwa seseorang yang seharusnya tidak memiliki akses telah berhasil mendapatkannya, melalui perilaku yang disengaja atau karena kecelakaan. Kegagalan kerahasiaan seperti itu, umumnya dikenal sebagai pelanggaran, biasanya tidak dapat diperbaiki. Setelah rahasia itu terungkap, tidak ada cara untuk mengetahuinya. Jika catatan bank diposting di situs web publik, semua orang dapat mengetahui nomor rekening bank, saldo, dll., Informasi itu tidak dapat dihapus dari pikiran, kertas, komputer, dan tempat lain mereka. Hampir semua insiden keamanan utama yang dilaporkan di media saat ini melibatkan kerugian besar kerahasiaan. Jadi, secara ringkas, pelanggaran kerahasiaan berarti bahwa seseorang memperoleh akses ke informasi yang seharusnya tidak memiliki akses ke sana.

2.2.7 Integritas

Integritas mengacu pada memastikan keaslian informasi bahwa informasi tidak diubah, dan bahwa sumber informasi itu asli. Bayangkan jika seseorang memiliki situs web dan seseorang menjual produk di situs itu. Sekarang bayangkan penyerang dapat berbelanja di situs web dan dengan jahat mengubah harga produk seseorang sehingga mereka dapat membeli apa pun dengan harga berapa pun yang mereka pilih. Itu akan menjadi kegagalan integritas karena informasi dalam hal ini, harga suatu produk telah diubah dan perubahan ini tidak dapat digagalkan. Contoh

lain dari kegagalan integritas adalah ketika sesorang mencoba terhubung ke situs web dan penyerang jahat antara seseorang dan situs web mengalihkan lalu lintas ke situs web yang berbeda. Dalam hal ini, situs yang dituju tidak asli.

2.2.8 Ketersediaan

Ketersediaan berarti informasi dapat diakses oleh pengguna yang berwenang. Jika penyerang tidak dapat mengkompromikan dua elemen pertama dari keamanan informasi (lihat di atas) mereka dapat mencoba melakukan serangan seperti penolakan layanan yang akan menurunkan *server*, membuat situs web tidak tersedia untuk pengguna yang sah karena kurangnya ketersediaan.

2.2.9 Kontrol Akses

Kesalahan terbesar yang bisa dilakukan oleh perancang aplikasi adalah mengabaikan kontrol akses sebagai bagian dari fungsionalitas yang diperlukan. Jarang bahwa setiap pengguna atau sistem yang berinteraksi dengan suatu aplikasi harus memiliki hak yang sama di seluruh aplikasi itu. Beberapa pengguna mungkin memerlukan akses ke data tertentu dan bukan yang lain; beberapa sistem harus atau tidak dapat mengakses aplikasi. Akses ke komponen, fungsi, atau modul tertentu dalam aplikasi juga harus dikontrol. Kontrol akses juga penting untuk kepatuhan audit dan peraturan. Beberapa cara umum mengelola kontrol akses adalah:

1. Baca, tulis, dan jalankan hak istimewa: File
2. Kontrol akses berbasis peran: administrator, pengguna
3. Alamat IP akses berbasis host, nama mesin

4. Objek kode kontrol akses tingkat objek, banyak pembaca / penulis tunggal

2.3 Algoritma

Pertimbangkan bagaimana seseorang menggunakan komputer pada hari-hari biasa. Misalnya, seseorang mulai mengerjakan laporan, dan setelah seseorang menyelesaikan paragraf, seseorang melakukan pemeriksaan ejaan. seseorang membuka aplikasi spreadsheet untuk melakukan beberapa proyeksi keuangan untuk melihat apakah seseorang dapat membeli pinjaman mobil baru. Seseorang menggunakan *browser web* untuk mencari secara online jenis mobil yang ingin seseorang beli (Gurevich, 2012).

Seseorang mungkin tidak memikirkan hal ini dengan sangat sadar, tetapi semua operasi yang dilakukan oleh komputer seseorang terdiri dari algoritma. Algoritma adalah prosedur yang didefinisikan dengan baik yang memungkinkan komputer untuk memecahkan masalah. Cara lain untuk menggambarkan suatu algoritma adalah urutan instruksi yang tidak ambigu. Penggunaan istilah 'tidak ambigu' menunjukkan bahwa tidak ada ruang untuk interpretasi subyektif. Setiap kali seseorang meminta komputer seseorang untuk melakukan algoritma yang sama, ia akan melakukannya dengan cara yang persis sama dengan hasil yang sama persis.

Pertimbangkan contoh-contoh sebelumnya lagi. Pengecekan ejaan menggunakan algoritma. Perhitungan keuangan menggunakan algoritma. Mesin pencari menggunakan algoritma. Bahkan, sulit untuk memikirkan tugas yang dilakukan oleh komputer seseorang yang tidak menggunakan algoritma.

Contoh algoritma yang sangat sederhana adalah menemukan angka terbesar dalam daftar angka yang tidak disortir. Jika Anda diberi daftar lima nomor yang berbeda, Anda akan dapat memecahkannya dalam waktu singkat, tidak perlu komputer. Sekarang, bagaimana dengan lima juta angka yang berbeda? Jelas, Anda akan membutuhkan komputer untuk melakukan ini, dan komputer membutuhkan algoritma.

Berikut ini adalah bagaimana algoritma itu terlihat. Katakanlah input terdiri dari daftar angka, dan daftar ini disebut L. Angka L1 akan menjadi angka pertama dalam daftar, L2 angka kedua, dll. Dan kita tahu daftar tidak diurutkan - jika tidak, jawabannya akan sangat mudah. Jadi, input ke algoritma adalah daftar angka, dan output harus menjadi angka terbesar dalam daftar.

Algoritma akan terlihat seperti ini:

Langkah 1: Biarkan Terbesar = L1

Ini berarti Anda mulai dengan mengasumsikan bahwa angka pertama adalah angka terbesar.

Langkah 2: Untuk setiap item dalam daftar:

Ini berarti Anda akan melalui daftar angka satu per satu.

Langkah 3: Jika item> Terbesar:

Jika Anda menemukan angka terbesar baru, lanjutkan ke langkah empat. Jika tidak, kembali ke langkah kedua, yang berarti Anda beralih ke nomor berikutnya dalam daftar.

Langkah 4: Kemudian Terbesar = item

Ini menggantikan angka terbesar lama dengan jumlah terbesar baru yang baru saja Anda temukan. Setelah ini selesai, kembali ke langkah dua hingga tidak ada lagi angka yang tersisa dalam daftar.

Langkah 5: Kembalikan Terbesar

Ini menghasilkan hasil yang diinginkan.

Perhatikan bahwa algoritma dijelaskan sebagai serangkaian langkah logis dalam bahasa yang mudah dipahami. Agar komputer dapat benar-benar menggunakan instruksi ini, mereka harus ditulis dalam bahasa yang dapat dimengerti oleh komputer, yang dikenal sebagai bahasa pemrograman (Zandbergen, 2019).

2.3.1 Desain Konseptual

Algoritma adalah serangkaian instruksi, sering disebut sebagai "proses," yang harus diikuti ketika memecahkan masalah tertentu. Meskipun secara teknis

tidak dibatasi oleh definisi, kata itu hampir selalu terkait dengan komputer, karena algoritma yang diproses komputer dapat mengatasi masalah yang jauh lebih besar daripada manusia, jauh lebih cepat. Karena komputasi modern menggunakan algoritma jauh lebih sering daripada pada titik lain dalam sejarah manusia, bidang telah tumbuh di sekitar desain, analisis, dan penyempurnaan. Bidang desain algoritma membutuhkan latar belakang matematika yang kuat, dengan gelar ilmu komputer yang sangat dicari kualifikasi. Ini menawarkan semakin banyak pilihan karir yang sangat dikompensasi, karena kebutuhan akan lebih banyak (dan juga lebih canggih) algoritma terus meningkat.

Pada tingkat yang paling sederhana, algoritma pada dasarnya hanya seperangkat instruksi yang diperlukan untuk menyelesaikan tugas. Pengembangan algoritma, meskipun umumnya tidak disebut demikian, telah menjadi kebiasaan yang populer dan pengejaran profesional untuk semua catatan sejarah. Jauh sebelum fajar era komputer modern, orang menetapkan rutinitas yang telah ditentukan untuk bagaimana mereka akan melakukan tugas sehari-hari, sering menuliskan daftar langkah-langkah yang harus diambil untuk mencapai tujuan penting, mengurangi risiko melupakan sesuatu yang penting. Ini, pada dasarnya, adalah apa itu algoritma. Desainer mengambil pendekatan yang mirip dengan pengembangan algoritma untuk tujuan komputasi: pertama, mereka melihat masalah. Kemudian, mereka menguraikan langkah-langkah yang akan diperlukan untuk menyelesaikannya. Akhirnya, mereka mengembangkan serangkaian operasi matematika untuk mencapai langkah-langkah tersebut.

2.3.2 Tugas Algoritma

Tugas sederhana dapat diselesaikan dengan algoritma yang dihasilkan dengan beberapa menit, atau paling banyak pekerjaan pagi. Tingkat kompleksitas menjalankan tantangan yang panjang, namun, sampai pada masalah yang sangat rumit sehingga mereka telah menghalangi matematikawan yang tak terhitung jumlahnya selama bertahun-tahun - atau bahkan berabad-abad. Komputer modern menghadapi masalah pada tingkat ini di bidang-bidang seperti keamanan dunia maya, serta penanganan data besar - penyortiran set data yang efisien dan menyeluruh sedemikian besar sehingga bahkan komputer tidak dapat memprosesnya secara tepat waktu. Contoh data besar mungkin termasuk "setiap artikel di Wikipedia," "setiap halaman web yang diindeks dan diarsipkan akan kembali ke tahun 1998," atau "enam bulan terakhir pembelian online yang dilakukan di Amerika."

2.3.3 Rekayasa Algoritma

Ketika desain algoritma baru diterapkan dalam istilah praktis, disiplin terkait dikenal sebagai rekayasa algoritma. Kedua fungsi tersebut sering dilakukan oleh orang yang sama, meskipun organisasi yang lebih besar (seperti Amazon dan Google) mempekerjakan desainer dan insinyur khusus, mengingat tingkat kebutuhan mereka akan algoritma baru dan khusus. Seperti proses desain, rekayasa algoritma sering kali melibatkan akreditasi sains komputer, dengan latar belakang yang kuat dalam matematika: di mana mereka ada sebagai profesi yang terpisah dan terspesialisasi, insinyur algoritma mengambil ide-ide konseptual dari desainer dan

proses kreatif dari mereka yang akan dipahami oleh komputer. Dengan kemajuan teknologi digital yang mantap, para insinyur yang berdedikasi akan terus menjadi semakin umum.

2.4 Kriptografi

Kriptografi adalah teknik mengubah dan mentransmisikan data rahasia dengan cara disandikan sehingga hanya pengguna yang berwenang dan dimaksudkan dapat memperoleh atau bekerja di dalamnya. Ini adalah kata asal Yunani di mana "*crypto*" berarti tersembunyi dan "*graphy*" berarti menulis, jadi kriptografi berarti tulisan tersembunyi atau rahasia. Ini memperkenalkan triad seperti kerahasiaan, non-penolakan, integritas dan keaslian dalam komunikasi data yang sedang berlangsung.

