**LAPORAN TUGAS AKHIR**

# EVALUASI KINERJA *DOMAIN FILTERING* MENGGUNAKAN BASIS DATA

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana
Jurusan Teknik Informatika


DISUSUN OLEH :

Nama : Bithana Paragustin
No. Mhs : 07 523 024

**JURUSAN TEKNIK INFORMATIKA**
**FAKULTAS TEKNOLOGI INDUSTRI**
**UNIVERSITAS ISLAM INDONESIA**
**YOGYAKARTA**
**2012**

**LAPORAN TUGAS AKHIR**

# EVALUASI KINERJA *DOMAIN FILTERING* MENGGUNAKAN BASIS DATA

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana
Jurusan Teknik Informatika


DISUSUN OLEH :

Nama : Bithana Paragustin
No. Mhs : 07 523 024

**JURUSAN TEKNIK INFORMATIKA**
**FAKULTAS TEKNOLOGI INDUSTRI**
**UNIVERSITAS ISLAM INDONESIA**
**YOGYAKARTA**
**2012**

**LEMBAR PENGESAHAN PEMBIMBING**

# EVALUASI KINERJA *DOMAIN FILTERING* MENGGUNAKAN BASIS DATA

**TUGAS AKHIR**

Oleh :

Nama : Bithana Paragustin

NIM : 07523024

Yogyakarta, 24 Mei 2012

Menyetujui,

Pembimbing Tugas Akhir

Syarif Hidayat, S.Kom, MIT



iii

**LEMBAR PENGESAHAN PENGUJI**

# EVALUASI KINERJA *DOMAIN FILTERING* MENGGUNAKAN BASIS DATA

**TUGAS AKHIR**

Oleh :

Nama : Bithana Paragustin

NIM : 07523024

Telah Dipertahankan di Depan Sidang Penguji Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 25 Juni 2012

**Tim Penguji**

Syarif Hidayat, S.Kom, MIT.
Ketua

R. Teduh Dirgahayu, ST., M.Sc., Ph.D.
Anggota 1

Ahmad Munasir Rafie P, ST., MIT
Anggota 2

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Teknologi Industri
Universitas Islam Indonesia

Yudi Prayudi, S.Si., M.Kom

## LEMBAR PERNYATAAN KEASLIAN HASIL TUGAS AKHIR

Saya yang bertandatangan di bawah ini,

Nama : Bithana Paragustin

No. Mahasiswa : 07 523 024

Menyatakan bahwa seluruh komponen isi dalam Laporan Tugas Akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti bahwa ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, maka saya siap menanggung resiko dan berkonsekuensi apapun.

Demikian pernyataan ini saya buat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 24 Mei 2012

Bithana Paragustin

## HALAMAN PERSEMBAHAN

*Untuk kedua Orang tua yang memberikan kesempatan melihat dunia,*

*Untuk mama yang telah memberikan restu, dukungan dan kesempatan untuk menuntut ilmu di informatika UII,*

*Untuk papa atas segala kasih sayang dan perhatiannya,*

*Untuk adik-adikku agar menjadi lebih baik dari pendahulunya,*

*dan untuk semua orang yang lelah melihat satu mahasiswa ini.*

## HALAMAN MOTTO

"Tidaklah Kami ciptakan bangsa jin dan manusia melainkan untuk beribadah kepada-Ku."

(Adz-Dzariyat [51]:56)

"Tuntutlah ilmu sejak dalam kandungan hingga ke liang lahat"

(Al Hadits)

"Terkadang keberhasilan bukan hanya dari ada atau tidaknya sesuatu, namun bisa juga dari berurutan atau tidaknya sesuatu tersebut. Seperti resep, algoritma, maupun parameter…"

(Bithana, Informatika UII 2007, 2012)

Usaha tanpa doa itu sombong, doa tanpa usaha itu bohong.

# KATA PENGANTAR

*Assalamu'alaikum Wr. Wb.*

Puji syukur kehadirat Allah SWT atas rahmat, taufik serta hidayah-Nya laporan tugas akhir dengan judul "Evaluasi Kinerja *Domain Filtering* Menggunakan Basis Data" ini dapat diselesaikan dengan baik.

Tugas akhir ini disusun untuk mengetahui waktu kinerja pencarian *domain filtering* proxy menggunakan basis data dan file teks dengan data acak (*worst-case* file) dan kinerja pencarian basis data dengan data acak dan terurut. Tugas akhir ini diharapkan dapat memberikan informasi yang bermanfaat bagi pengguna dan pengembang squid proxy server.

Ucapan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu dalam penyelesaian Tugas Akhir ini baik secara langsung maupun tidak langsung. Semoga Allah SWT memberikan limpahan rahmat dan ridho-Nya kepada mereka semua. Ucapan terimakasih dipersembahkan kepada:

1. Allah SWT atas segala cinta kasih, rahmat, berkah dan karunia berlimpah yang tiada pernah habis.
2. Rasulullah SAW atas ajaran, inspirasi, serta warisannya bagi umat manusia.
3. Bapak Syarif Hidayat, S.Kom, M.I.T. selaku Dosen Pembimbing yang telah banyak membantu dan mengarahkan dalam mengerjakan tugas akhir ini, semoga waktu yang diluangkan untuk tugas akhir ini berkah.
4. Bapak R. Teduh Dirgahayu, ST., M.Sc., Ph.D. dan Bapak Ahmad Munasir Rafie Pratama, ST., MIT selaku Dosen Penguji Pendadaran yang telah meluangkan waktu untuk memberi masukan dan kesempatan untuk melaju ke fase selanjutnya, semoga segala yang diberikan untuk mahasiswa berkah selalu.
5. Mama dr. Sri Pamintaningsih, Papa Drh. Bambang Sukartono, Ibu Dra. Sri Rahayu, Ibu Dra. Sri Rahayu, Ibu Dra. Sri Rahayu, Bapak Drs. Tri Mulyadi, Adik-adikku Lathifa Parayuha dan Anisa Parazulfa serta seluruh keluarga yang selalu memberikan perhatian, dukungan, doa, peringatan, pendidikan dan semangat tanpa lelah.

6. Bapak Yudi Prayudi, S.Si., M.Kom selaku Ketua Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
7. Masdar Zulfikar, S.Kom dan Septian Aditya, S.Kom yang telah membantu menemukan kunci dari skripsi ini pada 20 april 2012, semoga sukses selalu.
8. Bapak Tarudin atas berbagai bantuan untuk menyelesaikan tugas akhir ini.
9. Andika Kholifah Gilar Pratiwi, S.Kom dan Devi Latifah Hadi, S.Kom atas persahabatannya yang penuh cerita.
10. Citra Arfanudin,S.Kom., Rangga Adrian Akbar, S.Kom., Yopi Danis Irawan, S.Kom., Nurirwan Saputra, S.Kom serta Sugiarto, S.Kom., kawan-kawan seperjuangan yang telah banyak membantu dan bersama-sama berjuang menyelesaikan tugas akhir namun akhirnya S.Kom duluan, semog sukses selalu bapak-bapak.
11. Dedemit Jarkom 2011: Bapak Dodo, Gendro, Han si Kentang, Ibu negara Olip, Nitty S.Kom., Kiki, Kikuk, Valent, Sugi, Geli, Rifcong, Amin, Coco, Timun, Martin, Yos, dan Andit untuk segalanya yang telah diberikan dan direpotkan semoga lebih baik dari pendahulunya dan sukses selal.
12. Tukang jaringan 2006: Mas Lantip, Mas Bram, Mas Dhoni, Mbak Nita, Mas Rohmat, Mas Yopi, Mbak Imel, Mas Gopur dan para senior atas ilmu, pelajaran, pengalaman, kesempatan dan soft skill yang telah diturunkan.
13. Ayah Fahmi, Mbak Ning, Mas Fadli dan Mbak Sulis serta Mbak Azif dan seluruh karyawan atas segala bantuannya.
14. Teman seperkuliahan dan sepermainan di kelas A: Adit, Bumen, Sistha, Satyo, Iwan, Yogi, Noppa dan kawan-kawan.
15. Teman-teman INCLUDE dalam empat tahun kebersamaan yang tidak terasa.
16. Orang-orang yang tergabung dan terlibat di Unit 72 dengan segala latar belakang dan keunikannya: Bapak Ibu RT 03 Sembung Berbah Sleman, Gentong, Ferdi, Pak Ijal, Anis, Putri, Vidi, Juna, Abang Didit, Rovi serta Aan.
17. Semua pihak yang telah sengaja atau tanpa sengaja membantu menyelesaikan tugas akhir ini, yang tidak dapat disebutkan satu persatu, semoga Allah membalas segalanya dengan ridho-Nya yang berlimpah.

Tidak ada apapun di dunia yang sempurna, begitu juga laporan tugas akhir ini. Semoga laporan tugas akhir ini dapat memberi manfaat dikemudian hari bagi para pembaca dan pengembang sistem maupun pencari judul skripsi, serta mendatangkan ridha dari Allah SWT. Aamiin.

*Wassalamu'alaikum Wr. Wb.*

Yogyakarta, 6 Juni 2012

Penulis

# SARI

*Domain filtering* merupakan sebuah metode pengelolaan terhadap akses ke sebuah *domain*. *Domain filtering* mengijinkan atau menolak akses ke sebuah website dengan memeriksa *domain,* membandingkannya dengan aturan yang berlaku, kemudian memberikan kebijakan. *Domain filtering* dapat dilakukan oleh proxy server dengan menggunakan file teks. Basis data dirancang untuk mengelola data dengan cara yang lebih terstruktur sehingga dapat melakukan pencarian data lebih cepat dibandingkan dengan file teks. Kinerja *domain filtering* menggunakan basis data dan file tersebut perlu diteliti.

Penelitian ini membahas perbandingan waktu kinerja *domain filtering* proxy menggunakan basis data dan file teks dalam melakuan pencarian. Analisis dilakukan dengan metode penelitian pra-eksperimental dengan desain *one-shot case study* untuk membuktikan hipotesis. Penggunaan basis data pada *domain filtering* seharusnya memiliki waktu pencarian yang lebih singkat, namun hasil analisis pada penelitian ini tidak menunjukkan hal tersebut. Hasil penelitian ini menunjukkan kinerja pencarian file teks pada *domain filtering* membutuhkan waktu yang lebih singkat.

Kata Kunci: *Domain, Filtering,* Basis data

## TAKARIR

| | |
|---|---|
| *database* | basis data |
| *online* | terkoneksi jaringan (internet) |
| *firewall* | dinding api |
| *server* | penyedia layanan, biasanya terletak di internet |
| *domain* | representasi alamat IP dalam bahasa manusia |
| *filtering* | penyaringan |
| *keyword* | kata kunci |
| *request* | permintaan |
| *gateway* | penghubung jaringan lokal dengan internet |
| *address* | alamat |
| *cache* | memori |
| *bandwidth* | lebar jalan data |
| *malware* | aplikasi perusak |
| *load* | memuat |
| *epoch* | pencacah |
| *query* | permintaan informasi, pertanyaan |
| *error* | salah, galat |
| *worst-case* | skenario terburuk |
| *filler* | sian |

## DAFTAR ISI

# DAFTAR TABEL

## DAFTAR GAMBAR

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang Masalah

Jumlah pengguna internet dari berbagai kalangan di seluruh dunia semakin hari semakin banyak, demikian pula situs web yang terus bertambah seiring akses internet yang semakin mudah. Situs berisi informasi-informasi bermanfaat seolah menjawab semua pertanyaan dan menambah wawasan penggunanya. Namun internet juga dapat menimbulkan dampak buruk jika mengakses situs yang kurang bermanfaat seperti situs pornografi atau layanan judi *online*. Jejaring sosial pun tidak dilarang untuk diakses, namun jika akses jejaring sosial dilakukan pada saat jam pelajaran atau jam kuliah tentu dapat mengganggu konsentrasi belajar. Menurut Menteri Komunikasi dan Informatika (Kominfo), Tifatul Sembiring, penggunaan internet didominasi oleh pelajar yang belum genap 20 tahun, atau yang berada di kisaran 15-19 tahun. Untuk mengurangi dampak negatif tersebut, penyedia layanan internet perlu mengelola akses situs yang dapat dikunjungi oleh penggunanya, salah satunya dengan menggunakan *firewall*.

*Firewall* adalah sebuah alat atau sistem yang mampu mengelola lalu lintas antar jaringan dengan berbagai tipe keamanan (Wack, Cutler, & Pole, 2002). Fungsi utama *firewall* adalah menyaring komponen dalam jaringan, salah satunya alamat *domain*. *Firewall* dapat berupa perangkat keras, perangkat lunak maupun fungsi dari aplikasi lain seperti pada web proxy *server*. *Firewall* pada web proxy *server* bekerja dengan melihat alamat *domain* situs yang diakses klien. Fungsi *firewall* yang bekerja dengan melihat alamat *domain* ini diberi nama *domain filtering*. *Domain filtering* bekerja menggunakan daftar aturan berisi alamat *domain* yang diletakkan pada sebuah file teks. Daftar alamat tersebut akan dibandingkan dengan aturan yang berlaku, untuk selanjutnya dikenai kebijakan dari *firewall*.

Kelemahan dari *domain filtering* terletak pada pengelolaan daftar alamat yang disimpan dalam file teks. File teks melakukan pencarian data dengan

metode pencarian linear atau sekuensial. Jika data yang digunakan terlalu banyak, akan dibutuhkan waktu lama untuk pencarian pada file teks. Basis data merupakan perkembangan teknologi yang telah dioptimasi secara terstruktur, sehingga dapat melakukan pencarian dengan cepat (Rieffel, 2008). Pembuktian kinerja *domain filtering* dalam melakukan pencarian pada data yang besar dibutuhkan untuk mengetahui perbandingan kinerja basis data dan file pada *domain filtering*.

## 1.2 Rumusan Masalah

Dari uraian pada bagian latar belakang masalah terlihat bahwa masalah pada penelitian ini adalah:

1. Bagaimana kinerja *firewall* pada proxy menggunakan basis data.
2. Bagaimana perbandingan kinerja *firewall* pada proxy jika menggunakan file teks biasa dengan menggunakan basis data.
3. Bagaimana perbandingan kinerja *firewall* pada proxy jika menggunakan basis data dengan data acak dan terurut.

## 1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. Penelitian ini menggunakan data sejumlah 1.000 hingga 1.000.000
2. Penelitian ini membandingkan kinerja basis data dengan data acak (*worst-case* file) dan terurut.
3. Penelitian ini membandingkan kinerja antara *domain filtering* menggunakan file teks dan basis data.
4. Penelitian ini tidak mempertimbangkan spesifikasi perangkat keras, waktu dan permasalahan yang mungkin terjadi antara klien dan *server*.

## 1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah mengetahui kinerja *domain filtering* sebagai *firewall* pada proxy dengan menggunakan daftar aturan berupa file teks dan basis data, serta mengetahui kinerja *domain filtering* menggunakan basis data berisi data *domain* acak dan terurut.

## 1.5 Manfaat Penelitian

Hasil penelitian ini diharapkan dapat menjadi informasi bagi kinerja proxy *server* menggunakan basis data. Manfaat dari penelitian ini untuk mengetahui perbedaan kinerja *domain filtering* pada web proxy *server* dengan menggunakan basis data dan file teks biasa, sehingga pembatasan menggunakan *domain filtering* dapat dimaksimalkan kinerjanya.

## 1.6 Hipotesis

1. Kinerja *domain filtering* dengan jumlah *domain* 1.000-1.000.000 menggunakan basis data lebih baik daripada menggunakan file teks karena basis data telah dioptimasi secara lebih terstruktur sehingga dapat melakukan pencarian lebih cepat
2. Kinerja *domain filtering* menggunakan basis data dengan 1.000-1.000.000 data *domain* terurut lebih baik dari data acak (*worst-case* file) karena pencarian pada data terurut lebih mudah dilakukan.

## 1.7 Metodologi Penelitian

### 1.7.1 Studi Pustaka

Studi pustaka digunakan untuk mendapatkan teori dan informasi yang diperlukan terkait dengan penelitian *domain filtering* menggunakan basis data ini. Literatur yang digunakan berupa buku, jurnal maupun artikel dari situs-situs di internet yang berkaitan dengan penelitian ini.

### 1.7.2 Implementasi Sistem dan Penelitian

Metodologi yang digunakan dalam penelitian ini adalah kuantitatif eksperimen. Langkah-langkah dalam penelitian ekperimen antara lain sebagai berikut:

1. Analisis Masalah

   Pada tahap ini dilakukan pengamatan terhadap kinerja *domain filtering* pada proxy yang masih menggunakan file teks dan yang telah menggunakan basis data, untuk selnjutnya diambil menjadi rumusan masalah.

2. Pembangunan Sistem

   Pada tahap ini disiapkan kebutuhan komputer untuk penelitian, baik perangkat keras maupun perangkat lunak seperti proxy *server*, aplikasi manajemen basis data, modul penghubung proxy dengan basis data, serta perangkat lunak pendukung lainnya.

3. Pengambilan Data Sistem

   Pada tahap ini proxy *server* yang telah terhubung dengan basis data dan file akan diambil datanya yang berupa waktu kinerja dalam melakukan pencarian. Sebelumnya, file dan basis data sebagai variabel yang akan dibandingkan telah diisi dengan *domain* acak dan terurut.

4. Analisis Data

   Pada tahap ini dilakukan perbandingan analisis data pada *domain filtering* menggunakan basis data dan file menggunakan data acak (*worst-case* file), dan analisis data pada *domain filtering* menggunakan basis data dengan data acak dan terurut.

