BAB II

LANDASAN TEORI

2.1 Konsep Dasar Algoritma

Pemrograman sudah menjadi kegiatan yang sangat penting di era teknologi informasi saat ini. Program yang berjalan di berbagai *device* seperti komputer (*personal computer*), *netbook*, *handheld*, *web* (berbasis internet) pada dasarnya tidak dibangun begitu saja, melainkan ada suatu proses yang menjadi suatu pola kerja dari program itu sendiri yaitu algoritma.

2.1.1 Sejarah Algoritma

Algoritma mempunyai sejarah yang panjang. Jika dilihat dari asal kata nya yaitu "algoritma", kata ini tidak muncul dalam kamus Webster pada tahun 1957.

Menurut Rinaldi Munir (2011:10), Para ahli bahasa menemukan kata *algorism* berasal dari nama cendikiawan muslim yang terkenal yaitu Abu Ja'far Muhammad Ibnu Musa Al-Khuwarijmi (Al-Khuwarijmi dibaca oleh orang Barat menjadi *algorism*) dalam bukunya yang berjudul *Kitab Aljabar Wal-muqabala*, yang artinya "Buku Pemugaran dan Pengurangan" (*The book of restoration and reduction*). Dari judul buku itu kita memperoleh kata "aljabar" (*algebra*). Perubahan dari kata *algorism* menjadi *algorithm* muncul karena kata *algorism* sering dikelirukan dengan *arithmetic* sehingga akhiran –*sm* berubah menjadi –*thm*.

Pada tahun 1950 algoritma yang lebih condong ke arah *aritmatika* terbukti dengan dipakainya kata algoritma tersebut dalam "Algoritma Euclidean" yaitu algoritma yang mencari pembagi bersama terbesar (*Great*

Common Divisor) diantara kedua bilangan. Dalam algoritma ini sangat membantu dalam mencari nilai *enciphering* pada algoritma RSA.

Kemudian dari hal tersebut, algoritma dikembangkan ke arah prosedur komputasi sehingga komputer dapat bekerja seperti yang diharapkan seperti saat ini.

2.1.2 Definisi Algoritma

Algoritma berusaha melakukan langkah-langkah seefisien mungkin untuk mencapai tujuan semaksimal mungkin. Algoritma sebenarnya implementasi dari kehidupan sehari-hari misalnya algoritma stack dan algoritma queue yang merupakan implementasi dari antrian dan tumpukan yang terjadi dalam aktifitas sehari-hari.

Menurut Thomas H. Cormen (2009:5), Algoritma adalah prosedur komputasi yang mengambil beberapa nilai atau kumpulan nilai sebagai *input* kemudian di proses sebagai *output* sehingga algoritma merupakan urutan langkah komputasi yang mengubah input menjadi output.

2.2 Konsep Dasar Kriptografi

Kriptografi menjadi salah satu unsur penting dalam dunia informasi. Hal ini terkait dengan semakin canggihnya teknologi dan kemudahan dalam komunikasi sehingga memberikan efek samping yaitu semakin mudahnya orang untuk mendapatkan informasi.

2.2.1 Sejarah Kriptografi

Kriptografi mempunyai sejarah yang sangat menarik dan panjang. Kriptografi sudah digunakan 4000 tahun yang lalu yang diperkenalkan oleh orang-orang Mesir untuk mengirimkan pesan kepasukan Militer yang berada dilapangan dan supaya pesan tersebut tidak dapat dibaca oleh pihak musuh jika kurir pembawa pesan tersebut tertangkap.

Pada zaman Romawi kuno, ketika Julius Caesar ingin mengirimkan satu pesan rahasia kepada seorang Jendral di medan perang. Pesan tersebut harus dikirimkan melalui kurir, tetapi karena pesan tersebut mengandung sifat rahasia maka Julius Caesar tidak ingin pesan tersebut dibuka di tengah jalan. Di sini Julius Caesar memikirkan bagaimana cara mengatasinya yaitu dengan cara mengacak pesan tersebut menjadi suatu pesan yang tidak dapat dipahami oleh siapa pun kecuali hanya dapat dipahami oleh Jenderalnya saja. Tentu Sang Jenderal sudah diberitahu sebelumnya bagaimana cara membaca pesan yang teracak tersebut karena telah mengetahui kuncinya. Caesar mengganti semua susunan alphabet dari a, b, c yaitu a menjadi d, b menjadi e, c menjadi f dan seterusnya.

