BAB III
METODE PENELITIAN

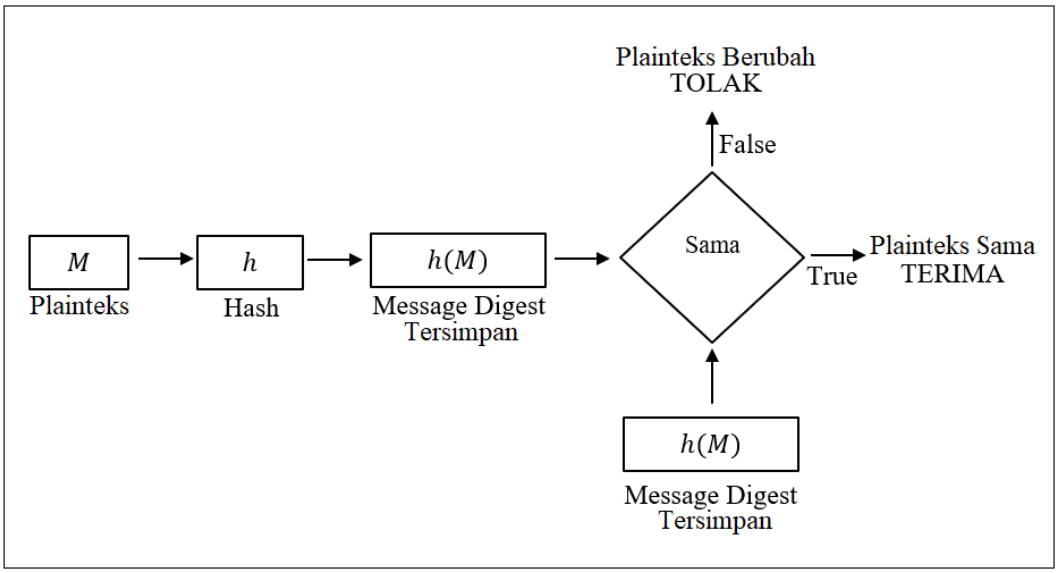
Metode yang digunakan dalam penulisan ini adalah studi literatur dan pengembangan model kriptografi, serta pengaplikasiannya pada program aplikasi yang akan diuraikan secara rinci dalam langkah-langkah berikut:

3.1 Identifikasi Masalah

Banyak terjadi kasus pemalsuan sebuah dokumen yang bersifat digital, contohnya e-sertifikat. Pada e-sertifikat, biasanya terdapat tanda tangan digital yang fungsinya untuk menjaga keaslian bahwa dokumen tersebut bersifat asli. Faktanya, tanda tangan digital tersebut tidak bisa menjamin bahwa dokumen tersebut asli. Di era yang modern ini, orang sudah dapat dengan mudah memanipulasi, mengubah, atau mengedit dokumen digital tersebut dengan bantuan *software*. Oleh karena itu, diperlukan proses autentikasi dengan fungsi *hash* SHA-256 dan kriptografi kunci publik *ElGamal Signature Scheme* untuk memeriksa keautentikan dari sebuah dokumen digital (Munir, 2017).

3.2 Model Dasar

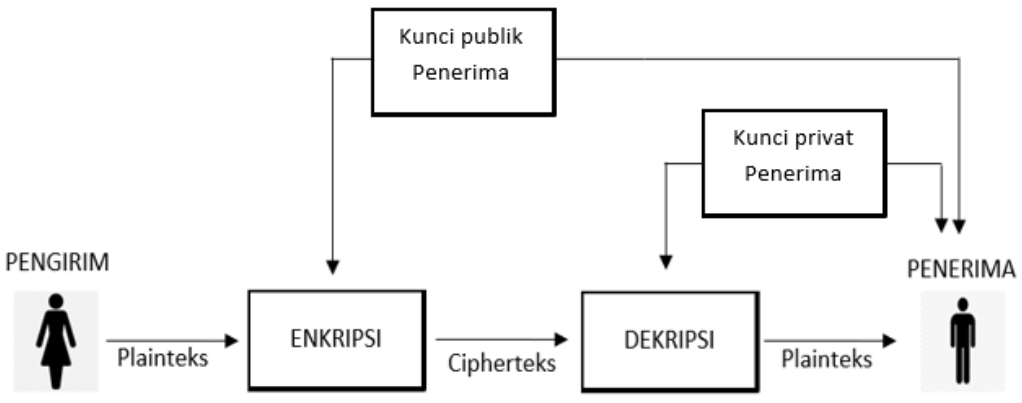
Model dasar yang digunakan dalam pada penlelitian ini adalah fungsi *hash* dan kriptografi kunci publik ElGamal. Fungsi *hash* adalah fungsi yang menerima masukan string yang panjangnya sembarang selanjutnya mentransformasikannya menjadi string keluaran yang panjangnya tetap (Munir, 2006). Menurut Kaufman et.al. (2002), fungsi hash dapat digunakan sebagai pengamanan *password*, *message integrity*, dan *message fingerprint*. Cara kerja fungsi *hash* sebagai *message integrity* akan dijelaskan pada skema berikut:



