PENGEMBANGAN APLIKASI *E-VOTING* MENGGUNAKAN ENKRIPSI HOMOMORFIK

Laporan Tugas Akhir

Disusun Sebagai Syarat Kelulusan Sarjana

Oleh

MUHTAR HARTOPO

NIM : 13513068

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO & INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2017

PENGEMBANGAN APLIKASI *E-VOTING* MENGGUNAKAN ENKRIPSI HOMOMORFIK

Draft Laporan Tugas Akhir

Oleh

MUHTAR HARTOPO

NIM : 13513068

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

Telah disetujui dan disahkan sebagai Laporan Tugas Akhir

Bandung, 16 Agustus 2017

Mengetahui,

Pembimbing,

Dr. Ir. Rinaldi Munir, ST, MT

NIP. 196512101994021001

LEMBAR PERNYATAAN

Dengan ini saya menyatakan bahwa:

1. Pengerjaan dan penulisan Laporan Tugas Akhir ini dilakukan tanpa menggunakan bantuan yang tidak dibenarkan.
2. Segala bentuk kutipan dan acuan terhadap tulisan orang lain yang digunakan di dalam penyusunan laporan tugas akhir ini telah dituliskan dengan baik dan benar.
3. Laporan Tugas Akhir ini belum pernah diajukan pada program pendidikan di perguruan tinggi mana pun.

Jika terbukti melanggar hal-hal di atas, saya bersedia dikenakan sanksi sesuai dengan Peraturan Akademik dan Kemahasiswaan Institut Teknologi Bandung bagian Penegakan Norma Akademik dan Kemahasiswaan khususnya Pasal 2.1 dan Pasal 2.2.

Bandung, 16 Agustus 2017

Muhtar Hartopo
NIM 13513068

ABSTRAK

PENGEMBANGAN APLIKASI *E-VOTING*

MENGGUNAKAN ENKRIPSI HOMOMORFIK

Oleh

MUHTAR HARTOPO

NIM : 13513068

Aspek kehidupan berdemokrasi telah dipengaruhi oleh perkembangan teknologi, salah satunya adalah penerapan *Electronics voting* atau pemilihan yang bersifat elektronik. *E-Voting* memiliki keuntungan yaitu meningkatkan efisiensi dari segi waktu, biaya dan mengurangi kesalahan perhitungan. Namun di samping keuntungan tersebut terdapat risiko keamanan yang menjadi kendala seperti risiko kebocoran dan manipulasi data. Algoritma kriptografi dapat diterapkan untuk mengatasi masalah ini.

Enkripsi homomorfik merupakan suatu bentuk enkripsi yang memungkinkan dilakukannya komputasi pada *ciphertext* tanpa mendekripsi *ciphertext* tersebut terlebih dahulu. Algoritma kriptografi Pailier adalah algoritma kriptografi yang bersifat homomorfik parsial. Algoritma Pailier mendukung operasi homomorfik untuk penjumlahan. Pada tugas akhir ini dibangun sebuah aplikasi *e-voting* yang memanfaatkan sifat homomorfis algoritma Pailier. Aplikasi tersebut terdiri atas dua bagian yaitu aplikasi untuk sisi klien dan untuk sisi server. Aplikasi untuk klien berperan untuk melakukan enkripsi data pada sisi klien sebelum mengirimkannya ke server. Aplikasi pada bagian *server* berguna untuk melakukan perhitungan homomorfis yang berperan untuk melakukan rekapitulasi suara. Proses enkripsi hanya dilakukan di awal yaitu enkripsi saat pemilih melakukan pemilihan dan proses dekripsi hanya dilakukan di akhir yaitu untuk melihat hasil pemilihan. Proses rekapitulasi suara di *server* tidak melibatkan proses dekripsi sama sekali.

Aplikasi yang dibangun menggunakan bahasa pemrograman Java. Klien berbasis GUI dibuat menggunakan JavaFX dan server yang berupa *voting place* dan *intermediate level* dikembangkan menggunakan *Spring Framework*. Hasil pengujian yang dilakukan menunjukkan bahwa aplikasi *e-voting* yang dibangun memiliki kualitas keamanan yang cukup baik terutama dari sisi kriptografi dan kesesuaian dengan asas-asas pemilihan umum yaitu dengan nilai *maturity level* masing-masing 5 dan 6.

Kata kunci: aplikasi, *e-voting*, enkripsi, homomorfik, pailier.

KATA PENGANTAR

Puji syukur kami panjatkan kehadirat Allah SWT, karena atas limpahan rahmat dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul "Pengembangan Aplikasi *E-Voting* Menggunakan Enkripsi Homomorfik". Tugas Akhir ini disusun sebagai persyaratan kelulusan pada Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika Bandung.

Dalam penyusunan Tugas Akhir ini penulis banyak mendapat saran, dorongan, bimbingan serta keterangan-keterangan dari berbagai pihak yang merupakan pengalaman yang tidak dapat diukur secara materi, namun dapat membukakan mata penulis bahwa sesungguhnya pengalaman dan pengetahuan tersebut adalah guru yang terbaik bagi penulis. Oleh karena itu dengan segala hormat dan kerendahan hati perkenankanlah penulis mengucapkan terima kasih kepada :

1. Bapak Dr. Ir. Rinaldi Munir M.T. selaku pembimbing yang senantiasa memberikan arahan dan masukan selama pengerjaan Tugas Akhir
2. Bapak Dr. Ir. Rila Mandala, M.Eng. dan bapak Dr.Techn. Muhammad Zuhri Catur Candra, S.T., M.T. selaku dosen penguji yang atas saran dan masukannya yang membuat Tugas Akhir ini menjadi lebih baik.
3. Kedua orang tua penulis dan semua anggota keluarga lainnya. Terima kasih atas doa, dukungan, nasihat dan kasih sayangnya selama ini yang tidak pernah ada hentinya.
4. Bapak Ir. Windy Gambetta selaku dosen wali yang telah membimbing penulis selama menempuh perkuliahan di Teknik Informatika ITB.
5. Seluruh dosen Teknik Informatika ITB atas ilmu pelajaran hidup yang telah diberikan selama ini yang akan bermanfaat untuk kehidupan penulis seterusnya.
6. Sahabat-sahabat perantau dari Sulawesi Selatan yang tergabung dalam Unit Kesenian Sulawesi Selatan. Terima kasih telah berbagi suka dan duka selama menjalani perkuliahan di ITB.

7. Teman-teman Teknik Informatika ITB angkatan 2013 yang senantiasa berbagi ilmu, pengalaman, cerita, suka dan duka selama menjalani perkuliahan di Teknik Informatika ITB.
8. Semua pihak yang membantu pengerjaan Tugas Akhir ini baik secara langsung maupun secara tidak langsung.

Dalam penyusunan Tugas Akhir ini, penulis menyadari masih terdapat banyak kekurangan yang dibuat baik sengaja maupun tidak sengaja, dikarenakan keterbatasan ilmu pengetahuan dan wawasan serta pengalaman yang penulis miliki. Untuk itu penulis mohon maaf atas segala kekurangan tersebut tidak menutup diri terhadap segala saran dan kritik yang bersifat konstruktif bagi diri penulis.

Akhir kata semoga tugas akhir ini dapat bermanfaat bagi penulis sendiri, institusi pendidikan dan masyarakat luas.

Bandung, Agustus 2017

Penulis

DAFTAR ISI

DAFTAR LAMPIRAN

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

I.1 Latar Belakang

Perkembangan teknologi informasi dan komunikasi telah banyak mengubah cara hidup manusia. Perubahan tersebut dapat dirasakan mulai dari cara berkomunikasi, cara berbelanja, cara belajar mengajar dan masih banyak lagi. Perubahan-perubahan tersebut didasari karena keinginan manusia untuk melakukan pekerjaan dengan lebih efisien.

Dalam aspek kehidupan berdemokrasi pun telah dipengaruhi oleh perkembangan teknologi, salah satunya adalah penerapan *electronic voting* pada pemilihan. *Electronic voting* atau yang biasa disingkat *e-voting* adalah penggunaan komputer atau komputerisasi pada proses pemungutan suara pada pemilihan (Kahani, 2005). Penggunaan *e-voting* sendiri telah digunakan di berbagai instansi dan negara. Beberapa negara yang telah menggunakan sistem *e-voting* di antaranya Brazil, Kanada, Jerman, Prancis, Belanda, Swiss dan beberapa negara maju lainnya. Tak hanya pada tingkat negara, sistem *e-voting* juga dapat diterapkan pada perusahaan, komunitas ataupun organisasi mahasiswa. Penggunaan *e-voting* tersebut dinilai menguntungkan karena dapat menghemat anggaran, menghemat waktu pemilihan, mempermudah perhitungan suara dan lain-lainnya. *E-voting* sendiri telah diterapkan di sebuah kabupaten di Indonesia dan hasilnya memuaskan serta dapat menurunkan anggaran pemilihan.

Penggunaan *e-voting* juga akan semakin dipermudah dengan semakin banyaknya penyedia jasa internet *cloud*. Layanan ini memungkinkan pihak instansi pemilihan untuk mengembangkan sistem *e-voting* sendiri di atas platform yang disediakan. Layanan *e-voting* melalui internet selain mengurangi biaya juga dapat meningkatkan partisipasi pada pemilih sebab tidak perlu datang dan antre di tempat pemungutan suara. Sistem *e-voting* memungkinkan pemilih yang sah dapat memilih dimanapun dan kapanpun selama ada akses internet.

Disamping banyaknya manfaat penggunaan sistem *e-voting* pada proses pemilihan, terdapat pula risiko yang mengancam. Beberapa diantara risiko tersebut adalah masalah keamanan dan masalah kerahasiaan pemilihan. Masalah keamanan yang dapat terjadi contohnya pencurian data pemilihan, penyadapan dan manipulasi suara. Kemudian pada masalah privasi juga menjadi masalah serius. Data yang tersimpan pada basis data bisa saja data *plaintext* , pemilih dan pilihannya akan terpampang jelas di basis data. Hal ini tidak sesuai dengan prinsip pemilihan yang bersifat rahasia. Kemudian muncul solusi agar data pemilih tersebut dienkripsi sebelum disimpan di basis data. Solusi tersebut cukup bagus, namun akan sulit apabila akan dilakukan proses pengolahan pada data tersebut karena data tersebut harus didekripsi menjadi *plaintext* terlebih dahulu kemudian dapat diolah. Proses dekripsi data di komputer sebelum diolah akan memiliki risiko keamanan. Proses pengiriman dan rekapitulasi yang cukup panjang juga akan meningkatkan risiko pencurian data. Risiko tersebut akan meningkat jika sistem pemilihan tersebut berjalan bukan pada komputer server milik pribadi.

Salah satu solusi untuk mengatasi permasalahan privasi dan keamanan pada *e-voting* adalah dengan menyimpan data yang terenkripsi lalu melakukan pengolahan pada data yang terenkripsi tersebut tanpa perlu melakukan proses dekripsi terlebih dahulu. Dengan menyimpan dan mengolah data dalam bentuk terenkripsi kita tidak perlu takut apabila data tersebut dicuri. Konsep untuk melakukan proses komputasi tertentu pada pesan terenkripsi disebut enkripsi homomorfik (Potzelsberger, 2013).

Enkripsi homomorfik merupakan suatu bentuk enkripsi yang memungkinkan dilakukannya komputasi pada *ciphertext* tanpa mendekripsi terlebih dahulu *ciphertext* tersebut. Operasi yang dilakukan pada *ciphertext* yang menggunakan enkripsi homomorfik akan menghasilkan *ciphertext* yang jika didekripsi akan menghasilkan hasil yang sama dengan operasi serupa pada *plaintext*. Cara ini dapat dilakukan untuk menambah keamanan pada *voting system* dan *cloud computing* (Morris, 2013).

Skema yang dapat digunakan untuk enkripsi homomorfik ada dua macam yaitu *partially homomorphic encryption* (PHE) dan *Filly homomorpfic encryption*

(FHE). PHE merupakan jenis enkripsi homomorfik yang memungkinkan dilakukannya satu jenis operasi tertentu pada *ciphertext*. Sementara itu FHE merupakan jenis enkripsi homomorfik yang memungkinkan kedua jenis operasi penjumlahan dan perkalian dilakukan pada *ciphertext* (Poetzelberger, 2013).

Penggunaan enkripsi homomorfik pada sistem *e-voting* dapat menjadi solusi untuk masalah keamanan pada aplikasi web, khususnya pada layanan server yang disediakan oleh pihak ketiga. Layanan server yang sifatnya publik memiliki keuntungan diantaranya lebih murah dan lebih mudah karena kita tidak perlu memilik server sendiri dan tidak perlu repot melakukan *maintenance* server. Enkripsi homomorfik dapat meningkatkan keamanan pengiriman data dan pengolahan data pada suatu komputer. Dengan skema cara ini kita dapat mengambil keuntungan-keuntungan dari layanan server yang sifatnya publik bahkan untuk menyimpan data yang sifatnya privat sekalipun.

Layanan yang ada saat ini belum menyediakan penggunaan enkripsi homomorfik. Bahkan masih sangat sulit untuk menemukan aplikasi di dunia nyata yang menggunakan enkripsi homomorfik. Oleh karena itu jika kita ingin menggunakan skema tersebut, kita harus membuat sendiri program untuk menjalankan enkripsi homomorfik. Hal ini tentu akan cukup berat dan menambah beban pekerjaan.

Tugas akhir yang berjudul Pengembangan Aplikasi *E-voting* Menggunakan Enkripsi Homomorfik. Sesuai dengan judulnya, Tugas Akhir ini akan membahas mengenai pengaplikasian konsep enkripsi homomorfik yang diterapkan pada sistem *e-voting*.

I.2 Rumusan Masalah

Masalah utama yang akan dibahas pada tugas akhir ini adalah mengenai bagaimana mengimplementasikan konsep enkripsi homomorfik pada data pemilihan yang terenkripsi yang dibuat dalam sebuah aplikasi. Masalah utama tersebut dapat diturunkan menjadi beberapa masalah yang lebih detail sebagai berikut :

1. Bagaimana menerapkan konsep enkripsi homomorfik pada pengolahan data pemilihan.

2. Apa skema enkripsi homomorfik yang cocok untuk diterapkan pada pengolahan data pemilihan
3. Bagaimana mengimplementasikan konsep enkripsi homomorfik pada sistem *e-voting*.
4. Bagaimana keamanan skema enkripsi yang digunakan.

I.3 Tujuan

Tujuan penulisan Tugas Akhir ini adalah sebagai berikut :

1. Mempelajari bagaimana menerapkan konsep enkripsi homomorfik pada pengolahan data pemilihan.
2. Menentukan skema enkripsi homomorfik yang cocok diterapkan pada pengolahan data pemilihan
3. Mengimplementasikan enkripsi homomorfik pada data pemilihan dalam bentuk aplikasi.
4. Menguji keamanan data pada skema enkripsi yang digunakan

I.4 Batasan Masalah

Dalam penyusunan Tugas Akhir ini, terdapat batasan permasalahan yang digunakan yaitu :

1. Aplikasi yang dibuat tidak menangani operasi yang dilakukan oleh lebih dari satu pengguna secara bersamaan.
2. Tugas Akhir ini tidak membahas mengenai pendistribusian kunci.

I.5 Metodologi

Pada pelaksanaan Tugas Akhir ini dilakukan metode sebagai berikut :

1. Analisis Permasalahan

Analisis permasalahan dilakukan analisis terhadap rumusan masalah dan mencari solusi dari rumusan masalah tersebut. Analisis ini juga termasuk analisis kebutuhan perangkat lunak.

2. Perancangan

   Menentukan spesifikasi-spesifikasi perangkat lunak yang akan dibangun. Kemudian melakukan perancangan perangkat lunak yang akan dibangun. Tahap perancangan ini termasuk perancangan skema enkripsi homomorfik yang akan digunakan.

3. Implementasi

   Membangun perangkat lunak berdasarkan rancangan yang telah dibuat

4. Pengujian

   Melakukan pengujian perangkat lunak hasil implementasi berdasarkan spesifikasi-spesifikasi yang telah dibuat sebelumnya.

I.6 Sistematika Laporan

Sistematika penulisan tugas akhir ini adalah sebagai berikut :

1. Bab I Pendahuluan, berisi uraian mengenai latar belakang pelaksanaan tugas akhir, rumusan masalah, tujuan, batasan masalah, metodologi dan sistematika laporan tugas akhir.
2. Bab II Dasar Teori, berisi uraian teori-teori yang berkaitan dengan tugas akhir ini serta kajian-kajian terkait makalah atau hasil karya orang lain yang terkait dengan pelaksanaan tugas akhir ini.
3. Bab II Analisis Masalah, berisi analisis permasalahan untuk memperoleh solusi yang terbaik untuk rumusan permasalahan.
4. Bab IV Analisis dan Perancangan Perangkat Lunak, menguraikan bagian analisis perangkat lunak yang terdiri deskripsi umum perangkat lunak yang akan dibangun, analisis kebutuhan, analisis penggunaan perangkat lunak, analisis aliran data dan model analisis. Selain itu juga menguraikan bagian perancangan arsitektur dan perancangan modul perangkat lunak.

5. Bab V Implementasi dan Pengujian, berisi uraian mengenai implementasi dan pengujian perangkat lunak yang telah dibangun.
6. Bab VI Kesimpulan dan Saran, berisi kesimpulan dan saran dari keseluruhan pelaksanaan tugas akhir.

BAB II

STUDI LITERATUR

Bab ini membahas mengenai landasan literatur mengenai tugas akhir yang dikerjakan. Bab ini berisi penjelasan mengenai kriptografi, enkripsi homomorfik dan skema enkripsi homomorfik yang biasa digunakan. Kemudian bab ini juga membahas mengenai sistem *e-voting* dan kaitannya dengan kriptografi.

II.1 Kriptografi

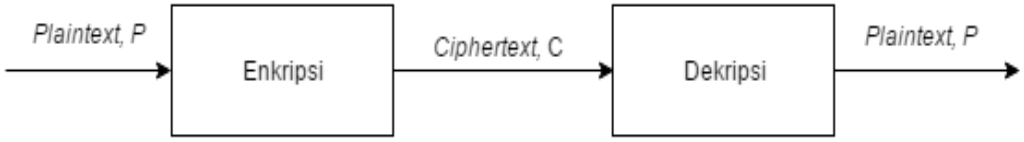
Subbab ini membahas mengenai kriptografi yang meliputi definisi kriptografi, algoritma kriptografi kunci simetri dan algoritma kriptografi kunci publik.

II.1.1 Definisi Kriptografi

Kriptografi didefinisikan sebagai ilmu dan seni untuk menjaga keamanan pesan (Schreiber, 1999). Teknik kriptografi bekerja dengan mengganti pesan *(plaintext)* menjadi pesan tidak bermakna *(ciphertext)*.

Saat ini algoritma kriptografi dapat diasumsikan bersifat umum. Artinya semua orang dapat mengetahui bagaimana algoritma kriptografi tersebut bekerja. Keamanan sistem kriptografi terletak pada kerahasiaan kunci yang digunakan. Keamanan sistem kriptografi ditentukan oleh sulitnya menemukan kunci yang digunakan karena ruang kemungkinan kuncinya sangat besar dan tingkat keacakan *ciphertext* yang dihasilkan algoritma kriptografi. Algoritma kriptografi yang digunakan harus memenuhi prinsip *confussion* dan *difusion*. *Confussion* yaitu tingkat keacakan *ciphertext* dan *difusion* adalah tingkat penyebaran. Kedua hal tersebut menyebabkan *ciphertext* sulit untuk dianalisis secara statistik.

Terdapat dua proses dalam kriptografi yaitu enkripsi dan dekripsi. Enkripsi adalah proses menyandikan *plaintext* menjadi *ciphertext*, proses ini juga biasa disebut *encphering*. Dekripsi adalah proses kebalikan dari enkripsi yaitu proses mengubah *ciphertext* menjadi *plaintext* semula, proses ini juga biasa disebut *deciphering*. Proses enkripsi dan dekripsi dapat dilihat pada Gambar II-1.



