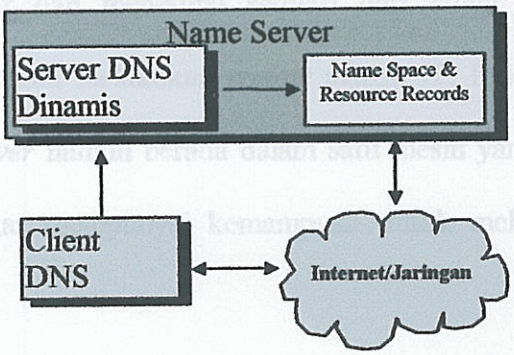
BAB IV

ANALISIS DAN IMPLEMENTASI

4.1 Model DNS Dinamis

Sistem DNS hanya efisien dilakukan pada sistem dengan *host* yang memiliki alamat IP yang tetap. Pada sistem dengan *host* yang memiliki alamat IP dinamis, maka kegiatan administrasi domain (*update*, hapus, tambah, dsb) sangat akan merepotkan bagi *system administrator*, mengingat frekuensi penggunaan IP yang sering pada sistem IP dinamis. DNS Dinamis berperan sebagai alat bantu bagi *system administrator* pada tipe sistem dengan alamat IP dinamis.



Gambar 4. Model DNS Dinamis

Model DNS Dinamis yang dibahas dalam tulisan ini dapat digambarkan seperti pada Gambar 4. Model ini bertujuan untuk menyediakan pemetaan nama host ke alamat IP dengan segera. Agar FQDN suatu *client* yang terdaftar pada *server* DNS Dinamis dapat dikenal di Internet/Jaringan, maka *client* tersebut harus `melaporkan' alamat IP dan

nama *host* ke *server* DNS Dinamis, yang kemudian memperbaharui RR pada name *server*. Model ini dapat dijelaskan secara detil dengan *Data Flow Diagram (DFD)* pada Lampiran 5.

Dalam gambar terlihat fungsi *server* DNS Dinamis yang berfungsi sebagai perantara antara *client* dengan *name server*. Segala perubahan alamat IP pada *client* dirubah oleh *server* DNS Dinamis pada *name server*.

DNS Dinamis dalam model tersebut terdiri dari dua komponen :

4.1.1 *Server* DNS Dinamis

Server DNS Dinamis bertugas melakukan hal-hal administratif pada domain secara *daemon* (*background process*). Artinya, *Server* DNS Dinamis ini merupakan suatu program *server* yang terus menerus mendengar dan menerima *request* dari *client* pengguna jasa DNS Dinamis untuk melakukan *update* alamat IP. Program ini terpisah dari *name server* namun berada dalam satu mesin yang sama dengan *name server* agar mempunyai kemampuan untuk melakukan *restart name server*.

Program ini akan menunggu *request* dari *trusted client* pada suatu *port* TCP tertentu dan akan mengupdate RR pada *master file* yang dialokasikan untuk *client* tersebut sesuai dengan informasi yang berupa alamat IP dan nama *host* yang diberikan oleh *client*. Program ini juga akan memutuskan hubungan dengan *client* yang bukan termasuk ke dalam daftar *trusted client*, sehingga tidak sembarang *host* boleh bergabung di bawah suatu domain yang menyelenggarakan layanan

DNS Dinamis ini. *Trusted client* adalah suatu file yang berisi daftar *client* mana saja yang dapat bergabung dengan jaringan DNS Dinamis ini.

4.1.2 *Client* DNS Dinamis

Client DNS Dinamis adalah program yang bertugas untuk mengirimkan informasi mengenai nama *host client* dan alamat IP yang dimiliki oleh *client*. Nama *host client* ini harus ada di dalam daftar *trusted client* yang dimiliki oleh *server* DNS Dinamis. *Trusted client* seyogyanya menghubungi *server* DNS Dinamis segera setelah mendapatkan alamat IP dinamis dari *server* pemberi IP agar FQDN-nya dapat langsung dikenal dalam jaringan. Metode penentuan untuk menghubungi *server* DNS Dinamis ini bergantung pada sistem operasi yang digunakan. Bila suatu saat *client* terhubung ke jaringan, maka program *client* DNS Dinamis akan secara otomatis mengirimkan informasi tentang dirinya ke *server* DNS Dinamis.

Apabila *client* terputus secara tiba-tiba dari jaringan, maka informasi alamat IP miliknya akan tetap ada pada *name server* domain. Akibatnya apabila ada suatu *host* A yang ingin mengakses *host* B (*client* DNS Dinamis yang baru saja terputus dari jaringan) dengan melakukan *request* FQDN milik *host* B maka *name server* akan memberikan alamat IP yang sudah tidak valid lagi.

