DESAIN DAN IMPLEMENTASI JARINGAN PADA LAPIS *CORE ROUTER* (REDUNDANSI) STUDI KASUS UNIVERSITAS DIPONEGORO

Cindy Sahera[*)], Sudjadi, and Adian Fatchur Rochim

Jurusan Teknik Elektro, Universitas Diponegoro
Jl. Prof. Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia

[*)]*E-mail: dysaimnida@gmail.com*

Abstrak

Pengguna internet yang bertambah tiap tahunnya dapat meningkatkan aktivitas lalu lintas jaringan. UNDIP sudah mengimplementasikan penyeimbangan beban dengan empat server proxy dan WCCP di lapis core. Namun, beban lalu lintas yang semakin meningkat sudah mulai mengganggu kinerja dari lapis core itu sendiri dan perangkat-perangkat tersebut masih berada di satu lokasi sehingga apabila terjadi listrik padam pada gedung tersebut, layanan internet mengalami gangguan juga. Oleh dasar ini dilakukan pembuatan jalur redundansi dengan meletakkan perangkat tambahan di lokasi yang berbeda. Tugas akhir ini melakukan desain sistem, implementasi sistem, dan pengujian sistem. Desain sistem menggunakan simulator Packet Tracer yang mencakup penggunaan IP. Implementasi menggunakan router Cisco sebagai Gateway-3 yang merupakan jalur redundansi dan dua buah server yang difungsikan sebagai server proxy. Bila salah satu server mengalami gagal fungsi maka server lainnya tetap melayani pengguna. Pengujian tugas akhir ini adalah membandingkan throughput sebelum dan setalah implementasi serta pengujian jalur redundansi server pada Gateway-3. Hasil yang diperoleh setelah menunjukkan bahwa beban throughput inbound berkurang hingga 45,41% dan beban throughput outbound berkurang hingga 48,76%. Pembagian beban ke kedua server proxy di Gateway-3 dapat dilakukan dengan seimbang. Jika salah satu server proxy mati maka server proxy lainnya menangani seluruh permintaan yang masuk ke Gateway-3.

*Kata kunci**:** internet, trafik, server proxy, throughput*

Abstract

Internet user are increasing each year may increase network traffic activity. UNDIP has implemented load-balancing with four proxy servers and WCCP in core layer. However, increasing traffic seems to interfere the performance of core layer itself and the devices are still in the same building. If there is a blackout in the building then internet service is also impaired. Therefore, this final task is making redundancy path with additional device in different building. Final task performs design, implementation, and testing. System design uses Packet Tracer simulator that includes the use of IP. Implementation uses Cisco router as Gateway-3 as redundancy gateway and two servers as proxy server. If one server is malfunctioning then other server continues to serve user. Tests were conducted in this final test compare throughput with condition before and after implementasion and test server redundancy path on Gateway-3. The results obtained after implementation showed that the load of throughput inbound is reduced to 45,41% and the load of throughput outbound is reduced to 48,76%. The division of load to both server proxy in Gateway-3 can be balanced. If one proxy server dies then the other proxy server handles all incoming requests to the Gateway-3.

Key words: internet, traffic, proxy server, throughput

1. Pendahuluan

Saat ini teknologi jaringan komputer telah berkembang pesat yang menyebabkan jumlah pengguna meningkat tiap tahun. Penggunaan jaringan komputer yang sering dilakukan adalah akses ke Internet baik sebagai media pembelajaran, akses ke media sosial, *streaming*, ataupun sekadar untuk menjelajahi internet. Universitas Diponegoro (UNDIP) merupakan salah satu instansi pendidikan yang jumlah penggunanya yang mencapai ribuan. Hal ini dapat menyebabkan padatnya lalu lintas jaringan karena jumlah permintaan akses ke internet bertambah setiap waktu.

Lapis *core* atau lapisan inti merupakan *backbone* jaringan. Lapisan ini bertanggung jawab untuk mengirim permintaan pengguna secara cepat dan dapat diandalkan. Masalah umum yang terjadi pada lapis ini adalah beban

kerja yang berat karena harus melewatkan permintaan dalam jumlah banyak dan perangkat yang selalu aktif untuk bekerja. Salah satu cara untuk menanggulangi beban ini, yaitu dengan menggunakan konsep redundansi pada lapis *core*. Redundansi dilakukan dengan menduplikasi perangkat-perangkat yang digunakan dan jalur komunikasi serta meletakkan perangkat di lokasi yang berbeda. Hal ini dilakukan agar lalu lintas jaringan dapat tetap beroperasi apabila salah satu perangkat atau jalur tidak berfungsi atau saat salah satu lokasi mengalami listrik padam.