Gambar II-1 Proses enkripsi dan dekripsi

Secara matematis fungsi enkripsi dan dekripsi berturut-turut dapat dinyatakan sebagai berikut:

$$E(P) = C \tag{2.1}$$

$$D(C) = P \tag{2.2}$$

Yang dalam hal ini C adalah *ciphertext* dan P adalah *plaintext*. Fungsi enkripsi dan dekripsi yang bersesuaian juga harus memenuhi syarat

$$D\big(E(P)\big) = P \tag{2.3}$$

II.1.2 Algoritma Kriptografi

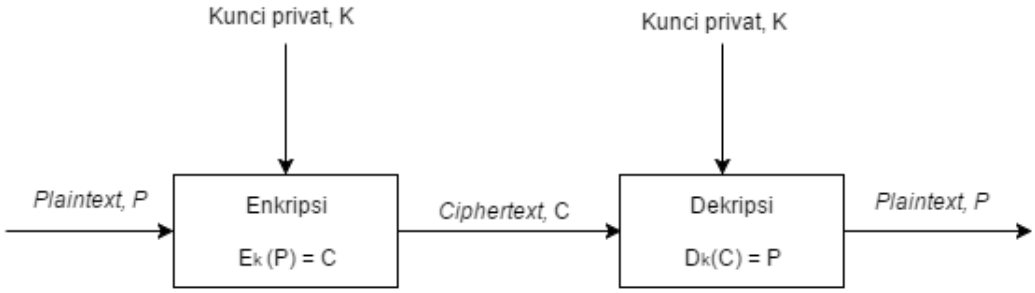
Algoritma kriptografi adalah aturan untuk melakukan proses enkripsi dan proses dekripsi atau fungsi matematika yang digunakan untuk melakukan enkripsi dan dekripsi pesan (Munir R, 2011).

Berdasarkan zamannya, algoritma kriptografi dapat dibagi menjadi dua jenis yaitu algoritma kriptografi klasik dan algoritma kriptografi modern. Algoritma kriptografi klasik adalah algoritma kriptografi yang ada sebelum komputer ditemukan. Algoritma kriptografi klasik menggunakan cara-cara sederhana untuk memetakan pesan menjadi pesan terenkripsi seperti substitusi dan transposisi. Contoh algoritma kriptografi klasik yaitu *caesar cipher, vigenere cipher, playfair cipher* dan *enigma cipher*. Algoritma kriptografi modern adalah algoritma kriptografi yang ditemukan setelah penemuan komputer. Algoritma kriptografi klasik menggunakan operasi dalam mode bit.

Berdasarkan jenis kunci yang digunakan, algoritma kriptografi dapat dibagi menjadi dua jenis yaitu algoritma kriptografi kunci simetri (*symetric key cryptography*) dan algoritma kriptografi kunci publik (*public key cryptography*).

II.1.3 Kriptografi Kunci Simetri

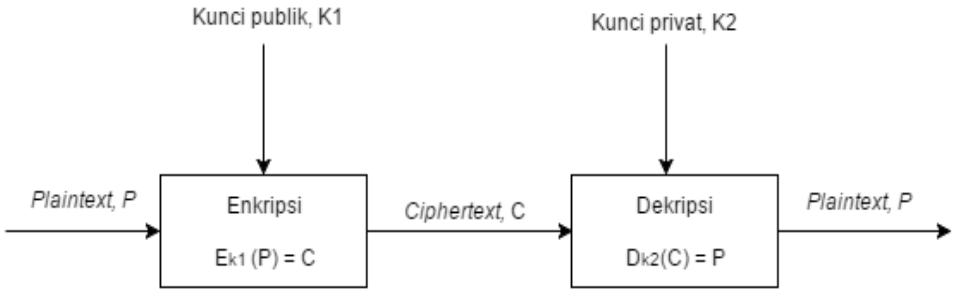
Algoritma kriptografi kunci simetri merupakan algoritma kriptografi yang menggunakan kunci yang sama untuk melakukan enkripsi dan dekripsi pesan. Contoh algoritma kunci simetri adalah *vigenere cipher*, *Rijndael*, *Blowfish*, RC4 dan lain-lainnya. Secara umum algoritma ini memiliki kelemahan pada risiko keamanan saat pengiriman kunci. Skema kriptografi kunci simetri dapat dilihat pada Gambar II-2.



Gambar II-2 Skema algoritma kriptografi kunci simetri

II.1.4 Kriptografi Kunci Publik

Algoritma kriptografi kunci publik merupakan algoritma kriptografi yang menggunakan kunci yang berbeda untuk melakukan enkripsi dan dekripsi pesan. Kunci publik digunakan untuk mengenkripsi pesan dan kunci privat digunakan untuk mendekripsi pesan. Skema kriptografi kunci publik dapat dilihat pada Gambar II-3.



Gambar II-3 Skema algoritma kriptografi kunci publik

Terdapat beberapa algoritma kriptografi kunci publik diantaranya adalah RSA, ElGamal, Paillier.

II.1.4.1 RSA

Salah satu algoritma kunci publik yang terkenal adalah RSA. Algoritma RSA dikembangkan oleh Ron Rivest, Adi Shamir dan Leonard Adleman. RSA itu sendiri merupakan singkatan dari ketiga orang pembuatnya. Keamanan algoritma RSA terletak pada sulitnya mencari faktor-faktor prima dari suatu bilangan yang besar. Pemfaktoran ini digunakan untuk membangkitkan kunci rahasia. Selama algoritma yang efisien untuk memfaktorkan suatu bilangan menjadi faktor-faktor primanya, maka keamanan RSA masih terjamin.

Untuk membangkitkan pasangan kunci (kunci publik dan kunci privat) pada algoritma RSA, berikut ini langkah-langkah yang perlu dilakukan :

1. Pilih dua buah bilangan prima p dan q
2. Hitung $n = p.q$. Sebaiknya pilih $p \neq q$, sehingga nilai p dan q lebih sulit ditemukan.
3. Hitung nilai $\emptyset(n)$ dengan rumus

$$\emptyset(n) = (p-1)(q-1) \tag{2.4}$$

Pilih kunci publik e yang relatif prima terhadap $\emptyset(n)$

4. Bangkitkan kunci rahasia d dengan persamaan

$$d = \frac{1 + k.\emptyset(n)}{e} \tag{2.5}$$

dengan k adalah suatu bilangan bulat.

Daris hasil dari proses di atas diperoleh nilai n dan d sebagai kunci privat. Kunci privat ini sifatnya rahasia. Kemudian diperoleh nilai dan e yang merupakan kunci publik. Kunci publik ini sifatnya tidak rahasia.

Proses enkripsi dengan algoritma RSA dilakukan dengan cara :

1. Ambil kunci publik e dan nilai n
2. Nyatakan *plaintext* dan blok-blok

3. Enkripsi tiap blok dengan rumus $c = m^e \ mod \ n$.

Sementara itu proses dekripsi algoritma RSA dilakukan dengan cara :

1. Ambil kunci privat d dan nilai n.
2. Dekripsi setiap blok dengan rumus

$$m = c^d \ mod \ n \tag{2.6}$$

3. Gabungkan blok-blok hasil dekripsi sehingga menjadi pesan yang utuh.

II.1.4.2 ElGamal

Algoritma ElGamal merupakan algoritma kriptografi kunci publik yang didesain oleh Taher Elgamal pada tahun 1985. Algoritma ini pada awalnya digunakan untuk tanda tangan digital (*digital signature*). Algoritma ini kemudian dimodifikasi agar dapat digunakan untuk keperluan enkripsi dan dekripsi.

Algoritma ElGamal didasarkan pada sulitnya menghitung nilai logaritma diskrit. Permasalahan logaritma diskrit yang dimaksud disini adalah jika terdapat bilangan prima p serta y dan g adalah sembarang bilangan bulat dan terdapat x sedemikian sehingga $g^x \equiv y \ (mod \ p)$.

Untuk membangkitkan pasangan kunci, berikut ini langkah-langkah yang perlu dilakukan :

1. Pilih sembarang bilangan prima p.
2. Pilih dua buah bilangan bulat g dan x dengan syarat $g < p$ dan $1 \leq x \leq p - 2$.
3. Hitung nilai y dengan rumus

$$y = g^x \ mod \ p \tag{2.7}$$

Hasil dari langkah-langkah tersebut adalah kunci publik yaitu y, g, p dan kunci privat yaitu pasangan x, p.

Proses enkripsi dengan algoritma ElGamal dilakukan dengan cara :

1. Ambil kunci publik g dan y.
2. Pilih bilangan bulat k sehingga $1 \leq k \leq p - 2$.

3. Nyatakan *plaintext* dalam blok-blok.
4. Enkripsi setiap blok dengan menggunakan rumus

$$a = g^k \, mod \, p \tag{2.8}$$

$$b = y^k \, m \, mod \, p \tag{2.9}$$

Hasil enkripsi berupa pasangan nilai a dan b. Ukuran pesan setelah dienkripsi akan menjadi dua kali lebih panjang.

Sementara itu proses dekripsi dilakukan dengan cara :

1. Ambil kunci privat x.
2. Dekripsi setiap pasangan nilai *ciphertext* a dan b dengan rumus

$$m = \frac{b}{a^x} \, mod \, p \tag{2.10}$$

3. Gabungkan blok-blok hasil dekripsi menjadi pesan utuh.

II.1.4.3 Pailier

Algoritma Pailier merupakan algoritma kriptografi kunci publik yang menggunakan *probabilistic symetric algorithm*. Algoritma ini ditemukan oleh Pascal Pailier pada tahun 1999. Algoritma Pailier didasarkan pada sulitnya menghitung residu kelas ke-n atau disebut *composite residuosity problem*. Suatu bilangan bulat z dikatakan sebagai residu ke-n modulo n^2 jika terdapat bilangan bulat y sehingga $z = y^n \, mod \, n^2$ (Pailier P, 1999).

Untuk membangkitkan pasangan kunci pada algoritma Pailier, berikut ini langkah-langkah yang harus dilakukan :

1. Pilih dua buah bilangan prima p dan q yang memenuhi syarat $gdc\big(pq, (p-1)(q-1)\big) = 1$
2. Hitung $n = pq$ dan $\lambda = lcm\,(p-1, q-1)$
3. Pilih sembarang bilangan bulat g, dengan $g < n^2$
4. Hitung $\mu = (L\big(g^\lambda \, mod \, n^2\big))^{-1} \, mod \, n$

Dengan fungsi L adalah $L(x) = \frac{x-1}{n}$

Hasil dari langkah-langkah tersebut adalah kunci publik yang berupa pasangan g, n dan kunci privat yang berupa pasangan λ, μ.

Proses enkripsi dengan algoritma Pailier dilakukan dengan cara :

1. Bagi *plaintext* menjadi blok-blok sehingga nilai setiap blok *plaintext* lebih kecil dari n.
2. Pilih bilangan bulat r dimana $r < n$.
3. Enkripsi setiap blok *plaintext* m dengan rumus

$$c = g^m r^n \bmod n^2$$

Residu kelas c dapat dianyatakan sebagai berikut (Pailier, 1999)

$$[\![c]\!]_g = m \tag{2.11}$$

Kemudian proses dekripsi pada algoritma Pailier dilakukan dengan cara :

1. Dekripsi tiap blok *ciphertext* dengan rumus (Pailier, 1999)

$$m = \frac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n \tag{2.12}$$

atau

$$m = L\left(c^\lambda \bmod n^2\right).\mu \bmod n \tag{2.13}$$

dengan sifat

$$L\left(c^\lambda \bmod n^2\right) = \lambda.[\![c]\!]_{(1+n)} \bmod n \tag{2.14}$$

$$\left(L\left(g^\lambda \bmod n^2\right)\right)^{-1} = \left(\lambda.[\![g]\!]_{(1+n)}\right)^{-1} \bmod n \tag{2.15}$$

2. Setelah dekripsi blok-blok *ciphertext* selesai, gabungkan blok-blok *plaintext* yang dihasilkan menjadi pesan utuh.

II.2 Enkripsi Homomorfik

Enkripsi Homomorfik (*homomorphic encryption*) adalah suatu bentuk enkripsi yang memungkinkan dilakukannya komputasi pada *ciphertext* tanpa mendekripsi terlebih dahulu *ciphertext* tersebut. Operasi yang dilakukan pada *ciphertext* yang

menggunakan enkripsi homomorfik akan menghasilkan *ciphertext* yang jika didekripsi akan menghasilkan hasil yang sama dengan operasi serupa pada *plaintext* (Morris, 2013).

Secara matematis, *homomorphic cryptosystem* adalah sebuah *cryptosystem* yang menggunakan fungsi enkripsi yang bersifat homomorfik dan memungkinkan dilakukannya operasi pada *ciphertext*. Terdapat dua jenis operasi utama yaitu penjumlahan dan pengurangan (Poetzelsberger, 2013).

Suatu *cryptosystem* dikatakan bersifat additif jika dan hanya jika :

$$\exists \Delta: \ \varepsilon(x1) \ \Delta \ \varepsilon(x2) = \ \varepsilon(x1 + x2) \tag{2.16}$$

Dengan $x1$ dan $x2$ adalah *plaintext*, ε adalah fungsi enkripsi dan Δ adalah suatu operasi yang bergantung pada sifat algoritma enkripsi yang digunakan. Kemudian suatu kriptosistem dikatakan bersifat multiplikatif jika dan hanya jika :

$$\exists \Delta: \ \varepsilon(x1) \ \Delta \ \varepsilon(x2) = \ \varepsilon(x1.x2) \tag{2.17}$$

Terdapat dua jenis enkripsi homomorfik yaitu *partially homomorphic encryption* (PHE) dan *fully homomorphic encryption* (FHE). PHE merupakan jenis enkripsi homomorfik yang memungkinkan dilakukannya satu jenis operasi tertentu pada *ciphertext*. Sementara itu FHE merupakan jenis enkripsi homomorfik yang memungkinkan kedua jenis operasi penjumlahan dan perkalian dilakukan pada *ciphertext* (Poetzelsberger, 2013).

II.2.1 *Partially Homomorhic Encryption*

Suatu *cryptosystem* dikatakan bersifat *partially homomorphic* jika *cryptosystem* tersebut memiliki salah satu dari sifat additif atau multiplikatif tapi tidak keduanya (Gentry C, 2009). Maksudnya dapat dilakukan salah satu operasi penjumlahan atau perkalian pada *ciphertext*. Beberapa contoh *cryptosystem* yang bersifat *partially homomorphic* yaitu RSA, ElGamal dan Pailier. RSA memiliki sifat multiplikatif, ElGamal memiliki sifat multiplikatif dan Pailier memiliki sifat additif.

II.2.1.1 Homomorphic RSA

RSA memiliki sifat multiplikatif. Dengan mengalikan dua buah *ciphertext* RSA maka hasil dekripsinya akan ekuivalen dengan perkalian dua buah nilai *plaintext*-nya.

Misalkan a dan b adalah dua buah *plaintext*, kemudian $c1$ dan $c2$ adalah *ciphertext* dari masing-masing a dan b dengan enkripsi RSA dan e adalah kunci publik maka

$$c1 = a^e \ mod \ n$$

$$c2 = b^e \ mod \ n$$

Lakukan perkalian pada c1 dan c2 sehingga diperoleh

$$c1.c2 = (a.b)^e \ mod \ n$$

Ketika didekripsi kembali maka dekripsi hasil perkalian $c1.c2$ hasilnya adalah

$$\begin{aligned} d(c1.c2) &= (c1.c2)^d \ mod \ n \\ &= ((a.b)^e)^{\ d} \ mod \ n \\ &= (a.b)^{e.d} \ mod \ n \\ &= (a.b)^{1+k\emptyset(n)} \ mod \ n \\ &= (a.b).(a.b)^{k\emptyset(n)} \ mod \ n \end{aligned}$$

Karena $x^{k\emptyset(n)} \equiv 1 \ (mod \ n)$, maka

$$d \ (c1.c2) = a.b \ mod \ n \tag{2.18}$$

II.2.1.2 Homomorphic ElGamal

ElGamal memiliki sifat multiplikatif sama seperti RSA. Dengan mengalikan dua buah *ciphertext* ElGamal maka hasil dekripsinya akan ekuivalen dengan perkalian dua buah nilai *plaintext*-nya.

Misalkan a dan b adalah dua buah *plaintext*, kemudian pasangan $(c1, \ c2)$ dan $(c3, c4)$ adalah *ciphertext* dari masing-masing a dan b dengan enkripsi ElGamal dan g, y adalah pasangan kunci publik maka

$$c1 = g^{k1} \bmod p, \qquad c2 = y^{k1}.a \bmod p$$
$$c3 = g^{k2} \bmod p, \qquad c4 = y^{k2}.b \bmod p$$

Lakukan perkalian dua buah pasangan *ciphertext* sehingga diperoleh :

$$(c1, c2).(c3, c4) = (c1.c3\ , c2.c4)$$
$$= (g^{k1}.g^{k2}, y^{k1}.a.y^{k2}.b\ )$$
$$= (g^{k1+k2}, a.b.y^{k1+k2})$$

Ketika didekripsi kembali maka hasil

$$d\big((c1, c2).(c3, c4)\big) = d(g^{k1+k2}, a.b.y^{k1+k2})$$
$$= \frac{a.b.y^{k1+k2}}{(g^{k1+k2})^x} \bmod p$$
$$= \frac{a.b.g^{x(k1+k2)}}{g^{(k1+k2)^x}} \bmod p$$
$$d\big((c1, c2).(c3, c4)\big) = a.b \bmod n \tag{2.19}$$

II.2.1.3 Homomorphic Pailier

Pailier memiliki sifat additif. Dengan mengalikan dua buah *ciphertext* Pailier maka hasil dekripsinya akan ekuivalen dengan penjumlahan dua buah nilai *plaintext*-nya.

Misalkan a dan b adalah dua buah *plaintext*, kemudian $c1$ dan $c2$ adalah *ciphertext* dari masing-masing a dan b dengan enkripsi ElGamal dan g, y adalah pasangan kunci publik maka

$$c1 = g^a r_1^n \bmod n^2$$
$$c2 = g^b r_2^n \bmod n^2$$

Dengan mengalikan c1 dan c2 maka akan diperoleh hasil :

$$c1.c2 = g^a r_1^n . g^b r_2^n \bmod n^2$$
$$= g^{a+b}(r_1 r_2)^n \bmod n^2$$

Ketika dilakukan dekripsi maka :

$$d(c1.c2) = \frac{L(c^{\lambda} \bmod n^2)}{L(g^{\lambda} \bmod n^2)} \bmod n$$

$$= \frac{\lambda.[\![c1.c2]\!]_{(1+n)}}{\lambda.[\![g]\!]_{(1+n)}} \bmod n$$

Karena $\mathrm{c1.c2} = g^{a+b}(r_1 r_2)^n$ maka $[\![c1.c2]\!]_g = a + b$

Sehingga

$$d(c1.c2) = \frac{\lambda.[\![c1.c2]\!]_{(1+n)}}{\lambda.[\![g]\!]_{(1+n)}} \bmod n$$

$$= [\![c1.c2]\!]_g \bmod n$$

$$d(c1.c2) = (a+b) \bmod n \tag{2.20}$$

II.2.2 *Fully Homomorphic Encryption*

Suatu *cryptosystem* dikatakan bersifat *fully homomorphic* jika *cryptosystem* tersebut memiliki sifat additif atau multiplikatif (Gentry C, 2009). Maksudnya terdapat operasi pada *ciphertext* yang dapat mewakili operasi penjumlahan dan pengurangan pada *plaintext*.

Skema FHE yang ada saat ini yaitu menggunakan *cryptosystem* yang dikembangkan oleh Craig Gentry pada tahun 2009. Skema tersebut menggunakan ideal lattice untuk merepresentasikan kunci dan *ciphertext*-nya (Morris L, 2013).

Kunci privat terdiri dari matriks V yang dibangkitkan secara random dan sebuah matriks W sehingga

$$V \times W = c \bmod f(x) \tag{2.21}$$

Dengan $f(x)$ adalah sebuah polinom dan c adalah konstanta. Kunci publik B merupakan *Hermite Normal Form* dari V dan kunci publik B dapat direpresentasikan dalam dua buah integer r dan d.

Untuk melakukan enkripsi bit b, maka terlebih dahulu dibangkitkan sebuah vektor $\boldsymbol{u}$ yang nilai elemen-elemennya adalah 0 dengan nilai probabilitas q dan ± 1 dengan nilai probabilitas masing-masing $(1-q)/2$. Proses enkripsi dilakukan dengan cara

$$\boldsymbol{a} = 2\boldsymbol{u} + b.\boldsymbol{e1} \tag{2.22}$$

$$\boldsymbol{c} = \boldsymbol{a}\ mod\ B \tag{2.23}$$

Hasil dari perhitungan tersebut (**c**) adalah *ciphertext*-nya (Gentry, Halevi, 2011).

Kemudian untuk melakukan dekripsi *ciphertext* $\boldsymbol{c}$ dilakukan dengan cara berikut

$$\boldsymbol{a} = \boldsymbol{c}\ mod\ V \tag{2.24}$$

$$b = a_0\ mod\ 2 \tag{2.25}$$

Nilai *b* adalah bit *plaintext* yang telah dienkripsi sebelumnya (Gentry, Halevi, 2011).

II.3 E-voting

Pada subbab ini akan dijelaskan mengenai definisi *e-voting* serta adaptasi *e-voting* yang dapat dilakukan berdasarkan sistem pemilihan yang ada di Indonesia.