5. Pengambilan Kesimpulan

   Pada tahap ini akan diambil kesimpulan yang berdasarkan perbandingan hasil analisis dari sistem yang telah dibangun.

**1.8 Sistematika Penulisan**

BAB I PENDAHULUAN

Membahas latar belakang masalah, batasan masalah, rumusan masalah, tujuan dan manfaat penelitian, serta hipotesis dan metodologi penelitian yang diangkat menjadi materi penulisan laporan tugas akhir ini.

BAB II LANDASAN TEORI

Membahas dasar-dasar teori dalam perancangan dan pembangunan proxy *server* dengan file teks dan basis data.

BAB III METODOLOGI

Memuat uraian tentang metode penelitian, variabel dan parameter, pengambilan data, sistem yang digunakan serta analisis data yang berupa statistik yang digunakan beserta tujuannya untuk membuktikan hipotesis dan menjawab rumusan masalah.

BAB IV HASIL DAN PEMBAHASAN

Memuat penjelasan mulai dari tahap instalasi perangkat dan konfigurasi aplikasi, pengambilan data sistem, hingga menganalisis data yang di dapat.

BAB V SIMPULAN DAN SARAN

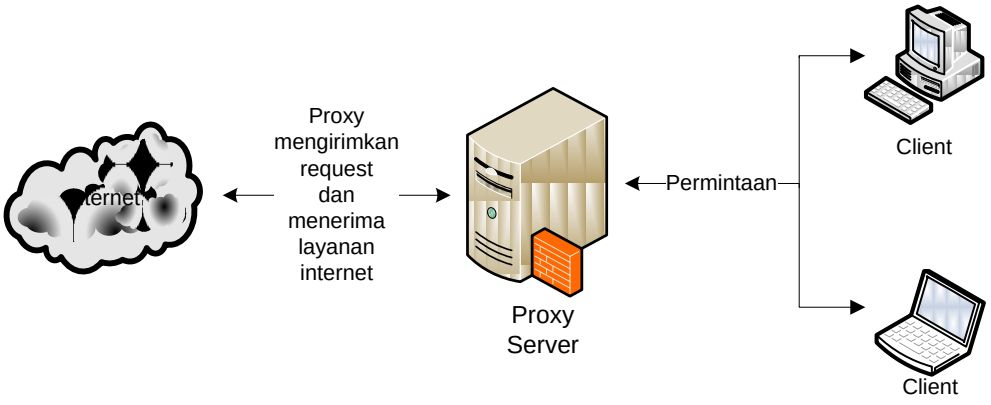
Memuat kesimpulan dari seluruh rangkaian proses penelitian, baik pada tahap perancangan maupun pembahasan, juga terdapat saran untuk pengembangan dari penelitian ini.

# BAB II
# LANDASAN TEORI

## 2.1 Proxy *Server*

Proxy *server* adalah sebuah aplikasi yang berfungsi menjadi perantara antara klien dengan *server*, sehingga proxy *server* selalu terletak diantara klien dan *server*. Dalam hal ini, *server* terletak pada jaringan luar (internet), dan klien berada pada jaringan lokal. Proxy *server* berfungsi mengelola permintaan klien dan menerima layanan dari internet sebelum diterima klien. Klien akan menganggap permintaan ditangani langsung oleh *server* di internet, bukan oleh sebuah proxy *server*. *Server* penerima permintaan dari proxy di internet juga akan melihat permintaan tersebut seolah-olah datang langsung dari klien, bukan dari sebuah proxy *server*. Alamat IP proxy akan menjadi alias bagi alamat IP privat pada klien dibawahnya.

Skema jaringan proxy *server* dijelaskan seperti pada gambar dibawah ini:




**Gambar 2. 1** Letak Proxy Server dalam Jaringan

Proxy *server* memiliki banyak tipe. Web proxy *server* adalah tipe yang paling sering digunakan, sehingga web proxy *server* dapat disebut dengan proxy *server* saja. Klien dapat menggunakan layanan proxy dengan menambahkan alamat proxy *server* secara manual pada pengaturan jaringan. Untuk menggunakan proxy secara otomatis, proxy harus diatur agar menjadi transparan proxy. Transparan proxy akan mengarahkan klien dalam jaringan untuk otomatis menggunakan layanan proxy sebelum menuju *server* asli di internet.

Proxy *server* memiliki tiga fungsi utama sebagai berikut:

1. *Gateway*

   Tiap komputer membutuhkan sebuah alamat IP (Internet Protokol) atau IP *address* yang bersifat publik agar dapat mengakses internet, sementara IP publik yang tersedia makin menipis. *Gateway* membutuhkan satu IP publik untuk menjembatani jaringan lokal agar dapat mengakses jaringan luar. Fungsi *gateway* menjembatani jaringan lokal dengan jaringan luar. *Gateway* memerlukan dua alamat IP yaitu lokal dan publik. IP lokal digunakan untuk berhubungan dengan jaringan lokal dan IP publik digunakan untuk koneksi ke jaringan luar atau internet. Fungsi proxy sebagai *gateway* menghubungkan klien-klien yang berada di jaringan lokal ke internet tanpa harus memiliki IP publik di tiap komputer.

2. *Cache*

   *Cache* pada proxy berfungsi untuk menyimpan objek-objek yang pernah diakses oleh klien dari internet. Permintaan-permintaan untuk objek yang sama akan memakan waktu dan *bandwidth*. Dengan fungsi *cache*, permintaan untuk objek yang sama cukup diambil dari *cache* proxy, tidak perlu mengambilnya kembali dari internet. Klien akan menganggap objek yang diterima seolah-olah berasal dari *server* di internet. Layanan internet akan terasa lebih cepat dengan fungsi *cache* selain juga dapat mengoptimalkan penggunaan *bandwidth*.

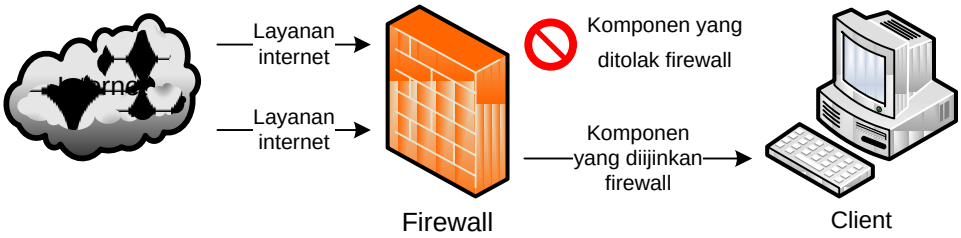
3. *Firewall*

   Proxy *server* mampu meningkatkan performa layanan dan melindungi klien dari bahaya yang berasal dari internet dengan konfigurasi tepat layaknya *firewall*. *Firewall* adalah alat atau sistem yang dapat mengelola keamanan pada jaringan. *Firewall* dapat mengamankan jaringan lokal yang terhubung internet. *Firewall* pada umumnya bekerja pada layer jaringan kedua, ketiga atau keempat. Namun *firewall* proxy bekerja pada lapisan jaringan ketujuh, sehingga memiliki lebih banyak fungsi dan kontrol terhadap akses jaringan. Web proxy *server* mampu mengecek URL dari permintaan klien ke internet dengan memeriksa pesan HTTP, GET, POST.

Nama *domain* dari portal-portal web yang diperbolehkan atau dilarang dapat didefinisikan dalam daftar kontrol akses *firewall* proxy (Sisjarkom, 2009).

## 2.2 *Firewall*

*Firewall* adalah sebuah alat atau sistem yang berfungsi mengelola keamanan suatu jaringan dari serangan atau ancaman dari luar. Beberapa ancaman dari luar misalnya virus, *malware*, halaman situs tertentu, bahkan peretas yang akan masuk ke jaringan dapat terhalang oleh *firewall*. *Firewall* terletak diantara klien dan *server* (internet) seperti gambar berikut:




**Gambar 2. 2** Skema Kinerja *Domain*

Fungsi utama *firewall* adalah mengontrol dan mengatur lalu lintas jaringan. Untuk mengelola akses jaringan, *firewall* memiliki aturan dan kebijakan yang digunakan. Aturan dalam *firewall* berisi daftar komponen-komponen paket yang akan disaring. Sedangkan kebijakan yang dimiliki *firewall* bertugas mengijinkan atau menolak paket. Paket yang melewati *firewall* akan diperiksa komponennya kemudian dibandingkan dengan daftar aturan yang dimiliki *firewall*. Jika paket tersebut memiliki kriteria yang sama dengan daftar aturan *firewall*, maka paket tersebut akan dikenai kebijakan, apakah akan diteruskan atau ditolak.

Aturan *firewall* secara umum mencakup hampir seluruh komponen jaringan seperti IP *address*, port, *domain*, protokol dan sebagainya, sehingga aturan *firewall* dapat disebut juga sebagai daftar aturan atau *access control lists* (ACL). ACL biasanya didefinisikan secara manual oleh administrator jaringan. ACL dalam *firewall* bersifat umum. Namun ACL dapat berupa

pengecualian aturan, yang disebut aturan khusus. Aturan khusus akan menambah keamanan jaringan, selain menambah rumit aturan *firewall*.

*Firewall* memiliki banyak jenis dan bekerja pada beberapa layer jaringan yang berbeda. *Firewall* yang bekerja pada layer jaringan kedua dapat mengelola paket berdasar tipe jaringan, misalnya ethernet. Paket *filtering firewall* bekerja pada layer ketiga (network). Paket *filtering firewall* dapat mengelola paket berdasarkan alamat IP. *Firewall* pada layer ke-empat dapat mengelola paket berdasarkan port, misalnya port 80 untuk HTTP, port 53 untuk DNS, port 21 untuk FTP dan sebagainya. *Firewall* yang terdapat pada proxy termasuk *application-level gateway firewall* yang bekerja pada layer ketujuh (layer aplikasi). *Firewall* pada layer ketujuh mampu mengelola jaringan secara lebih menyeluruh, karena *firewall* tersebut mampu membaca lebih banyak komponen jaringan seperti *domain*, waktu akses, IP user, autentikasi user, dan sebagainya (Wack, Cutler, & Pole, 2002).

### 2.2.1 Paket *Filtering*

Fungsi paling sederhana dari *firewall* adalah melakukan *filter* paket data atau disebut juga dengan paket *filtering firewall*. Paket data adalah komponen berisi data atau informasi digital yang ditransmisikan melalui jaringan komputer. Tiap paket data memiliki header yang berisi informasi penting seperti IP *address*, nama protokol, nomor port, alamat *domain*, ukuran paket dan sebagainya. Paket *filtering firewall* bekerja pada layer ketiga model jaringan OSI.

Paket *filtering firewall* bekerja dengan memeriksa atau menyaring header dari paket data yang masuk dalam jaringan, membandingkannya dengan aturan dalam ACL, kemudian memutuskan untuk menolak atau mengijinkan paket tersebut. Jika paket termasuk daftar yang ditolak maka paket akan dihentikan, jika termasuk daftar yang diijinkan paket akan diteruskan ke tujuan. Semakin banyak aturan yang diberlakukan pada *firewall*, ukuran ACL akan semakin besar. Paket *filtering firewall* banyak diimplementasikan pada perangkat jaringan router (Sisjarkom, 2009). Iptables pada linux merupakan salah satu contoh aplikasi paket *filtering firewall*.

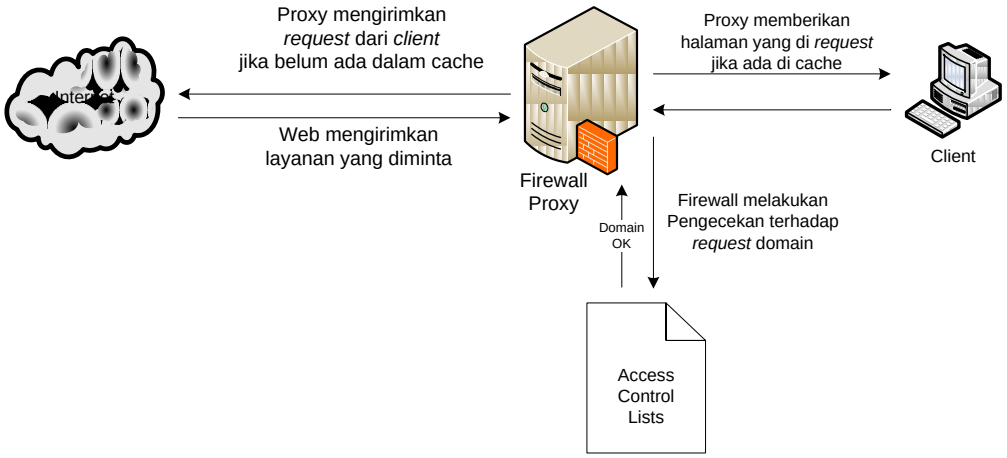
Paket *filtering firewall* tidak memiliki konfigurasi tertentu mengenai pengguna seperti autentikasi user, waktu akses, dan sebagainya. Aturan pada paket *filtering firewall* hanya mengijinkan atau menolak paket yang lewat berdasarkan header paket datanya, karena itu paket *filtering firewall* mampu bekerja cepat dan tidak memerlukan spesifikasi *hardware* yang tinggi. Kelemahan paket *filtering firewall* ada pada daftar aturan yang akan semakin banyak dan kompleks seiring bertambahnya aturan yang diberlakukan.

### 2.2.2 *Domain Filtering*

*Domain* merupakan nama yang digunakan untuk representasi alamat IP pada jaringan dalam bentuk bahasa manusia. Selain beberapa servis seperti FTP, *domain* banyak digunakan untuk penamaan sebuah situs. *Domain* itu sendiri dikelola oleh DNS (*Domain Name System*) *Server* yang berfungsi menerjemahkan nama *domain* ke alamat IP dan sebaliknya. DNS *server* bekerja pada layer ketujuh (aplikasi) model jaringan OSI, sama seperti proxy. Proxy memiliki fungsi sebagai *firewall* yang bekerja dengan prinsip *domain filtering*.

*Domain filtering* merupakan penyaringan alamat web atau *domain* secara keseluruhan maupun sebagian. *Domain filtering* bekerja dengan memeriksa URL yang diminta oleh klien, membandingkannya dengan aturan yang berlaku kemudian memutuskan untuk mengijinkan atau menolak akses. *Domain filtering* berbeda dengan *firewall* biasa, karena *firewall* biasa hanya dapat membaca header paket data, bukan *domain*. Proxy bekerja pada layer ketujuh jaringan komputer, lebih tinggi dari *firewall* biasa maupun router dengan fitur paket *filtering*. Hal ini membuat *firewall* pada proxy memiliki lebih banyak fungsi dan kontrol terhadap akses jaringan sehingga dapat melakukan *domain filtering*. Web proxy *server* dapat melihat URL dari halaman web yang diminta klien dengan memeriksa pesan HTTP, GET dan POST. Dengan kemampuan tersebut *firewall* pada proxy dapat mengatur akses ke *domain* tertentu. Kebijakan dan aturan pada *domain filtering* sama dengan *firewall* pada umumnya, yaitu mengijinkan atau menolak akses.

Seperti *firewall* lain, proxy *firewall* juga memiliki *access control lists* (ACL) mengatur *domain filtering*. ACL atau daftar aturan tersebut dapat disimpan dalam sebuah file agar dapat menyimpan aturan lebih banyak.
Berikut ini gambaran kinerja *firewall* pada proxy:




**Gambar 2. 3** Ilustrasi kerja *domain filtering* pada proxy

Berbeda dengan paket *filtering firewall* yang tidak memerlukan spesifikasi hardware tinggi, *domain filtering* memiliki kompleksitas kinerja tinggi sehingga membutuhkan sumber daya yang besar. *Domain filtering* mampu mengelola autentikasi *user* dan kontrol akses, sehingga akses jaringan lebih aman. *Domain filtering* bekerja berdampingan dengan proxy sehingga diperlukan aplikasi pendukung serta spesifikasi sumber daya yang tinggi agar proxy dapat berjalan dengan baik.

Salah satu proxy server yang diunggulkan adalah Squid. Selain dapat bekerja pada berbagai platform komputer, Squid proxy server termasuk perangkat lunak yang dikenal stabil dan berlisensi GPL, sama seperti linux. *Firewall* proxy pada Squid bekerja menggunakan file sebagai daftar aturan. File tersebut berisi alamat *domain* yang akan diatur oleh *firewall*. pendefinisian file tersebut dilakukan dengan dua cara:

1. dstdomain

   dstdomain hanya akan mengatur alamat *domain* yang sama persis dengan nama *domain* yang ada dalam ACL. Berbeda satu karakter akan membuat kata tersebut lolos seleksi.

2. url_regex

   url_regex akan menyeleksi alamat *domain* dengan membandingkannya dengan kata kunci sebagai aturan yang disimpan pada sebuah file. Sehingga file ACL yang menggunakan url_regex berisi kata kunci dari *domain* yang akan diatur, bukan *domain* itu sendiri.

