Perancangan Sistem *Proxy Server* Menggunakan Protokol WCCPv2 dengan Konfigurasi Multi *Router*

Fatah Mumtaz Al 'Ala[1)], Rinta Kridalukmana[2)], Eko Didik Widianto[2)]
Program Studi Sistem Komputer, Fakultas Teknik, Universitas Diponegoro
Jalan Prof. Sudharto, Tembalang, Semarang, Indonesia

Numbers of internet users are increasing incredibly fast. Ideally, this increasing numbers of users are also supported by capacity uplift, in this case an increase in bandwidth to maintain the standard services that received by the users. However, increasing the bandwidth not always becomes the first option since it is quite expensive. Implementing proxy server as content/cache engine is the other option available. It will cache the content that user requested and keep it for a while for servicing the other users that will request the same content in the future.

WCCP protocol is used to redirect user's traffic to the proxy server. The standard proxy server configurations are using single router with one or more proxy servers. This thesis is aims to design and implement proxy server system with multi routers configuration. Multi routers configuration is used as a failover mechanism to provide network high availability. It will use HSRP protocol to provide the high availability services.

Tests that conducted after implementation shows the increase in transaction and successful transaction by 296% and 284% also a decrease in response time as well as failed transaction by 18% and 99% consecutively. Failover test shows the percentage of packet loss amounted to 31,3% and 26,3% for clients in VLAN 10 and VLAN 20 consecutively. The average time required for clients to reconnect to the internet after router failure is 7 seconds for clients in VLAN 10 and 6 seconds for client in VLAN 20.

Keywords : internet, proxy server, WCCP, HSRP, failover

I PENDAHULUAN

Perkembangan teknologi jaringan komputer yang sangat cepat saat ini mengakibatkan terjadinya peningkatan pengguna teknologi tersebut secara signifikan. Hingga kebutuhan untuk mengakses jaringan komputer global (Internet) sudah menjadi layaknya kebutuhan sehari – hari untuk berbagai macam tujuan. Baik sekadar melihat *feed* media sosial, *video streaming,* maupun hanya untuk menjelajah internet. Fenomena peningkatan jumlah pengguna ini harus disiasati dengan penggunaan berbagai macam teknologi yang tersedia untuk tetap mempertahankan tingkat kualitas layanan internet yang diterima oleh pelanggan. Salah satu solusinya adalah menambahkan jumlah *bandwidth* yang dimiliki, dan penggunaan *content/cache engine*. Opsi penambahan *bandwidth* tidak menjadi prioritas utama karena memang cukup mahal dan harus dibayarkan secara berkala karena menggunakan sistem sewa. Penggunaan *content/cache engine* jauh lebih murah dan efisien dalam meningkatkan kualitas layanan, Blue Coat System menyediakan Blue Coat CacheFlow untuk jenis solusi yang berbayar, di sisi lain terdapat Squid/LUSCA yang merupakan solusi berbasis *open source*. Meskipun berbasis *open source,* Squid/LUSCA dapat diandalkan sebagai salah satu solusi.

Konsep penggunaan *content/cache engine* diterapkan dengan menempatkan *content/cache engine* sebagai *proxy server* antara jaringan lokal dengan internet. *Content/cache engine* akan menyimpan konten-konten dari internet yang pernah diakses oleh pengguna, ketika ada pengguna lain yang meminta konten yang sama, maka permintaan tersebut akan dipenuhi secara lokal oleh *content/cache engine*. Oleh karena itu, hal tersebut dapat mengurangi *transmission costs* dan *download time.* Pada sisi pengguna, mereka tidak menyadari bahwa sebagian besar permintaan mereka akan dipenuhi secara lokal oleh *content/cache engine*, karena proses intersepsi dan pengalihan lalu lintas pengguna ke *content/cache engine* dilakukan secara transparan oleh protokol WCCP (*Web Cache Communication Protocol*).

Konfigurasi *content/cache engine* dan *router* yang biasa diterapkan adalah dengan menggunakan satu atau lebih *cache engine* dan satu unit *router*. Fitur dari protokol WCCPv2 (*Web Cache Communication Protocol version 2*) mendukung penggunaan lebih dari satu unit *router* untuk setiap satu unit *cache engine.* Dalam dokumentasi yang disajikan oleh Cisco selaku pembuat *router* menyatakan bahwa dengan menggunakan WCCPv2, satu unit *cache engine* dapat berkomunikasi dengan jumlah maksimum sebanyak 32 unit *router*. Konsep penggunaan lebih dari 1 unit *router* dapat meningkatkan kehandalan jaringan dengan menyediakan mekanisme *failover* melalui desain yang redundan untuk memastikan tidak ada *single link failure*.[4]

II METODOLOGI PENELITIAN

Metodologi penelitian yang digunakan pada perancangan sistem *proxy server* menggunakan protokol WCCPv2 dengan konfigursi multi *router* memiliki tahapan sebagai berikut, definisi sistem yang merupakan tahap mendefinisikan sistem yang akan dibuat, meliputi penjabaran sistem, identifikasi kebutuhan sistem, tujuan dan manfaat sistem, cara kerja dan topologi yang digunakan.

