TUGAS AKHIR - TE 141599

ANALISA UNJUK KERJA PROXY SERVER MENGGUNAKAN ALGORITMA LRU, LFU, DAN GDSF

Adi Kurniawan
NRP 07111445000058

Dosen Pembimbing
Dr. Ir. Achmad Affandi DEA.
Dr. Istas Pratomo, S.T., M.T.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR - TE 141599

ANALISA UNJUK KERJA PROXY SERVER MENGGUNAKAN ALGORITMA LRU, LFU, DAN GDSF

Adi Kurniawan
NRP 07111445000058

Dosen Pembimbing
Dr. Ir. Achmad Affandi DEA.
Dr. Istas Pratomo, S.T., M.T.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT - TE 141599

PROXY SERVER WORKLOAD ANALYSIS USING LRU, LFU, AND GDSF ALGORITHM

Adi Kurniawan
NRP 07111445000058

Supervisor
Dr. Ir. Achmad Affandi DEA.
Dr. Istas Pratomo, S.T., M.T.

ELECTRICAL ENGINEERING DEPARTMENT
Electrical Technology Faculty
Sepuluh Nopember Technology Institute
Surabaya 2018

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan tugas akhir saya dengan judul **"ANALISA UNJUK KERJA PROXY SERVER MENGGUNAKAN ALGORITMA LRU, LFU, DAN GDSF"** adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juli 2018

Adi Kurniawan
NRP. 07111445000058

[Halaman ini sengaja dikosongkan]

ANALISA UNJUK KERJA PROXY SERVER MENGGUNAKAN ALGORITMA LRU, LFU, DAN GDSF

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan Untuk
Memperoleh Gelar Sarjana Teknik Elektro
Pada
Bidang Studi Telekomunikasi Multimedia
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui

Dosen Pembimbing I,	Dosen Pembimbing II,
Dr. Ir. Achmad Affandi DEA.	**Dr. Istas Pratomo, ST., MT.**
NIP. 196510141990021001	**NIP. 197903252003121001**

SURABAYA
JULI, 2018

[Halaman ini sengaja dikosongkan]

ANALISA UNJUK KERJA PROXY SERVER MENGGUNAKANALGORITMA LRU, LFU, DAN GDSF

Adi Kurniawan
07111445000058

Dosen Pembimbing I : Dr. Ir. Achmad Affandi DEA
Dosen Pembimbing II : Dr. Istas Pratomo, ST., MT.

ABSTRAK

Sebuah komputer atau jaringan lokal yang terhubung dengan internet dapat mengakses dan mendapatkan data apa saja dari internet. Hal ini menimbulkan beberapa kemungkinan, antara lain masuknya data yang tidak dinginkan ke dalam jaringan. Oleh karena itu sebuah proxy server dapat digunakan sebagai manajemen trafik. Proxy server merupakan aplikasi gateway yang digunakan untuk mengendalikan trafik yang masuk ke dalam jaringan. Proxy server memiliki beberapa fungsi, diantaranya adalah caching. Proses caching ini berguna untuk optimasi bandwidth yang digunakan dalam sebuah jaringan.Terdapat beberapa aturan peyimpanan caching yang ada di dalam proxy server, aturan-aturan tersebut berdasarkan pada algoritma page replacement yang sudah ada, diantara algoritma tersebut adalah adalah LRU, LFU atau LFUDA, dan GDSF.

Tujuan dari Tugas Akhir ini adalah untuk menganalisa unjuk kerja dari masing-masing algortima yang sudah disebutkan serta melakukan beberapa perbandingan sehingga dapat diketahui algoritma mana yang paling efektif.

Melaui simulasi dengan beberapa skenario yang suda ditentukan. Didapatkan bahwa algoritma GDSF adalah yang paling efektif karena memiliki hit rate terbesar yaitu 4,177. Sedangkan secara keseluruhan dengan melihat lima parameter, algoritma LFUDA dikatakan paling efektif melihat pada 3 dari 5 parameter, yaitu byte hit read, read rata-rata, dan swap I/O, sedangkan GDSF paling efektif melihat pada 2 dari 5 parameter, yaitu hit rate dan CPU time.

Kata Kunci : Proxy, *Page Replacement*, LRU, LFU, LFUDA, GDSF

[Halaman ini sengaja dikosongkan]

PROXY SERVER WORKLOAD ANALYSIS USINGLRU, LFU, AND GDSF ALGORITHM

Adi Kurniawan
07111445000058

Supervisor I : Dr. Ir. Achmad Affandi DEA
Supervisor II : Dr. Istas Pratomo, ST., MT.

ABSTRACT

A computer or local network that have conection with internet can get and access anything from the internet. But this could also cause another possibility, unwated information get through the local network. A proxy server could be a solution of the problem as proxy server can manage traffic that pass it's network. Proxy server also have another function, caching. Caching can be used to optimize bandwidth. There are some cache storage rules that is used by proxy server. Most of the rules are based on popular algorithm, such as LRU, LFU or LFUDA, and GDSF.

The purpose of this final project is to analize The workload of each algorithm and do some comparation so we can get the most effective of all.

Through simulation, it is known that from five output parameter, LFUDA is te most effective if we see byte hit rate, read average, and swap I/O. GDSF is the most effective if we see hit rate and CPU time

Keyword Proxy, *Page Replacement*, LRU, LFU, LFUDA, GDSF

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Alhamdullilah, puji syukur penulis ucapkan kehadirat Illahi Rabbi, Allah SWT atas segenap rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan penulisan buku Tugas Akhir dengan judul **"Analisa Unjuk Kerja Proxy Server Menggunakan Algoritma LRU, LFU, dan GDSF"**. Tugas akhir merupakan salah satu syarat yang harus dipenuhi untuk menyelesaikan program studi Strata-1 pada Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember Surabaya.

Pada kesempatan ini penulis ingin menyampaikan ucapan terimakasih dan penghargaan setingi-tingginya kepada :

Bapak, Ibu, Saudara, beserta segenap keluarga yang selalu memberikan dukungan.

Bapak Dr. Ir. Achmad Affandi DEA dan bapak Dr. Istas Pratomo, ST. MT. selaku Dosen Pembimbing atas arahan dan bimbingannya selamapengerjaan tugas akhir ini.

Bapak Dr. Ir. Endroyono, DEA selaku Koordinator Bidang Studi Telekomunikasi Multimedia Jurusan T. Elektro ITS.

Bapak Dr. Ardyono Priyadi, ST. M.Eng selaku Ketua Jurusan Teknik Elektro ITS.

Bapak dan Ibu dosen Jurusan Teknik Elektro khususnya bidang studi Telekomunikasi Multimedia. Semoga menjadi ilmu yang bermanfaat dan barokah.

Kawan-kawan seperjuangan LJ Gasal 2014.

Penulis menyadari bahwa dalam tugas akhir ini masih belum sempurna dan dapat dikembangkan lagi kedepanya. Semoga Tugas Akhir ini bisa menjadi bahan referensi dan memberikan manfaat bagi pengembangan dunia telekomunikasi.

Surabaya, Juli 2018
Penulis

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

HALAMAN JUDUL ..... **i**
PERNYATAAN KEASLIAN ..... **v**
LEMBAR PENGESAHAN ..... **vii**
ABSTRAK ..... **ix**
ABSTRACT ..... **xi**
KATA PENGANTAR ..... **xiii**
DAFTAR ISI ..... **xv**
DAFTAR GAMBAR ..... **xix**
DAFTAR TABEL ..... **xxiii**
BAB 1 PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan 3
1.5 Metodologi 3
1.6 Sistematika Pembahasan 4
1.7 Relevansi 4
BAB 2 TEORI PENUNJANG **5**
2.1 Jaringan Komputer 5
2.1.1 Sejarah Jaringan Komputer 5
2.1.2 Model Jaringan Peer to Peer 5
2.1.3 Model Jaringan Server – Client 6
2.1.4 Internet 7
2.1.5 TCP/IP 8
2.1.6 IP Address 9
2.2 Linux 9
2.2.1 Distro Linux 10
2.2.2 Baris Perintah Linux 12
2.2.3 Linux Bash 13
2.3 Proxy Server 14
2.3.1 Connection sharing 14
2.3.2 Filtering 15
2.3.3 Caching 15
2.3.4 Algoritma Page Replacement 17
2.3.4.1 LRU (Least Recent Used) 17
2.3.4.2 LFUDA (Least Frequently Used with Dynamic Aging) 18

2.3.4.3 GDSF (Greedy-Dual Size Frequency).....18
2.4 Aplikasi Proxy Server.....19
2.4.1 Parameter Konfigurasi Proxy.....19
2.4.2 Storage Scheme.....21
2.4.3 Log Proxy.....22
2.4.4 Hit dan Byte Hit ratio.....29
2.4.5 Library Proxy.....30
2.5 Cacti Network Monitoring.....30
2.5.1 Project NLANR.....31
2.6 Aplikasi Automasi.....32
2.7 Performa Proxy Server.....33
BAB 3 PEMODELAN DAN IMPLEMENTASI..... 35
3.1 Pemodelan.....35
3.1.1 Variabel Utama.....35
3.1.2 Variabel Sekunder.....36
3.1.2.1 Jumlah Client.....36
3.1.2.2 Kondisi Trafik.....38
3.1.3 Skenario dan Generate Data.....49
3.1.4 Monitoring dan Collecting data.....51
3.1.5 Output Parameter.....52
3.2 Implementasi.....54
3.2.1 Instalasi Proxy server.....55
3.2.2 Instalasi Aplikasi Monitoring Proxy Populer.....57
3.2.3 Instalasi Tool Tambahan Pada Proxy.....65
3.2.4 Instalasi Aplikasi Automasi Trafik Pada Client.....68
3.2.5 Instalasi Cron Scheduler pada Server.....71
3.2.6 Instalasi Tool Live Monitoring Tambahan Pada Server.....72
3.2.7 Konfigurasi Proxy Server.....73
3.2.8 Konfigurasi NLANR Monitoring Proxy.....76
3.2.9 Pembuatan dan Kofigurasi Aplikasi Monitoring berbasis BASH.....80
3.2.10 Konfigurasi Browser Client.....88
3.2.11 Automasi Trafik Pada Client.....91
3.2.12 Scheduling.....93
3.2.12.1 Scheduling Pada Client.....93
3.2.12.2 Scheduling Pada Server.....97
3.3 Pengujian Sistem.....99
3.3.1 Distribusi Trafik dan Periode Simulasi.....99

3.3.2 Skenario Pengujian 100
3.3.3 Parameter Pengukuran 103
BAB 4 HASIL DAN ANALISA DATA.....105
4.1 Hasil Simulasi 105
4.2 Analisa Skenario 109
4.3 Analisa Perbandingan Algoritma 111
BAB 5 KESIMPULAN DAN SARAN 115
5.1 Kesimpulan 115
5.2 Saran..... 116
DAFTAR PUSTAKA.....117
BIOGRAFI PENULIS 119
LAMPIRAN.....121
6.1 Script getAccessLogStat.sh..... 121
6.2 Script getAccessByteLogStat.sh 124
6.3 Script squidClientCountAIO.sh..... 127
6.4 Script squidClientUtilizationStat.sh 130

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

Gambar 2.1 Contoh topologi jaringan peer to peer 6
Gambar 2.2 Contoh topologi jaringan server - client 7
Gambar 2.3 Perbandingan arsitektur TCP/IP dengan OSI dan DARPA .8
Gambar 2.4 Maskot linux 10
Gambar 2.5 Logo Debian 10
Gambar 2.6 Logo Ubuntu 11
Gambar 2.7 Logo Fedora 11
Gambar 2.8 Contoh tampilan terminal di linux 12
Gambar 2.9 Alur akses pada proxy server 14
Gambar 2.10 Proxy sebagai gateway connection sharing 15
Gambar 2.11 Filtering obyek di dalam proxy 15
Gambar 2.12 Mekanisme caching pada proxy server 16
Gambar 2.13 Contoh algoritma LRU 18
Gambar 2.14 Squid proxy 19
Gambar 2.15 Struktur cache directory pada ufs 21
Gambar 2.16: Contoh format access.log 22
Gambar 2.17 Contoh eksekusi squidclient 30
Gambar 2.18 Cacti 30
Gambar 2.19 AutoIt 32
Gambar 3.1 Diagram alur pemodelan 35
Gambar 3.2 Jaringan I, satu server satu client 38
Gambar 3.3 Jaringan II, satu server dua client 38
Gambar 3.4 Jaringan III, satu server tiga client 38
Gambar 3.5 Tampilan satubahasa.com 39
Gambar 3.6 Tampilan cerpenmu.com 40
Gambar 3.7 Tampilan squid-cache.org 40
Gambar 3.8 Tampilan dunia21.tv 41
Gambar 3.9 Tampilan bioskopkeren.ws 41
Gambar 3.10 Tampilan detik.com 42
Gambar 3.11 Tampilan jendelasatra.com 42
Gambar 3.12 Tampilan ufroog.com 43
Gambar 3.13 Tampilan edipsw.com 43
Gambar 3.14 Tampilan primbon.com 44

Gambar 3.15 Tampilan sugeng.id 44
Gambar 3.16 Tampilan 1031gen.fm.....45
Gambar 3.17 Tampilan himit.pens.ac.id.....45
Gambar 3.18 Tampilan mangapark.net/manga/naruto/s3/c700.....46
Gambar 3.19 Tampilan news.liputan6.com 46
Gambar 3.20 Tampilan mangapark.net/manga/bleach/s3/c686.....47
Gambar 3.21 Melihat page size suatu halaman.....48
Gambar 3.22 Flowchart generate data.....51
Gambar 3.23 Flowchart penentuan HIT dan MISS.....52
Gambar 3.24 Flowchart penentuan Byte HIT dan Byte MISS.....53
Gambar 3.25 Instalasi awal squid3.....55
Gambar 3.26 Proses instalasi squid3 sedang berjalan 55
Gambar 3.27 Direktori /etc/squid3.....56
Gambar 3.28 Direktori /var/log/squid3.....56
Gambar 3.29 Isi access.log 57
Gambar 3.30 Instalasi awal cacti dan dependenciesnya 57
Gambar 3.31 Konfigurasi mysql server pada instalasi awal cacti 58
Gambar 3.32 Membuat password pada mysql server 58
Gambar 3.33 Konfirmasi password pada mysql server 58
Gambar 3.34 Konfigurasi php pada instalasi cacti 59
Gambar 3.35 Konfigurasi awal web server pada cacti.....59
Gambar 3.36 Opsi web server pada instalasi awal cacti 59
Gambar 3.37 Kotak dialog awal setting admin db pada cacti 60
Gambar 3.38 Konfirmasi konfigurasi database untuk cacti 60
Gambar 3.39 Mendefinisikan password untuk admin db cacti.....60
Gambar 3.40 Konfirmasi password untuk admin db cacti 61
Gambar 3.41 Cacti installation guide 61
Gambar 3.42 Opsi tipe instalasi cacti 61
Gambar 3.43 Review konfigurasi dependencies cacti 62
Gambar 3.44 Halaman login cacti.....62
Gambar 3.45 Instalasi snmp, snmpd, dan mibs downloader 63
Gambar 3.46 Konfigurasi snmp.conf 63
Gambar 3.47 Konfigurasi file snmpd.conf.....64
Gambar 3.48 Konfigurasi squid.conf untuk integrasi snmp.....64
Gambar 3.49 Output testing integrasi snmp dan squid proxy 65
Gambar 3.50 Instalasi squidclient 65

Gambar 3.51 Dialog box awal AutoIt 68
Gambar 3.52 Dialog license agreement AutoIt..... 69
Gambar 3.53 Dialog select defaut option AutoIt 69
Gambar 3.54 Dialog pilih komponen AutoIt 70
Gambar 3.55 Dialog pilih direktori AutoIt..... 70
Gambar 3.56 Dialog akhir instalasi AutoIt..... 71
Gambar 3.57 Instalasi cron..... 71
Gambar 3.58 Instalasi ccze..... 72
Gambar 3.59: Live checking log squid via terminal 72
Gambar 3.60 Tampilan direktori tempat squid.conf berada 75
Gambar 3.61 Status service squid..... 76
Gambar 3.62 Membuat direktori swap squid..... 76
Gambar 3.63 Tampilan default cacti monitoring 77
Gambar 3.64 Form add device cacti 77
Gambar 3.65 List device cacti 78
Gambar 3.66 Form add device cacti 78
Gambar 3.67 Add new graph pada cacti 79
Gambar 3.68 Tambah graph tree baru cacti..... 79
Gambar 3.69 Add graph tree baru berhasil..... 79
Gambar 3.70 Add tree items cacti..... 80
Gambar 3.71 Graph tree dan tree item cacti 80
Gambar 3.72 Grafik monitoring pada cacti 80
Gambar 3.73 Flowchart algoritma getAccessLogStat.sh..... 81
Gambar 3.74 Flowchart algoritma getAccessLogByteStat.sh 82
Gambar 3.75 Flowchart algoritma squidClientCountAIOStat.sh..... 82
Gambar 3.76 Flowchart algoritma squidClientUtilizationStat.sh 83
Gambar 3.77 Contoh file .sh..... 84
Gambar 3.78 Contoh mengubah privilege file..... 84
Gambar 3.79 Contoh hasil output program 85
Gambar 3.80 Contoh data pada AccessLogStatistic.log..... 85
Gambar 3.81 Contoh data pada AccessLogByteStatistic.log..... 86
Gambar 3.82 Contoh data pada squidClientAIOStatistic.log 86
Gambar 3.83 Contoh data pada squidClientUtilStatistic.log 87
Gambar 3.84 Contoh data file csv output program 87
Gambar 3.85 Shortcut setting pada mozilla firefox 88
Gambar 3.86 Opsi network proxy..... 88

Gambar 3.87 Contoh setting detail proxy 89
Gambar 3.88 Locaproxy firefox addon.....89
Gambar 3.89 List IP proxy pada locaproxy 90
Gambar 3.90 Konfigurasi pada locaproxy 90
Gambar 3.91 Menon-aktifkan caching pada mozilla firefox.....91
Gambar 3.92 File automasi pada masing-masing client 91
Gambar 3.93 Windows task scheduler 94
Gambar 3.94 Membuat task scheduler baru.....94
Gambar 3.95 Opsi trigger task scheduler.....95
Gambar 3.96 Setting periode task scheduler.....95
Gambar 3.97 Pilihan action pada task scheduler.....96
Gambar 3.98 Scheduling untuk file random_traffic01.au3.....96
Gambar 3.99 Review task scheduler 97
Gambar 3.100 Daftar task pada scheduler 97

DAFTAR TABEL

Tabel 2.1 Operator pada bash 13
Tabel 2.2 HTTP status code 23
Tabel 2.3 Request method 25
Tabel 2.4 Data hierarki code 26
Tabel 2.5 Tag proses pengiriman 27
Tabel 2.6 Tag asal request 28
Tabel 2.7 Tag pembuatan tipe obyek 28
Tabel 2.8 Tag asal respons 29
Tabel 2.9 Tag respon error spesifik 29
Tabel 3.1 Nilai pada cache_replacement policy dan memory_replacement_policy 36
Tabel 3.2 Detail spesifikasi hardware node 37
Tabel 3.3 Detail spesifikasi software node 37
Tabel 3.4 Jenis-jenis website dan kriterianya 39
Tabel 3.5 Pembagian jenis website 47
Tabel 3.6 Daftar halaman url pada masing-masing kondisi 49
Tabel 3.7 Contoh Output squidclient mgr:squidaio_counts 66
Tabel 3.8 Contoh output squidclient cache_object://localhost/counters 67
Tabel 3.9 Isi file squid.conf 73
Tabel 3.10 Data output Program monitoring berbasis BASH 85
Tabel 3.11 Contoh script automasi 92
Tabel 3.12 Cron scheduling untuk jaringan I 98
Tabel 3.13 Cron scheduling untuk jaringan II 98
Tabel 3.14 Cron scheduling untuk jaringan III 98
Tabel 4.1 Output parameter terkait performa trafik 105
Tabel 4.2 Output parameter terkait performa hardware 107
Tabel 4.3 Statistik nilai output setelah simulasi 109
Tabel 4.4 Urutan hit rate terbesar 111
Tabel 4.5 Urutan byte hit rate terbesar 112
Tabel 4.6 Urutan read rata-rata terbesar 112
Tabel 4.7 Urutan rasio swap I/O terkecil 113

Tabel 4.8 Urutan CPU time terkecil 113
Tabel 4.9 Urutan nilai unused file descriptors 114

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Dengan semakin berkembangnya teknologi, banyak aspek dalam kehidupan manusia yang menjadi lebih mudah dan cepat, salah satunya adalah proses pertukaran informasi. Dulu, proses pertukaran informasi jarak jauh masih mengunakan media manual/fisik. Proses pertukaran ini tentu saja memakan waktu yang lama dan memiliki resiko kehilangan yang cukup besar. Namun dengan perkembangan teknologi dan jaringan komputer, pengiriman informasi menjadi jauh lebih mudah. Informasi-informasinya ada dikonversi menjadi informasi digital dan dikumpulkan untuk kemudian menjadi sebuah paket informasi. Dari sisi efisiensi, tentu saja hal ini jauh lebih baik karena paket informasi yang masih berbentuk fisik memakan ruang yang sangat besar. Media pengiriman informasi pun juga mengalami perkembangan, informasi digital tidak dikirim menggunakan media transportasi konvensional, tapi melalui suatu jaringan komputer yang saling terhubung satu sama lain. Jaringan ini tidak hanya melingkupi wilayah lokal dan dekat saja, namun saat ini sudah bisa menjangkau seluruh dunia. Salah satu bentuk teknologi jaringan tersebut adalah internet.

Sebuah komputer atau jaringan lokal yang terhubung dengan internet bisa mengakses dan mendapatkan apa saja dari internet. Hal ini dapat menyebabkan kerugian, diantaranya masuknya data dan informasi yang tidak diinginkan ke dalam jaringan lokal, serta penggunaan bandwidth yang besar akibat masuknya informasi yang tidak diinginkan tersebut. Oleh karena itu, perlu adanya sistem manajemen yang mengatur penggunaan akses internet pada suatu komputer atau jaringan lokal. Salah satu sistem yang banyak digunakan adalah sistem proxy server.

Proxy server adalah sebuah server yang memiliki fungsi meneruskan atau menolak internet content request yang diajukan oleh client dengan atau tanpa suatu kondisi. Proxy server bekerja dengan mendengarkan request dari client internal dan mengirim request tersebut ke jaringan eksternal seolah-olah proxy server itu sendiri yang menjadi client. Pada waktu proxy server menerima respon dan server publik, ia

memberikan respon tersebut ke client yang asli seolah-olah ia public server.

Pada proxy server terdapat sebuah mekanisme caching. Mekanisme caching adalah proses penyimpanan secara temporer terhadap url-url yang telah diakses ke dalam suatu direktori khusus. Jika komputer client suatu jaringan melakukan content request terhadap suatu url maka proxy server akan malakukan proses pengecekan terlebih dahulu. Jika url masih berada dalam direktori penyimpanan, maka proxy server tidak perlu meneruskan request tersebut menuju internet dan cukup mengambil informasi tersebut dari penyimpanan. Hal ini tentu sangat menguntungkan, terutama dari dalam hal kecepatan akses dan bandwidth.

Pada mekanisme caching di dalam proxy server, banyak algoritma yang bisa dipakai, antara lain algoritma LRU, LFU, dan GDSF. Algoritma tersebut dipakai dalam proses penghapusan data di dalam direktori caching. Penggunaan algoritma ini tentu saja berpengaruh pada kinerja proxy server itu sendiri. Karena itu perlu adanya suatu proses perbadingan dalam penggunaan masing-masing algoritma yang ada sehingga bisa memilih algoritma yang paling efisien.

1.2 Rumusan Masalah

Permasalahan yang dibahas dalam tugas akhir ini adalah:

1. Bagaimana mekanisme kerja proses caching pada proxy server Bagaimana penerapan masing-masing algoritma LRU, LFU, dan GDSF pada proxy server
2. Bagaimana cara membangkitkan trafik yang sama pada proxy server yang menggunakan masing-masing algoritma tersebut
3. Parameter apa saja yang digunakan untuk penilaian kinerja proxy server

1.3 Batasan Masalah

Batasan masalah dari Tugas Akhir ini adalah:

1. Perbandingan unjuk kerja berdasarkan cache replacement Algorithm
2. Hanya membandingkan tiga algoritma, yaitu LRU, LFU, dan GDSF
3. Menggunakan proxy server pada jaringan tertentu saja

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah untuk membandingkan performa dan kinerja dari suatu proxy server untuk tiap-tiap caching algoritm yang digunakan, dalam hal ini algoritma LRU, LFUDA, dan GDSF, sehingga bisa mendapatkan algoritma yang paling pas sesuai dengan kondisi yang berlaku. Hal ini tentu dapat meningkatkan efisiensi kecepatan akses dan bandwidh dari proxy server.

1.5 Metodologi

Metodologi yang digunakan dalam tugas akhir ini terdiri dari beberapa tahap sebagai berikut:

1. Studi literatur
   Mencari dan mempelajari beberapa buku, jurnal maupun literatur ilmiah lainnya mengenai teori yang dibutuhkan untuk menyelesaikan tugas akhir. Selain itu juga melakukan *internet* research, berupa project-project terkait jaringan proxy mengingat perkembangan proxy server sendiri sebagian besar melalui sistem open source yang melibatkan banyak orang dan komunitas.
2. Pemodelan
   Membuat pemodelan berdasarkan variable-variable, baik variable utama mapun variabel sekunder yang digunakan untuk mengetahui unjuk kerja proxy server. Pemodelan tersebut dimulai dari model konfigurasi hingga model trafik yang akan dibangkitkan pada jaringan.
3. Implementasi
   Proses implementasi konfigurasi hingga simulasi trafik dilakukan sesuai dengan model yang sudah dibuat
4. Monitoring
   Termasuk ke dalam monitoring adalah proses collecting data dan pengamatan trafik selama proses implementasi berjalan. Data yang dikumpulkan adalah data parameter yang dibutuhkan sesuai dengan perencanaan
5. Analisis Data
   Pada tahap ini dilakukan data processing terhadap nilai parameter yang berhasil didapatkan setelah proses simulasi berjalan.
6. Penarikan Kesimpulan
   Analisis data berdasarkan dasar terori dan rujukan yang ada sehingga pada akhirnya berhasil ditarik sebuah kesimpulan.

1.6 Sistematika Pembahasan

Pembahasan dalam tugas akhir ini akan dibagi dalam lima bab dengan sistematika sebagai berikut:

Bab I **Pendahuluan**
Pada bab ini dijelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi, sistematika pembahasan dan relevansi.

Bab II **Teori Penunjang**
Pada bab ini dijelaskan beberapa tinjauan pustaka mengenai jaringan secara umum hingga spesifik berupa jaringan proxy beserta konsep monitoring jaringan.

Bab III **Pemodelan dan Implementasi**
Pada bab ini akan dilakukan implementasi konfigurasi lalu dilakukan proses simulasi trafik sehingga menghasilkan parameter-parameter yang akan digunkana dalam proses analisa data.

Bab IV **Analisis Hasil**
Bab ini berisi pengolahan data parameter-parameter yang dihasilkan selama proses simulasi jaringan yang nantinya akan diproses.

Bab V **Kesimpulan dan Saran**
Pada bab ini berisi kesimpulan dan saran.

1.7 Relevansi

Hasil dari yang didapat pada tugas akhir ini diharapkan dapat memberi solusi sekaligus sebagai referensi untuk dipakai pada jaringan proxy sehingga bisa memaksimalkan penggunaan resource, baik berupa bandwidth maupun kinerja komputer pada jaringan.

BAB 2
TEORI PENUNJANG

2.1 Jaringan Komputer

Jaringan komputer (Computer Network) merupakan sebuah sistem yang terdiri dari dua atau lebih komputer yang saling terhubung dan berkomunikasi melalui mekanisme tertentu. Komputer-komputer tersebut saling terhubung dan berkomunikasi sehingga bisa melakukan proses transfer informasi, baik itu satu arah maupun dua arah.

Jaringan komputer lalu kemudian berkembang. dari Jaringan yang tadinya hanya beberapa komputer dengan lingkup kecil (LAN) meluas hingga menjadi jaringan komputer yang menghubungkan antar wilayah yang luas (WAN). Dengan meluasnya wilayah jaringan ini pun media trasnmisi juga ikut berkembang, dari yang semula menggunakan media kabel menjadi nirkabel berbasis gelombang elektromagnetik yang mampu mencakup wilayah yang lebih luas.

2.1.1 Sejarah Jaringan Komputer

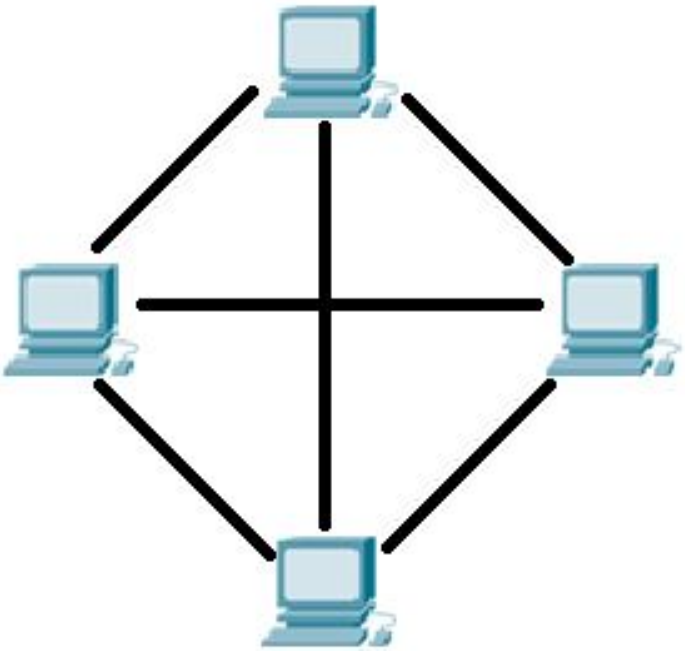
Pada tahun 1940-an, grup riset dari Universitas Harvard yang diketuai oleh Howard Aiken mengembangkan sebuah proyek komputer MODEL I, yaitu proyek pengembangan model Batch Processing, dimana beberapa program bisa dijalankan dalam sebuah komputer dengan kaidah antrian.