Tugas akhir ini melakukan perancangan dan implementasi jalur redundansi dengan *server proxy* menggunakan WCCP (*Web Cache Communication Protocol*) pada jaringan Universitas Diponegoro. Perangkat diletakkan di Gedung ICT Centre. Pengujian yang dilakukan ada dua, yaitu mengamati *throughput* jaringan di ke enam *server proxy* yang aktif dan penggunaan *server proxy* pada jalur redundansi.

Tujuan pembuatan penelitian ini adalah :
1. Desain dan implementasi *server proxy* menggunakan WCCP.
2. Mempelajari penggunaan IPTables untuk me-*redirect* permintaan dari pengguna ke *server proxy*.
3. Menurunkan beban kerja *server proxy*.

Untuk menyederhanakan permasalahan dalam penelitian ini maka akan diberikan batasan-batasan sebagai berikut:
1. Melakukan desain dan implementasi hasil rancangan pada lapis *core* di Universitas Diponegoro dengan pengalamatan IPv4.
2. *Server proxy* menggunakan sistem operasi Ubuntu Server dengan aplikasi Squid/LUSCA.
3. Tidak membahas keamanan jaringan.

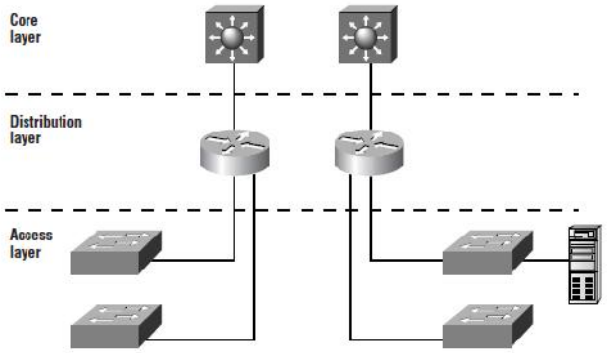
2. Metode

2.1 Jaringan Komputer

Jaringan komputer dapat diartikan sebagai kumpulan sejumlah terminal komunikasi yang berada di berbagai lokasi yang saling berhubungan. Tujuan dibuatnya suatu jaringan komputer adalah membawa informasi secara tepat dan tanpa adanya kesalahan dari sisi pengirim (*transmitter*) menuju ke sisi penerima (*receiver*) melalui media komunikasi[11].

2.2 Hirarki Jaringan

Hirarki memberikan pemahaman dimana sesuatu seharusnya berada, bagaimana segala sesuatunya dapat saling bekerja sama, dan apa fungsinya berada di situ. Hirarki menjelaskan urutan dan kemudahan dalam memahami model yang kompleks[6]. Beberapa kelebihan hirarki, antara lain jaringan dapat dengan mudah diperluas, penggunaan jalur redundansi, dan dapat dengan mudah dipantau oleh admin jaringan.



Gambar 1. Model Hirarki Cisco [6]

Core layer atau lapis inti bertanggung jawab untuk mengangkut lalu lintas dalam jumlah besar secara andal dan cepat. *Distribution layer* sebagai titik komunikasi antara *access layer* dan *core layer*. *Access layer* menyediakan jaringan yang dibutuhkan pengguna lokal[6].

2.3 Konsep Proxy

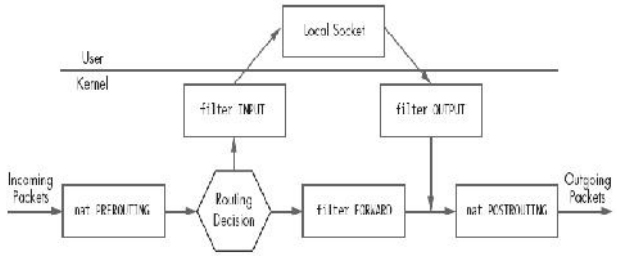
Proxy server adalah sebuah program komputer yang dapat bertindak sebagai komputer lainnya untuk melakukan permintaan terhadap isi dari internet atau intranet. Pengguna yang berinteraksi dengan internet melalui sebuah *proxy server* tidak akan mengetahui bahwa sebuah *proxy server* sedang menangani permintaan yang dilakukannya. *Web server* yang menerima permintaan dari *proxy server* menganggap permintaan-permintaan tersebut seolah-olah datang langsung dari komputer pengguna, bukan dari *proxy server*[1].