Proses spesifikasi kebutuhan akan menjabarkan tentang awal perancangan sistem dengan menentukan spesifikasi kebutuhan yang sesuai definisi sistem. Spesifikasi kebutuhan terdiri atas spesifikasi perangkat keras dan perangkat lunak. Kegiatan ini menentukan arsitektur sistem secara keseluruhan. Pada tahap ini telah ditentukan bahwa sistem ini akan dirancang dalam lingkungan virtual menggunakan aplikasi GNS3 sebagai emulator *router*. Sistem operasi yang digunakan

adalah Ubuntu Server dan Desktop 14.04 yang dijalankan secara virtual dengan aplikasi VMware Fusion. *Proxy server* yang dirancang menggunakan aplikasi Squid 3.3.8. Protokol WCCP digunakan untuk mengalihkan lalu lintas *web* dari klien menuju *proxy server*. Protokol HSRP akan digunakan sebagai mekanisme *failover* untuk menyediakan ketersediaan tertinggi pada jaringan. Perangkat keras yang digunakan adalah 1 unit laptop MacBook Pro dengan prosesor Intel Core 2 Duo P8700 2,53 GHz, *memory* 4 GB dan *hard disk* 500 GB.

Konfigurasi sistem, pada tahap ini spesifikasi kebutuhan yang telah ditentukan akan dirancang sesuai topologi/desain jaringan dan direalisasikan sebagai serangkaian sistem atau unit sistem yang memungkinkan untuk menjalankan tujuan sistem dan cara kerja sistem. Tahapan konfigurasi dilakukan melakukan konfigurasi yang sesuai pada semua komponen sistem, yaitu *router, proxy server* dan komputer *host*. Komputer *host* harus dikonfigurasi karena akan bertindak sebagai *router* yang menghubungkan semua perangkat dalam lingkungan virtual GNS3 ke jaringan di dunia nyata.

Pengujian sistem, yaitu proses pengujian dengan berbagai parameter tertentu untuk memastikan sistem dapat berjalan dengan baik dan dapat memenuhi fungsi dan tujuannya. Pada tahap ini, pengujian yang dilakukan mencakupi pengujian *proxy server* dan pengujian *failover*.

Analisis sistem, yaitu proses analisa data yang didapat dari pengujian kemudian dianalisis untuk mendapatkan hasil yang diinginkan. Analisis *proxy server* yang dilakukan untuk membuktikan apakah *proxy server* telah dapat bekerja dengan baik atau tidak. Hal tersebut dilakukan dengan melihat perbandingan data yang ada pada kondisi beberapa parameter ketika *proxy server* tidak aktif dan ketika *proxy server* telah aktif. Analisis performansi Squid juga dilakukan untuk menunjukkan bagaimana perilaku Squid selama pengujikan berlangsung. Analisis mekanisme *failover* dilakukan dengan melihat hasil pengujian yang akan menunjukkan waktu berapa lama *standby router* mengambil alih peran *active router* ketika *active router* mengalami kegagalan dan waktu berapa lama klien dapat terhubung kembali ke internet serta *packet loss* yang dialami oleh klien ketika terjadi kegagalan *active router*.

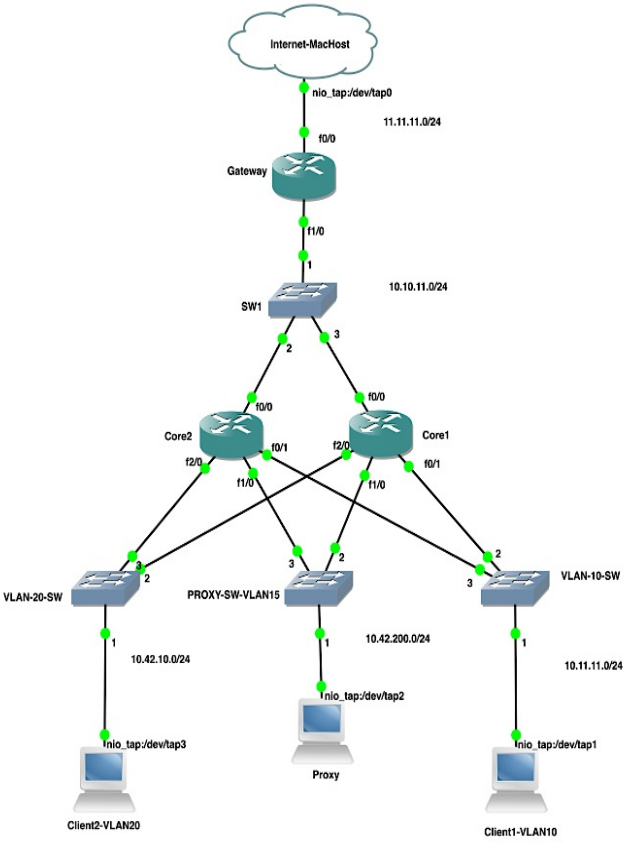
III PERANCANGAN SISTEM

Definisi sistem akan menjabarkan awal sistem dibuat, tujuan, topologi jaringan dan cara kerja sistem yang akan digunakan untuk menentukan spesifikasi kebutuhan sistem ke tahapan selanjutnya, beberapa tahapan untuk bisa mendefinisikan sistem yang telah dibuat adalah sebagai berikut:

