PERANCANGAN SISTEM FILTERING AKSES WEBSITE DENGAN MENERAPKAN OTORISASI LDAP GRUP PADA PROXY SERVER

Noven Indra Prasetya
Program Studi Teknik Informatika Fakultas Teknik
Universitas Wijaya Kusuma Surabaya
Jl. Dukuh Kupang XXV / 54, Surabaya, Jawa Timur
noven@uwks.ac.id

Abstrak. *Penggunaan internet yang tidak tepat dapat memberikan dampak negatif pada pengguna khususnya mahasiswa. Hal ini disebabkan karena waktu mereka banyak dihabiskan hanya untuk mengakses website penyedia jasa permainan online dan website pertemanan atau yang dikenal dengan sebutan social media, bahkan juga digunakan untuk mengakses dan men-download gambar maupun video yang tergolong dalam kategori pornografi. Oleh karena itu dibutuhkan suatu sistem yang dapat digunakan untuk membatasi akses penggunaan internet pada area kampus, dengan tujuan untuk mengatasi permasalahan yang terjadi dengan lebih baik. Penelitian ini dilakukan untuk merancang suatu sistem yang dapat digunakan untuk membatasi akses pengguna terhadap alamat website yang dituju dengan menerapkan sistem filtering akses website pada Proxy Server, serta dengan cara melakukan otorisasi pengguna berdasarkan jenis account yang digunakan untuk mengakses internet menggunakan mekanisme penggolongan account berdasarkan grup pada Lightweight Directory Access Protocol . Jika jenis account-nya umum, maka pengguna tidak akan bisa mengakses website permainan, social media, bahkan pornografi. Namun apabila jenis account-nya khusus, seperti contoh penggunanya adalah Dosen, maka pengguna tersebut dapat mengakses semua website, kecuali website yang mengandung unsur pornografi. Sehingga sistem yang dibangun diharapkan dapat mengurangi penggunaan internet yang tidak bermanfaat khususnya dilingkungan kampus, dan penggunaan fasilitas internet dapat bermanfaat dengan baik untuk proses belajar mengajar dilingkungan kampus tersebut.*

Kata Kunci: Proxy Server, ***Lightweight Directory Access Protocol***, LDAP

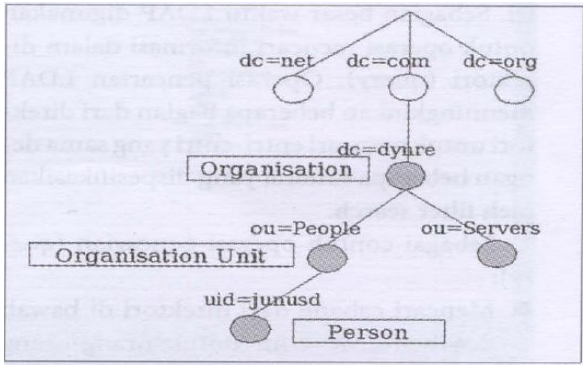
Penggunaan internet yang tidak tepat dapat memberikan dampak negatif pada pengguna khususnya mahasiswa. Hal ini dapat dilihat pada sebagian besar mahasiswa di salah satu universitas terkenal di surabaya, banyak menghabiskan waktunya di kampus hanya untuk mengakses *website* penyedia jasa permainan *online* dan *website* pertemanan atau yang dikenal dengan sebutan *social media*, bahkan juga digunakan untuk mengakses dan men-*download* gambar maupun video yang tergolong dalam kategori pornografi. Sehingga tujuan diberikannya fasilitas internet oleh kampus tidak digunakan secara maksimal untuk mendukung kegiatan belajar mereka sepenuhnya. Oleh karena itu diperlukan suatu sistem yang dapat digunakan untuk membatasi akses penggunaan internet pada area kampus, sehingga permasalahan yang terjadi dapat teratasi dengan baik.

Penelitian yang dilakukan kali ini dimaksudkan untuk mengatasi permasalahan yang terjadi seperti uraian diatas, yaitu merancang suatu sistem yang dapat digunakan untuk membatasi akses pengguna terhadap alamat *website* yang dituju. Semua pengguna internet yang menggunakan fasilitas internet kampus akan diotorisasi berdasarkan jenis *account* yang digunakan untuk mengakses internet. Apabila jenis *account*-nya umum, maka pengguna tidak akan bisa mengakses website permainan, social media, bahkan pornografi. Namun apabila jenis *account*-nya khusus, seperti contoh penggunanya adalah Dosen, maka pengguna tersebut dapat mengakses semua website, kecuali website yang mengandung unsur pornografi. Sehingga sistem yang diterapkan diharapkan dapat mengurangi penggunaan internet yang tidak bermanfaat khususnya dilingkungan kampus, dan penggunaan fasilitas internet dapat bermanfaat dengan baik untuk proses belajar mengajar dilingkungan kampus tersebut.

Tujuan dari penelitian ini adalah merancang sistem yang dapat digunakan untuk mem-filter atau membatasi alamat website yang diakses oleh pengguna internet diarea kampus serta untuk menerapkan *Lightweight Directory Access Protocol* pada *Proxy Server* yang digunakan untuk meng-otorisasi pengguna internet berdasarkan grup yang sudah ditentukan.