Daftar aturan dalam file tersebut selanjutnya akan dikenai kebijakan yang berlaku: diijinkan atau ditolak. Akses yang diijinkan, akan diteruskan proxy ke internet sehingga permintaan klien dapat dipenuhi, sedangkan akses yang ditolak akan diberikan pemberitahuan pada klien bahwa akses ditolak

## 2.3 Basis Data

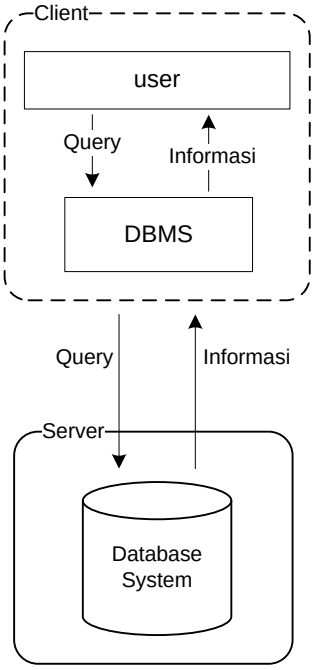
*Database* atau basis data dapat didefinisikan sebagai kumpulan data yang terorganisir dalam satu atau sejumlah tabel yang dapat saling terkait dan berfungsi sebagai penyedia informasi bagi pengguna atau user (Sirkel, 2008). Basisdata memiliki tabel dan kolom berisi atribut yang berfungsi merepresentasikan entitas atau benda di dunia nyata lengkap dengan komponen penyusunnya. Basisdata diciptakan untuk menggantikan fungsi file sebagai alat penyimpanan data. Basisdata mengelola data dengan cara yang lebih terstruktur dari file sistem.

Basisdata mampu mencari dan mengurutkan data dalam waktu yang lebih singkat dari file sistem. Basisdata juga melakukan pengelolaan data yang meliputi tambah data, ubah data, hapus data, mengurutkan data, dan sebagainya. Pencarian data yang disimpan menggunakan file akan membutuhkan waktu lama karena harus mencari data didalam file satu per satu. Pengambilan data dengan kondisi tertentu akan lebih mudah dan cepat jika menggunakan basisdata. Basisdata mampu melakukan pencarian dan pengurutan dalam waktu yang relatif lebih cepat dari file sistem.

Basis data merupakan hasil implementasi dari sebuah aplikasi pengelola basis data yang disebut DBMS (*Database* Management System). DBMS menggunakan bahasa SQL (*Structured Query Language*) untuk membangun basisdata. *Query* adalah permintaan informasi dari basisdata, dan *query*

*language* adalah bahasa khusus yang mudah digunakan yang memungkinkan komputer menjawab *query* (Nugroho, 2011). Beberapa aplikasi DBMS yang dapat digunakan untuk membangun basisdata antara lain: MySQL, Oracle, Postgree, Microsoft Access. User mendapatkan informasi dari basisdata dengan memasukkan *query* ke dalam DBMS.

Basisdata memerlukan *query* untuk bekerja. Mulai dari pembuatan tabel, pengisian data hingga pengelolaan data. *Query* merupakan cakupan bahasa SQL yang akan diproses oleh DBMS. Komponen pembangun basisdata dapat dilihat pada gambar dibawah ini:




**Gambar 2. 4** Komponen pembangun basis data

# BAB III
# METODOLOGI

## 3.1 Metode Penelitian

Penelitian ini menggunakan pendekatan kuantitatif berjenis eksperimen untuk melakukan pengambilan data dan memanipulasi atau mengontrol sistem. Penelitian eksperimen adalah suatu penelitian yang berusaha mencari pengaruh variabel tertentu terhadap variabel yang lain dalam kondisi yang terkontrol secara ketat (Abidin, 2011). Misalnya suatu penelitian dimaksudkan untuk membuktikan pengaruh suatu perlakuan pada urutan dalam basis data terhadap lama waktu pencarian atau menguji hipotesis tentang ada-tidaknya pengaruh tindakan itu jika dibandingkan dengan tindakan lain.

Desain penelitian yang digunakan adalah pra-eksperimental dimana penelitian tersebut hanya melibatkan satu variabel bebas untuk dimanipulasi atau dikontrol. Penelitian ini menggunakan media eksternal *domain filtering* proxy yaitu basis data dan file. Pra eksperimental desain bukan penelitian berjenis eksperimen yang sesungguhnya karena masih terdapat variabel luar yang ikut berpengaruh terhadap terbentuknya variabel terikat, hal ini dapat terjadi karena tidak adanya variabel kontrol dan sampel tidak diambil secara acak (Fataruba, 2012).

Pra-eksperimental desain memiliki beberapa metode, salah satunya metode *one-shot case study* yang digunakan untuk penelitian ini. Metode tersebut digunakan karena sesuai dengan hakekat penelitian ini dimana menurut Subianto (2007) dalam *one-shot case study* terdapat suatu kelompok diberi *treatment* atau perlakuan, selanjutnya diobservasi hasilnya. *Treatment* atau perlakuan pada metode statistik disebut variabel independen (bebas) yang merupakan tindakan manipulasi dan kontrol agar menjadi sesuai dengan yang dibutuhkan penelitian, sedangkan hasil dari perlakuan yang diberikan pada variabel independen disebut variabel dependen.

Jenis metode *one-shot case study* dimaksudkan untuk menunjukkan kekuatan pengukuran dan nilai ilmiah suatu desain penelitian seperti yang ditunjukkan pada bagan dari paradigma *one-shot case study* berikut ini:

| X | O |
|---|---|
| Perlakuan terhadap variabel independen.<br>Misalnya: Basis data diberikan data terurut dan *worst-case* dari file. | Pengamatan atau pengukuran terhadap variabel dependen.<br>Misalnya: Waktu pencarian pada basis data |

Keterangan:

X: kelompok yang akan diberi stimulus dalam eksperimen (*treatment*)

O: kejadian pengukuran atau pengamatan (observasi)

Bagan tersebut dapat dibaca sebagai berikut: terdapat suatu kelompok yang diberi perlakuan dan selanjutnya diobservasi hasilnya. Contoh pada bagan dapat dibaca sebagai pengaruh urutan dalam pengisian basis data (X) terhadap waktu pencarian (O). Dalam penelitian eksperimen ini subjek disajikan dengan beberapa jenis perlakuan lalu diukur hasilnya.

### 3.1.1 Variabel dan Parameter Penelitian

Terdapat dua jenis variabel yang digunakan pada penelitian ini:

1. Variabel bebas (independen)

   Variabel bebas adalah *treatment* (X) atau perlakuan yang diberikan untuk memanipulasi atau mengontrol dan membuat suatu kondisi atas dasar pertimbangan ilmiah agar menjadi sesuai dengan kondisi yang dibutuhkan penelitian. Manipulasi variabel bebas merupakan karakteristik penelitian eksperimen. Variabel bebas yang diberikan kepada *domain filtering* pada penelitian ini ada tiga:

   a. File dengan data acak (worst-case file)
   b. Basis data dengan data acak (worst-case file)
   c. Basis data dengan data terurut

   Kondisi acak (worst-case file) yang dimaksudkan pada penelitian ini adalah peletakan *domain* yang akan diakses klien pada posisi paling akhir. File akan melakukan pencarian secara sekuensial atau berurutan dari awal hingga akhir atau hingga data ditemukan. Jika *domain* yang

diakses diletakkan di posisi terakhir, file yang digunakan *domain filtering* harus melakukanpencarian dari awal hingga akhir agar dapat menemukan *domain* dan menolak aksesnya, kondisi tersebut pada penelitian ini disebut dengan *worst-case* file yang selanjutny disebut data acak untuk lebih memudahkan.

2. Variabel terikat (dependen)

   Variabel terikat adalah hasil observasi (O) atau keluaran yang muncul akibat dari perlakuan atau *treatment* yang diberikan kepada variabel bebas. Variabel terikat atau keluaran pada penelitian ini adalah waktu pencarian yang dilakukan oleh basis data dan file sebagai media eksternal dari *domain filtering* proxy. Pengukuran waktu pencarian tersebut dilakukan menggunakan perangkat lunak pemonitor jaringan. Kendala yang ditemui pada pengukuran ini terletak pada kebijakan untuk menolak akses *domain*. Sebagain besar perangkat lunak yang ada menganggap akses situs yang ditolak tidak membutuhkan waktu, sehingga waktu pencarian dianggap 0,00 detik.

   Alternatifnya, waktu pencarian *domain* diambil menggunakan waktu yang dibutuhkan klien untuk mengakses sebuah *domain* yang telah ditentukan. Waktu yang digunakan klien mulai dari melakukan permintaan akses, hingga pesan *error* diterima dianggap sama dengan lama waktu yang dibutuhkan basis data maupun file sebagai media eksternal *domain filtering* proxy untuk mencari sebuah *domain*. Sehingga pengambilan data dilakukan dengan menggunakan perangkat pemonitor jaringan yang diletakkan pada sisi klien.

Faktor-faktor yang berpengaruh pada waktu hasil penelitian seperti kecepatan transmisi paket dari klien ke *server* maupun sebaliknya, spesifikasi hardware klien dan *server* dan sebagainya dianggap sebagai variabel esktrane yang bernilai konstan atau tetap dan sama pada setiap pengambilan data. Variabel ekstrane atau *extraneous variable* adalah variabel yang berada diluar kekuasaan eksperimen untuk dikontrol maupun dikendalikan. Dalam setiap eksperimen, hasil yang berbeda disebabkan oleh

variabel eksperimental (misalnya variabel bebas) dan sebagaian lagi karena pengaruh variabel ekstrane (Nursyahidah, 2011).

Parameter yang digunakan pada penelitian ini adalah jumlah *domain* yang diberikan kepada basis data dan file. *Domain* yang diperlukan untuk penelitian ini mulai dari seribu hingga satu juta *domain*. Jumlah *domain* yang besar dimaksudkan untuk membandingkan waktu kinerja *domain filtering* menggunakan basis data dengan file. Semakin banyak data yang terdapat pada file, maka semakin lama pencarian yang dilakukan. Basis data telah menggunakan algoritma pencarian yang lebih baik dari file sehingga tidak memiliki masalah dengan banyaknya data yang harus dicari.

Kebutuhan jumlah *domain* yang harus tersedia untuk penelitian ini menjadi kendala tersendiri karena banyaknya jumlah, sehingga sebagian besar *domain* digantikan dengan *filler*. *Filler* merupakan karakter atau huruf yang disusun secara acak hingga membentuk suatu kata yang menjadi representasi *domain*. Parameter pada penelitian ini seluruhnya menggunakan *filler* kecuali satu *domain* asli yang akan diakses klien diletakkan pada posisi terakhir untuk memenuhi kondisi *worst-case* file. *Domain* tersebut dikenai kebijakan untuk ditolak.

### 3.1.2 Pengambilan Data

Pengambilan data pada sistem *domain filtering* dilakukan pada sisi klien. Data diambil dari selisih waktu permintaan dikirim hingga pesan *error* diterima klien. Pengambilan data dilakukan sepuluh kali dengan jumlah *domain* yang terdapat pada ACL proxy dimulai dari 1.000 hingga 1.000.000. *Domain-domain* sejumlah satu juta tersebut tidak diuji sekaligus dari seribu berkelanjutan hingga satu juta, namun dibagi menjadi tiga tingkatan:

1. 1.000-10.000 *domain.*
2. 10.000-100.000 *domain.*
3. 100.000-1.000.000 *domain.*

*Domain-domain* tersebut dikenai kebijakan dilarang akses, dan diletakkan pada basis data dan file sebagai media eksternal *domain filtering* proxy.

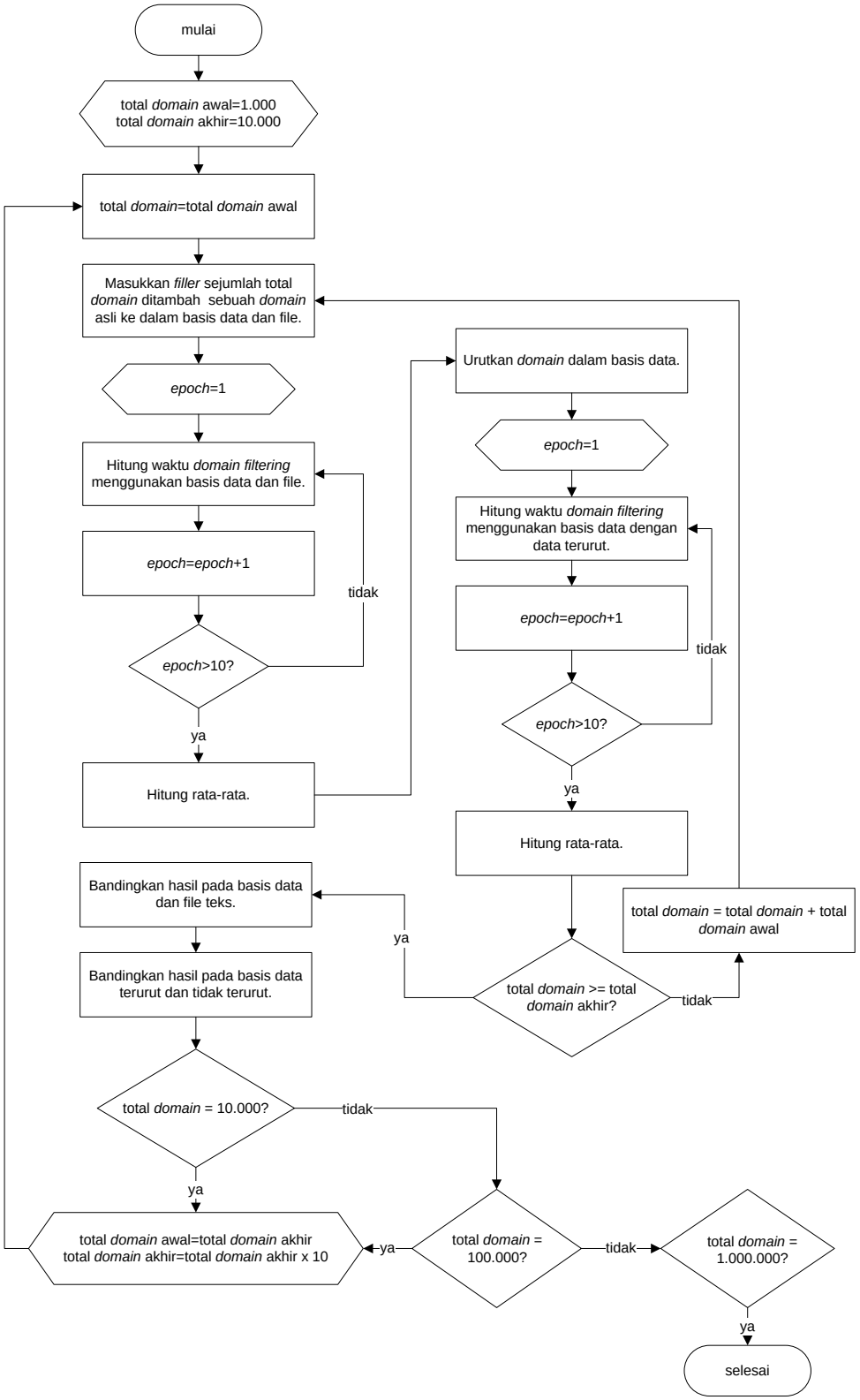
Alur pengambilan data dimulai dari pengisian *filler* ke dalam file dan basis data dalam jumlah yang ditentukan, misalnya seribu. Kemudian ditambahkan *domain* asli yang akan diakses oleh klien pada posisi terakhir, sehingga memenuhi kriteria *worst-case* file sebagai data acak. Kemudian setelah diyakini bahwa basis data dan file berisi *domain* dalam posisi yang diinginkan, alat pengukurnya dinyalakan. Selanjutnya klien akan melakukan akses *domain* dan alat pemonitor jaringan akan mengangkap paket berisi waktu permintaan klien dan pesan *error* yang dikirimkan server. Waktu tersebut kemudian dicatat dan dihitung selisihnya. Selisih waktu tersebut akan digunakan sebagai data penelitian. Pengambilan data dilakukan sebanyak sepuluh kali untuk tiap-tiap jumlah *domain*. Setelah mencapai sepuluh, data dirata-rata dan disimpan untuk selanjutnya digunakan sebagai data analisis.

Perlakuan yang kedua adalah pengurutan *domain* yang terdapat pada basis data. Setelah diurutkan, pengambilan data kembali dilakukan dengan langkah-langkah yang sama seperti alur pengujian *worst-case* file pada basis data dan file diatas, hingga didapat rata-rata dari sepuluh data.

Setelah seluruh rata-rata dari data waktu kinerja *domain filtering* dalam satu jumlah *domain* didapatkan, keadaan basis data dan file dikembalikan pada kondisi awal dengan *domain* pada basis data dan file teks dalam posisi acak (*worst-case* file) tanpa *domain* asli didalamnya, sehingga seluruh isinya adalah *filler*. Isi basis data dan file tersebut kemudian ditambah dengan *filler* sejumlah awal, sehingga menjadi tingkatan jumlah yang harus diukur berikutnya. Jumlah baru ini masih berisi *filler* seluruhnya, sehingga perlu ditambahkan *domain* asli pada akhir *filler* untuk selanjutnya kembali dilakukan pengambilan data.

Setelah pengambilan data mencapai jumlah maksimum pada satu tingkatan jumlah, misalnya 10.000, selanjutnya akan dilakukan perbandingan waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan data acak, dan perbandingan waktu kinerja *domain filtering* menggunakan basis data dengan data acak (*worst-case* file) dan data terurut. Selanjutnya, pengujian dilakukan pada tingkatan jumlah berikutnya.