Untuk mengatasi hal ini, maka *client* secara periodik terus melaporkan alamat IP kepada *server* DNS Dinamis. Bila ternyata *client* tersebut tidak lagi melaporkan alamat IP, maka *client* dianggap terputus dari jaringan dan alamat IP diganti dengan **0.0.0.0**.

4.2 Contoh Masalah

Seorang user ingin menggunakan komputernya sebagai *web server* dan ingin menggunakan FQDN **sipalui.linux.or.id** untuk komputernya sehingga nanti komputernya dapat menjadi web *server* dengan alamat **http://sipalui.linux.or.id**. Setiap kali tersambung ke Internet, komputer tersebut mendapatkan alamat IP dinamis dari ISP tempat user tersebut berlangganan. Dengan demikian, apapun alamat IP yang didapat harus dipetakan menjadi **sipalui.linux.or.id** setiap kali komputernya tersambung ke Internet.

Maka, saat mendapatkan IP, *client* DNS Dinamis pada komputer itu segera menghubungi *server* DNS Dinamis di *primary name server* domain **linux.or.id**. agar segera memetakan IP yang diterimanya menjadi FQDN **sipalui.linux.or.id.**

4.3 Protokol Dinamid v1.0

Model DNS Dinamis yang disajikan di sini diterapkan dengan menggunakan protokol dinamid v1.02. Protokol ini dibuat oleh penulis untuk digunakan sebagai protokol standar pada Virtual Lembaga Swadaya Masyarakat (VLSM) CarlosNet (**http://www.carlosnet.or.id**) pada layanan DNS Dinamis-nya.

Protokol dinamid digunakan sebagai tata cara pelaksanaan segala perubahan pada *name server* dan tata cara komunikasi antara *server* DNS Dinamis dengan *client*.

Ada dua aspek penting pada protokol ini, yaitu hubungan antara *client* DNS Dinamis dengan *server* DNS Dinamis dan hubungan antara *server* DNS Dinamis dengan name *server*.

Pada aspek pertama, *client* DNS Dinamis melangsungkan hubungan komunikasi dengan *server* DNS Dinamis pada *session layer* hingga *application layer* dari tujuh lapisan OSI dengan menggunakan *socket* TCP/IP yang terletak pada lapisan *transport layer* (TCP) dan *network layer* (IP) sebagai medianya (Silberschatz dan Galvin 1995). *Server* DNS Dinamis berjalan sebagai *daemon* (*background process*) dan mendengarkan *request* pada port TCP 6060.

Mulanya *client* DNS Dinamis akan melakukan koneksi TCP dengan *server* dalam *session layer*. Session dilakukan *client* dengan menghubungi port TCP 6060 pada *server*. Setelah *socket* terbentuk pada level *transport layer*, *client* lalu memulai *session* dengan mengirimkan informasi alamat IP kepada *server*, *server* kemudian mengambil informasi IP tersebut dengan menggunakan perintah getpeername(). Dalam DFD di Lampiran 5 ada pada nomor 1.

Pada saat hubungan komunikasi antara *client* dan *server* terlaksana pada *application layer*, terdapat dua perintah yang digunakan sebagai bahasa bagi *server* dan *client* dalam melaksanakan komunikasi. Perintah-perintah tersebut adalah ID dan PASSWD.

Pada saat hubungan antara *client* dan *server* terlaksana, maka protokol memasuki modus **AUTHORIZATION** kemudian memberi pesan selamat datang yang berupa string "100 +OK" serta nomor versi protokol dan mencatatnya di file *log*, misalnya

100 +OK dinamid DNS v1.02 Ready. Connected to 202.159.64.184

Setelah menampilkan pesan ini, bila *client* memiliki nomor versi yang sama dengan *server* maka akan melanjutkan komunikasi. Bila tidak maka *client* harus segera memutuskan koneksi. Dari sisi *server*, *timeout* harus diberlakukan agar *server* tidak selalu menunggu *client*. Default *timeout* adalah 60 detik. Dan mulai saat ini pula dalam *presentation layer* dapat dilakukan enkripsi lalu lintas data. Alternatif metode enkripsi akan diberikan di bagian **Enkripsi Data.**

Client mengirimkan identitas yang merupakan nama *host* dengan perintah ID. Format perintah ID adalah :

ID<SP><*host*name><LF>[<CR>]