Kemudian pada tahun 1950-an dimana teknologi komputer lebih berkembang, model distrubusi proses berdasarkan waktu, disebut juga Time Sharing System, mulai diciptakan dan diaplikasikan pertama kali oleh Departemen Pertahanan Amerika yang nantinya dikembangkan lebih lanjut oleh DARPA (Defence Advance Research project Agency) sehingga menjadi cikal bakal jaringan internet.

2.1.2 Model Jaringan Peer to Peer

Pada sebuah jaringan, masing-masing komputer mempunyai fungsi tambahan jika dibandingkan dengan komputer tunggal. Selain fungsi mengirim dan/atau menerima informasi dari komputer lain, terdapat fungsi tambahan lain yang membedakan komputer tersebut dengan komputer tunggal (stand-alone). Pada jaringan peer to peer, masing-masing node kedudukannya sama sehingga memiliki

fungsionalitas sama tanpa ada yang dikurangi atau ditambah pada salah satu komputer.



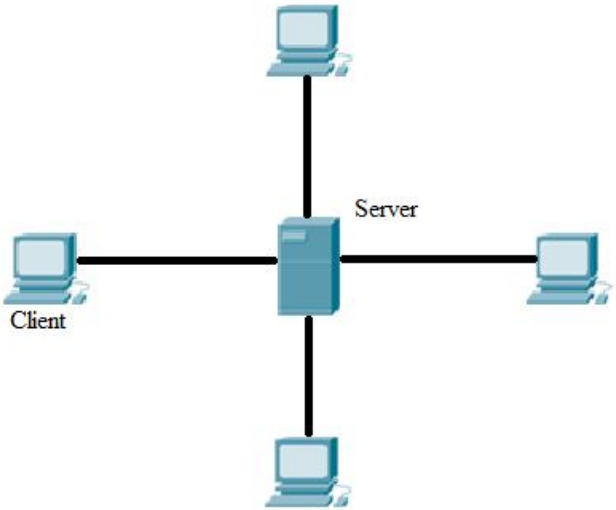
Gambar 2.1 Contoh topologi jaringan peer to peer

Dengan menggunakan model jaringan peer to peer, instalasi jaringan dapat dilakukan dengan mudah. Setiap komputer di dalam jaringan memiliki hak yang sama dan jika terdapat masalah pada satu komputer maka komputer lain dalam satu jaringan tidak akan terpengaruh. Akan tetapi, tingkat keamanan pada jaringan ini rendah dan tidak cocok jika memiliki banyak komputer yang terhubung dalam jaringan.

2.1.3 Model Jaringan Server – Client

Pada jaringan Server – Client. Terdapat komputer yang berperan sebagai client dan komputer yang berperan sebagai server. Dimana server memiliki dan menyediakan sumber daya sedangkan client hanya bisa menggunakan sumber daya tersebut. Server berfungsi sebagai pusat akses di dalam sebuah jaringan sehingga semua client

yang ingin melakukan proses transfer informasi harus melalui komputer server.



Gambar 2.2 Contoh topologi jaringan server - client

Jaringan server – client memiliki kelebihan diantaranya faktor keamanan yang lebih terjamin karena kontrol pusat, termasuk aksesibilitas, ada di dalam komputer server. Akan tetapi dengan fungsi server sebagai kontrol utama maka jika terdapat malfungsi pada server maka jaringan komputer akan mengalami kegagalan.

2.1.4 Internet

Internet merupakan jaringan komputer yang dibentuk oleh Departemen Pertahanan Amerika Serikat pada tahun 1969, melalui proyek lembaga ARPA yang mengembangkan jaringan yang dinamakan ARPANET (Advanced Research Project Agency Network). Pada mulanya ARPANET hanya menghubungkan 4 tempat saja yaitu Stanford Research Institute, University of California, Santa Barbara, University of Utah, di mana mereka membentuk satu jaringan terpadu pada tahun 1969. Kemudian karena berkembang dengan pesat, ARPANET dipecah manjadi dua, yaitu "MILNET" untuk keperluan militer dan "ARPANET". Gabungan kedua jaringan akhirnya dikenal dengan nama DARPA Internet, yang kemudian disederhanakan menjadi Internet.

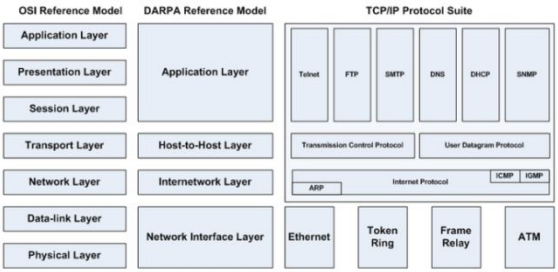
Saat ini, di dalam internet terdapat berbagai macam layanan yang dapat digunakan seperti email, streaming, file sharing, dll dengan banyak protokol yang digunakan. Protokol tersebut antara lain IP, TCP, UDP, DNS, PPP, SLIP, ICMP, POP3, IMAP, SMTP, HTTP, HTTPS, SSH, Telnet, FTP, LDAP, dan SSL.

2.1.5 TCP/IP

TCP/IP berkembang bersamaan dengan internet melalui proyek ARPANET. TCP/IP awalnya merupakan protokol umum yang digunakan pada internet untuk menghubungkan jaringan-jaringan dengan berbagai macam sistem operasi. Namun dengan semakin pesatnya perkembangan internet yang lalu menggunakan DNS, TCP/IP masih tetap digunakan sebagai protokol standar dalam menghubungkan jaringan, hal ini tentu saja karena sifat routable TCP/IP yang mampu bekerja dalam berbagai macam sistem.

Arsitektur TCP/IP dengan model OSI yang terdiri dari tujuh layer, sedangkan arsitektur TCP/IP berbasis pada model DARPA yang terdiri dari empat layer, yaitu

1. Layer Aplikasi Mencakup protokol FTP, SMTP, DNS, DHCP, SNMP, dan protokol lainnya.
2. Layer Host to HostTerdiri atas Transmission Control Protocol (TCP) dan User Datagram Protocol (UDP).
3. Layer InternetworkMenggunakan protokol IP, ARP, dan lain-lain
4. Layer Network InterfacePada layer ini banyak teknoloi yang bisa digunakan, antara lain Ethernet, Token Ring, Frame Relay, dan ATM



Gambar 2.3 Perbandingan arsitektur TCP/IP dengan OSI dan DARPA

2.1.6 IP Address

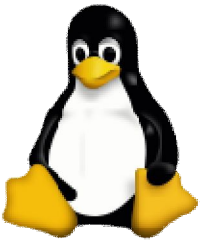
IP address digunakan untuk mengidentifikasi interface jaringan pada host dari suatu komputer. Dengan adanya IP address, komputer dapat terhubung dan saling bertukar informasi. IP address didefinisikan dengan sekelompok bilangan biner 32 bit yang dibagi menjadi 4 bagian yang masing-masing bagian itu terdiri dari 8 bit, dimana angka pada masing-masing bit tersebut adalah angka 1 dan 0.

IP address terdiri dari dua bagian yaitu bagian network ID dan host ID. Network ID menunjukkan ID alamat jaringan tempat host-host berada sedangkan host ID adalah bagian yang menunjukkan host itu berada. IP address dibagi menjadi kelas-kelas yang masing-masing mempunyai kapasitas jumlah IP yang berbeda-beda, yaitu

- Kelas A
  IP address kelas A terdiri dari 8 bit untuk network ID dan sisanya 24 bit digunakan untuk host ID, sehingga IP address kelas A digunakan untuk jaringan dengan jumlah host sangat besar. Pada bit pertama berikan angka 0 sampai dengan 127.Contoh 120.31.45.18
- Kelas B
  IP address kelas B terdiri dari 16 bit untuk network ID dan sisanya 16 bit digunakan untuk host ID, sehingga IP address kelas B digunakan untuk jaringan dengan jumlah host tidak terlalu besar. Pada 2 bit pertama berikan angka 10 sehingga bit awal IP tersebut mulai dari 128 – 191.Contoh 150.70.45.18
- Kelas C
  IP address kelas C terdiri dari 24 bit untuk network ID dan sisanya 8 bit digunakan untuk host ID, sehingga IP address kelas C digunakan untuk jaringan untuk ukuran kecil. Kelas C biasanya digunakan untuk jaringan Local Area Network atau LAN. Pada 3 bit pertama berikan angka 110 sehingga bit awal IP tersebut mulai dari 192 – 223.Contoh 192.168.1.1

2.2 Linux

Linux, yang diperkenalkan oleh Linus Torvalds, merupakan Sistem Operasi berbasis UNIX yang bersifat open source, bebas dikembangkan dan didistribusikan oleh siapa saja. Karena sifatnya yang bebas, sampai saat ini Linux banyak dikembangkan sehingga mempunyai banyak varian yang disebut Linux distribution, Linux distro.



Gambar 2.4 Maskot linux

2.2.1 Distro Linux

Sampai saat ini terdapat beberapa distro linux populer yang dikembangkan oleh komunitas-komunitas open source di seluruh dunia, diantaranya

A. Debian

Debian diperkenalkan pertama kali oleh Ian Murdock, mahasiswa Universitas Purde Amerika, pada tanggal 16 Agustus 1993. Proyek Debian merilis versi 0.9x pada tahun 1994 dan 1995. Pengalihan arsitektur ke selain i386 dimulai pada tahun 1995. Versi 1.x dimulai tahun 1996. Di akhir tahun 2000, proyek debian melakukan perubahan dalam archive dan managemen rilis. Serta pada tahun yang sama para pengembang memulai konferensi dan workshop tahunan "debconf".

Gambar 2.5 Logo Debian

Fitur yang menonjol dari Debian adalah sistem manajemen APT, repositori dengan jumlah paket yang banyak, kebijakan paket yang ketat, dan kualitas rilis yang terjaga. Praktik ini memungkinkan pemutakhiran yang sederhana antar rilis, begitupun untuk penghapusan paket.

B. Ubuntu

Ubuntu merupakan varian Linux yang berbasis Debian. ama Ubuntu berasal dari filosofi dari Afrika Selatan yang berarti "kemanusiaan kepada sesama". Proyek Ubuntu resmi disponsori oleh Canonical Ltd. yang merupakan sebuah perusahaan yang dimiliki oleh pengusaha Afrika Selatan Mark Shuttleworth.

Gambar 2.6 Logo Ubuntu

Ubuntu pertama kali dirilis pada 20 Oktober 2004. Semenjak itu, Canonical telah merilis versi Ubuntu yang baru setiap 6 bulan sekali. Setiap rilis didukung selama 18 bulan untuk pembaruan sistem, keamanan, dan kesalahan (bug). Setiap 2 tahun sekali (versi xx.04 dengan x angka genap) akan mendapatkan Long Term Support selama 3 tahun untuk desktop dan 5 tahun untuk edisi server.

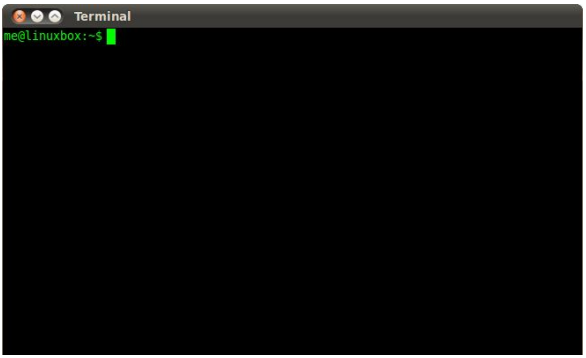
C. Fedora

Fedora merupakan distro linux berbasis RPM dan Yum yang dikembangkan oleh Fedora project. Fedora Project juga mendristribusikan beberapa variant tertentu dari Fedora, diantaranya Fedora Spins.

Gambar 2.7 Logo Fedora

2.2.2 Baris Perintah Linux

Sebagai sistem operasi yang umum digunakan pada server, Linux banyak dioperasikan melalui baris perintah yang dimasukan ke dalam terminal. Bahkan di beberapa kasus, sistem operasi linux yang digunakan pada server full berbasis baris perintah tanpa menggunakan tool GUI (Graphical User Interface).



Gambar 2.8 Contoh tampilan terminal di linux

Terdapat beberapa baris perintah yang sering digunakan pada terminal linux, diantaranya

1. `su` adalah perintah untuk masuk/login sebagai suatu user
2. `cd` adalah perintah untuk berpindah direktori
3. `ls` adalah perintah untuk menampilkan isi dari direktori yang sedang aktif di dalam terminal
4. `mkdir` adalah perintah untuk membuat direktori baru
5. `rmdir` adalah perintah untuk menghapus suatu direktori
6. `chown` digunakan untuk megnubah deskripsi owner sebuah file atau direktori
7. `chmod` digunakan untuk menambah atau mengurangi ijin akses suatu file atau direktori
8. `apt-get` adalah salah satu perintah berbasis debian dan turunannya. Digunakan untuk mengelolah package dan dependencies-nya

2.2.3 Linux Bash

Bash (Bourne Again Shell) adalah salah satu bahasa pemrograman berbasis Unix yang diciptakan oleh Brian Fox bersamaan dengan proyek GNU yang bertujuan untuk pengembangan sistem operasi linux. Bash dikembangkan dari pendahulunya yaitu Bourne Shell. Bash berjalan pada platform text windows, dimana command dijalankan pada sebuah terminal.

Pemrograman bash shell menggabungkan perintah-perintah untuk memilih suatu kondisi, memproses suatu I/O, looping, dan membuat fungsi-fungsi yang dapat dijalankan user. Bash juga dapat dieksekusi melalui sebuah file. Pada pemrograman bash, file yang dieksekusi adalah file dengan ekstensi .sh. command dan sintaks dasar pada bash hampir sama dengan sheel linux lain.

Beberapa poin penting yang harus diperhatikan pada bash diantaranya

1. Variabel

   terdapat beberapa macam variabel seperti Environment Variable, Positional variable, dan User Defined variable.

2. I/O

   perintah umum yang sering digunakan dalam operasi I/O pada bash adalah echo, print, printf, dan read.

3. Operator

   berikut adalah detail operator integer, logika, dan string pada bash

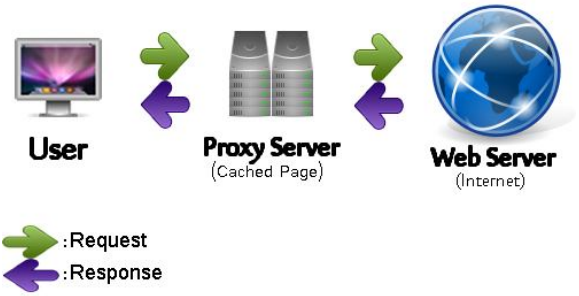
Tabel 2.1 Operator pada bash

No.	Operator	Jenis	Keterangan
1.	-eq	Integer	Sama dengan
2.	-ne	Integer	Tidak sama dengan
3.	-lt	Integer	Lebih kecil
4.	-le	Integer	Lebih kecil atau sama dengan
5.	-gt	Integer	Lebih besar
6.	-ge	Integer	Lebih besar atau sama dengan

7.	-z	String	Null atau kosong
8.	==	String	Sama dengan
9.	!=	String	Tidak sama dengan
10.	-o	Logika	OR
11.	-a	Logika	AND
12.	!	Logika	NOT

2.3 Proxy Server

Proxy server adalah sebuah server yang memiliki hak dan kemampuan untuk meneruskan atau menolak internet content request yang diajukan oleh client dengan atau tanpa suatu kondisi. Proxy server bekerja dengan mendengarkan request dari client internal dan mengirim request tersebut ke jaringan eksternal seolah-olah proxy server itu sendiri yang menjadi client. Pada waktu proxy server menerima respon dan server publik, server memberikan respon tersebut ke client yang asli seolah-olah sebagai public server.

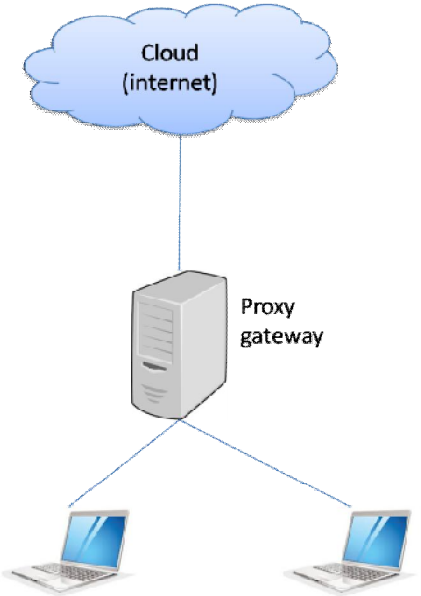


Gambar 2.9 Alur akses pada proxy server

Sebuah proxy server memiliki fungsi-fungsi tambahan dibandingkan dengan server biasa dalam sebuah jaringan. Diantara fungsi-fungsi tersebut adalah connection sharing, filtering, dan caching

2.3.1 Connection sharing

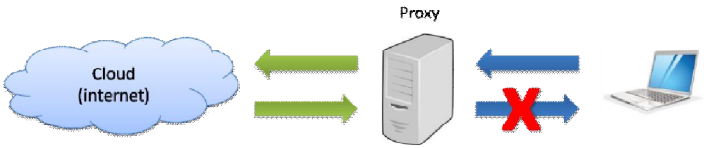
Proxy server dapat berfungsi sebagai gateway antara jaringan lokal dengan jaringan luar lainnya. Gateway berfungsi sebagai titik sambung antara client di dalam jeringan lokal dengan jaringan luar, termasuk internet, oleh karena itu gateway digunakan sebagai connection sharing di dalam jaringan lokal.



Gambar 2.10 Proxy sebagai gateway connection sharing

2.3.2 Filtering

Fungsi utama proxy adalah filtering, untuk meneruskan atau menolak paket informasi yang masuk ke dalam jaringan. Fungsi filtering dapat diatur berdasarkan kriteria dan parameter tertentu.



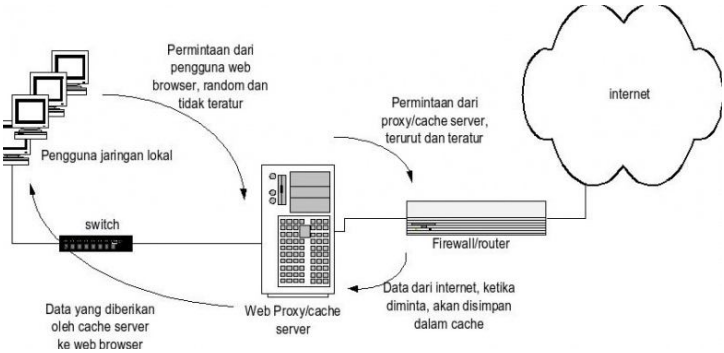
Gambar 2.11 Filtering obyek di dalam proxy

2.3.3 Caching

Salah satu fungsi dasar dari suatu proxy server adalah caching. Proxy server memiliki mekanisme penyimpanan obyek-obyek yang sudah pernah diminta dari server-server di internet, biasa disebut caching. Karena itu, proxy server yang juga melakukan proses caching juga biasa disebut cache server.

Mekanisme caching akan menyimpan obyek-obyek yang merupakan hasil permintaan dari dari para pengguna, yang didapat dari internet. Karena proxy server bertindak sebagai perantara, maka proxy server mendapatkan obyek-obyek tersebut lebih dahulu dari sumbernya untuk ekmudian diteruskan kepada peminta yang sesungguhnya. Dalam proses tersebut, proxy server juga sekaligus menyimpan obyek-obyek tersebut untuk dirinya sendiri dalam ruang disk yang disediakan (cache).

Dengan demikian, bila suatu saat ada pengguna yang meminta suatu layanan ke internet yang mengandung obyek-obyek yang sama dengan yang sudah pernah diminta sebelumnya, yaitu yang sudah ada dalam cache, maka proxy server akan dapat langsung memberikan obyek dari cache yang diminta kepada pengguna, tanpa harus meminta ulang ke server aslinya di internet. Bila permintaan tersebut tidak dapat ditemukan dalam cache di proxy server, baru kemudian proxy server meneruskan atau memintakannya ke server aslinya di internet.



Gambar 2.12 Mekanisme caching pada proxy server

Proses caching ini juga tidak kelihatan bagi pengguna (transparan), karena bagi pengguna tidak tampak siapa sebenarnya yang yang memberikan obyek yang dimintanya, apakah proxy server yang mengambil dari cache-nya atau server asli di internet. Dari sisi pengguna, semua akan nampak sebagai balasan langsung dari internet.

Ada dua jenis metode caching, yaitu pasif dan aktif. Seperti telah kita ketahui, object yang disimpan bisa saja mencapai expired, untuk memeriksanya dilakukan validasi. Jika validasi ini dilakukan setelah ada permintaan dari klien, metode ini disebut pasif. Pada caching aktif, cache server mengamati object dan pola perubahannya. Misalkan pada sebuah object didapati setiap harinya berubah setiap jam 12 siang

dan pengguna biasanya membacanya jam 14, maka cache server tanpa diminta klien akan memperbaharui object tersebut antara jam 12 dan 14 siang, dengan cara update otomatis ini waktu yang dibutuhkan pengguna untuk mendapatkan object yang fresh akan semakin sedikit.

Pada kondisi tertentu, kapasitas penyimpanan akan terkuras habis oleh object. Namun cache server mempunyai beberapa metode penghapusan untuk menjaga kapasitas tetap terjaga, sesuai dengan kopnfigurasi yang telah ditetapkan. Penghapusan ini didasarkan pada umur dan kepopuleran, semakin tua umur object akan tinggi prioritasnya untuk dihapus. Dan juga untuk object yang tidak popular akan lebih cepat dihapus juga.

2.3.4 Algoritma Page Replacement

Sebuah proxy server mempunyai kapasitas penyimpanan obyek yang terbatas, oleh karena itu dibutuhkan suatu mekanisme penggantian obyek yang disimpan berdasarkan aturan-aturan tertentu. Proxy server menggunakan beberapa algoritma page replacement yang sudah ada, diantaranyaadalah LRU, FLUDA, dan GDSF

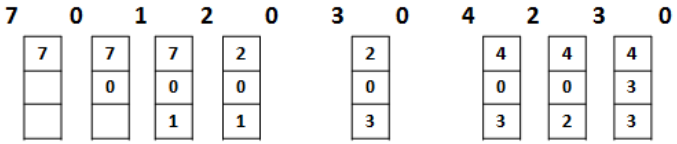
2.3.4.1 LRU (Least Recent Used)

Algoritma page replacement LRU mengikuti aturan penghapusan halaman yang sudah tidak digunakan dalam jangka waktu yang paling lama.Terdapat dua cara implementasi algoritma LRU, yaitu

1. Counter

   Cara ini dilakukan dengan menggunakan counter atau logical clock. Setiap halaman memiliki nilai yang pada awalnya diinisialisasi dengan 0. Ketika mengakses ke suatu halaman baru, nilai pada clock di halaman tersebut akan bertambah 1. Semakin sering halaman itu diakses, semakin besar pula nilai counter-nya dan sebaliknya. Untuk melakukan hal itu dibutuhkan extra write ke memori. Selain berisi halaman-halaman yang sedang di-load, memori juga berisi tentang counter masing-masing halaman. Halaman yang diganti adalah halaman yang memiliki nilai clock terkecil, yaitu halaman yang paling jarang diakses. Kekurangan dari cara ini adalah memerlukan dukungan tambahan counter pada hardware.
2. Stack

Cara ini dilakukan dengan menggunakan stack yang menandakan halaman-halaman yang berada di memori. Setiap kali suatu halaman diakses, akan diletakkan di bagian paling atas stack. Apabila ada halaman yang perlu diganti, maka halaman yang berada di bagian paling bawah stack akan diganti sehingga setiap kali halaman baru diakses tidak perlu mencari kembali halaman yang akan diganti. Dibandingkan pengimplementasian dengan counter, cost untuk mengimplementasikan algoritma LRU dengan menggunakan stack akan lebih mahal karena seluruh isi stack harus di-update setiap kali mengakses halaman, sedangkan dengan counter, yang dirubah hanya counter halaman yang sedang diakses, tidak perlu mengubah counter dari semua halaman yang ada.



Gambar 2.13 Contoh algoritma LRU

2.3.4.2 LFUDA (Least Frequently Used with Dynamic Aging)

Algoritma LFUDA melakukan penghapusan obyek/halaman yang jarang digunakan dan berdasarkan waktu kadaluarga. Terdapat proses analisa setiap file, jika memiliki tingkat freuensi akses kecil maka mendapat prioritas untuk dihapus.

Apabila terdapat lebih dari satu file dengan tingkat frekuensi akses yang sama kecil maka dilanjutkan melihat kepada waktu kadaluarsanya. Jika nilai kadalurasanya lebih tinggi maka obyek tersebut akan dihapus.

2.3.4.3 GDSF (Greedy-Dual Size Frequency)

Algoritma GDSF melalukan penghapusan obyek berdasarkan ukuran dan frekuensi akses. Obyek dengan ukuran yang lebih besar aka mendapat prioritas untuk dihapus. Dan jika ditemukan lebih dari satu obyek maka lanjut kepada prosec checking frekuensi akses obyek tersebut. Obyek yang frekuensi aksesnya lebih rendah akan dihapus.

2.4 Aplikasi Proxy Server

Squid proxy adalah salah satu aplikasi proxy server populer dan banyak digunakan oleh jaringan yang mempunyai fitur proxy di dalamnya. Squid proxy awalnya dikembangkan sebagai project Harvest Object Cache oleh Duane Wessels di University of Colorado. Lalu dilanjutkan oleh University of California, San Diego dengan bantuan dana dari National Science Foundation. Squid proxy merupakan software dibawah lisensi GNU General Public License yang memiliki sifat open source sehingga biasanya squid proxy berjalan pada platform sistem operasi berbasis linux. Namun saat ini squid proxy yang berjalan pada platform windows juga sudah tersedia

Sebagimana sebuah jaringan proxy bekerja, setelah suatu server terisi dengan squid proxy, web browser dapat dikonfigurasi sebagai proxy HTTP server sehingga masing-masing komputer dalam jaringan bisa mendapatkan akses informasi dari jaringan luar.

Gambar 2.14 Squid proxy

Squid proxy memiliki fitur-fitur khusus yang dapat digunakan pada sebuah jaringan dengan proxy. Diantaranya adalah squid proxy bisa mengaktifkan fitur anonim, squid juga bisa menambah atau mengurangi beberapa field header tertentu saat proses request HTTP dijalankan.

2.4.1 Parameter Konfigurasi Proxy

Squid proxy memiliki beberapa parameter tertentu yang dapat dikonfigurasi sehingga menghasilkan output tertentu. Walapun terdapat perbedaan setting konfigurasi pada squid proxy dengan platform linu mapun non-linux (windows), namun parameter-parameter ini biasanya terdapat dalam sebuah file bernama squid.conf. user atau admin yang ingin merubah konfigurasi bisa dengan mengubah isi konfigurasi yang terdapat dalam file sqid.conf.