Kriptografi adalah disiplin atau teknik yang digunakan dalam melindungi integritas atau kerahasiaan pesan elektronik dengan mengubahnya menjadi bentuk (*ciphertext*) yang tidak dapat dibaca. Hanya penggunaan kunci rahasia yang dapat mengubah teks sandi menjadi bentuk yang dapat dibaca manusia (teks jelas). Perangkat lunak kriptografi dan / atau perangkat keras menggunakan rumus matematika (algoritma) untuk mengubah teks dari satu bentuk ke bentuk lainnya.

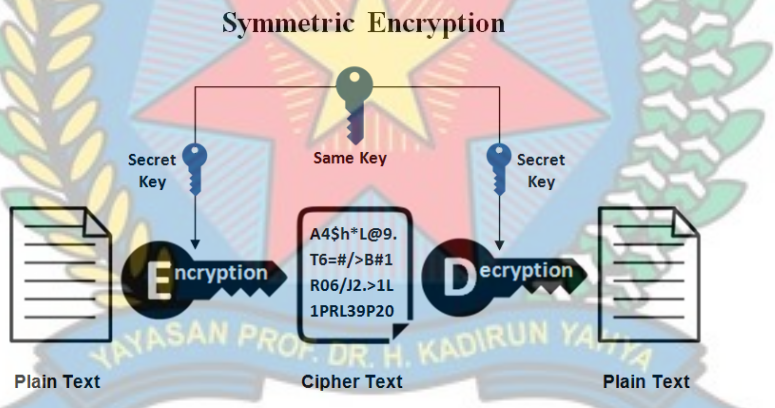
Komunikasi yang aman dapat disediakan menggunakan teknik, di hadapan konten pihak ketiga berbahaya yang disebut musuh. Teknik-teknik ini dapat disebut sebagai Kriptografi. Pesan pribadi apa pun dapat disembunyikan dari publik atau pihak ketiga, menggunakan seperangkat protokol. Protokol-protokol ini perlu dianalisis dan dibangun dengan cara yang efisien untuk menjaga kerahasiaan pesan

yang dikirim. Kriptografi modern memiliki aspek tertentu yang merupakan pusatnya seperti integritas data, otentikasi, kerahasiaan dll. Di dunia modern, kriptografi sangat bergantung pada mata pelajaran seperti matematika dan ilmu komputer. Algoritma untuk kriptografi dirancang sedemikian rupa sehingga sulit untuk dipecahkan dalam praktik oleh pihak ketiga jahat yang juga dikenal sebagai musuh. Pendekatan praktis terhadap pemecahan algoritma semacam itu akan gagal, namun, pendekatan teoritis mungkin memecahkan sistem tersebut. Dengan demikian, algoritma apa pun dapat disebut sebagai aman, jika sifat kuncinya tidak dapat disimpulkan, dengan ciphertext yang diberikan. Kriptografi dapat dikategorikan menjadi dua cabang: *Symmetric* dan *Asymmetric*. Dengan pendekatan simetris, satu kunci digunakan untuk proses enkripsi dan dekripsi yaitu pengirim dan penerima harus memiliki kunci bersama. Namun, dengan pendekatan ini, distribusi kunci adalah tautan yang lemah, yang memunculkan pendekatan baru.

2.4.1 Kriptografi Simetris

Kriptografi kunci simetris adalah setiap algoritma kriptografi yang didasarkan pada kunci bersama yang digunakan untuk mengenkripsi atau mendekripsi teks / *cyphertext*, dalam kontrak dengan kriptografi kunci asimetris, di mana kunci enkripsi dan dekripsi dihubungkan oleh berbeda. Enkripsi simetris umumnya lebih efisien daripada enkripsi asimetris dan karenanya lebih disukai ketika sejumlah besar data perlu dipertukarkan. Membuat kunci bersama sulit menggunakan hanya algoritma enkripsi simetris, sehingga dalam banyak kasus, enkripsi asimetris digunakan untuk membuat kunci bersama antara dua pihak.

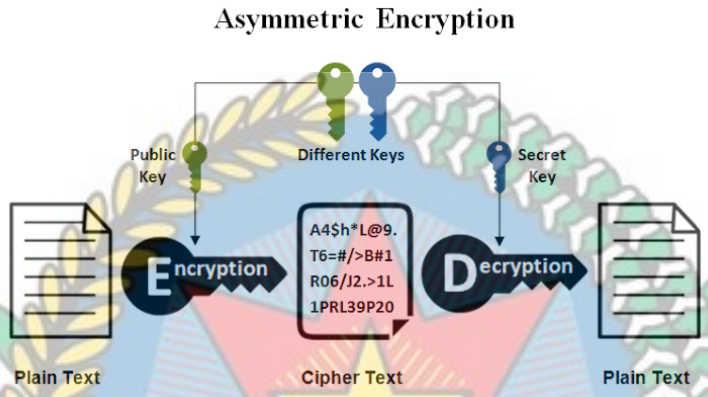
Contoh untuk kriptografi kunci simetris termasuk AES, DES, dan 3DES. Protokol pertukaran kunci yang digunakan untuk membangun kunci enkripsi bersama termasuk Diffie-Hellman (DH), Eliptic Curve (EC) dan RSA. Berikut ini skema dari kriptografi simetris (Ayushi, 2020).



Gambar 2.1 Skema kriptografi simetris
Sumber: (Ayushi, 2020)

2.4.2 Kriptografi Asimetris

Dalam versi kriptografi asimetris, pengirim dan penerima memiliki dua kunci, publik dan pribadi. Kunci pribadi dirahasiakan sedangkan kunci publik terbuka ke dunia luar. Set data apa pun, yang dienkripsi dengan kunci publik hanya dapat didekripsi menggunakan kunci pribadi yang sesuai. Ketika datang ke perbandingan, pendekatan simetris lebih cepat daripada yang asimetris. Contoh - seseorang tangan digital menggunakan kriptografi asimetris untuk mengenkripsi pesan dalam hash alih-alih pesan lengkap. Berikut ini skema kriptografi asimetris (S. et al., 2012).



Gambar 2.2 Skema kriptografi asimetris
Sumber: (Ayushi, 2020)

2.5 Kriptografi Modern

Kriptografi adalah praktik dan studi teknik untuk komunikasi yang aman di hadapan pihak ketiga. Secara lebih umum, ini adalah tentang membangun dan menganalisis protokol yang mengatasi pengaruh musuh dan yang terkait dengan berbagai aspek dalam keamanan informasi seperti kerahasiaan data, integritas data, otentikasi, dan non-penyangkalan. Kriptografi modern berpotongan dengan disiplin ilmu matematika, ilmu komputer, dan teknik kelistrikan. Aplikasi kriptografi meliputi kartu ATM, sandi komputer, dan perdagangan elektronik.

2.5.1 Penerapan Kriptografi Modern

Kriptografi modern tampaknya sangat cocok untuk menyelesaikan konflik yang tampak antara verifikasi dan privasi dalam sistem pemungutan suara, tetapi ada kendala dalam penerapannya. Membangun pemahaman dan kepercayaan dalam mekanisme dan jaminan yang diberikan oleh sistem kriptografi tidaklah mudah. Selain itu, penerapan kriptografi yang tepat dapat menjadi rumit dan bermasalah.

Karena privasi yang diberikan oleh sarana kriptografi biasanya bersifat komputasi, mungkin ada kekhawatiran tentang privasi jangka panjang suara. Namun, skema telah dirancang untuk memberikan privasi abadi. Sistem pemungutan suara berbasis kertas yang bebas enkripsi telah dijelaskan sebelumnya di bawah skema Randell dan Ryan. Kesederhanaan relatif dari sistem ini, bersama dengan kemiripannya dengan permainan kartu lotere, mungkin membantu dalam mendapatkan kepercayaan dan kepercayaan pemilih.

2.5.2 Perbandingan Kriptografi Modern dengan Klasik

Ada tiga karakteristik utama yang memisahkan kriptografi modern dari pendekatan klasik.

1. Kriptografi klasik memanipulasi karakter tradisional, yaitu huruf dan angka secara langsung sedangkan kriptografi modern beroperasi pada urutan bit biner.
2. Kriptografi klasik didasarkan pada 'keamanan melalui ketidakjelasan'. Teknik yang digunakan untuk pengkodean dirahasiakan dan hanya pihak yang terlibat dalam komunikasi yang tahu tentang mereka. Kriptografi modern bergantung pada algoritma matematika yang dikenal publik untuk mengkodekan informasi. Kerahasiaan diperoleh melalui kunci rahasia yang digunakan sebagai benih untuk algoritma. Kesulitan komputasi algoritme, tidak adanya kunci rahasia, dll., Membuat penyerang tidak mungkin mendapatkan informasi asli bahkan jika dia mengetahui algoritme yang digunakan untuk pengkodean.

3. Kriptografi klasik membutuhkan seluruh kriptosistem untuk berkomunikasi secara rahasia. Kriptografi modern mengharuskan pihak yang tertarik dengan komunikasi yang aman untuk memiliki kunci rahasia saja.

2.6 RSA

Algoritma RSA adalah algoritma kriptografi asimetris. Asimetris sebenarnya berarti bekerja pada dua kunci yang berbeda yaitu Kunci Publik dan Kunci Pribadi. Seperti namanya menjelaskan bahwa Kunci Publik diberikan kepada semua orang dan kunci Pribadi dirahasiakan. Contoh kriptografi asimetris:

1. Seorang klien (misalnya browser) mengirimkan kunci publiknya ke server dan meminta beberapa data.
2. *Server* mengenkripsi data menggunakan kunci publik klien dan mengirimkan data terenkripsi.
3. Klien menerima data ini dan mendekripsinya.

Karena ini asimetris, tidak ada orang lain kecuali browser yang dapat mendekripsi data meskipun pihak ketiga memiliki kunci publik browser. Ide RSA didasarkan pada kenyataan bahwa sulit untuk memfaktorkan bilangan bulat besar. Kunci publik terdiri dari dua bilangan dimana satu bilangan merupakan perkalian dari dua bilangan prima besar. Dan kunci privat juga diturunkan dari dua bilangan prima yang sama. Jadi, jika seseorang dapat memfaktorkan jumlah yang besar, kunci pribadi akan dikompromikan. Oleh karena itu kekuatan enkripsi sepenuhnya terletak pada ukuran kunci dan jika kita menggandakan atau melipatgandakan

ukuran kunci, kekuatan enkripsi meningkat secara eksponensial. Kunci RSA biasanya panjangnya 1024 atau 2048 bit, tetapi para ahli percaya bahwa kunci 1024 bit bisa rusak dalam waktu dekat. Tapi sampai sekarang tampaknya menjadi tugas yang tidak layak.

2.7 Merkle-Hellman

Merkle-Hellman adalah kriptosistem kunci asimetris, artinya untuk komunikasi, diperlukan dua kunci: kunci publik dan kunci pribadi. Lebih jauh, tidak seperti RSA, ini adalah satu arah -- kunci publik hanya digunakan untuk enkripsi, dan kunci privat hanya digunakan untuk dekripsi. Jadi tidak dapat digunakan untuk otentikasi dengan penandatanganan kriptografi.

Sistem Merkle-Hellman didasarkan pada masalah jumlah subset (kasus khusus dari masalah ransel). Masalahnya adalah sebagai berikut: diberikan satu set angka Templat: Matematika dan angka b, temukan subset Templat: Matematika, yang berjumlah b. Secara umum, masalah ini dikenal sebagai NP-complete. Namun, jika himpunan bilangan (disebut knapsack) bertambah besar — yaitu, setiap elemen himpunan lebih besar dari jumlah semua bilangan sebelumnya — masalahnya 'mudah' dan dapat diselesaikan dalam waktu polinomial dengan serakah sederhana algoritma.