Berikut ini *flowchart* pengambilan data pada penelitian ini:



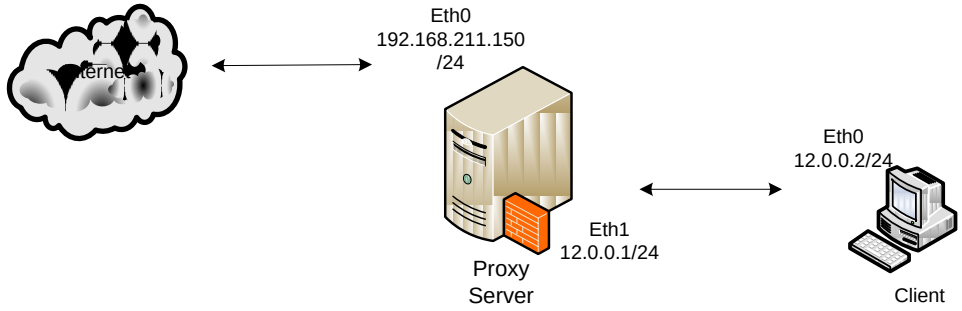

**Gambar 3. 1** *Flowchart* Pengambilan Data

### 3.1.3 Sistem yang Digunakan

Sistem yang digunakan untuk pengambilan data pada penelitian ini menggunakan proxy *server* sebagai *gateway* yang akan menghubungkan klien dengan jaringan luar (internet). Proxy *server* yang bertindak sebagai *gateway* menggunakan dua antarmuka jaringan:

1. Eth0 menggunakan ip dinamis (ipv4) yang diperoleh dari *host* menggunakan jaringan NAT, digunakan untuk berhubungan dengan jaringan luar (internet).
2. Eth1 menggunakan ip statik (ipv4) yang diberikan secara manual, digunakan untuk berhubungan dengan klien.

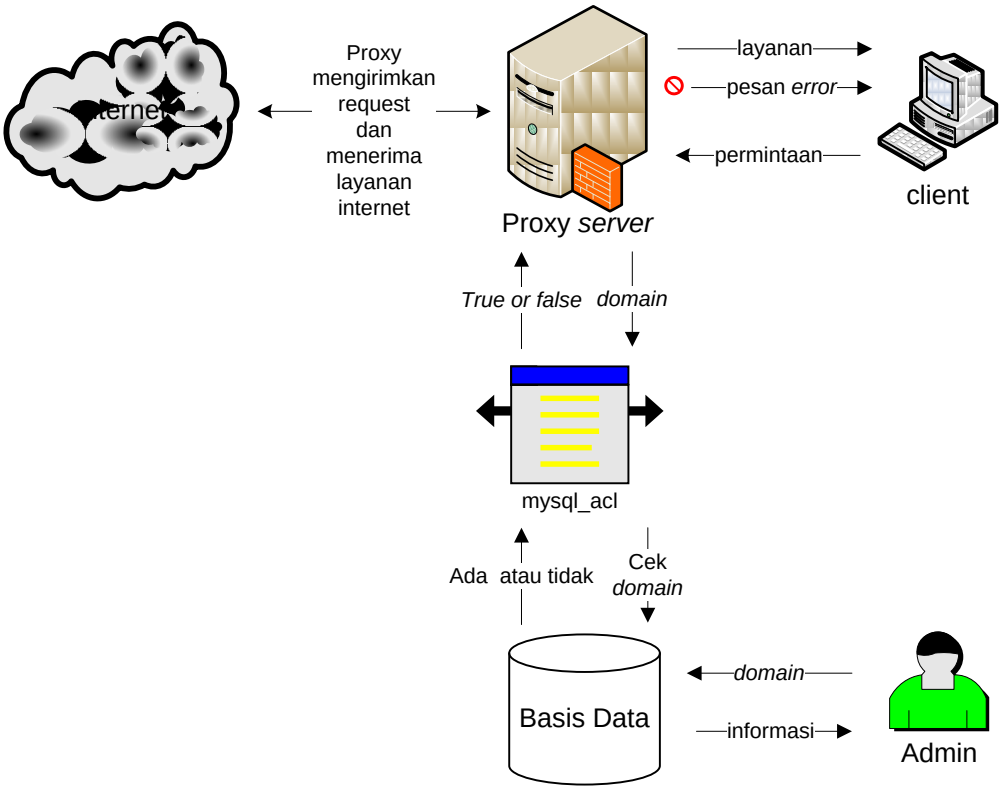
Komputer klien menggunakan satu antarmuka jaringan yang menggunakan IP statik untuk berhubungan dengan eth1 komputer *server*.
Berikut ini desain jaringan yang akan digunakan:




**Gambar 3. 2** Topologi jaringan

Pada proxy server, nama *domain* yang ditolak diletakkan kedalam aturan. Aturan pada proxy dapat berupa media eksternal seperti basis data atau file. Aturan tersebut akan diakses oleh proxy setiap kali permintaan akses datang dari klien. *Domain* yang diminta klien akan dibandingkan dengan isi aturan *firewall* proxy yang terdapat pada basis data maupun file. Jika *domain* yang diminta ada dalam daftar aturan yang ditolak, maka akses akan dihentikan, kemudian proxy akan mengirimkan pesan kepada klien bahwa akses ditolak (*error*). Jika *domain* yang diminta klien termasuk dalam aturan yang diperbolehkan, proxy akan meneruskan permintaan ke jaringan luar (internet) hingga ke tujuan.

Alur data akses pada penelitian ini lebih banyak pada jaringan lokal (klien-*server*), tidak sampai ke jaringan luar (internet) karena kebijakan yang diberlakukan pada proxy *server* penelitan ini adalah penolakan pada *domain*. Berikut ini desain alur data pada sistem *domain filtering* menggunakan basis data:




**Gambar 3. 3** Desain alur data

Perangkat keras yang digunakan pada penelitian ini adalah sebuah komputer yang digunakan untuk *server* dan klien dalam dua mesin virtual yang berbeda. Spesifikasi perangkat keras fisik yang digunakan pada penelitian ini sebagai berikut:

1. Processor 1.06 GHz
2. Harddisk 20 GB
3. RAM 4GB
4. Sebuah LAN card

Perangkat keras fisik diatas sebagian digunakan untuk perangkat keras *server* virtual, dengan spesifikasi sebagai berikut:

1. Processor 1.06 GHz
2. Harddisk 15 GB
3. RAM 1536 MB
4. Dua buah LAN card

Bagian lain dari perangkat keras diatas digunakan untuk perangkat keras klien virtual, dengan spesifikasi sebagai berikut:

1. Processor 1.06 GHz
2. Harddisk 3 GB
3. RAM 512 MB
4. Sebuah LAN card

Perangkat lunak yang digunakan untuk membangun sistem pada penelitian ini sebagai berikut:

1. VMware Workstation

   VMware merupakan aplikasi yang digunakan untuk virtualisasi perangkat keras komputer sehingga tidak diperlukan terlalu banyak perangkat keras secara fisik. Spesifikasi mesin dalam VMware disesuaikan dengan perangkat keras fisik yang ada.

2. Ubuntu 10.10

   Ubuntu 10.10 merupakan sistem operasi berbasis linux yang digunakan untuk *server* dalam penelitian ini. Linux dikembangkan dibawah lisensi GPL (*General Public Lisence*) dimana kebebasan bagi penggunanya untuk ikut mengembangkan dan mendistribusikan perangkat lunak berlisensi GPL. Bebas dalam GPL juga berarti gratis, dan bebas mendistribusikan dengan memungut biaya. Hal ini membuat linux berkembang sesuai dengan kondisi penggunanya.

3. Squid

   Squid merupakan perangkat lunak aplikasi untuk membangun web proxy *server* pada penelitian ini. Fungsi squid sebagai *domain filtering* digunakan untuk peningkatan keamanan dan optimasi jaringan, serta pembatasan akses seperti pada penelitian ini.Squid dapat berjalan pada beberapa sistem

operasi seperti linux dan windows. Squid yang digunakan pada penelitian ini squid versi linux.

4. MySQL

   MySQL merupakan perangkat lunak yang digunakan untuk manajemen basis data SQL. MySQL merupakan salah satu aplikasi manajemen sistem basis data relasional (RDBMS) yang berada dibawah lisensi *General Public Lisence* (GPL). MySQL menggunakan bahasa SQL (*Structured Query Languages*) yang merupakan bahasa umum pengoperasian basis data. SQL diproses dengan *query*. *Query* merupakan aturan yang akan diproses oleh aplikasi manajemen sistem basis data untuk mendapatkan informasi yang diinginkan.

5. Mysql_acl

   Mysql_acl merupakan modul perangkat lunak penghubung basis data MySQL dengan *firewall* pada squid proxy *server*. Mysql_acl menangkap parameter yang diberikan squid untuk dicocokkan dengan isi dalam basis data MySQL. Konfigurasi mysql_acl berhubungan langsung dengan konfigurasi squid. Sama seperti squid dan ubuntu, mysql_acl juga berlisensi GPL.

6. Windows XP

   Windows XP merupakan sistem operasi yang digunakan komputer klien untuk uji coba sistem.

7. Wireshark

   Wireshark merupakan perangkat lunak untuk memonitor jaringan komputer. Seluruh informasi dari paket data yang lewat dapat ditampilkan oleh wireshark secara lengkap, seperti waktu, asal dan tujuan paket, protokol hingga data-data pribadi yang tidak dienkripsi. Wireshark merupakan aplikasi yang bersifat *open-source*. Wireshark dapat digunakan pada sistem operasi berbasis linux maupun windows.

## 3.2 Analisis Data

Tahap ini merupakan tahap bagaimana cara analisis dari data yang telah didapat. Analisis pada penelitian ini menggunakan statistis deksriptif dan induktif serta hipotesis yang telah diberikan di awal penelitian. Data yang akan dianalisis adalah perbandingan waktu kinerja basis data dan file dalam melakukan pencarian.

1. Statistik Deskriptif

   Statistik deskriptif yaitu berkenaan dengan pengumpulan, pengolahan dan penyajian data (dalam bentuk tabel, grafik, maupun diagram). Statistik deskriptif dapat memberikan informasi berdasarkan keadaan secara umum. Statistik deskriptif ini digunakan untuk menjawab rumusan masalah yang sudah ada.

2. Statistik Induktif

   Statistik induktif (inferens) adalah metode yang berkenaan dengan analisis data yang digunakan untuk mengambil kesimpulan. Hasil analisis yang di dapat akan di uji dengan *independent sample-t test* untuk mengetahui apakah ada perbedaan waktu kinerja basis data dengan file teks dengan data acak dan basis data dengan data acak dan terurut. Statistik induktif pada penelitian ini digunakan untuk membuktikan hipotesis yang sudah ada.

# BAB IV
# HASIL DAN PEMBAHASAN

## 4.1 Implementasi Sistem Secara Umum

### 4.1.1 Instalasi Basis Data Server

1. Pada tahap ini dilakukan instalasi perangkat lunak MySQL sebagai *server* basis data. Jenis paket yang digunakan bertipe *binary*, diinstal dengan perintah apt-get

```
$ sudo apt-get install mysql-server-5.1
```

   Masukkan *password* ketika diminta.

2. Masuk ke aplikasi mysql

```
$sudo mysql -u root -p
```

   Masukkan *password* sistem dan *password* mysql.

3. Buat basis data mysql_acl

```
mysql> CREATE DATABASE mysql_acl;
```

4. Buat tabel host_acl_allow

```
mysql> CREATE TABLE mysql_acl.host_acl_allow (
       no INT( 5 ) NOT NULL AUTO_INCREMENT ,
       host VARCHAR( 50 ) NOT NULL ,
       PRIMARY KEY ( `no` ) ,
       UNIQUE (`no`)
       );
```

### 4.1.2 Instalasi Proxy Server

1. Pada tahap ini dibutuhkan paket *source code* dari squid yang didapatkan dari situs http://www.squid-cache.org/Versions/ secara gratis.
2. Instalasi paket dengan ekstraksi paket *source code*

```
$ tar -xzvf squid-3.0.STABLE25.tar.gz
$ cd squid-3.0.STABLE25/
```

3. Kompilasi hasil ekstraksi

```
$./configure --prefix=/usr/local/squid
$make
```

```
$sudo make install
```

### 4.1.3 Instalasi dan Konfigurasi Modul Tambahan

1. Instalasi mysql_acl membutuhkan program tambahan mysql.h yang didapat dari instalasi *library* libmysqld-dev

```
$sudo apt-get install libmysqld-dev
```

2. Paket mysql_acl-0.1.tar.gz didapat dari www.onebit.hu. Instalasi *source code* mysql_acl dilakukan dengan perintah berikut:
3. Ekstraksi paket

```
$tar -xzvf mysql_acl-0.1.tar.gz
```

Masuk ke dalam direktori mysql_acl-0.1

```
$cd mysql_acl-0.1/
```

Ubah baris berikut pada Makefile

```
$gedit Makefile
```

```
MAN5 = mysql_acl.conf.5

$(INSTALL) -o proxydb -g proxydb -m 755 $(BIN)
  /usr/local/squid/libexec/mysql_acl
$(INSTALL) -o proxydb -g proxydb -m 600 $(CONF)
  /usr/local/squid/etc/mysql_acl.conf
$(INSTALL) -o proxydb -g proxydb -m 600 $(CONF)
  /usr/local/squid/etc/mysql_acl.conf.default
$(INSTALL) -o proxydb -g proxydb -m 655 $(MAN8)
  /usr/local/man/man8/mysql_acl.8
$(INSTALL) -o proxydb -g proxydb -m 655 $(MAN5)
  /usr/local/man/man5/mysql_acl.conf.5
```

Ubah baris berikut pada mysql_acl.conf:

```
$gedit mysql_acl.conf
```

```
username        root
password        bithana
mysqld_socket   /var/run/mysqld/mysqld.sock
squidparams     %DST
```

```
#dstparam      %DST
sqlquery   default   eq,direct,break   select   1   from
host_acl_allow where host="%DST"
sqlquery default eq,direct  select 0
```

Membuat file *binary* dan file lain yang dibutuhkan mysql_acl:

```
$make
$sudo mkdir /usr/local/man/man8/
$sudo mkdir /usr/local/man/man5/
```

Lakukan instalasi dari file *binary* yang telah terbentuk:

```
$sudo make install
```

### 4.1.4 Konfigurasi Proxy Server

1. Ubah kepemilikan squid

```
$sudo chown –R proxydb:proxydb /usr/local/squid
```

2. Konfigurasi visible_hostname squid.conf:

```
$gedit /usr/local/squid/etc/squid.conf
```

```
external_acl_type              mysql_acl              %DST
    /usr/local/squid/libexec/mysql_acl
acl mysqlacl2 external mysql_acl default
http_access deny mysqlacl2
http_access allow all

visible_hostname bithana.TA
```

Keterangan:

Baris 1:

external_acl_type : sintak untuk mendefinisikan program acl diluar program asli squid

mysql_acl: nama program eksternal yang digunakan squid untuk acl

%DST : parameter yang akan menangkap *domain* yang diakses klien

/usr/local/squid/libexec/mysql_acl : letak program mysql_acl

Baris 2:

mysqlacl2 : nama acl yang digunakan adalah mysqlacl2

external : sintak yang mengarahkan acl pada program eksternal

mysql_acl : nama program eksternal yang telah didefinisikan

default : parameter yang akan mengirimkan *domain* yang diakses klien kepada program mysql_acl

Baris 3 : memberikan kebijakan agar acl dengan nama mysqlacl2 tidak diijinkan untuk meneruskan paket.

Baris 4 : digunakan untuk memberikan kebijakan secara umum agar acl yang tidak didefinisikan sebelumnya diperbolehkan untuk meneruskan paket data.

Baris 5 : pendefinisian pemilik dari proxy yang digunakan

3. Buat *cache* squid

```
$/usr/local/squid/sbin/squid -z
```

4. Start squid

```
$/usr/local/squid/sbin/squid start
```

### 4.1.5 Konfigurasi Basisdata server

Isi tabel host_acl_allow

```
mysql> INSERT INTO mysql_acl.host_acl_allow (host) VALUES
       ('www.facebook.com');
```

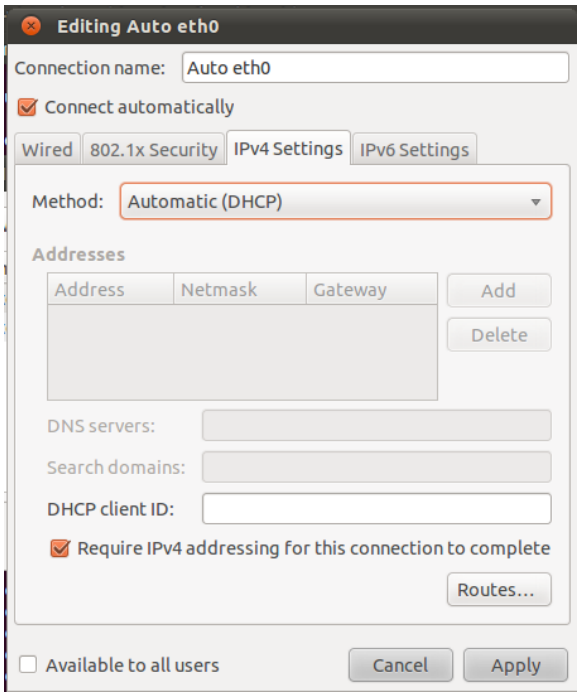
### 4.1.6 Konfigurasi Alamat IP pada Server Ubuntu

1. Konfigurasi IP eth0

   Kartu jaringan eth0 bersifat NAT dan digunakan untuk menghubungkan server dengan komputer host yang ditumpangi VMware.

   Untuk mengatur IP eth0, pada menu taskbar Ubuntu 10.10 yang digunakan sebagai server, pilih: *System →Preferences → Network Connection*

   Pilih *Auto eth0* dan klik *edit*. Pada tab *IPv4 Settings,* bagian *Method* pilih *Automatic (DHCP).*




**Gambar 4. 1** Konfigurasi alamat IP eth0

Klik *Apply*, kemudian masukkan kata sandi Ubuntu untuk autentikasi.