Pada perang dunia kedua, Jerman menggunakan *enigma* atau juga disebut dengan mesin rotor yang digunakan Hitler untuk mengirimkan pesan ketentaranya. Jerman sangat percaya pesan yang di kirim melalui Enigma tidak terpecahkan kode-kode enkripsinya. Tapi anggapan itu keliru, setelah bertahun-tahun sekutu dapat memecahkan kode-kode tersebut setelah mempelajarinya. Setelah Jerman mengetahui kode-kode

tersebut terpecahkan, maka enigma yang digunakan pada perang dunia kedua mengalami beberapa kali perubahan.

Selama bertahun-tahun kriptografi menjadi bidang khusus yang hanya dipelajari oleh pihak militer, seperti agen Keamanan Nasional Amerika (*National Security Agency*), Uni Soviet, Inggris, Perancis, Israel dan Negara-negara lainnya yang telah membelanjakan miliaran dolar untuk mengamankan komunikasi mereka dari pihak luar, tapi mereka selalu mempelajari kode-kode rahasia Negara lain, dengan adanya persaingan ini maka kriptografi terus berkembang sesuai dengan perkembangan zaman.

Namun pada 30 tahun terakhir ini, kriptografi tidak hanya dimonopoli oleh pihak militer saja, hal yang sama juga dilakukan oleh individu-individu yang menginginkan pesan dan komunikasi mereka tidak di ketahui oleh pihak lain. Apalagi pada zaman sekarang ini persaingan yang begitu tinggi, mereka rela mengeluarkan sekian dolar hanya untuk menjaga privasi mereka.

2.2.2 Pengenalan Algoritma Kriptografi

Secara terminologi Algoritma merupakan urutan langkah-langkah logis untuk penyelseian masalah yang disusun secara sistematis.

Menurut Ariyus (2006:36), Algoritma kriptografi merupakan langkah-langkah logis bagaimana menyembunyikan pesan dari orang-orang yang tidak berhak atas pesan tersebut dengan melakukan pembangkitan kunci, enkripsi dan dekripsi.

Algoritma Kriptografi terdiri dari tiga fungsi dasar yaitu :

1. Kunci

   Kunci yang di pakai untuk melakukan enkripsi dan dekripsi, kunci terbagi dua bagian yaitu kunci publik (*public key*) dan kunci privat (*private key*).

   Keamanan dari kriptografi modern hanya dengan merahasiakan kunci yang dimiliki oleh orang lain tanpa harus merahasiakan algoritma itu sendiri

2. Enkripsi

   Enkripsi merupakan hal yang sangat penting dalam kriptografi yang merupakan pengamanan data yang dikirimkan terjaga kerahasiaannya. Pesan asli disebut *plaintext* yang dirubah menjadi kode-kode yang tidak dimengerti. Enkripsi bisa di artikan dengan *cipher* atau kode. Beda halnya dengan enkripsi, untuk merubah *plaintext* ke *ciphertext* kita menggunakan algoritma yang dapat mengkodekan data.

3. Dekripsi

   Dekripsi merupakan kebalikan dari enkripsi, pesan telah di enkripsi dikembalikan kebentuk asalnya (*plaintext*) disebut dengan dekripsi pesan. Algoritma yang digunakan untuk dekripsi tentu berbeda dengan yang digunakan untuk enkripsi.

2.2.3 Macam-macam Algoritma Kriptografi

Menurut Ariyus (2006:44) Algoritma kriptografi terbagi menjadi tiga bagian berdasarkan kunci yang dipakainya yaitu :

1. Algoritma Simetri (menggunakan kunci yang sama saat enkripsi dan dekripsi)
2. Algoritma Asimetri (menggunakan kunci yang berbeda saat enkripsi dan dekripsi)
3. Fungsi *Hash*

2.2.3.1 Algoritma Simetris (Konvensional)

Algoritma simetri disebut juga sebagai algoritma konvensional adalah algoritma yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Disebut konvensional karena algoritma yang biasa digunakan orang sejak berabad-abad yang lalu adalah jenis ini. Keamanan algoritma simetri tergantung pada kunci. Membocorkan kunci berarti bahwa orang lain dapat mengenkrip dan mendekrip pesan tersebut. Agar komunikasi tetap aman, kunci harus tetap dirahasiakan. Yang termasuk algoritma kunci simetri adalah OTP, DES, RC2, RC4, RC5, RC6, IDEA, Twofish, Magenta, FEAL, SAFER, LOKI, CAST, Rijndael (AES), Blowfish, GOST, A5, Kasumi dan lain-lain.

