BAB III
METODOLOGI PENELITIAN

3.1 Identifikasi Masalah

Dokumen digital cenderung memiliki karakteristik terbuka atau dapat diakses oleh pihak yang memiliki izin, yang membuatnya rentan terhadap *data tampering* atau modifikasi yang tidak sah selama proses transfer atau penyimpanan. *Data tampering* atau modifikasi yang tidak sah terhadap data akan menyebabkan kekhawatiran akan keaslian dokumen, kurangnya kepercayaan, risiko keamanan, dan masih banyak lagi. Oleh karena itu, untuk melindungi dokumen digital berformat .pdf yang disimpan atau dibagikan melalui layanan *cloud* atau awan dari berbagai jenis serangan siber yang dapat merusak integritas dokumen, diperlukan tindakan perlindungan, seperti penggunaan tanda tangan digital atau *digital signature*.

Tanda tangan digital atau *digital signature* merupakan sebuah metode untuk mengecek keautentikan atau keaslian sebuah dokumen digital berformat .pdf. Dengan mengecek *signature* yang tertanam dalam dokumen digital kita dapat memverifikasi apakah telah terjadi perubahan terhadap dokumen digital atau tanda tangan yang tertanam pada dokumen tersebut.

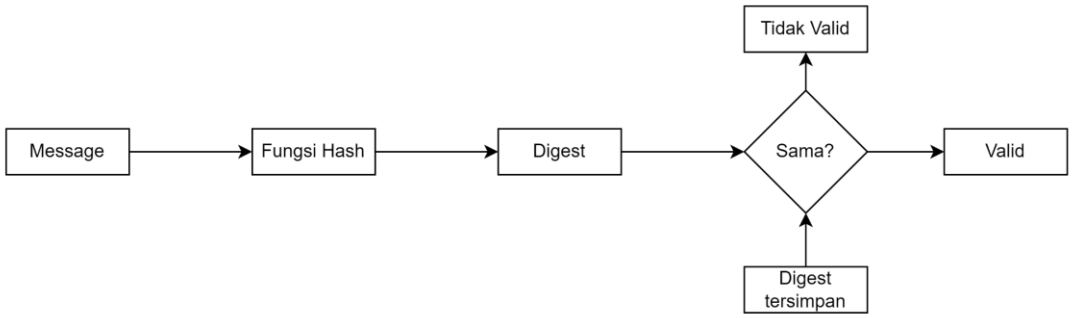
3.2 Model dasar

Model dasar yang digunakan pada penelitian ini adalah fungsi *hash* dan algoritma penandatanganan RSA. Algoritma tanda tangan digital RSA adalah algoritma dengan kunci asimetris yang terdapat kunci privat yang digunakan dalam proses pembuatan tanda tangan dan kunci publik untuk verifikasi tanda tangan. Dalam proses penandatanganan dan verifikasi tanda tangan RSA menggunakan nilai hash yang diperoleh dari fungsi hash BLAKE2. Dalam penelitian ini, input yang digunakan adalah *file* dengan format pdf yang merupakan format umum untuk menyimpan data sensitif dan merupakan salah satu format paling umum dari sebuah *file* yang biasa disebarluaskan.

3.2.1 Fungsi Hash BLAKE2

Fungsi *hash* digunakan untuk membuat "*fingerprint*" singkat dari beberapa data; jika data tersebut diubah, maka sidik jari tersebut tidak akan lagi valid (Stinson dan Paterson, 2018). Misalkan sidik jari tersebut disimpan di tempat yang aman

