BAB III
METODE PENELITIAN

Secara umum, dalam penelitian ini dikembangkan suatu protokol dan skema autentikasi *user* dengan memanfaatkan kriptografi kunci publik. Kemudian dikonstruksi sebuah program aplikasi yang merupakan penerapan dari protokol dan skema yang telah diracang. Penelitian dilakukan menggunakan studi literatur, dengan tahapan-tahapan sebagai berikut:

3.1 Identifikasi Masalah

Dalam rangka perlindungan terhadap informasi yang terdapat pada suatu akun, maka diterapkan autentikasi bagi pihak yang ingin mengaksesnya sesuai dengan protokol tertentu. Protokol autentikasi yang sudah diterapkan secara umum adalah dengan menggunakan *username* dan *password*. Namun hal ini memiliki kendala terkait dengan kebocoran *password* sehingga membuat pihak asing dapat mengakses informasi yang bukan miliknya.

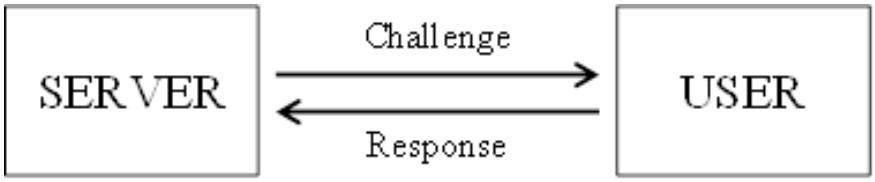
Dalam penelitian ini, dirancang sebuah protokol autentikasi *user* saat *log-in* dengan tujuan agar tidak ada pihak asing yang dapat mengakses akun sekalipun *password* yang digunakan bocor ke publik. Protokol autentikasi tersebut adalah dengan menggunakan *Two Way Challenge Response Protocol*. Adapun pemberian *challenge* dan *response* menggunakan Algoritma *Elliptic Curve Diffie Hellman*, yang sering dipakai sebagai algoritma pertukaran kunci simetri. Dengan algortima *Elliptic Curve Diffie-Hellman*, *server* akan membangkitkan bilangan acak setiap *user* melakukan *log-in*. Hal ini membuat *challenge* yang diberikan pada *user* akan berbeda-beda tiap waktunya, sehingga *response* dari *user* pun akan berbeda.

3.2 Model Dasar

3.2.1 Two Way Challenge-Response Protocol

Protokol autentikasi dengan *Two Way Challenge-Response Protocol* ini akan melibatkan dua entitas, yaitu *server* dan *user*. *Server* akan menjadi pemberi *challenge*, dan *user* yang diautentikasi akan memberikan *response*. Dengan cara ini, *server* memastikan apakah *user* yang sedang melakukan *log-in* adalah asli

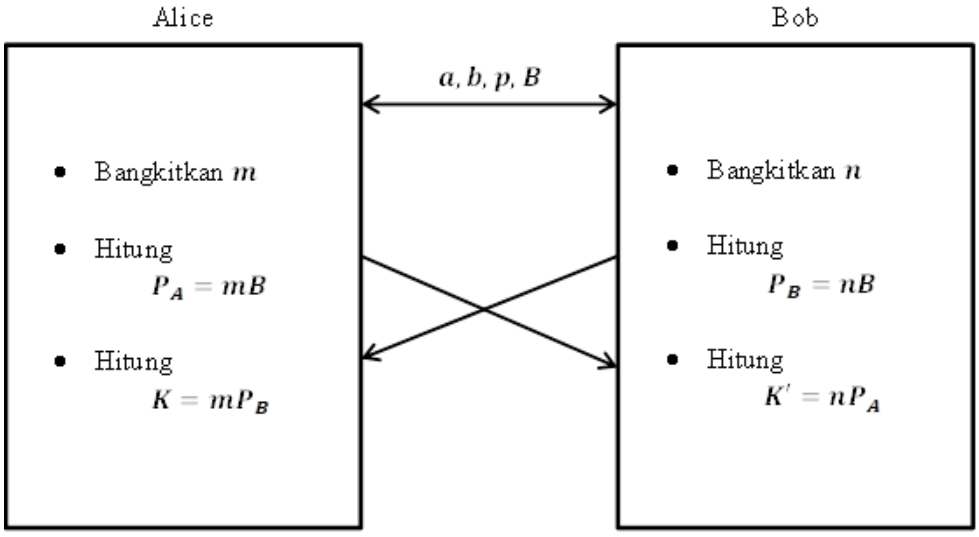
atau tidak. Berikut merupakan skema dari *Two Way Challenge Response Protocol*:



Gambar 3. 1 Skema *Two Way Challenge-Response Protocol*

3.2.2 Algoritma Elliptic Curve Diffie Hellman

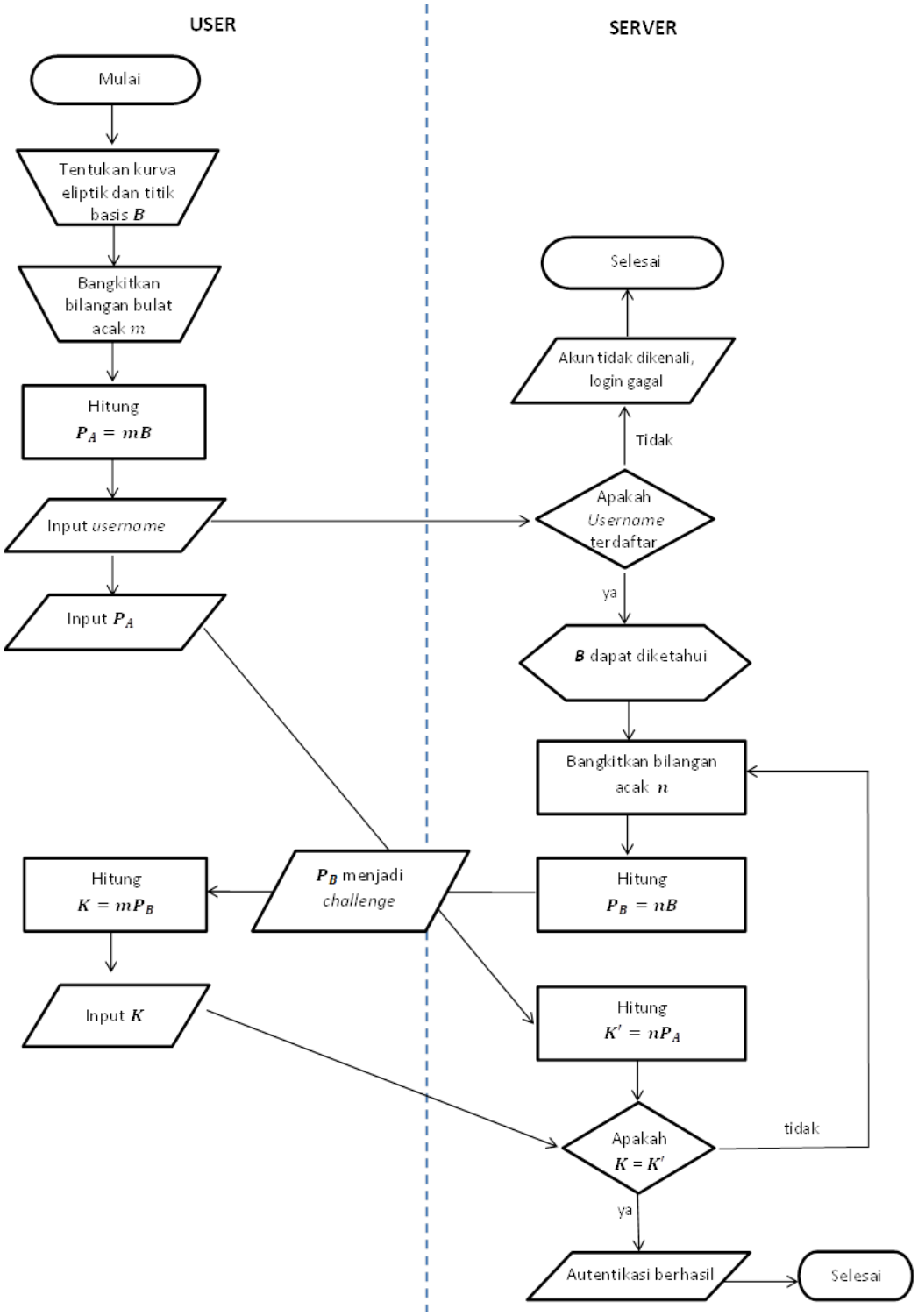
Adapun Algoritma *Elliptic Curve Diffie Hellaman* digunakan dalam kriptografi kunci publik sebagai protokol untuk pertukaran kunci simteri. Misal terdapat dua pihak yang terlibat dalam algoritma ini, yaitu Alice dan Bob. Alice dan Bob terlebih dahulu menyepakati kunci publik yang digunakan, yaitu parameter bilangan bulat a, b dan bilangan prima p pada persamaan $y^2 \equiv x^3 + ax + b \ (mod\ p)$, dan sebuah titik basis $B(x, y)$ yang dipilih dari kurva eliptik tersebut. Kemudian, keduanya membangkitan masing-masing sebuah bilangan acak sebagai kunci *private* untuk mengolah kunci publik yang telah disepakati. Hasil perhitungan ini dipertukarkan dan diolah kembali dengan kunci *private* masing-masing. Jika prosesnya benar, maka di akhir proses Alice dan Bob akan memiiki kunci simetri yang sama. Berikut ini adalah skema pertukaran kunci dengan Algoritma Elliptic Curve Diffie Hellman:



Gambar 3. 2 Skema *Elliptic Curve Diffie Hellman*

3.3 Pengembangan Model

Pengembangan model yang digunakan adalah dengan menggunakan Algoritma *Elliptic Curve Diffie Hellman* sebagai protokol pemberian *challenge* dan *response*. Berikut merupakan skema untuk protokol ini:



Gambar 3. 3 Skema Autentikasi dengan *Elliptic Curve Diffie Hellman*

3.4 Konstruksi Program

Program yang akan dibuat terdiri atas dua bagian, yaitu program utama dan alat hitung yang dikhususkan untuk *user*. Program utama memiliki tampilan awal yang terdiri atas menu *sign-in* (pembuatan akun) dan *log-in*. Sedangkan alat hitung merupakan suatu program yang digunakan untuk mengkalkulasi titik-titik yang akan diinputkan oleh *user*. Alat hitung terpisah dari program utama dan hanya dapat digunakan oleh *user*. Untuk seterusnya, alat hitung ini akan disebut sebagai kalkulator *user*.

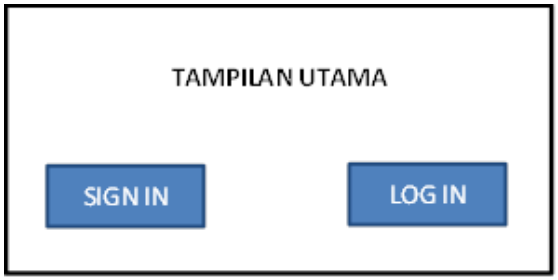
3.4.1 Input Output

Program yang akan dikonstruksi memiliki dua fungsi utama, yaitu untuk *sign-in* dan *log-in*. Pada menu *sign-in*, *user* melakukan pembuatan akun baru, yang mana *input* dari menu ini adalah *username*, parameter bilangan bulat a, b dan bilangan prima p, serta titik basis $B\ (x, y)$. Untuk penentuan titik basis, daftar titik yang bisa dipilih akan muncul setelah *user* menentukan bilangan prima p yang akan digunakan. *Output* dari menu ini adalah pernyataan konfirmasi bahwa akun berhasil dibuat.