2.2.3.2 Algoritma Asimetris

Algoritma asimetris di desain sedemikian rupa sehingga kunci yang digunakan untuk enkripsi berbeda daru kunci yang digunakan untuk dekripsi. Lebih jauh lagi, kunci dapat dihitung dari kunci enkripsi. Algoritma ini disebut algoritma kunci publik karena kunci enkripsi dapat dibuat publik sehingga orang lain dapat mengetahuinya, namun hanya orang tertentu dan sekaligus pemilik kunci dekripsi yang sekaligus dapat melakukan dekripsi pesan tersebut. Dalam hal ini kunci enkripsi sering disebut dengan kunci publik dan kunci dekripsi sering disebut dengan kunci privat. Kunci privat kadang-kadang sering disebut kunci rahasia.

Adapun yang termasuk ke dalam algoritma ini diantaranya adalah *Digital Signature Algorithm* (DSA), RSA (Rivest Shamir Adleman), Diffe-Helman (DH), *Elli[tic Curve Cryptography* (ECC), Kriptografi Quantum dan lain-lain.

2.2.3.3 Fungsi *Hash*

Fungsi Hash sering disebut dengan fungsi Hash satu arah (*one-way function*), *message digest*, *fingerprint*, fungsi kompresi dan *message authentication code* (MAC), merupakan suatu fungsi dimana pesan yang sudah diubah menjadi *message digest* tidak dapat dikembalikan lagi menjadi pesan semula. Dua pesan berbeda akan menghasilkan nilai *hash* yang berbeda pula.

2.3 Konsep Dasar Algoritma RSA

RSA merupakan algoritma kriptografi kunci publik (*public key cryptography*) dengan penggunaan kunci yang berbeda pada proses enkripsi dan dekripsi. Algoritma RSA dijabarkan pada tahun 1997 oleh tiga orang : Ron Rivest, Adi Shamir dan Len Adleman dari MIT (*Massachusetts Institute of Technology*). Hurup RSA sendiri berasal dari inisial nama mereka (**R**ivest-**S**hamir-**A**dleman).

Algoritma RSA melakukan pemfaktoran bilangan yang sangat besar, oleh karena alasan tersebut RSA dianggap aman. Untuk membangkitkan kedua kunci, yang dipilih dua bilangan acak yang besar. Skema yang dikembangkan oleh Rivest, Shamir dan Adleman yang mengekspresikan bahwa *plaintext* dienkripsi menjadi blok-blok yang setiap blok memiliki nilai bilangan biner yang diberi simbol "n", *plaintext* blok "M" dan *ciphertext* blok "C". Untuk melakukan enkripsi pesan "M" dibagi ke dalam blok-blok *numeric* yang lebih kecil dari pada "n" (data biner dengan pangkat terbesar), jika bilangan prima yang panjangnya 200 digit dan dapat menambah beberapa bit 0 di kiri bilangan untuk menjaga agar pesan tetap kurang dari nilai "n".

Menurut Ir. Rinaldi Munir, M.T (2004:2), rumus pembentukan algoritma RSA didasarkan pada persamaan matematika dan didasarkan pada teorema Euler sehingga didapat rumus untuk enkripsi.

Adapun rumus untuk enkripsi yaitu :

$$C = M^e \bmod n$$

keterangan :

C : *ciphertext* (blok *plaintext* yang sudah dienkripsi)

M : *message* (blok pesan yang akan dienkripsi)

e : *enciphering*

n : nilai modulus

dan rumus untuk melakukan proses enkripsi yaitu :

$$M = C^d \bmod n$$

keterangan :

C : *ciphertext* (blok *plaintext* yang sudah dienkripsi)

M : *message* (blok pesan yang akan dienkripsi)

d : *deciphering*

n : nilai modulus

2.4 Pengantar Java

Java adalah sebuah bahasa pemrograman yang banyak digunakan. Dengan mengadopsi OOP (*Object Oriented Programming*) dan JVM (*Java Virtual Machine*), Java mampu berjalan pada *device* apapun sehingga Java menjadi sangat *familiar* di jaman sekarang ini.