II.3.1 Definisi *E-voting*

Electronic voting atau *e-voting* adalah penggunaan komputer atau komputerisasi pada proses pemungutan suara pada pemilihan (Kahani M, 2015). Teknologi *electronic voting* dimulai pada tahun 1970 yang disebut teknologi pencatatan langsung secara elektronik atau lebih dikenal dengan istilah DRE (*direct recording electronic*). Cara memilih dengan sistem ini adalah dengan memilih kandidat yang sudah tercetak pada layar komputer. Pemilih hanya menekan tombol untuk memilih pilihan yang diinginkan.

E-voting dapat dilakukan pada suatu tempat tertentu contohnya tempat pemungutan suara dengan menggunakan DRE. Mekanisme ini sama dengan voting konvensional, hanya saja menggunakan alat elektronik untuk merekam pilihan dari

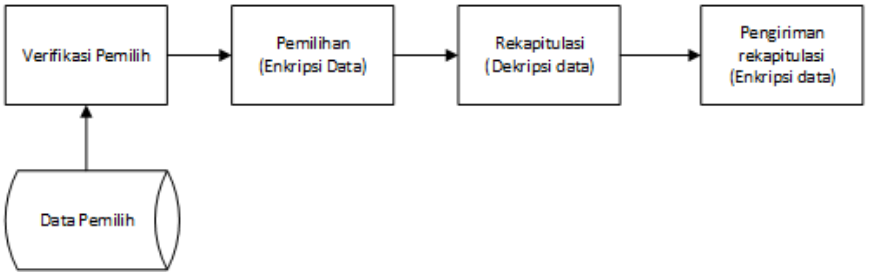
pemilih. Terdapat juga mekanisme *remote voting* atau voting yang dilakukan oleh pemilih dari jarak jauh. Pada voting konvensional, *remote voting* ini biasanya dilakukan dengan mengirimkan surat pernyataan dari pemilih. Pada sistem *e-voting*, pemilihan dari jarak jauh lebih dimungkinkan dengan cara :

1. Pemilihan melalui telepon. Pemilihan ini dilakukan dengan cara pemilih menelepon nomor tertentu kemudian bot akan meminta masukan pilihan. Pemilih hanya perlu menekan tombol pada telepon lalu akan terekam dan disimpan. Kekurangan dari cara ini adalah tidak dapat melakukan banyak pemilihan dalam waktu bersamaan dan beberapa negara tidak memberlakukan enkripsi pada jaringan telepon sehingga rawan akan penyadapan.
2. Pemilihan melalui SMS. Pemilihan ini dilakukan dengan cara pemilih mengirimkan *username* dan PIN terlebih dahulu kemudian akan di konfirmasi oleh sistem. Jika berhasil maka sistem akan mengirimkan formulir pemilihan dan pemilih hanya perlu mengirimkan pilihannya lalu pilihan tersebut akan terekam oleh sistem. Masalah utama pada pemilihan model ini adalah kerahasiaan data pemilihan sebab beberapa negara tidak melakukan enkripsi terhadap jaringan telepon seluler.
3. Pemilihan melalui internet. Pemilih melakukan pemilihan melalui situs web tertentu. Pemilih harus *login* terlebih dahulu kemudian memilih melalui formulir yang disediakan. Data yang dikirim dienkripsi terlebih dahulu untuk menghindari kemungkinan penyadapan dan perubahan data.

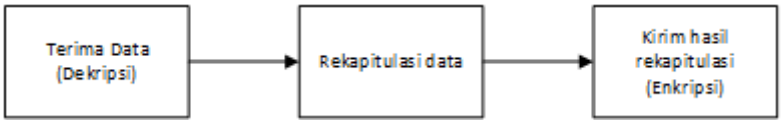
II.3.2 Adaptasi *E-voting* Berdasarkan Sistem Pemilihan di Indonesia

Proses pemilihan di Indonesia dilakukan pada Tempat Pemungutan Suara (TPS). Data hasil pemilihan tiap TPS akan direkapitulasi kemudian hasil tersebut akan dienkripsi kemudian dikirimkan ke tingkat diatasnya. Alur pemilihan pada TPS dapat dilihat pada Gambar II-4. Hasil rekapitulasi yang diterima dari TPS akan didekripsi terlebih dahulu kemudian direkapitulasi oleh tingkat diatasnya contohnya panitia kecamatan. Hasil rekapitulasi tingkat kecamatan akan dienkripsi

kemudian dikirim ke tingkat diatasnya yaitu kabupaten dan seterusnya hingga ke panitia pusat. Alur rekapitulasi hasil pemilihan dapat dilihat pada Gambar II-5.



Gambar II-4 Alur pemilihan pada TPS



Gambar II-5 Alur Rekapitulasi

Sementara itu data pemilih atau yang biasa dikenal sebagai Daftar Pemilih Tetap (DPT) akan dikelola oleh sistem pusat. Data tersebut akan dibagi dikirimkan ke masing-masing daerah pemilihan hingga sampai ke TPS. Data tersebut akan digunakan untuk memverifikasi pemilih.

Pemilihan yang dilakukan instansi-instansi lain juga mengikuti pola tersebut. Hanya saja distribusi data pemilih dan rekapitulasi hasil pemilihan bergantung pada besarnya instansi/organisasi dan banyaknya cabang dari instansi/organisasi tersebut.

II.4 *Review* Penelitian Terkait

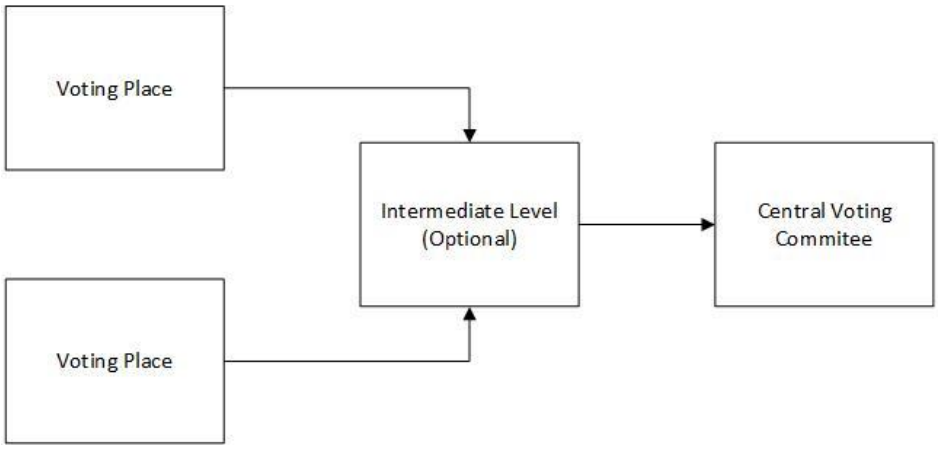
Pada subbab ini akan diuraikan mengenai penelitian-penelitian terkait yang mendukung tugas akhir ini.

II.4.1 *Design and Development of Voting Data Security for Electronic Voting (E-voting)*

Djanali Supeno, dkk (2016) mempublikasikan makalah *Design and Development of Voting Data Security for Electronic Voting (E-voting)*. Sistem *e-voting* yang

dijelaskan pada makalah tersebut mengadaptasi sistem voting untuk pemilihan umum di Indonesia. Terdapat beberapa level perhitungan hasil pemilihan mulai dari level terbawah yaitu TPS hingga ke level teratas yaitu panitia pusat dan diantaranya terdapat kabupaten/kota dan provinsi.

Arsitektur yang diusulkan pada makalah tersebut adalah arsitektur yang terdiri dari sebuah *central voting comitee* yang menerima hasil dari beberapa *intermediate level*. Jumlah *intermediate level* bisa berjumlah nol atau lebih. Setiap Intermediate layer menerima masukan berupa hasil dari *voting place* yang ada di bawahnya. Metode pemilihan yang digunakan pada *voting place* bisa bermacam-macam. Arsitektur ini mendukung metode voting yang digunakan mulai dari voting secara konvensional seperti menggunakan surat suara yang hasilnya kemudian di entri dan disimpan di server hingga voting yang menggunakan sistem elektronik. Arsitektur *e-voting* yang dibangun dapat dilihat pada Gambar II-6.



Gambar II-6 Arsitektur *e-voting* (Djanali Supeno (2016))

Struktur data untuk menyimpan data hasil voting dibuat sehingga kerahasiaan pemilihan dapat terjaga. Struktur data tersebut berisi *vote id, participant, candidate vote, ballout number, vote hash, vote signature, voting place id* dan *sender code*. Data *participant* dan *candidate vote* dienkripsi untuk menjaga agar data pemilih dan kandidat yang dipilihnya terjaga kerahasiaannya. Data *vote hash* dibuat untuk memverifikasi keaslian data pasangan *participant* dan *candidate vote*.

II.4.2 *Analysis Partially Homomorphic Encryption and Fully Homomorphic Encryption*

Morris Liam (2013) mempublikasikan makalah berjudul *analysis partially homomorphic encryption and fully homomorphic encryption*. Makalah tersebut membahas tentang performa *partially homomorphic encryption* khususnya pada algoritma Pailier dan *fully homomorphic encryption*.

Skema *partially homomorphic encryption* menggunakan algoritma Pailier diuji dengan contoh studi kasus menjumlahkan data rekapitulasi jumlah pemilihan dari tiga kandidat. Hasilnya adalah algoritma Pailier mendukung penjumlahan yang bersifat homomorfis pada dua buah *ciphertext* dengan mengalikan *ciphertext* tersebut. Waktu enkripsi berada pada orde 0.1 mili detik dan untuk operasi penjumlahan homomorfis memakan waktu 5 mili detik. Performa algoritma Pailier bisa dikatakan cukup baik.

Pada skema *fully homomorphic encryption* dengan menggunakan *Gentry Cryptosystem*, operasi penjumlahan dan perkalian secara homomorfis dapat dilakukan. Namun masalahnya adalah ukuran kunci yang digunakan haruslah sangat besar agar aspek keamanan yang diharapkan bisa terpenuhi. Pada makalah tersebut ukuran *ciphertext* yang dihasilkan untuk parameter keamanan yang direkomendasikan adalah pada kisaran 128MB dan ukuran kunci publik 128PB. Aspek keamanan dapat diturunkan sehingga ukuran kunci publik hanya pada orde beberapa GB saja namun waktu untuk melakukan enkripsi tetap memakan waktu yang lama yaitu 30 menit untuk mengenkripsi satu bit.

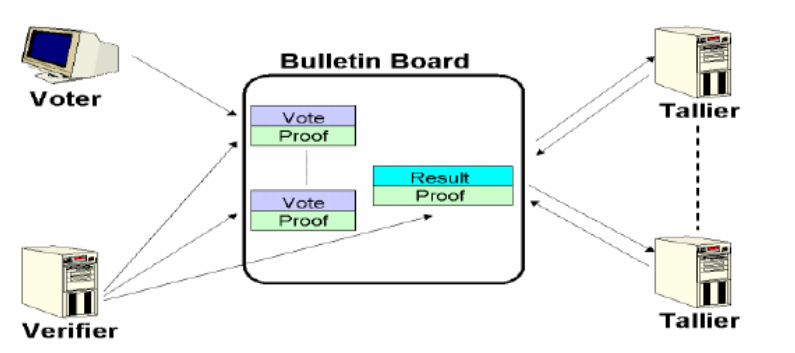
II.4.3 *Secure E-voting Using Homomorphic Technology*

Shinde Shubhangi dkk (2013) mempublikasikan makalah berjudul *Secure E-voting Using Homomorphic Technology*. Makalah tersebut mengusulkan skema *e-voting* menggunakan enkripsi homomorfik. Pada skema tersebut terdapat empat tahapan dalam melakukan pemilihan yaitu :

1. Fase registrasi, yaitu pendaftaran identitas pemilih. Pemilih yang terdaftar akan mendapatkan *username* dan *password*.

2. Fase validasi, fase ini dilakukan saat pemilih akan melakukan pemilihan. Sistem akan mengecek apakah *username* dan *password* pemilih benar dan pemilih tersebut belum melakukan pemilihan.
3. Fase pemilihan, yaitu fase pemilihan kandidat oleh pemilih. Pemilih dan kandidat yang dipilih akan dienkripsi untuk menjaga kerahasiaan pemilihan.
4. Fase perhitungan, yaitu tahap perhitungan suara tiap kandidat.

Sistem voting dengan enkripsi homomorfik yang diusulkan adalah sistem yang bekerja menggunakan model yang disebut *bulletin board*. Pada model tersebut terdapat empat komponen yaitu *voter, verifier, tallier* dan *bulletin board*. Semua informasi yang telah dikirim ke *bulletin board* dapat dibaca oleh semua komponen yang ada. Setiap *user* yang terautorisasi dapat menambahkan pesan pada areanya dan user lain tidak dapat menghapusnya. Skema yang diusulkan dapat dilihat pada Gambar II-7.



Gambar II-7 Skema *Bulletin Board* (Shinde Shubhagi dkk (2013))

BAB III

ANALISIS DAN PERANCANGAN

Bab ini berisi penjelasan analisis permasalahan serta solusi dalam penanganan masalah dalam tugas akhir ini. Solusi tersebut berupa langkah-langkah lojik yang diambil berdasarkan teori yang ada.

III.1 Analisis Permasalahan

Dalam kehidupan demokrasi, pemilihan (voting) merupakan suatu hal yang penting dalam mengambil sebuah keputusan. Keputusan tersebut mulai dari pengambilan kebijakan, pemilihan pemimpin, pengambilan keputusan dan lain-lainnya. Pada era digital saat ini sistem pemilihan mulai bertransformasi dari pemilihan dengan sarana fisik contohnya dengan surat suara menjadi pemilihan yang memanfaatkan sistem elektronik. Sistem pemilihan elektronik atau *e-voting* memiliki kelebihan seperti kemudahan melakukan pemilihan, proses perhitungan dan rekapitulasi yang cepat serta dapat menghemat anggaran untuk surat suara.

Namun dibalik keuntungan dan kemudahan yang ditawarkan sistem *e-voting* terdapat ancaman serius yaitu ancaman dibidang keamanan. Risiko keamanan tersebut dapat terjadi pada proses *e-voting* antaran lain :

1. Keamanan data pemilih, yaitu bagaimana menjaga keamanan data pemilih.
2. Keamanan pada proses pemilihan, yaitu bagaimana menjaga agar proses pemilihan berlangsung aman. Contohnya menjaga pemilihan agar tetap rahasia, menjaga agar pemilih terdaftar saja yang dapat memilih serta menjaga agar satu pemilih hanya dapat melakukan pemilihan sebanyak satu kali.
3. Keamanan saat rekapitulasi hasil, yaitu menjaga keamanan saat proses rekapitulasi seperti perhitungan suara dan rekapitulasi suara dari beberapa daerah pemilihan.
4. Keamanan pengiriman hasil pemilihan, yaitu menjaga keamanan saat dilakukan transmisi hasil pemilihan.

Di antara masalah keamanan yang telah disebutkan diatas, masalah yang menjadi bahasan pada tugas akhir ini adalah masalah keamanan pada proses pemilihan yaitu bagaimana pilihan dari pemilih bersifat rahasia, keamanan saat rekapitulasi hasil pemilihan dan keamanan hasil pemilihan yang dikirim.

Pentingnya keamanan agar pilihan dari pemilihan bersifat rahasia sebab salah satu asas pemilihan adalah bersifat rahasia. Artinya pilihan dari pemilih tidak boleh diketahui oleh orang lain. Oleh karena itu pada proses pemilihan baik itu yang dilakukan secara konvensional maupun secara elektronik harus menjaga agar kerahasiaan pilihan dari pemilih dapat terjaga.

Keamanan saat proses rekapitulasi data juga menjadi hal yang penting sebab pada proses ini sering terjadi tindak kecurangan. Contohnya adalah mengubah hasil perhitungan suara. Kemudian pada saat rekapitulasi hasil rentan akan risiko penggelembungan suara. Keamanan saat transmisi hasil pemilihan juga penting untuk dijaga sebab data yang dikirim mungkin saja disadap oleh pihak tertentu. Keamanan data yang dikirim dimaksudkan agar kerahasiaan pemilihan dapat dijaga.

Data pemilih juga penting dijaga sebab data pemilih memuat informasi-informasi yang bersifat pribadi seperti alamat dan tempat/tanggal lahir. Data pemilih ini juga digunakan untuk memastikan bahwa hanya pemilih yang terdaftar saja yang berhak untuk mengikuti pemungutan suara. Data pemilih ini dikelola oleh panitia pusat penyelenggara pemilihan dan terpisah dari sistem e-voting yang dirancang sehingga keamanan data pemilih tidak dibahas pada tugas akhir ini.

Selain itu pada pemilihan yang sifatnya terdesentralisasi dengan banyak tempat pemilihan perlu dibuat sistem pemilihan sehingga aspek-aspek keamanan pada pemilihan dapat diimplementasikan dengan lebih mudah. Jadi permasalahan pokok dari tugas akhir ini yaitu bagaimana menjaga keamanan data pemilihan baik itu pada saat proses pemilihan maupun saat rekapitulasi hasil pemilihan serta kemudahan pengimplementasiannya.

III.2 Analisis Solusi Permasalahan

Penggunaan sistem *e-voting* dapat meningkatkan kemudahan dalam melakukan pemungutan suara. Dengan *e-voting* proses pemilihan dan rekapitulasi hasil pemilihan menjadi lebih mudah. Permasalahan keamanan merupakan salah satu isu dalam penerapan *e-voting*. Permasalahan yang telah dijelaskan pada subbab sebelumnya dapat diselesaikan dengan cara :

1. Solusi keamanan data pemilih

   Permasalahan ini dapat diatasi dengan membatasi akses terhadap data pemilih sehingga hanya orang-orang tertentu yang dapat mengakses data tersebut. Kemudian untuk menambah keamanan, data pemilih dapat dienkripsi sehingga jika terjadi pencurian data, data tersebut masih tidak dapat dibaca.
2. Solusi keamanan proses pemilihan

   Untuk memastikan hanya orang terdaftar yang dapat memilih dan pemilih hanya dapat memilih sekali maka sistem terlebih dahulu melakukan autentikasi terhadap pemilih sebelum pemilih tersebut melakukan pemilihan. Kemudian untuk menjaga agar pilihan dari pemilih bersifat rahasia maka data pilihan dari pemilih tersebut dapat dienkripsi.
3. Solusi keamanan rekapitulasi hasil pemilihan

   Keamanan saat rekapitulasi hasil pemilihan dapat dilakukan dengan menggunakan enkripsi homomorfik pada data yang akan direkapitulasi.
4. Solusi keamanan pengiriman hasil pemilihan

   Keamanan pengiriman dapat dijaga dengan melakukan enkripsi pada data yang dikirim.

Permasalahan utama yang dibahas pada tugas akhir ini yaitu bagaimana menjaga keamanan data pemilihan pada saat proses pemilihan, saat rekapitulasi hasil pemilihan dan keamanan data hasil pemilihan yang dikirim. Untuk rekapitulasi hasil pemilihan perlu ditentukan skema enkripsi dan algoritma apa yang cocok untuk digunakan. Kemudian untuk menjaga keamanan pada proses pemilihan perlu

ditentukan skema *e-voting* yang akan digunakan. Solusi untuk permasalahan-permasalahan tersebut akan dibahas lebih detail pada subbab selanjutnya.

III.2.1 Pemilihan Skema *E-voting*

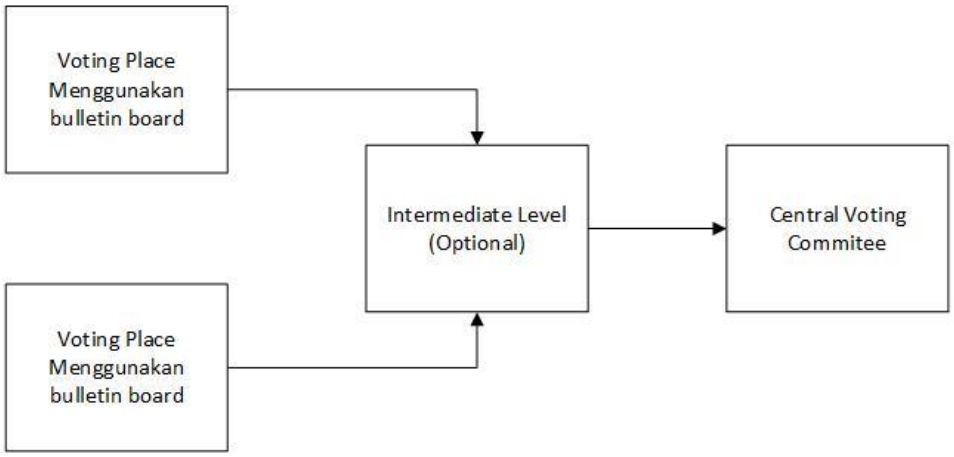
Rekapitulasi data hasil pemilihan dilakukan dengan mencari total suara tiap kandidat dari seluruh tempat pemilihan. Untuk data yang tidak terenkripsi tentu mudah untuk menghitung total suara tiap kandidat, kita hanya perlu menjumlahkannya langsung. Namun berbeda dengan data yang terenkripsi, untuk mencari jumlahnya kita harus mendekripsi data tersebut lalu menjumlahkannya kemudian mengenkripsinya lagi. Sama halnya dengan sistem *e-voting* yang dibuat oleh Djanali Supeno (2016). Kelemahan sistem tersebut adalah kunci untuk mendekripsi pesan harus tersebar ke seluruh *intermediate level* contohnya kabupaten/kota dan provinsi. Hal ini tentu menjadi risiko sebab semakin banyak yang mengetahui kuncinya maka akan semakin besar peluang kunci tersebut dicuri atau diketahui pihak lain yang tidak berhak.