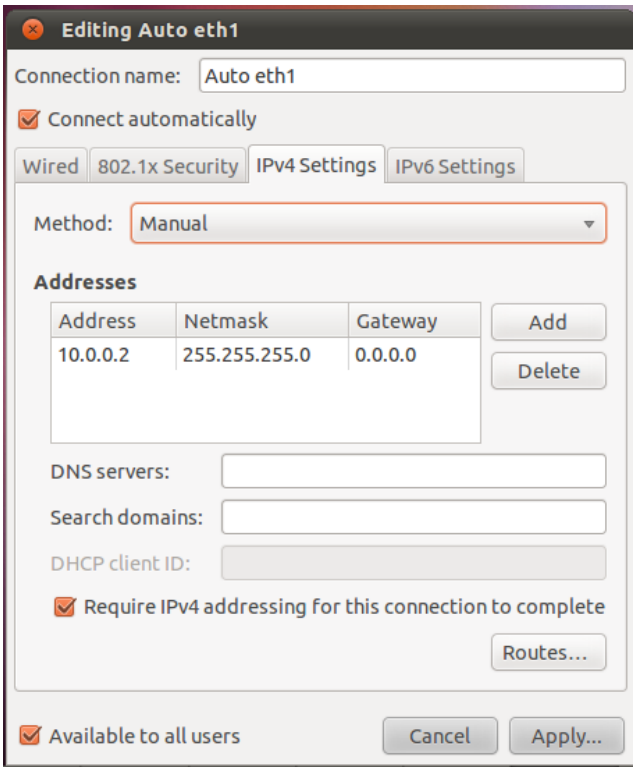
2. Konfigurasi IP eth1

   Kartu jaringan eth1 bersifat bridge dan digunakan untuk menghubungkan server dengan klien yang akan digunakan untuk pengujian.

   Untuk mengatur IP eth1, pada menu taskbar Ubuntu 10.10 yang digunakan sebagai server, pilih: *System →Preferences → Network Connection*

   Pilih *Auto eth1* dan klik *edit*. Pada tab *IPv4 Settings*, bagian *Method* pilih *Manual*.

   Klik tombol *Add* untuk menambahkan alamat IP sebagai berikut:



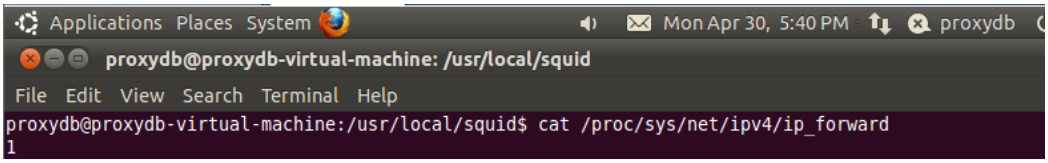

**Gambar 4. 2** Konfigurasi alamat IP eth1

Setelah selesai, klik *Apply*.

3. Konfigurasi IP forward

   Konfigurasi IP forward dilakukan agar server mampu menghubungkan dua jaringan yang berbeda ari dua antarmuka jaringan yang dimiliki, sehingga paket yang datang dari klien maupun layanan internet dapat diteruskan oleh server. Agar server mampu meneruskan paket tersebut, ubah isi file /proc/sys/net/ipv4/ip_forward menjadi angka satu.

```
$sudo gedit /proc/sys/net/ipv4/ip_forward
```




**Gambar 4. 3** Konfigurasi IP forward

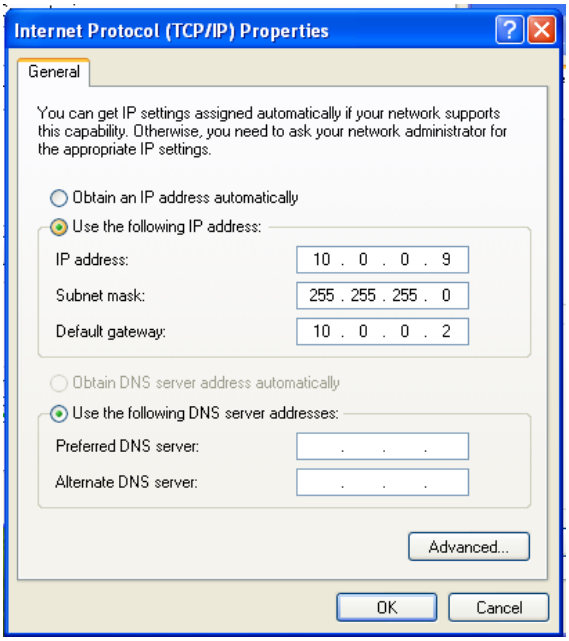
### 4.1.7 Konfigurasi Alamat IP Klien

1. IP klien diatur agar satu jaringan dengan server.

Klik pada WindowsXP: *Start→Control Panel → Network Connections→ Local Area Network.*

Pada tab *General*, klik menu *Properties*, kemudian klik dua kali pada *Internet Protocol (TCP/IP).*

Pilih *Use the following IP address* untuk memberikan IP secara manual sebagai berikut:




**Gambar 4. 4** Konfigurasi alamat IP klien

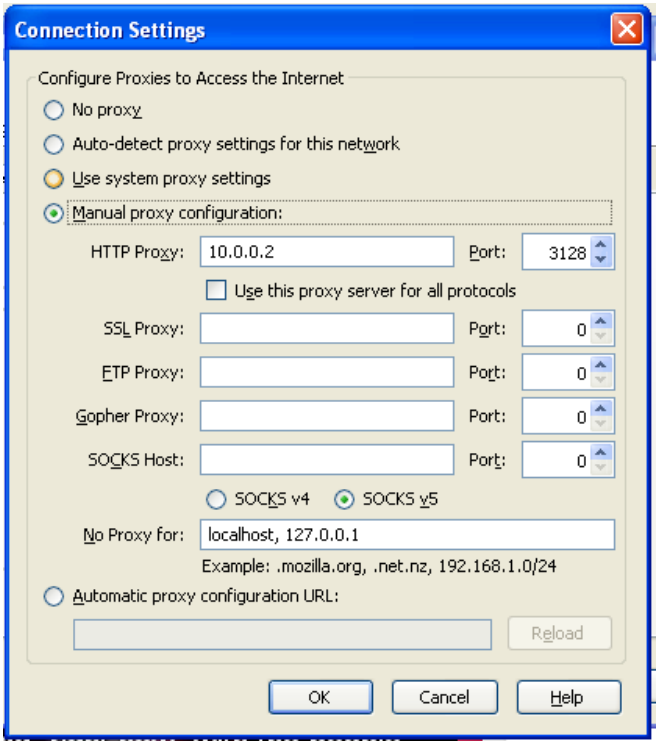
### 4.1.8 Definisi Proxy Server

Proxy server yang digunakan pada penelitian ini tidak dibuat transparan, sehingga harus didefinisikan secara manual pada browser.

Berikut ini pendefinisian proxy pada browser mozzila firefox di klien Windows:

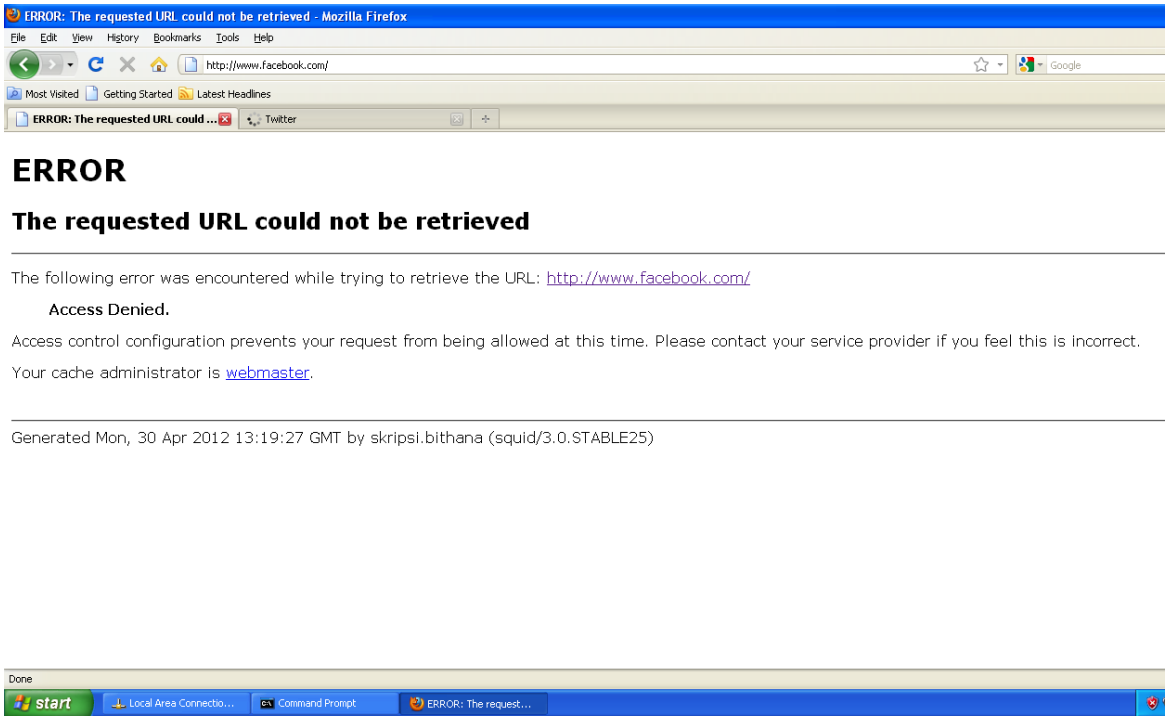
*Firefox→Tools→Options→Advanced→Network→Settings.*

Pilih *Manual proxy configuration* kemudian isikan IP *server* sebagai berikut:




**Gambar 4. 5** Definisi IP proxy *server*

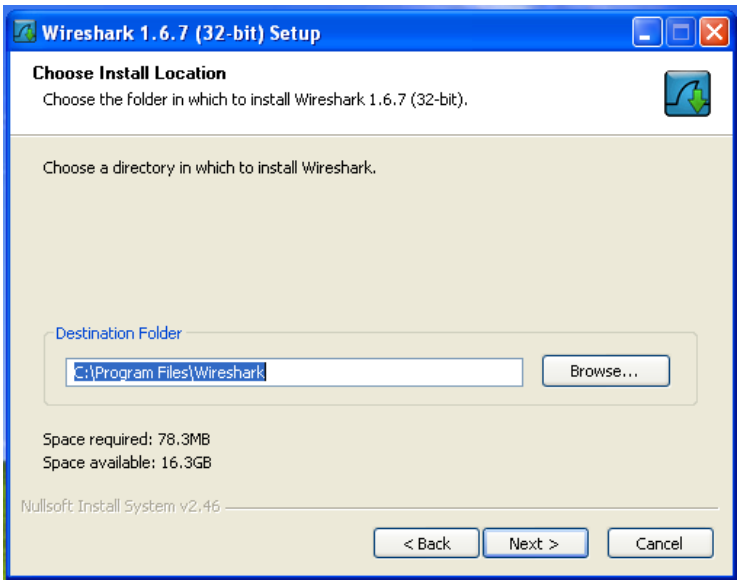
Uji proxy server dengan akses facebook.com menghasilkan pesan *error*.




**Gambar 4. 6** Hasil Uji Proxy server

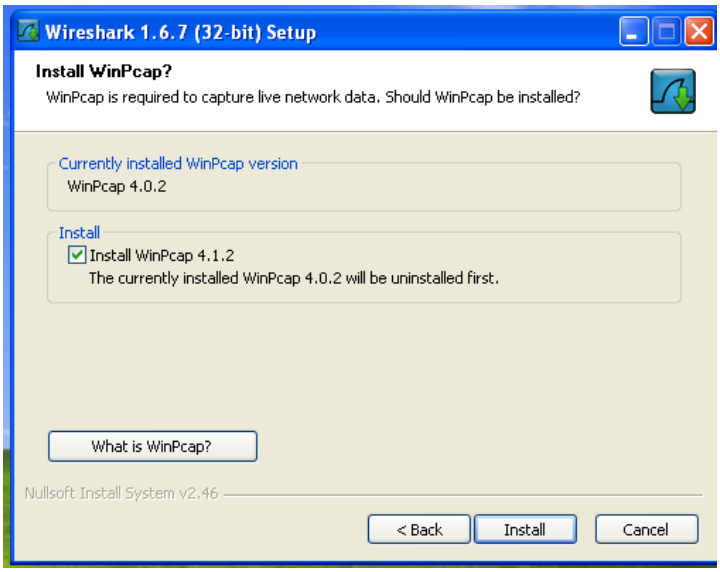
### 4.1.9 Instalasi dan Penggunaan Wireshark

Pada tahap ini dilakukan instalasi wirsehark pada sisi klien untuk kebutuhan memonitor paket data.



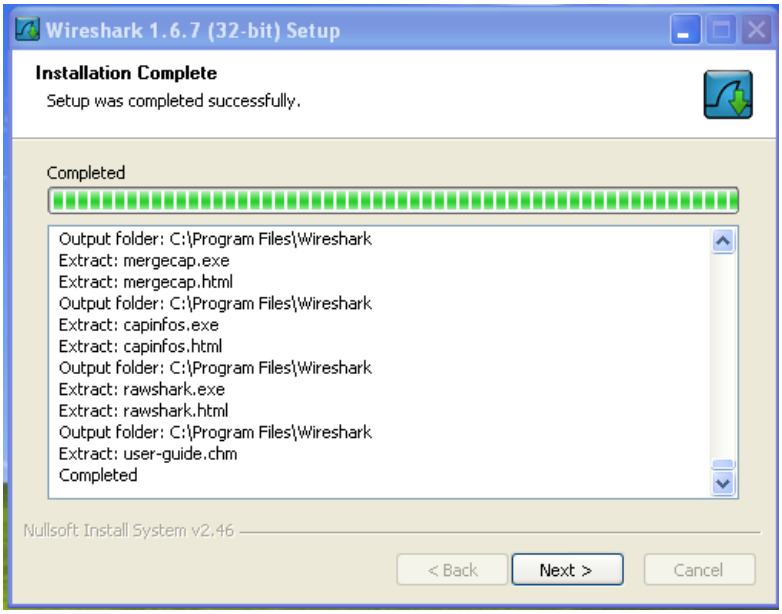

**Gambar 4. 7** Pemilihan Letak Instalasi Wireshark

Pada pertengahan instalasi, akan ada permintaan instalasi WinPcap sebagai program penangkap paket pada Wireshark. Pilih *install*.



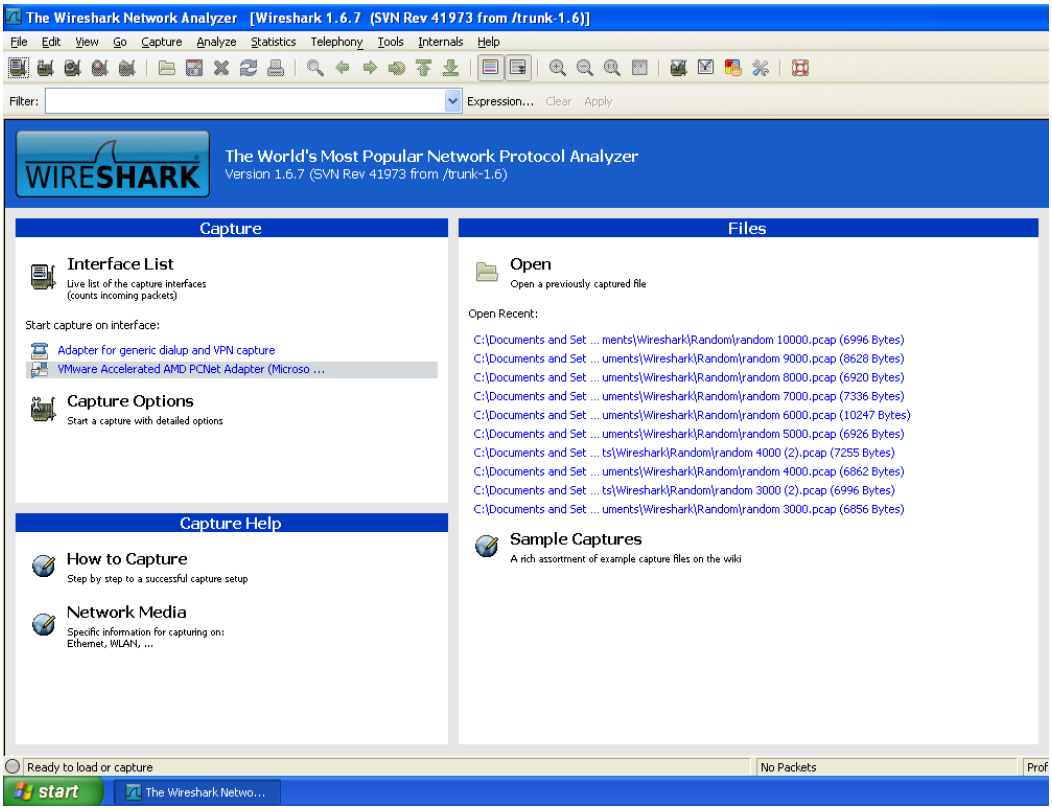

**Gambar 4. 8** Instalasi WinPcap

Tunggu hingga instalasi WinPcap dan Wireshark selesai. Kemudian *next*.