Di Merkle-Hellman, kuncinya adalah ransel. Kunci publik adalah knapsack 'keras', dan kunci privat adalah knapsack 'mudah', atau superincreasing, dikombinasikan dengan dua angka tambahan, pengali dan modulus, yang

digunakan untuk mengubah knapsack superincreasing menjadi hard knapsack. Angka-angka yang sama ini digunakan untuk mengubah jumlah subset dari hard knapsack menjadi jumlah dari subset dari easy knapsack, yang dapat diselesaikan dalam waktu polinomial.

Untuk mengenkripsi pesan, subset dari hard knapsack dipilih dengan membandingkannya dengan sekumpulan bit (plaintext), sama panjang dengan kunci, dan membuat setiap istilah dalam kunci publik yang sesuai dengan 1 di plaintext menjadi elemen subset, sementara mengabaikan istilah yang sesuai dengan 0 istilah dalam plaintext. Elemen-elemen dari subset ini ditambahkan bersama-sama, dan jumlah yang dihasilkan adalah ciphertext.

Dekripsi dimungkinkan karena pengali dan modulus yang digunakan untuk mentransformasikan superincreasing knapsack menjadi kunci publik juga dapat digunakan untuk mengubah angka yang mewakili ciphertext menjadi jumlah elemen yang sesuai dari superincreasing knapsack. Kemudian, dengan menggunakan algoritma serakah sederhana, ransel mudah dapat diselesaikan menggunakan operasi aritmatika O(n), yang mendekripsi pesan.

2.8 *Unified Modeling Language (UML)*

Unified Modeling Language (UML) adalah bahasa pemodelan yang memungkinkan pengembang menentukan, memvisualisasikan, membuat, dan mendokumentasikan artefak sistem perangkat lunak (Technopedia, 2019). Dengan demikian, UML membuat artefak ini dapat diskalakan, aman, dan kuat dalam eksekusi. UML adalah aspek penting yang terlibat dalam pengembangan perangkat

lunak berorientasi objek. Ini menggunakan notasi grafis untuk membuat model visual dari sistem perangkat lunak. Arsitektur UML didasarkan pada fasilitas meta-objek, yang mendefinisikan dasar untuk membuat bahasa pemodelan. Mereka cukup tepat untuk menghasilkan seluruh aplikasi. UML yang sepenuhnya dapat dieksekusi dapat digunakan untuk berbagai platform menggunakan teknologi yang berbeda dan dapat digunakan dengan semua proses sepanjang siklus pengembangan perangkat lunak. UML dirancang untuk memungkinkan pengguna mengembangkan bahasa pemodelan visual yang ekspresif, siap pakai. Selain itu, mendukung konsep pengembangan tingkat tinggi seperti kerangka kerja, pola, dan kolaborasi (Wasserkrug et al., 2019).

Penggunaan model ini bertujuan untuk mengidentifikasikan bagian-bagian yang termasuk dalam lingkup sistem yang dibahas dan bagaimana hubungan antara sistem dengan subsistem maupun sistem lain diluarnya (Sukmawati & Priyadi, 2019).

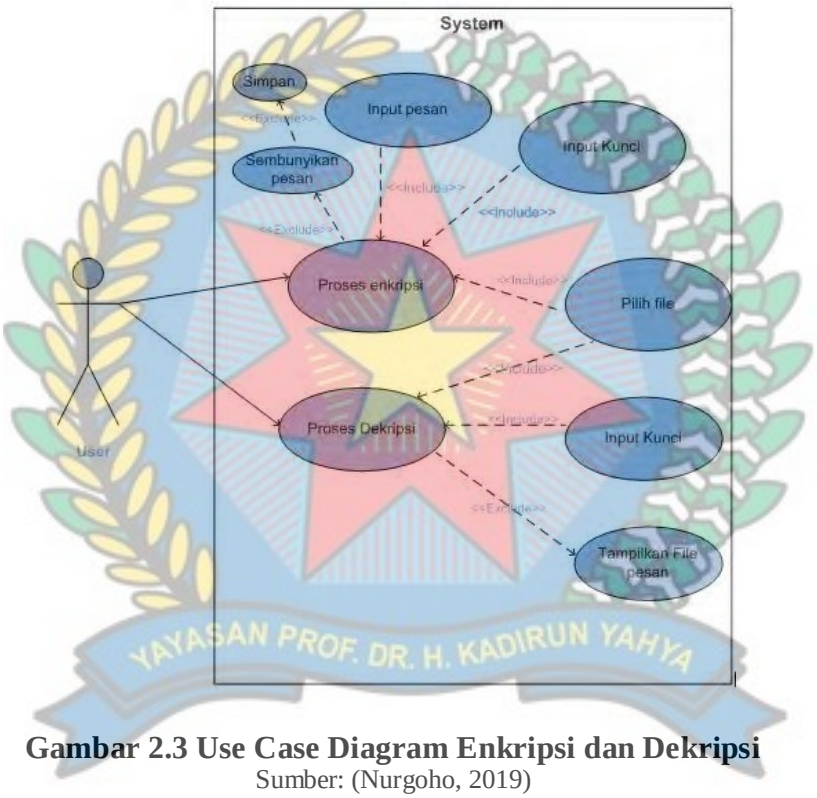
2.8.1 *Use Case Diagram*

Use Case Diagram adalah model tentang bagaimana berbagai jenis pengguna berinteraksi dengan sistem untuk memecahkan masalah. Dengan demikian, ini menggambarkan tujuan pengguna, interaksi antara pengguna dan sistem, dan perilaku sistem yang diperlukan dalam memenuhi tujuan-tujuan ini. *Use Case Diagram* terdiri dari sejumlah elemen model. Elemen model yang paling penting adalah kasus penggunaan, aktor dan hubungan di antara mereka. *Use Case Diagram* digunakan untuk menggambarkan secara grafis subset dari model untuk

menyederhanakan komunikasi. Biasanya akan ada beberapa diagram kasus penggunaan yang terkait dengan model yang diberikan, masing-masing menunjukkan subset elemen model yang relevan untuk tujuan tertentu. Elemen model yang sama dapat ditampilkan pada beberapa *Use Case Diagram*, tetapi setiap instance harus konsisten. Jika alat digunakan untuk mempertahankan *Use Case Diagram*, kendala konsistensi ini otomatis sehingga setiap perubahan pada elemen model (mengubah nama misalnya) akan secara otomatis tercermin dalam setiap *Use Case Diagram* yang menunjukkan elemen itu (UTM, 2019).

Use Case Diagram dapat berisi paket yang digunakan untuk menyusun model untuk menyederhanakan analisis, komunikasi, navigasi, pengembangan, pemeliharaan, dan perencanaan. Faktanya, sebagian besar *Use Case Diagram* adalah tekstual, dengan teks yang ditangkap dalam Spesifikasi Use-Case yang terkait dengan setiap elemen *Use Case Diagram*. Spesifikasi ini menjelaskan alur peristiwa *Use Case*. *Use Case Diagram* berfungsi sebagai utas pemersatu sepanjang pengembangan sistem. Ini digunakan sebagai spesifikasi utama dari persyaratan fungsional untuk sistem, sebagai dasar untuk analisis dan desain, sebagai input untuk perencanaan iterasi, sebagai dasar mendefinisikan kasus uji dan sebagai dasar untuk dokumentasi pengguna. (Kurniawan, 2018).

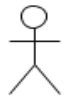
Use Case Diagram merupakan suatu diagram yang berisi *Use Case*, *actor*, serta *relationship* diantaranya. *Use Case Diagram* dapat digunakan untuk kebutuhan apa saja yang diperlukan dalam suatu sistem, sehingga sistem dapat digambarkan dengan jelas bagaimana proses dari sistem tersebut, bagaimana cara aktor menggunakan sistem, serta apa saja yang dapat dilakukan pada suatu sistem.

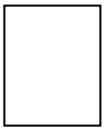


Gambar 2.3 Use Case Diagram Enkripsi dan Dekripsi
Sumber: (Nurgoho, 2019)

Gambar 2.3 adalah contoh dari diagram pada proses enkripsi dan dekripsi. Use-case memiliki beberapa simbol untuk menyatakan kegiatan dari use-case tersebut. Adapun simbol dari *Use Case* adalah sebagai berikut:

Tabel 2.1 Simbol *Use Case Diagram*

No	Gambar	Nama	Keterangan
1		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *Use Case*.

2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
3		*Generalization*	Hubungan dimana objek anak berbagi perilaku dan struktur data dari objek yang ada diatasnya.
4		*Include*	Menspesifikasikan bahwa *Use Case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *Use Case* target memperluas perilaku dari *Use Case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang

			menghasilkan suatu hasil yang terukur bagi suatu actor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Sumber: (Kurniawan, 2018)

2.8.2 *Activity Diagram*

Activity Diagram (Diagram Aktifitas) menggambarkan berbagai alir aktifitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir (Ladjamudin, 2017).

Activity Diagram menurut adalah salah satu cara untuk memodelkan *event-event* yang terjadi dalam suatu *Use Case*. Diagram ini juga dapat digantikan dengan sejumlah teks.

Tabel 2.2 Simbol *Activity Diagram*

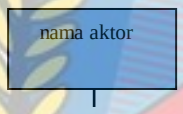
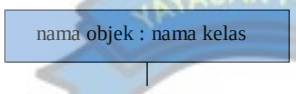
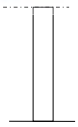
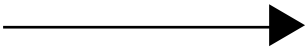
No	Gambar	Nama	Keterangan
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk /diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

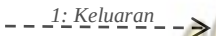
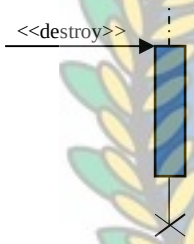
Sumber: (Kurniawan, 2018)

2.8.3 *Sequence Diagram*

Diagram sekuen menggambarkan kelakuan objek pada *Use Case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *Use Case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Tabel berikut adalah simbol-simbol yang ada pada diagram sekuen.

Tabel 2.3 Simbol *Sequence Diagram*

Simbol-simbol	Deskripsi
Aktor  Atau 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi itu sendiri, jadi walaupun simbol dari aktor adalah orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda diawal *frase* nama aktor
Garis hidup / *Lifeline* 	Menyatakan kehidpan suatu objek
Objek 	Menyatakan objek yang berinteraksi
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.
Pesan tipe *create*  <<create>>	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki operasi / metode, karena ini memanggil operasi / metode maka operasi / metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim

Pesan tipe *return* 	Menyatakan suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada *destroy*

Sumber: (Kurniawan, 2018)

2.9 Flowchart

Flowchart digunakan dalam mendesain dan mendokumentasikan proses atau program sederhana. Seperti jenis diagram lainnya, diagram membantu memvisualisasikan apa yang sedang terjadi dan dengan demikian membantu memahami suatu proses, dan mungkin juga menemukan fitur-fitur yang kurang jelas dalam proses tersebut, seperti kekurangan dan hambatan. Ada berbagai jenis diagram alur: masing-masing jenis memiliki set kotak dan notasi sendiri. Dua jenis kotak yang paling umum dalam diagram alur adalah:

1 langkah pemrosesan, biasanya disebut aktivitas dan dilambangkan sebagai kotak persegi panjang.

2 keputusan biasanya dilambangkan sebagai berlian.