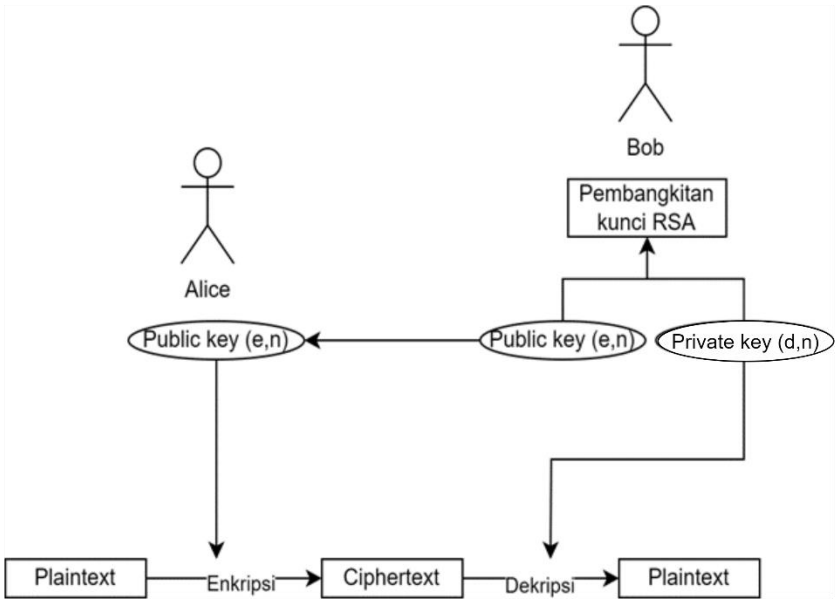
maka, meskipun data disimpan di tempat yang tidak aman, integritasnya dapat diperiksa dari waktu ke waktu dengan menghitung ulang sidik jari dan memverifikasi bahwa sidik jari tersebut tidak berubah. Dalam penelitian ini *input message* yang digunakan adalah *file* dengan format pdf yang selanjutnya akan di-*hashing* menggunakan fungsi *hash* BLAKE2 untuk mendapatkan *message digest*. Berikut adalah gambaran cara kerja fungsi *hash* sebagai tanda tangan digital:



Gambar 3.1 Skema Fungsi Hash Sebagai Tanda Tangan

3.2.2 Kriptografi RSA

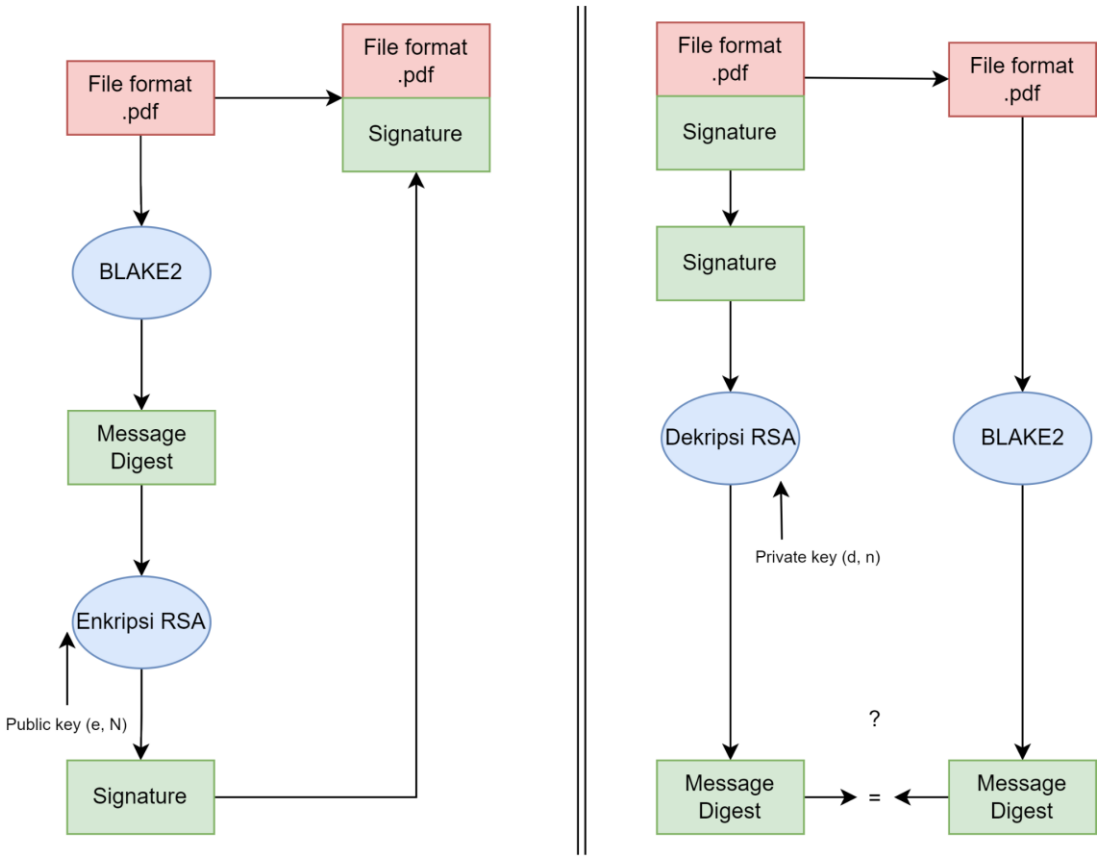
Algoritma tanda tangan digital RSA bekerja dengan menggunakan sepasang kunci, yaitu kunci publik dan kunci privat. Proses tanda tangan digital RSA melibatkan penggunaan kunci privat untuk menghasilkan tanda tangan digital, yang nantinya dapat diverifikasi oleh pihak lain menggunakan kunci publik yang sesuai.