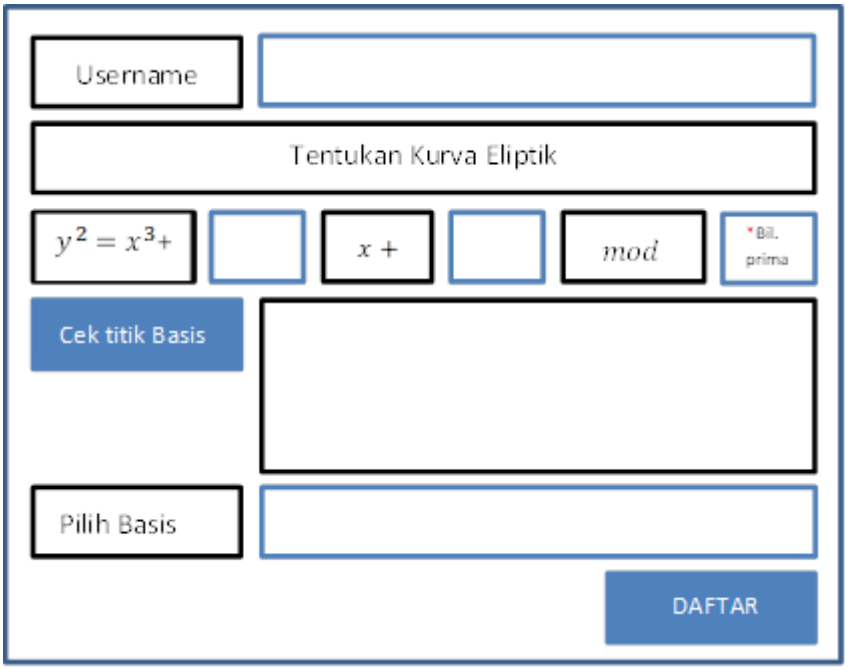
Untuk menu *log-in*, *user* akan menginput *username* dan P_A yang merupakan hasil perhitungan dari titik basis dengan bilangan acak tertentu. Setelah itu, *server* akan menampilkan P_B sebagai *challenge* dan *user* kembali menginput K sebagai *response*. *Output* dari menu ini adalah *user* terverifikasi dan dapat mengakses akunnya.

Selain itu, untuk kalkulator *user*, yang menjadi *input* adalah bilangan acak dan sebuah titik yang akan dikalkulasi. *Output* dari kalkulator *user* adalah sebuah titik P_A yang akan digunakan saat awal *log-in* atau sebagai *response* atas *challenge* dari *server*.