Gambar 3.1 Skema fungsi *hash* sebagai *message integrity*

Berdasarkan skema pada Gambar 3.2, plainteks (M) dilakukan fungsi *hash* menghasilkan nilai *hash* ($h(m)$) atau *message digest*, kemudian *message digest* dicocokkan. Jika *message digest* cocok maka plainteks diterima, dan jika *message digest* tidak cocok maka plainteks ditolak.

Algoritma ElGamal merupakan salah satu kriptografi kunci publik di mana untuk melakukan proses enkripsi dan dekripsi menggunakan dua kunci yang berbeda. Proses enkripsi dan dekripsi pada algoritma ElGamal akan dijelaskan pada skema berikut:

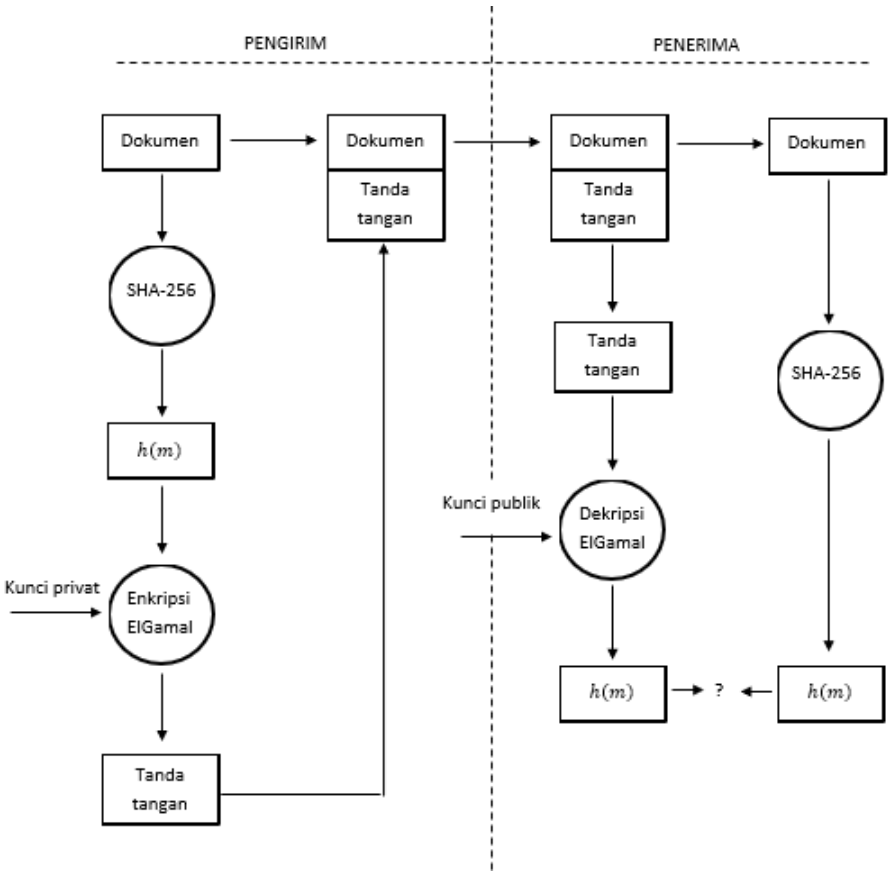


Gambar 3.2 Skema enkripsi dan dekripsi ElGamal

3.3 Pengembangan Model Dasar

Berdasarkan model dasar yang telah diuraikan, pengembangan model yang akan dilakukan pada penulisan ini adalah menggabungkan kedua algoritma yang telah diuraikan pada model dasar, yaitu fungsi *hash* dan algoritma kriptografi kunci publik ElGamal untuk autentikasi dokumen digital, di mana fungsi *hash* yang digunakan pada penulisan ini adalah SHA-256 yang menghasilkan string keluaran berupa nilai *hash* sebesar 256 bit.