Tabel III-1 Perbandingan skema e-voting Djanali dan Shinde

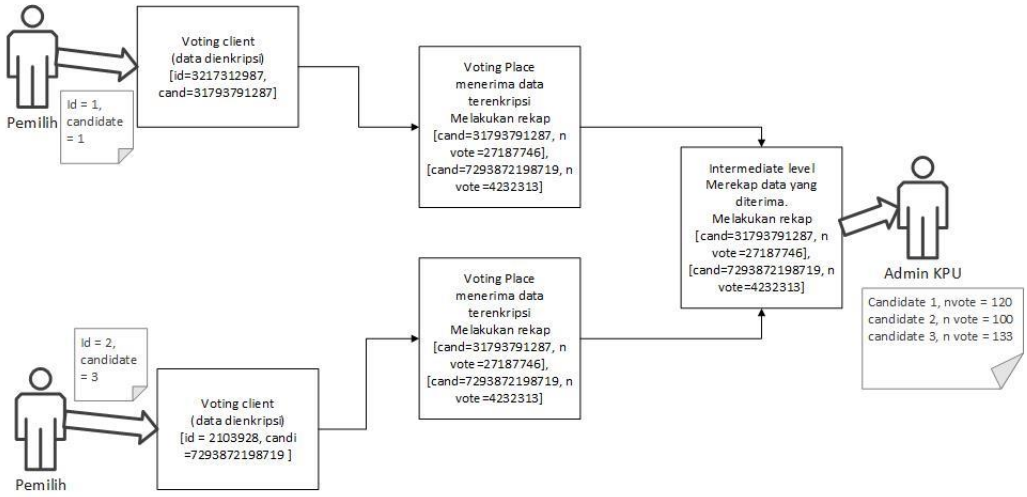
	Algoritma Enkripsi	Kelebihan	Kekurangan
Djanali Supeno dkk	RSA	Adaptif terhadap perbedaan metode pemilihan yang digunakan	kunci untuk mendekripsi pesan harus tersebar ke seluruh *intermediate level*
Shinde Shubhangi dkk	Pailier	Memungkinkan rekapitulasi hasil pemilihan tanpa perlu melakukan dekripsi terlebih dahulu terhadap data	Tersentralisasi, tidak adaptif terhadap perbedaan metode pemilihan

Shinde Shubhangi dkk (2013) mengembangkan sistem *e-voting* yang memanfaatkan enkripsi homomorfik. Sistem ini memungkinkan rekapitulasi hasil pemilihan tanpa perlu melakukan dekripsi terlebih dahulu terhadap data. Namun

kekurangannya adalah sistem *e-voting* yang dikembangkan tersentralisasi. Hal tersebut tidak cocok diterapkan pada pemilihan dengan jumlah pemilih yang banyak dengan wilayah pemilihan yang banyak. Kedua skema *e-voting* tersebut dapat dilihat pada Tabel III-1.



Gambar III-1 Skema gabungan e-voting Djanali Supeno dengan Shinde Shubhangi



Gambar III-2 Ilustrasi proses pemungutan suara

Untuk menutupi kelemahan dari kedua sistem *e-voting* tersebut kita dapat menggabungkan keduanya sehingga dapat saling menutupi kelemahan. Kita dapat menambahkan sifat homomorfis pada sistem *e-voting* Djanali Supeno (2016) dengan menggunakan *bulletin board* pada *voting place* seperti yang dikembangkan Shinde Shubhangi dkk (2013). Gabungan dari kedua skema *e-voting* tersebut dapat dilihat pada Gambar III-1.

Pada sisi klien, voter melakukan *vote* kepada suatu kandidat lalu data mengenai *vote* yang diberikan oleh voter dienkripsi kemudian dikirim ke *voting place*. Di *voting place*, data yang diterima adalah data dalam bentuk terenkripsi. Dari data terenkripsi tersebut dilakukanlah rekapitulasi hasil pemungutan suara di *voting place*. Proses rekapitulasi ini menggunakan penjumlahan secara homomorfik jadi data yang akan direkapitulasi tidak didekripsi sama sekali. Hasil rekapitulasi tersebut juga merupakan data dalam bentuk terenkripsi. Kemudian hasil rekapitulasi di *voting place* akan dikirim ke *intermediate level*. Disana dilakukan juga proses rekapitulasi terhadap data-data yang diterima dari *voting place – voting place* yang ada di bawahnya. Proses rekapitulasi yang dilakukan melibatkan penjumlahan homomorfik sama dengan proses yang terjadi di *voting place.* Ilustrasi tersebut dapat dilihat pada Gambar III-2.

III.2.2 Pemilihan Skema Enkripsi dan Algoritma Enkripsi

Pada sistem *e-voting*, skema enkripsi yang digunakan harus mendukung penjumlahan homomorfis pada *ciphertext*. Pada skema PHE, kita dapat memanfaatkan PHE dengan algoritma Pailier yang memiliki sifat additif homomorfik. Selain PHE Pailier, skema FHE juga memenuhi syarat tersebut sebab FHE mendukung perkalian dan penjumlahan homomorfik.

Berdasarkan penelitian yang dilakukan oleh Morris Liam (2013), FHE saat ini masih membutuhkan *resource* yang sangat besar untuk melakukan enkripsi dan dekripsi sehingga tidak efisien untuk digunakan. Sementara itu PHE dengan algoritma Pailier memberikan hasil yang cukup memuaskan dalam hal enkripsi, dekripsi serta penjumlahan homomorfik. Walaupun PHE dengan algoritma Pailier

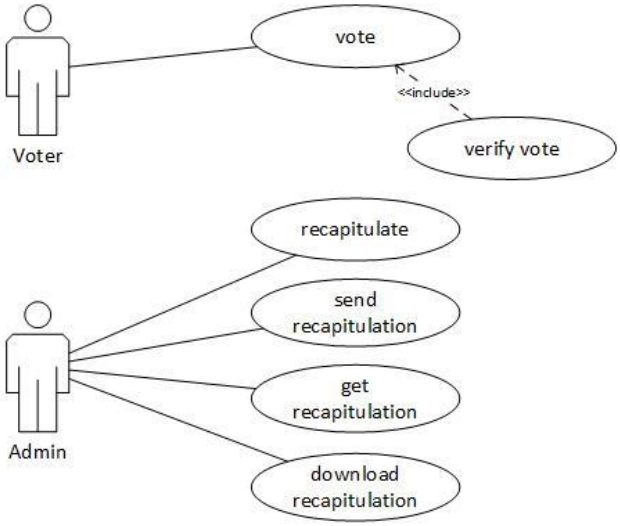
hanya mensupport penjumlahan homomorfik saja, hal tersebut sudah cukup sebab rekapitulasi data hanya membutuhkan operasi penjumlahan saja. Jadi skema enkripsi yang dipilih adalah skema *partially homomorphic encryption* dengan algoritma Pailier.

III.3 Rancangan Solusi

Pada subbab ini akan diuraikan mengenai rancangan solusi yang akan dibangun berdasarkan analisis solusi permasalahan yang telah diuraikan pada subbab III.2.

III.3.1 Model *Use Case*

Perangkat lunak yang dikembangkan menggunakan metode pengembangan perangkat lunak berorientasi objek. Diagram *use case* perangkat lunak dapat dilihat pada Gambar III-3.



Gambar III-3 Diagram *Use Case*

Pada diagram *use case* terdapat dua aktor yaitu admin dan *voter*. *Voter* atau pemilih merupakan aktor yang dapat melakukan *vote* atau peserta pemilihan. Admin merupakan administrator perangkat lunak yang dalam konteks pemilihan merupakan panitia penyelenggara pemilihan. Terdapat lima *use case* pada diagram

use case yaitu *vote, recapitulate, send recapitulation, get recapitulation* dan *download recapitulation*.

Use case vote menangani aksi *vote* yang dilakukan oleh pemilih. Penanganan yang dilakukan yaitu dengan melakukan enkripsi data *vote* pada sisi *client* dengan enkripsi homomorfik. Setelah itu data tersebut dikirim ke server *voting place* . Di sana data akan diverifikasi kemudian disimpan pada basis data.

Use case racapitulate menangani rekapitulasi data hasil pemilihan. *Use case* ini melakukan rekapitulasi hasil pemilihan pada *voting place* dengan menghitung jumlah suara tiap kandidat. Pada *intermediate level, use case* ini melakukan rekapitulasi hasil berdasarkan hasil rekapitulasi yang diterima dari level yang berada dibawahnya.

Use case send recapitulation berperan untuk mengirimkan hasil rekapitulasi data hasil pemilihan dari *voting place* ke *intermediate level* yang berada diatasnya atau dari suatu *intermediate level* ke *intermediate level* diatasnya.

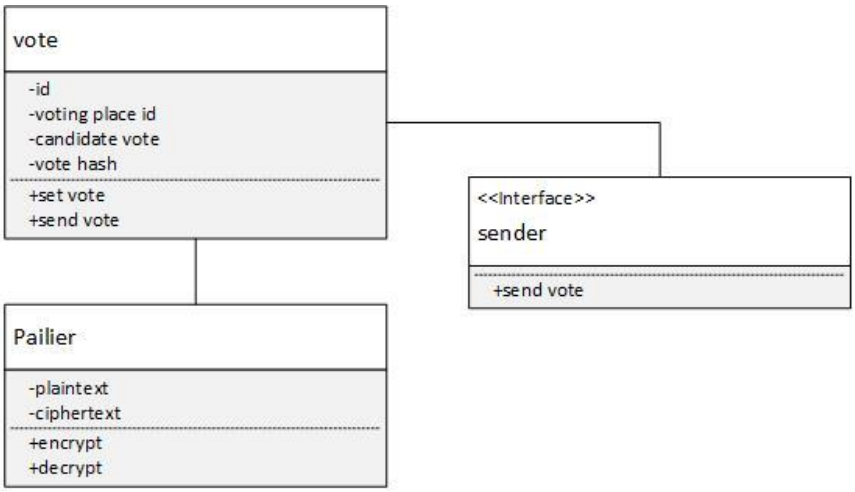
Use case get recapitulation berperan untuk menerima hasil rekapitulasi data yang dikirim oleh *use case send recapitulation*. *Use case* ini akan melakukan verifikasi data terlebih dahulu kemudian menyimpan hasil yang telah diverifikasi ke dalam basis data.

Use case download recapitulation berperan untuk mengunduh hasil rekapitulasi data. Hasil unduhan dari *server* masih dalam bentuk data yang terenkripsi.

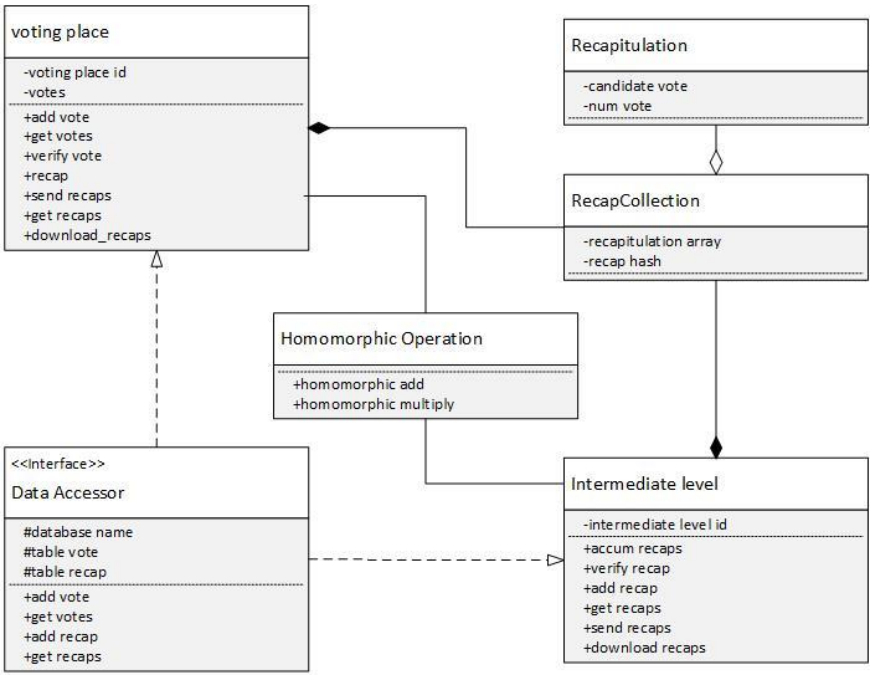
III.3.2 Rancangan Kelas

Rancangan kelas dari aplikasi yang akan dibangun dapat dilihat pada Gambar III-3 dan Gambar III-4. Terdapat dua komponen yang akan dibangun. Komponen yang pertama yaitu komponen berjalan pada sisi klien yang berfungsi untuk mengenkripsi dan mengirimkan data pemungutan suara ke *voting place* . Diagram kelas komponen tersebut dapat dilihat pada Gambar III-4. Komponen yang kedua yaitu komponen yang berjalan pada sisi server. Komponen tersebut bertugas untuk mengumpulkan data pemungutan suara, memverifikasi kemudian melakukan

perhitungan rekapitulasi data. Diagram kelas komponen tersebut dapat dilihat pada Gambar III-5.



Gambar III-4 Rancangan Kelas pada Sisi Klien



Gambar III-5 Rancangan Kelas pada Sisi Server

III.3.3 Perancangan Detail Kelas

Bagian ini berisi penjelasan detail daftar kelas yang ada pada Tabel III-2 dan Tabel III-3.

Tabel III-2 Daftar Kelas pada Klien

No	Nama Kelas/Interface
1	*Vote*
2	Pailier
3	Sender

Tabel III-3 Daftar Kelas pada *Server*

No	Nama Kelas/Interface
1	*Voting place*
2	Recapitulation
3	RecapCollection
4	Homomorphic Operasional
5	Intermediate Level
6	Data Accessor

Kelas *Vote*

Kelas ini merupakan kelas utama pada sisi klien yang berfungsi untuk melakukan *vote* kemudian mengirimkan hasil *vote* tersebut ke *voting place* . *Vote* yang dikirim adalah *vote* dalam bentuk terenkripsi disertai dengan *hash* yang nanti akan digunakan untuk melakukan verifikasi *vote*. Detail kelas *vote* dapat dilihat pada Tabel III-4.

Tabel III-4 Kelas *Vote*

Nama Operasi	**Visibility**	**Keterangan**
set *vote*	public	melakukan *vote* yang berisi id pemilih dan kandidat yang dipilih kemudian dienkripsi.
send *vote*	public	mengirim *vote* ke *voting place*
Nama Atribut	**Visibility**	**Tipe**
id	private	integer

*vote*r id	private	integer
voting place id	private	integer
candidate *vote*	private	integer
vote hash	private	string

Kelas Pailier

Kelas ini berfungsi untuk melakukan enkripsi dan dekripsi pada suatu data dengan algoritma Pailier. Detail kelas dapat dilihat pada Tabel III-5.

Tabel III-5 Kelas Pailier

Nama Operasi	**Visibility**	**Keterangan**
encrypt	public	Melakukan enkripsi dengan algoritma pailier.
decrypt	public	Melakukan dekripsi dengan algoritma pailier.
Nama Atribut	**Visibility**	**Tipe**
plain text	private	array of byte
cipher text	private	array of byte

Interface Sender

Interface sender merupakan *interface* yang berfungsi untuk mengirimkan *vote* ke *voting place* . Pengguna dapat membuat kelas turunan dari *interface* ini untuk mengimplementasikan sendiri bagaimana teknis pengiriman *vote* ke *voting place*. Detail *interface* ini dapat dilihat pada Tabel III-6.

Tabel III-6 Interface Sender

Nama Operasi	**Visibility**	**Keterangan**
send *vote*	public	Mengirim *vote* ke *voting place*

Kelas *Voting place*

Kelas *voting place* merupakan kelas yang berfungsi untuk menampung data *vote* yang dikirimkan oleh pemilih kemudian setelah pemilihan selesai kelas ini dapat menghitung perolehan suara tiap calon dan mengirimkannya ke *Intermediate level* yang ada di atasnya. Kelas ini dapat diibaratkan sebagai Tempat Pemungutan Suara (TPS). Detail kelas dapat dilihat pada Tabel III-7.

Tabel III-7 Kelas *Voting place*

Nama Operasi	**Visibility**	**Keterangan**
add *vote*	public	Menambahkan *vote* ke dalam list *vote* dibasis data.
get *vote*s	public	Mengambil *vote*s
verify *vote*	public	Melakukan verifikasi *vote*
recap	public	Melakukan rekapitulasi dari *vote-vote* yang ada. *Vote* yang dihitung adalah *vote* yang sah yang telah melalui verifikasi dan dihitung berdasarkan kandidat dan jumlah yang memilihnya. Hasil rekapitulasi berupa data yang terenkripsi.
send recap	public	Mengirim rekapitulasi ke intermediate level
download recap	public	Mendownload hasil rekapitulasi
Nama Atribut	**Visibility**	**Tipe**
voting place id	private	integer
recapCollection	private	RecapCollection
*vote*s	private	array of *Vote*s

Kelas Recapitulation

Kelas Recapitulation merupakan kelas yang berisi data rekapitulasi pemilihan. Detail kelas dapat dilihat pada Tabel III-8.

Tabel III-8 Kelas Recapitulation

Nama Atribut	Visibility	Tipe
candidate *vote*	private	integer
number of *vote*	private	integer
hash	private	string

Kelas RecapCollection

Kelas *RecapCollection* merupakan kelas yang berisi data rekapitulasi pemilihan tiap kandidat. Detail kelas dapat dilihat pada Tabel III-9.

Tabel III-9 Kelas RecapCollection

Nama Atribut	Visibility	Tipe
recapitulation tab	private	array of recapitulation
hash	private	string

Kelas Homomorphic Operation

Kelas Homomorphic Operation berfungsi untuk melakukan operasi homomorfik pada data yang terenkripsi. Detail kelas dapat dilihat pada Tabel III-10.

Tabel III-10 Kelas Homomorpfic Operation

Nama Operasi	Visibility	Keterangan
add	public	Melakukan penjumlahan homomorfik
multiply	public	Melakukan perkalian homomorfik. Perkalian dilakukan dengan penjumlahan berulang.

Kelas Intermediate Level

Kelas *Intermediate Level* berfungsi untuk menerima rekapitulasi data pemilihan dari *voting place* atau *intermediate level* yang ada di bawanya. Kemudian kelas ini

dapat melakukan rekapitulasi dari data-data yang diterima. Kelas ini dapat diibaratkan sebagai rekapitulasi suara tingkat kecamatan, kabupaten atau provinsi. Detail kelas dapat dilihat pada Tabel III-11.

Tabel III-11 Kelas *Intermediate Level*

Nama Operasi	**Visibility**	**Keterangan**
add recap	public	Menambahkan hasil rekapitulasi yang diperoleh dari *voting place* atau intermediate level yang ada di bawahnya.
verify recap	public	Melakukan verifikasi terhadap rekapitulasi data yang diterima dari *voting place* atau intermediate level yang ada di bawahnya.
get recap	public	Mengambil rekapitulasi data dari *voting place* atau intermediate level yang ada dibawahnya.
accum recap	public	melakukan akumulasi berdasarkan hasil rekapitulasi yang diterima dan telah diverifikasi.
send recap	public	mengirim rekapitulasi hasil ke intermediate level yang berada di atasnya
download recap	public	Mengunduh hasil rekapitulasi
Nama Atribut	**Visibility**	**Tipe**
intermediate level id	private	integer

Interface Database Accessor

Interface Database Accessor berfungsi sebagai perantara untuk mengakses data yang ada di *database*. Interface ini dapat diturunkan oleh pembuat sistem e-voting untuk mengakses basis data sesuai dengan basis data yang digunakannya. Detail interface ini dapat dilihat pada Tabel III-4.