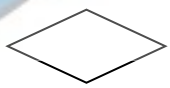
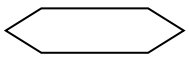
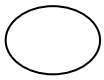
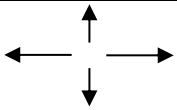
Diagram alir digambarkan sebagai "lintas fungsional" ketika bagan dibagi menjadi bagian vertikal atau horizontal yang berbeda, untuk menggambarkan kontrol unit organisasi yang berbeda. Simbol yang muncul di bagian tertentu berada dalam kendali unit organisasi itu. Flowchart lintas fungsional memungkinkan penulis untuk menemukan tanggung jawab untuk melakukan suatu tindakan atau membuat keputusan dengan benar, dan untuk menunjukkan tanggung jawab masing-masing unit organisasi untuk bagian berbeda dari satu proses tunggal.

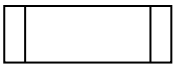
Diagram alir menggambarkan aspek-aspek tertentu dari proses dan biasanya dilengkapi dengan jenis diagram lainnya. Misalnya, Kaoru Ishikawa, mendefinisikan diagram alir sebagai salah satu dari tujuh alat dasar kendali mutu, di sebelah histogram, diagram Pareto, lembar periksa, diagram kontrol, diagram sebab-akibat, dan diagram sebaran. Demikian pula, di UML, notasi pemodelan konsep yang digunakan dalam pengembangan perangkat lunak, diagram aktivitas, yang merupakan jenis diagram alur, hanyalah salah satu dari banyak jenis diagram yang berbeda.

Diagram Nassi-Shneiderman dan Drakon-chart adalah notasi alternatif untuk aliran proses. Nama alternatif umum termasuk diagram alir, diagram alur proses, diagram alur fungsional, peta proses, diagram proses, diagram proses fungsional, model proses bisnis, model proses, diagram alir proses, diagram alur kerja, diagram alir bisnis. Istilah "diagram alur" dan "diagram alir" digunakan secara bergantian (Nakatsu, 2019).

Struktur grafik yang mendasari diagram alur adalah grafik aliran, yang mengabstraksi jenis simpul, isinya, dan informasi tambahan lainnya. Adapun simbol-simbol flowchart lihat pada tabel sebagai berikut :

Tabel 2.4 Simbol *Flowchart*

NO	SIMBOL	FUNGSI
1.		**Terminal**, untuk memulai atau mengakhiri suatu program
2.		**Proses**, suatu simbol yang menunjukkan setiap pengolahan yang dilakukan.
3.		**Input-Output**, untuk memasukkan menunjukkan hasil dari suatu proses
4.		**Decision**, suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau pilihan
5.		**Preparation,** suatu symbol yang menyediakan tempat pengolahan
6.		**Connector**, suatu prosedur penghubung yang akan masuk atau keluar melalui symbol ini dalam lembar yang sama
7.		**Off-Page Connector**, merupakan symbol masuk atau keluarannya suatu prosedur pada lembaran kertas lainnya
8.		**Arus/Flow**, dari pada prosedur yang dapat dilakukan atas ke bawah dari bawah ke atas, ke atas dari kiri ke kanan ataupun dari kanan ke kiri

9.		**Predefined Process**, untuk menyatakan sekumpulan langkah proses yang ditulis sebagai prosedur
10.		Simbol untuk output, yang ditunjukkan ke suatu device, seperti printer, dan sebagainya
11		Penyimpanan file secara sementara
12		Menunjukkan input / Output Hardisk (media penyimpanan)

Sumber: (Kurniawan, 2018)

2.10 Microsoft Visual Studio

Microsoft Visual Studio adalah lingkungan pengembangan terintegrasi (IDE) dari Microsoft. Ini digunakan untuk mengembangkan program komputer, serta situs web, aplikasi web, layanan web, dan aplikasi seluler. Visual Studio menggunakan platform pengembangan perangkat lunak Microsoft seperti Windows API, Windows Forms, Windows Presentation Foundation, Windows Store dan Microsoft Silverlight. Ini dapat menghasilkan kode asli dan kode terkelola.

Visual Studio menyertakan editor kode yang mendukung IntelliSense (komponen penyelesaian kode) serta pemfaktoran ulang kode. Debugger terintegrasi berfungsi baik sebagai debugger tingkat sumber dan debugger tingkat mesin. Alat built-in lainnya termasuk code profiler, designer untuk membangun aplikasi GUI, *web designer*, *class designer*, dan *database* schema *designer*. Ini menerima plug-in yang memperluas fungsionalitas di hampir setiap level —

termasuk menambahkan dukungan untuk sistem kendali sumber (seperti Subversion dan Git) dan menambahkan perangkat baru seperti editor dan desainer visual untuk bahasa atau perangkat khusus domain untuk aspek lain dari pengembangan perangkat lunak siklus hidup (seperti klien Azure DevOps: Team Explorer).

Visual Studio mendukung 36 bahasa pemrograman yang berbeda dan memungkinkan editor kode dan debugger untuk mendukung (dalam berbagai tingkat) hampir semua bahasa pemrograman, asalkan ada layanan khusus bahasa. Bahasa bawaan termasuk C, C ++, C ++ / CLI, Visual Basic .NET, C #, F #, JavaScript, TypeScript, XML, XSLT, HTML, dan CSS. Dukungan untuk bahasa lain seperti Python, Ruby, Node.js, dan M antara lain tersedia melalui plug-in. Java (dan J #) telah didukung sebelumnya.

Edisi paling dasar dari Visual Studio, edisi Komunitas, tersedia secara gratis. Slogan untuk edisi Visual Studio Community adalah "IDE gratis dengan fitur lengkap untuk pelajar, pengembang sumber terbuka dan individu". Versi Visual Studio siap produksi terbaru adalah 2019, dengan versi yang lebih lama seperti 2012 dan 2013 pada Dukungan Perpanjangan, dan 2015 dan 2017 pada Dukungan Mainstream.

2.10.1 Edisi Visual Studio

Microsoft Visual Studio memiliki tiga buah edisi yang memiliki fitur yang berbeda, antara lain:

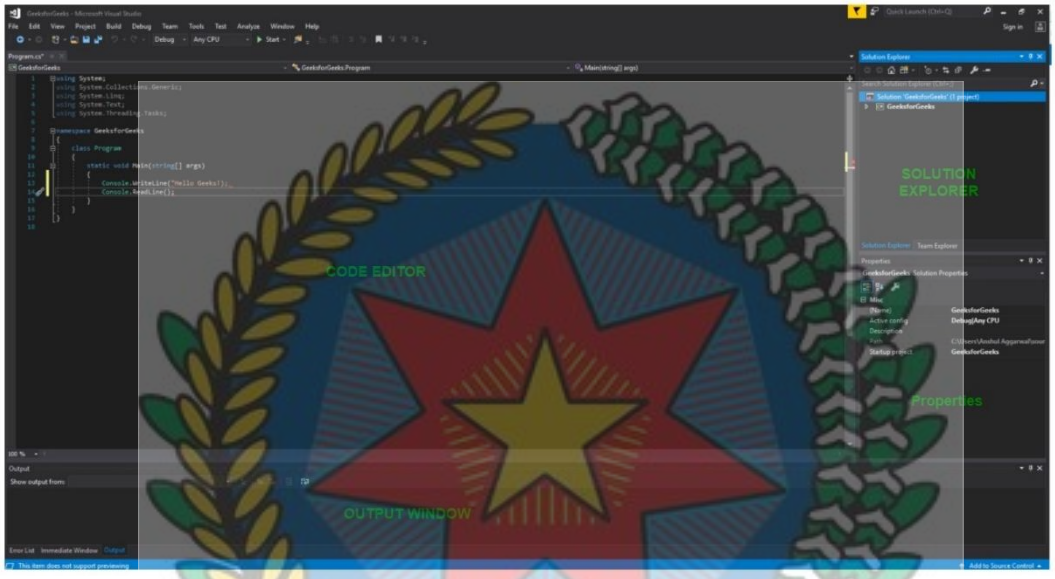
1. Komunitas: Ini adalah versi gratis yang diumumkan pada tahun 2014. Semua edisi lainnya berbayar. Ini berisi fitur yang mirip dengan edisi Profesional. Dengan menggunakan edisi ini, setiap pengembang dapat mengembangkan aplikasi gratis atau berbayar mereka sendiri seperti aplikasi .Net, aplikasi Web, dan banyak lagi. Dalam organisasi perusahaan, edisi ini memiliki beberapa batasan. Misalnya, jika organisasi Anda memiliki lebih dari 250 PC dan memiliki pendapatan tahunan lebih dari $ 1 Juta (Dolar AS), Anda tidak diizinkan untuk menggunakan edisi ini. Dalam organisasi non-perusahaan, hingga lima pengguna dapat menggunakan edisi ini. Tujuan utamanya adalah untuk menyediakan dukungan Ekosistem (Akses ke ribuan ekstensi) dan Bahasa (Anda dapat membuat kode dalam C #, VB, F #, C ++, HTML, JavaScript, Python, dll.).
2. Profesional: Ini adalah edisi komersial Visual Studio. Itu datang dalam Visual Studio 2010 dan versi yang lebih baru. Ini memberikan dukungan untuk pengeditan XML dan XSLT dan termasuk alat seperti Server Explorer dan integrasi dengan Microsoft SQL Server. Microsoft menyediakan uji coba gratis edisi ini dan setelah masa uji coba, pengguna harus membayar untuk terus menggunakannya. Tujuan utamanya adalah untuk menyediakan Fleksibilitas (Alat pengembang profesional untuk membangun semua jenis aplikasi), Produktivitas (Fitur canggih seperti CodeLens meningkatkan produktivitas tim Anda), Kolaborasi (Alat perencanaan proyek yang tangkas, bagan, dll.) Dan manfaat Pelanggan seperti perangkat lunak Microsoft, ditambah Azure, Pluralsight, dll.

3. Perusahaan: Ini adalah solusi ujung ke ujung yang terintegrasi untuk tim dari berbagai ukuran dengan kebutuhan skala dan kualitas yang menuntut. Microsoft menyediakan uji coba gratis selama 90 hari untuk edisi ini dan setelah masa uji coba, pengguna harus membayar untuk terus menggunakannya. Manfaat utama edisi ini adalah bahwa edisi ini sangat dapat diskalakan dan menghadirkan perangkat lunak berkualitas tinggi.

2.10.2 Antarmuka Visual Studio

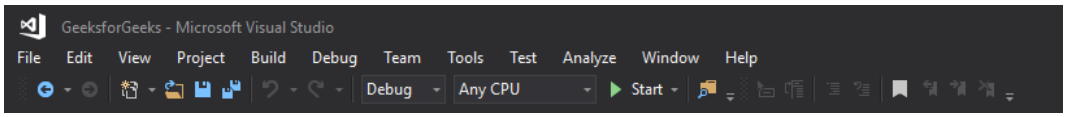
Antarmuka Visual Studio digunakan untuk melakukan pemrograman. Ada beberapa bagian yang terdapat dari tampilan Visual Studio, antara lain:

1. Editor Kode: Di mana pengguna akan menulis kode.
2. Output Window: Di sini Visual Studio menunjukkan output, peringatan compiler, pesan kesalahan dan informasi debugging.
3. Penjelajah Solusi: Ini menunjukkan file di mana pengguna saat ini bekerja.
4. Properti: Ini akan memberikan informasi dan konteks tambahan tentang bagian-bagian yang dipilih dari proyek saat ini.