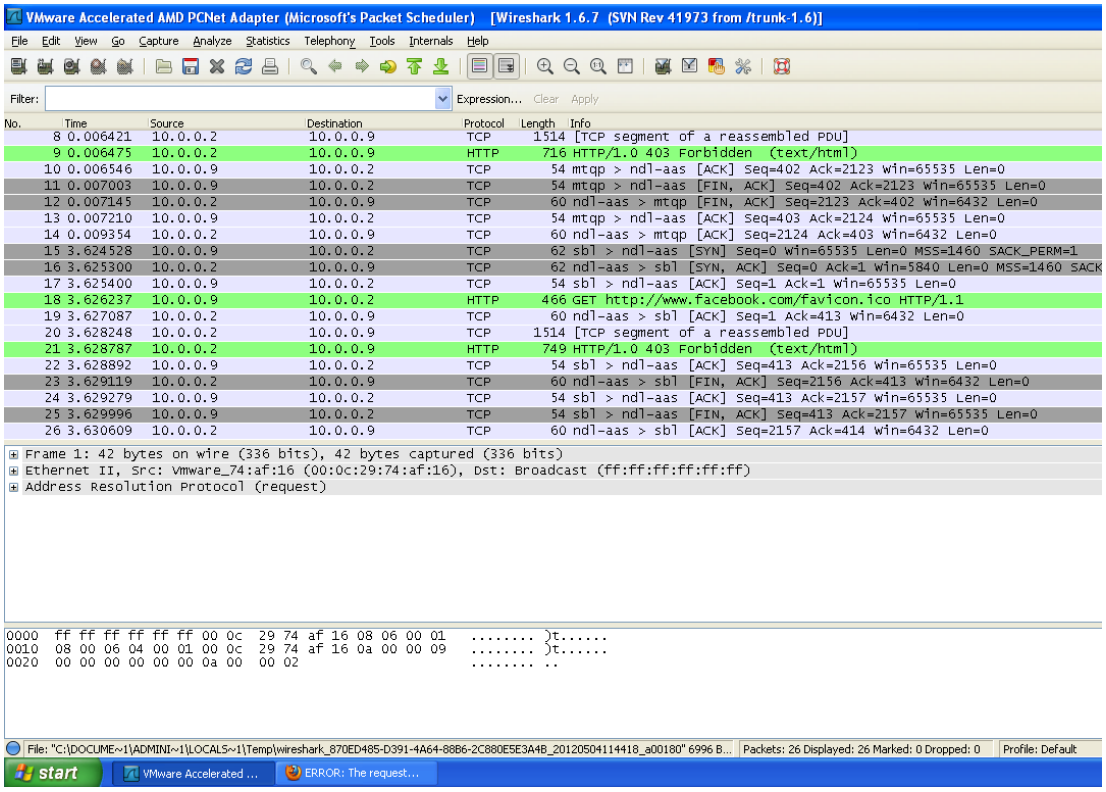

**Gambar 4. 9** Instalasi Wireshark

Setelah proses instalasi selesai, wireshark akan menampilkan halaman utamanya.




**Gambar 4. 10** Halaman Utama Wireshark

Untuk mulai menangkap paket, adapter yang akan digunakan dipilih pada bagian *Interface List*. Pada penelitian ini, wireshark berada pada komputer klien. Adapter yang digunakan adalah kartu jaringan virtual milik VMware. Pemilihan adapter pada komputer akan mengaktifkan penerimaan seluruh paket data. Untuk menghentikan penangkapan paket, klik *stop*. Jenis paket data yang ditangkap dapat diatur pada menu *filter*. Jika fungsi *filter* tidak digunakan, seluruh paket yang ditangkap oleh adapter akan ditampilkan wireshark. Berikut ini tampilan paket-paket yang berhasil ditangkap wireshark tanpa *filter*:




**Gambar 4. 11** Tampilan Paket yang berhasil ditangkap Wireshark

Untuk menangkap paket-paket tertentu dapat digunakan fungsi *filter*, agar tidak menyulitkan pengamatan.

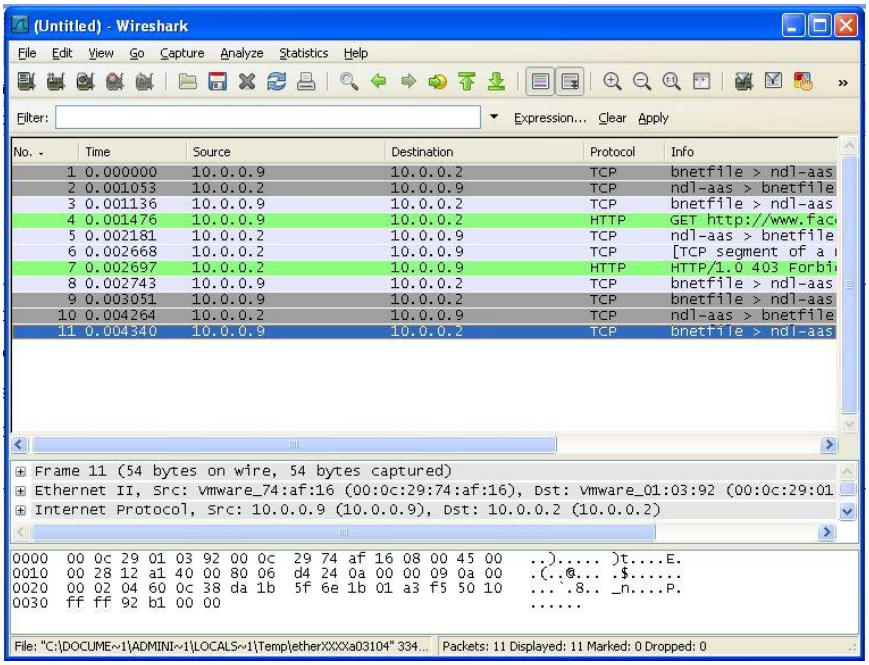
## 4.2 Pengumpulan Data

Waktu yang digunakan *domain filtering* proxy untuk mencari dan membandingkan alamat yang diakses klien dijadikan sebagai data analisis

dalam penelitian ini. Waktu yang digunakan proxy untuk memeriksa aturan tersebut, pada sisi klien akan terasa sama dengan waktu yang berjalan ketika permintaan akses dikirimkan hingga pesan *error* diterima. Wireshark sebagai aplikasi penangkap paket mampu merekam dan menampilkan waktu saat paket-paket tersebut melewati adapter. Waktu tersebut akan dicatat, sehingga data untuk analisis didapatkan dari selisih waktu ketika pesan *error* diterima dengan waktu permintaan akses dikirimkan. Pengambilan data untuk analisis dilakukan sepuluh kali pada tiap-tiap jumlah alamat *domain* yang berada dalam ACL.

Perbandingan kinerja *domain filtering* menggunakan basis data dan file teks dilakukan dengan analisis data, dimana data akan diambil ketika ACL pada proxy berisi alamat *domain* yang disusun acak pada basis data maupun file teks. Sedangkan perbandingan kinerja *domain filtering* menggunakan basis data dilakukan dengan alamat *domain* yang disusun secara acak dan terurut pada basis data. Analisis ini dilakukan untuk mengetahui kinerja *domain filtering* menggunakan basis data dengan data acak dan terurut.

Berikut ini gambar paket data HTTP yang berhasil ditangkap oleh aplikasi wireshark:




**Gambar 4. 12** Paket Data HTTP yang Tertangkap Wireshark

Pada gambar 4.12, paket dengan warna hijau merupakan paket yang melewati protokol HTTP.

Dari tampilan paket yang telah berhasil ditangkap wireshark tersebut, dapat dilihat bahwa pengiriman permintaan adalah paket nomor 4, yang ditandai dengan:

| | | |
|---|---|---|
| Asal paket (*Source*) | : 10.0.0.9 | (IP klien) |
| Tujuan (*Destination*) | : 10.0.0.2 | ( IP proxy server) |
| Protokol | : HTTP | |
| Info | : GET | (metode permintaan layanan) |

Sehingga waktu pengiriman permintaan adalah *time* : **0,001476**

Dari gambar diatas juga dapat dilihat bahwa pengiriman layanan balasan atau pesan *error* dari proxy server adalah paket nomor 7, yang ditandai dengan:

| | | |
|---|---|---|
| Asal paket (*Source*) | : 10.0.0.2 | ( IP proxy server) |
| Tujuan (*Destination*) | : 10.0.0.9 | (IP klien) |
| Protokol | : HTTP | |
| Info | : 403 Forbidden | (akses tidak diijinkan) |

Sehingga waktu pengiriman permintaan adalah *time* : **0.002697**

Dari informasi diatas didapat:

Waktu perngiriman permintaan = 0.001476

Waktu penerimaan pesan *error* = 0.002697

Dari data tersebut dapat diperoleh selisih waktu:

Selisih waktu = 0.002697 - 0.001476

= 0.00122

Selisih tersebut akan digunakan sebagai data pengujian. Rata-rata selisih setelah sepuluh kali pengujian digunakan sebagai data analisis.

## 4.3 Pengisian *Domain*

### 4.3.1 Pengisian Basis Data menggunakan Data Acak

Untuk pengumpulan data waktu kinerja *domain filtering* menggunakan basis data dengan *domain* acak, daftar alamat disusun secara *worst-case* file dimana keseluruhan isi *domain* adalah *filler* kecuali posisi terakhir yang diisi dengan *domain* yang sesungguhnya. *Filler* yang dibuat

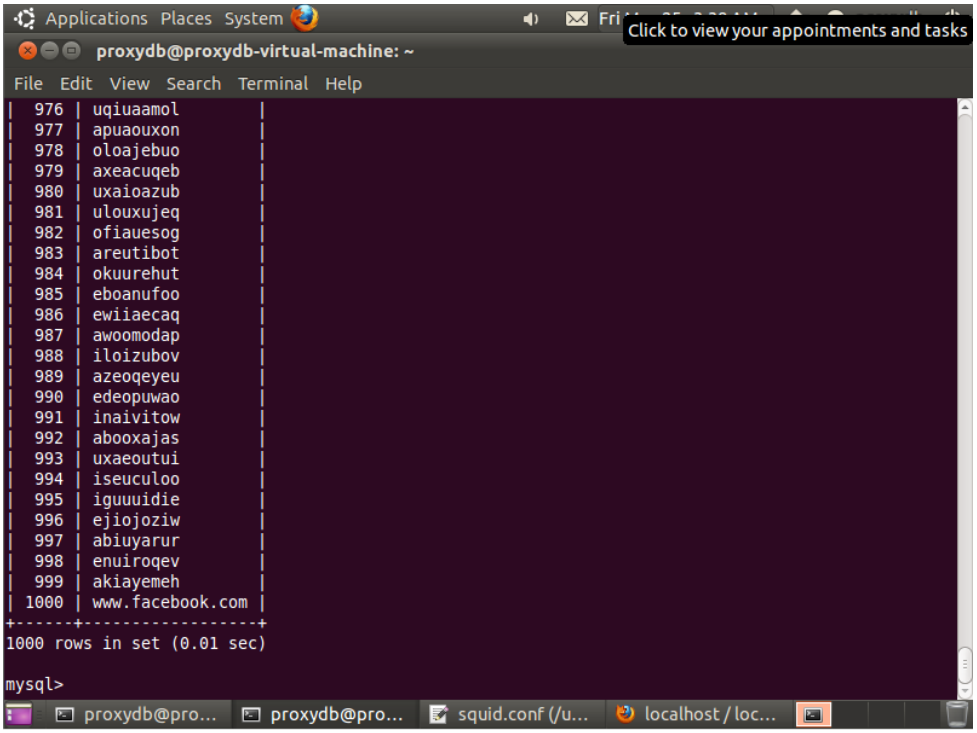
dari susunan huruf-huruf acak diisikan dalam basis data dengan perintah berikut:

```
mysql>INSERT INTO mysql_acl.host_acl_allow (host) VALUES
('iluucodie');
mysql> INSERT INTO mysql_acl.host_acl_allow (host) VALUES
('oguewetal');
...
...
```

Untuk memenuhi kriteria *worst-case* file, posisi terakhir setelah *filler* diisi dengan alamat *domain* yang akan diblok, misalnya www.facebook.com

```
mysql> INSERT INTO mysql_acl.host_acl_allow (host) VALUES
       ('www.facebook.com');
```

Berikut ini tampilan basis data yang telah diisi dengan alamat *domain* acak (*worst-case* file):




**Gambar 4. 13** Tampilan isi basisdata acak

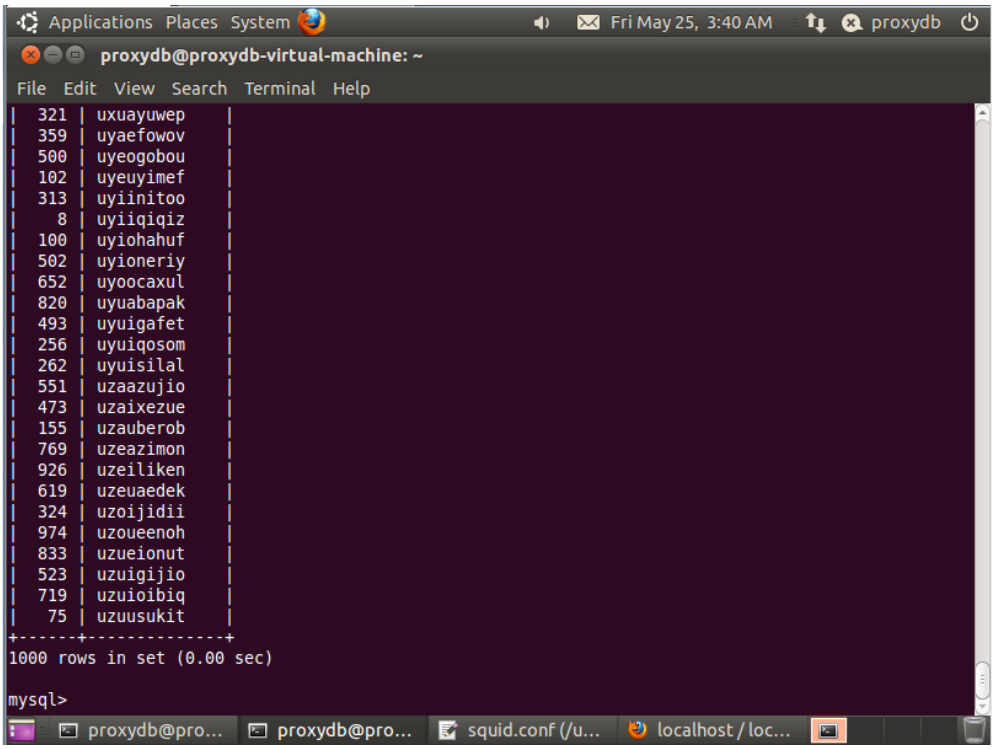
### 4.3.2 Pengisian Basis Data menggunakan Data Terurut

Pengumpulan data waktu kinerja *domain filtering* dengan basis data menggunakan data terurut akan dibandingkan dengan waktu kinerja basis data menggunakan data acak (*worst-case* file). Perbandingan tersebut digunakan untuk mengetahui kinerja *domain filtering* menggunakan basis data. Susunan basis data terurut didapat dengan mengubah urutan basis data yang telah diisikan sebelumnya secara acak (*worst-case* file) menjadi terurut secara alfabetis. Basis data acak diurutkan menggunakan perintah:

```
mysql> ALTER TABLE mysql_acl.host_acl_allow ORDER BY 'host';
```

Alamat *domain* yang terdapat pada basis data baik acak dan terurut sama persis. Perbedaan hanya terdapat pada urutannya saja.

Berikut ini tampilan isi dalam basis data yang telah diurutkan:




**Gambar 4. 14** Tampilan isi basisdata terurut

### 4.3.3 Pengisian File menggunakan Data Acak

Penggunaan file teks sebagai acl pada squid proxy server memerlukan perubahan konfigurasi sebagai berikut:

```
$ gedit /usr/local/squid/etc/squid.conf
```

```
acl mysqlacl2 dstdomain –i "/usr/local/squid/blok.txt"


http_access deny mysqlacl2
http_access allow localnet
http_access allow all
```

Keterangan:

Baris 1:

mysqlacl2 : nama acl yang digunakan adalah mysqlacl2

dstdomain : membandingkan alamat *domain*/URL situs yang diminta klien dengan alamat yang persis sama dengan yang terdaftar dalam acl mysqlacl2

–i : *incase-sensitive* yang berarti tidak membedakan huruf besar dan huruf kecil.

/usr/local/squid/blok.txt : file yang digunakan untuk menyimpan alamat *domain* pada acl mysqlacl2

Baris 2 : memberikan kebijakan agar acl dengan nama mysqlacl2 ditolak meneruskan paket.

Baris 3 : digunakan untuk memberikan kebijakan akses ke jaringan lokal agar dapat dialuhkan (*redirect*) ke jaringan NAT, sehingga kebijakan untuk *localnet* tergolong umum. Squid akan mengerjakan perintah secara berurutan, yang atas terlebih dulu, karenanya aturan khusus harus diletakkan diatas aturan umum. Aturan khusus yang dibuat tidak akan berfungsi jika aturan umum terlebih dulu diberlakukan, hal tersebut akan membuat squid seperti tidak berfungsi. Solusinya baris ini harus diletakkan setelah aturan khusus yang dibuat, atau jika diletakkan diatas aturan khusus maka harus diberi pagar agar tidak dieksekusi.