2.4.1 Sejarah Java

Pada tahun 1991, sekelompok insinyur Sun Microsystem, Inc., dipimpin oleh Patrick Naughton dan James Gosling merancang bahasa untuk perangkat *consumer* seperti *cable TV box*. Karena perangkat tidak mempunyai banyak memori, bahasa harus berukuran kecil dan

menghasilkan kode yang liat. Karena pabrikan-pabrikan berbeda memilih pemroses-pemroses berbeda maka bahasa harus netral arsitektur manapun. Proyek ini diberi nama kode "**Green**".

Kebutuhan untuk kecil, liat dan lintas *platform* mengantar tim mempelajari implementasi pascal yang telah dicoba. Niklaus Wirth, pencipta Pascal telah merancang bahasa kode perantara (*intermediate code*) portabel untuk mesin hipotesis. Kode antara ini kemudian dapat digunakan disembarang mesin yang memiliki *interpreter*. Proyek Green menggunakan mesin maya untuk mengatasi isu netral terhadap arsitektur mesin.

Produk pertama proyek "Green" adalah "*7", sebuah kendali jauh yang cerdas. Karena pasar belum tertarik dengan produk *consumer* cerdas maka proyek Green harus menemukan pasar lain. Penerapan mengarah menjadi teknologi di *web*. Pada tahun 1995, Netscape memutuskan membuat *browser* dilengkapi Java. Setelah itu diikuti oleh IBM, Symantec bahkan Microsoft.

Setelah itu, gaung Java mulai terdengar. Berbagai industri meliriknya. Dengan strategi terbuka, banyak industri menyambutnya. Berbagai universitas Amerika, Jepang dan Eropa mengubah pengenalan bahasa pemrograman menjadi Java, meninggalkan C++. Java lebih sederhana dan mengakomodasikan hampir seluruh fitur penting bahasa-bahasa pemrograman ciptaan manusia.

2.4.2 Kelebihan Java

Bahasa pemrograman berorientasi objek telah menjadi aliran utama (*mainstream*), Java berorientasi objek sejati melebihi C++. Segala sesuatu di Java kecuali sedikit tipe dasar (*int, float, double, char*) adalah objek. Dibanding bahasa C++, Java menawarkan kemudahan diantaranya :

1. Java menghilangkan keperluan dealokasi *manual*. Java dilengkapi *garbage collector* yang bertugas mendealokasi memori yang tidak diperlukan. Tidak ada lagi upaya pemrogram untuk *dispose()*. Pemrogram tidak dibebani urusan korupsi dan sampah memori.
2. Java menerapkan *array* sebenarnya dan menghilangkan keperluan aritmatika *pointer* yang berbahaya dan berpeluang besar mengakibatkan kesalahan.
3. Menghilangkan keniscayaan operasi penugasan (*assignment*) sebagai pengujian atas kesamaan di kalimat bersyarat.
4. Menghilangkan perwarisan jamak (*multiple inheritance*) dengan fasilitas **interface**. **Interface** memberi banyak member banyak manfaat antara lain untuk pewarisan jamak tanpa dibebani kompleksitas dari pengelolaan hirarki pewarisan jamak.

2.4.3 Edisi Java

Sebagian besar bahasa pemrograman modern berdiri di atas pustaka-pustaka kelas yang telah ada untuk mendukung fungsionaltias. Pada bahasa Java, kelompok-kelompok kelas yang berkaitan erat dimasukan di

satu paket, bervariasi sesuai edisi Java. Masing-masing paket untuk maksud tertentu : *applet*, aplikasi standar, skala *enterprise* dan produk konsumer.