Beberapa paramater yang ada dan dikonfigurasi di dalam file squid.conf antara lain

1. ACL
   acl digunakan untuk mendefinisikan rule akses pada sebuah proxy server. Rule akses ini mencakup banyak hal, mulai dari IP source address, domain, port, method, dll. Oleh karena itu format pamataer acl biasanya disertai dengan detail parameter tambahan. Misalkan saja

`acl jarlokal src 192.168.1.0/24`	⇔	Mendefinisikan IP souce
`acl situs_porno dstdomain .porno.com`	⇔	Mendefiniskan domain tujuan
`acl filter_kata url_regex porn`	⇔	Mendefinisikan kata atau sebagian kata
`acl jam_kerja time MTWHF 08:00-16:00`	⇔	Mendefinisikan waktu atau periode waktu

2. http_access
   digunakan untuk memberikan hak terhadap acl yang sudah didefinidikan. Hak akses tersebut bisa berupa diperboleh (allow) maupun ditolak (deny). contoh http_access allow jarlokal
3. http_port
   digunakan untuk mengatur port mana yang akan dipakai oleh aplikasi squid proxy
4. cache_dir
   secara default, squid hanya menyimpan cache object pada memory komputer saja.Agar obyek cache juga bisa disimpan di dalam media penyimpanan disk komputer maka aktifkan dengan menggunakan konfigurasi cache_dir
5. minimum_object_size dan maximum_onject_size
   digunakan untuk mengatur batasan minimum dan maximum object satuan file yang dapat disimpan di dalam disk komputer
6. cache_swap_low dan cache_swap_high
   berupakan limit penanda apakah rule penghapusan obyek cache mulai dilaksanakan
7. memory_replacement_policy
   merupakan setting algoritma page replacement untuk proses penyimpanan cache pada RAM komputer
8. cache_replacement_policy
   merupakan setting algoritma page replacement untuk proses penyimpanan cache pada media penyimpanan disk komputer

9. refresh_pattern
   merupakan parameter yang penting dalam proses tuning dan optimasi cache. Squid proxy sendiri menyediakan konfigurasi minimal refresh_pattern untuk mengoptimalkan proses tuning. Berikut konfigurasinya

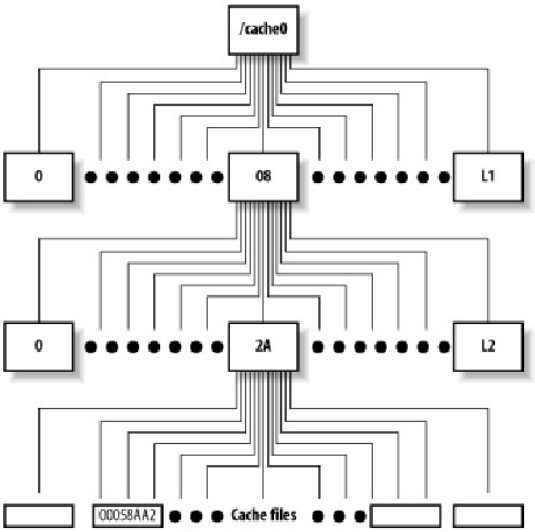
refresh_pattern ^ftp:	1440	20%	10080
refresh_pattern ^gopher:	1440	0%	1440
refresh_pattern -i (/cgi-bin/|\?)	0	0%	0
refresh_pattern .	0	20%	4320

10. snmp_incoming_address dan snmp_outgoing address
    untuk mengatur hak akses terhadap SNMP yang masuk atau keluar di dalam jaringan proxy

2.4.2 Storage Scheme

Setting skema penyimpanan termasuk dalam parameter cache_dir yang ada di dalam squid.conf. Ada beberapa macam logic skema penyimpanan yang ada di dalam squid proxy, antara lain

1. ufs
   merupakan setting default storage scheme ketika parameter cache_dir diaktifkan

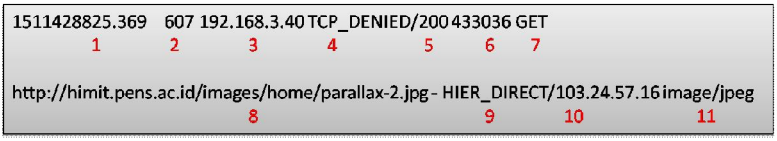


Gambar 2.15 Struktur cache directory pada ufs

2. aufs
   merupakan pengembahangan dari skema ufs. Pada aufs, I/O proses tidak dijalankan oleh main squid process. Setiap sebuah I/O (read, close, open, dll) proses terhadap file dijalankan. Squid akan membuat task proses sendiri (terpisah) dan akan mengembalikan sbuah return value code jika proses tersebut sudah selesai
3. diskd
   sama seperti aufs, hanya saja tidak ada task tersendiri melainkan inter-process communication. Diskd digunakan jika cache_dir mendefinisikan sebuah penyimpanan eksternal
4. null
   sama dengan menyatakan bahwa squid tidak menyimpan obyek cache

2.4.3 Log Proxy

Semua record transaksi HTTP yang ada di server tersimpan di dalam access.log dengan format data tertentu. Berikut adalah penjelasan tentang format record data pada access.log



Gambar 2.16: Contoh format access.log

1 ⇔ Timestamp
2 ⇔ Respons time
3 ⇔IP address client
4 ⇔ Code status
5 ⇔ HTTP status
6 ⇔ Transfer size
7 ⇔Request method
8 ⇔ URL
9 ⇔ Hierarki
10 ⇔ IP adress server
11 ⇔ Content type

Tabel 2.2 HTTP status code

No.	HTTP Status Code	Keterangan
1.	000	Trafik UDP
2.	100	Meneruskan/melanjutkan
3.	101	Pergantian protokol
4.	012	Memproses
5.	200	Status OK
6.	201	Dibuat
7.	202	Diterima
8.	203	Data yang diproses tidak mempunyai keamanan
9.	204	Konten kosong
10.	205	Reset konten
11.	206	Konten diterima sebagian
12.	207	Multi status
13.	300	Terdapat opsi
14.	301	Dipindahkan permanan
15.	302	Dipindahkan sementara
16.	303	Melihat yang lain
17.	304	Tidak berubah
18.	305	Menggunakan proxy
19.	307	Redirect sementara
20.	400	Permintaan salah
21.	401	Membutuhkan persetujuan
22.	402	Prabayar
23.	403	Dilarang mengakses
24.	404	Tidak ditemukan
25.	405	Method tidak diperbolehkan

26.	406	Tidak disetujui
27.	407	Memerlukan proxy aproval
28.	408	Time Out
29.	409	Konflik
30.	410	Hilang
31.	411	Dibutuhkan akses lebih
32.	412	Permintaan gagal
33.	413	Entitas terlalu besar
34.	414	Permintaatn URL terlalu besar
35.	415	Tipe media unsupport
36.	416	Length request tidak terhitung
37.	417	Ekspektasi gagal
38.	422	Entitas tidak diproses
39.	424	Terknci
40.	433	Tidak terproses
41.	500	Internal server error
42.	501	Tidak terpasang
43.	502	Bad gateway
44.	504	Gateway timeout
45.	505	Unsupported HTTP version
46.	507	Proses penyimpanan data client error
47.	600	Squid header error
48.	601	Header squid terlalu besar
49.	602	Squid configuration error
50.	603	Autorisasi tidak dikenal

Sementara itu, request code adalah informasi method dan menentukan obyek dan konten informasi apa yang bisa disimpan di dalam cache proxy. Request method terdiri dari

Tabel 2.3 Request method

No.	Method	Definisi	Probabilitas
1.	GET	HTTP/0.9	V
2.	HEAD	HTTP/1.0	X
3.	POST	HTTP/1.0	-
4.	PUT	HTTP/1.1	X
5.	DELETE	HTTP/1.1	X
6.	TRACE	HTTP/1.1	X
7.	OPTION	HTTP/1.1	X
8.	CONNECT	HTTP/1.Ir3	X
9.	ICP_QUERY	Squid	X
10.	PURGE	Squid	X
11.	PROPFIND	Rfc2518	-
12.	PROPACTH	Rfc2518	-
13.	MKCOL	Rfc2518	X
14.	COPY	Rfc2518	X
15.	MOVE	Rfc2158	X
16.	LOCK	Rfc2518	X
17.	UNLOCK	Rfc2518	X

Hierarki code merupakan aktifitas squid dalam merespon request yang datang. Berikut adalah data Hierarki code pada proxy server

Tabel 2.4 Data hierarki code

No.	Hierarki Code	Keterangan
1.	NONE	Digunakan untuk TCP HIT, TCP FAILURES, request cachemgr, dan request UDP. Tidak mempunyai hierarki informasi
2.	DIRECT	Obyek didapatkan langsung dari server asal
3.	SIBLING_HIT	Obyek diambil dari sibling cache (dengan respon UDP HIT)
4.	PARENT_HIT	Obyek diambil dari parent cache (dengan respon UDP HIT)
5.	DEFAULT_PARENT	Tidak ada query ICP, parent merupakan opsi default
6.	FIRST_UP_PARENT	Obyek diambil dari parent pertama dari keseluruhan list
7.	NO_PARENT_DIRECT	Obyek diambil dari server asal, tanpa parent
8.	FIRST_PARENT_MISS	Obyek diambil dari parent yang paling cepat merespons
9.	CLOSEST_PARENT_MISS	Parent sudah terpilih
10.	CLOSEST_PARENT	Parent berdasarkan pengukuran RTT
11.	CLOSEST_DIRECT	Respon RTT lebih cepat

		daripada parent yang lain
12.	NO_DIRECT_FAIL	Gagal request karena firewall
13.	SOURCE_FASTEST	Respon ping paling cepat
14.	ROUND_ROBIN_PARENT	Tidak ada respon ICP karena konfigurasi round robin
15.	CACHE_DIGEST_HIT	Cache digest memprediksi hit
16.	CD_PARENT_HIT	Parent terpilih karena prediksi cache digest
17.	CD_SIBLING_HIT	Ssibling terpilih karena prediksi cache digest
18.	NO_CACHE_DIGEST_DIRECT	-
19.	CARP	Peer dipilih oleh CARP
20.	PINNED	Koneksi server ditandai oleh NTLM
21.	ORIGINAL_DST	Dibatasi oleh IP destination yang disediakan provider
22.	INVALID_CODE	Invalid code

Code status terdiri dari lima tag, dan suatu code status bisa terdiri dari kombinasi-kombinasi tag. Tag tersebut yaitu

1. Tag yang menjelaskan proses pengiriman

Tabel 2.5 Tag proses pengiriman

No.	Code	Keterangan
1.	TCP	Request HTTP port
2.	UDP	Request ICP port
3.	NONE	Tidak ada respons

2. Tag yang menjelaskan asal suatu request

Tabel 2.6 Tag asal request

No.	Code	Keterangan
1.	CLIENT	Cache wajib menvalidasi obyek
2.	IMS	Client mengirim validasi ulang (kondisional)
3.	ASYNC	Digunakan secara internal oleh squid
4.	SWAPFAIL	Recek gagal, meminta kopi ulang dari server
5.	REFRESH	Validasi ulang terkirim kepada server
6.	SHARED	-
7.	REPLY	HTTP replay dari server atau peer

3. Tag yang menjelaskan proses pembuatan tipe obyek

Tabel 2.7 Tag pembuatan tipe obyek

No.	Code	Keterangan
1.	NEGATIVE	Menghasilkan error respons, misal 404 Not Found
2.	STALE	Disebabkan oleh stale-while-revalidate atau stale-if-error
3.	OFFLINE	Respon diterima saat offline mode
4.	INVALID	Invalid request
5.	FAIL	Validasi ulang gagal
6.	MODIFIED	Validasi ulang menghasilkan obyek yang dimodifikasi
7.	UNMODIFIED	Validasi ulang menghasilkan obyek yang sama (tidak dimodifikasi)
8.	REDIRECT	Menghasilkan HTTP redirect respons

4. Tag yang menjelaskan asal respons

Tabel 2.8 Tag asal respons

No.	Code	Keterangan
1.	HIT	Respon terdirim kepada obyek cache lokal
2.	MEM	Respon berasal dari memory cache
3.	MISS	Respon berupa network response object
4.	DENIED	Diblok oleh access control
5.	NOFETCH	Respon ICP spesial, menandakan service masih hidup/berjalan
6.	TUNNEL	Adanya binary tunnel

5. Tang yang menjelaskan respon error spesifik

Tabel 2.9 Tag respon error spesifik

No.	Code	Keterangan
1.	ABORTED	Respon tidak selesai karena dibatalkan
2.	TIMEOUT	Respon tidak selesai karena timeout
3.	IGNORED	Respon diabaikan

2.4.4 Hit dan Byte Hit ratio

Hit ratio merupakan perbandingan antara jumlah request yang mendapat respon dari cache squid dengan jumlah request total dari client.

$$BR = \frac{\sum HIT}{\sum \mathrm{Re}\,quest} x100\% \quad \text{.....(2.1)}$$

Sedangkan byte hit ratio adalah jumlah byte request yang diambil dari cache proxy server dibandingkan dengan jumlah byte request total yang diminta client.

$$BHR = \frac{\sum ByteHIT}{\sum Byte\,\mathrm{Re}\,quest} x100\% \quad \text{.....(2.2)}$$

Disimpulkan juga bahwa semakin besar hit ratio maka semakin efektif sebuah proxy server karena dapat menghemat trafik jaringan.

2.4.5 Library Proxy

Squidclient merupakan salah satu library proxy yang juga merupakan additional tool yang disediakan oleh squid. Squidclient secara default tidak terinstall automatis saat user menginstall squid proxy server. Oleh karena itu user melakukan proses instalasi terpisah. Eksekusi squidclient sendiri bisa menggunakan banyak parameter dan masing masing parameter akan menghasilkan putput yang berbeda.

```
[root@centos7 builder]# squidclient mgr:info | tail  -n 20
        Total size:             68704 KB
Memory accounted for:
        Total accounted:        14034 KB  20%
        memPool accounted:      14034 KB  20%
        memPool unaccounted:    54670 KB  80%
        memPoolAlloc calls:    821240
        memPoolFree calls:     852351
File descriptor usage for squid:
        Maximum number of file descriptors:   16384
        Largest file desc currently in use:     77
        Number of file desc currently in use:   42
        Files queued for open:                   0
        Available number of file descriptors: 16342
        Reserved number of file descriptors:   100
        Store Disk files open:                   0
Internal Data Structures:
          330 StoreEntries
          330 StoreEntries with MemObjects
          317 Hot Object Cache Items
            0 on-disk objects
[root@centos7 builder]#
```

Gambar 2.17 Contoh eksekusi squidclient

2.5 Cacti Network Monitoring

Cacti adalah salah satu tool monitoring jaringan yang populer. Cacti merupakan aplikasi open source berbasis web. Salah satu kelebihan Cacti adalah adanya fugnsi RRDTool sebagai penympadanan data dan pembuatan grafik. Cacti menyediakan pengumpulan data yang cepat, pola grafik advanced, metoda perolehan multiple data, dan fitur pengelolaan user.

Gambar 2.18 Cacti

Beberapa poin penting terkait RDDTool yang ada di dalam aplikasi monitoring jaringan Cacti adalah

1. Data Retrieval
   Data dikumpulkan dengan Poller yang dieksekusi oleh Operating System. Interval pengumpulan data atau dengan kata lain eksekusi Poller dapat di atur melalui fasilitas penjadwalan yang ada di dalam sistem operasi
2. Data Storage
   Data yang telah dikumpulkan oleh Poller, selanjutnya akan disimpan secara teratur di bawah /rra. Untuk proses ini, cacti menggunakan Round Robin Database (RRD) dimana data akan ditata dalam urutan waktu (time-series). Data yang dapat berupa trafik jaringan, suhu mesin, server load average, mounting load dan lainnya berbentuk file berekstensi .rra dan selanjutnya siap dipresentasikan dalam bentuk grafik
3. Data Presentation
   Data-data yang tertata dalam /rra akan di presentasikan dalam grafik dan ditampilkan oleh webserver yang kita gunakan. Cacti juga menyediakan halaman pengaturan grafik untuk mempermudah proses memanajemen.

Untuk menjalankan aplikasi Cacti, dibutuhkan beberapa tool dependencies, diantaranya

1. RRDTool
2. httpd/apache
3. php
4. php-mysql
5. php-snmp
6. mysql
7. mysql-server
8. net-snmp

2.5.1 Project NLANR

Sebagai aplikasi dengan platform open source, Cacti banyak dikembangkan oleh komunitas-komunitas yang ada di seluruh dunia. Salah satunya adalah proyek NLANR. NLANR merupakan pengembangan template Cacti khusus untuk proxy server yang dikerjakan oleh Pete de Zwart pada tahun 2012.

Paramater-paramater yang ada di dalam NLANR meliputi

1. FQDN cache entries
2. IP cache entries
3. IP cache statistics
4. Request ratios
5. Service times
6. Clients
7. DNS statistics
8. File descriptors
9. FQDN cache statistics
10. HTTP statistics
11. HTTP traffic
12. ICP traffic
13. Number of objects
14. Read statistics
15. Server statistics
16. Server traffic
17. Swap size

2.6 Aplikasi Automasi

AutoIt adalah aplikasi pemrograman berbasis BASIC yang digunakan untuk pross automasi pada sistem operasi windows. AutoIt menggunakan kombinasi keystrokes, pergerakan pointer, dan kontrol windows untuk membuat task automasi.

Gambar 2.19 AutoIt

AutoIt aka mengeksekusi file yang berisi kumpulan baris dan kombinasi perintah untuk proses automasi. AutoIt terbaru adalah AutoIt versi 3 dimana file program AutoIt memiliki ekstensi .au3.

2.7 Performa Proxy Server

Sebagaimana jaringan pada umumnya, performa suatu proxy server diukur dari efisiensi pada segi trafik dan hardware. Pada segi trafik, tingkat efektivitas dinyatakan oleh beberapa parameter diantaranya

- Hit rate
- Byte hit rate.
- Nilai swap input dan output

Sedangkan pada sisi hardware, poin utama yang dilihat adalah kinerja utilisasi CPU dari server proxy. Dan parameter yang dilihat antara lain

- Proses I/O, terutama read rata-rata yang merupakan perbandingan nilai read dan check_callback pada proxy server

  $$read\ rata - rata\ (AR) = \frac{read}{check_callback} \quad \text{.....(2.3)}$$

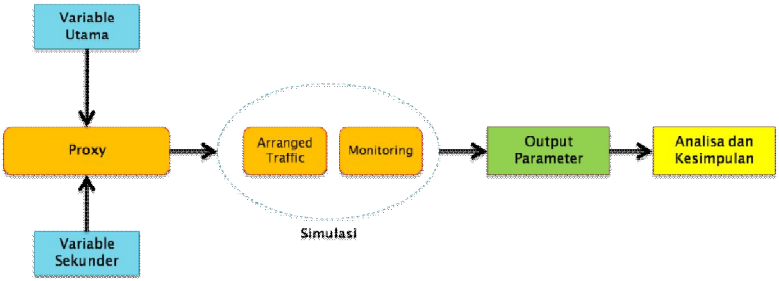
- CPU time
- File descriptors

[Halaman ini sengaja dikosongkan]

BAB 3
PEMODELAN DAN IMPLEMENTASI

3.1 Pemodelan

Tujuan dari tugas akhir ini adalah untuk menganalisa unjuk kerja masing-masing algoritma dan menganalisanya sehingga ditemukan pilihan algoritma paling optimal sesuai dengan kondisi jaringan yang disepakati. Skema tahapan pemodelan digambarkan melalui diagram berikut



Gambar 3.1 Diagram alur pemodelan

Berdasarkan diagram alur pemodelan, maka ada beberapa poin yang perlu diperhatikan, diantaraya

1. Variabel utama, yaitu page replacement
2. Variabel sekunder, variabel-variabel lain yang berpengaruh dalam proses simulasi
3. Pengujian skenario dan generate data, untuk menghasilkan traffic yang diatur sedemikian hingga, perlu adanya skenario yang jelas
4. Monitoring, yaitu proses monitoring hingga collecting data
5. Output parameter yang diukur

3.1.1 Variabel Utama

Variabel utama dalam penelitian ini adalah algoritma page replacement. Di dalam squid proxy server, terdapat dua input parameter yang mengunakan algoritma page raplacement sebagai nilainya. Dua parameter tersebut adalah `memory_replacement_policy` dan `cache_replacement_policy`.

Di dalam squid proxy, Algoritma page replacement LRU, LFU, dan GDSF sudah diterjemahkan dan termasuk dalam opsi yang disediakan oleh squid proxy dan juga terdapat optimasi khusus dengan adanya opsi nilai heap. Setelah menyesuaikan dengan format yang ada di dalam squid proxy maka didapatkan range nilaiuntuk variabel utama adalah sebagai berikut

Tabel 3.1 Nilai pada cache_replacement policy dan memory_replacement_policy

Parameter Konfigurasi Proxy	**LRU**		**LFU**	**GDSF**
memory_replacement_policy	LRU	heap LRU	heap LFUDA	heap GDSF
cache_replacement_policy	LRU	heap LRU	heap LFUDA	heap GDSF

3.1.2 Variabel Sekunder

Terdapat dua variabel sekunder yang digunakan pada penelitian kali ini, variabel sekunder tersebut adalah jumlah client dan kondisi trafik. Variabel jumlah client pada penelitian kali ini berkisar antara satu s.d. tiga client sehingga terdapat 3 jaringan lokal sedangkan variabel kondisi trafik dipengarhi oleh jenis halaman yang diminta oleh client.

3.1.2.1 Jumlah Client

Di dalam variabel jumlah client, tidak hanya perlu dianalisa mengenai jumlah node client, tapi juga perlu adanya analisa kebutuhan sistem. Kebutuhan-kebutuhan tersebut antara lain

1. Perangkat keras masing-masing node

   Spesifikasi perangkat keras, terutama pada CPU masing-masing node yang ada di dalam jaringan tentu boleh berbeda, harus ada spesifikasi minimum agar proses simulasi berjalan lancar. Berdasarkan sumber daya yang tersedia maka ditentukan bahwa spesifikasi masing-masing node adalah sebagai berikut

Tabel 3.2 Detail spesifikasi hardware node

Node	Spesifikasi
SERVER	Intel Core i3 RAM 4 Gb 500 Gb Disk Capacity
CLIENT 1	Intel Core i5 RAM 4 Gb 700 Gb Disk Capacity
CLIENT 2	Intel Core i5 RAM 4 Gb 300 Gb Disk capacity
CLIENT 3	AMD E-300 RAM 4 Gb 300 Disk Capacity

2. Perangkat lunak masing-masing node

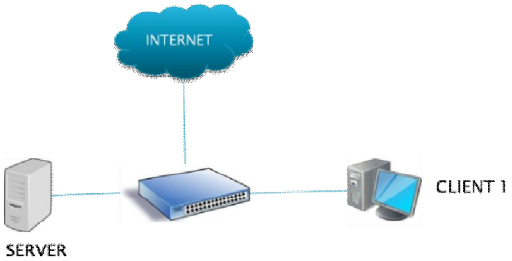
   Spesifikasi perangkat lunak, dalam hal ini adalah sistem operasi, yang ada di dalam masing-masing node yang ada di jaringan adalah sebagai berikut

Tabel 3.3 Detail spesifikasi software node

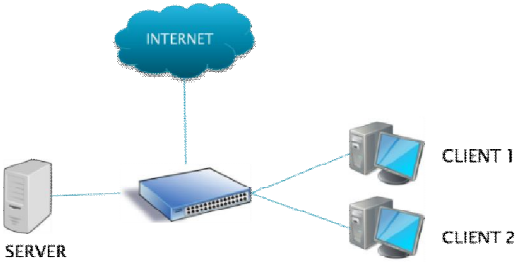
Node	Spesifikasi
SERVER	Linux Ubuntu 14.04 32 Bit
CLIENT 1	Windows 7 32 Bit
CLIENT 2	Windows 7 64 Bit
CLIENT 3	Windows 7 32 Bit

3. Arsitektur dan topologi jaringan

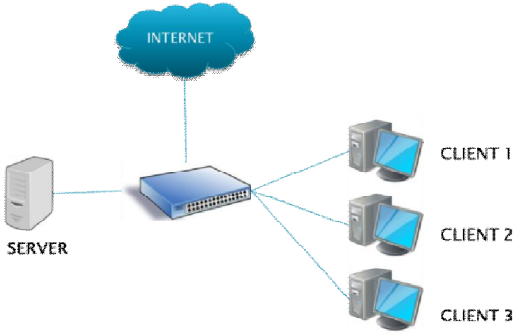
   Berdasarkan nilai maksimal jumlah client yang hanya sebanyak 3 node maka ditentukan bahwa jaringan yang dibuat merupakan jaringan kelas C dengan topologi jaringan berupa topologi star. Sehingga terdapat total 3 jaringan dan simulasi trafik nantinya akan dijalankan pada masing-masing jaringan.



Gambar 3.2 Jaringan I, satu server satu client



Gambar 3.3 Jaringan II, satu server dua client



Gambar 3.4 Jaringan III, satu server tiga client

3.1.2.2 Kondisi Trafik

Pada sebuah jaringan proxy, client melakukan request halaman internet kepada server dan server akan menerurkan request tersebut. Nantinya server akan mendapatkan respon dan meneruskan hasil respon tersebut menuju client. Halaman yang diminta oleh client sangat

bervariasi, tergantung dari kebutuhan dan habis dari client yang ada di dalam sebuah jaringan.

Pada penelitian kali ini, request-request halaman website dari client akan dibedakan menjadi dua jenis. Dari jenis-jenis website ini kemudian memunculkan kondisi-kondisi yang bisa saja muncul di dalam suatu jaringan.

Berikut adalah dua jenis website yang ada berdasarkan beberapa kriteria perbedaannya

Tabel 3.4 Jenis-jenis website dan kriterianya

	Website A	**Website B**
Tampilan umum	Didominasi konten berupa tulisan	Didominasi konten berupa gambar atau visul
Ukuran halaman	< 2 Mb	> 2 Mb

Trafik yang muncul pada jaringan merupakan trafik yang sudah diatur sedemikian hingga. Untuk mengatur trafik tersebut, salah satu caranya adalah dengan mengatur url halaman yang diminta oleh client sesuai dengan list sampel halaman yang sudah ada. Sampel-sampel halaman URL yang akan digunakan untuk memunculkan trafik pada jaringan antara lain

1. http://www.satubahasa.com/

Gambar 3.5 Tampilan satubahasa.com

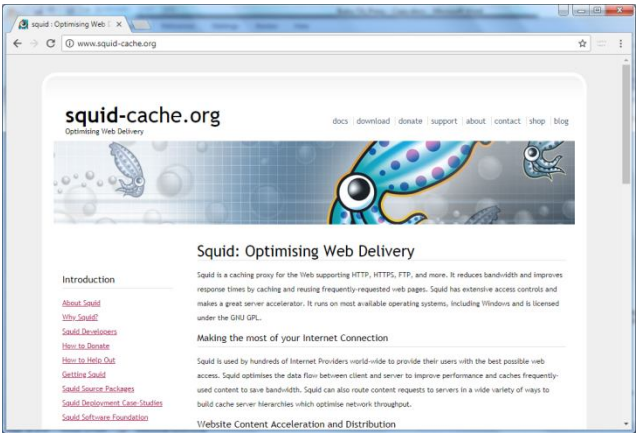
Merupakan halaman utama dari website yang berisi kumpulan cerpen, puisi, dan karya sastralain yang sejenis

2. http://cerpenmu.com/

Gambar 3.6 Tampilan cerpenmu.com

Merupakan halaman utama yang berisi kumpulan cerpen dan tips-tips yang berhubungan

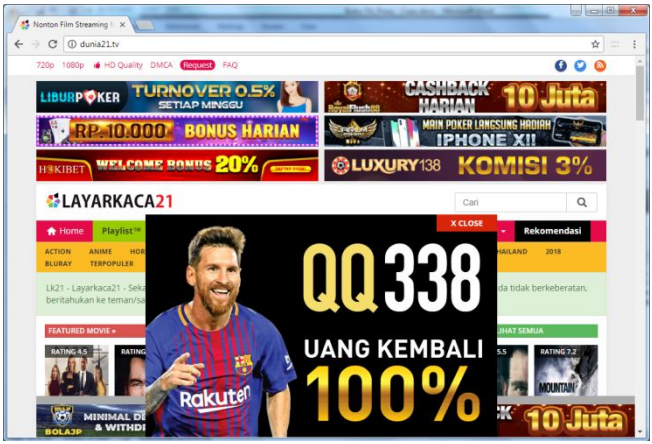
3. http://www.squid-cache.org/



Gambar 3.7 Tampilan squid-cache.org

Merupakan halaman awal dari situs resmi dari squid proxy. Menampilkan definisi singkat squid proxy dan navigasi-navigasi penting berkaitan dengan informasi tentang squid proxy.

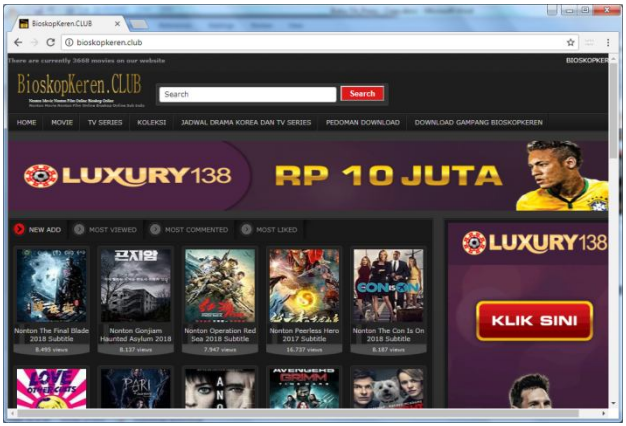
4. http://dunia21.tv/



Gambar 3.8 Tampilan dunia21.tv

Merupakan salah satu situs lokal penyedia layanan streaming film yang populer. Di dalamnya juga terdapat banyak iklan yang menempel

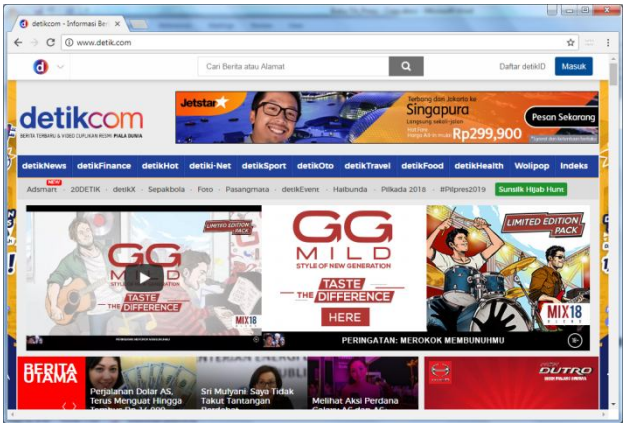
5. http://bioskopkeren.ws/



Gambar 3.9 Tampilan bioskopkeren.ws

Merupakan salah satu situs lokal penyedia layanan streaming film yang populer. Dengan tampilan dominan berwarna gelap. Terdapat beberapa iklan yang menempel

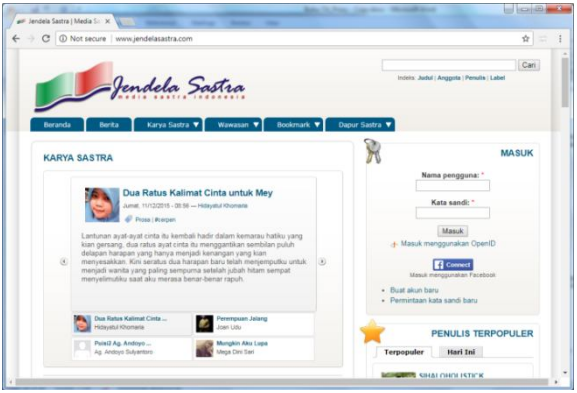
6. http://www.detik.com/



Gambar 3.10 Tampilan detik.com

Merupakan situs populer dengan segala macalm informasi. Biasanya menjadi rujukan pengguna internet dalam mendapatkan informasi dan isu yang sedang hangat.