<SP> adalah karakter spasi (ASCII no 0x20). *host*name merupakan nama *host client* yang terdaftar dalam trusted *client* pada *server* DNS Dinamis. String *host*name dibentuk dari serangkaian karakter alfanumerik ASCII (a-z, 0-9) yang jumlahnya tidak lebih dari 63 karakter yang merupakan batasan dari konsep *name space* (Mockapetris 1987a) dan harus diakhiri oleh karakter. <LF> adalah karakter *Line Feed* (ASCII no 0x0A). <CR> adalah karakter *Carriage Return* (ASCII no 0x0D) dan tidak harus ada dalam perintah ini (*optional*). Dalam DFD di Lampiran 5 mode **AUTHORIZATION** ini ada pada nomor 2.

Setelah mendapatkan string identitas ini, *server* akan melakukan *parsing* string ini dan mencari hostname dalam daftar *trusted client*. Metode penyimpanan daftar *trusted client* dibahas di bagian **Metode Penyimpanan File**.

Bila IP hostname ternyata sama dengan IP yang tercatat pada *database master file*, maka dianggap alamat IP tidak berubah dan *server* akan memutuskan koneksi dengan pesan "130 +OK", misalnya:

130 +OK Goodbye

Bila nama *host* yang diterima tidak ada dalam daftar *trusted client*, maka *server* segera memutuskan hubungannya dan mencatat kejadian ini dalam file *log* dan memberi pesan *error* yang merupakan string "500 -ERR", misalnya:

500 -ERR Not In Trusted *Client*

Bila nama *host* tersebut sah (terdapat dalam daftar *trusted client*), maka *server* akan memberikan pesan sukses yang merupakan string "110 +OK" dan mencatatnya di file *log*, misalnya:

110 +OK Continue

dan menunggu *client* mengirimkan password dengan perintah PASSWD. Format perintah ini adalah :

PASSWD<SP><password><LF>[<CR>]

password adalah untaian karakter alfanumerik sebanyak maksimal 8 karakter dan minimal 1 karakter. Bila kosong atau tidak sah maka *server* memutuskan hubungan dengan *client* dan mencatat kejadian ini pada file *log* dan memberikan pesan *error* yang merupakan string "510 -ERR", misalnya:

510 -ERR Invalid Password

Bila password sah, maka protokol memasuki modus **UPDATE** dengan memberikan pesan sukses yang merupakan string "120 +OK", misalnya :

120 +OK Password Accepted

Pada modus ini, *server* DNS Dinamis memasukan *hostname* dan alamat IP-nya dalam daftar *batch* untuk dilakukan update kemudian. Sesaat sebelumnya, *batch* tersebut harus di*lock* terlebih dahulu agar bila ada *client* lain yang melakukan update RR bersamaan tidak terjadi tabrakan. Dan setelah itu, *batch* dapat langsung di*unlock*. Dalam DFD di Lampiran 5 proses *update* ini ada pada nomor

Pada aspek kedua, setelah RR diupdate pada master file dirubah, maka *name server* harus di*restart* agar informasi alamat IP pada *master file* dapat segera dikenali pada jaringan. Pelaksanaan *restart name server* dapat dilakukan secara terpisah oleh program *scheduler* lain yang dilaksanakan dalam suatu periode waktu tertentu. Misalnya pada sistem Unix, *restart name server* dapat diserahkan pada program *Cron* atau program sejenisnya. Periode waktu yang digunakan sebaiknya kecil agar FQDN *client* dapat segera dikenali di jaringan. Nilai yang cukup untuk periode waktu ini adalah 5 menit.

Sebelum melakukan *restart*, *server* melakukan *update master file* dengan bersumber dari *batch* yang telah dibuat. Pada saat ini, file *batch* di*lock* dan kemudian di*unlock* dan dihapus isinya bila update master file telah selesai.

RR yang di*update* pada master file adalah SOA dan IN A dari masing-masing hostname *client*. Pada RR SOA, field yang diganti hanya field SERIAL saja.

4.4 Metode Penyimpanan File

Trusted *Client*. File yang menyimpan daftar trusted *client* berisi dua field, yaitu field hostname dan field password.

***Batch* File.** File ini menyimpan daftar *client* yang akan diupdate pada periode update mendatang. File ini berisi dua field, yaitu field hostname dan field IP. File ini harus dikosongkan setelah *name server* melakukan *restart*.

File *Log*. File *log* yang digunakan dipisahkan antara *log* akses dengan *log error*. *Log* akses mencatat setiap transaksi non *error* yang terjadi, dan *log error* mencatat setiap transaksi *error* yang terjadi.