A. Definisi Sistem

Pada tahap awal ini, sistem dibuat dalam lingkungan virtual menggunakan aplikasi GNS3 dan aplikasi VMWare Fusion untuk mensimulasikan sistem operasi bagi *proxy server* dan klien. GNS3 merupakan aplikasi emulasi jaringan lintas – platform yang dikembangkan oleh Christophe Fillot, Jeremy Grossman, Julien Duponchelle. Aplikasi ini memudahkan penggunanya melakukan percobaan dalam bentuk jaringan *virtual* di PC, (termasuk namun tidak terbatas pada) Cisco IOS, Juniper, Mikrotik, Arista dan Vyatta Network.[5]

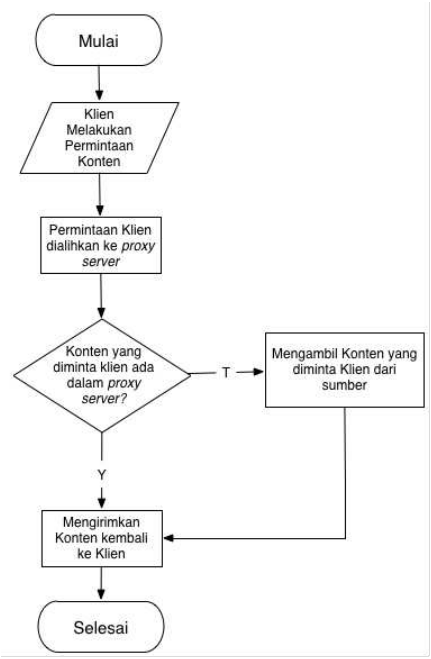
Pilihan untuk melakukan penelitian pada lingkungan virtual karena tidak adanya perangkat *router* yang mendukung, dibutuhkan paling tidak 4 buah *port Ethernet* pada *router* untuk merealisasikan penelitian ini ke dalam perangkat fisik. Sistem ini didesain terlebih dahulu menggunakan topologi jaringan. Topologi jaringan merupakan hal yang menjelaskan hubungan geometris antara unsur-unsur dasar penyusun jaringan, yaitu *node*, *link*, dan *station*. Topologi jaringan sistem ini dibuat dengan aplikasi GNS3 yang disusun sesuai kebutuhan sistem yang akan dibuat.



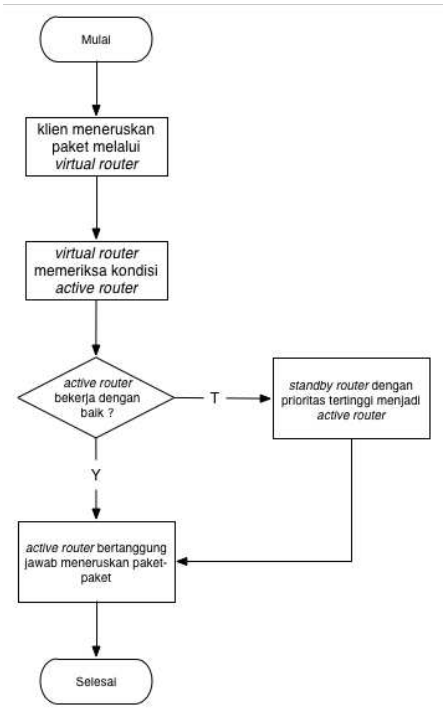
Gambar 1 Topologi Jaringan

Pada Gambar 1 di atas, terdapat dua buah sistem utama yang bekerja di dalamnya, yaitu sistem *proxy server* dengan menggunakan protokol WCCP yang digunakan untuk mengalihkan permintaan konten klien menuju *proxy server,* protokol WCCP menggunakan *tunnel* GRE (*General Routing Encapsulation*) diantara *router* dan *proxy server* yang menjalankan Squid. Permintaan yang dialihkan akan dibungkus (enkapsulasi) dengan GRE dan dikirim ke *proxy server* melalui *tunnel* GRE. Proses dekapsulasi paket GRE dan pengalihan ke Squid/LUSCA dilakukan oleh mesin *host* menggunakan IPTables. Kemudian Squid/LUSCA akan mengambil konten yang diminta dari *server* asli ataupun dari *cache* dan mengembalikan konten ke *router. Router* akan mengembalikan tanggapan dari *proxy server* tersebut ke pengguna.[1]

Mekanisme *failover* dirancang dengan protokol HSRP yang digunakan untuk menyediakan ketersediaan tinggi pada jaringan.[3] Cara kerja sistem *proxy server* dan mekanisme *failover* dari desain yang dibuat ditunjukkan oleh Gambar 2 dan Gambar 3.



Gambar 2 Diagram alir cara kerja sistem *proxy server*



Gambar 3 Diagram alir cara kerja mekanisme *failover*

B. Spesifikasi Kebutuhan

Penelitian ini menggunakan perangkat keras berupa 1 buah perangkat laptop MacBook Pro dengan spesifikasi prosesor Intel Core 2 Duo P8700 2,53 GHz, *hard disk* 500GB dan *memory* sebesar 4GB. Perangkat lunak yang digunakan selama penelitian ini adalah sebagai berikut :

1. VMWare Fusion

Sistem operasi *proxy server* dan klien dijalankan secara virtual menggunakan aplikasi VMware Fusion 6.0.6 yang dipasang pada komputer *host* MacBook.