Gambar 2.4 Antarmuka Visual Studio

Gambar 2.4 adalah antarmuka Visual Studio. Pengguna juga dapat menambahkan jendela sesuai kebutuhan dengan memilihnya dari menu View. Dalam Visual Studio, jendela alat dapat disesuaikan karena pengguna dapat menambahkan lebih banyak jendela, menghapus jendela yang ada atau dapat memindahkan jendela agar sesuai. Berbagai Menu di Visual Studio: Pengguna dapat menemukan banyak menu di layar atas Visual Studio seperti yang ditunjukkan pada gambar 2.5.



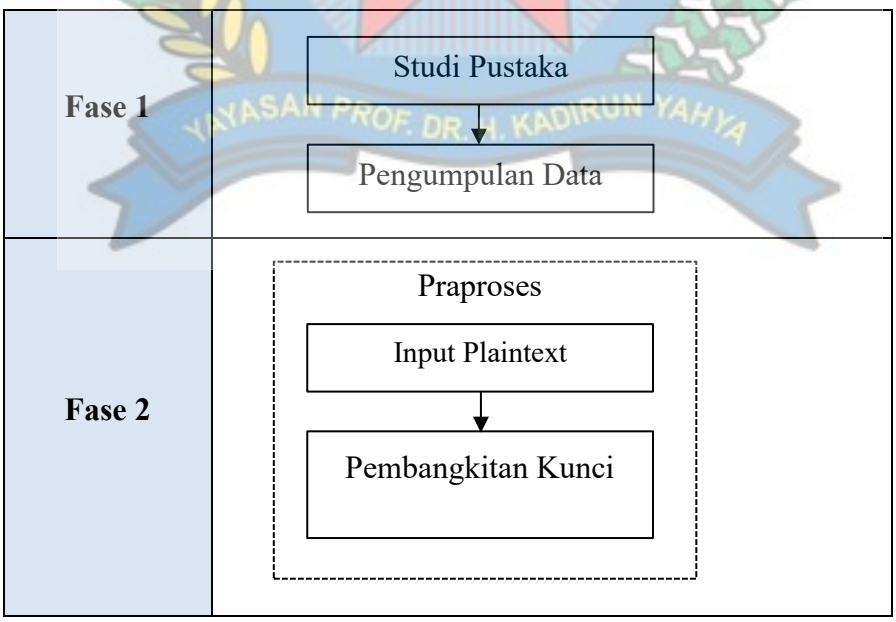
Gambar 2.5 Menubar Visual Studio

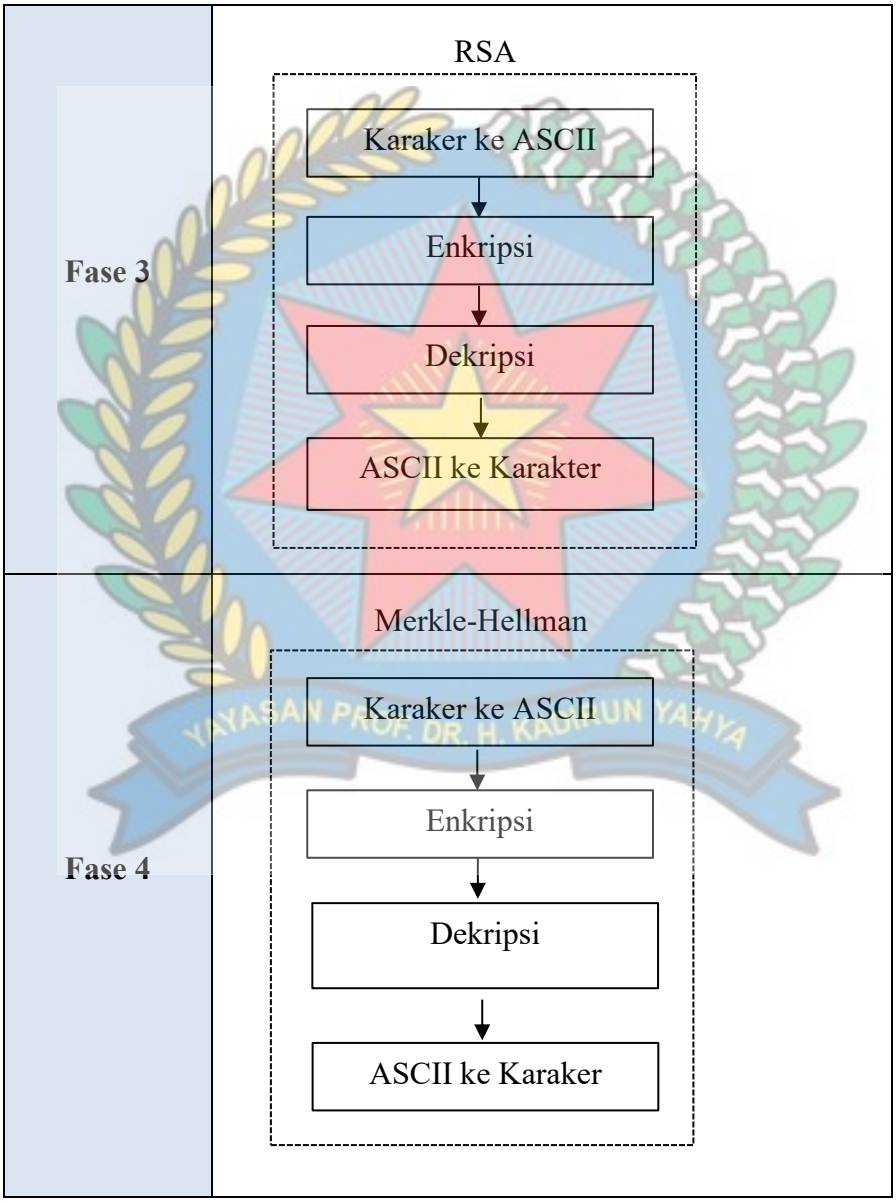
BAB III

METODE PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian dilakukan oleh penulis untuk mendapatkan alur yang lengkap dalam proses perbandingan kedua algoritma kriptografi. Gambar 3.1 akan menjelaskan secara lengkap tentang tahapan penelitian.





Gambar 3.1 Tahapan Penelitian

Berikut adalah tahapan penelitian yang dilakukan:

1. Studi Literatur

   Studi literatur berfungsi untuk memperoleh informasi tentang ilmu kriptografi khususnya algoritma RSA dan Merkle-Hellman. Studi ini dapat dilakukan melalui buku dan informasi yang ada di internet.

2. Analisa

   Analisa berfungsi untuk melihat perbandingan kedua algoritma dalam menjalankan proses enkripsi dan dekripsi berdasarkan *plaintext* yang disediakan.

3. Pembahasan

   Pembahasan menceritakan tentang formula yang digunakan oleh pengguna tentang algoritma RSA dan Merkle-Hellman untuk melakukan proses enkripsi dan dekripsi terhadap plaintext dan ciphertext.

4. Implementasi dan pengujian

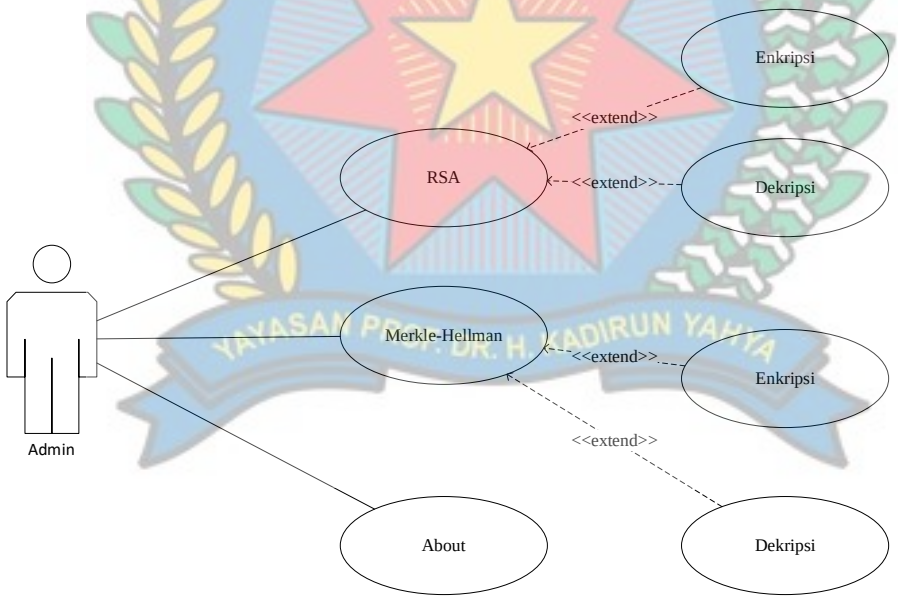
   Implementasi dan pengujian dilakukan pengujian aplikasi program yang dibangun dengan Microsoft Visual Basic.Net 2010. Pengujian memperlihatkan sejauh mana perbandingan algoritma RSA dan Merkle-Hellman dari segi kecepatan.

3.2 Perancangan Penelitian

Penelitian perlu dirancang dengan baik terutama pada bagian alur dan antarmuka aplikasi. Perancangan penelitian ditunjukkan dengan beberapa alur diagram yang dikenal dengan bentuk diagram *Unified Modelling Language (UML)*. Diagram yang dibuat menunjukkan proses dan cara kerja aplikasi yang akan dibangun kedepannya. *UML* memberi kemudahan dalam melakukan penelusuran apabila ada perubahan dari alur program aplikasi.