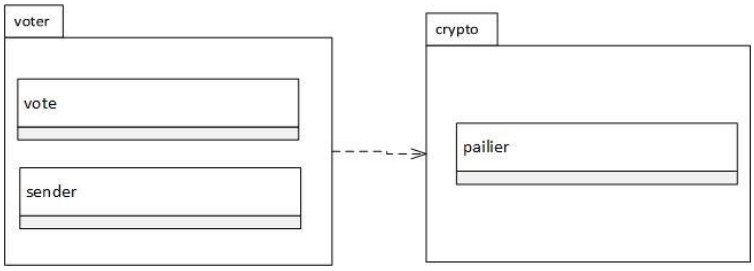
Tabel III-12 Interface Database Accessor

Nama Operasi	**Visibility**	**Keterangan**
add *vote*	public	Menambahkan *vote* ke dalam tabel *vote*s.
get *vote*s	public	Mengambil *vote*s dari basis tabel *vote*s
add recap	public	Menambahkan rekapitulasi ke tabel recapCollection

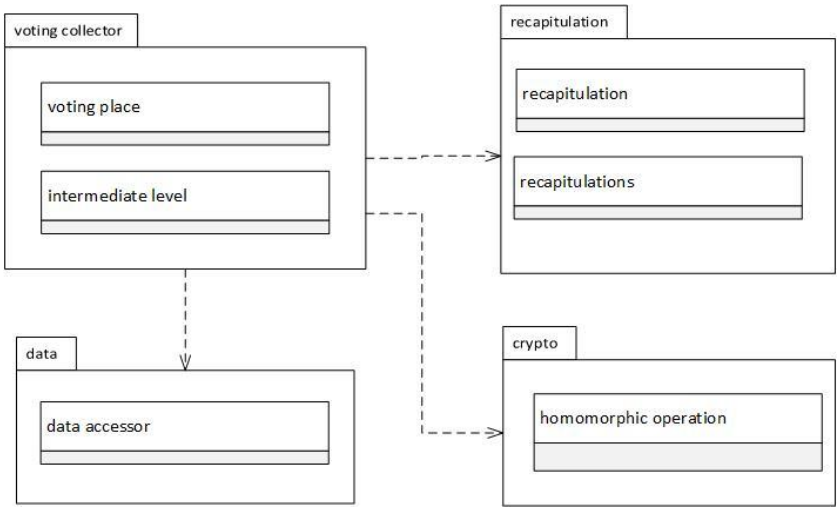
Nama Operasi	Visibility	Keterangan
get recaps	public	Mengambil rekapitulasi dari tabel recapCollection.
Nama Atribut	**Visibility**	**Tipe**
database name	protected	string
table *vote*s name	protected	string
table recaps name	protected	string

III.3.4 Rancangan Diagram Paket

Kelas-kelas yang ada akan dikelompokkan ke dalam paket-paket sesuai dengan keterkaitan fungsionalitas tiap kelas. Diagram paket tersebut dapat dilihat pada Gambar III-7 dan Gambar III-7.



Gambar III-6 Diagram Paket pada Sisi Klien



Gambar III-7 Diagram Paket pada Sisi Server

BAB IV

IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi tentang uraian implementasi dan pengujian perangkat lunak yang dibuat pada tugas akhir ini. Pembahasan dibagi menjadi dua bagian yaitu bagian implementasi dan bagian pengujian.

IV.1 Implementasi

Subbab ini terdiri dari tiga bagian. Pada bagian pertama akan dibahas mengenai lingkungan implementasi. Pada bagian kedua akan dibahas mengenai batasan implementasi. Pada bagian ketiga akan dibahas mengenai spesifikasi program.

IV.1.1 Lingkungan Implementasi

Pengembangan aplikasi *e-voting* dilakukan menggunakan bahasa pemrograman Java. Detail spesifikasi lingkungan implementasi dapat dilihat pada Tabel IV-1.

Tabel IV-1 Spesifikasi lingkungan implementasi

Perihal	Spesifikasi
Sistem Operasi	Windows 8.1 Pro 64-bit
CPU	Intel(R) Core(TM) i3-3110M CPU @ 2.40GHz (4 CPUs), ~2.4GHz
RAM	6 GB
Bahasa pemrograman	Java
JVM	1.8.0_74
IDE	Eclipse
GUI	JavaFX
Framework	Spring
Database	MySQL

IV.1.2 Batasan Implementasi

Implementasi dilakukan dengan batasan-batasan berikut :

1. Autentikasi atau *login user* yang dapat melakukan voting tidak ditangani oleh aplikasi.
2. Implementasi algoritma Pailier menggunakan kelas *BigInteger* yang disediakan oleh bahasa pemrograman Java.

IV.1.3 Hasil Implementasi

Subbab ini berisi penjelasan mengenai hasil implementasi perangkat lunak. Hasil implementasi terdiri dari dua bagian yaitu implementasi yang berupa kelas-kelas dan implementasi yang berupa *file-file* konfigurasi.

IV.1.3.1 Implementasi Kelas-Kelas

Hasil implementasi kelas-kelas perangkat lunak dapat dilihat pada Tabel IV-2. Detail mengenai kelas-kelas tersebut dapat dilihat pada subbab berikutnya.

Tabel IV-2 Hasil implementasi

No	Nama Kelas/Interface	Nama File Fisik
1	*Vote*	*Vote*.java, *Vote*wHash.java
2	Pailier	Pailier.java, KeyPair.java, KeyBuilder.java, Private.java dan Public.java
3	Sender	Sender.java
4	*Voting place*	VotingPlace.java
5	Intermediate Level	IntermediateLevel.java
6	Recapitulation	Recapitulation.java
7	RecapCollection	RecapCollection.java
8	Homomorphic Operation	HomomorphicOperation.java
9	Data Accessor	DataAccessor.java

Berikut ini deskripsi implementasi kelas-kelas sesuai dengan hasil implementasi pada Tabel IV-2 :

1. *Vote*

   Implementasi kelas *Vote* bertugas untuk menyimpan data mengenai *vote* yang dilakukan oleh pemilih. Data *vote* yang disimpan dienkripsi terlebih dahulu kemudian dihitung nilai hash nya. Data *vote* terenkripsi ini kemudian akan dikirim ke *voting place* . Rincian implementasi kelas *vote* dapat dilihat pada lampiran B bagian B1.

2. Pailier

   Kelas Pailier dalam rancangan diimplementasikan menjadi kelas *Pailier.java, KeyBuilder.java, Private.java* dan *Public.java*. *Pailier.java* berfungsi untuk melakukan enkripsi dan dekripsi data. Kelas *KeyBuilder* berfungsi untuk membangkitkan pasangan kunci privat dan kunci publik. Kemudian kelas *Public.java* dan *Private.java* masing-masing merupakan kunci publik dan kunci privat yang digunakan pada algoritma Pailier. Rincian implementasi kelas Pailier dapat dilihat pada lampiran B bagian B2.

3. *Sender*

   Interface ini merupakan interface untuk mengirimkan *vote* dari *client* ke server. Rincian interface Sender dapat dilihat pada lampiran B bagian B3.

4. *Voting place*

   Kelas ini merupakan kelas yang menampung data *vote* yang dikirimkan oleh client. Kelas ini juga berfungsi untuk memverifikasi data yang dikirim dan melakukan rekapitulasi dari *vote-vote* yang diterima. Rincian implementasi kelas *Voting place* dapat dilihat pada lampiran B bagian B4.

5. *Intermediate level*

   Kelas ini merupakan kelas yang berfungsi untuk menampung data hasil rekapitulasi yang diterima dari *voting place* atau *intermediate level* lain. Kelas ini bertugas untuk melakukan rekapitulasi hasil pemilihan. Rincian kelas *Intermediate level* dapat dilihat pada lampiran B bagian B5.

6. *Recapitulation*

Kelas ini berfungsi sebagai struktur data untuk menyimpan rekapitulasi suara yang diterima oleh suatu kandidat. Rincian implementasi kelas ini dapat dilihat pada lampiran B bagian B6.

7. *Recap Collection*

   Kelas ini berisi beberapa recapitulation yang mewakili rekapitulasi suara tiap kandidat. Rincian implementasi kelas ini dapat dilihat pada lampiran B bagian B7.

8. *Homomorphic Operation*

   Kelas *Homomorphic Operation* berfungsi untuk melakukan operasi homomorfik pada *ciphertext*. Operasi tersebut berupa operasi penjumlahan homomorfik dan operasi perkalian *ciphertext* dengan suatu konstanta. Rincian implementasi kelas ini dapat dilihat pada lampiran B bagian B8.

9. *Data Accessor*

   Interface ini merupakan *interface* untuk melakukan pengaksesan data pada basis data atau media penyimpanan data lain yang digunakan.

IV.1.3.2 Implementasi *File* Konfigurasi

Selain kelas-kelas, dihasilkan pula *file* konfigurasi yang bernama *config.properties*. *File* ini berisi parameter-parameter yang dibutuhkan contohnya kunci privat dan kunci publik. Detail *file* konfigurasi dapat dilihat pada Tabel IV-3.

Tabel IV-3 File konfigurasi

No	Nama	Deskripsi
1	n	Nilai n pada kunci publik
2	nSquared	Nilai n kuadrat
3	g	Nilai g pada kunci publik
4	bits	Panjang bit kunci
5	Upperbound	Batas atas bilangan yang dapat dienkripsi
6	lambda	Nilai *lambda* (λ) pada kunci privat
7	denominator	Nilai *denominator* (μ) pada kunci privat
8	hm_zero	*Ciphertext* untuk angka 0
9	hm_one	*Ciphertext* untuk angka 1
10	r	Nilai variabel acak yang digunakan

Contoh penulisan *file* konfigurasi adalah sebagai berikut

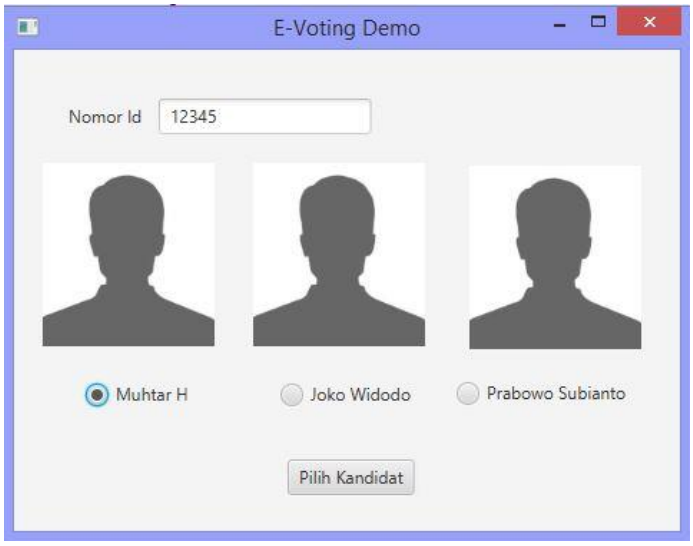
```
#Private
lambda=16717407935959754368936662831821235569816258262354606988003700648173951853 60
denominator=331242388831021437001435260935073560180681479082716420362993884316671343322 40


#Public
n=735565949182229192233213164600134365077477597830087249175076591186502120548 69
nSquared=541057265596353778474261219978751436703533892570069821055745432906018210593204686649329566146746089678811868649987840535202497302274430400738096746660716 1
g=3637421208741661057158537033885259960276001418376512862358869081026817333882
bits=256
#UpperBound
upperBound=2147483647
#number
hm_one=2068520696783573922608936017892774389432033667750676943607635994608186586386069982848794648810716150864031608440876905099020792584814575595977590923517262
hm_zero=3535143190798470358047588826457206157395407178918769151094124625198869564923456259227003620393703778876747368761531257917060854667191228432590855583184 66
```

IV.1.3.3 Aplikasi E-Voting

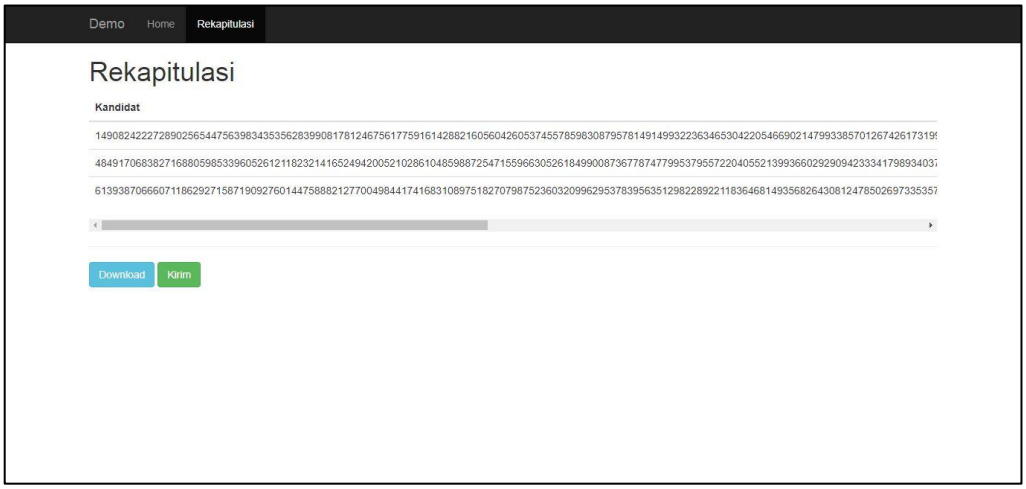
Aplikasi *e-voting* yang dibangun berupa *e-voting* dengan tiga kandidat. Aplikasi tersebut terdiri dari tiga bagian. Bagian *client*, *voting place* dan *intermediate level*.

Bagian *client* merupakan bagian yang berinteraksi langsung dengan pemilih. Bagian ini menerima masukan yang berupa data pemilihan yang dilakukan oleh pemilih. Data yang telah diterima kemudian dienkripsi menggunakan enkripsi homomorfik Pailier. Nomor *id* pemilih dienkripsi menggunakan variabel acak yang dibangkitkan di dalam aplikasi. Kandidat yang dipilih dienkripsi dengan algoritma yang sama tetapi menggunakan variabel acak yang telah ditentukan di *file* konfigurasi. Hal ini dilakukan agar pada saat rekapitulasi, kandidat yang terenkripsi

tersebut dapat diidentifikasi. Setelah masukan diterima kemudian klien akan mengirim data voting yang terlebih dahulu dienkripsi ke *voting place* . Bagian *client* dibuat GUI untuk memudahkan melakukan pemilihan. Bagian ini dibuat dengan bahasa pemrograman Java dan menggunakan JavaFX untuk membuat GUI. Tampilan *client* dapat dilihat pada gambar IV-1.



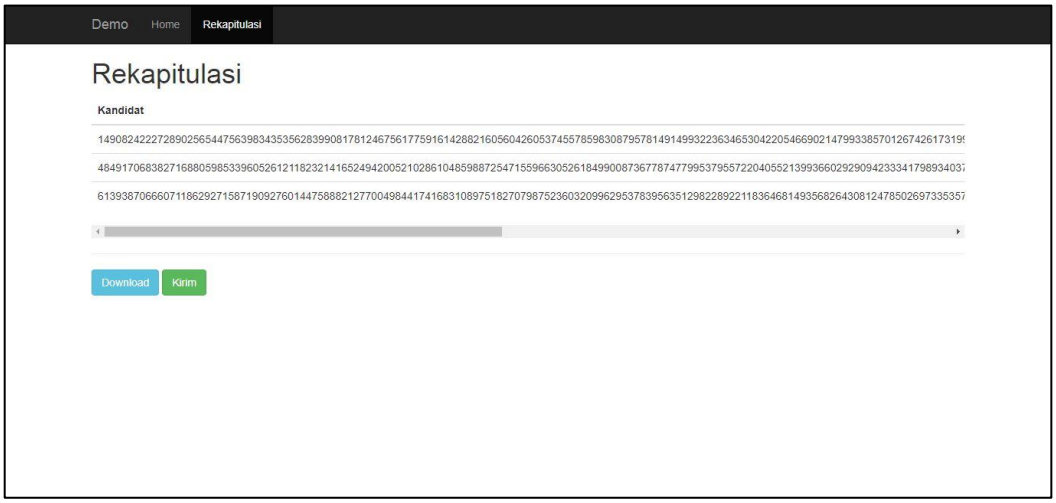
Gambar IV-1 Tampilan *client*



Gambar IV-2 Tampilan *voting place* bagian rekapitulasi

Bagian *voting place* merupakan realisasi dari *voting place* yang dimaksudkan sebelumnya. Bagian ini menerima data hasil *vote* dari pemilih kemudian menyimpan ke basis data. Hasil *vote* yang diterima dan disimpan merupakan data dalam yang masih dalam bentuk terenkripsi. Bagian ini juga berfungsi untuk melakukan rekapitulasi dari *vote* yang telah dilakukan. Rekapitulasi dilakukan pada data yang masih dalam bentuk *ciphertext* menggunakan operasi penjumlahan homomorfik yang telah disediakan oleh aplikasi. Komponen *Voting place* ini dibuat menggunakan *framework spring* untuk memudahkan implementasi. Contoh tampilan *voting place* yang dibuat dapat dilihat pada gambar IV-2.

Bagian ketiga adalah *intermediate level* yang pada pengujian kali ini juga sekaligus berperan sebagai *central voting comitee*. Bagian ini menerima hasil rekapitulasi dalam bentuk terenkripsi dari *voting place* kemudian melakukan rekapitulasi dari data-data yang diterima. Rekapitulasi dilakukan dengan menggunakan operasi penjumlahan homomorfik memanfaatkan aplikasi yang telah dibuat. Bagian ini dibuat menggunakan *framework spring* untuk memudahkan implementasi. Contoh tampilan *intermediate level* yang dibuat dapat dilihat pada Gambar IV-3.



Gambar IV-3 Tampilan *intermediate level* bagian rekapitulasi

Bagian *voting place* dan *intermediate level* dijalankan pada satu komputer yang sama dengan server lokal. Pada pengujian ini *voting place* dijalankan pada http://localhost:8081 dan *intermediate level* dijalankan pada http://localhost:8082. Keduanya dijalankan pada server Apache Tomcat. Program *e-voting* yang dibuat menggunakan basis data MySQL untuk menyimpan data-data pemilihan. Data tersebut berupa data *vote* dan data rekapitulasi. Rincian struktur tabel yang digunakan dapat dilihat pada lampiran D.

IV.2 Pengujian

Detail pengujian dibagi menjadi tiga bagian. Bagian pertama berisi tujuan pengujian. Bagian kedua berisi detail pelaksanaan pengujian. Bagian ketiga berisi analisis hasil pengujian.

IV.2.1 Tujuan Pengujian

Pengujian dilakukan untuk mengetahui keamanan perangkat lunak yang dibuat. Pengujian dilakukan dengan tujuan :

1. Memastikan kebutuhan perangkat lunak yang terdapat pada *use case* terpenuhi. Pengujian ini berupa pengujian terhadap modul-modul yang ada pada aplikasi yang dibuat dan pengujian transmisi data yang melewati jaringan.
2. Mengetahui kualitas keamanan perangkat lunak yang dibuat. Pengujian ini dilakukan dengan menggunakan metode EVSSO.

IV.2.2 Lingkungan Pengujian

Untuk detail lingkungan pengujian dapat dilihat pada Tabel IV-4.

Tabel IV-4 Lingkungan pengujian

Perihal	Keterangan
CPU	Intel(R) Core(TM) i3-3110M CPU @ 2.40GHz (4 CPUs), ~2.4GHz
Sistem Operasi	Windows 8.1 64-bit

Perihal	Keterangan
RAM	6 GB
JVM	1.8.0_74
Server	Apache Tomcat
Database	MySQL
Framework	Spring

IV.2.3 Pelaksanaan Pengujian

Pengujian keamanan dilaksanakan dalam dua bagian yaitu :

1. Pengujian fungsionalitas modul-modul yang dibuat. Pengujian ini berdasarkan *use case* yang telah dibuat. Pengujian ini termasuk pengujian data yang dikirim melalui jaringan.
2. Pengujian keamanan berdasarkan metode EVSSO. Pengujian ini dilakukan untuk mengetahui kualitas keamanan perangkat lunak *e-voting*.

Pengujian *e-voting* dilakukan langsung dengan membangun program e-voting sederhana yang memanfaatkan aplikasi *e-voting* yang dibuat. Penjelasan lebih detail mengenai program e-voting yang dibangun untuk pengujian ini akan diuraikan lebih detail.

IV.2.3.1 Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan berdasarkan tiap *use case* yang telah didefinisikan pada subbab III.3.1. Pengujian dilakukan dengan menggunakan program *e-voting* yang dibuat berdasarkan deskripsi pada subbab IV.2.2.1. Pengujian dilakukan dengan terlebih dahulu memastikan bahwa tidak ada masalah dengan integrasi aplikasi dengan *framework spring* pada server dan tidak ada masalah pada *user interface client*. Daftar pengujian yang dilakukan pada tiap *use case* dapat dilihat pada tabel IV-5.