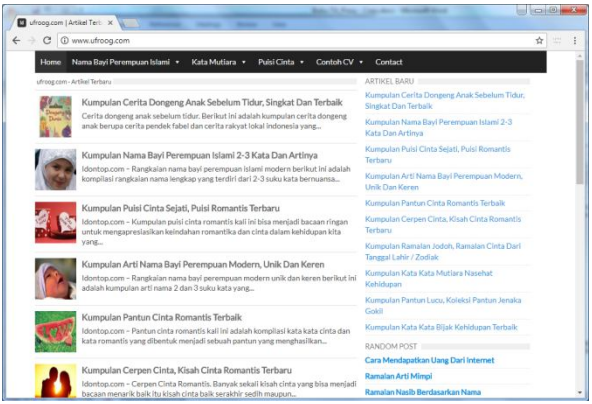
7. http://www.jendelasastra.com/



Gambar 3.11 Tampilan jendelasastra.com

Merupakan situs yang menyediakan konten berupa karya sastra semacam cerpen dan puisi, juga terdapat figur anggota yang mengharuskan user untuk login terlebih dahulu

8. http://www.ufroog.com/



Gambar 3.12 Tampilan ufroog.com

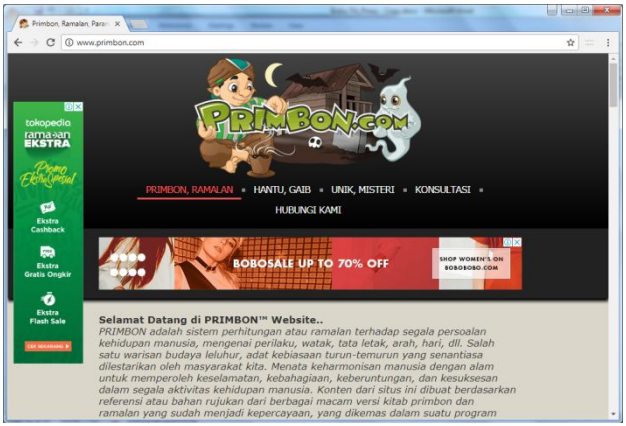
Merupakan halaman situs yang berisi artikel tentang nama-nama bayi dan tips terkait

9. http://www.edipsw.com/

Gambar 3.13 Tampilan edipsw.com

Merupakan situs personal (blog) dari seorang guru SMA yang berisi info-info terkait dan beberapa artikel sesuai dengan passion yang dimiliki pemilik situs.

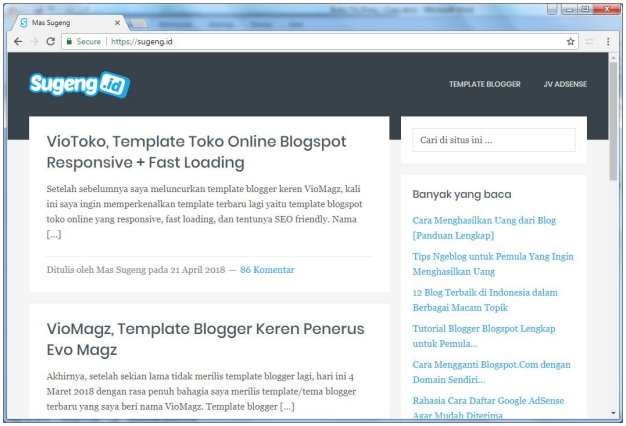
10. http://www.primbon.com/



Gambar 3.14 Tampilan primbon.com

Merupakan situs berisi tentang primbon jawa seperti perhitungan weton, hal-hal gaib berdasarkan kepercayaan masyarakat jawa, dll.

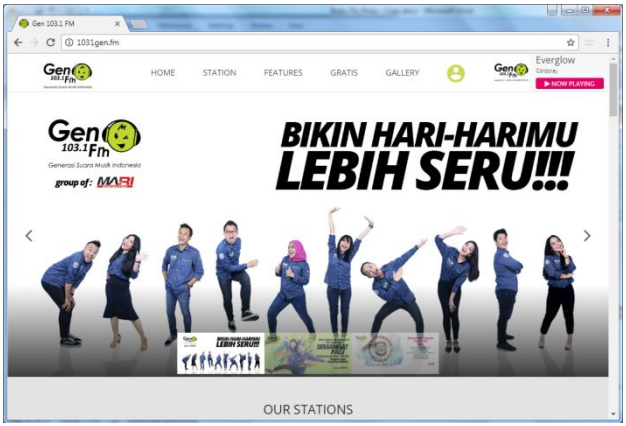
11. http://sugeng.id/



Gambar 3.15 Tampilan sugeng.id

Merupakan website personal yang berisi tips dan trik yang berhubungan dengan dunia IT, teritama terkait internet dan blogging.

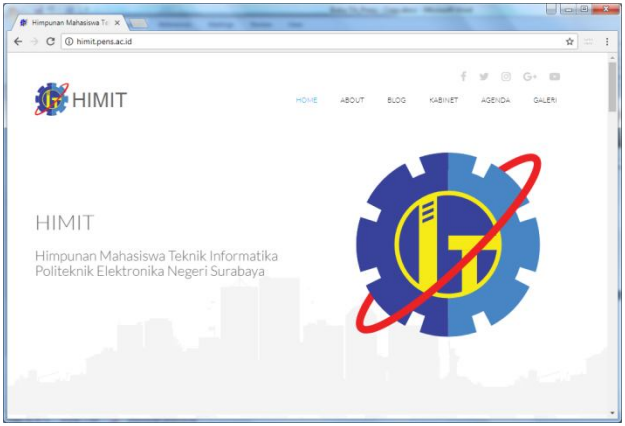
12. http://1031gen.fm/



Gambar 3.16 Tampilan 1031gen.fm

Merupakan halaman utama resmi dari stasiun radio GEN FM Surabaya, berisi informasi terbaru dari GEN FM dan beberapa fitur lain seperti radio streaming.

13. http://himit.pens.ac.id/



Gambar 3.17 Tampilan himit.pens.ac.id

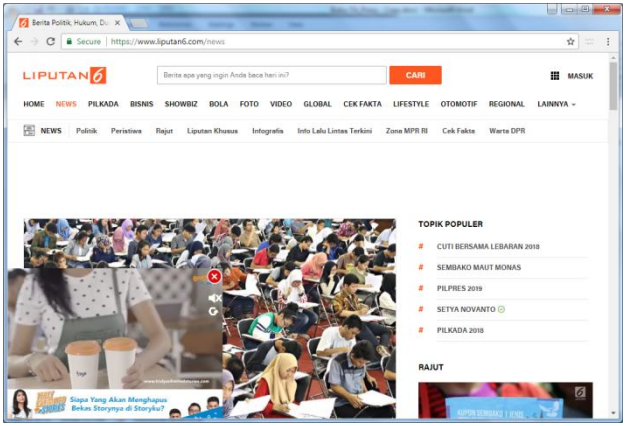
Merupakan halaman utama resmi dari Himpunan Mahasiswa Jurusan IT PENS Surabaya. Berisi informasi-informasi terkait aktifitas kemahasiswaan dan kabar terbaru dari himpunan.

14. http://mangapark.net/manga/naruto/s3/c700

Gambar 3.18 Tampilan mangapark.net/manga/naruto/s3/c700

Merupakan halaman yang berisi manga Naruto cahapter ke 700. Terdapat 21 gambar dari chapter tersebut yang dijadikan satu dalam sebuah halaman.

15. http://news.liputan6.com/



Gambar 3.19 Tampilan news.liputan6.com

Merupakan sub tag dari situs liputan6 yang berisi tentang informasi-informasi berita baik dalam negeri maupun mancanegara.

16. http://mangapark.net/manga/bleach/s3/c686

Gambar 3.20 Tampilan mangapark.net/manga/bleach/s3/c686

Merupakan halaman yang berisi manga Belach cahapter ke 686. Terdapat 18 gambar dari chapter tersebut yang dijadikan satu dalam sebuah halaman

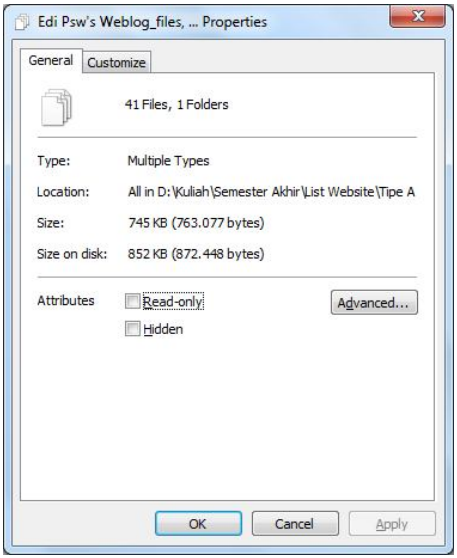
Berdasarkan definsi jenis tersebut maka pembagian jenis website berdasarkan sampel yang sudah ada adalah sebagai berikut

Tabel 3.5 Pembagian jenis website

URL	Page Size	Jenis Website
www.satubahasa.com	954 KB	A
cerpenmu.com	917 KB	A
www.squid-cache.org	41,2 KB	A
dunia21.tv	6,77 MB	B
bioskopkeren.ws	2,09 MB	B
www.detik.com	3,46 MB	B
jendelasastra.com	1,22 MB	A

ufroog.com	468 KB	A
edipsw.com	745 KB	A
www.primbon.com	974 KB	A
sugeng.id	1,46 MB	A
1031gen.fm	14,2 MB	B
himit.pens.ac.id	3,93 MB	B
mangapark.net/ manga/naruto/s3/c700	6,43 MB	B
news.liputan6.com	3,01 MB	B
mangapark.net/manga/bleach/s3/c686	6 MB	B

Cara untuk mengetahui page size suatu website adalah dengan menyimpan utuh seluruh halaman tersebut kemudian melihat propertise keseluruhan halaman yang disimpan.



Gambar 3.21 Melihat page size suatu halaman

Dari sampel dan jenisnya tersebut kemudian disepakati bahwa terdapat tiga jenis kondisi yang muncul dengan menganalisa behavior dari client. Tiga kondisi tersebut adalah

1. Jenis 1, yaitu kondisi trafik dimana website jenis A lebih sedikit diakses daripada jenis B (prosentase 2 : 8)
2. Jenis 2, yaitu kondisi trafik dimana jumlah akses websiter jenis A dan jenis B seimbang (prosentase 5 : 5)
3. Jenis 3, yaitu kondisi trafik dimana website jenis A lebih banyak diakses daripada jenis B (prosentase 8 : 2)

3.1.3 Skenario dan Generate Data

Pada penetapan skenario dan proses generating data, langkah-langkah yang dilakukan adalah sebagai berikut

1. Menentukan skenario
   Pada penelitian ini, terdapat tiga variabel yang digunakan dan berpengaruh pada proses simulasi. Masing-masing variabel tersebut adalah algoritma page replacement sebagai variabel utama, lalu jumlah client dan kondisi trafik sebagai variabel sekunder. Jumlah skenario yang digunakan adalah hasil kali masing-masing total nilai variabel tersebut.
2. Menentukan total trafik
   Salah satu pengaturan environment simulasi adalah dengan mengatur total trafik. Total trafik masing-masing skenario harus sama sehingga nilainya valid ketika output pengukuran dibandingkan.
3. Membuat list halaman url masing-masing kondisi

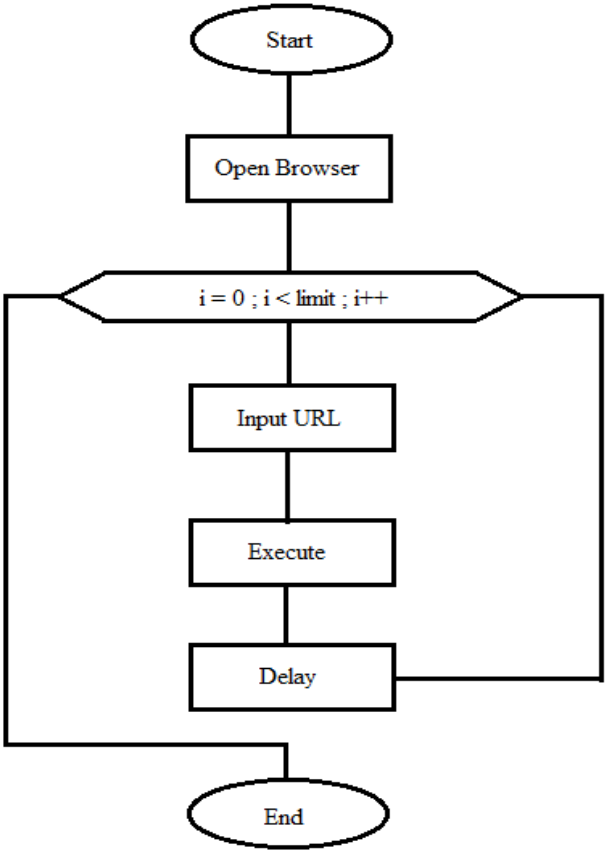
   Berdasarkan variabel kondisi trafik, dari sampel halaman yang ada ditentukan daftar url yang memiliki 10 halaman pada masing-masing. Berikut adalah daftar url untuk kondisi-kondisi trafik yang ada di dalam jaringan.

Tabel 3.6 Daftar halaman url pada masing-masing kondisi

Kondisi 1	Kondisi 2	Kondisi 3
satubahasa.com	dunia21.tv	dunia21.tv
cerpenmu.com	satubahasa.com	bioskopkeren.ws

squid-cache.org	detik.com	detik.com
ufroog.com	cerpenmu.com	1031gen.fm
edipsw.com	1031gen.fm	himit.pens.ac.id
primbon.com	ufrooog.com	mangapark.net/ manga/naruto/s3/c700
sugeng.id	himit.pens.ac.id	news.liputan6.com
dunia21.tv	edipsw.com	mangapark.net/manga /bleach/s3/c686
news.liputan6.com	news.liputan6.com	ufroog.com
jendelasastra.com	primbon.com	primbon.com

4. Menentukan distribusi trafik
   Dengan melihat variabel list halaman url dan total trafik, ditemukan iterasi yang akan dipakai dalam proses generate data. Lalu untuk menentukan distribusi trafiknya, nilai-nilai pada variabel kondisi trafik digunakan. Nantinya jumlah akses website A dan B pad a masing-masing kondisi akan berbeda namun jumlah akhir tetap sama.
5. Generate trafik berdasarkan distribusi halaman url
   Pada proses generating data, dilakukan automasi menggunakan tool bantuan autoIt. Generating data berupa urutan perintah kerja yang automatis berjalan. Dengan menggunakan bantuan tool AutoIt, akan dibuat suatu program yang menjalankan proses automasi mulai dari membuka browser hingga proses membuka halaman url. Berikut adalah flowchart dari program tersebut



Gambar 3.22 Flowchart generate data

3.1.4 Monitoring dan Collecting data

Pada setiap sebuah skenario simulasi berjalan, proses monitoring dilakukan baik itu menggunakan aplikasi, dalam hal ini aplikasi yang digunakan adalah Cacti dan aplikasi buatan sendiri, maupun manual monitoring dengan melihat file-file log squid.

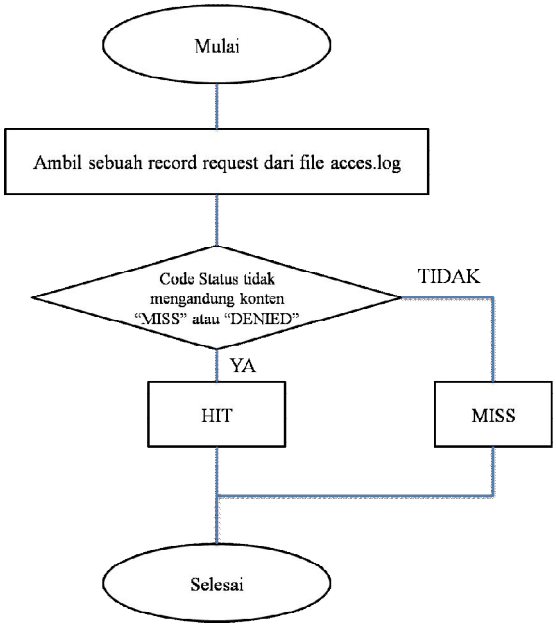
Setelah satu skenario simulasi selesai, proses collecting data dilakukan. Data-data yang dikumpulkan berupa file log dan file csv.

3.1.5 Output Parameter

Terdapat lima jenis output parameter yang digunakan, yaitu Hit Rate, Byte Hit Rate, I/O parameter, Utilization parameter, dan Cacti NLANR parameter.

1. Hit Rate

   Hit rate merupakan perbandingan antara request hit dengan total keseluruhan request Hit rate berkaitan erat dengan code status yang ada di dalam file access.log. dari semua jenis code status yang munking muncul pada proxy squid, nantinya akan dibedakan menjadi dua jenis, yaitu code status berjenis HIT dan MISS. Berikut adalah algoritma flowchart pembagian jenis HIT dan MISS pada code status di dalam file access.log

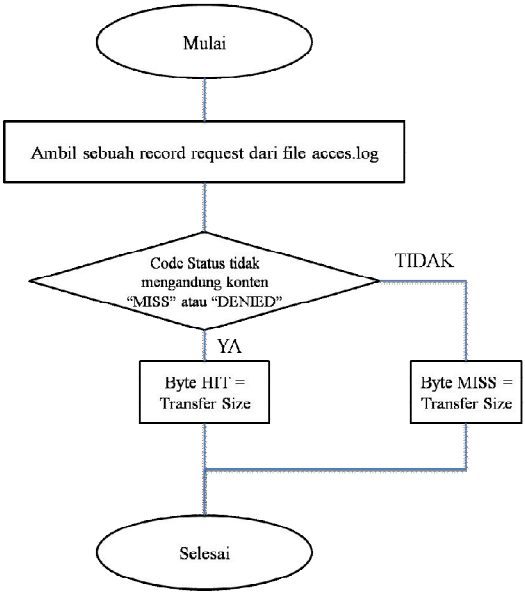


Gambar 3.23 Flowchart penentuan HIT dan MISS

2. Byte Hit Rate

   Merupakan perbandingan antara jumlah byte HIT dengan jumlah byte request total. Jika pada Hit rate yang diambil adalah jenis

HIT atau MISSnya maka pada Byte Hit ratio yang diambil adalah nilai transfer size.



Gambar 3.24 Flowchart penentuan Byte HIT dan Byte MISS

3. I/O parameter

   I/O parameter berhubungan dengan skema penyimpanan pada squid proxy, terutama untuk skema `aufs`. Beberapa parameter I/O antara lain open, close, cancel, write, read, stat, unlink, check_callback. Dari paremeter-parameter tersebut, bisa diketaui salah satunya adalah read rata-rata, yaitu perbandingan antara nilai read dan check_callback

4. Utilization parameter

   Merupakan parameter-parameter penggunaan sumber daya baik itu sumber daya traffik mapun sumber daya masing-masing komputer itu sendiri, baik server maupun client. Parameter utilisasi yang diambil adalah nilai CPU time dan swap I/O

5. Parameter NLANR Cacti

   Menggunakan parameter output yang ada di dalam project NLANR Cacti, yaitu File descriptors.

3.2 Implementasi

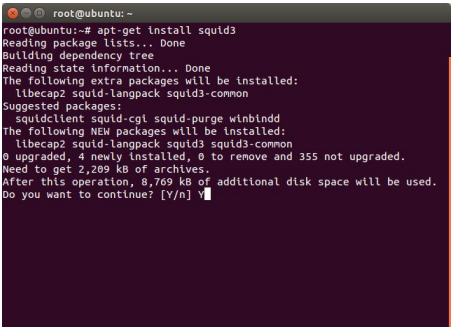
Setelah semua pemodelan berhasil dibuat maka tahap selanjutnya adalah implementasi. Beberapa poin penting terkait proses implementasi adalah

1. Detail umum server dan client serta kondisi jaringan
   Sistem operasi yang digunakan oleh server adalah Ubuntu versi 14.04 sedangkan pada client, semua Sistem Operasi yang digunakan adalah Windows 7. Terdapat dua macam komputer client pada simulasi kali ini, yaitu yang berjalan pada platform 32 bit dan 64 bit. Pastikan juga node pada jaringan sudah terhubung pada jaringan yang sama. Untuk proses checking bisa menggunakan command ping pada terminal maupun command prompt
2. Aplikasi pendukung telah terinstall
   Tahapan pertama sebelum memulai simulasi trafik adalah memastikan semua aplikasi-aplikasi yang diperlukan, sudah terinstall, baik di sisi server seperti squid, program monitoring, dan cronjob maupun di sisi client seperti browser, tool Automasi, hingga task scheduler
3. Konfigurasi aplikasi di sisi server sesuai pemodelan
   Konfigurasi disini baik konfigurasi pada aplikasi proxy server maupun aplikasi-aplikasi yang digunakan pada saat proses simulasi trafik.
4. Konfigurasi aplikasi di sisi client sesuai pemodelan
   Di sisi client, konfigurasi dilakukan di dalam browser, yaitu setting ip proxy server, dan proses automasi dengan cara menjadwalkan proses automasi
5. Running simulasi trafik
   Saat proses running simulasi, beberapa hal yang perlu diperhatikan antara lain urutan dan alur simulasi harus sesuai dengan yang ada di dalam pemodelan, kemudian proses scheduling di sisi server dan client
6. Hasil monitoring dan collecting data
   Aplikasi monitoring menghasilka parameter-parameter output berdasarkan simulasi yang sudah dijalankan. Beberapa parameter yang terkait lalu digunakan dan dianalisa sehingga dihasilkan kesimpulan

3.2.1 Instalasi Proxy server

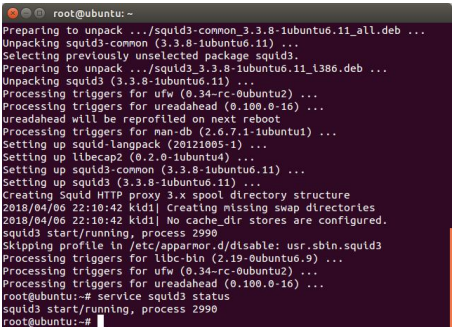
Proxy server yang digunakan pada penelitian kali ini adalah squid proxy. Squid proxy merupakan salah satu alikasi proxy server paling populer. Pada simulasi kali ini, squid proxy dipasang pada server jaringan yang berbasis linux sehingga instalasi squid proxy dilakukan melalui terminal. Ada pun langkah-langkah instalasinya adalah sebagai berikut:

1. Buka terminal, kemudian install squid versi terbaru, dalam hal ini squid 3, menggunakan bantuan command apt, format instalasinya adalah `apt-get install squid3`. Akan ada notfikasi lanjutan untuk proses instalasi.



Gambar 3.25 Instalasi awal squid3

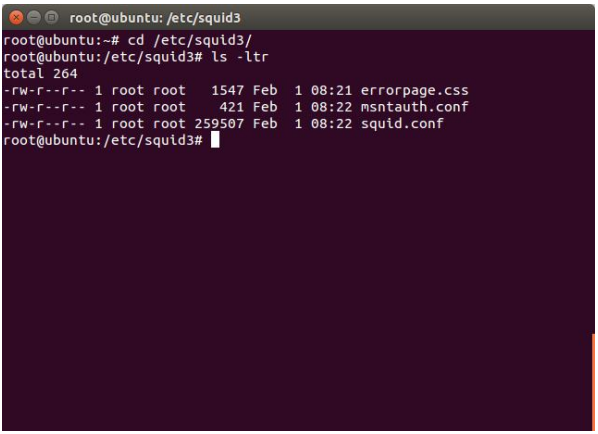
2. Tunggu proses instalasi berjalan



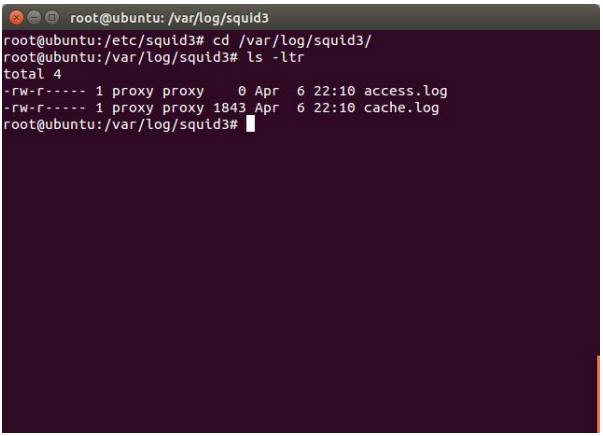
Gambar 3.26 Proses instalasi squid3 sedang berjalan

3. Squid3 sudah berhasil terinstall. Di dalam squid3 terdapat direktori penting yang biasanya diakses dan digunakan.

Direktori`/etc/squid3` berisi file-file default yang digunaka untuk konfigurasi squid proxy. Sedangkan direktori `/var/log/squid3` adalah direktori default tempat penyimpanan file-file yang mencapat log transaksi pada squid proxy.

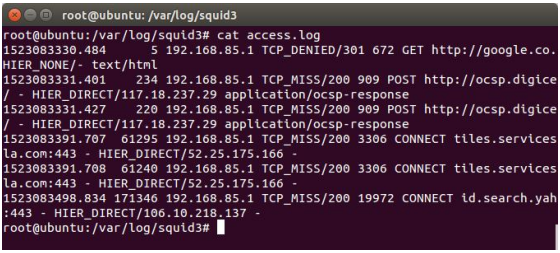


Gambar 3.27 Direktori /etc/squid3



Gambar 3.28 Direktori /var/log/squid3

Pada direktori /var/log/squid3 terdapat file access.log yang merupakan tempat penyimpanan log record dari proses request halaman url yang terjadi pada jaringan tersebut.

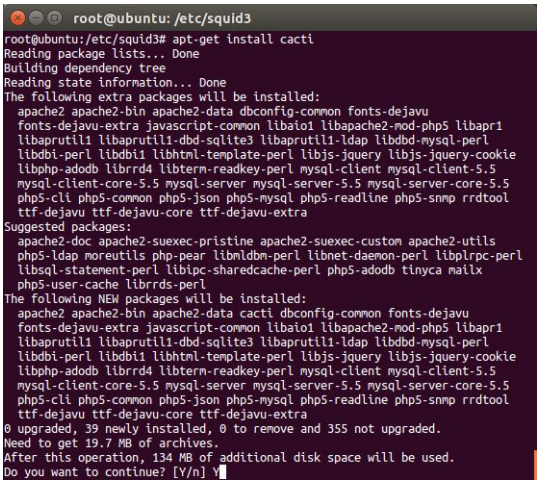


Gambar 3.29 Isi access.log

3.2.2 Instalasi Aplikasi Monitoring Proxy Populer

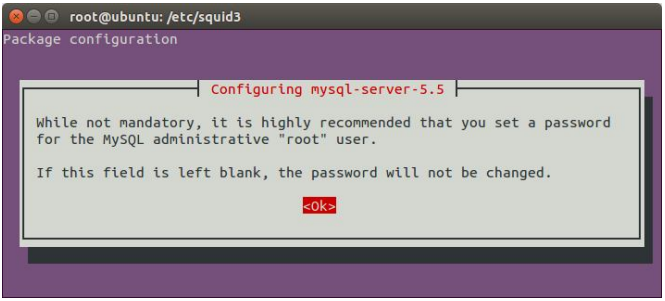
Isntalasi aplikasi monitoring populer, yaitu cacti, dilakukan via terminal, sedangkan integrasi cacti dengan squid proxy memerlukan konfigurasi lanjutan. Berikut langkah-langkah instalasinya

1. Pada instalasi cacti menggunakan fitur apt pada terminal, terdapat banyak dependencies yang harus terinstall jika ingin menjalankan aplikasi cacti. Biasanya fitur apt menampilkan list dependencies tersebut.