2.4 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* 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. Squid sering digunakan untuk protokol HTTP dan *File Transfer Protocol* (FTP), namun selain itu Squid juga mendukung beberapa protokol lainnya termasuk *Transport Layer Security* (TLS), *Secure Socket Layer* (SSL), Internet Gopher, dan *HTTP Secure* (HTTPS)[8].

2.5 IPTables

IPTables digunakan untuk mengontrol sepenuhnya jaringan melalui paket IP pada sistem Linux yang. Sebuah aturan pada IPTables dibuat berdasarkan sekumpulan peraturan yang diberikan pada *kernel* untuk mengatur setiap paket yang datang.



Gambar 2. Diagram Perjalanan Paket data pada IPTables

Gambar 2 menggambarkan perjalanan paket data pada IPTables. Simbol persegipanjang merupakan *filter INPUT*, *filter OUTPUT*, dan *filter FORWARD*. Ketiga *filter* tersebut menggambarkan rantai atau *chain*. Rantai akan memutuskan nasib paket tersebut. Paket akan dibuang jika keputusannya *DROP* dan diteruskan jika keputusannya *ACCEPT*[1].

Sebuah rantai adalah aturan yang ditentukan. Setiap aturan menyatakan "informasi awal (header) pada paket menentukan perlakuan terhadap paket". Paket di*filter* setiap baris aturan yang dikonfigurasi pada IPTables urut dari aturan pertama sampai aturan terakhir. *Kernel* akan melihat kebijakan bawaan (*default*) untuk memutuskan apa yang harus dilakukan kepada paket tersebut, apabila sampai aturan terakhir paket tersebut belum memenuhi salah satu aturan. Ada dua kebijakan bawaan yaitu *default DROP* dan *default ACCEPT*[1].

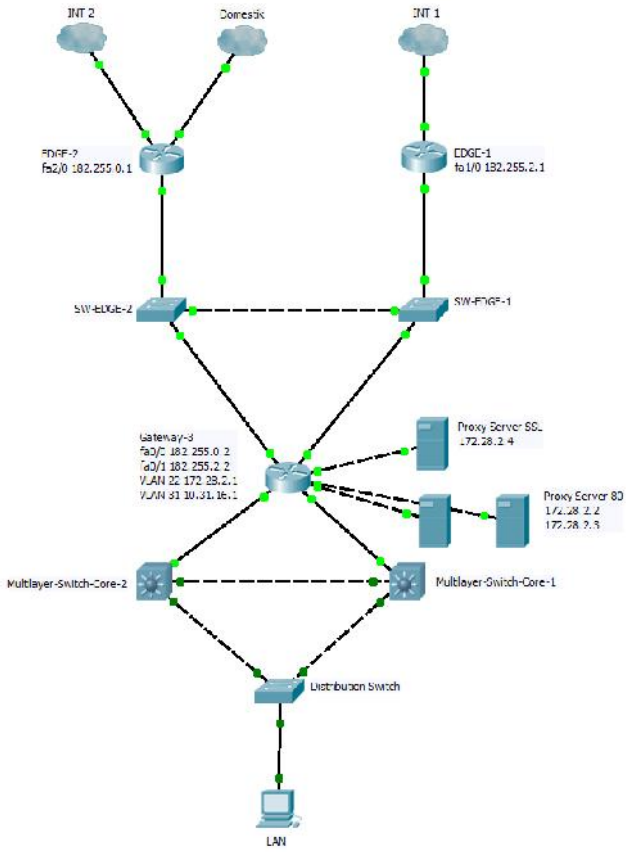
2.6 WCCP (*Web Cache Communication Protocol*)

WCCP adalah protokol yang dikembangkan oleh Cisco untuk mengalihkan lalu lintas menuju *web cache*[5]. Untuk menggunakan WCCP, dibuat *tunnel* GRE (*Generic Routing Encapsulation*) di antara *router* dan mesin yang menjalankan Squid *proxy server*. Pengalihan permintaan dari *router* di-encapsulasi dengan GRE dan dikirim ke *server proxy* melalui *tunnel* GRE. Proses decapsulasi paket GRE dan pengalihan ke Squid dilakukan oleh mesin *host* menggunakan IPTables. Kemudian Squid akan mengambil konten yang diminta dari server asli atau dari *cache* dan mengembalikan konten ke *router*. *Router* akan mengirimkan hasil tanggapan dari *proxy server* ke pengguna[2].