Tabel IV-5 Daftar pengujian fungsionalitas tiap *use case*

No	Use Case	Pengujian	Jenis Pengujian	Identifikasi
1	*Vote*	Melakukan *vote* dengan memasukkan id *vote*r yang belum melakukan *vote*	Black box	RP-1-001
		Melakukan *vote* dengan memasukkan id *vote*r yang telah melakukan *vote*	Black box	RP-1-002
		Melakukan *vote* kemudian hasil *vote* tersebut didekripsi	Black box	RP-1-003
		Melakukan pengecekan data selama transmisi	Black box	RP-1-004
2	Verify *vote*	Melakukan *vote* kemudian mengubah nilai hash sebelum dikirim	Black box	RP-2-001
		Melakukan *vote* kemudian mengubah id *vote*r sebelum dikirim	Black box	RP-2-002
		Melakukan *vote* kemudian mengubah candidate *vote* sebelum dikirim	Black box	RP-2-003
3	Recapitulate	Melakukan rekapitulasi suara dari suara yang telah masuk pada *voting place*	Black box	RP-3-001
		Melakukan rekapitulasi suara dari suara yang telah masuk pada intermediate level	Black box	RP-2-002
		Melakukan *vote* kemudian melakukan rekapitulasi ulang	Black box	RP-3-003
4	Send recapitulation	Mengirimkan hasil rekapitulasi ke intermediate level	Black box	RP-4-001
		Melakukan pengecekan data selama transmisi	Black box	RP-4-002
5	Get recapitulation	Mengirimkan rekapitulasi dari *voting place* ke intermediate level. Kemudian di intermediate level di cek apakah rekapitulasi yang dikirim disimpan dalam struktur data dengan format yang benar	Black box	RP-5-001
		Mengirimkan rekapitulasi dari *voting place* ke intermediate level dengan terlebih dahulu mengubah data rekapitulasi. Kemudian di intermediate level di cek apakah rekapitulasi yang dikirim disimpan dalam struktur data dengan format yang benar	Black box	RP-5-002
6	Download recapitulation	Mengunduh rekapitulasi dari *voting place*	Black box	RP-6-001

No	Use Case	Pengujian	Jenis Pengujian	Identifikasi
		Melakukan *vote* kemudian melakukan rekapitulasi ulang, setelah itu mengunduh rekapitulasi dari *voting place*	Black box	RP-6-002
		Mengunduh rekapitulasi dari intermediate level	Black box	RP-6-003
		Mengirimkan hasil rekapitulasi baru dari *voting place* ke intermediate level, setelah itu di intermediate level dilakukan rekapitulasi ulang kemudian hasil rekapitulasi diunduh	Black box	RP-6-004

IV.2.3.2 Pengujian EVSSO

Pengujian dengan EVSSO (*Electronic Voting System Security Optimization*) akan digunakan pada pengujian kali ini. Metode pengujian ini dipilih karena dapat menguantifikasikan kualitas keamanan pada sistem *e-voting*. Metode pengujian tersebut bersifat generik dan dapat diterapkan pada sistem *e-voting* yang berbeda-beda. Selain mengukur keamanan, metode ini juga dapat memberikan kita rekomendasi bagian mana dari sistem *e-voting* yang harus diperbaiki untuk menambah kualitas keamanan.

Pengujian dengan metode EVSSO dilakukan dengan menggunakan matriks EVSSO. Matriks ini berisi *core area* yaitu aspek keamanan yang diuji dan *maturity level* yang berarti posisi pada tiap level keamanan. Evaluasi dilakukan berdasarkan kriteria-kriteria yang telah ditentukan. Terdapat tiga level yaitu level A, B dan C. Matriks EVSSO dan kriteria evaluasinya dapat dilihat pada lampiran C.

Secara umum *core area* pada matriks EVSSO terbagi atas tiga yaitu *hardware*, *software* dan *human factor*. Pada pengujian ini hanya diambil *core area software*. Alasannya karena tugas akhir ini hanya berfokus pada pengembangan perangkat lunak aplikasi *e-voting* saja. *Hardware* dan sumber daya manusia pelaksana yang digunakan oleh sistem *e-voting* berada di luar skop pembahasan tugas akhir. Kemudian pada core area software, kategori *software engineering* tidak dimasukkan karena tugas akhir ini tidak membahas mengenai metode

pengembangan *software* yang digunakan. Matriks pengujian yang digunakan dapat dilihat pada Tabel IV-6.

Tabel IV-6 Matrix EVSSO pada *core area software*

No	Core Area	Maturity Level										
		0	1	2	3	4	5	6	7	8	9	10
1	Software - Compliance with Election Principles		A				B					
2	Software - Data integrity			A			B					
3	Software - Cryptography			A				B			C	
4	Software - Transparency				A				B			
5	Software - Protection of Software				A					B		C

IV.2.4 Hasil Pengujian

Subbab ini berisi penjelasan tentang hasil pengujian yang telah dilakukan sesuai dengan penjelasan di subbab IV.2.2. Subbab ini berisi tentang hasil pengujian fungsional dan hasil pengujian menggunakan matriks EVSSO.

IV.2.4.1 Hasil Pengujian Fungsional

Pengujian fungsional dilaksanakan sesuai penjelasan pada subbab IV.2.2. Pengujian tersebut dilaksanakan dengan menyusun kasus uji tiap pengujian, menentukan target. Setelah pengujian dilakukan maka hasil yang diperoleh dicatat kemudian dibandingkan dengan target pengujian yang diharapkan. Adapun rangkuman hasil pengujian dapat dilihat pada Tabel IV-7.

Tabel IV-7 Rangkuman hasil pengujian

No	Use Case	Id Pengujian	Hasil
1	*Vote*	RP-1-001	Diterima
		RP-1-002	Diterima
		RP-1-003	Diterima
		RP-1-004	Diterima
2	Verify *vote*	RP-2-001	Diterima

No	Use Case	Id Pengujian	Hasil
		RP-2-002	Diterima
		RP-2-003	Diterima
3	Recapitulate	RP-3-001	Diterima
		RP-2-002	Diterima
		RP-3-003	Diterima
4	Send recapitulation	RP-4-001	Diterima
		RP-4-002	Diterima
5	Get recapitulation	RP-5-001	Diterima
		RP-5-002	Diterima
6	Download recapitulation	RP-6-001	Diterima
		RP-6-002	Diterima
		RP-6-003	Diterima
		RP-6-004	Diterima

Berdasarkan tabel IV-7 dapat diambil kesimpulan bahwa aplikasi yang dibangun telah memenuhi kebutuhan pada tiap-tiap *use case*. Detail mengenai hasil pengujian dapat dilihat pada lampiran D. Kemudian contoh data masukan untuk pengujian yang dijelaskan pada lampiran D dapat dilihat *ciphertext* dan *plaintext* yang bersesuaian pada lampiran E.

IV.2.4.2 Hasil Pengujian EVSSO

Hasil pengujian menggunakan kriteria pada matriks EVSSO dapat dilihat pada Tabel IV-8.

Tabel IV-8 Hasil pengujian EVSSO

No	Core Area	Maturity Level										
		0	1	2	3	4	5	6	7	8	9	10
1	Software - Compliance with Election Principles		A				B					
2	Software - Data integrity			A			B					
3	Software - Cryptography			A				B			C	
4	Software - Transparency				A				B			

No	Core Area	Maturity Level										
		0	1	2	3	4	5	6	7	8	9	10
5	Software - Protection of Software				A					B		C

Ketercapaian kriteria-kriteria pada tiap *core area* dijelaskan sebagai berikut

1. *Compliance with Election Principles*

   Aplikasi yang dibuat telah mencapai kriteria level B. Sebab data pemilihan telah tersimpan pada database, hal ini sesuai dengan pengujian RP-1-003. Walaupun data tersebut merupakan data terenkripsi namun berkorespondensi dengan data asli ketika didekripsi. Kemudian kesetaraan *vote* juga telah terpenuhi. Hal ini dibuktikan pada saat rekapitulasi hasil pada pengujian RP-3-001 dan RP-3-003. Semua pemilihan yang dilakukan dihitung satu. Kerahasiaan *vote* juga telah terpenuhi dengan dilakukannya enkripsi pada data *vote*. Kemudian standar *publicity and transparency* juga terpenuhi sebab kode sumber dari aplikasi *e-voting* ini dapat dibaca secara publik.

2. *Data integrity*

   Aplikasi yang dibuat telah mampu mencapai level A. Kategori tersebut yaitu kebenaran penyimpanan *vote*, perhitungan *vote* dan penampilan hasil. Kebenaran data saat menyimpan hasil *vote* telah tercapai sesuai dengan hasil pengujian RP-1-001, RP-1-002 dan RP-1-003. Kebenaran saat penghitungan *vote* dan penampilan hasil juga telah tercapai sesuai dengan hasil pengujian RP-3-001.

3. *Cryptography*

   Aplikasi yang dibuat sudah mampu mencapai kriteria level B. Sesuai dengan kriteria level B yaitu menggunakan algoritma yang kuat, manajemen kunci dan metode untuk mengetahui adanya perubahan data. Aplikasi yang dibuat menggunakan algoritma kriptografi kunci asimetris pailier. Secara *default*, aplikasi menggunakan kunci yang panjangnya 1024 bit sehingga dapat dikatakan aman. Aplikasi yang dibangun memanfaatkan enkripsi homomorfik, dekripsi data tidak perlu dilakukan selama proses perhitungan suara. Dekripsi data hanya dilakukan pada level paling atas sehingga kunci dekripsi hanya dimiliki oleh satu entitas saja. Aplikasi juga telah mampu

menangani perubahan pada data dengan menggunakan *hash*. Kategori level C masih belum dicapai sebab *file* konfigurasi masih menggunakan *plaintext* biasa.

4. *Transparency*

   Aplikasi yang dibuat sudah mencapai level A. Kode sumber aplikasi dapat diakses secara bebas sehingga dapat direview oleh siapa pun. Kemudian aplikasi juga telah diterapkan untuk membangun sistem e-voting untuk pengujian.

5. *Protection of Software*

   Aplikasi telah memenuhi kriteria level A. Aplikasi yang dibangun tidak terkait dengan sistem operasi yang digunakan sebab ditulis dalam bahasa Java dan berjalan diatas J*ava Virtual Machine*. Kemudian aplikasi tidak menggunakan kunci atau PIN *default*.

IV.2.5 Evaluasi Hasil Pengujian

Berdasarkan pengujian yang telah dilaksanakan selama pengujian aplikasi *e-voting* pada sistem *e-voting* sederhana yang dibuat, diperoleh evaluasi sebagai berikut :

1. Berdasarkan hasil kebutuhan fungsional, dapat ditarik kesimpulan bahwa aplikasi e-voting yang dibuat sudah mencakup *use case – use case* yang telah didefinisikan sebelumnya. Aplikasi tersebut dapat digunakan untuk membangun sistem *e-voting*. Pada pengujian dilakukan integrasi aplikasi *e-voting* dengan *framework spring*. Dengan memanfaatkan aplikasi *e-voting*, pembangunan sistem *e-voting* menjadi lebih sederhana. Contohnya untuk membangun sistem sederhana seperti yang digunakan pada pengujian, hanya membutuhkan tambahan dua kelas *controller* dan sebuah kelas yang berisi definisi data yang menyatakan model pada basis data.
2. Berdasarkan matriks EVSSO pada Tabel IV-8 terlihat bahwa hal yang menjadi prioritas pengembangan saat pembuatan perangkat lunak e-voting yang memanfaatkan aplikasi yang telah dibuat adalah *data integrity* level B. Hal tersebut menjadi prioritas sebab pada matriks EVSSO menunjukkan bahwa level B pada data integrity berada pada *maturity level* dengan nilai 5, paling kecil di antara yang lainnya. Hal yang dapat dilakukan adalah

melakukan proteksi untuk menjaga *reliability* data. Hal tersebut dapat dilakukan dengan melakukan *backup* data secara berkala di *non-volatile storage* contohnya *hard disk*. Hal tersebut dapat ditangani salah satunya dengan menggunakan DBMS (*Database Management System*) untuk menyimpan data.

3. Enkripsi dengan algoritma Pailier menggunakan variabel acak sehingga *plaintext* yang sama apabila dienkripsi dapat menghasilkan *cipher text* yang berbeda. Hal tersebut menjadi masalah pada saat rekapitulasi data sebab jumlah suara akan dihitung berdasarkan *id* kandidat. Permasalahan tersebut dapat diatasi dengan mengatur nilai variabel acak pada saat enkripsi untuk kasus-kasus yang memerlukan hasil enkripsi yang konsisten untuk tiap *plaintext*.
4. Operasi perhitungan suara lebih lambat dengan menggunakan enkripsi homomorfik menjadi lebih lambat dibandingkan dengan jika data tidak terenkripsi. Sebagai perbandingan, operasi penjumlahan secara homomorfik rata-rata memakan waktu 45330 ns, sedangkan operasi penjumlahan biasa hanya memakan waktu rata-rata 482 ns. Hal ini disebabkan karena operasi penjumlahan homomorfik sama halnya dengan perkalian dua buah *big integer*. Terlebih lagi *big integer* yang digunakan sangat besar yakni 1024 bit. Hal tersebut menjadi masalah jika sistem *e-voting* yang dibuat menampilkan hasil pemilihan secara *real time* dengan jumlah pemilih yang besar.
5. Aplikasi *e-voting* yang dibuat belum menyediakan fitur autentikasi dan autorisasi. Fitur autentikasi dan autorisasi dapat dibuat terpisah dengan sistem *e-voting* yang dibuat.

BAB V

KESIMPULAN DAN SARAN

Bab ini berisi mengenai kesimpulan yang didapat dari proses pelaksanaan tugas akhir dan saran yang dapat digunakan untuk pengembangan selanjutnya.

V.1 Kesimpulan

Tugas akhir yang membahas mengenai pengembangan aplikasi untuk data terenkripsi homomorfik memberikan kesimpulan :

1. Enkripsi homomorfik parsial yang bersifat additif dapat diterapkan pada sistem pemilihan elektronik (*e-voting*). Enkripsi homomorfik yang bersifat additif dipilih karena operasi perhitungan suara pada pemilihan menggunakan operasi penjumlahan. Selain itu enkripsi homomorfik memungkinkan tidak dilakukannya dekripsi data terlebih dahulu ketika melakukan rekapitulasi hasil pemilihan.
2. Pada tugas akhir ini dihasilkan sebuah perangkat lunak berupa aplikasi yang berguna untuk membangun sistem *e-voting*. Aplikasi yang dibangun menggunakan algoritma kriptografi Pailier sebagai algoritma enkripsi homomorfik parsial yang digunakan.
3. Aplikasi yang dibangun berhasil diterapkan untuk membangun sistem *e-voting* sederhana dan telah dilakukan pengujian pada sistem *e-voting* tersebut.
4. Aplikasi yang dibangun telah menyediakan fitur keamanan yang baik dari sisi kesesuaian dengan asas-asas pemilihan dan dari sisi kriptografi. Kekurangan yang masih dialami adalah belum menyediakan fitur untuk *reliability* data.

V.2 Saran

Beberapa saran untuk pengembangan perangkat lunak dalam tugas akhir ini atau pun penelitian yang dapat dilakukan lebih lanjut adalah sebagai berikut :

1. Aplikasi dapat ditambahkan modul baru yaitu modul untuk melakukan autentikasi dan autorisasi.
2. Pengembangan aplikasi lebih lanjut dapat dilakukan dengan menambahkan fitur enkripsi pada *file* konfigurasi.
3. Aplikasi dapat ditambahkan modul untuk melakukan pertukaran kunci yang berguna untuk mendistribusikan kunci publik kepada klien-klien, sehingga kunci publik tidak perlu disimpan pada *file* konfigurasi.
4. Pengembangan lebih lanjut dapat dilakukan dengan membuat aplikasi klien pada *Javascript* sehingga pemilihan dapat dilakukan pada web.

DAFTAR PUSTAKA

Azhari, Rakhmad (2005). *e-voting*. Fakultas Ilmu Komputer Universitas Indonesia.

Djanali Supeno dkk (2016). *Design and Development of Voting Data Security for Electronic Voting (E-voting).* Institut Teknologi Sepuluh Nopember.

Gentri C., Halevi S (2011). *Implementing Gentry's Fully-Homomorphic Encryption Scheme*. IBM Research.

Gentry, Craig (2009). *A Fully Homomorphic Encryption Scheme.* Stanford University.

Kahani, M. (2005), *Experiancing small-scale e-democracy in Iran.* The Electronic Journal On Information System in Developing Contries.

Lauther, Kristin dkk (2011). *Can Homomorphic Encryption Be Practical ?*. Microsoft.

https://www.microsoft.com/en-us/research/publication/can-homomorphic-encryption-be-practical/ . Diakses pada 15 Oktober 2016.

Morris, Liam (2013). *Analysis of Partial and Fully Homomorphic Encription.* Rochester Institute of Technology.

Munir, Rinaldi (2005). *Diktat Kuliah IF5051 Kriptografi*. Departemen Teknik Informatika Institut Teknologi Bandung.

Ondrisek, Barbara (2009). *E-Voting Security Optimization*. Vienna University of Technology.

Pailier, Pascal (1999). *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*. Gemplus Card International.

Potzelsberger (2013). *KV Web Security: Application of Homomorphic Encryption.*

http://www.fim.uni-linz.ac.at/lva/Web_Security/Abgaben/Poetzelsberger-Homomorphic.pdf . Diakses pada 15 Oktober 2016

Schneier, Bruce (1999). *Applied Cryptography 2nd edition*, John Wiley & Sons.

Shinde Shubhangi dkk (2013). *Secure E-voting Using Homomorphic Technology*. Terna Engineering College.

Sophan, M (2012). *Design Model TPS Dalam Sistem Pemilihan Kepala Daerah.* Universitas Trunojoyo Madura.