Java adalah bahasa yang dapat digunakan di sembarang *platform*, di beragam lingkungan. *The Java 2 Platform* terdiri dari tiga edisi untuk keperluan berbeda berikut :

1. *Java 2 Standard Edition* (J2SE)

   Menyediakan lingkungan pengembangan yang kaya fitur, stabil, aman dan *cross platform*. Edisi ini mendukung konektivitas basis data, rancangan antarmuka pemakai, masukan/keluaran (*I/O*) dan pemrograman jaringan dan termasuk sebagai paket-paket dasar bahasa Java.

2. *Java 2 Enterprise Edition* (J2EE)

   Menyediakan kakas untuk membangun dan menjalankan *multitier enterprise applications*. J2EE berisi paket-paket di J2SE ditambah paket-paket untuk mendukung pengembangan *Enterprise Javabeans*, *Java Servlets*, *JavaServer Pages*, XML dan kendali transaksi yang fleksibel.

3. *Java 2 Micro Edition* (J2ME)

   Untuk beragam *consumer electronic product*, seperti *pager*, *smart card*, *cell phone*, *handheld PDA* dan *set-top box*. J2ME sembari menyediakan bahasa Java yang sama, unggul dalam portabilitas dan *safe network delivery* seperti J2SE dan J2EE. J2ME menggunakan

sekumpulan paket lebih kecil. J2ME berisi *subset* J2SE ditambah *subset Micro Edition* javax.microedition.io. Aplikasi J2ME juga dapat diskala agar dapat bekerja dengan J2SE dan J2ME.

2.5 Konsep *Client-Server*

Client-Server merupakan sebuah paradigma dalam teknologi informasi yang merujuk kepada cara mendistribusikan aplikasi ke dalam dua pihak yaitu pihak klien dan pihak server.

Dalam model klien/server, sebuah aplikasi dibagi menjadi dua bagian yang terpisah, tapi masih merupakan sebuah kesatuan yakni komponen klien dan komponen server. Komponen klien juga sering disebut sebagai *front-end*, sementara komponen server disebut sebagai *back-end*. Komponen klien dari aplikasi tersebut dijalankan dalam sebuah *workstation* dan menerima masukan data dari pengguna. Komponen klien tersebut menyiapkan data yang dimasukan oleh pengguna dengan menggunakan teknologi pemrosesan tertentu dan mengirimkannya kepada komponen server yang dijalankan di atas mesin server, umumnya dalam bentuk *request* terhadap beberapa layanan yang dimiliki oleh server itu sendiri. Komponen server akan menerima request dari klien, dan langsung memprosesnya dan mengembalikan hasil pemrosesan tersebut ke klien yang melakukan *request.* Klien menerima informasi hasil pemrosesan data yang dilakukan server dan menampilkannya kepada pengguna, dengan menggunakan aplikasi yang berinteraksi dengan pengguna.

Dalam penelitian ini, untuk keperluan sistem *client server* penulis menggunakan dua konsep Java yaitu RMI (*Remote Method Invocation*) dan Socket.

2.6.1 RMI (*Remote Method Invocation*)

RMI menyediakan sarana dimana *client server* dapat berkomunikasi dan saling bertukar informasi. RMI memungkinkan pengembang perangkat lunak untuk merancang aplikasi terdistribusi dimana *methods* dari *remote object* dapat dipanggil dari JVM (*Java Virtual Machine*) lain, yang mungkin berjalan pada *host* yang berbeda.

Remote object adalah *object* dari RMI yang direferensikan secara *remote*. Pemrogram seakan-akan memanggil *method* dari kelas file lokal sedangkan dalam kenyataannya *method* yang bersifat *remote* disimpan di JVM *server* dan diakses oleh *client*. Dalam RMI, *server* akan menunggu *remote object*, membuat referensi dan menunggu *client* untuk memanggil *methods* dari *remote object* ini. Sedangkan *client* akan mendapatkan *remote reference* dari satu atau lebih *remote object* dan memanggil *methods* untuk *remote object* tersebut.

Adapun dalam membangun RMI diperlukan langkah-langkah yaitu :

1. Membuat interface yang turunan dari kelas java.rmi.Remote dan setiap method yang dideklarasikan melemparkan eksepsi *java.rmi.RemoteException*.

2. Membuat kelas yang turunan dari kelas *java.rmi.UnicastRemoteObject* dan mengimplementasi semua *method* dari *interface* yang sudah dibuat tadi.
3. Kelas yang telah dibuat tadi harus memiliki konstruktor yang melempar eksepsi *java.rmi.RemoteException*.
4. Seluruh *method* yang di-*override* di *interface* tersebut harus melempar eksepsi *java.rmi.RemoteException*.