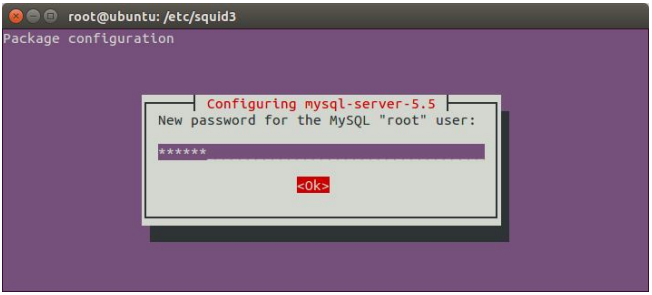
Gambar 3.30 Instalasi awal cacti dan dependenciesnya

2. Jika proses instalasi berlanjut maka akan muncul opsi instalasi dari dependencies yang digunakan oleh cacti. Yang pertama adalah mysql server



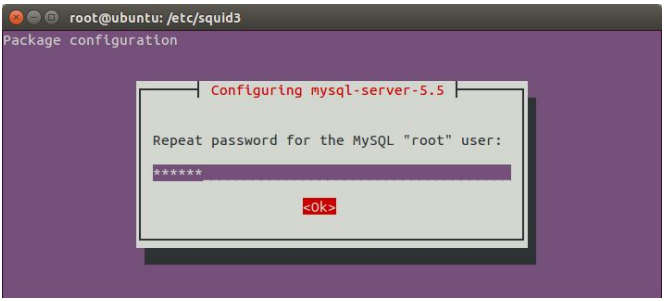
Gambar 3.31 Konfigurasi mysql server pada instalasi awal cacti

Konfigurasi mysql server tersebut termasuk password yang digunakan untuk mengakses mysql server



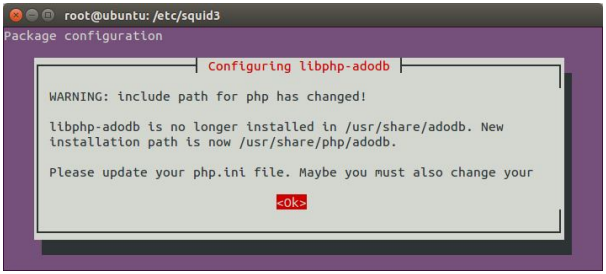
Gambar 3.32 Membuat password pada mysql server

Input password dilanjutkan dengan konfirmasi password untuk memastikan apakah data password yang dimasukan benar



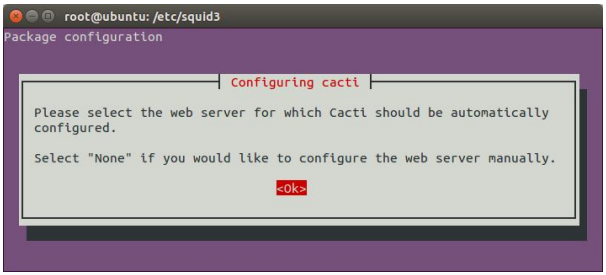
Gambar 3.33 Konfirmasi password pada mysql server

3. Konfigurasi dependecies selanjutnya yaitu konfigurasi php



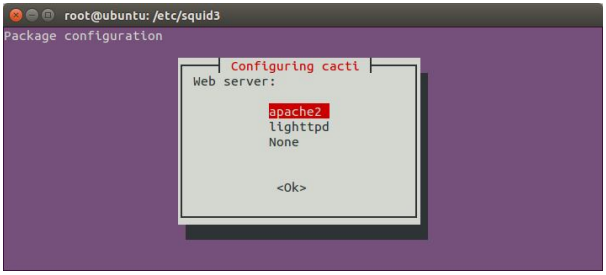
Gambar 3.34 Konfigurasi php pada instalasi cacti

4. Setelah itu masuk muncul konfigurasi awal untuk web server, dalam hal ini web server yang dipilih adalah apache web server



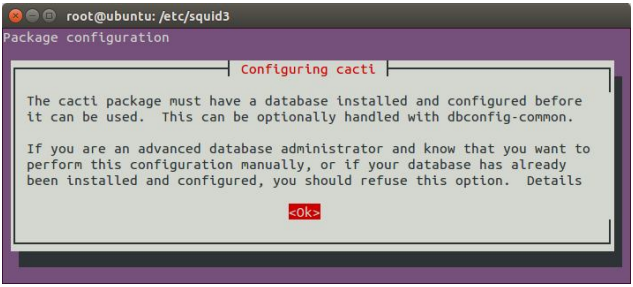
Gambar 3.35 Konfigurasi awal web server pada cacti

Terdapat beberapa opsi web server yang bisa digunakan, pada penelitian kali ini, web server yang digunakan adalah apache



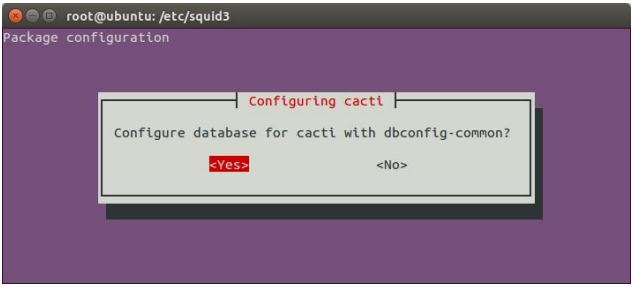
Gambar 3.36 Opsi web server pada instalasi awal cacti

5. Langkah selanjutnya adalah konfigurasi admin database untuk menyimpan data cacti. Terdapat beberapa kotak dialog dengan opsi masing-masing



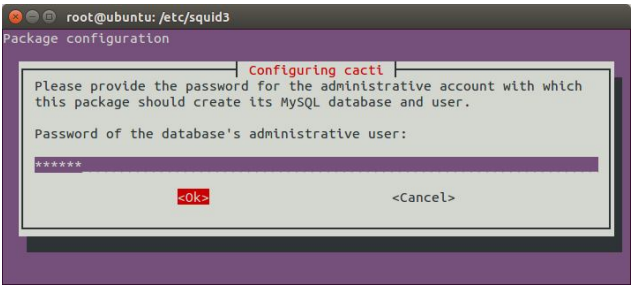
Gambar 3.37 Kotak dialog awal setting admin db pada cacti

Selanjutnya yang muncul adalah pilih konfigurasi menggunakan dbconfig-common. Pilih opsi yes untuk menggunakan dbconfig-common sebagai pilihan

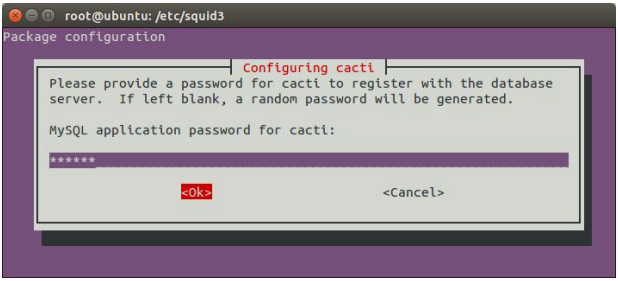


Gambar 3.38 Konfirmasi konfigurasi database untuk cacti

Setelah memilih dbconfig-common, langkah pertama adalah setting password untuk db aplikasi monitoring cacti

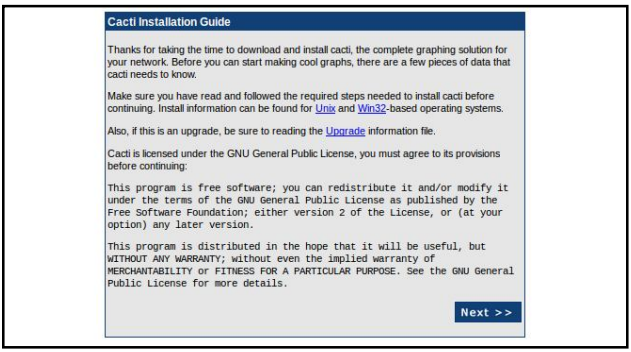


Gambar 3.39 Mendefinisikan password untuk admin db cacti



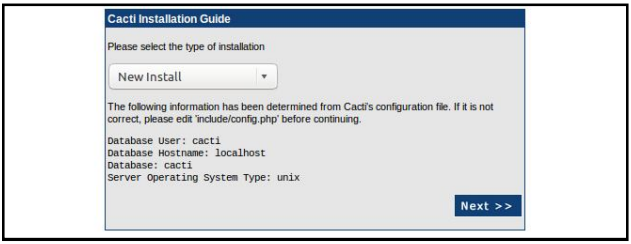
Gambar 3.40 Konfirmasi password untuk admin db cacti

6. Setelah itu instalasi selesai, selanjutnya adalah konfigurasi awal dengan membuka cacti melalui browser. Alamat default untuk cacti adalah `[IP server]/cacti`. JIka kondisi cacti adalah sesaat setelah selesai instalasi maka muncul cacti installation guide



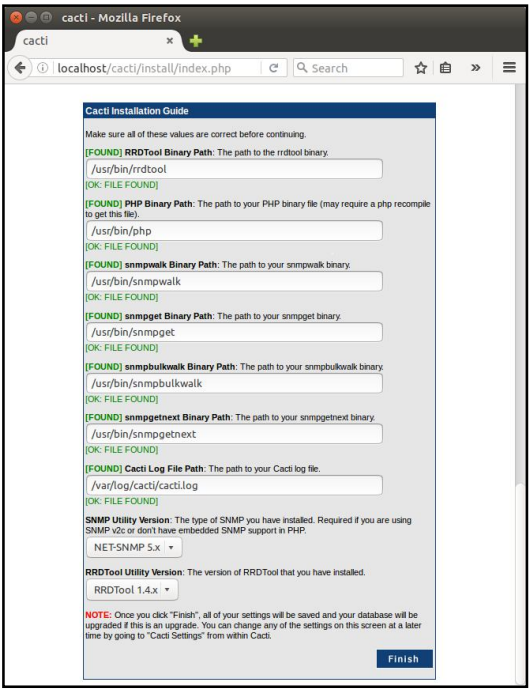
Gambar 3.41 Cacti installation guide

7. Terdapat opsi tipe instalasi



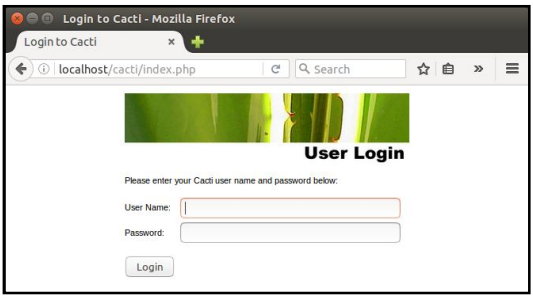
Gambar 3.42 Opsi tipe instalasi cacti

8. Lalu selanjutnya adalah review setting awal dari dependencies yang sebelumnya sudah dikonfigurasi



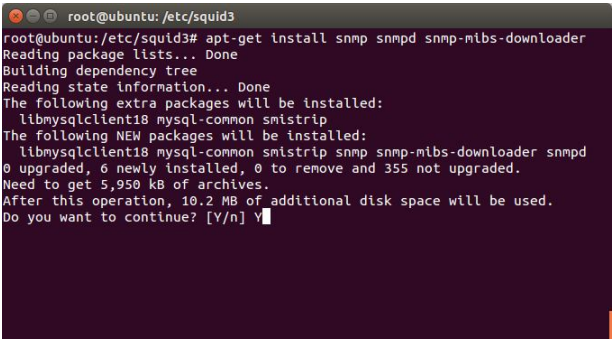
Gambar 3.43 Review konfigurasi dependencies cacti

9. Maka secara default, cacti siap digunakan



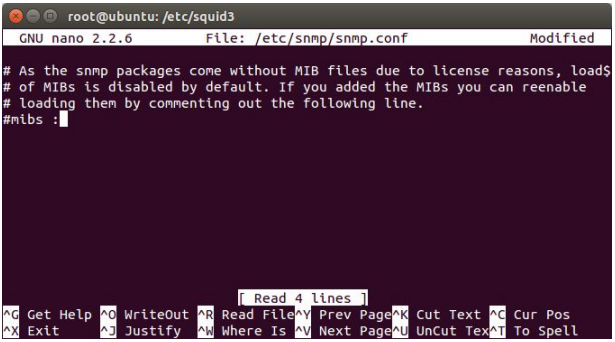
Gambar 3.44 Halaman login cacti

10. Cacti yang sudah terinstall belum terintegrasi dengan proxy server. Untuk proses integrasi dperlukan snmp dan beberapa dependecies



Gambar 3.45 Instalasi snmp, snmpd, dan mibs downloader

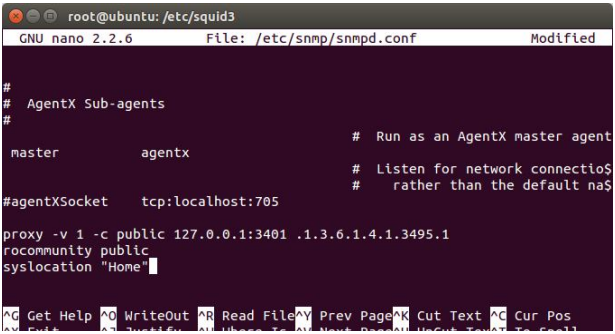
11. Selanjutnya adalah konfigurasi pada snmp. Pada file snmp.conf, hapus parameter mibs dengan cara menambahkan tanda pagar di depan parameter tersebut



Gambar 3.46 Konfigurasi snmp.conf

Sementara pada file snmpd.conf, tambahkan baris parameter sebagai berikut

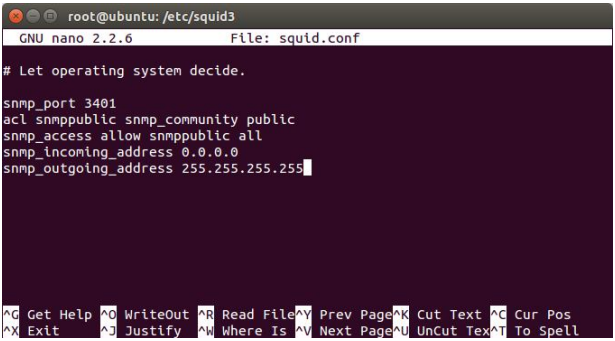
```
proxy -v 1 -c public 127.0.0.1:3401
.1.3.6.1.4.1.3595.1
rocommunity public
syslocation “Home”
```



Gambar 3.47 Konfigurasi file snmpd.conf

12. Agar bisa terintegrasi, maka pada squid proxy, parameter snmp dan parameter lain yg terkait harus diaktifkan dan diberi nilai. Berikut parameter tambahan tersebut

```
snmp_port 3401
acl snmppublic snmp_community public
snmp_access allow snmppublic all
snmp_incoming_address 0.0.0.0
snmp_outgoing_address 255.255.255.255
```

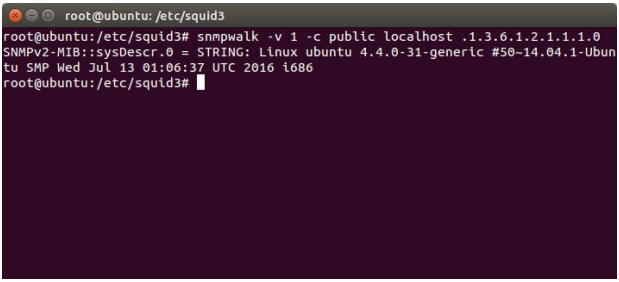


Gambar 3.48 Konfigurasi squid.conf untuk integrasi snmp

13. Setelah selesai langkah selanjutnya adalah testing apakah snmp dan squid proxy sudah berjalan dengan baik. Untuk proses testing dapat menggunakan baris perintah sebagai berikut

```
snmpwalk-v 1 -c public 127.0.0.1:3401
.1.3.6.1.4.1.3595.1
```

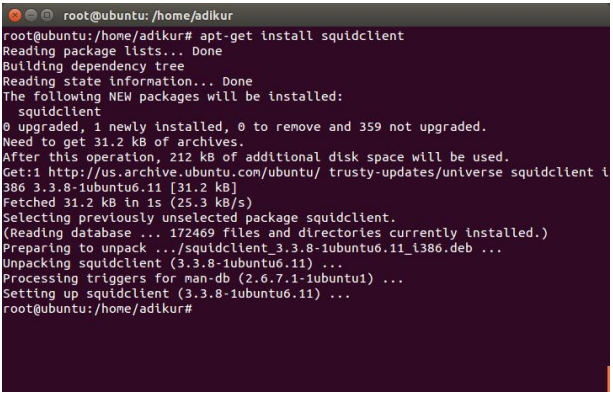
hasil dari baris perintah tersebut akan menghasilkan output seperti ini



Gambar 3.49 Output testing integrasi snmp dan squid proxy

3.2.3 Instalasi Tool Tambahan Pada Proxy

Tool tambahan disini adalah squidclient. Untuk menginstall squidclient, sebelumnya harus menginstall squid terlebih dahulu. Proses instalasi squidclientmenggnakan bantuan apt, yaitu `apt-get install squidclient`



Gambar 3.50 Instalasi squidclient

format umum penggunaan squidclient adalah

```
squidclient -h [IP host] -p [port] cache_object
:[value] mgr:[value]
```

dimana semua parameternya tidak wajib. Misal baris perintah squidclient mgr:info akan menampilkan informasi umum keseluruhan dari sebuah jaringan proxy.

Pada penelitian kali ini terdapat 2 nilai yang digunakan pada parameter mgr, yaitu

`squidclient mgr:squidaio_counts` dan
`squidclient cache_object://localhost/counters`

Berikut adalah contoh hasil output squidclient dengan menggunakan parameter-parameter tersebut

Tabel 3.7 Contoh Output squidclient mgr:squidaio_counts

```
ASYNC IO Counters:
Operation     # Requests    Number serviced
open   1080   1080
close  0      0
cancel 1077   -
write  0      0
read   103595 103595
stat   0      0
unlink 0      0
check_callback      1595998      -
queue  0      -

Threads Status:
#      ID     # Requests
1      0xb5201b40    6497
2      0xb5242b40    6615
3      0xb5283b40    6529
4      0xb52c4b40    6538
5      0xb5305b40    6432
6      0xb5346b40    6637
7      0xb5387b40    6494
8      0xb53c8b40    6613
9      0xb5409b40    6567
10     0xb544ab40    6454
11     0xb548bb40    6529
12     0xb54ccb40    6620
13     0xb550db40    6480
14     0xb554eb40    6568
15     0xb558fb40    6500
16     0xb55d0b40    6602
```

Sementara itu output squidclient cache_object://localhost/counters adalah sebagai berikut

Tabel 3.8 Contoh output squidclient cache_object://localhost/counters

```
sample_time = 1526351672.343379 (Tue, 15 May 2018
02:34:32 GMT)
server.all.requests = 3425
server.all.errors = 0
server.all.kbytes_in = 226183
server.all.kbytes_out = 5486
server.http.requests = 2463
server.http.errors = 0
server.http.kbytes_in = 35560
server.http.kbytes_out = 1548
server.ftp.requests = 0
server.ftp.errors = 0
server.ftp.kbytes_in = 0
server.ftp.kbytes_out = 0
server.other.requests = 962
server.other.errors = 0
server.other.kbytes_in = 190623
server.other.kbytes_out = 3938
icp.pkts_sent = 0
icp.pkts_recv = 0
icp.queries_sent = 0
icp.replies_sent = 0
icp.queries_recv = 0
icp.replies_recv = 0
icp.query_timeouts = 0
icp.replies_queued = 0
icp.kbytes_sent = 0
icp.kbytes_recv = 0
icp.q_kbytes_sent = 0
icp.r_kbytes_sent = 0
icp.q_kbytes_recv = 0
icp.r_kbytes_recv = 0
icp.times_used = 0
cd.times_used = 0
cd.msgs_sent = 0
```

```
cd.msgs_recv = 0
cd.memory = 0
cd.local_memory = 1
cd.kbytes_sent = 0
cd.kbytes_recv = 0
unlink.requests = 0
page_faults = 0
select_loops = 1614127
cpu_time = 72.192000
wall_time = 22.880578
swap.outs = 756
swap.ins = 714
swap.files_cleaned = 0
aborted_requests = 457
```

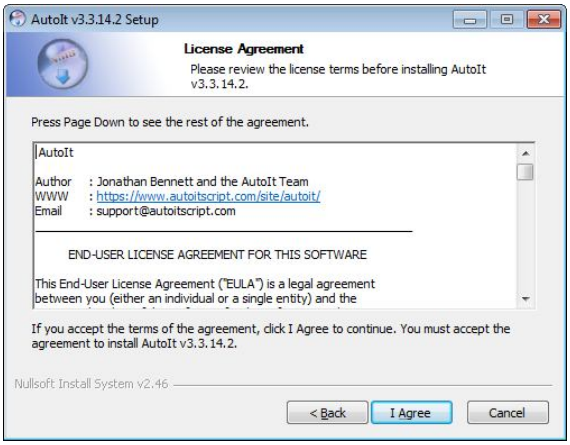
3.2.4 Instalasi Aplikasi Automasi Trafik Pada Client

Aplikasi automasi yan gdigunakan adalah AutoIt. AutoIt merupakan salah satu software yang digunakan untuk proses automasi pada sistem operasi berbasis windows. Berikut adalah langkah-langkah instalasinya

1. Setelah proses download dan membuka file setup AutoIt, muncul kotak dialog awal

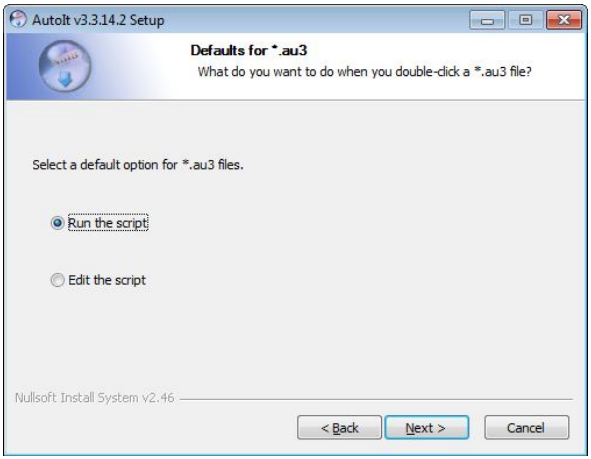
Gambar 3.51 Dialog box awal AutoIt

2. Selanjutnya adalah License Agreement



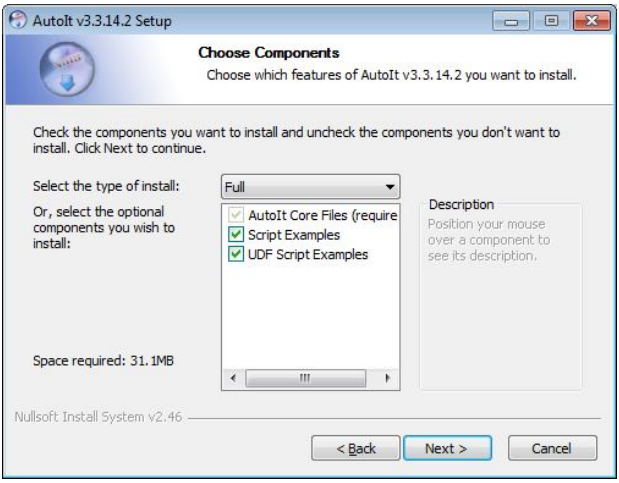
Gambar 3.52 Dialog license agreement AutoIt

3. Kotak dialog yang muncul selanjutnya adalah opsi untuk memilih default proses membuka untuk file standar AutoIt yaitu file berekstensi .au3. Terdapat 2 pilihan, membuka file atau mengeksekusi file



Gambar 3.53 Dialog select defaut option AutoIt

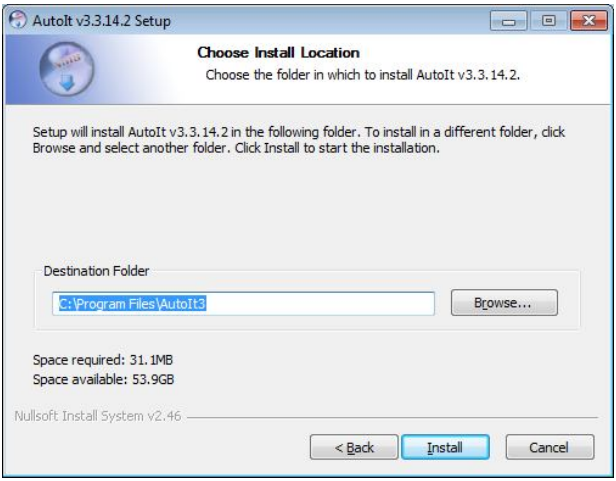
4. Selanjutnya adalah memilih komponen program yang akan diinstall



Gambar 3.54 Dialog pilih komponen AutoIt

Terdapat pilihan komponen apa saja yang bisa dipilih untuk dipasang dengan aplikasi AutoIt, yaitu script examples dan UDF scrpit examples

5. Lalu memilih direktori program



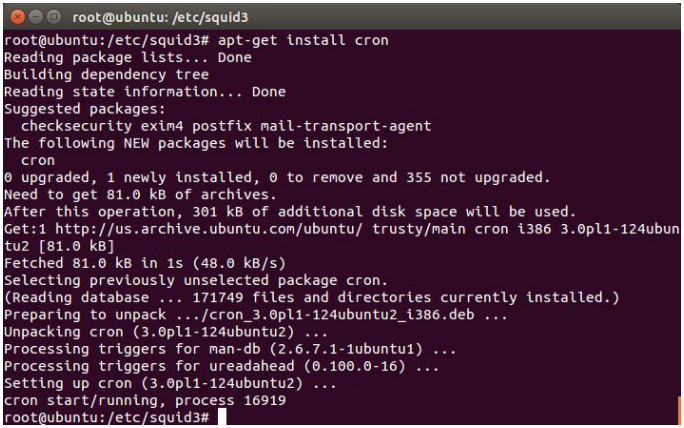
Gambar 3.55 Dialog pilih direktori AutoIt

6. Selesai

Gambar 3.56 Dialog akhir instalasi AutoIt

3.2.5 Instalasi Cron Scheduler pada Server

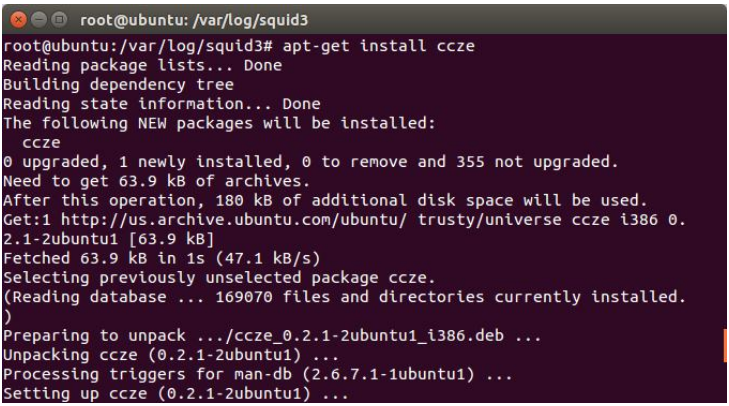
Pada sistem operasi berbasis linux terdapat fitur scheduling yang bernama cronjob. Untuk mengaktifkan fitur cronjob pada sistem operasi berbasis linux, salah satu aplikasi yang bisa digunakan adalah cron. Instalasi cron pada linux ubuntu dilakukan via terminal menggunakan comand `apt-get install cron`



Gambar 3.57 Instalasi cron

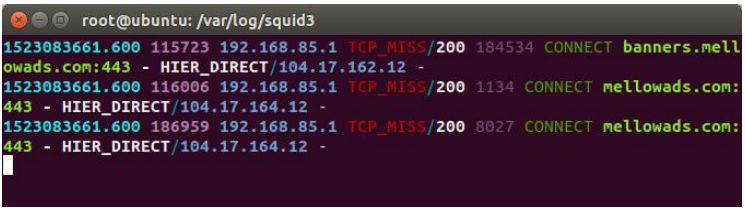
3.2.6 Instalasi Tool Live Monitoring Tambahan Pada Server

ccze adalah salah satu tool optional yang digunakan pada proses live monitoring log via terminal. ccze membuat log squid menjadi bervariasi warnanya sehingga mudah dalam proses checking manual. Proses instalasi pada linux ubuntu menggunakan command `apt-get install ccze`.



Gambar 3.58 Instalasi ccze

Untuk proses live monitoring log squid via terminal, pertama-tama pindahkan direktori aktif pada terminal ke dalam direktori log squid standar atau yang sudah dikonfigurasi. Pada simulasi kali ini, direktori log menggunakan lokasi standar pada squid yaitu `/var/log/squid3`. Pada direktori tersebut terdapat file bernama access.log. Untuk melakukan loive checking log gunakan perintah `tail -f access.log | ccze`



Gambar 3.59: Live checking log squid via terminal

3.2.7 Konfigurasi Proxy Server

Konfigurasi squid proxy server pada simulasi ini terbagi dua, yaitu konfigurasi file squid.conf dan konfigurasi keseluruhan squid hingga berjalan

Berdasarkan pemodelan yang sudah ada maka script di dalam file squid.conf adalah sebagai berikut

Tabel 3.9 Isi file squid.conf

```
#Mendefinisikan ACL yang digunakan
acl snmppublic snmp_community public
acl localnet src 192.168.0.0/16
acl lokalhost src 127.0.0.1/32
acl SSL_ports port 443
acl Safe_ports port 80
acl Safe_ports port 21
acl Safe_ports port 443
acl Safe_ports port 70
acl Safe_ports port 210
acl Safe_ports port 1025-65535
acl Safe_ports port 280
acl Safe_ports port 488
acl Safe_ports port 591
acl Safe_ports port 777
acl CONNECT method CONNECT

#Mengatur akses ACL yang sudah dibuat
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost manager
http_access allow lokalhost manager
http_access deny manager
http_access allow localnet
http_access allow localhost
http_access deny all
snmp_access allow snmppublic all

#Mendefinisikan port yang digunakan proxy
http_port 3128
snmp_port 3401
```

```
#Setting cache dan swap
cache_mem 64 MB
cache_swap_low 70
cache_swap_high 75

#Implementasi variabel utama pada konfigurasi squid
memory_replacement_policy lru
cache_replacement_policy lru
cache_dir aufs /cache_lru_case01/squid 100 16 256

#Medefinisikan coredump direktori
coredump_dir /var/spool/squid3

#Tuning default yang disediakan squid proxy
refresh_pattern ^ftp:            1440   20%    10080
refresh_pattern ^gopher:  1440   0%     1440
refresh_pattern -i (/cgi-bin/|\?) 0     0%     0
refresh_pattern (Release|Packages(.gz)*)$      0
20%     2880
refresh_pattern .          0      20%    4320

#Setting SNMP
snmp_incoming_address 0.0.0.0
snmp_outgoing_address 255.255.255.255
```

Berikut adalah beberapa penjelasan dari script file squid.comf

1. ACL

   Terdapat empat jenis ACL yang terdapat di dalam file konfigurasi, yaitu snmp, IP source address, port dan method.

2. Aksesibilitas

   Aksesibilitas di dalam file konfigurasi termasuk dalam akses terhadap port http dan snmp.