LAMPIRAN

Lampiran A. Pseudo Code

A.1 Method kelas *vote*

```
function setvote(integer id, integer candidate)
     candidat_vote = candidate
     id = id

functions sendvote(vote vote, url)
     send(endpoint, vote)
```

A.2 Method kelas *voting place*

```
 function add_vote(vote)
     if(verifiy_vote(vote))
          votes.add(vote)
          add_to_database(vote)

 function get_vote(integer id)
     votes.get(id)

 function verify_vote(vote) : boolean
     id = vote.getid.getbytes
     candidate = vote.getcandidate.getbytes
     data = concatbytes(id, candidate);
     hash = gethash(data)
     return vote.hash == hash

 function recap(votes) : recapCollection
     one = homomorphic.one
     zero = homomorphic.zero
     for each(vote in votes) do
     nvotes=recapCollection.get(vote.candidate).getnumvotes()
          nvotes = homomorphic_add(nvotes, one)
          recapCollection.get(vote.candidate).setnumvotes(nvote)
     return recapCollection

 functions send_recap(recapCollection, endpoint)
     send(endpoint, recapCollection)
```

A.3 Method kelas homomorphic operation

```
function add(num1, num2) : integer
     return num1 * num 2

function multiply(num, factor) -> integer
     result = num
     for(i = 1; i < factor; i++)
          result = add(result, num)
```

```
     return  result
```

A.4 Method kelas intermediate level

```
function add_recap(recapitulation)
     if(verify_recap(recapitulation))
          recapCollection.add(recapitulation)

function verify_recap(recapitulation) : boolean
      candidate = recapitulation.getCandidate()
      numvote = recapitulation.getNumVote()
      data          =          concatebytes(candidate.getbytes(),
      numvote.getbytes())
      hash = hash(data)
     return hash == recapitulation.getHash()
function accum_recap()
      recap_accum=new Recapitulatio[number_of_candidate]
      foreach(recapitulation in recapCollection) do
          recap_accum[recapitulation.candidate]                    =
      hm_add(recap_accum[recapitulation.candidate],
      recapitulation.getNumVotes)
      recapCollection =new RecapCollection(recap_accum)
      return recapCollection

function send_recap(endpoint)
      recapitulation.generateHash()
     send(recapitulation, endpoint)
```

A.5 Method kelas pailier

```
function encrypt(plaintext, publicKey) : integer
      do {
          r = new BigInteger(bits, new Random())
      } while (r.compareTo(n) >= 0)
      result = g.modPow(m, nSquared)
      x = r.modPow(n, nSquared)
      result = result.multiply(x)
      result = result.mod(nSquared)
     return result

function decrypt(chiper, key) : integer
      n = key.getN()
      nSquare = key.getnSquared()
      lambda = key.getLambda()
      u = key.getPreCalculatedDenominator()
      p=c.modPow(lambda,
      nSquare).subtract(BigInteger.ONE).divide(n).multiply(u).mo
      d(n)
      if (upperBound != null && p.compareTo(upperBound) > 0) {
          p = p.subtract(n)
      }
     return p
```

Lampiran B. Implementasi Kode Program

B.1 Implementasi Kelas *Vote*

```
package vote;

import java.io.Serializable;
import java.math.BigInteger;

public class Vote implements Serializable{
     private static final long serialVersionUID = 1L;
     private BigInteger id;
     private BigInteger votingPlaceId;
     private BigInteger candidateVote;

     public Vote(BigInteger id, BigInteger votingPlaceId,
BigInteger candidateVote)

     public BigInteger getId()

     public void setId(BigInteger id)

     public BigInteger getVotingPlaceId()

     public void setVotingPlaceId(BigInteger votingPlaceId)

     public BigInteger getCandidateVote()

     public void setCandidateVote(BigInteger candidateVote)
}
```

```
package vote;
import java.io.IOException;
import java.io.Serializable;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import utils.Serializer;

public class VotewHash implements Serializable{
     private static final long serialVersionUID = 1L;
     private Vote vote;
     private byte[] hash;

     public VotewHash(Vote vote, byte[] hash)

     public VotewHash(Vote vote)

     private byte[] voteDigest() throws IOException,
NoSuchAlgorithmException

     public void generateHash()

     public boolean vefifyHash()
```

```
    public Vote getVote()

    public byte[] getHash()
}
```

B.2 Implementasi Kelas Pailier

```
package crypto;
import java.math.BigInteger;
import java.util.Properties;
import utils.Prop;

public class Pailier {

    private final PrivateKey privateKey;
    private final PublicKey publicKey;
    private final BigInteger upperBound;

    public Pailier(PrivateKey privateKey, PublicKey publicKey,
BigInteger upperBound)

    public Pailier(String config)

    public PrivateKey getPrivateKey()

    public PublicKey getPublicKey()

    public final BigInteger decrypt(BigInteger c)
}
```

```
package crypto;
import java.math.BigInteger;
import java.util.Properties;
import java.util.Random;
import utils.Prop;

public class PublicKey {
    private final int bits;
    private final BigInteger n;
    private final BigInteger nSquared;
    private final BigInteger g;

    public PublicKey(BigInteger n, BigInteger nSquared,
BigInteger g, int bits)
    public PublicKey(String config)

    public final BigInteger encrypt(BigInteger m)
}
```

```
package crypto;
import java.math.BigInteger;

public class PrivateKey {
    private final BigInteger lambda;
    private final BigInteger preCalculatedDenominator;

    public     PrivateKey(BigInteger     lambda,     BigInteger
preCalculatedDenominator)

    public BigInteger getLambda()

    public BigInteger getPreCalculatedDenominator()
}
```

```
package crypto;
import java.math.BigInteger;
import java.security.SecureRandom;
import java.util.Random;

public class KeyPairBuilder {
    private int bits = 1024;
    private int certainty = 0;
    private Random rng;
    private BigInteger upperBound;

    public KeyPairBuilder bits(int bits)

    public KeyPairBuilder certainty(int certainty)

    public KeyPairBuilder randomNumberGenerator(Random rng)

    public KeyPairBuilder upperBound(BigInteger b)

    public Pailier generateKeyPair()

    private BigInteger calculateL(BigInteger u, BigInteger n)

    private BigInteger lcm(BigInteger a, BigInteger b)
}
```

B.3 Implementasi Interface Sender

```
package vote;

public interface Sender {
     public void send(String url) throws Exception;
}
```

B.4 Implementasi Kelas *Voting place*

```
package collector;

import java.util.ArrayList;
import crypto.HomomorphicOperation;
import recap.Recapitulation;
import recap.RecapCollection;

public class VotingPlace {

     private int id;
     private ArrayList<VotewHash> votes;
     private RecapCollection recapCollection;

     public VotingPlace()

     public  VotingPlace(int  id,  ArrayList<VotewHash>  votes,
RecapCollection recapCollection)

     public VotingPlace(int id, ArrayList<VotewHash> votes)

     public void addVote(VotewHash vote)

     public void recapVote()

     public int getId()

     public void setId(int id)

     public ArrayList<VotewHash> getVotes()

     public void setVotes(ArrayList<VotewHash> votes)

     public RecapCollection getRecapCollection()

     public        void        setRecapCollection(RecapCollection
recapCollection)
}
```

B.5 Implementasi Kelas Intermediate Level

```
package collector;

import java.util.ArrayList;
import recap.Recapitulation;
import recap.RecapCollection;

public class IntermediateLevel {
     private int levelId;
     private RecapCollection recapCollection;
```

```
    public IntermediateLevel(int levelId, RecapCollection
recapCollection)

    public void addRecapitulation(RecapCollection recaps)

    public void updateHash()

    public RecapCollection getRecapCollection()

    public void setRecapCollection(RecapCollection
recapCollection)

    public int getLevelId()

    public void setLevelId(int levelId)
}
```

B.6 Implementasi Kelas Recapitulation

```
package recap;

import java.io.Serializable;
import java.math.BigInteger;

import crypto.HomomorphicOperation;

public class Recapitulation implements Serializable {

    private BigInteger candidate;
    private BigInteger numVotes;

    public Recapitulation()

    public Recapitulation(BigInteger candidate, BigInteger
numVotes)

    public void increaseVotes()

    public void addNumVotes(BigInteger n)

    public BigInteger getCandidate()

    public void setCandidate(BigInteger candidate)

    public BigInteger getNumVotes()

    public void setNumVotes(BigInteger numVotes)
}
```

B.7 Implementasi Kelas RecapCollection

```
package recap;

import java.io.IOException;
import java.io.Serializable;
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Arrays;

import utils.Serializer;

public class RecapCollection implements Serializable{

    private ArrayList<Recapitulation> recapArray;
    private byte[] hash;

    public RecapCollection()

    public RecapCollection(ArrayList<Recapitulation> recaps)

    public  RecapCollection(ArrayList<Recapitulation>  recaps,
byte[] hash)

    public void addRecap(Recapitulation recap)

    public       byte[]       getRecapsDigest()       throws
NoSuchAlgorithmException, IOException

    public boolean checkHash()

    public void generateHash()

    public boolean hasCandidateId(BigInteger id)

    public void increase_Vote_Candidate(BigInteger id)

    public void addNum_Vote_Candidate(BigInteger id, BigInteger n)

    public ArrayList<Recapitulation> getRecapArray()

    public     void     setRecapArray(ArrayList<Recapitulation>
recapArray)

    public byte[] getHash()

    public void setHash(byte[] hash)
}
```

B.8 Implementasi Kelas Homomorphic Operation

```
package crypto;

import java.math.BigInteger;

public class HomomorphicOperation {

     public static BigInteger add(BigInteger a , BigInteger b) {
          return a.multiply(b);
     }

     public static BigInteger multiply(BigInteger number, int
factor) {
          BigInteger result = number;
          for(int i = 0; i < factor-1; i++) {
               result = result.multiply(number);
          }
          return result;
     }
}
```

Lampiran C. Matriks EVSSO

Matriks EVSSO lengkap dapat dilihat pada Tabel 1.

Tabel 1 Matriks EVSSO lengkap

No	Core Area	Maturity Level										
		0	1	2	3	4	5	6	7	8	9	10
1	Hardware - Compliance with Election Principles		A			B			C			
2	Hardware - Safety			A			B			C		
3	Hardware - Physical Security			A			B				C	
4	Hardware - Cryptography			A		B						
5	Software - Complience with Election Principles		A				B					
6	Software - Data integrity			A			B					
7	Software - Cryptography			A				B			C	
8	Software - Transparency				A				B			
9	Software - Software Engineering			A			B					C
10	Software - Protection of Software				A					B		C
11	Human Factor - Complience with Election Principles		A			B						
12	Human Factor - Secutiry Management				A				B			
13	Human Factor - User Interface			A		B						C
14	Human Factor - Transparency			A			B				C	
15	Human Factor - Oranization of Election					A			B			
16	Human Factor - Validation of Independent Testing Authority							A				

1 Hardware - Compliance with Election Principles

Level A:

- secrecy of *votes*: no linear traceability or traceability by time
- secrecy of *votes*: use of polling booths

Level B:

- publicity and transparency: access to devices, storage media (cards), and their documentation granted before election
- limit access to devices and storage media during build-up or use

Level C:

- storage, locking, and sealing of devices and storage media, periodic controls and documentation of safety precautions

- secrecy of *vote*s: protection from Tempest attacks and minimization of noise level

2 Hardware - Safety

Level A:

- correctness of construction
- homogeneous architecture
- safety of construction and installation
- availability

Level B:

- capacitance
- durability, reliability
- protection during transport and storage
- power supply

Level C:

- emergency plan
- absence of reaction

3 Hardware - Physical Security

Level A:

- protection from physical attacks
- no (possibility for an) internet connection
- removal of unused devices and interfaces
- access control for machines

Level B:

- protection of storage media: physical attacks, ability to remove modules, methods to identify copies

Level C:

- protection from (internal) Denial of Service Attacks, redundant systems
- setting of BIOS passwords

4 Hardware - Cryptography

Level A:

- use of secure anonymous connections for confidentiality
- telecommunications security for confidentiality and data integrity

Level B:

- hardware encryption of hard disks

5 Software - Compliance with Election Principles

Level A:

- general right to *vote*: correct collection of the *vote*rs' data in the *vote*rs' register
- equality of *vote*s: one ballot per *vote*r, every *vote* counts the same
- secrecy of *vote*s: anonymous channels, encrypted connections, anonymous ballot casting to urn server

Level B:

- publicity and transparency: analysis of source code with peer reviews, plausibility of casting of *vote*s
- public list of additional software used

Level C:

- publicity and transparency: analysis of source code of firmware, device drivers and used Commercial Off-the-Shelf (COTS) products used through peer reviews

6 Software - Data Integrity

Level A:

- correct implementation of *vote* storage, *vote* counting, and result display
- management and audit functions of application

Level B:

- protection of data and reliability: loss of data after crash of machine (backup systems)
- use of synchronized internal clocks of machines

7 Software - Cryptography

Level A:

- trusted paths / channels
- protection of user data and ballots
- use of asymmetric keys for identification, authentication, and authorization

Level B:

- use of strong up-to-date algorithms
- key management
- methods to recognize duplicates or manipulations of storage media

Level C:

- encryption of configuration files and databases

8 Software - Transparency

Level A:

- open source code for inspection by third parties, peer reviews repeated test elections prior to legally valid elections

Level B:

- test of COTS products and operating system internal error analysis system

9 Software - Software Engineering

Level A:

- quality management: reviews in every phase of development process
- risk management in planning phase
- traceability of anonymous *vote*s
- user input checks
- interoperability with existing systems

Level B:

- quality management: automatic and manual testability
- implementation guidelines
- security tests and auditing after implementation: black box tests

Level C:

- team split-up: dual development
- security tests and auditing after implementation: review of security characteristics of application, white box tests, penetration tests
- code analyses (metrics)
- grant future interoperability by using open, not proprietary standards

10 Software - Protection of Software

Level A:

- homogeneous operation systems with up-to-date security updates
- no default passwords or PINs, strong passwords

Level B:

- version checks and checks of integrity of source code, of external (standard) libraries used, and of configuration files
- secure update mechanism
- uninstallation of unused pre-installed software
- scalability

Level C:

- authenticity checks of compiler
- protection from diverse software security risks
- protection from man-in-the-middle attacks

11 Human Factors - Compliance with Election Principles

Level A:

- free right to *vote*: neutral design of ballot and voting machine. The *vote*r is able to cast an invalid *vote* (for no party / candidate).
- personal right to *vote*: *vote* personally without representative
- general right to *vote*: clarification and introductory training for *vote*rs in e-voting system

Level B:

- publicity and transparency: inspection of source code by electoral commission, access to voting machines, and verifiability with paper receipts

12 Human Factors - Security Management

Level A:

- introductory training for employees and creation of security plans
- security measure: dual control
- documentation of handling of software, hardware, and storage media for a permanent comprehensible control of all processes

Level B:

- background checks of employees
- security awareness trainings for all (external) employees

13 Human Factors - User Interface

Level A:

- representation of ballot on one screen page without scrolling
- order of parties / candidates has to correspond to order on paper ballot
- adequate representation of course of casting of *vote*s
- no illegal interference with process of election
- privacy during voting
- correct representation of ballot
- feedback during and after casting of *vote*s

- online help pages for every single step (context sensitive)
- acceptable response times of application

Level B:

- multilingualism
- magnification of screen (magnifying glass function) for visually impaired *vote*rs
- audio support for blind *vote*rs

Level C:

- usability checks with a representative test group

14 Human Factors - Transparency

Level A:

- provision of information on all topics of elections
- public announcement of district results
- possibility for independent election observation prior to, during, and after elections
- checks if number of ballots cast corresponds to number of *vote*rs who requested a ballot
- anonymous paper ballots

Level B:

- audit phase after election with lessons learned for next election comparison of exit polls of current and previous years
- definition of thresholds for manual recounts

Level C:

- voting protocol which can be discussed in public and that is publicly available, strong cryptographic algorithms

15 Human Factors - Organization of Election

Level A:

- registration for electronic voting should not be a stumbling block for citizens
- begin and end of electronic elections at the same time as for conventional elections
- prevention of delays of casting of *vote*s
- parallel service: alternative voting with paper ballot is possible
- possibility of a test *vote*

Level B:

- acceptance and distribution (expense factor): absorption of costs by state / county

16 Human Factors - Validation of Independent Testing Authority

Level A:

- examination, certification or test of correctness and security by independent testing authorities (ITA)

Lampiran D. Pengujian

D.1 Struktur Tabel Pada Pengujian

Struktur tabel basis data yang digunakan dapat dilihat pada Tabel 2.

Tabel 2 Struktur tabel yang digunakan pada pengujian

No	Nama Tabel	Nama Kolom	Tipe Data	Ukuran
1	Voting data	id (primary key)	integer	11
		Uid	varchar	255
		Candidate	varchar	255
		vpid	integer	11
		hash	varchar	255
2	Recap data	id (primary key)	integer	11
		candidate	varchar	255
		n*vote*	varchar	255
		vpid	integer	11

D.2 Pengujian *Use Case Vote*

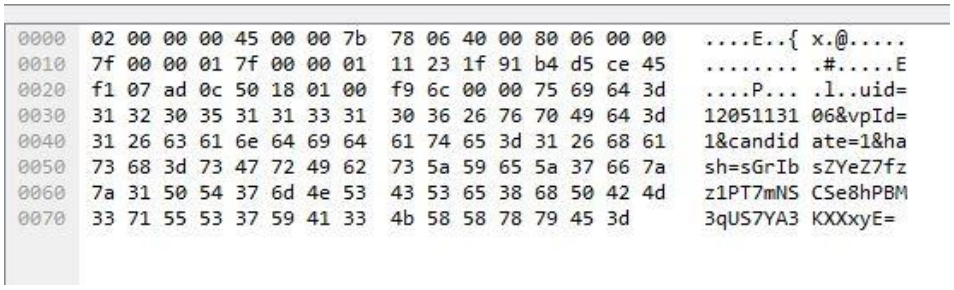
Pengujian *use case vote* dapat dilihat pada Tabel 3.

Tabel 3 Pengujian *use case vote*

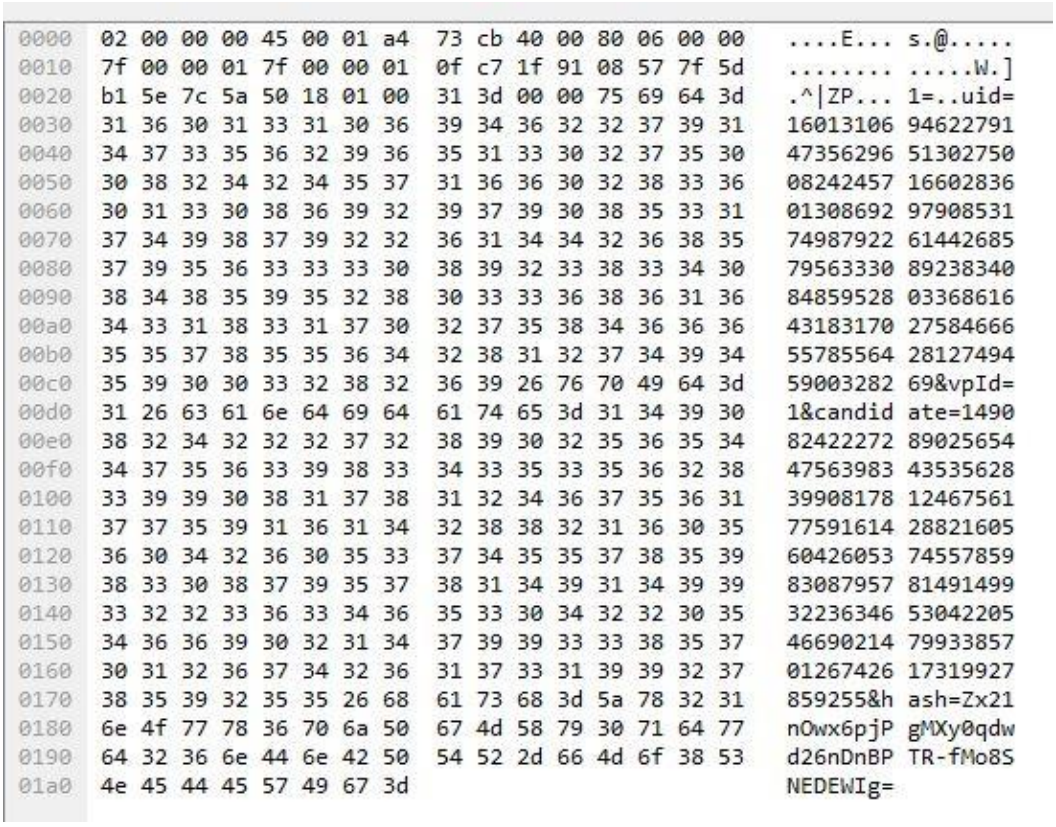
Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-1-001	Pengujian *vote*	1. Masukkan id 1205113101 2. Memilih kandidat pertama 3. Mengklik tombol pilih kandidat 4. Mengambil record database yang bertambah lalu mendekripsinya	Mengetikkan id = 1205113101 dan memilih radio button kandidat pertama (Muhtar)	1. Record database bertambah. 2. Hasil dekripsi record baru adalah uid = 1205113101 dan candidate *vote* = 1	1. Record database bertambah. 2. Hasil dekripsi record baru adalah uid = 1205113101 dan candidate *vote* = 1	Diterima

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-1-002	Pengujian duplikasi *vote*	1. Masukkan id 1205113101 2. Memilih kandidat pertama 3. Mengklik tombol pilih kandidat 4. Mengambil record database yang bertambah lalu mendekripsinya	Mengetikkan id = 1205113101 dan memilih radio button kandidat kedua	Tidak ada record baru yang bertambah	Tidak ada record baru yang bertambah	Diterima
RP-1-003	Pengujian visibilitas pengiriman	1. Masukkan id 1205113102 2. Memilih kandidat pertama 3. Mengklik tombol pilih kandidat 4. Buka wireshark dan amati paket yang dikirim	Mengetikkan id = 1205113102 dan memilih radio button kandidat pertama	Transmisi data yang tampil pada wireshark memperlihatkan bahwa data terenkripsi	Transmisi data yang tampil pada wireshark memperlihatkan bahwa data terenkripsi	Diterima

Hasil pengujian pengiriman data pemilihan dapat dilihat pada Gambar 2. Sebagai perbandingan Gambar 1 memperlihatkan data dengan nilai yang sama dengan Gambar 2 yang dikirim namun pada Gambar 1 data tidak dienkripsi.



Gambar 1 Hasil penangkapan pengiriman *vote* tanpa dienkripsi



Gambar 2 Hasil penangkapan pengiriman *vote*

D.3 Pengujian *Use Case Verify Vote*

Rincian pengujian *Use Case Verify Vote* dapat dilihat pada Tabel 4.

Tabel 4 Pengujian use case verify *vote*

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-2-001	Pengujian perubahan *hash*	1. Masukkan id 1205113103 2. Memilih kandidat pertama 3. Mengklik tombol pilih kandidat 4. Mengubah satu byte hash. Perubahan	Mengetikkan id = 1205113103 dan memilih radio button kandidat pertama	Record database tidak bertambah	Record database tidak bertambah	Diterima

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
		dilakukan pada kode program				
RP-2-002	pengujian perubahan *id*	1. Masukkan id 1205113104 2. Memilih kandidat pertama 3. Mengklik tombol pilih kandidat 4. Mengubah id menjadi 10. Perubahan dilakukan pada kode program	Mengetikkan id = 1205113104 dan memilih radio button kandidat pertama	Record database tidak bertambah	Record database tidak bertambah	Diterima
RP-3-003	Pengujian perubahan *candidate vote*	1. Masukkan id 1205113105 2. Memilih kandidat pertama 3. Mengklik tombol pilih kandidat 4. Mengubah candidate *vote* menjadi 3. Perubahan dilakukan pada kode program	Mengetikkan id = 1205113105 dan memilih radio button kandidat pertama	Record database tidak bertambah	Record database tidak bertambah	diterima

D.4 Pengujian Use Case *Recapitulate*

Rincian pengujian *Use Case Recapitulate* dapat dilihat pada Tabel 5.