3. Hasil dan Analisa

Perancangan sistem menggunakan perangkat lunak Packet Tracer. Perancangan dilakukan untuk mengetahui bahwa perangkat dapat terhubung dengan IP *address* yang digunakan.

Jalur redundansi yang ditambahkan di Universitas Diponegoro menggunakan Cisco Router dan tiga buah *server* dimana satu *server* berfungsi sebagai *proxy* untuk menangani permintaan yang menggunakan SSL (port 443) dan dua *server proxy* untuk menangani permintaan dari *port* 80.



Gambar 3. Topologi logikal di Universitas Diponegoro

Setelah perancangan dengan menggunakan Packet Tracer, kemudian dilakukan implementasi pada perangkat riil di UNDIP dan dilakukan pengujian. Pengujian yang dilakukan ada dua, yaitu pengujian dengan *throughput* dan redundansi dari *proxy server*. Berikut adalah hasil pengambilan *throughput* sebelum dan setelah implementasi.

Tabel 1. Perbandingan *throughput inbound dan outbound* sebelum dan setelah implementasi

Proxy	Sebelum implementasi		Setelah implementasi	
	inbound	outbound	inbound	outbound
1	19,56 Mb/s	20,39 Mb/s	12,5 Mb/s	13,2 Mb/s
2	36,99 Mb/s	38,71 Mb/s	11,0 Mb/s	12,2 Mb/s
3	23,12 Mb/s	24,95 Mb/s	10,7 Mb/s	11,5 Mb/s
4	20,50 Mb/s	21,20 Mb/s	12,5 Mb/s	13,0 Mb/s
5	X	X	10,6 Mb/s	11,2 Mb/s
6	X	X	10,9 Mb/s	13,2 Mb/s
Jumlah	100,17 Mb/s	105,25 Mb/s	68,2 Mb/s	74,3 Mb/s
Rata-rata	25,04 Mb/s	26,31 Mb/s	11,37 Mb/s	12,83 Mb/s

Tabel 1 memberi informasi bahwa *throughput inbound* lebih kecil daripada *throughput outbound* baik sebelum

maupun setelah implementasi. Hal ini disebabkan permintaan konten dari pengguna sebagian besar telah tersedia di *proxy server* sehingga konten tersebut diambilkan dari penyimpanan di *proxy server*. Perbedaan nilai *throughput inbound* dan *throughput outbound* pada saat sebelum dan setelah implementasi juga berbeda cukup jauh. Hal ini menunjukkan bahwa dengan penambahan dua *proxy server* beban yang ditanggung menjadi lebih ringan. Pada Proxy-1 perbedaan *throughput inbound* sebesar 7,06 Mb dan *throughput outbound* sebesar 7,19 Mb. Proxy-2 memiliki perbedaan *throughput inbound* sebesar 25,99 Mb dan *throughput outbound* sebesar 26,51 Mb. Proxy-3 memiliki perbedaan *throughput inbound* sebesar 12,42 Mb dan *throughput outbound* sebesar 13,45 Mb. Proxy-4 memiliki perbedaan *throughput inbound* sebesar 8 Mb dan *throughput outbound* sebesar 8,2 Mb. Berdasarkan data-data tersebut dapat disimpulkan bahwa beban kerja yang ditanggung oleh perangkat menjadi lebih ringan karena beban kerja telah dibagi ke enam *server proxy*. Sebelum implementasi, keempat *server proxy* rata-rata menanggung 25,04 Mb untuk *throughput inbound* dan 26,3 Mb untuk *throughput outbound* sedangkan setelah implementasi keenam *server proxy* rata-rata menanggung 11,37 Mb untuk *throughput inbound* dan 12,38 Mb untuk *throughput outbound*. Berikut adalah perhitungan pengurangan *throughput inbound* dan *throughput outbound* dalam presentase.