2. Sistem Operasi Ubuntu Server 14.04.3

Ubuntu merupakan salah satu distribusi *Linux* yang berbasiskan Debian dan didistribusikan sebagai perangkat lunak bebas.Ubuntu versi ini dirancang untuk kepentingan penggunaan *server*. Sistem operasi ini digunakan sebagai *proxy server* untuk menjalankan aplikasi Squid.

3. Sistem Operasi Ubuntu Desktop 14.04

Sistem operasi ini merupakan versi *desktop* dari Ubuntu Server yang sudah menyediakan tampilan GUI (*Graphical User Interface*) untuk kemudahan pengguna.

4. Squid/LUSCA

Squid adalah sebuah *daemon* yang digunakan sebagai *server proxy* dan web *cache*. Squid memiliki beberapa kegunaan, mulai dari mempercepat *server web* dengan melakukan caching permintaan yang berulang-ulang, *caching Domain Name Server* (DNS), *caching* situs *web*, dan *caching* pencarian komputer di dalam jaringan untuk sekelompok komputer yang berada pada jaringan yang sama, dan bisa juga untuk membantu keamanan dengan cara melakukan penyaringan (*filtering*) trafik.[6] Sedangkan LUSCA adalah cabang pengembangan dari Squid versi 2. Proyek dari LUSCA digunakan untuk memperbaiki kekurangan dari Squid versi 2 namun tetap mempertahankan fungsi dan stabilitas dari Squid versi 2.[2]. Squid yang digunakan pada *proxy server* adalah versi 3.3.8.

III PENGUJIAN SISTEM

Setelah konfigurasi sistem selesai diimplementasikan, pengujian dilakukan untuk memastikan sistem yang dibuat dapat berfungsi dengan baik. Pengujian yang dilakukan antara lain adalah sebagai berikut , yaitu pengujian *proxy server* dan pengujian mekanisme *failover*.

Pengujian *proxy server* menggunakan dua kondisi, yaitu pengujian ketika *proxy server* dalam keadaan mati dan pengujian dalam ketika *proxy server* dalam keadaan hidup. Skenario pengujian yang digunakan adalah klien dari VLAN 10 dan VLAN 20 akan melakukan simulasi permintaan HTTP sebanyak masing-masing 100 klien selama 30 menit ke 4 *website* yang berbeda. Selain itu, pengujian *proxy server* juga mencakup pengujian kemampuan dari *proxy server* untuk melakukan *caching* terhadap paket – paket *update* sistem operasi.

Pengujian mekanisme *failover* dilakukan dengan skenario dimana *active router* dari VLAN klien akan dimatikan dan parameter yang diuji ialah waktu pengambil alihan peran *active router* oleh *standby router*, waktu klien dapat kembali terhubung ke internet dan presentase *packet loss*.

A. Pengujian *proxy server*

Pengujian ini dilakukan dengan menggunakan aplikasi Siege yang dipasang pada komputer klien VLAN 10 dan VLAN 20 yang dilakukan pada dua kondisi, yaitu kondisi saat *proxy server* dalam keadaan tidak aktif dan pada saat *proxy server* dalam keadaan aktif. Selama pengujian, aplikasi Siege akan mensimulasikan permintaan HTPP setara dengan 100 orang ke 4 *website* yang berbeda secara acak. Setelah proses simulasi selesai, aplikasi Siege akan menunjukkan hasil dengan beberapa parameter. Pengujian *proxy server* juga mencakup pengujian kemampuan dari *proxy server* untuk melakukan *caching* terhadap paket – paket *update* sistem operasi. Parameter yang dihitung adalah waktu dan kecepatan unduh dari paket *update* sistem operasi ketika belum disimpan dalam *proxy server* dibandingkan dengan ketika paket *update* sudah tersimpan dalam *proxy server*. Presentase perubahan nilai setiap parameter dapat menunjukkan apakah *proxy server* sudah dapat bekerja dengan baik ataupun tidak. Tabel 1 di

bawah ini menunjukkan presentase perubahan nilai rerata dari parameter yang ditunjukkan oleh aplikasi Siege setelah pengujian selesai.

Tabel 1 Tabel pengujian *proxy server*

Parameter	Sebelum Proxy Server Aktif	Setelah Proxy Server Aktif	Presentase Perubahan
	Nilai Rerata	Nilai Rerata	
Transaction (hits)*	4.529,5	17.922	296%
Availability (%)*	82.67	99.96	21%
Elapsed Time (detik)	1.799,54	1.799,94	0%
Data Transferred (MB)*	55,97	200,24	258%
Response Time (detik)**	13,32	10,91	-18%
Transaction Rate (trans/detik)*	2,52	9,96	296%
Throughput (MB/detik)*	0,03	0,11	267%
*Concurrency**	37,9	90,92	140%
*Successful Transaction**	4.686,5	17.973,5	284%
*Failed Transaction***	765,5	4,5	-99%
Longest Transaction (detik)**	266,3	226,25	-15%
Shortest Transaction (detik)**	0.69	1.6	134%

* = semakin besar semakin baik
** = semakin kecil semakin baik

Pada Tabel 1 di atas, dapat diambil beberapa parameter yang menunjukkan bahwa *proxy server* telah dapat bekerja dengan baik, antara lain parameter *transaction, response time* dan *failed transaction*. Parameter *transction* mengalami kenaikan hingga 296% dari 4.529,2 *hit* menjadi 17.922 hit saat *proxy server* aktif. Parameter *response time* dan *failed transaction* secara berurutan mengalami penurunan 18% dan 99% pada saat *proxy server* aktif.