3.2.1 *Use Case Diagram*

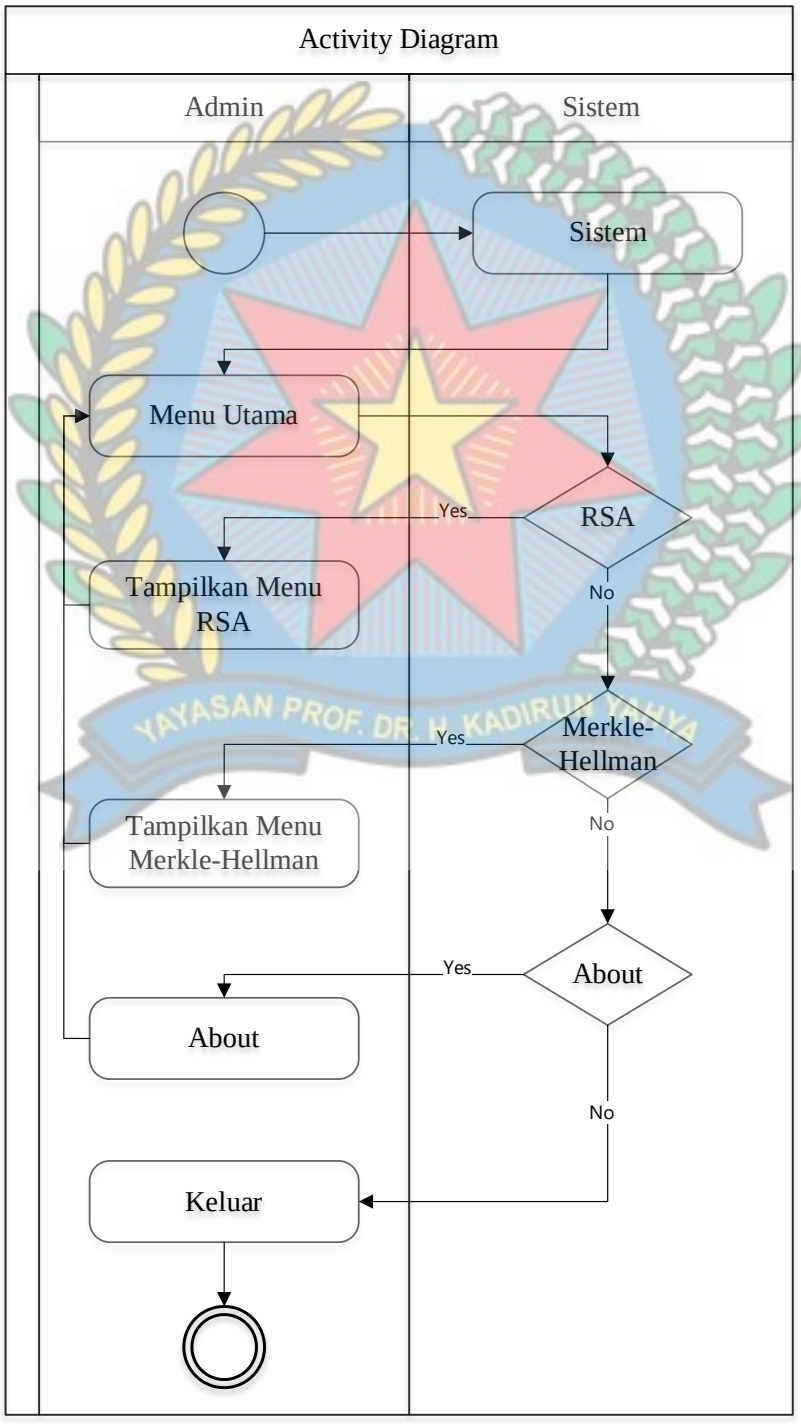
Use case diagram akan menjelaskan cara kerja dan fungsi dari program aplikasi kriptografi RSA dan Merkle-Hellman. Gamabr 3.1 adalah perancangan *use case diagram* penelitian yang dilakukan penulis.



Gambar 3.2 *Use case diagram* penelitian

3.2.2 *Activity Diagram*

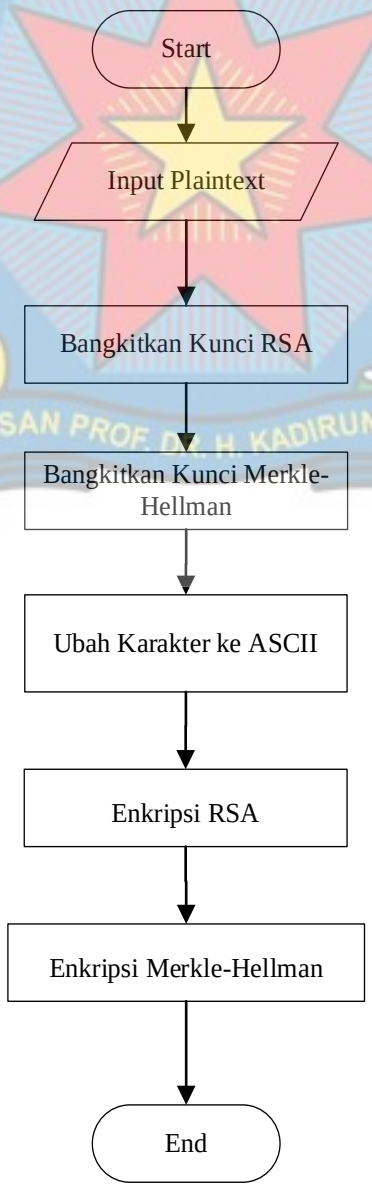
Activity Diagram menjelaskan proses kerja dari aplikasi kriptografi RSA dan Merkle-Hellman. *Activity diagram* menggambarkan keadaan dari suatu sistem dengan menguraikan setiap langkah yang akan dikerjakan pada program aplikasi tersebut. Gambar 3.2 menjelaskan *activity diagram* tersebut.



Gambar 3.3 *Activity diagram* penelitian

3.2.3 Flowchart Enkripsi

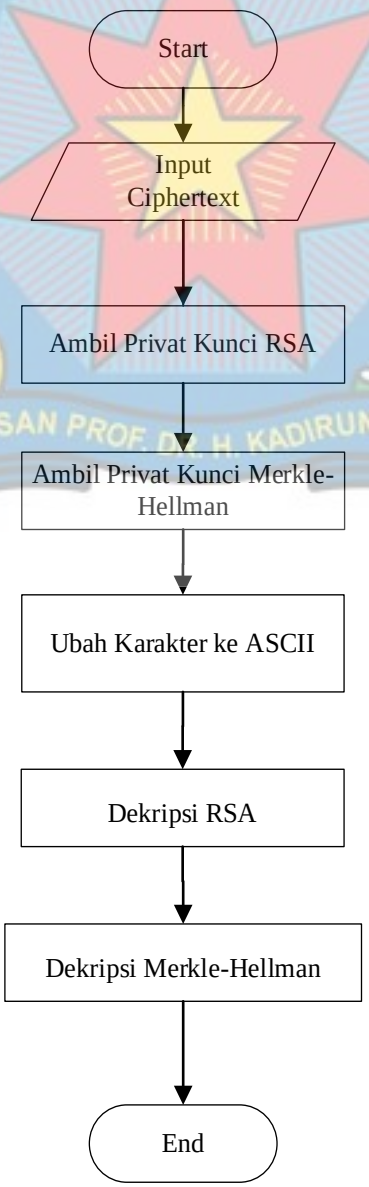
Flowchart enkripsi akan menerangkan alur proses enkripsi algoritma RSA dan Merkle-Hellman. *Flowchart* enkripsi dapat dilihat pada gambar 3.4.



Gambar 3.4 Flowchart enkripsi

3.2.4 Flowchart Dekripsi

Flowchart dekripsi menjelaskan alur proses dekripsi pada algoritma RSA dan Merkle-Hellman. *Flowchart* dekripsi dapat dilihat pada gambar 3.5.



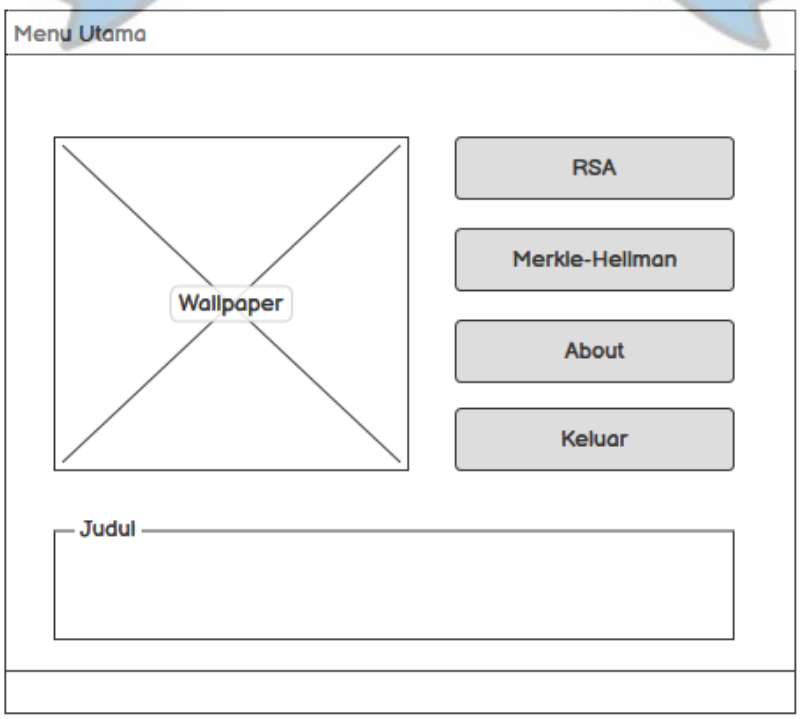
Gambar 3.5 Flowchart dekripsi

3.3 Perancangan Antarmuka

Antarmuka digunakan untuk menghubungkan pengguna dengan aplikasi kriptografi. Ada beberapa bagian dalam perancangan antarmuka yang dilakukan pada penelitian ini. Setiap antarmuka memiliki fungsi tertentu untuk menyelesaikan permasalahan.

3.3.1 Menu Utama

Menu utama adalah antarmuka yang ditampilkan ketika program aplikasi kriptografi RSA dan Merkle-Hellman dijalankan. Gambar 3.6 adalah hasil perancangan menu utama.



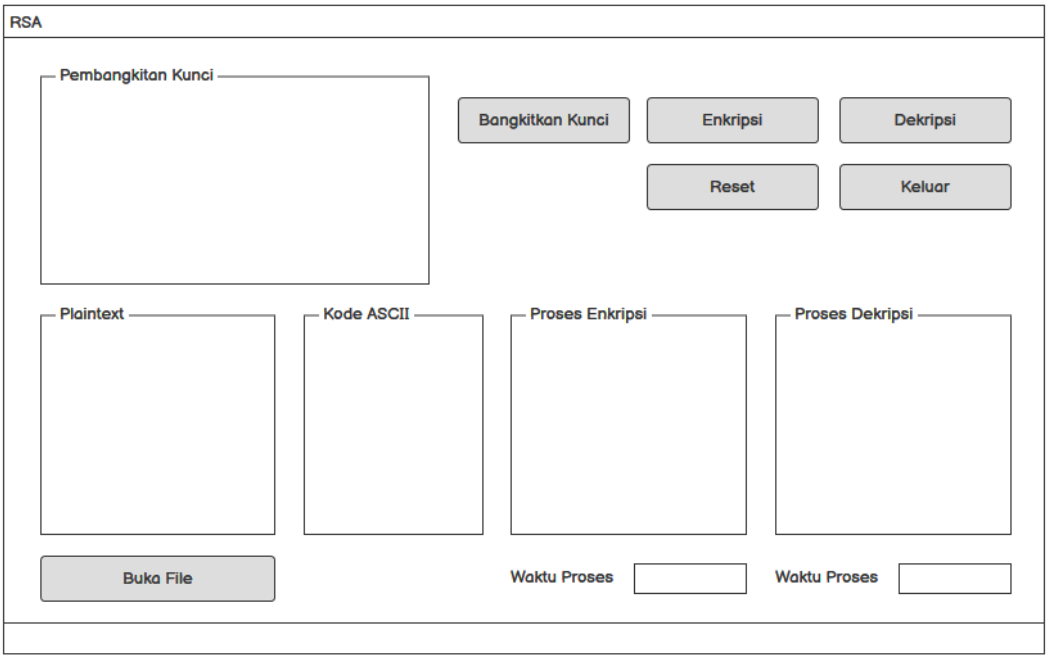
Gambar 3.6 Tampilan menu utama

Tampilan ini memiliki berapa sub-menu antara lain:

1. Wallpaper
2. RSA
3. Merkle-Hellman
4. About
5. Keluar
6. Judul Tugas Akhir

3.3.2 Menu RSA

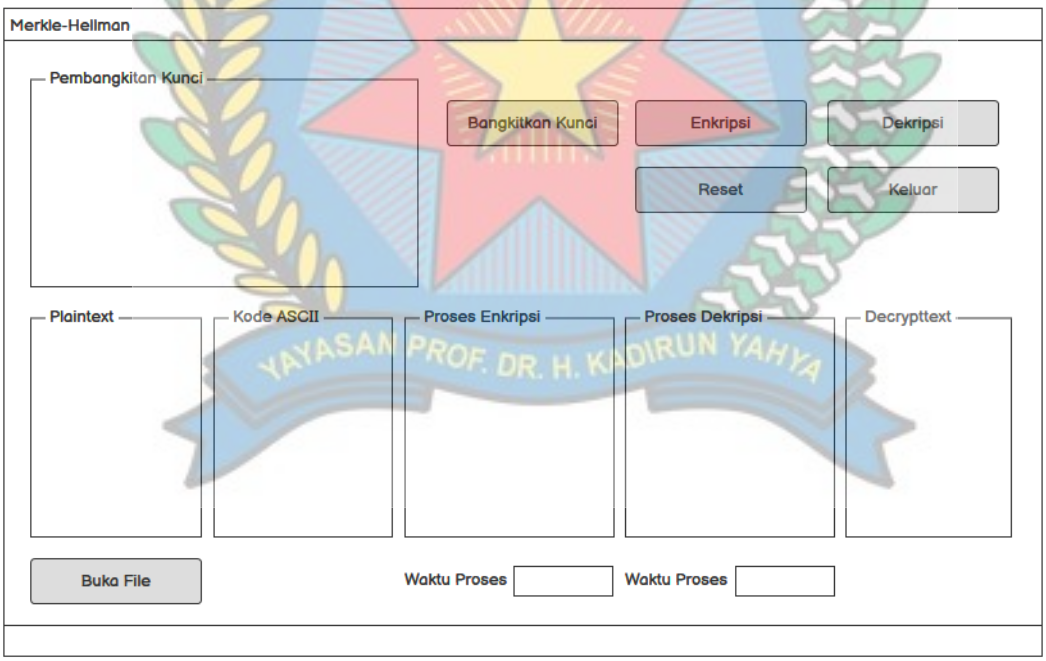
Menu RSA ini adalah perancangan untuk melakukan enkripsi dan dekripsi dengan menggunakan algoritma RSA. Gambar 3.7 adalah tampilan menu RSA.



Gambar 3.7 Tampilan menu RSA

3.3.3 Menu Merkle-Hellman

Menu Merkle-Hellman ini adalah perancangan untuk melakukan enkripsi dan dekripsi dengan menggunakan algoritma Merkle-Hellman. Gambar 3.8 adalah tampilan menu Merkle-Hellman.



Gambar 3.8 Tampilan menu Merkle-Hellman

3.3.4 Menu About

Menu ini menampilkan informasi penulis dan deskripsi tentang algoritma RSA dan Merkle-Hellman. Gambar 3.9 adalah hasil tampilan dari menu about.

Gambar 3.9 Tampilan Menu About

BAB IV

HASIL DAN PEMBAHASAN

Program aplikasi kriptografi yang penulis bangun berfungsi untuk melihat proses dan hasil dari enkripsi dan dekripsi. Kebutuhan sistem sangat diperlukan dalam menjamin agar program aplikasi bekerja dengan baik.

4.1 Kebutuhan Sistem

Kebutuhan sistem terdiri dari kebutuhan perangkat keras dan perangkat lunak. Setiap perangkat harus saling bekerja sama agar tidak terjadi kelemahan sistem. Kebutuhan sistem yang digunakan pada penelitian ini, antara lain:

1. Kebutuhan perangkat keras
2. Kebutuhan perangkat lunak

4.1.1 Kebutuhan Perangkat Keras

Program RSA dan Merkle-Hellman memerlukan perangkat keras yang baik. Tabel 4.1 adalah spesifikasi minium dari perangkat keras yang digunakan.

Tabel 4.1 Spesifikasi perangkat keras

No.	Komponen	Spesifikasi
1	Processor	Intel Core i3 2.4 GHz
2	RAM	2048 MB
3	Penyimpanan	320 GB
4	Display	14 inch

4.1.2 Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan pada penelitian ini terdiri dari beberapa item. Tabel 4.2 adalah spsesifikasi perangkat lunak yang digunakan pada penelitian ini.

Tabel 4.2 Spesifikasi perangkat lunak

No.	Komponen	Spesifikasi
1	Sistem Operasi	Windows 10 64 Bit
2	IDE Pemrograman	Microsoft Visual Basic.NET 2010
3	Word Editor	Microsoft Word
4	Data Editor	Microsoft Excel

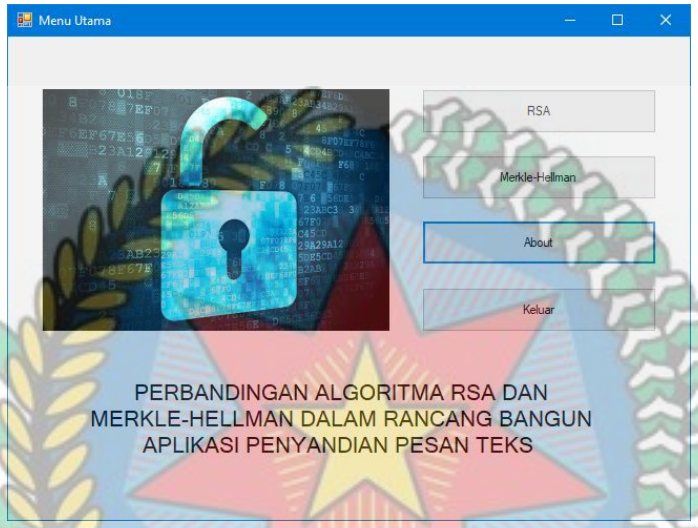
4.2 Implementasi Sistem

Implementasi merupakan hasil program aplikasi yang terdiri dari beberapa bagian antarmuka. Ada beberapa menu yang terdapat dari program aplikasi yang dibangun pada penelitian ini.

4.2.1 Halaman Menu Utama

Halaman *Menu Utama* merupakan halaman pada program aplikasi RSA dan Merkle-Hellman. Gambar 4.1 adalah hasil tampilan *Menu Utama*. Halaman menu memiliki beberapa menu yang terhubung, antara lain:

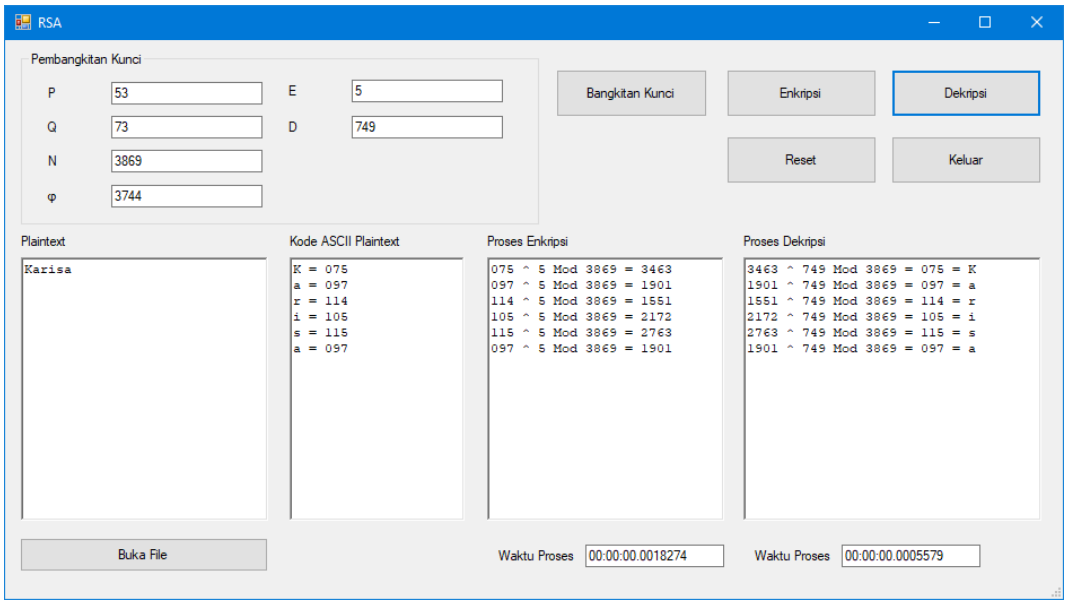
1. RSA
2. Merkle-Hellman
3. About
4. Keluar



Gambar 4.1 Halaman Menu Utama

4.2.2 Halaman RSA

Halaman *RSA* adalah menu yang menampilkan proses kriptografi RSA. Gambar 4.2 adalah hasil tampilan dari halaman *RSA*.



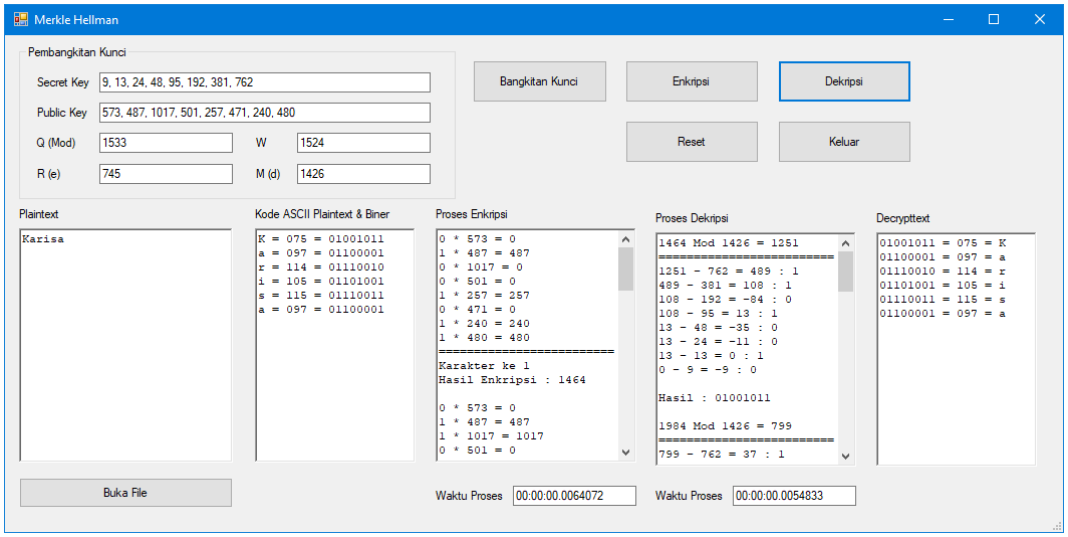
Gambar 4.2 Halaman RSA

Urutan proses enkripsi dan dekripsi yang dilakukan oleh algoritma RSA, antara lain:

1. Menyediakan plaintext yang akan diproses.
2. Mengubah plaintext menjadi kode ASCII dan biner.
3. Melakukan proses enkripsi.
4. Melakukan proses dekripsi.
5. Mengembalikan nilai biner ke bentuk ASCII dan karakter.

4.2.3 Halaman Merkle-Hellman

Halaman *Merkle-Hellman* menampilkan proses kriptografi dari algoritma Merkle-Hellman yang digunakan pada penelitian ini. Gambar 4.3 adalah tampilan dari halaman *Merkle-Hellman*.



Gambar 4.3 Halaman Merkle-Hellman

Urutan proses enkripsi dan dekripsi yang dilakukan oleh algoritma Merkle-Hellman, antara lain:

1. Menyediakan plaintext yang akan diproses.
2. Mengubah plaintext menjadi kode ASCII dan biner.
3. Melakukan proses enkripsi.
4. Melakukan proses dekripsi.
5. Mengembalikan nilai biner ke bentuk ASCII dan karakter.

4.2.4 Halaman About

Halaman *About* merupakan halaman yang berfungsi untuk menampilkan biodata penulis. Gambar 4.4 adalah hasil tampilan dari halaman *About*.