Tabel 5 Pengujian *use case recapitulate*

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-3-001	Pengujian rekapitulasi *voting place*	1. Menghapus seluruh record yang tersimpan di database. Menghapus seluruh data yang disimpan pada tabel voting data.	id dan candidate *vote* yaitu id = 1205113101, candidate = 1, id = 1205113102, candidate = 1, id =	Muncul hasil rekapitulasi dengan format terenkripsi. Hasil dekripsi menghasilkan kandidat dan	Muncul hasil rekapitulasi dengan format terenkripsi. Hasil dekripsi menghasilkan kandidat dan	diterima

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
		2. Melakukan *vote* empat *vote* dengan id dan candidate *vote* yaitu id = 1205113101, candidate = 1, id = 1205113102, candidate = 1, id = 1205113103, candidate = 2, dan id = 1205113104, candidate = 3 3. Membuka http://localhost"8081 4. Mengambil data rekapitulasi kemudian mendekripsinya	1205113103, candidate = 2, dan id = 1205113104, candidate = 3	jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	
RP-3-002	Pengujian rekapitulasi pada intermediate level	Mengirimkan rekapitulasi pada *voting place* ke intermediate level	Menekan tombol kirim	Muncul hasil rekapitulasi dengan format terenkripsi. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	Muncul hasil rekapitulasi dengan format terenkripsi. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	Diterima
RP-3-003	Pengujian rekapitulasi dengan penambahan *vote*	1. Melakukan *vote* dengan id = 1205113105 dan kandidat 2 2. Membuka laman rekapitulasi http://localhost:8081	Mengetikkan id = 1205113102 dan memilih radio button kandidat kedua	Muncul hasil rekapitulasi dengan format terenkripsi. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2	Muncul hasil rekapitulasi dengan format terenkripsi. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2	Diterima

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
				jumlah *vote* 2 dan kandidat 3 jumlah *vote* 1	jumlah *vote* 2 dan kandidat 3 jumlah *vote* 1	

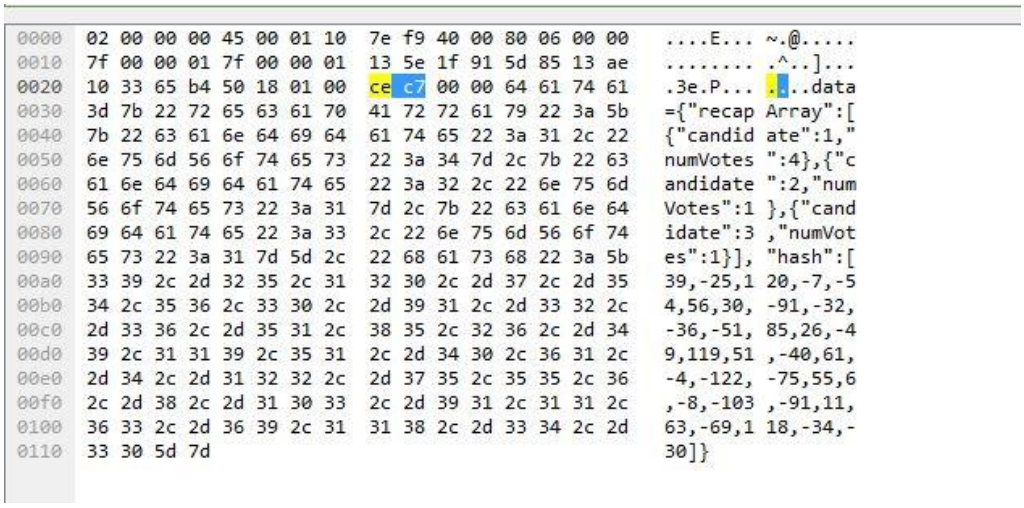
D.5 Pengujian *Use Case Send Recapitulation*

Rincian pengujian *Use Case Send Recapitulation* dapat dilihat pada Tabel 6.

Tabel 6 Pengujian *use case send recapitulation*

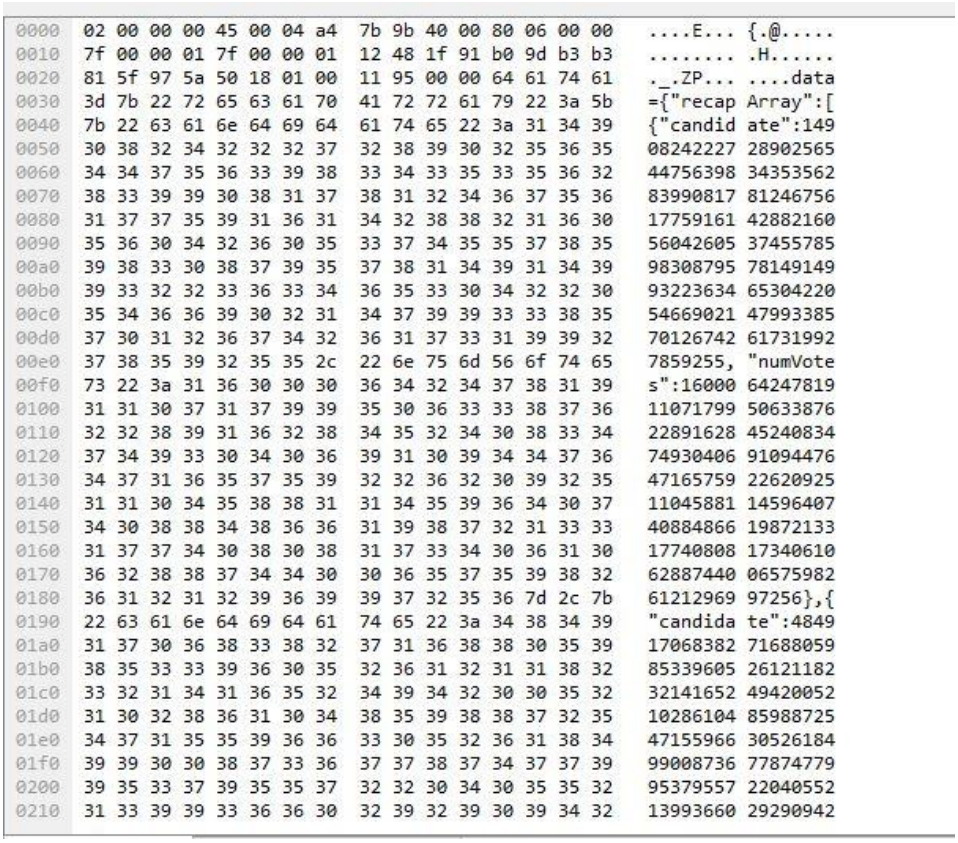
Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-4-001	Pengujian pengiriman rekapitulasi ke intermediate level	Pada laman rekapitulasi di *voting place* , tekan tombol kirim.	Menekan tombol kirim	Data diterima oleh intermediate level	Data diterima oleh intermediate level	diterima
RP-4-002	Pengujian visibilitas pengiriman	1. Pada laman rekapitulasi di *voting place* , tekan tombol kirim. 2. Buka wireshark dan amati paket yang dikirim	Menekan tombol kirim	Transmisi data yang tampil pada wireshark memperlihatkan bahwa data terenkripsi	Transmisi data yang tampil pada wireshark memperlihatkan bahwa data terenkripsi	Diterima

Hasil pengujian pengiriman paket rekapitulasi dapat dilihat pada Gambar 4. Sebagai perbandingan Gambar 3 memperlihatkan data dengan nilai yang sama dengan Gambar 4 yang dikirim namun pada Gambar 3, data tidak dienkripsi.



```
0000  02 00 00 00 45 00 01 10  7e f9 40 00 80 06 00 00   ....E... ~.@.....
0010  7f 00 00 01 7f 00 00 01  13 5e 1f 91 5d 85 13 ae   ........ .^..]...
0020  10 33 65 b4 50 18 01 00  ce c7 00 00 64 61 74 61   .3e.P... ....data
0030  3d 7b 22 72 65 63 61 70  41 72 72 61 79 22 3a 5b   ={"recap Array":[
0040  7b 22 63 61 6e 64 69 64  61 74 65 22 3a 31 2c 22   {"candid ate":1,"
0050  6e 75 6d 56 6f 74 65 73  22 3a 34 7d 2c 7b 22 63   numVotes ":4},{"c
0060  61 6e 64 69 64 61 74 65  22 3a 32 2c 22 6e 75 6d   andidate ":2,"num
0070  56 6f 74 65 73 22 3a 31  7d 2c 7b 22 63 61 6e 64   Votes":1 },{"cand
0080  69 64 61 74 65 22 3a 33  2c 22 6e 75 6d 56 6f 74   idate":3 ,"numVot
0090  65 73 22 3a 31 7d 5d 2c  22 68 61 73 68 22 3a 5b   es":1}], "hash":[
00a0  33 39 2c 2d 32 35 2c 31  32 30 2c 2d 37 2c 2d 35   39,-25,1 20,-7,-5
00b0  34 2c 35 36 2c 33 30 2c  2d 39 31 2c 2d 33 32 2c   4,56,30, -91,-32,
00c0  2d 33 36 2c 2d 35 31 2c  38 35 2c 32 36 2c 2d 34   -36,-51, 85,26,-4
00d0  39 2c 31 31 39 2c 35 31  2c 2d 34 30 2c 36 31 2c   9,119,51 ,-40,61,
00e0  2d 34 2c 2d 31 32 32 2c  2d 37 35 2c 35 35 2c 36   -4,-122, -75,55,6
00f0  2c 2d 38 2c 2d 31 30 33  2c 2d 39 31 2c 31 31 2c   ,-8,-103 ,-91,11,
0100  36 33 2c 2d 36 39 2c 31  31 38 2c 2d 33 34 2c 2d   63,-69,1 18,-34,-
0110  33 30 5d 7d                                       30]}
```

Gambar 3. Hasil pengiriman pengiriman rekapitulasi melalui wireshark tanpa enkripsi



```
0000  02 00 00 00 45 00 04 a4  7b 9b 40 00 80 06 00 00   ....E... {.@.....
0010  7f 00 00 01 7f 00 00 01  12 48 1f 91 b0 9d b3 b3   ........ .H......
0020  81 5f 97 5a 50 18 01 00  11 95 00 00 64 61 74 61   ._.ZP... ....data
0030  3d 7b 22 72 65 63 61 70  41 72 72 61 79 22 3a 5b   ={"recap Array":[
0040  7b 22 63 61 6e 64 69 64  61 74 65 22 3a 31 34 39   {"candid ate":149
0050  30 38 32 34 32 32 32 37  32 38 39 30 32 35 36 35   08242227 28902565
0060  34 34 37 35 36 33 39 38  33 34 33 35 33 35 36 32   44756398 34353562
0070  38 33 39 39 30 38 31 37  38 31 32 34 36 37 35 36   83990817 81246756
0080  31 37 37 35 39 31 36 31  34 32 38 38 32 31 36 30   17759161 42882160
0090  35 36 30 34 32 36 30 35  33 37 34 35 35 37 38 35   56042605 37455785
00a0  39 38 33 30 38 37 39 35  37 38 31 34 39 31 34 39   98308795 78149149
00b0  39 33 32 32 33 36 33 34  36 35 33 30 34 32 32 30   93223634 65304220
00c0  35 34 36 36 39 30 32 31  34 37 39 39 33 33 38 35   54669021 47993385
00d0  37 30 31 32 36 37 34 32  36 31 37 33 31 39 39 32   70126742 61731992
00e0  37 38 35 39 32 35 35 2c  22 6e 75 6d 56 6f 74 65   7859255, "numVote
00f0  73 22 3a 31 36 30 30 30  36 34 32 34 37 38 31 39   s":16000 64247819
0100  31 31 30 37 31 37 39 39  35 30 36 33 33 38 37 36   11071799 50633876
0110  32 32 38 39 31 36 32 38  34 35 32 34 30 38 33 34   22891628 45240834
0120  37 34 39 33 30 34 30 36  39 31 30 39 34 34 37 36   74930406 91094476
0130  34 37 31 36 35 37 35 39  32 32 36 32 30 39 32 35   47165759 22620925
0140  31 31 30 34 35 38 38 31  31 34 35 39 36 34 30 37   11045881 14596407
0150  34 30 38 38 34 38 36 36  31 39 38 37 32 31 33 33   40884866 19872133
0160  31 37 37 34 30 38 30 38  31 37 33 34 30 36 31 30   17740808 17340610
0170  36 32 38 38 37 34 34 30  30 36 35 37 35 39 38 32   62887440 06575982
0180  36 31 32 31 32 39 36 39  39 37 32 35 36 7d 2c 7b   61212969 97256},{
0190  22 63 61 6e 64 69 64 61  74 65 22 3a 34 38 34 39   "candida te":4849
01a0  31 37 30 36 38 33 38 32  37 31 36 38 38 30 35 39   17068382 71688059
01b0  38 35 33 33 39 36 30 35  32 36 31 32 31 31 38 32   85339605 26121182
01c0  33 32 31 34 31 36 35 32  34 39 34 32 30 30 35 32   32141652 49420052
01d0  31 30 32 38 36 31 30 34  38 35 39 38 38 37 32 35   10286104 85988725
01e0  34 37 31 35 35 39 36 36  33 30 35 32 36 31 38 34   47155966 30526184
01f0  39 39 30 30 38 37 33 36  37 37 38 37 34 37 37 39   99008736 77874779
0200  39 35 33 37 39 35 35 37  32 32 30 34 30 35 35 32   95379557 22040552
0210  31 33 39 39 33 36 36 30  32 39 32 39 30 39 34 32   13993660 29290942
```

Gambar 4 Hasil penangkapan pengiriman rekapitulasi melalui wireshark

D.6 Pengujian *Use Case Get Recapitulation*

Rincian pengujian *Use Case Get Recapitulation* dapat dilihat pada Tabel 7.

Tabel 7 Pengujian *use case get recapitulation*

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-5-001	Pengujian penambahan rekapitulasi	1. Pada laman rekapitulasi di *voting place* , tekan tombol kirim. 2. Cek penambahan record pada database	Menekan tombol kirim pada laman rekapitulasi *voting place*	Record database bertambah	Record database bertambah	Diterima
RP-5-002	Pengujian penambahan rekapitulasi	1. Pada laman rekapitulasi di *voting place* , tekan tombol kirim. 2. Ubah data id pada rekapitulasi. Perubahan dilakukan pada kode program 3. Cek penambahan record pada database	Menekan tombol kirim pada laman rekapitulasi *voting place*	Record database tidak bertambah	Record database tidak bertambah	Diterima

D.7 Pengujian *Use Download Recapitulation*

R incian pengujian *Use Download Recapitulation* dapat dilihat pada Tabel 8.

Tabel 8 Pengujian *use case download recapitulation*

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
RP-6-001	Pengujian unduh dari *voting place*	1. Menghapus seluruh record yang tersimpan di database. Menghapus seluruh data yang disimpan pada tabel voting data.	id dan candidate *vote* yaitu id = 1205113101, candidate = 1, id = 1205113102, candidate = 1,	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil	Diterima

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
		2. Melakukan *vote* empat *vote* dengan id dan candidate *vote* yaitu id = 1205113101, candidate = 1, id = 1205113102, candidate = 1, id = 1205113103, candidate = 2, dan id = 1205113104, candidate = 3 3. Membuka http://localhost”8081 4. Mengunduh data rekapitulasi kemudian mendekripsinya	id = 1205113103, candidate = 2, dan id = 1205113104, candidate = 3 Menekan tombol download pada laman rekapitulasi	dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	
RP-6-002	Pengujian unduh pada intermediate level	1. Mengirimkan rekapitulasi pada *voting place* ke intermediate level 2. Membuka http://localhost:8082 3. Mengunduh data kemudain mendekripsinya	Menekan tombol kirim Mebuka laman rekapitulasi intermediate level kemudain menekan tombol download	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 1 dan kandidat 3 jumlah *vote* 1	Diterima
RP-6-003	Pengujian unduh setelah penambahan *vote* pada *voting place*	1. Melakukan *vote* dengan id = 1205113105 dan kandidat 2 2. Membuka laman rekapitulasi http://localhost:8081 3. Mengunduh data kemudain mendekripsinya	Mengetikkan id = 1205113102 dan memilih radio button kandidat kedua Menekan tombol download pada laman rekapitulasi	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut	Diterima

Id	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Hasil yang didapatkan	Kesimpulan
				masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 2 dan kandidat 3 jumlah *vote* 1	masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 2 dan kandidat 3 jumlah *vote* 1	
RP-6-004	Pengujian unduh setelah penambahan *vote* pada intermediate level	1. Mengirimkan rekapitulasi pada *voting place* ke intermediate level 2. Membuka http://localhost:8082 3. Mengunduh data kemudain mendekripsinya	Menekan tombol kirim Mebuka laman rekapitulasi intermediate level kemudain menekan tombol download	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 2 dan kandidat 3 jumlah *vote* 1	File teks hasil rekapitulasi dengan format terenkripsi terunduh ke komputer. Hasil dekripsi menghasilkan kandidat dan jumalah *vote* berturut-turut masing-masing kandidat 1 jumlah *vote* 2, kandidat 2 jumlah *vote* 2 dan kandidat 3 jumlah *vote* 1	Diterima

Lampiran E. Contoh Data Pengujian

Data *vote* yang digunakan pada pengujian dapat dilihat pada Tabel 9. *Ciphertext* dan *plaintext* dalam tabel memiliki format <id pemilih, kandidat yang dipilih>.

Tabel 9 Data *vote*

NO	Cipher text	Plain text
1	121003425754001322352588760581414290734907807054723315485070263964545699123665965122586925609022139345827047720712783388741942284201552171751099028425098814908242227289025654475639834353562839908178124675617759161428821605604260537455785983087957814914993223634653042205466902147993385701267426173199278592551205113101,1
2	447705105220255696790921380706496724335724158564874953654540938259549605748731870485927914789182635995794210620946257987081495377739100088524207536290027014908242227289025654475639834353562839908178124675617759161428821605604260537455785983087957814914993223634653042205466902147993385701267426173199278592551205113102,1
3	352277109872262262111701889382018697240093662751897672191187914614010242814196080852588331470538107064128668650523781909838695503619908841995438553371799148491706838271688059853396052612118232141652494200521028610485988725471559663052618499008736778747799537955722040552139936602929094233341798934037908857531205113103,2
4	201485238111394357379780230339152045932158198174415823002784871909871350324784426489555428131856892875580867597780187910505375952267414659442516996091512161393870666071186292715871909276014475888212770049844174168310897518270798752360320996295378395635129822892211836468149356826430812478502697335357336901	1205113104,3
5	323123938404031267504363127423394376806540136380087600461064124322747828329405113638682790192553344591396330348115642504753547727220424916588566306118185 9, 484917068382716880598533960526121182321416524942005210286104859887254715596630526184990087367787477995379557220405521399366029290942333417989340379088575 3	1205113105,2

NO	Cipher text	Plain text
6	327287938425430254305043183755956264333073959828669813222633751705811764036354037979122185408759682955013989682286881653804356988977376271604294056473375 2, 149082422272890256544756398343535628399081781246756177591614288216056042605374557859830879578149149932236346530422054669021479933857012674261731992785925 5	1205113105,1

Data rekapitulasi dapat dilihat pada Tabel 9. *Ciphertext* dan *plaintext* dalam tabel memiliki format <kandidat, jumlah suara>

Tabel 10 Contoh data rekapitulasi

No	Cipher text	Plain Text
1	149082422272890256544756398343535628399081781246756177591614288216056042605374557859830879578149149932236346530422054669021479933857012674261731992785925 5, 439120356652838028607291125689358027982971358489202628383488119213705059924744660978002129649186008865069505302103507496842425426161029125987821612774484 0	1,2
2	484917068382716880598533960526121182321416524942005210286104859887254715596630526184990087367787477995379557220405521399366029290942333417989340379088575 3, 497790300492069275682638547125537982342342995232205746966872252122196409299131949414449221311936767495379515295121651982664718564933426863767280099160150 0	2,1
3	613938706660711862927158719092760144758882127700498441741683108975182707987523603209962953783956351298228922118364681493568264308124785026973353573369 01, 497790300492069275682638547125537982342342995232205746966872252122196409299131949414449221311936767495379515295121651982664718564933426863767280099160150 0	3,1