3. Port

   Mendefinisikan port yang digunakan paa konfigurasi proxy, dalam hal ini pot yang digunakan adalah port http dan snmp

4. Cache dan swap

   Setting-setting yang berhubungan dengan caching dan swap. Termasuk di dalamnya adalah parameter cache_mem,

cache_swap, serta implementasi variabel utama di dalam parameter memory_replacement_policy, cache_replacement_policy, dan cache_dir

5. Core Dump

   Mendefinisikan direktri coredump squid proxy

6. Tuning

   Tuning standar yang disediakan oleh squid proxy adalah setting pada refresh_pattern

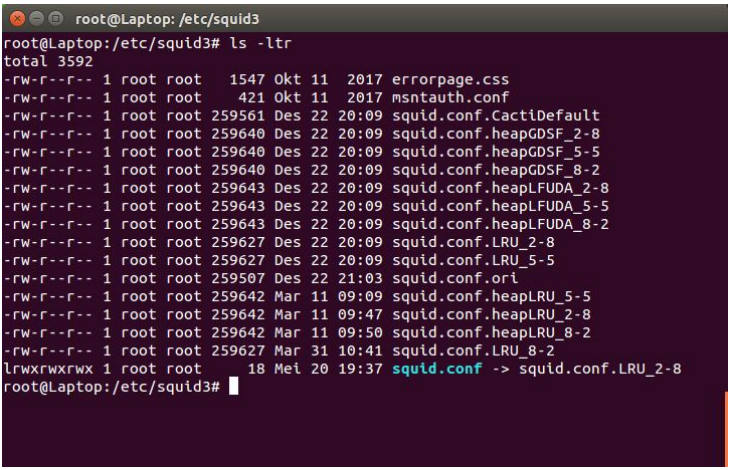
7. SNMP

   Setting snmp seperti outgoing address dan incoming address

karena terdapat 4 page replacement dengan 3 kondisi traffik maka akan terdapat 12 variasi konfigurasi squid.conf. buat 12 file konfigurasi tersebut dalam folder yang sama dengan file squid.conf. Agar nantinya file squid.conf yang terbaca adalah file konfigurasi squid yang diinginkan, gunakan perintah link untuk mengarahkan file squid.conf agar membaca file konfigurasi squid yang sesuai. Baris perintah tersebut adalah

```
ln –sf [nama_file_squid_yang_dimaksud] squid.conf
```

jika perintah tersebut berhasil maka akan menghasilkan contoh tampilan seperti ini



Gambar 3.60 Tampilan direktori tempat squid.conf berada

Selanjutnya sebelum menjalankan atau merestart ulang squid agar konfigurasi dapat terbaca ulang. Pastikan swap direktori sudah dibuat. Command untuk membuat swap direktori adalah

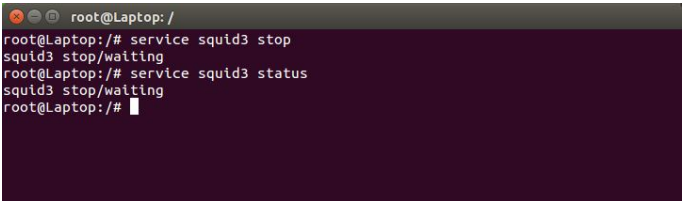
```
squid3 –z
```

namun sebelum menjalankan baris perintah stb, pastikan file direktori tersebut ada dan squid dalam keadaan mati. Untuk melihat status squid adalah

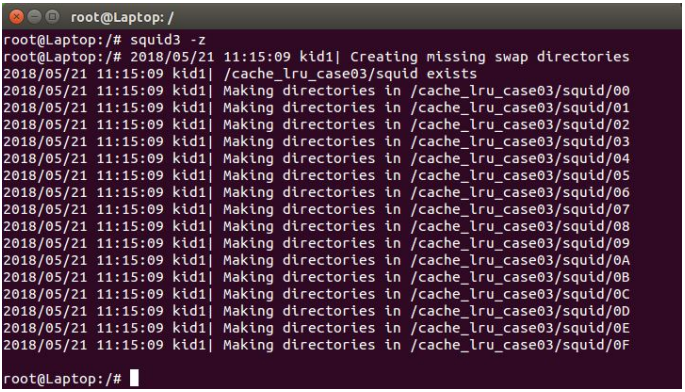
```
service squid3 status
```

sedangkan untuk mematikan atau menjalankan program squid adalah sebagai berikut

```
service squid [stop]/[start]
```



Gambar 3.61 Status service squid



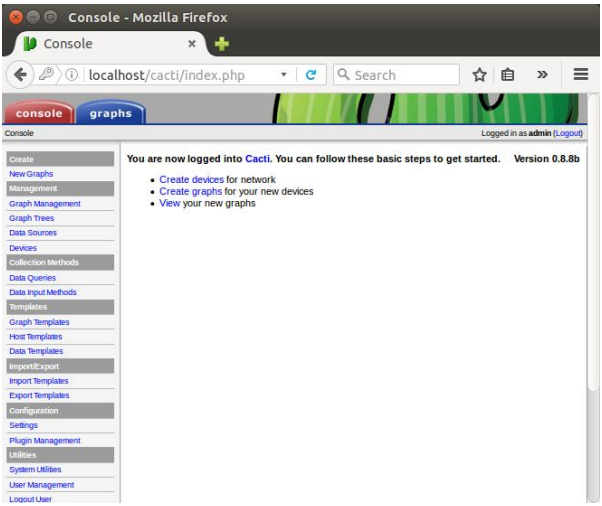
Gambar 3.62 Membuat direktori swap squid

3.2.8 Konfigurasi NLANR Monitoring Proxy

Project NLANR adalah salah satu project yang mengembangkan template monitoring squid berbasis cacti. Hasil dari

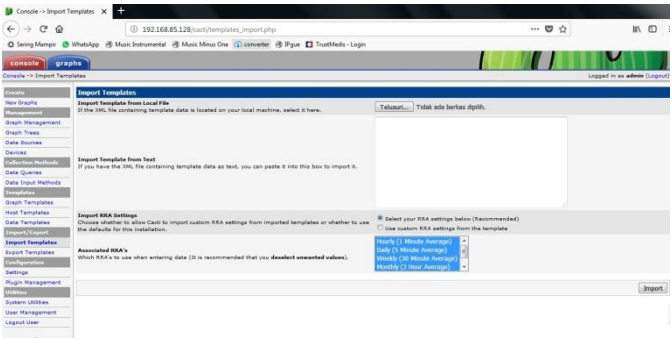
project ini adalah sebuah file XML template cacti. Langkah-langkah proses konfigurasi NLANR pada cacti adalah sebagai berikut

1. Buka Cacti monitoring pada browser, pilih Import/Export – Import Templates



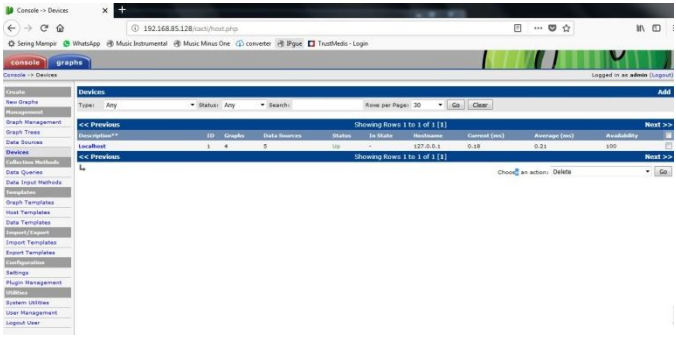
Gambar 3.63 Tampilan default cacti monitoring

2. Muncul form Import Templates, pilih file NLANR dan opsi-opsi lain. Misal pada opsi Associated RRA, pilih semua opsi sehigga pada proses monitoring, range waktu bisa disetting dan dipilih sesuai keinginan



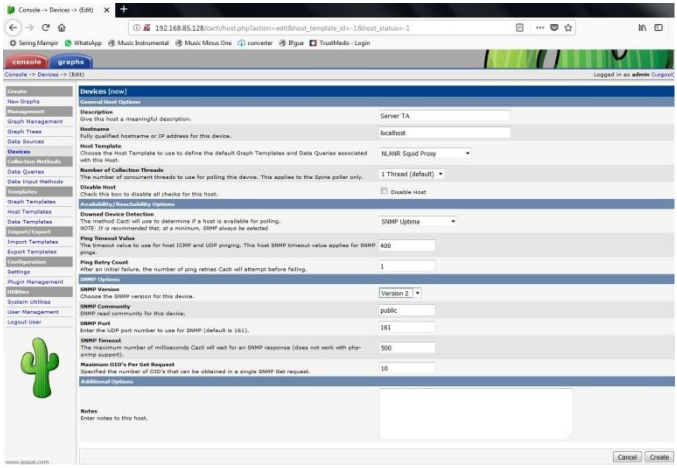
Gambar 3.64 Form add device cacti

3. Selanjutnya pilih Console – Device. Akan muncul List Device. Untuk menambahkan device baru klik Add



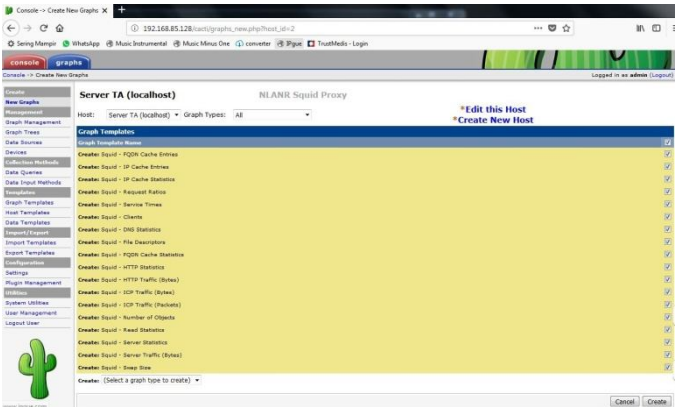
Gambar 3.65 List device cacti

4. Muncul form tambah device baru. Tambahkan device baru bernama Server TA. Karena Instalasi cacti dan squid berada pada mesin yang sama, yaitu server, maka pada Hostname masukan alamat host lokal, yaitu 127.0.0.1 atau localhost. Pilih NLANR sebagai input Host template, lalu pilih SNMP version 2



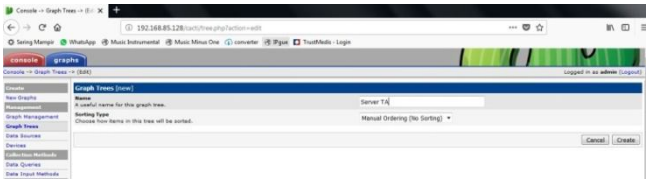
Gambar 3.66 Form add device cacti

5. Selanjutnya pilih Create – New Graphs. Pada opsi host pilih device yang sudah dbuat sebelumnya, yaitu Server TA. Terdapat 18 pilihan template, pilih semua



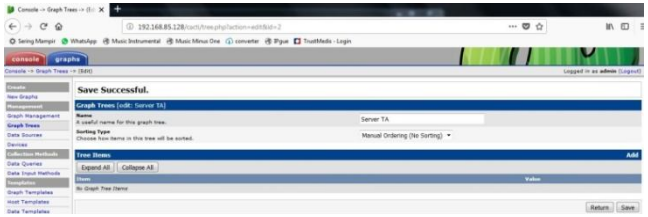
Gambar 3.67 Add new graph pada cacti

6. Selanjutnya adalah membuat tree baru. Untuk mebuat tree baru pilih opsi Management – Graph Trees lalu klik Add



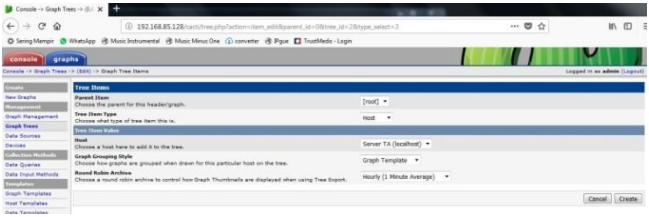
Gambar 3.68 Tambah graph tree baru cacti

Setelah tambah Graph Tree berhasil langkah selanjutnya adalah menambahkan item untuk tree yang dipilih. Klik add untuk menambahkan Tree Items



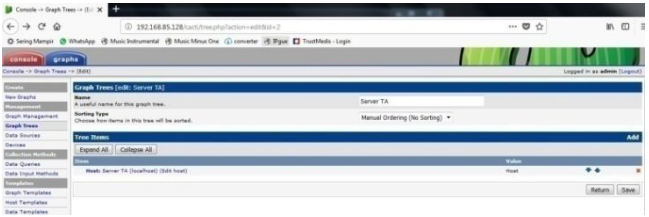
Gambar 3.69 Add graph tree baru berhasil

Pada opsi Tree Item Type, pilih Host. Lalu pada pilihan host lanjutan pilih device yang sudah dibuat sebelumnya, yaitu Server TA.



Gambar 3.70 Add tree items cacti

Setelah selesai maka akan muncul graph tree beserta tree item yang sudah dibuat.



Gambar 3.71 Graph tree dan tree item cacti

7. Untuk melihat grafik monitoring, pilih Graph, lalu pilih nama tree yang sudah dibuat, dalam hal ini Server TA

Gambar 3.72 Grafik monitoring pada cacti

Pada list grafik tersebut juga terdapat opsi untuk mendownload data ke dalam format csv.

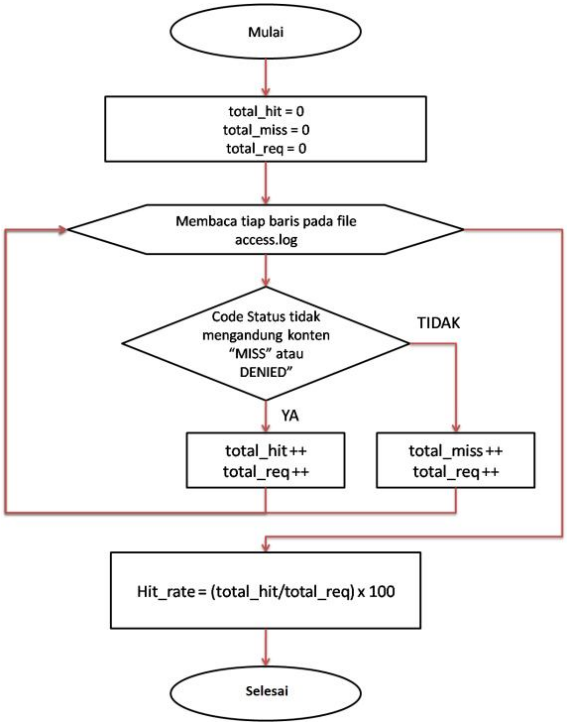
3.2.9 Pembuatan dan Kofigurasi Aplikasi Monitoring berbasis BASH

Terdapat 4 output parameter yang digunakan pada penelitian kali ini. Untuk itu juga terdapat 4 file program berkenstensi .sh yang

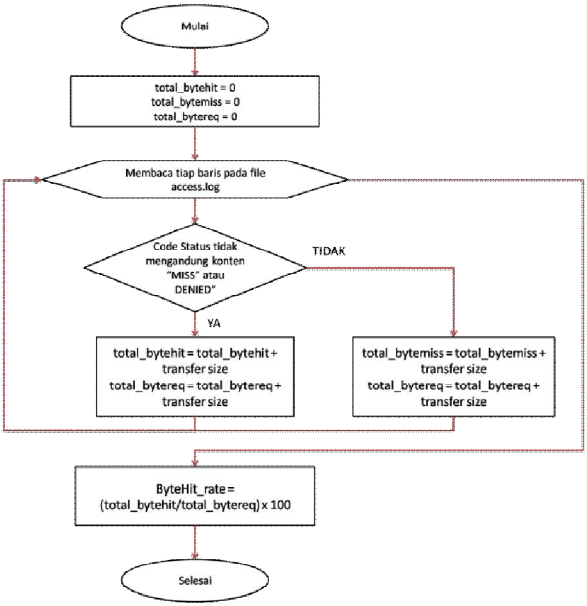
masing-masing digunakan untuk proses monitoring dan collecting data. Nama file program tersebut adalah

1. getAccessLogStat.sh
2. getAccessLogByteStat.sh
3. squidClientCountAIOStat.sh
4. squidClientUtilizationStat.sh

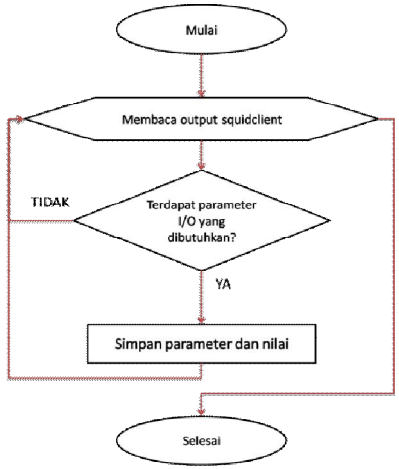
Masing-masing file program memiliki algoritma dan flowchart yang berbeda karena input yang digunakan juga berbeda. getAccessLogStat.sh dan getAccessbyteStat.sh menggunakan file access.log sebagai input yang akan diproses. Sedangkan squidClientCountAIOStat.sh dan squidClientUtilizationStat.sh menggunakan output squidclient sebagai input yang akan diproses.



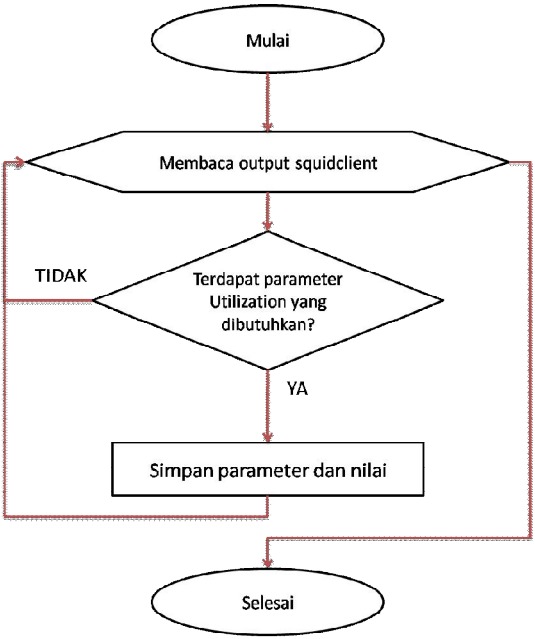
Gambar 3.73 Flowchart algoritma getAccessLogStat.sh



Gambar 3.74 Flowchart algoritma getAccessLogByteStat.sh



Gambar 3.75 Flowchart algoritma squidClientCountAIOStat.sh



Gambar 3.76 Flowchart algoritma squidClientUtilizationStat.sh

Dari algoritma tersebut nantinya akan dibuat sebuah program berbasis BASH yang bisa berjalan pada sistem operasi linux ubuntu. Hasil pembuatan program berbasis bash dengan mengacu pada flowchart yang sudah ada dapat dilihat pada bagian lampiran.

Untuk membuat program bash yang dapat dieksekusi di sistem operasi linux, langkah-langkahnya adalah

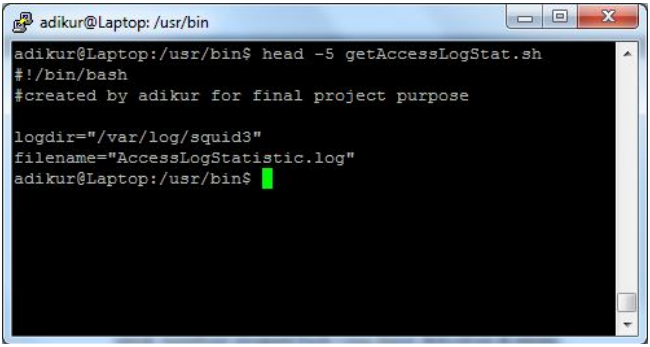
1. membuat file berekstensi shell, yaitu menggunakan command vi, vim, atau nano

   ```
   nano [nama_file].sh
   ```

   Pada baris awal file tambahkan script berikut

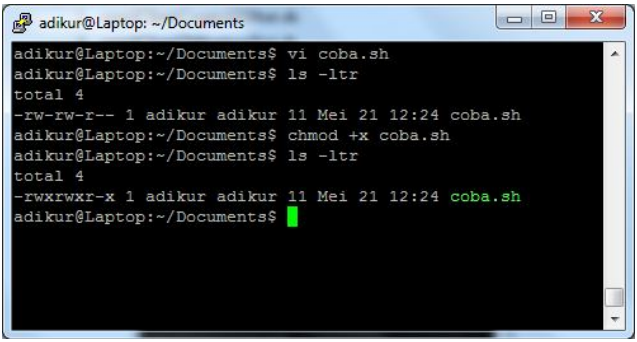
   ```
   #!/bin/bash
   ```

   agar nantinya ketika ditaruh didalam direktori bin, file bisa dieksekusi dimana saja



Gambar 3.77 Contoh file .sh

2. selanjutnya agar file tersebut bisa dieksekus,i tambahkan privilege dengan baris perintah
   `chmod +x [nama_file].sh`



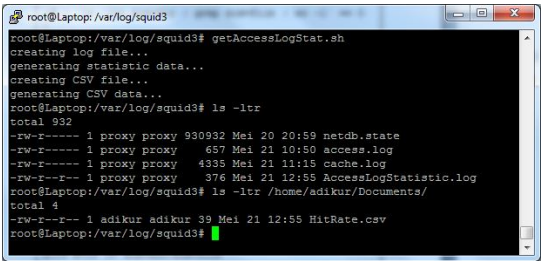
Gambar 3.78 Contoh mengubah privilege file

3. langkah terakhir adalah memindahkan file program ke dalam direktori bin. Direktori bin pada masing-masing distro sistem operasi berbeda. Pada linux ubuntu direktori bin trdapat pada `/usr/bin` sehinggfa perintah untuk memindahkan file adalah
   `mv [nama_file].sh /usr/bin`

Setelah file-file tersebut dieksekusi, terdapat file output yang muncul pada direktori /var/log/squid3 dan direktori default dokumen. Pada direktorti /var/lg/squid3. Program mengenerate file dengan nama dan format tertentu.

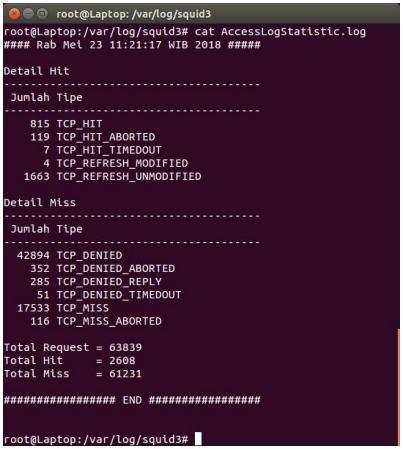
Tabel 3.10 Data output Program monitoring berbasis BASH

Program	Output
getAccessLogStat.sh	AccessLogStatistic.log Hit rate.csv
getAccessBytelogStat.sh	AccessLogByteStatistic.log ByteHit rate.csv
squidClientCountAIOStat.sh	squidClientAIOStatistic.log squidClientAIO.csv
squidClientUtilizationStat.sh	squidClientUtilStatistic.log squidClientUtilization.csv



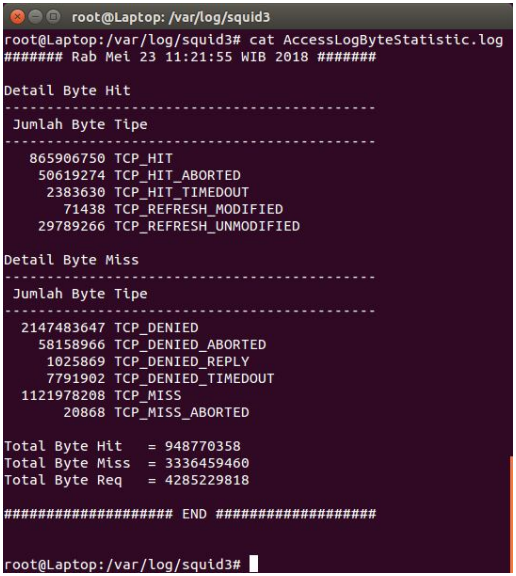
Gambar 3.79 Contoh hasil output program

File AccessLogStatistic.log menyimpan data per menit yang berisi jumlah hit, miss, request dan detailnya



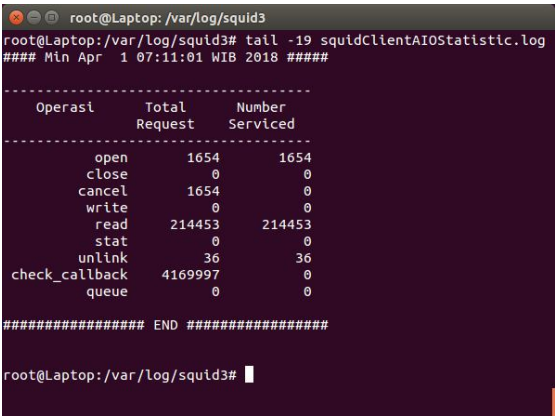
Gambar 3.80 Contoh data pada AccessLogStatistic.log

File AccessLogByteStatistic.log menyimpan data per menit yang berisi jumlah Byte hit, miss, request dan detailnya



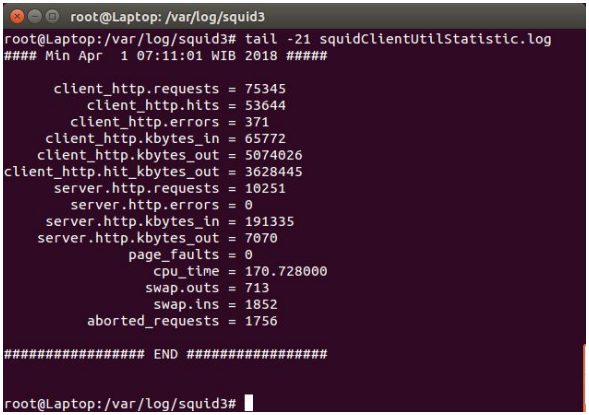
Gambar 3.81 Contoh data pada AccessLogByteStatistic.log

File squidClientAIOStatistic.log menyimpan data per menit yang berisi operasi I/O berserta nilainya



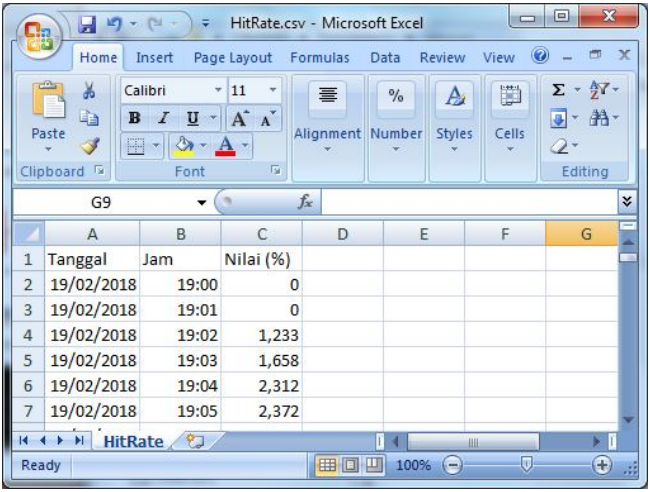
Gambar 3.82 Contoh data pada squidClientAIOStatistic.log

File squidClientUtilStatistic.log menyimpan data per menit yang berisi data parameter utilitas jaringan beserta nilainya



Gambar 3.83 Contoh data pada squidClientUtilStatistic.log

File csv masing-masing output program menyimpan data waktu dan parameter yang ada di dalam file log. File csv tersebut bisa langsung dibuka dan diolah menggunakan aplikasi pengolahan data yang sudah ada, semisal Microsoft Excel.



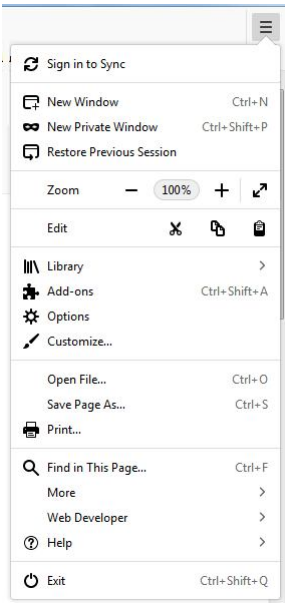
Gambar 3.84 Contoh data file csv output program

3.2.10 Konfigurasi Browser Client

Hal dasar yang perlu dikonfigurasi pada browser client adalah mengatur IP proxy. IP proxy dapat diatur secara manual maupun menggunakan addon browser. Browser yang digunakan pada simulasi ini adalah Mozilla Firefox

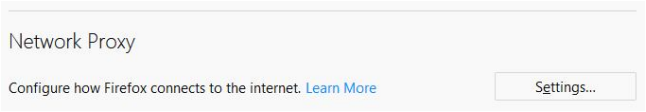
Langkah-langkah setting IP proxy secara manual pada Mozilla Firefox yaitu

1. Klik Tool – Options, atau gunakan shortcut pada pojok kanan atas browser



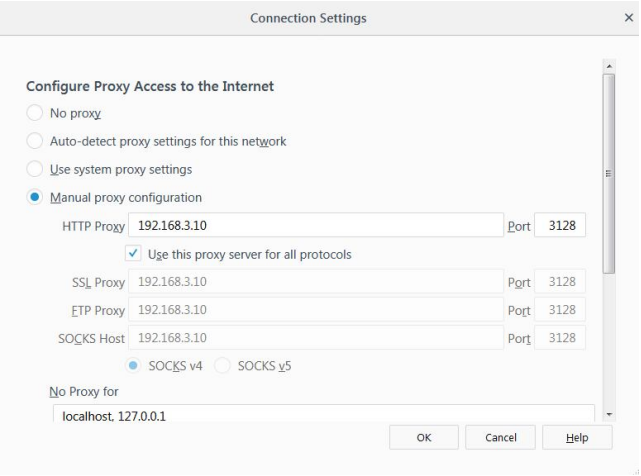
Gambar 3.85 Shortcut setting pada mozilla firefox

2. User akan dibawa menuju halaman konfigurasi. Setting proxy terdapat pada pilihan Network Proxy. Klik Settings...



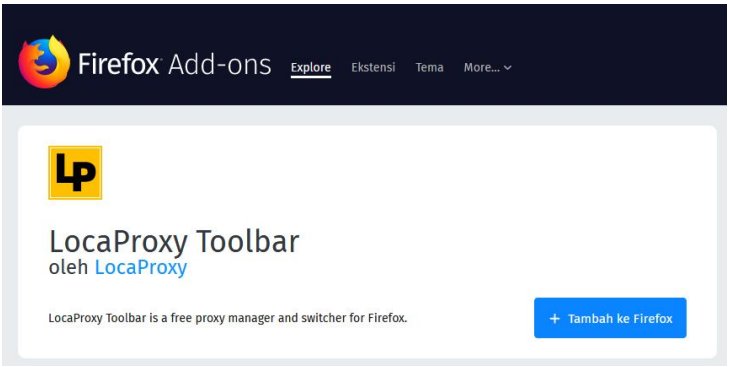
Gambar 3.86 Opsi network proxy

3. Pilih opsi manual proxy configuration, lalu masukan IP server proxy dan prot proxy yang digunakan



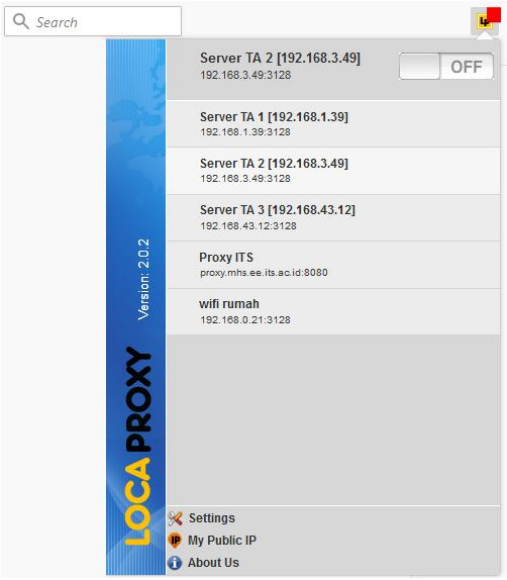
Gambar 3.87 Contoh setting detail proxy

Sedangkan addon browser yang digunakan untuk setting proxy, salah satu yang cukup populer adalah Locaproxy. Locaproxy terdapat dalam database addon mozilla fireox dan bisa diinstall ke dalam browser



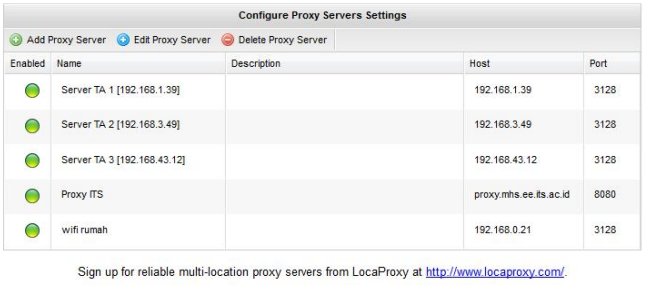
Gambar 3.88 Locaproxy firefox addon

Jika locaproxy sudah terpasang, pada pojok kanan atas browser akan muncul icon locaproxy yang jika di-klik maka akan muncul list IP proxy yang ada



Gambar 3.89 List IP proxy pada locaproxy

Button Settings digunakan untuk mengkonfigurasi IP proxy server yang ada.