Throughput inbound:

$$\frac{11{,}37}{25{,}04} \times 100\% = 45{,}41\%$$

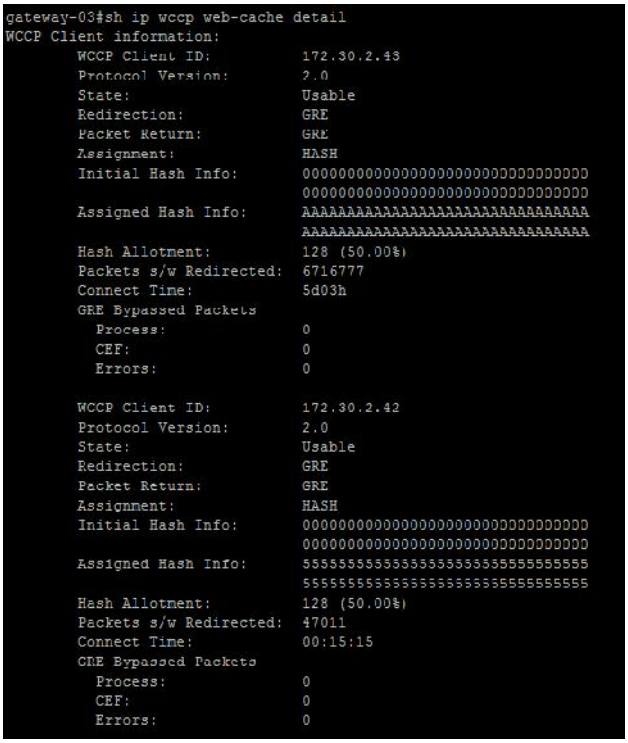
Throughput outbound:

$$\frac{12{,}83}{26{,}31} \times 100\% = 48{,}76\%$$

Jika dihitung dalam presentase maka beban *throughput inbound* berkurang hingga 45,41% dan beban *throughput outbound* berkurang hingga 48,76%.

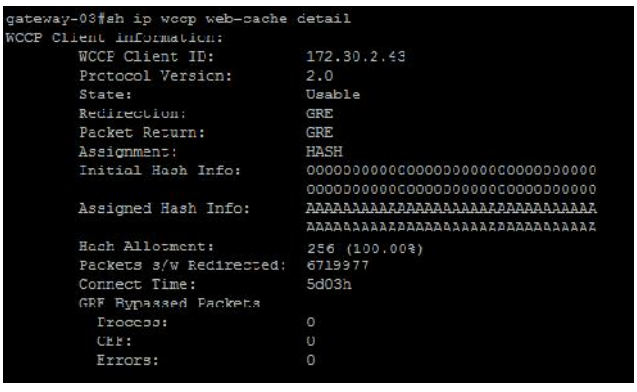
Dari tabel 1 juga dapat diketahui bahwa sebelum implementasi, pada Proxy-1 dapat mengurangi *bandwidth download* dari internet sebesar 0,83 Mb/s, Proxy-2 sebesar 1,72 Mb/s, Proxy-3 sebesar 1,83 Mb/s, dan Proxy-4 sebesar 0,7 Mb/s. Sedangkan setelah implementasi, pada Proxy-1 dapat mengurangi *bandwidth download* dari internet sebesar 0,7 Mb/s, Proxy-2 sebesar 1,2 Mb/s, Proxy-3 sebesar 0,8 Mb/s, Proxy-4 sebesar 0,5 Mb/s, Proxy-5 sebesar 0,6 Mb/s, dan Proxy-6 sebesar 2,3 Mb/s. Jika dijumlah maka *server proxy* sebelum implementasi dapat mengurangi penggunaan *bandwidth* sebesar 5,08 Mb/s sedangkan setelah implementasi dapat mengurangi penggunaan *bandwidth* sebesar 6,1 Mb. Jumlah *bandwidth* yang dapat dihemat setelah implementasi terpaut 1,02 Mb/s.

Pengujian lain yang dilakukan adalah pengujian jalur redundansi pada *proxy server*. Dimana untuk menangani permintaan dari port 80 digunakan dua *proxy server*. Kedua *proxy server* tersebut dapat membagi beban kerja yang sama, yaitu 50% pada masing-masing *proxy* dengan menggunakan layanan WCCP. Presentase penggunaan masing-masing *proxy* dapat dilihat pada Gambar 4.