Gambar 3.2 Skema Algoritma Kriptografi RSA

3.3 Pengembangan Model

Berdasarkan model dasar yang telah diuraikan sebelumnya, pengembangan model yang akan dilakukan pada penelitian ini adalah dengan mengimplementasikan fungsi *hash* BLAKE2 pada skema penandatangan RSA yang akan menghasilkan *hash value* sebesar 512-bit pada *environment* atau lingkungan *cloud* atau awan. Pengembangan model tersebut digambarkan pada Gambar 3.3



Gambar 3.3 Skema Pengembangan Tanda Tangan Digital

Pada Gambar 3.3 di atas digambarkan bagaimana program aplikasi yang akan dibuat nanti bekerja, berawal dari *user* yang akan melakukan sebuah input *file* dengan format .pdf selanjutnya akan dilakukan proses *hashing* menggunakan fungsi *hash* BLAKE2 yang menghasilkan *message digest* yang akan dienkripsi dengan algoritma enkripsi RSA dan menghasilkan *signature* untuk dibubuhkan pada *file* berformat .pdf yang diinput oleh *user*.

Selanjutnya proses validasi integritas file dilakukan dengan *user* melakukan input file berformat .pdf yang akan didekripsi oleh program aplikasi dan disaat bersamaan

dilakukan hashing terhadap *file* tersebut untuk mendapatkan *message digest* yang kemudian akan disamakan antara *message digest* hasil dekripsi dengan *message digest* hasil *hashing*. Jika nilainya terbukti sama berarti integritas dari *file* tersebut telah tervalidasi benar dan berlaku sebaliknya.

3.4 Konstruksi Aplikasi Tanda Tangan Digital.

3.4.1 Input dan Output Aplikasi Tanda Tangan Digital

Aplikasi Tanda Tangan Digital yang dibuat akan menerima input berupa dokumen digital berformat .pdf yang diunggah oleh pengguna lalu dokumen tersebut akan melalui proses penandatangan dengan fungsi *hash* BLAKE2 serta RSA yang akan menghasilkan keluaran berupa dokumen digital berformat .pdf yang sudah tertanam tanda tangan

3.4.2 Algoritma Deskriptif

Terdapat tiga algoritma utama dari skema pengembangan model dasar pada Gambar 3.3, yaitu *hashing*, pembangkitan kunci publik dan kunci privat dari kriptografi RSA, dan enkripsi dan dekripsi kriptografi RSA.

- **Hashing**
  Langkah-langkah untuk melakukan hash pada program aplikasi akan dijelaskan sebagai berikut:
  1. User mengunggah file berformat .pdf pada program aplikasi
  2. File yang telah diunggah akan dilakukan proses *hashing*
  3. Setelah pemrosesan selesai maka akan didapatkan keluaran berupa *message digest* yang nantinya menjadi tanda tangan digital untuk dienkripsi.

- **Pembangkitan kunci RSA**
  Proses pembangkitan kunci menggunakan algoritma RSA adalah sebagai berikut:
  - Pilih 2 bilangan prima p dan q
  - Hitung $n = p \cdot q$ di mana $p \neq q$
  - Hitung $\Phi(n) = (p - 1) \cdot (q - 1)$
  - Pilih kunci publik e yang relatif prima terhadap $\Phi(n)$
  - Bangkitkan kunci privat dengan menggunakan $e.d \equiv 1 \, mod \, \Phi(n)$, perhatikan bahwa persamaan tersebut ekuivalen dengan $e \cdot d = k \cdot$

$\Phi(n) + 1$ sehingga $d = k \cdot \Phi(n) + 1$ e dengan k suatu bilangan bulat terkecil yang memberikan hasil bilangan bulat d