Berikut ini merupakan pengujian *caching* terhadap paket – paket yang digunakan dalam melakukan pembaruan sistem operasi. Pengujian berlangsung dengan cara memberikan perintah `apt-get upgrade` pada aplikasi *terminal*. `Apt-get upgrade` berfungsi untuk melakukan pembaruan paket – paket pada aplikasi yang telah terpasang pada sistem operasi. Bagaimana Squid akan menangani umur paket dapat dimanipulasi menggunakan perintah `refresh_pattern` pada konfigurasi Squid. Pada pengujian kali ini umur paket pembaruan dianggap dalam kondisi FRESH selama 30 menit saja, jika lebih dari itu paket tersebut akan dianggap STALE (kedaluwarsa). Ukuran dari paket pembaruan yang akan diunduh selama pengujian adalah 64,4 MB.

Pengujian dilakukan sebanyak 4 kali, yaitu Pengujian 1 hingga Pengujian 4. Pengujian 1 pada klien VLAN 10 saat *proxy server* dalam kondisi kosong, artinya tidak ada paket pembaruan yang tersimpan dalam *cache*. Pengujian 2 dilakukan pada klien VLAN 20, dengan kondisi *proxy server* telah menyimpan paket pembaruan yang telah diunduh pada Pengujian 1. Pengujian 3 dilakukan pada klien VLAN 20, 30 menit setelah Pengujian 2 selesai dengan asumsi paket pembaruan yang telah disimpan oleh *proxy server* sudah dalam keadaan STALE (kedaluwarsa). Pengujian 4 dilakukan setelah pada klien VLAN 10 dengan keadaan paket pembaruan dalam *proxy server* telah berada dalam kondisi FRESH kembali. Tabel 2 dan 3 dibawah ini menunjukkan hasil Pengujian 1 hingga Pengujian 4.

Tabel 2 Tabel pengujian pembaruan sistem operasi klien

Parameter	Pengujian 1 (P1)	Pengujian 2 (P2)	
	VLAN 10	VLAN 20	Presentase Perubahan (P2 : P1)
Waktu Unduh (detik)**	293	42	-86%
Kecepatan (kB/detik)*	219	1528	598%

Tabel 3 Tabel pengujian pembaruan sistem operasi klien - 2

Parameter	Pengujian 3 (P3)		Pengujian 4 (P4)	
	VLAN 20	Presentase Perubahan (P3 : P2)	VLAN 10	Presentase Perubahan (P4 : P3)
Waktu Unduh (detik)**	55	31%	40	-27%
Kecepatan (kB/detik)*	1154	-24%	1571	36%

*= semakin besar semakin baik
**= semakin kecil semakin baik

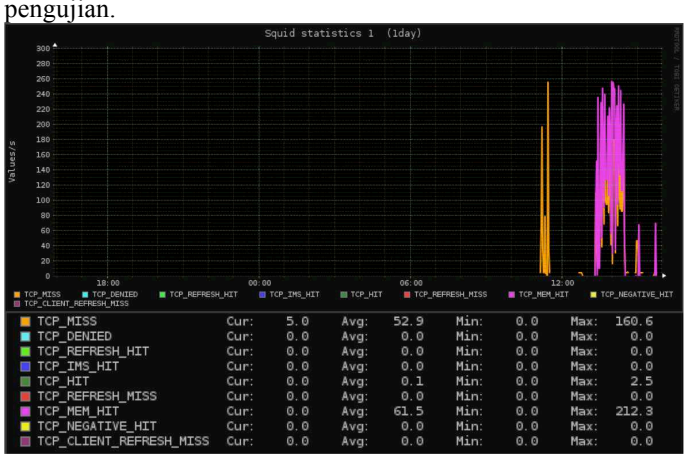
Tabel 2 dan 3 menunjukkan bagaimana perubahan waktu dan kecepatan unduh paket pembaruan selama pengujian. Pengujian 1 menunjukkan kecepatan unduh sebesar 219 kB/detik dengan waktu selama 293 detik. Pengujian 2 dilakukan setelah Pengujian 1 dilakukan dengan kondisi paket pembaruan telah tersimpan dalam *proxy server*. Pengujian 2 menunjukkan kecepatan unduh sebesar 1.528 kB/detik dengan waktu selama 42 detik. Presentase perubahan waktu dan kecepatan dibandingkan dengan Pengujian 1 adalah sebesar -86% dan 598%. Perubahan yang signifikan tersebut disebabkan oleh *proxy server* yang telah menyimpan paket pembaruan yang sebelumnya telah diunduh pada Pengujian 1, sehingga permintaan klien dapat dipenuhi secara lokal tanpa harus langsung meminta ke *server* tujuan.