Dalam pengembangannya, algoritma ElGamal tidak hanya berfungsi sebagai enkripsi dan dekripsi, melainkan dapat digunakan untuk tanda tangan digital yang disebut sebagai *ElGamal Signature Scheme*. Berikut skema autentikasi dokumen digital dengan SHA-256 dan *ElGamal Signature Scheme*:



Gambar 3.3 Skema pengembangan model dasar

Berdasarkan skema pada gambar 3.3, langkah pertama yang dilakukan oleh pengirim adalah melakukan *hashing* dengan fungsi *hash* SHA-256 pada dokumen lalu mengenkripsinya dengan *ElGamal Signature Scheme* dan menggunakan kunci publik pengirim untuk mendapatkan tanda tangan digital pada dokumen. Setelah diperoleh tanda tangan digital, tanda tangan ditempelkan ke dokumen dan dokumen dengan tanda tangan digital tersebut sudah dapat dikirimkan ke penerima dokumen. Sedangkan di pihak penerima dokumen, untuk melakukan autentikasi dokumen, dokumen dipisahkan dengan tanda tangan digital kemudian tanda tangan digital tersebut didekripsi dengan *ElGamal Signature Scheme* menghasilkan nilai *hash*, sedangkan dokumen dilakukan *hashing* dengan fungsi *hash* SHA-256 menghasilkan nilai *hash*. Langkah terakhir adalah dengan menyamakan nilai *hash* hasil dekripsi *ElGamal Signature Scheme* dan nilai *hash* hasil dari fungsi *hash* SHA-256. Jika sama maka dokumen dengan tanda tangan digital valid, berlaku sebaliknya, jika tidak sama maka dokumen dengan tanda tangan digital tidak valid.

3.4 Konstruksi Program Aplikasi

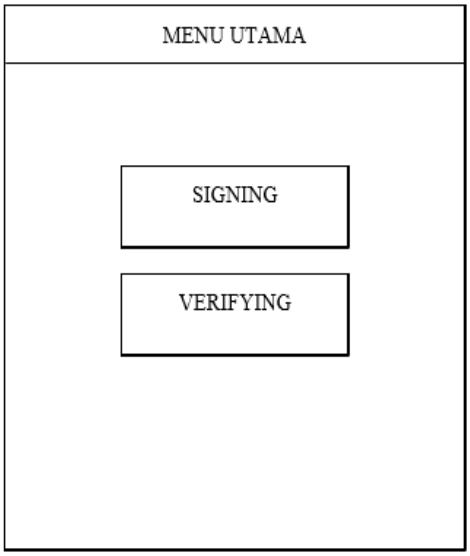
Program autentikasi digital dengan SHA-256 dan *ElGamal Signature Scheme* akan dikonstruksi dengan menggunakan aplikasi Python GUI (*Graphical User Interface*) versi 3.8 dengan bantuan *software* Notepad dan Visual Studio Code.

3.4.1 Input dan Output Program Aplikasi

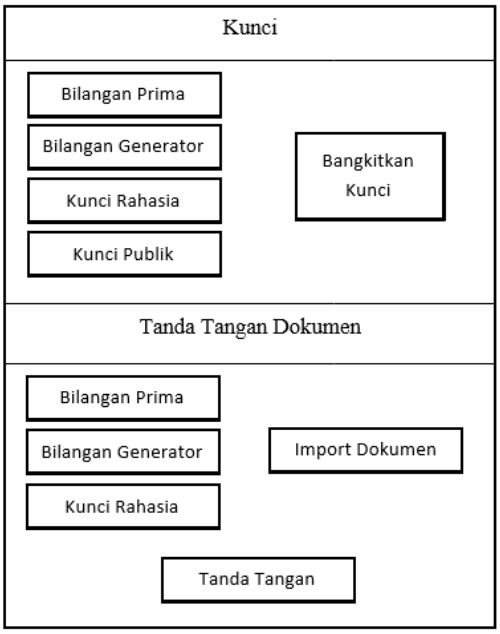
Program aplikasi terdiri dari 2 menu utama, yaitu menu *signing* dan *verifying*. Menu *signing* digunakan oleh pengirim dalam membangkitkan kunci dan tanda tangan digital dengan input berupa dokumen, bilangan prima, bilangan generator, dan kunci privat, serta tanda tangan sebagai outputnya, sedangkan menu *verifying* digunakan oleh penerima untuk memverifikasi apakah dokumen yang diterima autentik dan berasal dari orang yang sebenarnya atau tidak. Dalam prosesnya, dibutuhkan input berupa bilangan prima, bilangan generator, kunci publik, dokumen dan tanda tangan, dengan outputnya berupa label “Dokumen dengan tanda tangan cocok” atau “Dokumen dengan tanda tangan tidak cocok”