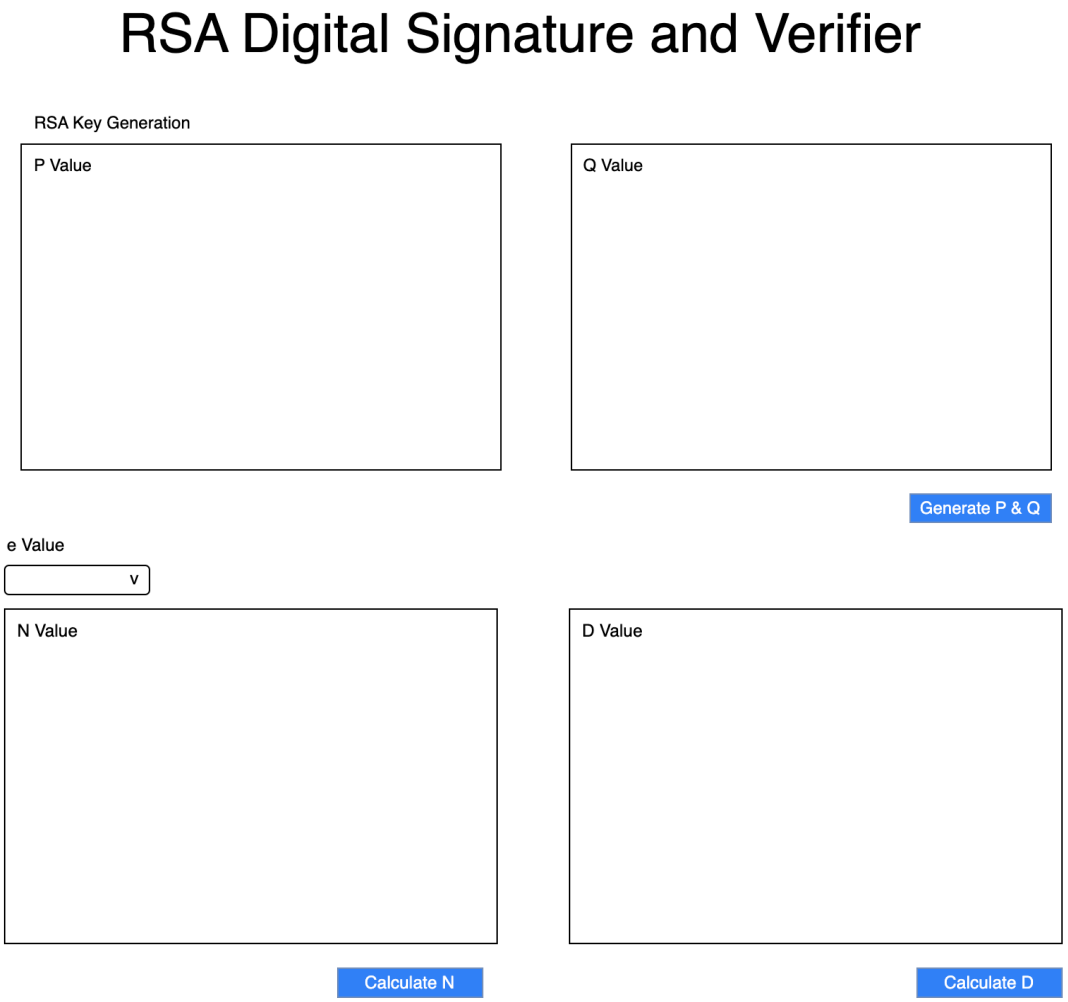
- **Proses Penandatanganan Digital**
  Setelah proses *hashing* dan pembangkitan kunci dilakukan maka akan dilanjutkan dengan proses enkripsi dengan perhitungan yang telah dijelaskan pada BAB II sub bab 2.4. Enkripsi dilakukan untuk mendapatkan cipherteks yang akan dibubuhkan pada *file* yang diunggah oleh *user*, langkah-langkahnya sebagai berikut:

Tabel 3.1 Enkripsi dan Dekripsi RSA

Enkripsi	Dekripsi
1. *User* mendapatkan nilai *hash* dari *file* .pdf yang diinput serta kunci RSA 2. User menginput nilai *hash* dan kunci publik pada program aplikasi yang selanjutnya akan dienkripsi 3. User mendapatkan cipherteks dari enkripsi nilai *hash* yang merupakan tanda tangan digital yang telah dienkripsi	Proses dekripsi pada skema penandatangan RSA dengan fungsi *hash* BLAKE2 dilakukan untuk mendapatkan nilai tanda tangan digital yang telah dienkripsi. Hasil dekripsi tersebut akan digunakan untuk memvalidasi integritas dari *file*, langkah-langkahnya adalah sebagai berikut: 1. User mengunggah ulang *file* yang diterima 2. User yang telah menerima kunci privat dalam tahap pembangkitan kunci sebelumnya dapat mendekripsi *file* tersebut dengan kunci privat RSA 3. Setelah proses pen-dekripsian selesai program aplikasi akan melakukan validasi terhadap integritas dari *file* tersebut