Pengujian 3 dilakukan 30 menit setelah Pengujian 2 dengan keadaan paket pembaruan yang tadi telah diunduh pada Pengujian 1 sudah dalam keadaan STALE (kedaluwarsa). Pengujian 3 menunjukkan kecepatan unduh sebesar 1.154 kB/detik dengan waktu selama 55 detik. Presentase perubahan waktu unduh terhadap Pengujian 2 mengalami kenaikan sebesar 31% dan kecepatan unduh mengalami penurunan sebesar 24%. Kenaikan waktu unduh dan penurunan kecepatan disebabkan ketika klien melakukan permintaan terhadap paket

pembaruan, paket tersebut sudah dalam keadaan STALE, sehingga Squid terlebih dahulu akan menghubungi *server* asal paket untuk melakukan validasi terhadap paket tersebut. Jika paket tersebut masih dinyatakan valid oleh *server* asal, Squid akan mengirimkan paket yang telah tersimpan dalam *cache* ke klien. Jika paket dinyatakan tidak valid, Squid akan mengunduh paket terbaru yang valid dan kemudian baru dikirimkan menuju klien. Pengujian 4 dilakukan setelah paket pembaruan yang tadi telah STALE kembali diperbarui sehingga dalam kondisi FRESH. Pengujian 4 menunjukkan kecepatan unduh sebesar 1.571 kB/detik dengan waktu selama 40 detik. Presentase perubahan waktu dan kecepatan unduh Pengujian 4 terhadap Pengujian 3 adalah -27% dan 36%. Penurunan waktu unduh dan kenaikan kecepatan unduh paket disebabkan oleh permintaan klien yang sudah dapat dipenuhi secara lokal oleh *proxy server*.

B. Performansi Squid selama pengujian *proxy server*

Selama pengujian berlangsung, kinerja dari Squid dapat dipantaun melalui aplikasi monitorix. Aplikasi ini akan menerjemahkan aktifitas dari Squid melalui access.log Squid. Semua aktifitas akses Squid tercatat pada log tersebut. Gambar 5 di bawah ini menunjukkan grafik kinerja Squid selama pengujian.



Gambar 4 Grafik kinerja Squid selama pengujian

Gambar 4 di atas menunjukkan informasi mengenai respon Squid selama pengujian berlangsung. TCP_HIT merupakan respon ketika konten yang diminta oleh klien sudah berada dalam *cache* dan dalam kondisi *fresh*. TCP_IMS_HIT merupakan kondisi ketika klien mengeluarkan permintaan IMS (*If-Modified-Since*) terhadap objek/konten yang telah ada dalam *cache* dengan kondisi *fresh*. TCP_MEM_HIT merupakan kondisi ketika konten yang diminta oleh klien masih tersimpan dalam *memory,* dalam hal ini merupakan RAM dari *proxy server*. TCP_MISS merupakan kondisi dimana konten yang diminta oleh klien tidak ada dalam *cache* dan Squid akan langsung mengambil konten yang diminta tersebut dari sumbernya. Informasi mengenai jumlah dari respon Squid di atas dapat dilihat pada Tabel 4 dibawah ini.

Tabel 4 Tabel Performansi Squid

No	Respon Squid	Jumlah rata-rata (*values/s*)	Jumlah Maksimum (*values/s*)
1	TCP_MISS	52,9	160,6
TOTAL MISS		**52,9**	**160,6**
2	TCP_HIT	0,1	2,5
3	TCP_MEM_HIT	61,5	212,3
TOTAL HIT		**61,6**	**214,8**

Tabel 4 di atas menunjukkan informasi performa Squid ketika pengujian. Berdasarkan tabel tersebut diketahui bahwa respon rerata TCP_MISS selama pengujian adalah 52,9 *values*/detik dengan jumlah maksimum 160,6 *values*/detik. Respon rerata TCP_HIT selama pengujian adalah 0,1 *values*/detik dengan jumlah maksimum 2,5 *values*/detik. Rerata respon TCP_MEM_HIT selama pengujian adalah 61,5 *values*/detik dengan jumlah maksimum 212,3 *values*/detik. Jumlah total permintaan dengan respon MISS adalah 52,9 *vales*/detik dan total respon permintaan HIT adalah 61,6 *values*/detik.

Berdasarkan keterangan di atas diketahui bahwa selama pengujian berlangsung, Squid telah dapat melayani permintaan klien dengan cukup baik. Hal ini ditandai dengan jumlah permintaan dengan respon HIT berada di atas permintaan dengan respon MISS yang bernilai 61,6 *values*/detik berbanding 52,9 *values*/detik.

C. Pengujian Mekanisme *failover*

Pengujian *failover* dilakukan dengan menonaktifkan *interface* dari *active router* VLAN. Parameter yang digunakan adalah selang waktu pengambil alihan peran *active router* oleh *standby router* dan presentase *packet loss* pada klien. Pengujian ini menggunakan perintah `ping` yang ditujukan ke situs www.google.com yang dilakukan sebanyak 3 kali untuk masing – masing VLAN klien. Nilai yang didapat dari ketiga pengujian akan dirata-rata untuk mendapat hasil pengujian.