3.4.2 Rancangan Tampilan Program Aplikasi

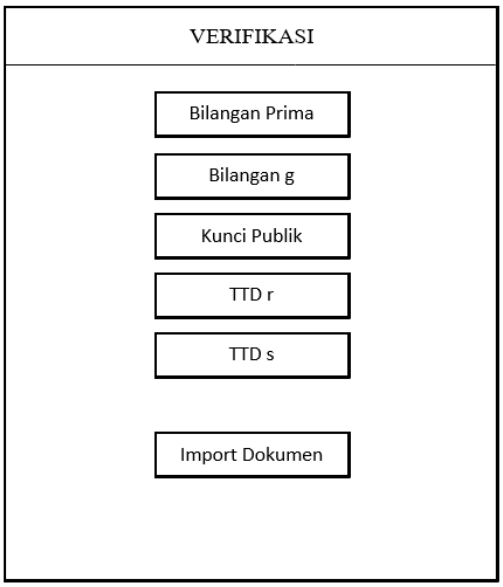
Rancangan awal tampilan program aplikasi yang akan dibuat adalah sebagai berikut:



Gambar 3.4 Rancangan tampilan menu utama program aplikasi



Gambar 3.5 Rancangan tampilan menu *signing* program aplikasi



Gambar 3.6 Rancangan tampilan menu *verifying* program aplikasi

1.4.3 Algoritma Autentikasi Dokumen Digital dengan SHA-256 dan *ElGamal Signature Scheme*

Terdapat tiga algoritma utama pada autentikasi dokumen digital dengan SHA-256 dan *ElGamal Signature Scheme*, yaitu Pembangkitan kunci, pembuatan tanda tangan digital, dan autentikasi dokumen digital. Adapun algoritma tersebut dijelaskan sebagai berikut:

1) Pembangkitan Kunci

   Pembangkitan kunci dilakukan oleh pengirim dokumen, kunci yang dibangkitkan berupa kunci publik dan kunci privat yang merupakan parameter-parameter ElGamal digunakan untuk kebutuhan autentikasi dokumen digital. Kunci privat digunakan untuk pembuatan tanda tangan digital, sedangkan kunci publik digunakan untuk autentikasi dokumen digital.

2) Pembuatan Tanda Tangan Digital

   Hal penting yang diperlukan untuk membuat tanda tangan digital pada dokumen adalah melakukan *hashing* dengan fungsi *hash* SHA-256 yang bertujuan meringkas dokumen menjadi pesan yang berukuran sebesar 256-bit. Kemudian dokumen dapat ditandatangani dengan dengan

kunci privat pengirim, menghasilkan dua buah bilangan sebagai tanda tangan digital pada dokumen tersebut.

3) Autentikasi Dokumen Digital

Penerima dokumen dapat melakukan autentikasi dengan memverifikasi tanda tangan digital dari dokumen tersebut. Proses verifikasi dilakukan dengan mencocokkan hasil dari dua proses perhitungan dengan input berupa kunci publik pengirim dan tanda tangan digital dari dokumen.

3.5 Validasi

Pada tahap ini dilakukan validasi atau pengecekan terhadap program, apakah program berjalan dengan baik atau tidak. Misalkan diberikan pesan yang berupa dokumen digital m, lalu bangkitkan tanda tangan digitalnya berupa r, s yang kemudian ditempelkan ke dokumen tersebut (m, r, s). Pada proses autentikasi, jika diinputkan (m, r, s) yang telah dibangkitkan pada proses pembangkitan tanda tangan mengeluarkan output "Dokumen dengan tanda tangan cocok", dan jika diinputkan (m', r, s) atau (m, r', s'), dengan kata lain telah terjadi perubahan terhadap dokumen atau tanda tangannya mengeluarkan output "Dokumen dengan tanda tangan tidak cocok", maka dipastikan program berjalan dengan baik.