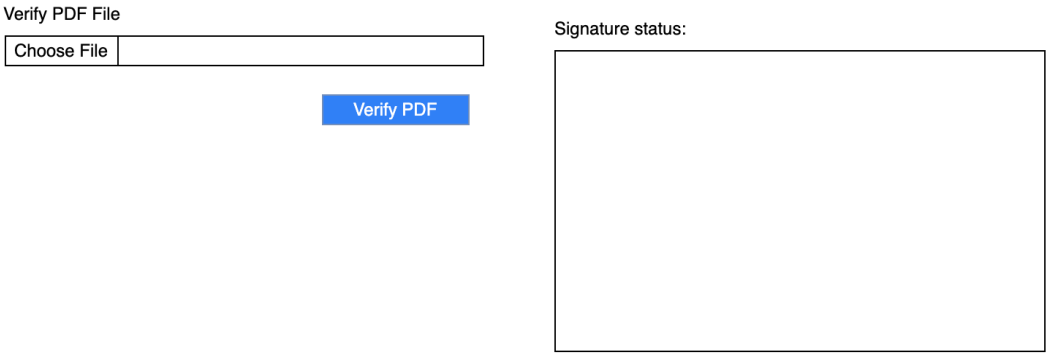
3.4.3 Rancangan Tampilan Aplikasi Tanda Tangan Digital

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



Gambar 3.4 Tampilan Fitur RSA *Key Generation*

Gambar 3.5 Tampilan Fitur Tanda Tangan PDF



Gambar 3.6 Tampilan Fitur Verifikasi Tanda Tangan PDF

3.4.4 Implementasi *Tools* Pemrograman

Library Python

Dalam pembuatan aplikasi Tanda Tangan Digital, bahasa pemrograman python akan digunakan untuk membangun fitur utama dari aplikasi Tanda Tangan Digital *library* untuk menunjang pembuatan program aplikasi. Berikut adalah *library* yang akan digunakan

1. Hashlib

   Modul hashlib akan digunakan dalam konstruksi program aplikasi sebagai modul penyedia fungsi *hash* BLAKE2. Fungsi *hash* BLAKE2 akan digunakan dalam pembuatan tanda tangan digital dengan input file format .pdf.

2. Flask

   Modul Flask akan digunakan untuk membantu pembuatan *webpage* dari Aplikasi Tanda Tangan Digital

Penggunaan HTML

Dalam pembuatan aplikasi Tanda Tangan Digital, HTML akan digunakan sebagai bahasa untuk membuat antarmuka pengguna dari Aplikasi Tanda Tangan Digital dan mengintegrasikan hasil dari implementasi kriptografi tanda tangan digital ke dalam antarmuka pengguna.

Implementasi pada lingkungan *cloud*

Setelah aplikasi Tanda Tangan Digital telah dibuat, peneliti memilih untuk menjalankannya dalam lingkungan *cloud*. Lingkungan cloud dipilih karena biaya

yang lebih rendah, fleksibilitas yang lebih baik, dan merupakan lingkungan yang optimal untuk fungsi *hash* BLAKE2 yang digunakan dalam aplikasi.

3.5 Proses Validasi

Pada tahap ini dilakukan proses validasi dengan melakukan pengecekan terhadap keluaran dari aplikasi Tanda Tangan Digital yang telah dibuat untuk melihat apakah aplikasi berjalan sesuai fungsinya atau tidak. Validasi dilakukan dengan dua buah jenis kasus, yaitu:

1. Verifikasi dokumen digital yang autentik dengan terdapatnya kesesuaian tanda tangan digital yang diunggah
2. Verifikasi dokumen digital yang tidak autentik dan terdapat perubahan pada tanda tangan digital yang tertanam

Apabila pada keluaran dari aplikasi yang telah dibuat sesuai maka dapat disimpulkan aplikasi Tanda Tangan Digital berjalan dengan baik.

3.6 Penarikan Kesimpulan

Setelah kesesuaian dari keluaran aplikasi Tanda Tangan Digital telah tervalidasi benar dan program sudah dipastikan berjalan dengan benar, maka algoritma pengembangan antara fungsi *hash* BLAKE2 dengan skema penandatanganan RSA algoritma ini dapat digunakan dalam implementasi tanda tangan digital pada *cloud*. Dengan kombinasi antara kedua algoritma ini, diharapkan dapat meningkatkan keamanan serta integritas data yang disimpan dan ditransfer di lingkungan cloud, serta memberikan tingkat verifikasi yang tinggi terhadap keaslian informasi yang disampaikan melalui tanda tangan digital.