Ketiga file di atas dapat disimpan menggunakan database khusus ataupun berbentuk *flat plain text*.

4.5 Instalasi Server Dinamid

Server dinamid harus memiliki hak (*priviliges*) untuk melakukan *restart name server*, karena itu server dinamid harus dijalankan oleh *root* atau *supervisor*. Server ini juga dapat dijalankan secara otomatis saat mesin server dinyalakan melalui *superserver* (Unix) atau *NT Services* (Windows NT).

Apabila ada *host* yang ingin ditambahkan maka *system administrator* harus mempersiapkan:

1. penambahan entri *host* tersebut di daftar *trusted client* dan penentuan password.

2. penambahan entri *host* tersebut di dalam master file..

4.7 Enkripsi Data

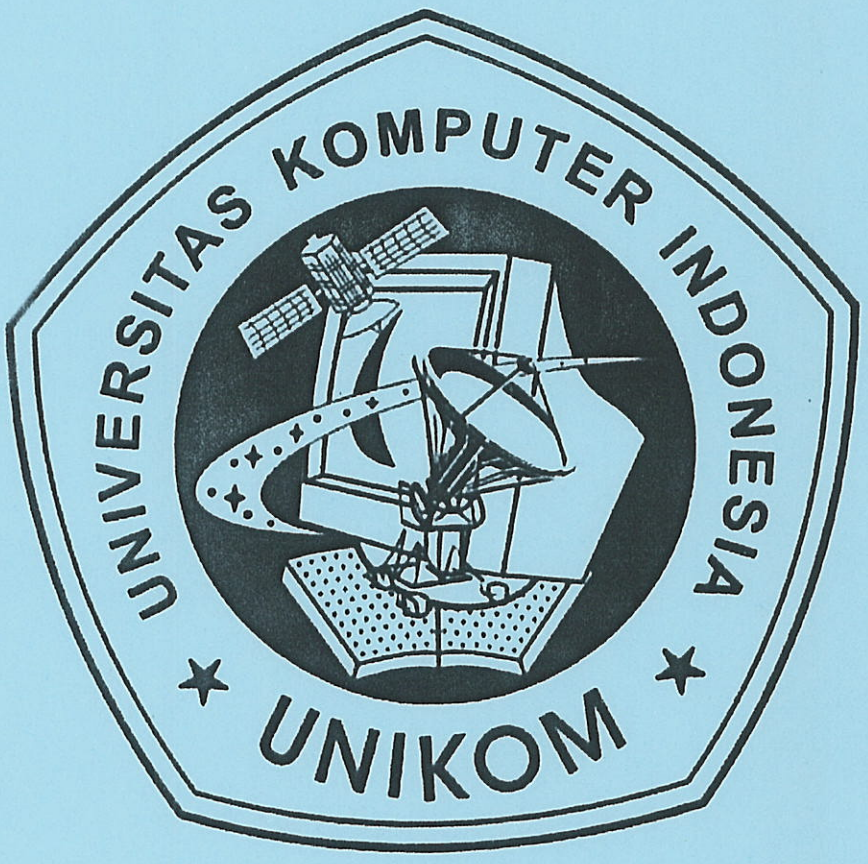
Enkripsi data dapat menggunakan metode enkripsi yang biasa dipakai, misalnya metode *International Data Encryption Algorithm* (IDEA) atau *Message Digest Encryption* (MD5). Metode-metode ini tidak dibahas dalam penelitian ini.

Sebagai kunci enkripsi digunakan password yang sama dengan yang diberikan pada perintah PASSWD. Dengan enkripsi data lalu lintas data terlindung dengan aman. Bila menggunakan enkripsi, maka tidak mungkin bagi seorang *cracker* untuk menyadap dan mendekripsi lalu lintas data yang lewat.

Sebagai gambaran, bila menggunakan metode IDEA maka bila setiap orang di bumi (1996) memiliki satu komputer yang digunakan untuk mencoba mendekripsi data, maka akan butuh waktu 2×10^{12} tahun, sedangkan umur bumi baru 4.5×10^{9} tahun (Visser 1996).

4.7 Contoh Implementasi

Contoh implementasi protokol dinamid disajikan pada lampiran 1. *Server* dinamid dibuat dengan bahasa PERL pada platform *Caldera OpenLinux Base 1.2* versi *kernel* 2.0.33. *Server* menggunakan *superserver* inetd dipadu dengan program tcpenv dari paket distribusi *qmail* 1.03.


UNIVERSITAS KOMPUTER INDONESIA
UNIKOM