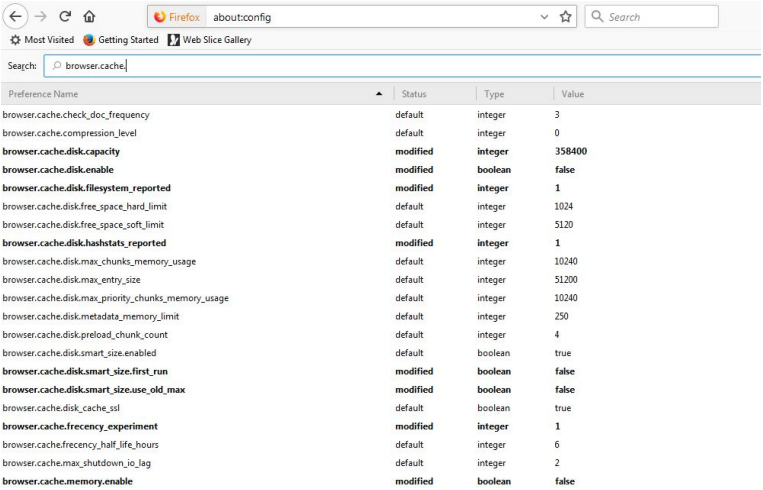
Gambar 3.90 Konfigurasi pada locaproxy

Sementara itu, agar bisa mengoptimalkan analisa algoritma page replacement, hendaknya proses caching hanya terjadi di server proxy. Sehingga konfigurasi caching pada browser harus dimatikan.

Seting untuk menon-aktifkan cache ada pada halaman konfigurasi fireox. Untuk masuk ke dalam halaman konfigurasi, ketikan

`about:config` pada url box. Lalu ubah nilai pada parameter `browser.cache.disk.enable` dan

`browser.cache.memory.enable` menjadi false

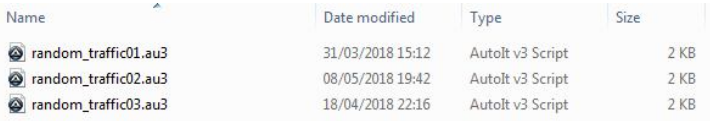


Gambar 3.91 Menon-aktifkan caching pada mozilla firefox

3.2.11 Automasi Trafik Pada Client

Selain beberapa hal yang sudah dijelaskan pada pemodelan, ada beberapa poin yang perlu diperhatikan pada pembuatn proses automasi trafik menggunakan AutoIt. Poin-poin tersebut diantaranya

1. Tiap-tiap client memiliki urutan akses halaman yang berbeda. Nantinya akan terdapat 3 file automasi, dimana urutan halaman yang dibuka berbeda antara client satu dengan yang lain.



Gambar 3.92 File automasi pada masing-masing client

2. Delay, delay digunakan setiap selesai suatu langkah automasi. Hal ini untuk mengantisipasi agar kinerja komputer tidak menjadi terlalu berat. Disepakati bahwa delay setelah membuka browser adalah 5 detik, delay setelah membuka website jenis A adalah 7 detik dan website jenis B adalah 14 detik.

Program automasi dibuat untuk mengakses data halaman url berdasarkan list yang sudah ditentukan sebelumnya. Di dalam program automasi, ditambahkan nilai delay untuk masing-masing website tergantung dari jenisnya. Berikut adalah contoh programnya

Tabel 3.11 Contoh script automasi

```
Run("C:\Program Files (x86)\Mozilla
Firefox\firefox.exe", "", @SW_MAXIMIZE)

$i = 0
While $i <= 52
   Opt("WinTitleMatchMode",2)
   WinActivate("firefox","")
   Sleep(5000)
   Send("      http://himit.pens.ac.id/")
   Send("{ENTER}")
   sleep(14000)
   Send("{F6}")
   Send("      http://1031gen.fm/")
   Send("{ENTER}")
   sleep(14000)
   Send("{F6}")
   Send("      http://news.liputan6.com/")
   Send("{ENTER}")
   sleep(14000)
   Send("{F6}")
   Send("
http://mangapark.net/manga/naruto/s3/c700")
   Send("{ENTER}")
   sleep(14000)
   Send("{F6}")
   Send("
http://mangapark.net/manga/bleach/s3/c686")
   Send("{ENTER}")
   sleep(14000)
   Send("{F6}")
```

```
    Send("      http://bioskopkeren.ws/")
    Send("{ENTER}")
    sleep(14000)
    Send("{F6}")
    Send("      http://cerpenmu.com/")
    Send("{ENTER}")
    sleep(7000)
    Send("{F6}")
    $i = $i + 1
WEnd

MsgBox(0, "Pesan", "Proses selesai");
```

Varibel i berkaitan dengan iterasi. Jumlah iterasi sesuai dengan pemodelan yang sudah dibuat. Sedangkan sleep menunjukan delay, dimana nilainya berbeda antara website jenis A dan B.

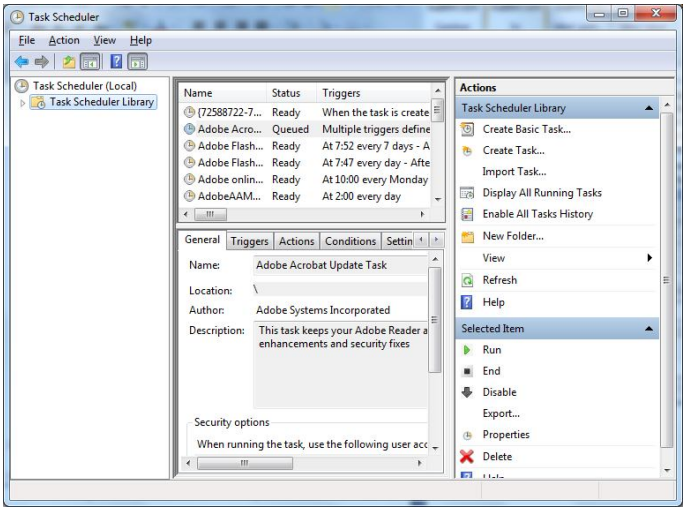
3.2.12 Scheduling

Terdapat dua jenis scheduling pad simulasi, yaitu scheduling pada client dan server. Pada client, scheduling digunakan untuk memulai proses automasi generate traffic guna memulai simulasi pada jaringan yang sudah ditentukan. Sedangkan pada server scheduling digunakan oleh program monitoring berbasis bash untuk mengumpulkan dan memonitoring data. Proses pengumpulan dan monitoring data berlangsung tiap menit

3.2.12.1 Scheduling Pada Client

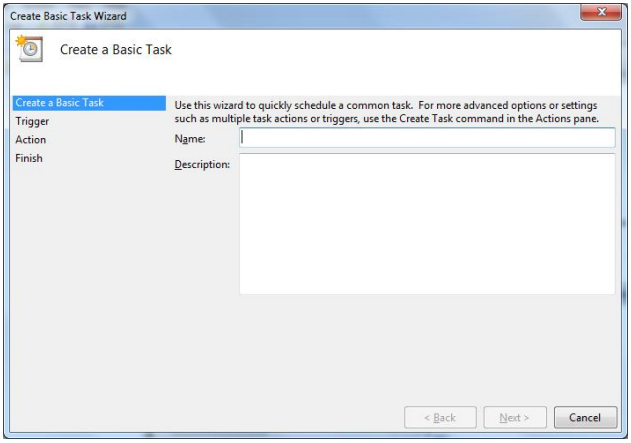
Pada proses simulasi, ditentukan bahwa dalam satu hari hanya satu kali skenario yang dijalankan setiap pukul 19.00 WIB. Scheduling diperlukan agar simulasi dapat berjalan secara automatis setiap harinya. Scheduling pada client menggunakan program Task Scheduler yang ada pada Windows 7. Berikut adalah langkah-langkahnya.

1. Sebelumnya pastikan file automasi sudah dibuat. Setelah itu buka task Scheduler. Ketikan task scheduler pada kotak pencarian dan pilih Windows task scheduler. Kemudian pilih bagian Task Scheduler Library



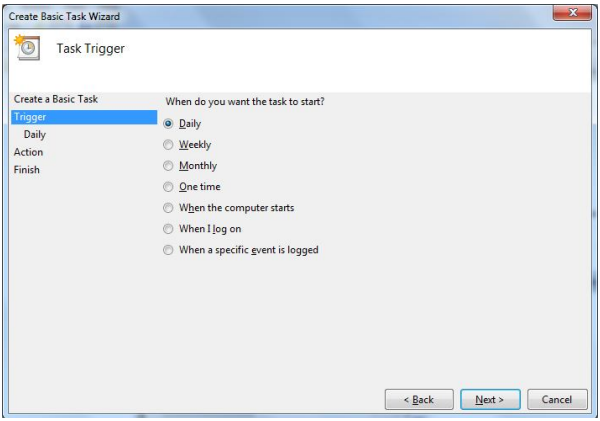
Gambar 3.93 Windows task scheduler

2. Plih Create Basic task, kemudian akan muncul kotak dialog. Input tertama adalah nama task



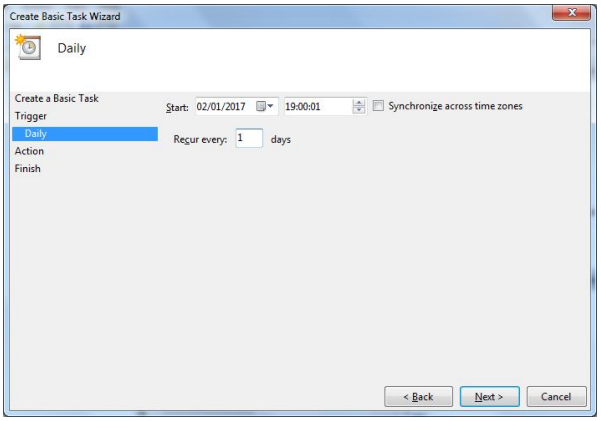
Gambar 3.94 Membuat task scheduler baru

3. Selanjutnya adalah pilihan trigger task. Pilih opsi daily



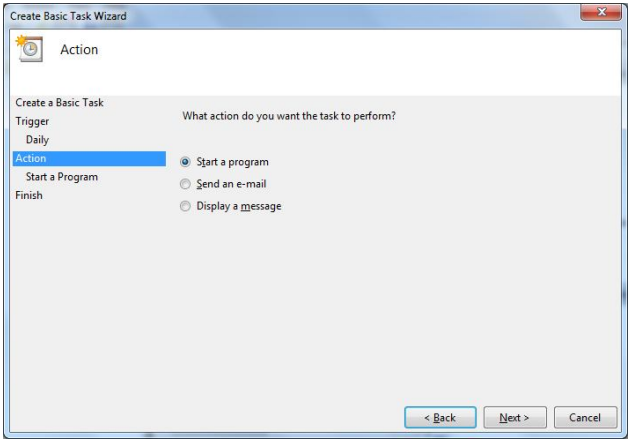
Gambar 3.95 Opsi trigger task scheduler

4. Selanjutnya masukan periode scheduling. Simulasi berjalan setiap pukul 19.00 pada periode tanggal tertentu

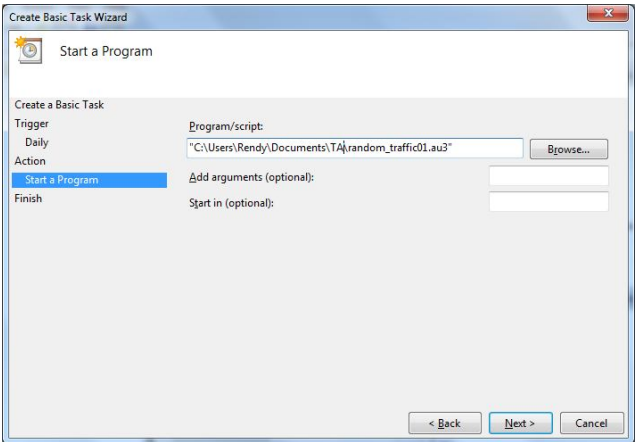


Gambar 3.96 Setting periode task scheduler

5. Pilih Start a Program kemudian pilih file automasi yang sudah dibuat.

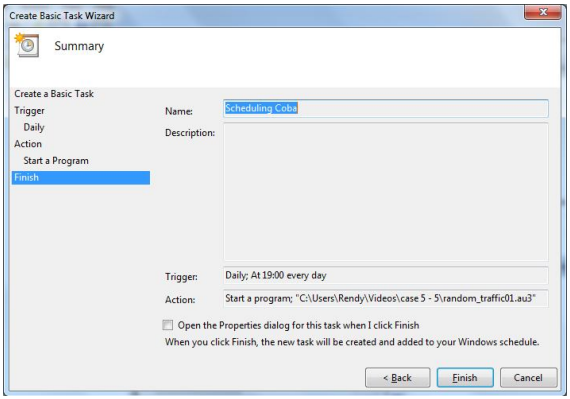


Gambar 3.97 Pilihan action pada task scheduler



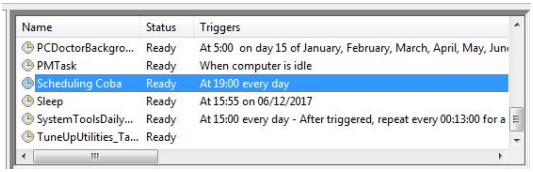
Gambar 3.98 Scheduling untuk file random_traffic01.au3

6. Kotak dialog akan menampilkan review task yang dibuat. Klik finish untuk menyelesaikan



Gambar 3.99 Review task scheduler

7. Task yang sudah dibuat akan terlihat di dalam daftar task yang ada di dalam task scheduler



Gambar 3.100 Daftar task pada scheduler

3.2.12.2 Scheduling Pada Server

Program monitoring berbasis bash yang sudah dibuat dijalankan melalui terminal dan akan menghasilkan output berupa file log dan csv. Porgram mnitoring ini hanya membaca input, yaitu access.log dan output squidclient, pada saat program dipanggil. Oleh karena itu diperlukan scheduling agar programm bisa berjalan tiap menit. Sebelum scheduling dibuat terlebih dulu ditentukan start time dan periode scheduling. Berdasarkan pemodelan dan mempertimbangkan kinerja komputer, ditentukan bahwa start time dimulai pukul 19.00. Sedangkan pada periode scheduling, nilainya berbeda dan sesuai dengan pemodelan.

Scheduling pada server linux ubuntu menggunakan bantuan cron. Untuk mengatur scheduling, gunakan pereintah `crontab -e`. Lalu masukan setting scheduling sesuai dengan start time dan periodenya.

Tabel 3.12 Cron scheduling untuk jaringan I

```
# m h  dom mon dow   command
*/1 19-23 * * * getAccessLogStat.sh
*/1 19-23 * * * getAccessLogByteStat.sh
*/1 19-23 * * * squidClientUtilizationStat.sh
*/1 19-23 * * * squidClientCountAIOStat.sh
*/1 0 * * * getAccessLogStat.sh
*/1 0 * * * getAccessLogByteStat.sh
*/1 0 * * * squidClientUtilizationStat.sh
*/1 0 * * * squidClientCountAIOStat.sh
0 1 * * * getAccessLogStat.sh
0 1 * * * getAccessLogByteStat.sh
0 1 * * * squidClientUtilizationStat.sh
0 1 * * * squidClientCountAIOStat.sh
```

Tabel 3.13 Cron scheduling untuk jaringan II

```
# m h  dom mon dow   command
*/1 19-21 * * * getAccessLogStat.sh
*/1 19-21 * * * getAccessLogByteStat.sh
*/1 19-21 * * * squidClientUtilizationStat.sh
*/1 19-21 * * * squidClientCountAIOStat.sh
0 22 * * * getAccessLogStat.sh
0 22 * * * getAccessLogByteStat.sh
0 22 * * * squidClientUtilizationStat.sh
0 22 * * * squidClientCountAIOStat.sh
```

Tabel 3.14 Cron scheduling untuk jaringan III

```
# m h  dom mon dow   command
*/1 19-20 * * * getAccessLogStat.sh
*/1 19-20 * * * getAccessLogByteStat.sh
*/1 19-20 * * * squidClientUtilizationStat.sh
*/1 19-20 * * * squidClientCountAIOStat.sh
0 21 * * * getAccessLogStat.sh
0 21 * * * getAccessLogByteStat.sh
0 21 * * * squidClientUtilizationStat.sh
0 21 * * * squidClientCountAIOStat.sh
```

3.3 Pengujian Sistem

3.3.1 Distribusi Trafik dan Periode Simulasi

Pengujian dilakukan dengan memasukan total trafik sebanyak 1560 halaman ke dalam jaringan. dengan mempertimbangan beberaoa poin, diantaranya variabel kondisi trafik dan jumlah list halaman yang sudah ditentukan saat pemodelan, maka didapatkan perhitungan distribusi trafik dan periode simulasinya.

Pada pengaturan distribusi trafik, iterasi untuk masing-masing kondisi jaringan berbeda sehingga total trafik yang digunakan nantinya tetap sama.

Tabel Distribusi trafik

Jumlah Client	**Iterasi**	**Kondisi 1 (2 : 8)**		**Kondisi 2 (5 : 5)**		**Kondisi 3 (8 : 2)**	
		Total A	Total B	Total A	Total B	Total A	Total B
1	156	312	1248	780	780	1248	312
2	78	156	624	390	390	624	156
3	42	104	416	260	260	416	104

Sementara pada periode simulasi, hasil perhitungan periode yang mempertimbangan nilai delay akan dibulatkan ke dalam satuan lima menit. Hal ini untuk memudahkan proses monitoring, terutama pada aplikasi monitoring cacti.

Tabel Periode simulasi

Jaringan	**Kondisi 1**	**Kondisi 2**	**Kondisi 3**
I	± 220 menit	± 275 menit	± 330 menit
II	± 110 menit	± 140 menit	± 165 menit
III	± 75 menit	± 95 menit	± 110 menit

Scheduling di sisi server untuk proses collecting dan monitoring nantinya akan mengambil nilai terbesar pada masing-masing jaringan

3.3.2 Skenario Pengujian

Pengujian dilakukan dengan mempertimbangkan variasi nilai yang ada pada masing-masing variabel, baik itu variabel primer maupun sekunder. Beberapa variasi yang ada diantaranya adalah

❖ Jaringan yang digunakan

Tabel

No.	Jaringan	Keterangan
1.	Jaringan I	Dalam satu jaringan, terdapat satu client
2.	Jaringan II	Dalam satu jaringan, terdapat dua client
3.	Jaringan III	Dalam satu jaringan, terdapat tiga client

❖ Jenis trafik

Tabel

No.	Jenis Trafik	Keterangan
1.	Jenis 1	Website jenis A lebih sedikit diakses daripada jenis B (prosentase 2 : 8)
2.	Jenis 2	Website jenis A dan jenis B seimbang (prosentase 5 : 5)
3.	Jenis 3	Website jenis A lebih banyak diakses daripada jenis B (prosentase 8 : 2)

❖ Algoritma page replacement

Tabel

No.	Algoritma	Keterangan
1.	Algoritma 1	Page replacement policy menggunakan algoritma LRU
2.	Algoritma 2	Page replacement policy menggunakan algoritma heap LRU
3.	Algoritma 3	Page replacement policy menggunakan algoritma LFUDA
4.	Algoritma 4	Page replacement policy menggunakan algoritma GDSF

Dari masing-masing variabel, ditemukan total skenario yang dilakukan adalah sebanyak

3 x 3 x 4 = 36 Skenario.

Berikut adalah daftar skenario simulasi yang dijalankan pada penelitian kali ini.

Tabel skenario

Skenario	Jaringan	Jenis	Algoritma
1	Jaringan I	Jenis 1	Algoritma 1
2	Jaringan I	Jenis 1	Algoritma 2
3	Jaringan I	Jenis 1	Algoritma 3
4	Jaringan I	Jenis 1	Algoritma 4
5	Jaringan I	Jenis 2	Algoritma 1
6	Jaringan I	Jenis 2	Algoritma 2
7	Jaringan I	Jenis 2	Algoritma 3
8	Jaringan I	Jenis 2	Algoritma 4
9	Jaringan I	Jenis 3	Algoritma 1
10	Jaringan I	Jenis 3	Algoritma 2
11	Jaringan I	Jenis 3	Algoritma 3
12	Jaringan I	Jenis 3	Algoritma 4
13	Jaringan II	Jenis 1	Algoritma 1
14	Jaringan II	Jenis 1	Algoritma 2
15	Jaringan II	Jenis 1	Algoritma 3
16	Jaringan II	Jenis 1	Algoritma 4
17	Jaringan II	Jenis 2	Algoritma 1

18	Jaringan II	Jenis 2	Algoritma 2
19	Jaringan II	Jenis 2	Algoritma 3
20	Jaringan II	Jenis 2	Algoritma 4
21	Jaringan II	Jenis 3	Algoritma 1
22	Jaringan II	Jenis 3	Algoritma 2
23	Jaringan II	Jenis 3	Algoritma 3
24	Jaringan II	Jenis 3	Algoritma 4
25	Jaringan III	Jenis 1	Algoritma 1
26	Jaringan III	Jenis 1	Algoritma 2
27	Jaringan III	Jenis 1	Algoritma 3
28	Jaringan III	Jenis 1	Algoritma 4
29	Jaringan III	Jenis 2	Algoritma 1
30	Jaringan III	Jenis 2	Algoritma 2
31	Jaringan III	Jenis 2	Algoritma 3
32	Jaringan III	Jenis 2	Algoritma 4
33	Jaringan III	Jenis 3	Algoritma 1
34	Jaringan III	Jenis 3	Algoritma 2
35	Jaringan III	Jenis 3	Algoritma 3
36	Jaringan III	Jenis 3	Algoritma 4

Satu skenario dijalankan setiap hari Senin s.d. Jumat, dimulai pada pada pukul 19.00 sampai dengan selesai. Sehingga total hari yang dibutuhkan adalah 36 hari.

3.3.3 Parameter Pengukuran

Guna menilai performa proxy server, pada setiap simulasi, akan didapatkan data nilai output dari beberapa parameter yang sudah ditentukan. Performa proxy secara umum dilihat dari dua hal, yaitu dari segi performa trafik dan performa hardware server. Parameter yang masuk ke dalam kategori performa trafik diantaranya hit rate, bit hite rate dan swap I/O. Sedangkan parameter yang masuk ke dalam pertimbangan performa hardware adalah read rata-rata, CPU time, dan file descriptors.

Perhitungan parameter-parameter tersebut, seperti hit rate pada rumus 2.1 , byte hit rate pada rumus 2.2 , dan read rata-rata pada rumus 2.3 Akan diimplementasikan ke dalam program monitoring berbasis bash. Begitu juga parameter lainnya kecuali file descriptor yang dihitung menggunakan tool monitoring Cacti.

[Halaman ini sengaja dikosongkan]

BAB 4
HASIL DAN ANALISA DATA

4.1 Hasil Simulasi

Data hasil simulasi diperoleh dan dikumpulkan setiap sebuah skenario selesai dijalankan. Program monitoring berbasis BASH dan cacti monitoring tool digunakan untuk mengumpulkan hasil simulasi berupa output parameter yaitu Hit rate, Byte hit rate, I/O parameter (read dan check_callback), Utilization parameter (CPU time dan swap I/O), serta parameter NLANR cacti (file descrptors).

Data hasil output tersebut lalu dikelompokan ke dalam masing-masing faktor parameter yang mempengaruhi performa trafik dan hardware sebuah proxy server. Performa trafik proxy server dilihat dari output hit rate, byte hit rate, swap input dan swap output. Sedangkan performa hardware dilihat dari read, check_callback, CPU time, dan unused file descriptors.

Tabel 4.1 Output parameter terkait performa trafik

Skenario	Hit rate peak (%)	Hit rate akhir (%)	Byte hit rate peak (%)	Byte hit rate akhir (%)	Swap input	Swap output
1	2,695	2,643	31,258	27,845	466	921
2	2,654	2,643	33,739	29,661	464	891
3	2,664	2,406	34,160	28,310	448	949
4	2,605	2,551	34,579	29,717	504	921
5	3,522	3,519	36,618	28,902	1976	780
6	3,327	3,323	31,920	28,546	1958	870
7	3,526	3,522	36,334	30,096	1996	861
8	3,384	3,384	34,696	28,763	2000	828
9	3,673	3,637	37,015	36,640	2444	952
10	3,633	3,598	35,581	34,672	2388	999
11	3,663	3,603	36,003	35,210	2364	919
12	3,635	3,579	33,558	33,300	2320	959

13	3,037	3,029	34,767	30,274	504	1053
14	3,014	2,983	34,293	31,302	504	1050
15	3,005	2,997	32,850	30,519	504	1034
16	3,043	3,030	29,269	29,269	488	1048
17	3,616	3,616	22,571	21,506	2018	1011
18	3,304	3,304	21,881	20,067	1542	665
19	3,496	3,490	21,314	20,874	1622	1006
20	3,793	3,790	22,160	20,977	1528	730
21	3,511	3,489	24,004	20,252	2055	676
22	3,990	3,981	32,432	27,235	2363	922
23	3,955	3,954	30,415	27,630	2364	961
24	4,200	4,091	34,241	26,921	2176	1043
25	3,124	3,123	25,657	24,296	489	1022
26	3,098	3,084	27,589	26,168	500	988
27	3,185	3,185	30,069	27,387	510	1989
28	3,161	3,152	29,929	26,328	487	1036
29	3,426	3,425	21,250	21,033	1530	1338
30	3,613	3,598	22,698	22,379	1587	714
31	3,561	3,540	22,525	21,343	1572	888
32	4,253	4,177	23,381	21,496	1536	735
33	3,434	3,294	22,618	16,104	1590	1018
34	4,065	4,054	31,250	25,988	1884	559
35	4,366	4,366	32,513	26,217	2325	899
36	4,380	4,354	31,554	25,944	689	1267

Peak Hit rate merupakan nilai hit rate terbeser yang di dapat selama proses simulasi skenario berjalan. Hit rate akhir adalah nilai hit rate setelah simulasi skenario selesai dijalankan. Demikian pula untuk Peak Byte hit rate selama simulasi dan byte hit rate akhir setelah proses simulasi. Swap input adalah jumlah boyek yang masuk ke dalam direktori swap, sedangkan swap output adalah jumlah obyek yang keluar/diganti pada direktori swap.