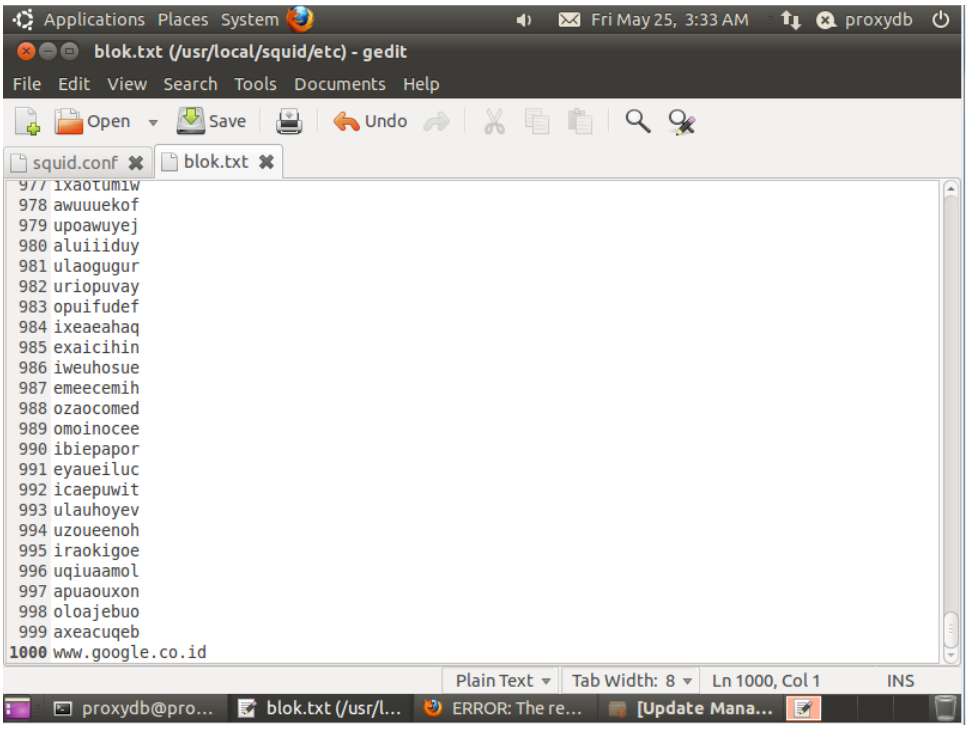
Baris 4 : digunakan untuk memberikan kebijakan secara umum agar acl yang tidak didefinisikan sebelumnya diperbolehkan untuk meneruskan paket data.

Penggunaan dstdomain pada konfigurasi proxy agar file pada *domain filtering* memiliki kinerja yang sama seperti mysql_acl yang bekerja dengan alamat *domain* secara lengkap. Jika menggunakan url_regex kondisi pada basis data dan file akan berbeda, hal ini akan membuat data penelitian menjadi tidak dapat dibandingkan.

Setelah konfigurasi diubah seperti konfigurasi diatas, file yang dibutuhkan dibuat dengan perintah:

```
$ touch /usr/local/squid/blok.txt
```

File `/usr/local/squid/blok.txt` tersebut diisi dengan *filler* sejumlah *domain* yang sama dengan basis data, dengan baris terakhir adalah *domain* atau alamat yang akan dilarang untuk diakses, misalnya www.google.co.id. Berikut ini tampilan file teks berisi alamat *domain* acak:




**Gambar 4. 15** Tampilan isi file teks berisi alamat *domain* acak

## 4.4 Tabel Hasil Pengujian

### 4.4.1. Pengujian dengan 1000-10.000 Data

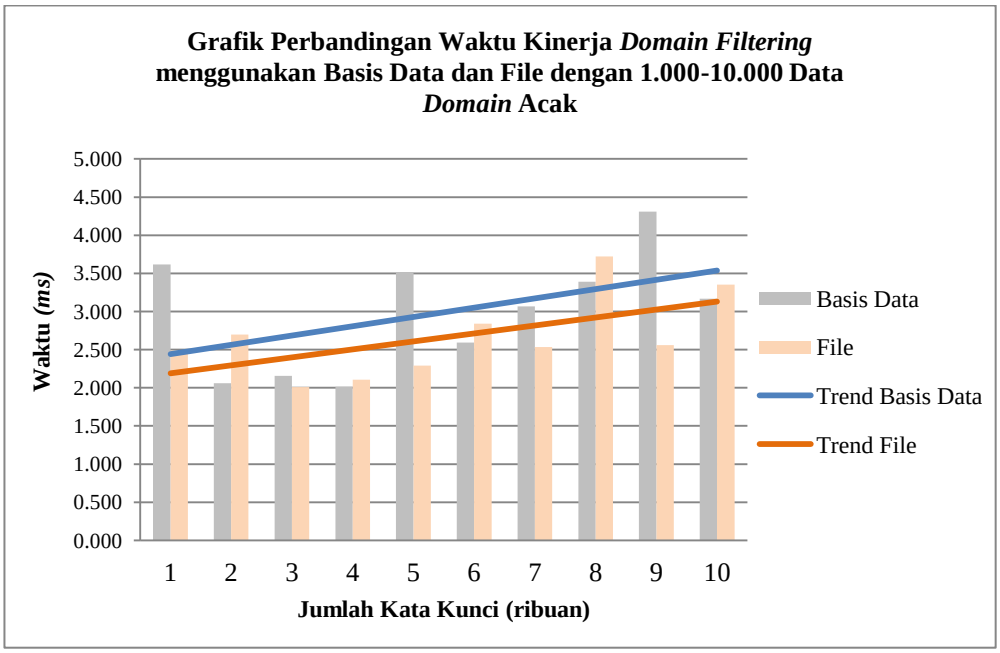
#### 4.4.1.1 Pengujian menggunakan Basis Data dan File Teks Berisi Data Acak

Perbandingan kinerja *domain filtering* menggunakan basis data dan file teks dilakukan dengan alamat *domain* yang disusun secara acak (*worst-case* file) pada basis data dan file teks. Jumlah alamat *domain* yang digunakan pada pengujian ini dimulai mulai dari 1.000-10.000 data, dengan penambahan 1.000 data untuk tiap pengujian. Pengujian dilakukan sepuluh kali pada tiap-tiap jumlah *domain*, kemudian dirata-rata untuk mendapatkan data analisis.

Berikut ini hasil pengujian menggunakan basis data dan file teks yang berisi kata acak dengan jumlah bervariasi mulai dari 1.000 hingga 10.000:

**Tabel 4.1** Hasil rata-rata waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 1.000-10.000 data *domain* acak

| Jumlah *domain* | Basis data (*ms*) | File (*ms*) |
|---|---|---|
| 1.000 | 3,6185 | 2,4881 |
| 2.000 | 2,0614 | 2,6964 |
| 3.000 | 2,1584 | 2,0079 |
| 4.000 | 2,0143 | 2,1069 |
| 5.000 | 3,5151 | 2,2924 |
| 6.000 | 2,5941 | 2,8407 |
| 7.000 | 3,0659 | 2,5341 |
| 8.000 | 3,3892 | 3,7227 |
| 9.000 | 4,3116 | 2,5586 |
| 10.000 | 3,1677 | 3,3528 |




**Gambar 4. 16** Grafik perbandingan waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 1.000-10.000 data *domain* acak

Waktu yang digunakan *domain filtering* untuk mencari alamat *domain* pada basis data dan file teks cenderung meningkat dengan bertambahnya jumlah kata dalam ACL. Peningkatan waktu tersebut berkisar antara 0,002%-0,004%. Selisih antara grafik basis data dengan file teks rata-rata 0,3 milidetik, dengan waktu yang digunakan *domain filtering* dengan file teks cenderung lebih singkat daripada basis data. Perbedaan yang sangat kecil tersebut sama seperti hasil uji *independent T-test* berikut ini:

**Tabel 4.2** *Independent Samples Test* waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 1.000-10.000 data *domain* acak

| | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | 95% Confidence Interval of the Difference | |
| | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| Equal variances not assumed | 1.307 | 9.664 | .221 | 1.13040 | .86481 | -.80564 | 3.06644 |

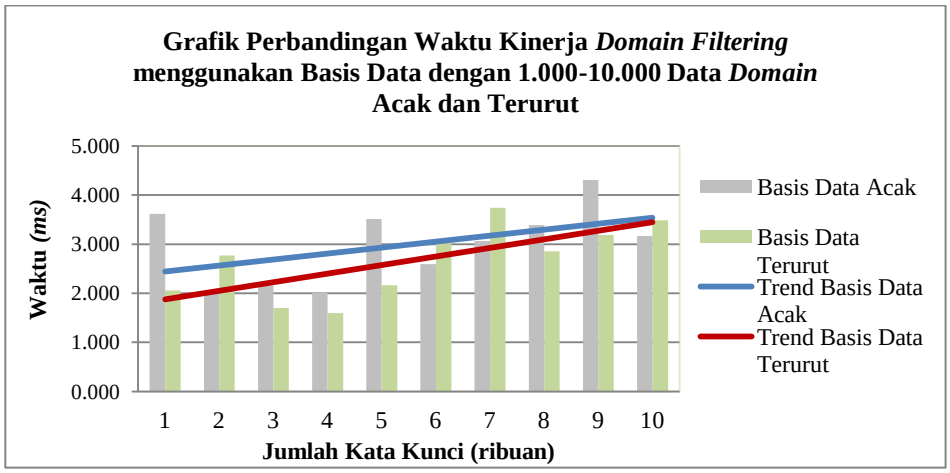
Nilai p(0,221)>0,05 berarti tidak ada perbedaan yang signifikan antara waktu kinerja basis data dan file teks dalam melakukan pencarian. Meskipun begitu, dapat dilihat pada grafik diatas penggunaan file membutuhkan waktu lebih singkat dibandingkan basis data. Hal ini dapat terjadi karena basis data memerlukan *load* aplikasi sebelum melakukan pencarian, sedangkan file dapat langsung bekerja tanpa membutuhkan *load* aplikasi tambahan.

#### 4.4.1.2 Pengujian menggunakan Basis Data Berisi Data Acak dan Terurut

Berikut ini pengujian menggunakan basis data berisi kata acak dan terurut dengan jumlah bervariasi mulai dari 1.000 hingga 10.000:

**Tabel 4.3** Hasil rata-rata waktu kinerja *domain filtering* menggunakan basis data acak dan terurut dengan 1.000-10.000 data *domain*

| Jumlah *domain* | Acak (*ms*) | Urut (*ms*) |
|---|---|---|
| 1.000 | 3,6185 | 2,0620 |
| 2.000 | 2,0614 | 2,7680 |
| 3.000 | 2,1584 | 1,7003 |
| 4.000 | 2,0143 | 1,5966 |
| 5.000 | 3,5151 | 2,1635 |
| 6.000 | 2,5941 | 3,0771 |
| 7.000 | 3,0659 | 3,7386 |
| 8.000 | 3,3892 | 2,8522 |
| 9.000 | 4,3116 | 3,1894 |
| 10.000 | 3,1677 | 3,4880 |




**Gambar 4. 17** Grafik perbandingan waktu kinerja *domain filtering* menggunakan basisdata dengan 1.000-10.000 data *domain* acak dan terurut

Kenaikan terjadi dalam kisaran 0,0009-0,003 detik dengan rata-rata kenaikan 0,002%. Selisih antara penggunaan basis data terurut dengan acak (*worst-case* file) sangat kecil, hanya berkisar 0,0007-0,001 detik. Perbedaan yang tidak signifikan tersebut juga dihasilkan uji *independent T-test* seperti pada tabel dibawah ini:

**Tabel 4.4** *Independent Samples Test* waktu kinerja *domain filtering* menggunakan basis data dengan 1.000-10.000 data *domain* acak dan terurut

| | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | 95% Confidence Interval of the Difference | |
| | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| Equal variances not assumed | -.779 | 14.125 | .449 | -.48300 | .62035 | -1.81241 | .84641 |

Nilai p(0,449)>0,05 berarti tidak ada perbedaan yang signifikan antara waktu kinerja basis data dengan data acak dan terurut dalam melakukan pencarian. Meskipun begitu, dapat dilihat pada grafik waktu yang dibutuhkan *domain filtering* menggunakan basis data acak dan terurut

dengan jumlah *domain* 1.000-10.000 cenderung naik seiring bertambahnya jumlah *domain*. Hal ini terjadi karena makin banyak data makin banyak pencarian yang harus dilakukan, sehingga waktu yang dibutuhkan untuk bekerja akan meningkat. Namun, waktu yang dibutuhkan *domain filtering* dengan basis data terurut lebih singkat daripada basis data acak, seperti yang dapat dilihat dari grafik basis data terurut yang cenderung berada di bawah grafik basis data acak. Hal ini dapat dikarenakan pencarian pada data terurut lebih cepat daripada data acak.

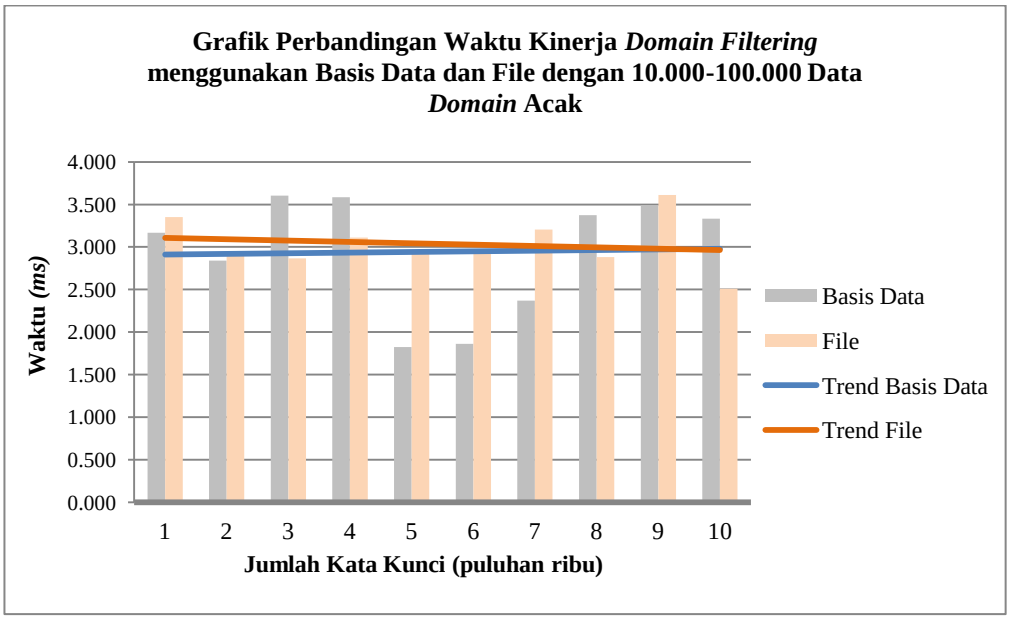
### 4.4.2. Pengujian dengan 10.000-100.000 Data

#### 4.4.2.1 Pengujian menggunakan Basis Data dan File Berisi Data Acak

Berikut tabel hasil rata-rata pengujian proxy server menggunakan wireshark dengan acl basis data dan file teks berisi kata acak:

**Tabel 4.5** Hasil rata-rata waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 10.000-100.000 data *domain*

| Jumlah *domain* | Basis Data (*ms*) | File (*ms*) |
|---|---|---|
| 10.000 | 3,168 | 3,353 |
| 20.000 | 2,841 | 2,952 |
| 30.000 | 3,604 | 2,865 |
| 40.000 | 3,584 | 3,113 |
| 50.000 | 1,826 | 2,936 |
| 60.000 | 1,862 | 2,918 |
| 70.000 | 2,371 | 3,205 |
| 80.000 | 3,373 | 2,882 |
| 90.000 | 3,491 | 3,609 |
| 100.000 | 3,334 | 2,511 |




**Gambar 4. 18** Grafik perbandingan waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 10.000-100.000 data *domain* acak

Konsumsi waktu untuk kinerja *domain filtering* menggunakan basis data maupun file teks cenderung sama pada jumlah *domain* 10.000-100.000. Perbedaan waktu yang dibutuhkan basisdata acak berada pada kisaran 0,001-0,003 detik. Sedangkan pada file teks, perbedaan waktu pada tiap kenaikan jumlah kata berkisar antara 0,7-2 *ms*. Rata-rata kenaikan waktu yang dibutuhkan *domain filtering* menggunakan basisdata maupun file teks dengan jumlah *domain* 10.000-100.000 adalah 0,003%. Perbedaan waktu antara basis data dan file teks berada diatara 0,0025-0,0036%.

Berikut ini tabel hasil uji *independent T-test*:

**Tabel 4.6** *Independent Samples Test* waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 10.000-100.000 data *domain*

| | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | 95% Confidence Interval of the Difference | |
| | | | | | | Lower | Upper |
| Equal variances not assumed | 1.164 | 15.764 | .262 | .601025 | .516145 | -.494490 | 1.696540 |

Nilai p(0,262)>0,05 berarti tidak ada perbedaan yang signifikan antara file dan basis data pada jumlah *domain* 10.000-100.000

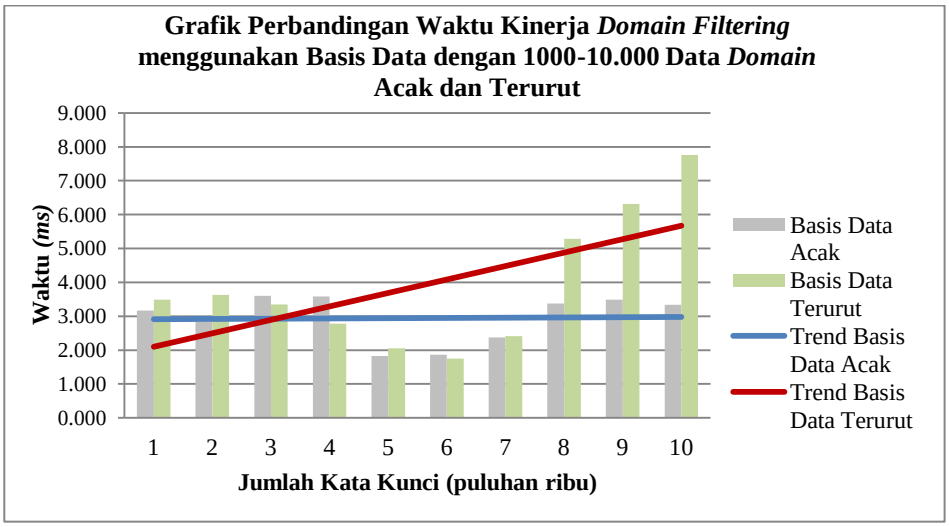
Jika melihat pada grafik di atas, penggunaan file pada *domain filtering* memerlukan waktu yang cukup stabil, bahkan cenderung menurun. Sedangkan penggunaan basis data pada *domain filtering* memerlukan waktu yang lebih fluktuatif walaupun grafik kecenderungannya memperlihatkan waktu yang stabil. Hal ini menandakan bahwa kenaikan 10.000 dari jumlah 10.000 hingga 100.000 tidak mempengaruhi waktu pencarian basis data dan file teks pada *domain filtering*.