Tabel 5 Tabel pengujian *failover router* VLAN 10

Parameter/*Network*	VLAN 10			
	Pengujian			
	1	2	3	Rerata
Waktu Non Aktif *Active Router* (hh:mm:ss)	02:45:25.381	02:49:25.198	02:57:32.991	-
Waktu *Standby Router* menjadi *Active Router* (hh:mm:ss)	02:45:25.422	02:49:25.258	02:57:33.060	-
Selisih Waktu *Takeover* (detik)	0,041	0,06	0,069	0,057

Tabel 6 Tabel pengujian *failover* klien VLAN 10

VLAN 10				
Pengujian	Sekuens Koneksi Terputus	Sekuens Koneksi Tersambung Kembali	Selisih Waktu (detik)	Packet Loss (%)
1	13	22	9	34
2	9	15	6	30
3	11	17	6	30
Total			21	-
Rerata			7	31.3

Tabel 5 dan Tabel 6 di atas merupakan hasil pengujian *failover* pada klien VLAN 10. Berdasarkan Tabel 4 di atas, menunjukkan selisih waktu pengambil alihan *active router* oleh *standby router* dari VLAN 10 dalam tiga pengujian secara berurutan adalah 0,041 detik, 0,06 detik dan 0,069 detik dengan rerata 0,057 detik. Berdasarkan informasi pada Tabel 5 di atas diketahui bahwa pada pengujian 1 hingga pengujian 3 selisih waktu klien untuk dapat tersambung kembali ke internet secara berurutan adalah 9 detik, 6 detik dan 6 detik dengan rerata 7,1 detik. Presentase *packet loss* selama pengujian pertama hingga ketiga secara berurutan adalah 34%, 30% dan 30% dengan rerata 31,3%.

Tabel 7 Tabel pengujian *failover router* VLAN 20

Parameter/*Network*	VLAN 20			
	Pengujian			
	1	2	3	Rerata
Waktu Non Aktif *Active Router* (hh:mm:ss)	01:26:56.051	01:29:07.559	01:35:44.182	-
Waktu *Standby Router* menjadi *Active Router* (hh:mm:ss)	01:26:56.077	01:29:07.661	01:35:44.236	-
Selisih Waktu *Takeover* (detik)	0,026	0,102	0,054	0,061

Tabel 8 Tabel pengujian *failover* klien VLAN 20

VLAN 20				
Pengujian	Sekuens Koneksi Terputus	Sekuens Koneksi Tersambung Kembali	Selisih Waktu (detik)	*Packet Loss* (%)
1	4	10	6	27
2	8	14	6	25
3	5	11	6	27
Total			18	-
Rerata			6	26.3

Tabel 7 dan Tabel 8 di atas merupakan hasil pengujian *failover* pada klien VLAN 20. Berdasarkan Tabel 6 di atas, menunjukkan selisih waktu pengambil alihan *active router* oleh *standby router* dari VLAN 10 dalam tiga pengujian secara berurutan adalah 0,026 detik, 0,102 detik dan 0,054 detik dengan rerata 0,061 detik. Berdasarkan informasi pada Tabel 7 di atas diketahui bahwa pada pengujian 1 hingga pengujian 3 selisih waktu klien untuk dapat tersambung kembali ke internet secara berurutan adalah 6 detik dengan rerata 6 detik. Presentase *packet loss* selama pengujian pertama hingga ketiga secara berurutan adalah 27%, 25% dan 27% dengan rerata 26,3%.

Ketika *active router* dari masing – masing VLAN sudah dapat kembali bekerja secara normal, kondisi (*state*) dari *active router* tersebut akan berubah menjadi dari *Init* menjadi *Standby* yang kemudian akan berubah kembali menjadi *Active* dengan waktu *delay* sebanyak minimum 30 detik untuk memberikan waktu pada *router* dalam mengumpulkan tabel *routing*-nya.

IV PENUTUP

A. Kesimpulan

Selama pengembangan sistem ini terdapat beberapa hal yang bisa disimpulkan. Kesimpulan yang didapatkan antara lain :