Gambar 4.4 Halaman About

4.3 Pengujian

Pengujian pada penelitian ini dilakukan untuk melihat perhitungan manual yang dilakukan oleh kedua algoritma. Bagian berikut ini melihat bagaimana proses enkripsi dan dekripsi dilakukan oleh algoritma RSA dan Merkle-Hellman.

Perhitungan RSA

Plaintext : Karisa

Pembangkitan Kunci

```
P          :  53
Q          :  73
N          :  53 x 73
           :  3869
T          :  (53 - 1) x (73 - 1)
           :  3744
E          :  5
D          :  749
```

ASCII

```
K = 075
a = 097
r = 114
i = 105
s = 115
a = 097
```

Proses Enkripsi

```
075 ^ 5 Mod 3869 = 3463
097 ^ 5 Mod 3869 = 1901
114 ^ 5 Mod 3869 = 1551
105 ^ 5 Mod 3869 = 2172
115 ^ 5 Mod 3869 = 2763
097 ^ 5 Mod 3869 = 1901
```

Proses Dekripsi

```
3463 ^ 749 Mod 3869 = 075 = K
1901 ^ 749 Mod 3869 = 097 = a
1551 ^ 749 Mod 3869 = 114 = r
2172 ^ 749 Mod 3869 = 105 = i
2763 ^ 749 Mod 3869 = 115 = s
1901 ^ 749 Mod 3869 = 097 = a
```

Perhitungan Merkle-Hellman

Plaintext : Karisa

Pembangkitan Kunci

Secret Key : 9, 13, 24, 48, 95, 192, 381, 762
Public Key : 573, 487, 1017, 501, 257, 471, 240, 480

ASCII

K = 075 = 01001011
a = 097 = 01100001
r = 114 = 01110010
i = 105 = 01101001
s = 115 = 01110011
a = 097 = 01100001

Proses Enkripsi

0 * 573 = 0
1 * 487 = 487
0 * 1017 = 0
0 * 501 = 0
1 * 257 = 257
0 * 471 = 0
1 * 240 = 240
1 * 480 = 480
==========
Karakter ke 1
Hasil Enkripsi : 1464

0 * 573 = 0
1 * 487 = 487
1 * 1017 = 1017
0 * 501 = 0
0 * 257 = 0
0 * 471 = 0
0 * 240 = 0
1 * 480 = 480
==========
Karakter ke 2
Hasil Enkripsi : 1984

0 * 573 = 0
1 * 487 = 487
1 * 1017 = 1017
1 * 501 = 501
0 * 257 = 0
0 * 471 = 0
1 * 240 = 240
0 * 480 = 0
==========
Karakter ke 3
Hasil Enkripsi : 2245

0 * 573 = 0
1 * 487 = 487
1 * 1017 = 1017
0 * 501 = 0
1 * 257 = 257
0 * 471 = 0
0 * 240 = 0
1 * 480 = 480
==========

```
Karakter ke 4
Hasil Enkripsi : 2241

0 * 573 = 0
1 * 487 = 487
1 * 1017 = 1017
1 * 501 = 501
0 * 257 = 0
0 * 471 = 0
1 * 240 = 240
1 * 480 = 480
=========================
Karakter ke 5
Hasil Enkripsi : 2725

0 * 573 = 0
1 * 487 = 487
1 * 1017 = 1017
0 * 501 = 0
0 * 257 = 0
0 * 471 = 0
0 * 240 = 0
1 * 480 = 480
=========================
Karakter ke 6
Hasil Enkripsi : 1984
```

Proses Dekripsi

```
1464 Mod 1426 = 1251
=========================
1251 - 762 = 489 : 1
489 - 381 = 108 : 1
108 - 192 = -84 : 0
108 - 95 = 13 : 1
13 - 48 = -35 : 0
13 - 24 = -11 : 0
13 - 13 = 0 : 1
0 - 9 = -9 : 0

Hasil : 01001011

1984 Mod 1426 = 799
=========================
799 - 762 = 37 : 1
37 - 381 = -344 : 0
37 - 192 = -155 : 0
37 - 95 = -58 : 0
37 - 48 = -11 : 0
37 - 24 = 13 : 1
13 - 13 = 0 : 1
0 - 9 = -9 : 0

Hasil : 01100001

2245 Mod 1426 = 466
=========================
466 - 762 = -296 : 0
466 - 381 = 85 : 1
85 - 192 = -107 : 0
85 - 95 = -10 : 0
85 - 48 = 37 : 1
37 - 24 = 13 : 1
13 - 13 = 0 : 1
0 - 9 = -9 : 0
```

```
Hasil : 01110010

2241 Mod 1426 = 894
=========================
894 - 762 = 132 : 1
132 - 381 = -249 : 0
132 - 192 = -60 : 0
132 - 95 = 37 : 1
37 - 48 = -11 : 0
37 - 24 = 13 : 1
13 - 13 = 0 : 1
0 - 9 = -9 : 0

Hasil : 01101001

2725 Mod 1426 = 1228
=========================
1228 - 762 = 466 : 1
466 - 381 = 85 : 1
85 - 192 = -107 : 0
85 - 95 = -10 : 0
85 - 48 = 37 : 1
37 - 24 = 13 : 1
13 - 13 = 0 : 1
0 - 9 = -9 : 0

Hasil : 01110011

1984 Mod 1426 = 799
=========================
799 - 762 = 37 : 1
37 - 381 = -344 : 0
37 - 192 = -155 : 0
37 - 95 = -58 : 0
37 - 48 = -11 : 0
37 - 24 = 13 : 1
13 - 13 = 0 : 1
0 - 9 = -9 : 0

Hasil : 01100001


01001011 = 075 = K
01100001 = 097 = a
01110010 = 114 = r
01101001 = 105 = i
01110011 = 115 = s
01100001 = 097 = a
```


4.4 Pembahasan

Pembahasan pada penelitian ini dilakukan untuk melihat perbandingan kecepatan yang dilakukan oleh algoritma RSA dan Merkle-Hellman. Pengujian dilakukan dengan cara memberikan sejumlah pesan teks terhadap kedua algoritma tersebut. Berikut ini adalah hasil perbandingan kedua algoritma.

Tabel 4.3 Hasil perbandingan Kecepatan

No.	Ukuran Pesan (byte)	Enkripsi RSA	Dekripsi RSA	Enkripsi Merkle-Hellman	Dekripsi Merkle-Hellman
1	662	8	7	295	911
2	1325	21	18	1601	3778
3	2650	123	116	6270	15101
4	10600	2131	2024	122048	299052

Dari tabel 4.3 dapat diambil kesimpulan bahwa algoritma RSA lebih cepat dalam melakukan proses enkripsi dan dekripsi. Hasil enkripsi dan dekripsi diukur dalam satuan milidetik untuk kedua algoritma. Setelah mendapatkan hasil pengukuran, dapat dilihat dari waktu proses yang dilakukan dalam satuan milidetik. Proses enkripsi dan dekripsi Merkle-Hellman memiliki tahapan yang lebih banyak dari algoritma RSA karena pada algoritma Merkle-Hellman, karakter yang akan diproses akan diproses dalam satuan bit sehingga ada delapan proses yang dilakukan pada algoritma ini. Dari sini dapat dilihat bahwa algoritma RSA memiliki kecepatan lebih kurang delapan kali lebih cepat dari algoritma Merkle-Hellman.

BAB V

PENUTUP

5.1 Kesimpulan

Penulis telah melakukan penelitian dan berhasil sehingga dapat menarik beberapa kesimpulan, antara lain:

1. Aplikasi kriptografi RSA dan Merkle-Hellman dibangun menggunakan Microsoft Visual Basic Net.
2. Kecepatan proses enkripsi dan dekripsi RSA memiliki kecepatan yang lebih baik dari pada kecepatan pada algoritma Merkle-Hellman.
3. Proses enkripsi dan dekripsi dari kedua algoritma menggunakan karakter ASCII dalam proses perhitungan matematika dalam mendapatkan nilai dari *ciphertext.*

5.2 Saran

Penelitian masih perlu dikembangkan agar menjadi lebih baik. Adapun saran tersebut adalah antara lain:

1. Sebaiknya program aplikasi dapat menyimpan hasil enkripsi menjadi sebuah file.
2. Sebaiknya menambah jumlah karakter yang dapat diproses dan dapat memproses berbagai macam tipe file selain “.txt”.

DAFTAR PUSTAKA

Arton. (2021). *All About Passports*. Passport Index. https://www.passportindex.org/passport.php

Ayushi, M. (2020). A Symmetric Key Cryptographic Algorithm. *International Journal of Computer Applications*, *1*(15), 1–6. https://doi.org/10.5120/331-502

Barot, S. (2018). *Why Is Data Security Important for Everyone?* Security Zone. https://dzone.com/articles/why-is-data-security-important-for-everyone

Gurevich, Y. (2012). *What Is an Algorithm?* (pp. 31–42). https://doi.org/10.1007/978-3-642-27660-6_3

Jovancic, N. (2019). *5 Data Collection Methods for Obtaining Quantitative and Qualitative Data*. LeadQuizzes. https://www.leadquizzes.com/blog/data-collection-methods/

Kurniawan, T. A. (2018). Pemodelan Use Case (UML): Evaluasi Terhadap beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(1), 77. https://doi.org/10.25126/jtiik.201851610

Ladjamudin, A.-B. bin. (2017). *Analisis dan Desain Sistem Informasi*. Graha Ilmu.

Nakatsu, R. T. (2019). *Reasoning with Diagrams : Decision-Making and Problem-Solving with Diagrams.* John Wiley & Sons.

Nurgoho, A. (2019). *Rekayasa Perangkat Lunak Menggunakan UML dan JAVA*. Andi Offset.

Rao, R. V., & Selvamani, K. (2015). Data Security Challenges and Its Solutions in Cloud Computing. *Procedia Computer Science*, *48*, 204–209. https://doi.org/10.1016/j.procs.2015.04.171

S., G., L. Ribeiro, A. R., & David, E. (2012). Asymmetric Encryption in Wireless Sensor Networks. In *Wireless Sensor Networks - Technology and Protocols*. InTech. https://doi.org/10.5772/48464

Sukmawati, R., & Priyadi, Y. (2019). Perancangan Proses Bisnis Menggunakan UML Berdasarkan Fit/Gap Analysis Pada Modul Inventory Odoo. *INTENSIF: Jurnal Ilmiah Penelitian Dan Penerapan Teknologi Sistem Informasi*, *3*(2), 104. https://doi.org/10.29407/intensif.v3i2.12697

Technopedia. (2019). *Unified Modeling Language (UML)*. Technopedia.

https://www.techopedia.com/definition/3243/unified-modeling-language-uml

UTM. (2019). *Concept: Use-Case Model*. Univesidad Technologica de La Mixteca. http://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use_case_model_CD178AF9.html

Wasserkrug, S., Dalvi, N., Munson, E. V., Gogolla, M., Sirangelo, C., Fischer-Hübner, S., Ives, Z., Velegrakis, Y., Bevan, N., Jensen, C. S., & Snodgrass, R. T. (2019). Unified Modeling Language. In *Encyclopedia of Database Systems* (pp. 3232–3239). Springer US. https://doi.org/10.1007/978-0-387-39940-9_440

Zandbergen, P. (2019). *What is a Computer Algorithm?* Study.Com. https://study.com/academy/lesson/what-is-a-computer-algorithm-design-examples-optimization.html