#### 4.4.2.2 Pengujian menggunakan Basis Data Berisi Data Acak dan Terurut

Berikut ini tabel hasil pengujian proxy server menggunakan wirehark dengan acl basis data berisi kata acak dan terurut:

**Tabel 4.7** Hasil rata-rata waktu kinerja *domain filtering* menggunakan basis data acak dan terurut dengan 10.000-100.000 data *domain*

| Jumlah *domain* | Acak (*ms*) | Urut (*ms*) |
|---|---|---|
| 10.000 | 3,168 | 3,488 |
| 20.000 | 2,841 | 3,632 |
| 30.000 | 3,604 | 3,351 |
| 40.000 | 3,584 | 2,777 |
| 50.000 | 1,826 | 2,054 |
| 60.000 | 1,862 | 1,747 |
| 70.000 | 2,371 | 2,416 |
| 80.000 | 3,373 | 5,284 |
| 90.000 | 3,491 | 6,316 |
| 100.000 | 3,334 | 7,763 |




**Gambar 4. 19** Grafik perbandingan konsumsi waktu kinerja *domain filtering* menggunakan basisdata acak dan terurut dengan 10.000-100.000 data *domain*

Pada *domain* berjumlah 1.000-10.000, grafik basis data acak dan terurut tidak memiliki selisih yang besar. Kedua grafik cenderung naik dengan selisih yang tidak terlalu banyak. Perbedaan dengan grafik pada jumlah data 10.000 hingga 100.000 diatas sangat besar. Grafik diatas menunjukkan basis data terurut mengalami kenaikan yang cukup banyak, sedangkan penggunaan data acak memiliki grafik yang hampir sama untuk tiap-tiap pengujian. Grafik basis data acak yang cenderung berada dibawah grafik basis data terurut menandakan kinerja *domain filtering* dengan basis data acak membutuhkan waktu yang lebih singkat daripada basis data terurut.

Kenaikan waktu yang digunakan *domain filtering* menggunakan basis data terurut rata-rata 0,003%, dengan interval kenaikan 0,001%-0,006%. Selisih waktu antara basis data terurut dengan acak memiliki rentang cukup banyak, yaitu berkisar antara 0,001%-0,007%. Sangat berbeda dengan selisih waktu pada data berjumlah 1000-10.000 sebelumnya yang normalnya berada pada kisaran 0,001%-0,003%.

**Tabel 4.8** *Independent Samples Test* waktu kinerja *domain filtering* menggunakan basis data dengan 10.000-100.000 data *domain* acak dan terurut

| | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | 95% Confidence Interval of the Difference | |
| | | | | | | Lower | Upper |
| Equal variances not assumed | -.754 | 9.166 | .470 | -2.824400 | 3.746275 | -11.275772 | 5.626972 |

Nilai p(0,470)>0,05 berarti tidak ada perbedaan yang signifikan antara waktu kinerja *domain filtering* menggunakan basis data dengan data terurut maupun data acak.

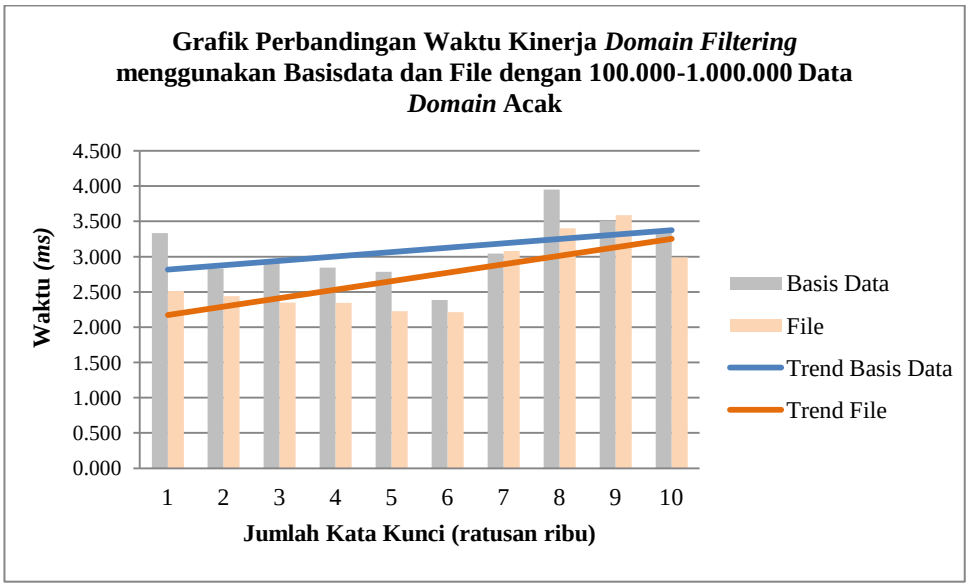
### 4.4.3. Pengujian dengan 100.000-1.000.000 Data

#### 4.4.3.1 Pengujian menggunakan Basis Data dan File Teks Berisi Data Acak

Berikut ini tabel hasil rata-rata pengujian proxy server menggunakan wireshark dengan acl basis data dan file teks berisi kata acak:

**Tabel 4.9** Hasil rata-rata waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 100.000-1.000.000 data *domain*

| Jumlah *domain* | Basis Data (*ms*) | File (*ms*) |
|---|---|---|
| 100.000 | 3,334 | 2,511 |
| 200.000 | 2,830 | 2,439 |
| 300.000 | 2,939 | 2,347 |
| 400.000 | 2,843 | 2,347 |
| 500.000 | 2,785 | 2,228 |
| 600.000 | 2,384 | 2,213 |
| 700.000 | 3,043 | 3,081 |
| 800.000 | 3,951 | 3,404 |
| 900.000 | 3,509 | 3,588 |
| 1.000.000 | 3,355 | 2,988 |




**Gambar 4. 20** Grafik perbandingan konsumsi waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 100.000-1.000.000 data *domain*

Pada grafik konsumsi waktu kinerja *domain filtering* menggunakan basis data dan file teks dengan 100.000-1.000.000 data *domain* diatas, tidak terlihat selisih yang jauh berbeda. Kedua grafik cenderung meningkat dengan rentang kenaikan berkisar 0,002%-0,003%. Grafik file teks yang cenderung berada di bawah grafik basis data menandakan bahwa penggunaan file teks pada *domain filtering* membutuhkan waktu yang lebih singkat daripada penggunaan basis data. Meskipun data yang digunakan hingga 1.000.000 kata, *domain filtering* menggunakan file teks tetap memiliki kinerja yang relatif lebih singkat dari pada basis data. Hal ini tidak membuktikan hipotesis bahwa basis data mampu bekerja dengan waktu pencarian yang lebih singkat dari pada file teks, bahkan pada data yang besar yaitu hingga satu juta data.

**Tabel 4.10** *Independent Samples Test* waktu kinerja *domain filtering* menggunakan basis data dan file dengan 100.000-1.000.000 data *domain* acak

| | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | 95% Confidence Interval of the Difference | |
| | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| Equal variances not assumed | 1.001 | 17.460 | .330 | .496500 | .495895 | -.547653 | 1.540653 |

Nilai p(0,330)>0,05 berarti tidak ada perbedaan yang signifikan antara waktu kinerja *domain filtering* menggunakan basis data dan file dengan 100.000-1.000.000 data *domain* acak.

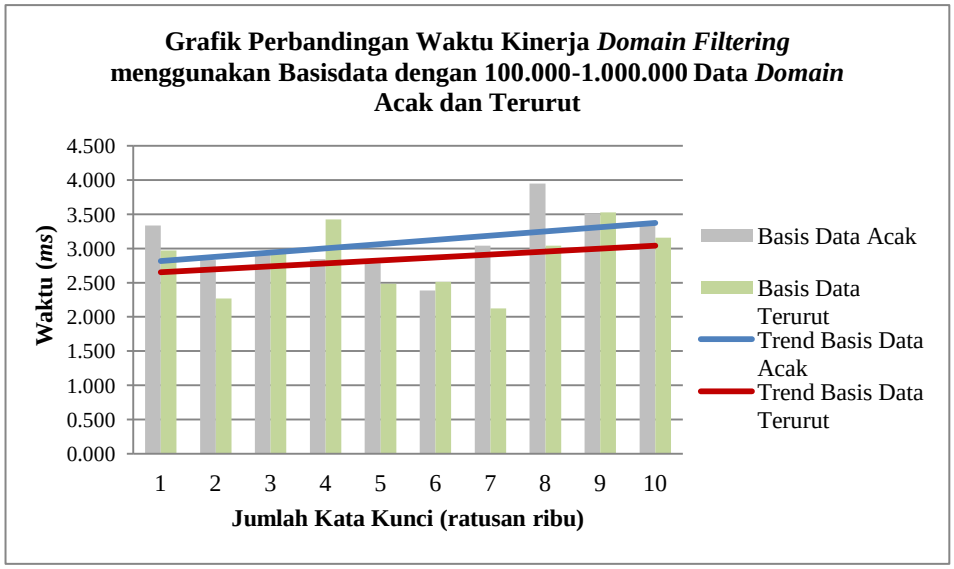
Kinerja file teks yang lebih singkat dari pada basis data dapat terjadi karena aturan squid proxy server yang terdapat pada file dipindahkan ke memori internal squid pada saat sistem dinyalakan atau di konfigurasi ulang, sehingga akses yang dilakukan klien kemungkinan tidak dibandingkan dengan aturan yang terdapat pada file eksternal, melainkan langsung dicari pada memori internal squid, sehingga pencarian file dapat dilakukan dengan lebih cepat dari basis data. Basis data merupakan media yang murni diluar squid, sehingga kemungkinan isi basis data di simpan dalam memori internal lebih kecil dari file. Selain itu, penyebab basis data memiliki waktu kinerja yang lebih lama dari file dapat disebabkan pada kebutuhan basis data akan aplikasi tambahan seperti DBMS untuk dapat melakukan akses.

#### 4.4.3.2 Pengujian menggunakan Basis Data Berisi Data Acak dan Terurut

Berikut ini tabel hasil rata-rata pengujian proxy server menggunakan basis data berisi 100.000-1.000.000 kata acak dan terurut:

**Tabel 4.11** Hasil rata-rata waktu kinerja *domain filtering* menggunakan basis data acak dan terurut dengan 100.000-1.000.000 data *domain*

| Jumlah *domain* | Acak (*ms*) | Urut (*ms*) |
|---|---|---|
| 100.000 | 3,334 | 2,970 |
| 200.000 | 2,830 | 2,267 |
| 300.000 | 2,939 | 2,955 |
| 400.000 | 2,843 | 3,427 |
| 500.000 | 2,785 | 2,484 |
| 600.000 | 2,384 | 2,515 |
| 700.000 | 3,043 | 2,124 |
| 800.000 | 3,951 | 3,044 |
| 900.000 | 3,509 | 3,528 |
| 1.000.000 | 3,355 | 3,158 |




**Gambar 4. 21** Grafik perbandingan konsumsi waktu kinerja *domain filtering* menggunakan basisdata acak dan terurut dengan 100.000-1.000.000 data *domain*

Waktu yang digunakan *domain filtering* pada basis data acak dan terurut dengan jumlah *domain* 100.000-1.000.000 relatif naik dengan selisih hampir tetap pada sebagian besar grafik. Selisih antar grafik rata-

rata 0,003%, dan kenaikan kedua grafik berkisar antara 0,002%-0,003%. Basis data terurut memiliki grafik yang cenderung berada diawah basis data acak, hal ini menandakan pencarian data terurut membutuhkan waktu yang lebih singkat dari data acak, karena pencarian pada data terurut lebih mudah dibandingkan dengan pencarian pada data acak.

Berikut ini tabel hasil uji independent t test untuk perbandingan konsumsi waktu kinerja *domain filtering* menggunakan basisdata acak dan terurut dengan 100.000-100.000 data *domain:*

**Tabel 4.12** *Independent Samples Test* waktu kinerja *domain filtering* menggunakan basis data 100.000-1.000.000 data *domain* terurut dan acak

| | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | 95% Confidence Interval of the Difference | |
| | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| Equal variances not assumed | 1.901 | 12.726 | .080 | .919400 | .483617 | -.127686 | 1.966486 |

Nilai p(0,080)>0,05 berarti tidak ada perbedaan yang signifikan antara waktu kinerja *domain filtering* menggunakan basis data 100.000-1.000.000 data *domain* terurut dan acak.

# BAB V
# SIMPULAN DAN SARAN

## 5.1 Simpulan

Berdasarkan analisis dari penggunaan basis data pada *domain filtering* dengan parameter waktu pencarian untuk kata berjumlah 1000-1.000.000 dapat ditarik kesimpulan sebagai berikut:

1. *Domain filtering* dapat bekerja menggunakan basis data dengan konfigurasi eksternal dan menggunakan program tambahan seperti modul mysql_acl.
2. *Domain filtering* pada proxy dengan menggunakan acl file teks memerlukan waktu yang lebih singkat jika dibandingkan dengan penggunaan basis data. Hal tersebut dibuktikan pada grafik 4.16 dan 4.20.
3. Pencarian data terurut pada basis data membutuhkan waktu lebih singkat dari data acak, seperti pada grafik 4.17 dan 4.21.
4. Basis data memiliki fitur optimasi *query* yang mampu melakukan pencarian dan pengurutan data yang semestinya lebih baik dari file teks, namun dalam penelitian *domain filtering* menggunakan basis daya ini, hal tersebut tidak terbukti benar.

## 5.2 Saran

Beberapa saran untuk pengembangan penelitian ini:

1. Penelitian ini dapat dikembangkan dengan membangun sistem yang akan mengotomasi *filtering* pada mesin pencari, dimana hasil *filtering* disimpan dalam basis data untu digunakan proxy server.
2. Penggunaan basis data no-sql untuk toleransi redundansi data.
3. Menggunakan DBMS yang lebih handal seperti oracle memungkinkan hasil yang lebih baik.
4. Penggunaan Binary Search untuk data terurut akan mempercepat waktu pencarian pada file

# DAFTAR PUSTAKA


Abidin, M. Z. (2011). *Penelitian, Macam Penelitian dan Data Penelitian.*

Fataruba, H. (2012). *Metode Penelitian Eksperimen.* Samarinda: Universitas 17 Agustus 1945.

Nugroho, W. (2011). *Modul 10 Database dan Sistem Manajemen Database*.

Nursyahidah, F. (2011). *Penelitian Eksperimen.*

Rieffel, E. (2008). *Quantum Computing.* FX Palo Alto Laboratory.

Sirkel, A. L. (2008). *Modul Praktikum Basis Data.* Yogyakarta: Laboratorium Sistem Informasi dan Rekayasa Perangkat Lunak Jurusan Teknik Informatika FTI UII.

Sisjarkom, A. L. (2009). *Modul Praktikum Jaringan Komputer.* Yogyakarta: Laboratorium Sistem dan Jaringan Komputer Jurusan Teknik Informatika FTI UII.

Subianto, M. (2007). *Metode Penelitian Eksperimen.* Syah Kuala University.

Wack, J., Cutler, K., & Pole, J. (2002). *Guidelines on Firewalls and Firewall Policy.*

FORM-TA/TF-A3

**UNIVERSITAS ISLAM INDONESIA**
**Jurusan Teknik Informatika FTI**

## SARAN/USULAN PRESENTASI KEMAJUAN TUGAS AKHIR

Nama Mhs. : Bithana Paragustin
No. Mhs. : 07 523 024
Judul TA : _____

Tambahkan keterangan durasi waktu pada batasan masalah

**Nilai kemajuan Tugas Akhir:** _____ **(0 – 100)**
**(studi pustaka, perancangan, penguasaan materi, ketepatan)**

Yogyakarta, 13-12-2011

Dosen,

RINTA H. (nama terang)

*Dilampirkan pada Laporan TA yang diajukan untuk pendadaran*

FORM-TA/TF-A3

**UNIVERSITAS ISLAM INDONESIA**
**Jurusan Teknik Informatika FTI**

## SARAN/USULAN PRESENTASI KEMAJUAN TUGAS AKHIR

Nama Mhs. : Bithara .P
No. Mhs. : 07523024
Judul TA : _____

- Batasan masalah diperbaiki, sesuaikan dgn konsep penelitian / pengujian / implementasi aplikasi
- Segera diselesaikan !

Nilai kemajuan Tugas Akhir: — (0 – 100)
(studi pustaka, perancangan, penguasaan materi, ketepatan)

Yogyakarta, 13 Desember 2011

Dosen,

Mursalim T (nama terang)

*Dilampirkan pada Laporan TA yang diajukan untuk pendadaran*