Tabel 4.2 Output parameter terkait performa hardware

Ske-nario	Read	Check_callback	Read rata-rata (%)	CPU time	Unused File Descriptors
1	124910	3336127	3,744	133,768	65,39
2	123594	3385904	3,650	126,964	65,39
3	121362	3595661	3,375	133,096	65,38
4	135324	3653663	3,704	140,832	65,38
5	310350	5959522	5,208	268,524	65,38
6	311559	6009881	5,184	269,792	65,38
7	317734	5922830	5,365	257,552	65,39
8	318729	6327904	5,037	281,308	65,38
9	387717	5803371	6,681	245,256	65,36
10	367501	5588392	6,576	236,228	65,34
11	375350	5589513	6,715	247,724	65,35
12	351573	5625217	6,250	235,528	65,35
13	112492	1965548	5,723	101,720	65,31
14	113459	2093260	5,420	102,920	65,31
15	113207	2149586	5,266	101,512	65,33
16	109606	2268028	4,833	108,072	65,31
17	178611	3708308	4,817	184,796	65,17
18	180641	3997801	4,519	176,736	65,17
19	181820	3419871	5,317	163,468	65,16

20	185380	3252349	5,700	153,380	65,23
21	196466	4690367	4,189	223,276	65,06
22	303040	4124728	7,347	203,296	65,20
23	310907	4175479	7,446	206,888	65,24
24	290770	4155952	6,996	209,628	65,22
25	83604	2448043	3,415	91,712	65,22
26	88595	1895970	4,673	98,304	65,22
27	89797	2018295	4,449	95,976	65,21
28	87991	1900366	4,630	96,620	65,22
29	176999	3181685	5,563	182,164	65,00
30	185036	3300355	5,607	154,888	65,07
31	180599	3192797	5,656	164,800	65,04
32	178993	2758545	6,489	144,188	65,11
33	176977	3378534	5,238	196,776	65,11
34	261345	3536863	7,389	218,568	65,06
35	259599	3327281	7,802	187,632	65,07
36	257468	3391705	7,591	163,192	65,08

Read merupakan proses membaca request dan menunjukan bahwa buffering sedang berjalan, sementara check_callback menandakan proses checking untuk antrian yang sudah selesah dieksekusi. CPU time merupakan waktu yang digunakan oleh server untuk memproses data pada squid proxy. File descriptors merupakan suatu object yang digunakan untuk mekanisme I/O suatu socket atau proses, dalam hal ini adalah proses squid proxy. Unused file descriptors artinay adalah sisa alokasi memori setelah digunakan untuk file descriptors.

4.2 Analisa Skenario

Dari hasil output pada semua skenario yang dilakukan, didapatkan data statistik dari masing-masing parameter adalah sebagai berikut

Tabel 4.3 Statistik nilai output setelah simulasi

Parameter	Nilai Rata-rata	Maksimal/Minimal	
		Nilai	Skenario ke-
Hit rate peak	3,461	4,380	36
Hit rate akhir	3,431	4,366	35
Byte Hit rate peak	29,908	37,015	9
Byte hit rate akhir	26,755	36,640	9
Swap Input	1435,972	2444	9
Swap Output	958,389	1989	27
Read rata-rata	5,488	7,802	35
CPU time	175,197	91,712	25
Unused file descriptors	65,239	65	29

Hasil output parameter-parameter tsersebut lalu akan dibandingkan untuk tiap skenario dan casenya. Case adalah perpindahan dari satu skenario menuju skenario lain dalam kondisi salah satu nilai variabel sekunder sama. Sehingga terpdapat 36 skenario atau 24 case yang akan diolah dan dianalisa.

Proses analisa dilakukan untuk setiap paramater output yang digunakan, yaitu hit rate, byte hit rate, swap I/O, read rata-rata, CPU time, dan unused file descriptors. Analisa dilakukan untuk mengetahui pengaruh variabel sekunder terhadap parameter output. Berikut hasil analisa yang dilakukan

1) Pada jaringan dengan jumlah client yang sama, semakin besar rasio website B dibanding dengan website A, semakin besar hit rate yang dihasilkan
2) Pada kondisi trafik yang sama, dari 24 case, 21 case menunjukan bahwa semakin banyak jumlah client, semakin besar hit rate yang dihasilkan
3) Pada jaringan dengan jumlah client yang sama, dari 24 case, 11 case menunjukan menunjukan kenaikan byte hit rate dan 13 case menunjukan penurunan byte hit rate
4) Pada kondisi trafik yang sama, dari 24 case, 19 case menunjukan bahwa semakin banyak jumlah client, semakin kecil byte hit rate yang dihasilkan
5) Pada kondisi dimana rasio website B lebih besar atau sama dengan A, jumlah swap input selalu lebih besar dari pada swap output
6) Pada jaringan dengan jumlah client yang sama, dari 24 case, 21 case menunjukan bahwa semakin besar rasio website B dibanding dengan website A, semakin besar read rata-rata yang dihasilkan
7) Pada kondisi trafik yang sama, dari 24 case, 16 case menunjukan bahwa semakin banyak jumlah client, semakin besar read rata-rata yang dihasilkan
8) Pada jaringan dengan jumlah client yang sama, dari 24 case, 20 case menunjukan bahwa semakin besar rasio website B dibanding dengan website A, semakin besar CPU time pada sebuah server
9) Pada jaringan kondisi trafik yang sama, dari 24 case, 20 case menunjukan bahwa semakin banyak jumlah client, semakin besar CPU time pada sebuah server
10) Pada kondisi trafik yang sama, semakin banyak jumlah client, semakin kecil rata-rata unused file descriptors
11) Pada jaringan dengan jumlah client yang sama, dari 24 case, 23 case menunjukan bahwa semakin besar rasio website B dibanding dengan website A, semakin kecil rata-rata unused file descriptors

4.3 Analisa Perbandingan Algoritma

Analisa perbandingan algoritma dilakukan dalam kondisi dimana nilai variabel sekundernya sama. Jadi dari 36 skenario yang ada, total case yang digunakan sebanyak 9 case. Proses analisa dilakukan untuk setiap paramater output yang digunakan, yaitu hit rate, byte hit rate, swap I/O, read rata-rata, CPU time, dan unused file descriptors. Pada masing-masing case akan dilihat urutan berdasarkan kriteria maksimal atau minimal yang sudah ditentukan sebelumnya.

Tabel 4.4 Urutan hit rate terbesar

Jaringan	**Jenis Trafik**	**LRU**	**heap LRU**	**LFUDA**	**GDSF**
I	1	1st	2nd	4th	3rd
I	2	2nd	4th	1st	3rd
I	3	1st	3rd	2nd	4th
II	1	2nd	4th	3rd	1st
II	2	2nd	4th	3rd	1st
II	3	4th	2nd	3rd	1st
III	1	3rd	4th	1st	2nd
III	2	4th	2nd	3rd	1st
III	3	4th	3rd	1st	2nd
Bobot		23	34	21	18
Urutan		*3rd*	*4th*	*2nd*	*1st*

Perbandingan antar algoritma dilihat dari nilai yang palig besar. Didapatkan pada parameter hit rate (HR), hasil terbaik didapatkan saat menggunakan algoritma GDSF sedangkan penggunaan algoritma heap LRU menghasilkan hit rate paling buruk.

Tabel 4.5 Urutan byte hit rate terbesar

Jaringan	Jenis Trafik	LRU	heap LRU	LFUDA	GDSF
I	1	4th	2nd	3rd	1st
I	2	2nd	4th	1st	3rd
I	3	1st	3rd	2nd	4th
II	1	3rd	1st	2nd	4th
II	2	1st	4th	3rd	2nd
II	3	4th	3rd	2nd	1st
III	1	4th	3rd	1st	2nd
III	2	4th	1st	3rd	2nd
III	3	4th	2nd	1st	3rd
Bobot		27	23	18	22
Urutan		*4th*	*3rd*	*1st*	*2nd*

Sama seperti hit rate, perbandingan algortima dilihat dari nilai terbesar. Dari hasil output byte hit rate (BHR), hasil terbaik didapatkan saat menggunakan algoritma LFUDA sedangkan penggunaan algoritma LRU menghasilkan hit rate paling buruk

Tabel 4.6 Urutan read rata-rata terbesar

Jaringan	Jenis Trafik	LRU	heap LRU	LFUDA	GDSF
I	1	1st	3rd	4th	2nd
I	2	2nd	3rd	1st	4th
I	3	2nd	3rd	1st	4th
II	1	1st	2nd	3rd	4th
II	2	3rd	4th	2nd	1st
II	3	4th	2nd	1st	3rd
III	1	4th	1st	3rd	2nd
III	2	4th	3rd	2nd	1st
III	3	4th	3rd	1st	2nd
Bobot		25	24	18	23
Urutan		4th	3rd	1st	2nd

Melihat pada tabel read rata-rata (AR), hasil terbaik didapatkan saat menggunakan algoritma LFUDA sedangkan penggunaan algoritma heap LRU menghasilkan hit rate paling buruk.

Tabel 4.7 Urutan rasio swap I/O terkecil

Jaringan	**Jenis Trafik**	**LRU**	**heap LRU**	**LFUDA**	**GDSF**
I	1	2nd	3rd	1st	4th
I	2	4th	1st	2nd	3rd
I	3	3rd	1st	3rd	2nd
II	1	2nd	3rd	3rd	1st
II	2	2nd	4th	1st	3rd
II	3	4th	3rd	2nd	1st
III	1	3rd	4th	1st	2nd
III	2	1st	4th	2nd	3rd
III	3	2nd	4th	3rd	1st
Bobot		23	27	18	20
Urutan		3rd	4th	1st	2nd

Pada Swap I/O disini adalah perbandingan swap input dengan swap output. Pada tiap case, efektivitas antar algoritma diambil dengan melihat nilai terkecil. Sehingga hasil terbaik didapatkan saat menggunakan algoritma LFUDA sedangkan penggunaan algoritma heap LRU menghasilkan hit rate paling buruk.

Tabel 4.8 Urutan CPU time terkecil

Jaringan	**Jenis Trafik**	**LRU**	**heap LRU**	**LFUDA**	**GDSF**
I	1	3rd	1st	2nd	4th
I	2	2nd	3rd	1st	4th
I	3	3rd	2nd	4th	1st
II	1	1st	3rd	2nd	4th
II	2	4th	3rd	2nd	1st
II	3	4th	1st	2nd	3rd
III	1	1st	4th	3rd	2nd
III	2	4th	3rd	2nd	1st
III	3	3rd	4th	2nd	1st
Bobot		25	24	20	21
Urutan		4th	3rd	1st	2nd

efektivitas antar algoritma pada tabel CPU time diambil dengan melihat nilai terkecil Didapatkan saat menggunakan algoritma LFUDA sedangkan penggunaan algoritma heap heap LRU menghasilkan hit rate paling buruk.

Tabel 4.9Urutan nilai unused file descriptors

Jaringan	**Jenis Trafik**	**LRU**	**heap LRU**	**LFUDA**	**GDSF**
I	1	1st	1st	2nd	2nd
I	2	2nd	2nd	1st	2nd
I	3	1st	3st	2nd	2nd
II	1	1st	1st	2nd	1st
II	2	2nd	2nd	1st	3rd
II	3	4th	3rd	1st	2nd
III	1	1st	1st	2nd	1st
III	2	4th	2nd	3rd	1st
III	3	1st	4th	3rd	2nd
Bobot		17	19	17	16
Urutan		2nd	3rd	2nd	1st

Sementara pada parameter unused file descriptors, perbandingan antar algoritma dilihat dair nilai yang laing besar. Hasil terbaik didapatkan saat menggunakan algoritma GDSF sedangkan penggunaan algoritma heap LRU menghasilkan hit rate paling buruk.

BAB 5
KESIMPULAN DAN SARAN

Setelah serangkaian pemodelan dan penentuan parameter dan proses simulasi sehingga menghasilkan data-data output yang lalu kemudian dianalisa, maka sebuah kesimpulan dapat dibuat. Pembahasan pada penelitina kali ini juga dapat digunakan dalam pengembangan penelitian dengan tema sejenis.

5.1 Kesimpulan

Beberapa kesimpulan yang dapat ditarik dari proses penelitian kali ini yaitu

1. Terdapat beberapa macam cara monitoring sebuah jaringan proxy server, diantaranya menggunakan squid log file, squidclient, dan aplikasi monitoring cacti.
2. Jumlah client mempengaruhi nilai hit rate, byte hit rate, read rata-rata, CPU time, dan unused file descriptors. Sedangkan kondisi trafik mempengaruhi nilai hit rate, read rata-rata, CPU time, swap I/O, dan unused file descriptors
3. Berdasarkan parameter utama yang biasa dijadikan acuan efektivitas, yaitu nilai hit rate, dari hasil analisa dapat disimpulkan bahwa algoritma paling efektif adalah GDSF, dimana dari 9 case, algoritma GDSF memiliki hit rate terbesar pada 4 case, yaitu 3.303, 3.79, 4.091, dan 4.177
4. Berdasarkan 6 nilai parameter secara keseluruhan, dari hasil analisa dapat disimpulkan bahwa algoritma paling efektif adalah LFUDA, lalu disusul dengan GDSF. Algoritma LFUDA memiliki nilai paling efektif pada 4 parameter output, yaitu byte hit rate (BHR), read rata-rata (AR), Swap I/O, dan CPU time. Sedangkan GDSF memiliki nilai paling efektif pada 2 parameter output, yaitu hit rate dan unused file descriptors.
5. Dari hasil analisa juga didapatkan bahwa algoritma paling tidak efektif adalah LRU, kemudian disusul heap LRU.

5.2 Saran

Sangat disadari bahwa penelitian kali ini masih jauh dat kata sempurna. Untu itu beberapa saran yang dapat digunakan untuk pengembangan penelitian diantaranya

1. Perbanyak nilai variabel sekunder, jumlah client dan kondisi trafik
2. Penentuan sampel halaman url yang lebih spesifik
3. Perbanayak jumlah trafik simulasi

DAFTAR PUSTAKA

[1] Bramantyo, F. W. 2016. “Perbandingan Hit Ratio dan Byte Hit Ratio Squid Untuk Proxy Server Dalam Penghapusan File Caching Berdasarkan Metode LRU, LFUDA, dan GDSF”. Yogjakarta Universitas Sanata Dharma

[2] Hardyanto, Tri. 2015. “Analisa Unjuk Kerja Pengaruh Hit Ratio Pada Squid Proxy Terhadap Sumber Daya Komputer Server”. Yogjakarta Universitas Sanata Dharma

[3] Wessels, Duane. 2004. “Squid The Definite Guide”. California O’Reilly

[4] Balukh, B. A. 2010. “Analisis Log dan Metode Cache Replacement Untuk Optimalisasi Proxy Server Studi Kasus PT. Garuda Indonesia”. Jakarta Universitas Gunadharma

[5] Srijit Banerjee's Blog. “How to monitor Squid Proxy Server using Cacti in Ubuntu Server 13.04”. 4 Desember 2017. http://srijit.com/how-to-monitor-squid-proxy-server-using-cacti-in-ubuntu-server-13-04/

[6] Squid home page. “squid: Optimizing Web Delivery”. 9 Oktober 2017. http://www.squid-cache.org/

[7] Dezwart Cacti Squid. “Cacti NLANR Squid Host Template”. 4 Desember 2017. https://github.com/dezwart/cacti-squid

[Halaman ini sengaja dikosongkan]

BIOGRAFI PENULIS

Adi Kurniawan, lahir di Surabaya pada tanggal 27 Juli 1991. Merupakan anak ke-2 dari 3 bersaudara dan satu-satunya laki-laki diantara ketiga anak pasangan dari Maskur dan Anik Rochani. Setelah lulus dari SMPN 26 Surabaya pada tahun 2002, penulis lalu melanjutkan jenjang pendidikan mengengah atas di SMAN 11 Surabaya. Setelah mengambil program vokasi selama satu tahun, penulis lalu melanjutkan kuliah diploma jurusan Teknik Informatika PENS dan lulus pada tahun 2012. Lalu pada tahun 2014, penulis melanjutkan mengambil program lintas jalur guna mendapatkan gelar sarjana Telekomunikasi pada jurusan Elektro ITS. Selama masa kuliah, penulis aktif pada beberapa kegiatan kemahsiswaan dan komunitas, diantaranya menjadi ketua departemen Pengembangan Sumber Daya Mahasiswa Himpunan Mahasiswa Teknik Informatika PENS serta aktif dalam komunitas multimedia PENS. Penulis juga aktif dibeberapa kegiatan sosial berplatform Indonesia Mengajar seperti Kelas Inspirasi dan Festival Gim. Penulis memiliki beberapa pengalaman kerja, diantaranya menjadi Core Engineer selama dua tahun di PT. Huawei Services dan Staff Business and Support di PT. Trust Solusindo selama satu tahun. Penulis juga mempunyai pengalaman sebagai IT freelance dengan beberapa portofolio proyek.

[Halaman ini sengaja dikosongkan]

LAMPIRAN

6.1 Script getAccessLogStat.sh

```
#!/bin/bash
#created by adikur for final project purpose

logdir="/var/log/squid3"
filename="AccessLogStatistic.log"
getTime=`date +%H:%M`
getDate=`date +%d/%m/%y`
activeUser=`who | awk '{print $1}' | uniq -c | awk
'{print $2}'`
csvfile="Hit rate.csv"
csvdir="/home/$activeUser/Documents"
totalReq=`cat $logdir/access.log | wc -l`
totalHit=`cat $logdir/access.log | grep -v _MISS |
grep -v _DENIED | awk '{print $4}' | awk -F'/'
'{print $1}' | wc -l`
totalMiss=$((totalReq-totalHit))
Hit rate=$(echo "scale=3; $totalHit*100/$totalReq"
| bc)

if [ `ls -ltr $logdir | grep $filename | wc -l` ==
0 ]
then
       echo "creating log file..."
       > $logdir/$filename
       chown proxy:proxy $logdir/$filename
fi

echo "generating statistic data..."
printf "#### `date` #####\n\n"  >>
$logdir/$filename
echo "Total Request = $totalReq" >>
$logdir/$filename
```

```
echo "Total Hit     = $totalHit" >>
$logdir/$filename
echo "Total Miss    = $totalMiss" >>
$logdir/$filename


echo -e "\nDetail Hit" >> $logdir/$filename
echo "--------------------------------------" >>
$logdir/$filename
echo " Jumlah Tipe" >> $logdir/$filename
echo "--------------------------------------" >>
$logdir/$filename
cat $logdir/access.log | grep -v _MISS | grep -v
_DENIED | awk '{print $4}' | awk -F'/' '{print $1}'
| sort | uniq -c >> $logdir/$filename
if [ $totalHit -eq 0 ]
then
       echo "      - -" >> $logdir/$filename
fi


echo -e "\nDetail Miss" >> $logdir/$filename
echo "--------------------------------------" >>
$logdir/$filename
echo " Jumlah Tipe" >> $logdir/$filename
echo "--------------------------------------" >>
$logdir/$filename
cat $logdir/access.log | grep '_MISS\|_DENIED' |
awk '{print $4}' | awk -F'/' '{print $1}' | sort |
uniq -c >> $logdir/$filename
if [ $totalMiss -eq 0 ]
then
       echo "      - -" >> $logdir/$filename
fi


echo -e "\n################## END
##################\n\n" >> $logdir/$filename


if [ `ls -ltr $csvdir | grep $csvfile | wc -l` == 0
```

```
]
then
       echo "creating CSV file..."
       echo "Tanggal;Jam;Nilai (%)" >
$csvdir/$csvfile
       chown $activeUser:$activeUser
$csvdir/$csvfile
fi
echo "generating CSV data..."
printf "%s;%s;%s\n" $getDate $getTime $Hit rate>>
$csvdir/$csvfile
```

6.2 Script getAccessByteLogStat.sh

```
#!/bin/bash
#Created by Adikur for final project purpose

logdir="/var/log/squid3"
filename="AccessLogByteStatistic.log"
getTime=`date +%H:%M`
getDate=`date +%d/%m/%y`
activeUser=`who | awk '{print $1}' | uniq -c | awk
'{print $2}'`
csvfile="ByteHit rate.csv"
TotalByteHit=0
TotalByteMiss=0
csvdir="/home/$activeUser/Documents"
listHit=`cat $logdir/access.log | grep -v
'MISS\|DENIED' | awk '{print $4}' | awk -F'/' '{print
$1}' | sort | uniq -c | awk '{print $2}'`
listMiss=`cat $logdir/access.log | grep
'MISS\|DENIED' | awk '{print $4}' | awk -F'/' '{print
$1}' | sort | uniq -c | awk '{print $2}'`

if [ `ls -ltr $logdir | grep $filename | wc -l` == 0
]
then
       echo "creating log file..."
       > $logdir/$filename
       chown proxy:proxy $logdir/$filename
fi

echo "generating data..."
printf "####### `date` #######\n\n"  >>
$logdir/$filename
echo "Detail Byte Hit" >> $logdir/$filename
echo "-----------------------------------------"
>> $logdir/$filename
```

```
echo " Jumlah Byte Tipe"  >> $logdir/$filename
echo "----------------------------------------"
>> $logdir/$filename


for i in $listHit
do
       jumlahByteHit=`cat $logdir/access.log | grep
$i | awk '{print \$5}' | awk '{total+=\$1} END
{printf "%d", total}'`
       printf "%12 %s\n" $jumlahByteHit $i  >>
$logdir/$filename
       TotalByteHit=$((TotalByteHit+jumlahByteHit))
done
if [ $TotalByteHit -eq 0 ]
then
       echo "           -  -" >> $logdir/$filename
fi


echo -e "\nDetail Byte Miss" >> $logdir/$filename
echo "----------------------------------------"
>> $logdir/$filename
echo " Jumlah Byte Tipe" >> $logdir/$filename
echo "----------------------------------------"
>> $logdir/$filename


for i in $listMiss
do
       jumlahByteMiss=`cat $logdir/access.log | grep
$i | awk '{print $5}' | awk '{total+=\$1} END {printf
"%d", total}'`
       printf "%12s %s\n" $jumlahByteMiss $i >>
$logdir/$filename
       TotalByteMiss=$((TotalByteMiss+jumlahByteMiss)
)
done
if [ $TotalByteMiss -eq 0 ]
```

```
then
       echo "            -   -" >> $logdir/$filename
fi


TotalByteReq=$((TotalByteHit+TotalByteMiss))
ByteHit rate=$(echo "scale=3;
$TotalByteHit*100/$TotalByteReq" | bc)


printf "\nTotal Byte Hit \t = %d\n" $TotalByteHit >>
$logdir/$filename
printf "Total Byte Miss\t = %d\n" $TotalByteMiss >>
$logdir/$filename
printf "Total Byte Req \t = %d\n" $TotalByteReq >>
$logdir/$filename


echo -e "\n#################### END
####################\n\n" >> $logdir/$filename


if [ `ls -ltr $csvdir | grep $csvfile | wc -l` == 0 ]
then
       echo "creating CSV file..."
       echo "Tanggal;Jam;Nilai (%)" >
$csvdir/$csvfile
       chown $activeUser:$activeUser $csvdir/$csvfile
fi
echo "generating CSV data..."
printf "%s;%s;%s\n" $getDate $getTime $ByteHit rate>>
$csvdir/$csvfile
```

6.3 Script squidClientCountAIO.sh

```
#!/bin/bash
#created by adikur for final project purpose

logdir="/var/log/squid3"
filename="squidClientAIOStatistic.log"
getTime=`date +%H:%M`
getDate=`date +%d/%m/%y`
activeUser=`who | awk '{print $1}' | uniq -c | awk
'{print $2}'`
csvfile="squidClientAIO.csv"
csvdir="/home/$activeUser/Documents"

if [ `ls -ltr $logdir | grep $filename | wc -l` == 0
]
then
       echo "creating log file..."
       > $logdir/$filename
       chown proxy:proxy $logdir/$filename
fi

echo "generating statistic data..."
printf "#### `date` #####\n\n"  >> $logdir/$filename
echo "-------------------------------------" >>
$logdir/$filename
printf "%11s %10s %11s\n" "Operasi" "Total" "Number"
>> $logdir/$filename
printf "%23s %11s\n" "Request" "Serviced" >>
$logdir/$filename
echo "-------------------------------------" >>
$logdir/$filename

for i in 9 8 7 6 5 4 3 2 1
do
       operasi=`squidclient mgr:squidaio_counts |
```

```
head -23 | tail -$i | head -1 | awk '{print $1}'`
      totalReq=`squidclient mgr:squidaio_counts |
head -23 | tail -$i | head -1 | awk '{print $2}'`
      numbServiced=`squidclient mgr:squidaio_counts
| head -23 | tail -$i | head -1 | awk '{print $3}'`


      if [ $totalReq == "-" ]
      then
            totalReq="0"
      fi
      if [ $numbServiced == "-" ]
      then
            numbServiced="0"
      fi


      printf "%15s %10s %10s\n" $operasi $totalReq
$numbServiced >> $logdir/$filename
done


echo -e "\n################# END
#################\n\n" >> $logdir/$filename


if [ `ls -ltr $csvdir | grep $csvfile | wc -l` == 0
]
then
      echo "creating CSV file..."
      echo "Tanggal;Waktu;Operasi;Total
Request;Number serviced" > $csvdir/$csvfile
chown $activeUser:$activeUser $csvdir/$csvfile
fi
echo "generating CSV data..."


for i in 9 8 7 6 5 4 3 2 1
do
      operasi=`squidclient mgr:squidaio_counts |
```

```
head -23 | tail -$i | head -1 | awk '{print $1}'`
       totalReq=`squidclient mgr:squidaio_counts |
head -23 | tail -$i | head -1 | awk '{print $2}'`
       numberServiced=`squidclient
mgr:squidaio_counts | head -23 | tail -$i | head -1 |
awk '{print $3}'`


       if [ $totalReq == "-" ]; then totalReq="0"; fi
       if [ $numbServiced == "-" ]; then
numbServiced="0"; fi


       printf "%s;%s;%s;%s;%s\n" $getDate $getTime
$operasi $totalReq $numbServiced >> $csvdir/$csvfile
 done
```

6.4 Script squidClientUtilizationStat.sh

```
#!/bin/bash
#created by adikur for final project purpose


activeUser=`who | awk '{print $1}' | uniq -c | awk
'{print $2}'`
logdir="/var/log/squid3"
logfile="squidClientUtilStatistic.log"
csvfile="squidClientUtilization.csv"
csvdir="/home/$activeUser/Documents"


v1=`squidclient -h localhost
cache_object://localhost/counters | grep
client_http.requests | awk '{print $3}'`
v2=`squidclient -h localhost
cache_object://localhost/counters | grep
client_http.hits | awk '{print $3}'`
v3=`squidclient -h localhost
cache_object://localhost/counters | grep
client_http.errors | awk '{print $3}'`
v4=`squidclient -h localhost
cache_object://localhost/counters | grep
client_http.kbytes_in | awk '{print $3}'`
v5=`squidclient -h localhost
cache_object://localhost/counters | grep
client_http.kbytes_out | awk '{print $3}'`
v6=`squidclient -h localhost
cache_object://localhost/counters | grep
client_http.hit_kbytes_out | awk '{print $3}'`
v7=`squidclient -h localhost
cache_object://localhost/counters | grep
server.http.requests | awk '{print $3}'`
v8=`squidclient -h localhost
cache_object://localhost/counters | grep
server.http.errors | awk '{print $3}'`
v9=`squidclient -h localhost
cache_object://localhost/counters | grep
server.http.kbytes_in | awk '{print $3}'`
```

```
v10=`squidclient -h localhost
cache_object://localhost/counters | grep
server.http.kbytes_out | awk '{print $3}'`
v11=`squidclient -h localhost
cache_object://localhost/counters | grep page_faults
| awk '{print $3}'`
v12=`squidclient -h localhost
cache_object://localhost/counters | grep cpu_time |
awk '{print $3}'`
v13=`squidclient -h localhost
cache_object://localhost/counters | grep swap.outs |
awk '{print $3}'`
v14=`squidclient -h localhost
cache_object://localhost/counters | grep swap.ins |
awk '{print $3}'`
v15=`squidclient -h localhost
cache_object://localhost/counters | grep
aborted_requests | awk '{print $3}'`


if [ `ls -ltr $logdir | grep $logfile | wc -l` == 0
]
then
       echo "creating log file..."
       > $logdir/$logfile
       chown proxy:proxy $logdir/$logfile
fi


echo "generating statistic data..."


printf "#### `date` #####\n\n"  >> $logdir/$logfile
printf "%28s %s\n" "client_http.requests =" $v1 >>
$logdir/$logfile
printf "%28s %s\n" "client_http.hits =" $v2 >>
$logdir/$logfile
printf "%28s %s\n" "client_http.errors =" $v3 >>
$logdir/$logfile
printf "%28s %s\n" "client_http.kbytes_in =" $v4 >>
$logdir/$logfile
```

```
printf "%28s %s\n" "client_http.kbytes_out =" $v5 >>
$logdir/$logfile
printf "%28s %s\n" "client_http.hit_kbytes_out ="
$v6 >> $logdir/$logfile
printf "%28s %s\n" "server.http.requests =" $v7 >>
$logdir/$logfile
printf "%28s %s\n" "server.http.errors =" $v8 >>
$logdir/$logfile
printf "%28s %s\n" "server.http.kbytes_in =" $v9 >>
$logdir/$logfile
printf "%28s %s\n" "server.http.kbytes_out =" $v10
>> $logdir/$logfile
printf "%28s %s\n" "page_faults =" $v11 >>
$logdir/$logfile
printf "%28s %s\n" "cpu_time =" $v12 >>
$logdir/$logfile
printf "%28s %s\n" "swap.outs =" $v13 >>
$logdir/$logfile
printf "%28s %s\n" "swap.ins =" $v14 >>
$logdir/$logfile
printf "%28s %s\n" "aborted_requests =" $v15 >>
$logdir/$logfile


echo -e "\n################## END
##################\n\n" >> $logdir/$logfile


if [ `ls -ltr $csvdir | grep $csvfile | wc -l` == 0
]
then
       echo "creating CSV file..."
       echo
"client_http.requests;client_http.hits;client_http.er
rors;client_http.kbytes_in;client_http.kbytes_out;cli
ent_http.hit_kbytes_out;server.http.requests;server.h
ttp.errors;server.http.kbytes_in;server.http.kbytes_o
ut;page_faults;cpu_time;swap.outs;swap.ins;aborted_re
quests" > $csvdir/$csvfile
       chown $activeUser:$activeUser $csvdir/$csvfile
```

```
fi


echo "generating CSV data..."
printf
"%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s\n" $v1
$v2 $v3 $v4 $v5 $v6 $v7 $v8 $v9 $v10 $v11 $v12 $v13
$v14 $v15 >> $csvdir/$csvfile
```