1. Desain dan implementasi *proxy server* menggunakan protokol WCCP (*Web Cache Communication Protocol*) dengan konfigurasi multi *router* telah berhasil dibuat.
2. *Proxy server* yang dibuat menggunakan aplikasi Squid. Pengujian *proxy server* yang dilakukan dalam dua kondisi, kondisi pertama pengujian yang dilakukan ketika Squid berada dalam keadaan mati dan pengujian kedua dilakukan ketika Squid berada dalam keadaan hidup. Perbandingan nilai rerata beberapa parameter selama pengujian menunjukkan bahwa Squid sudah dapat bekerja dengan baik. Parameter *transaction* dan *successful transaction*, yang secara berurutan mengalami kenaikan sebesar 296% dan 284% serta *response time* dan *failed transaction* yang mengalami penurunan sebesar 18% dan 99%.
3. Berdasarkan pengamatan terhadap performansi Squid selama pengujian, didapatkan rerata respon TCP_MISS selama pengujian adalah 52,9 *values*/detik dengan jumlah maksimum 160,6 *values*/detik. Rerata respon TCP_HIT selama pengujian adalah 0,1 *values*/detik dengan jumlah maksimum 2,5 *values*/detik. Rerata respon TCP_MEM_HIT selama pengujian adalah 61,5 *values*/detik dengan jumlah maksimum 212,3 *values*/detik. Berdasarkan keterangan di atas diketahui bahwa selama pengujian berlangsung, Squid telah dapat melayani permintaan klien dengan cukup baik. Hal ini ditandai dengan jumlah permintaan dengan respon HIT berada di atas permintaan dengan respon MISS yang bernilai 61,6 *values*/detik berbanding 52,9 *values*/detik.
4. Pengujian *proxy server* juga mencakup pengujian kemampuan Squid untuk melakukan *caching* terhadap paket – paket pembaruan sistem operasi. Pengujian 1 menunjukkan kecepatan unduh sebesar 219 kB/detik dengan waktu selama 293 detik. Pengujian 2 menunjukkan kecepatan unduh sebesar 1.528 kB/detik dengan waktu selama 42 detik. Presentase perubahan waktu dan kecepatan dibandingkan dengan Pengujian 1 adalah sebesar -86% dan 598%. Perubahan yang signifikan tersebut disebabkan oleh *proxy server* yang telah menyimpan paket pembaruan yang sebelumnya telah diunduh pada Pengujian 1, sehingga permintaan klien dapat dipenuhi secara lokal tanpa harus langsung meminta ke *server* tujuan. Pengujian 3 menunjukkan kecepatan unduh sebesar 1.154 kB/detik dengan waktu selama 55 detik. Presentase perubahan waktu unduh terhadap Pengujian 2 mengalami kenaikan sebesar 31% dan kecepatan unduh mengalami penurunan sebesar 24%. Kenaikan waktu unduh dan penurunan kecepatan disebabkan ketika klien melakukan permintaan terhadap paket pembaruan, paket tersebut sudah dalam keadaan STALE, sehingga Squid terlebih dahulu akan menghubungi *server* asal paket untuk melakukan validasi terhadap paket tersebut. Pengujian 4 menunjukkan kecepatan unduh sebesar 1.571 kB/detik dengan waktu selama 40 detik. Presentase perubahan waktu dan kecepatan unduh Pengujian 4 terhadap Pengujian 3 adalah -27% dan 36%. Penurunan waktu unduh dan kenaikan

kecepatan unduh paket disebabkan oleh permintaan klien yang sudah dapat dipenuhi secara lokal oleh *proxy server*.

5. Pengujian *failover* pada *active router* VLAN klien menunjukkan bahwa masing – masing s*tandby router* dari setiap klien dapat mengambil alih peran *active router* dalam rerata waktu kurang dari 1 detik. *Standby router* VLAN 10 membutuhkan rerata waktu sebesar 0,057 detik dan VLAN 20 sebesar 0,061 detik.
6. Pada sisi klien, ketika pengujian *failover* berlangsung dibutuhkan waktu dengan rerata sebesar 7 detik untuk klien VLAN 10 dan 6 detik untuk VLAN 20 agar dapat kembali terhubung ke internet. Rerata jumlah *packet loss* yang dialami oleh setiap klien secara berurutan untuk VLAN 10 dan VLAN 20 adalah 31,3% dan 26,3%.

B. Saran

1. Implementasi yang dilakukan pada penelitian ini baru sebatas di lingkungan *virtual*. Hasil yang ditunjukkan oleh pengujian sangat bergantung pada kemampuan komputer *host* yang dipakai. Dibutuhkan implementasi lebih lanjut pada perangkat fisik aslinya untuk mendapatkan hasil yang lebih baik. Implementasi pada perangkat fisik membutuhkan paling tidak perangkat *router* yang dilengkapi minimal 4 buah *port Ethernet*
2. Penelitian lanjutan dengan menambahkan jumlah *proxy server* yang digunakan untuk meringankan beban kerja dari *proxy server* itu sendiri.
3. Penelitian lanjutan dengan menambahkan fitur untuk menambah dukungan pada protokol HTTPS.

DAFTAR PUSTAKA

[1] Andriyono, “Impact Proxy Server Terhadap Jaringan Internet,” no. 3, pp. 1–7, 2007.

[2] Chadd , Adrian, “Lusca Web Proxy Cache,” 2010. [Online]. Available: https://sites.google.com/site/luscaproxy/Home. [Accessed: 29-Nov-2015].

[3] Cisco Systems Inc., *Cisco IOS IP Configuration Guide*. San Jose, CA: Cisco Systems, Inc, 2006.

[4] Lammle, Todd, *CCNA - Cisco Certified Network Associate*, Second. Alameda, CA: SYBEX, Inc, 2012.

[5] Neumann, Jason C., *The Book of GNS3*, First. San Fransisco, CA: No Starch Press Inc., 2015.

[6] Saini, Kulbir, *Squid Proxy Server 3.1 Beginner’s Guide*. Birmingham, UK: Packt Publishing, 2011.