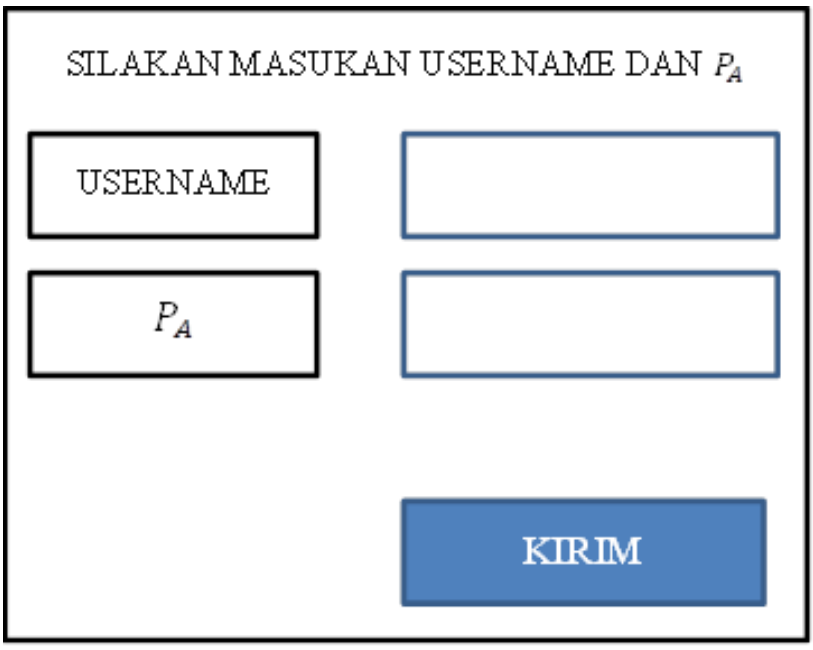
3.4.2 Rancangan Tampilan



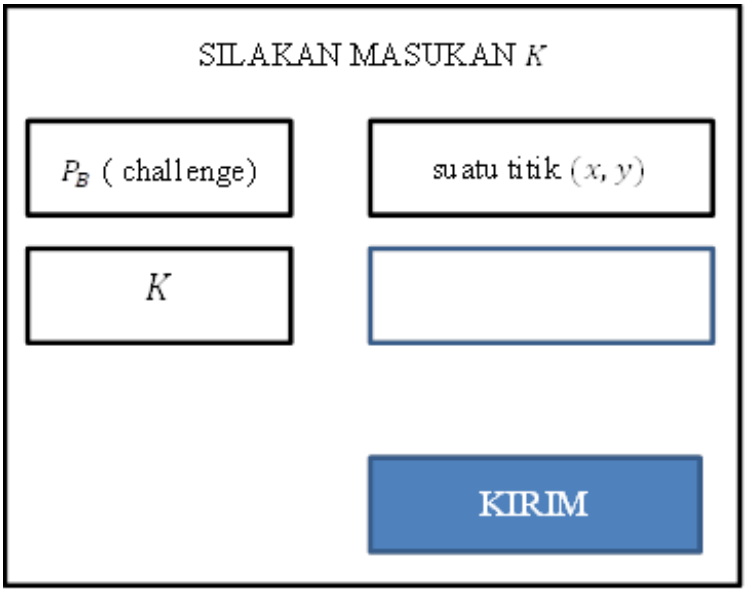
Gambar 3. 4 Tampilan Utama



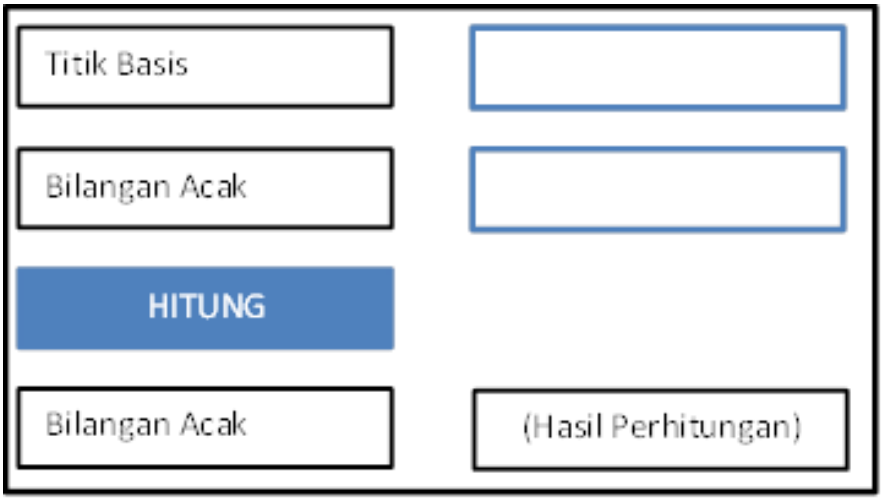
Gambar 3. 5 Tampilan Menu *Sign-In*



Gambar 3. 6 Input *Username* dan $\boldsymbol{P_A}$



Gambar 3. 7 Input ***K***



Gambar 3. 8 Kalkulator *user*

3.4.3 Algoritma

a. Sign-In

Berikut merupakan algoritma untuk menu *Sign-In* yang dilakukan oleh *user*:

- Tentukan *username* yang ingin digunakan.
- Input a.
- Input b.
- Input bilangan prima p.

- Cek titik pada kurva, jika input p bukan prima, tidak ada titik yang dapat muncul.
- Input $B\ (x, y)$.

b. Log-In

Berikut merupakan algoritma untuk menu *Log-in*:

- *User* membangkitkan bilangan acak m.
- *User* menghitung P_A.
- *User* menginput *username* dan P_A.
- *Server* membangkitkan bilangan acak n.
- *Server* menghitung P_B.
- *Server* menampilkan P_B (*challenge*).
- *User* menghitung K (*response*).
- *User* menginput K.
- *Server* menghitung K'.
- *Server* membandingkan K dan K', jika sama maka *log-in* berhasil.

c. Kalkulator *User*

Berikut merupakan algoritma untuk kalkulator *user*:

- Input titik yang akan dihitung.
- Pilih satu bilangan acak.
- Input bilangan acak yang dipilih.
- Program akan menampilkan hasil perhitungan.

3.5 Validasi

Pada tahap ini, dilakukan validasi terhadap program yang telah dibuat dengan cara membandingkan hasil yang diperoleh melalui program dengan hasil pengerjaan manual. Hasil yang diperoleh dari program dikatakan valid jika hasilnya sama dengan hasil perhitungan secara manual.

3.6 Kesimpulan

Tahap terakhir yang dilakukan adalah pengambilan keputusan berdasarkan hasil yang diperoleh selama penelitian. Selain itu juga menyertakan rekomendasi untuk penelitian berikutnya agar memperoleh hasil yang lebih maksimal.