Gambar 4. Pembagian beban kerja pada *Proxy Server*

Sedangkan saat salah satu *proxy server* mengalami masalah, misalnya perangkat mendadak mati, maka *proxy server* lainnya akan mengambil alih semua permintaan yang masuk ke *gateway* kedua *proxy* tersebut. *Proxy server* tersebut akan menangani 100% dari semua permintaan yang ada seperti yang ditunjukkan pada Gambar 5 di bawah ini.



Gambar 5. Beban kerja satu *Proxy Server*

Berdasarkan Gambar 5 dapat diketahui, layanan tetap dapat digunakan walaupun salah satu *proxy server* mati.

4. Kesimpulan

Pada desain dan implementasi di Sistem Informasi Fakultas Teknik diketahui bahwa *Virtual Local Area Network* (VLAN) tidak dapat melewatkan sebuah enkapsulasi *Generic Routing Encapsulation* (GRE). Hasil setelah implementasi menunjukkan bahwa beban *throughput inbound* berkurang hingga 45,41% dan beban *throughput outbound* berkurang hingga 48,76%. Jumlah *bandwidth* yang dapat dihemat sebelum dan setelah implementasi hanya memiliki selisih 1,02 Mb/s. *Router* yang menggunakan fungsi WCCP dapat membagi beban secara automatis ke beberapa *server proxy* sehingga tidak diperlukan konfigurasi *balancing*. Jika salah satu *proxy server* mati atau tidak dapat melayani permintaan maka secara otomatis *proxy server* yang lain akan mengambil alih seluruh permintaan yang masuk ke *gateway*. Adapun kelemahan pada sistem ini adalah *gateway* belum bisa secara otomatis mengalihkan permintaan jika proses pada *gateway* terlalu padat. Untuk mengatasi kendala tersebut, dapat menggunakan GLBP (*Gateway Load Balancing Protocol*) yang dapat secara otomatis melempar permintaan yang tidak dapat ditangani oleh satu *gateway* ke *gateway* yang lain.

Referensi

[1]. Alvin, Yosua, Adian F.R., R. Rizal Isnanto, *Penyeimbangan Beban Transparent Squid/Lusca Proxy dengan Metode Destination NAT Round Robin dengan Multiple Captive Portal Sebagai Media Autentikasi Untuk VLAN Terpadu*, Skripsi-S1, Universitas Diponegoro, Semarang, 2012.

[2]. Andriyono, *Impact Proxy Server Terhadap Jaringan Internet*, http://andrtux.wordpress.com/2012/04/09/impact-proxy-server-terhadap-internet/, 2007.

[3]. Chadd, Adrian, *LUSCA Web Proxy Cache*, http://www.lusca.org/Home, 2010.

[4]. Chatel, M., *Classical versus Transparent IP Proxies*, Network Working Group RFC:1919, 1996.

[5]. Cisco Team, *Chapter 10 Configuring DHCP, DDNS, and WCCP Services*, http://www.cisco.com/en/US/docs/security/asa/asa72/configuration/guide/dhcp.html, 1 November 2013.

[6]. Lammle, Todd, *CCNA Cisco Certified Network Associate Study Guide Second Edition*, Sybex Inc., USA, 2000.

[7]. Lukman, Tutorial IPTables, http://rootbox.or.id/tips/iptables.html, 7 November 2013.

[8]. Saini, Kulbir, *Squid Proxy Server 3.1 Beginners Guide*, Packt Publishing, Birmingham, 2011.

[9]. Thompson, Kerry, *Transparent Web Proxying with Cisco, Squid, and WCCP*, http://www.crypt.gen.nz/papers/cisco_squid_wccp.html, Juli 2010.

[10]. Visolve Squid Team, *Transparent Cache Implementation Using Squid*, 2006.

[11]. --- , *Pelatihan Jaringan Komputer*, Modul, http://repository.usu.ac.id/bitstream/123456789/20228/4/Chapter%20II.pdf, 18 Agustus 2013.

[12]. C. Wyld, David, Michal Wozniak, dkk. *Advances in Network Security and Applications*, Springer, New York, 2011.