2.6.2 Socket

Socket menyediakan sebuah *interface* untuk pemrograman jaringan yang berada pada *transport* layer. Komunikasi jaringan menggunakan *socket* berkaitan erat dengan pemrograman *I/O*. Itu artinya, sebuah *socket* yang dibangun dalam bahasa Java dapat berkomunikasi dengan *socket* program Non-Java.

Menurut Jan Graba (3:2007), *socket* adalah salah satu titik akhir dari link komunikasi dua arah antara dua program atau lebih yang berjalan pada jaringan dan terikat pada nomor port tertentu. *Port* adalah sebuah koneksi *logical* untuk sebuah komputer yang diidentifikasi dengan sebuah angka yang berada dalam rentang 1-65535.

2.6 Pengertian *Instant Messenger*

Instant Messenger adalah sebuah teknologi internet yang memungkinkan orang dapat melakukan komunikasi atau mengirimkan pesan-pesan singkat secara bersamaan (*realtime*) dengan menggunakan teks kepada pengguna lainnya yang sedang terhubung ke dalam jaringan yang sama.

Istilah pesan instan (*instant messenger*) saat ini bermula pada munculnya sebuah teknologi yang dipopulerkan oleh American Online (AOL) yang kemudian diikuti oleh banyak *provider* seperti Yahoo (*Yahoo Messenger*), Google (*GTalk*) dan Microsoft (Window Live Messenger) dan perusahaan-perusahaan lainnya.

2.7 Perangkat Lunak Pembantu

Dalam penelitian ini, penulis menggunakan perangkat lunak yang digunakan untuk dapat memudahkan mencari sebuah solusi yang telah ditetapkan. Adapun perangkat lunak tersebut diantaranya :

2.7.1 NetBeans IDE

NetBeans adalah *Integrated Development Environment* (IDE) yang berbasiskan Java dari Sun Microsystems yang dapat berjalan di atas Swing. Swing adalah sebuah teknologi Java untuk pengembangan aplikasi *Desktop* yang dapat berjalan di berbagai macam *platform* seperti Windows, Linux, Mac OS X dan Solaris.

Suatu IDE adalah lingkup pemrograman yang diintegrasikan ke dalam suatu aplikasi perangkat lunak yang menyediakan pembangun *Graphical User Interface* (GUI).

2.7.2 JDK 7 (*Java Development Kit 7*)

JDK adalah perangkat lunak Java yang didalamnya sudah *include library* untuk keperluan *debugging* dan *library* untuk keperluan *running*. JDK adalah perangkat lunak yang digunakan untuk membangun suatu aplikasi-aplikasi Java. Tanpa JDK, tidak akan bisa membangun aplikasi tersebut. Sedangkan JRE (*Java Runtime Environment*) adalah perangkat lunak yang digunakan untuk keperluan *runtime* atau menjalankan program yang telah dikompilasi sebelumnya oleh JDK. Dalam JRE inilah tersimpan JVM (*Java Virual Machine*) sebagai mesin maya dalam Java.

Sumber

1. id.wikipedia.org/wiki/Java
2. Shildt, Herbert.2005. *Java: A Beginner Guide, Third Edition*.Mc Graw Hill. New York.
3. Nugroho, Adi.2008. *Algoritma dan struktur data dalam bahasa java*.Andi. Yogyakarta
4. http://teknologi.kompasiana.com/internet/2010/07/23/definisi-algoritma/
5. http://id.wikipedia.org/wiki/Algoritma
6. http://www.studentpulse.com/articles/41/a-brief-history-of-cryptography
7. Hariyanto, Bambang.2007. *Esensi-esensi bahasa pemrograman Java*.Informatika.Bandung
8. http://id.wikipedia.org/wiki/Klien-server
9. http://fhenyfhen.blogspot.com/2011/02/sejarah-netbeans.html
10. http://id.wikipedia.org/wiki/MySQL
11. Ariyus, Dony.2006.*KRIPTOGRAFI Keamanan Data dan Komunikasi*.Graha ilmu.Yogyakarta