Lightweight Data Access Protocol

LDAP merupakan singkatan dari *Lightweight Directory Access Protocol* (Protokol Akses Direktori Ringan). Artinya, ini adalah protokol kelas ringan untuk mengakses servis direktori, yang berdasarkan pada protokol servis direktori *X.500*. *LDAP* berjalan melalui protokol *TCP/IP*. Pendefinisian secara detail *LDAP* ada dalam *RFC 225l "The Lightweight Directory Access Protocol (v3)"* dan dokumen lainnya menyatakan spesifikasi teknik ada pada *RFC 3377*. Model informasi *LDAP* adalah berdasarkan entri. Sebuah entri adalah koleksi atribut yang mempunyai nama yang terbedakan (*Distinguished Name* / *DN*) secara global. *DN* ini digunakan sebagai referensi ke entri yang secara unik berbeda dengan nilai *DN* yang lainnya. Setiap atribut entri mempunyai sebuah tipe dengan satu nilai atau lebih. Tipe biasanya *string* singkatan khusus, seperti "*cn*" untuk *common name*, atau "*mail*" untuk alamat *e-mail*. Sintaks dari nilai bergantung kepada tipe atribut. Contoh, atribut *cn* mungkin berisi kata-kata "Junus Djunawidjaja". Atribut *mail* mungkin berisi alamat *email* "*junus@dynre.com*". Atribut *jpeg* Photo mungkin terdiri sebuah foto dalam format binari *JPEG* [1].



Gambar 1. Pohon Hirarki Direktori LDAP

Dalam *LDAP*, entri direktori disusun dalam sebuah hirarki struktur seperti pohon (*tree*). Struktur pohon *LDAP* pada umumnya saat sekarang ini berdasarkan nama *domain internet*. Pendekatan penamaan servis direktori mirip dengan penamaan pada *DNS* ini yang paling popular. Gambar 1 menunjukkan sebuah contoh pohon direktori *LDAP* menggunakan penamaan berdasarkan domain. Selain secara struktur pohon dengan penamaan internet, juga dapat berupa struktur pohon dengan cara penamaan tradisional. Struktur ini merefleksikan geografis atau lingkup organisasi. Entri-entri mewakili negara-negara, terlihat di atas dari pohon (*tree*). Di bawah mereka adalah entri yang menyatakan provinsi dan organisasi nasional. Di bawah nya lagi mungkin entri yang menyatakan unit organisasi, orang, printer, dokumen, dan lain-lain [5].

LDAP dapat mengontrol atribut-atribut yang diperlukan dan diizinkan dalam sebuah entri, melalui penggunaan atribut spesial yang dinamakan *objectClass*. Angka-angka dari atribut *objectClass* menyatakan aturan (*rule*) *schema* yang ditaati oleh entri. Sebuah entri direferensi oleh nama yang berbeda dari yang lain, yang dibentuk dari nama entri itu sendiri. Ini dinamakan nama relatif yang membedakan (*Relative Distinguished Name* / *RDN*) dan menggabungkan nama-nama dari entri-entri di atasnya atau sebelumnya. Sebagi contoh, entri untuk "Junus Djunawidjaja" pada penamaan berdasarkan *internet* contoh di atas mempunyai *RDN*: *uid=junusd* dan *DN* dari *uld=junusd, ou=People, dc=dynre, dc=com*. Operasi *update* yang ada, seperti untuk menambah dan menghapus sebuah entri dari direktori, adalah mengubah entri yang ada dan mengubah nama dari sebuah entri. Sebagian besar waktu *LDAP* digunakan untuk operasi mencari informasi dalam direktori (*query*). Operasi pencarian *LDAP* memungkinkan beberapa bagian dari direktori untuk mencari entri-entri yang sama dengan beberapa kriteria yang dispesifikasikan oleh *filter search*. Sebagai contoh operasi pencarian (*query*): (1) Mencari cabang dari direktori di bawah *dc=dynre, dc=com* untuk orang-orang dengan nama *"Junus"*, lalu memanggil alamat kantor dari setiap entri yang

ditemukan. (2) Atau juga dimungkinkan untuk mencari entri secara keseluruhan di bawah entri *ou=Group* untuk organisasi yang mempunyai kelompok *Group="admin"*, lalu menampilkan semua anggota group di dalamnya [1].

Layanan Direktori LDAP

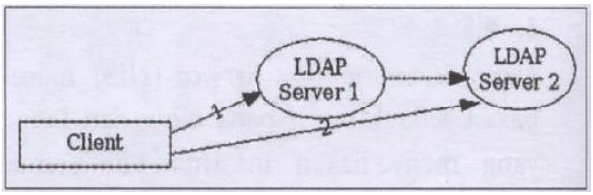
Lightweight Data Access Protocol atau LDAP memiliki beberapa keuntungan layanan direktori yaitu: (1) Mempermudah administrasi suatu jaringan. (2) Struktur atribut penamaan yang seragam. (3) Manajemen terpusat dari informasi data personal, konfigurasi mesin dan komputer, serta profil pengguna. (4) Potensial untuk *single sing-on* dalam suatu sumber daya suatu jaringan. (5) Memberikan hanya satu tujuan bagi pengguna untuk melakukan suatu pencarian data. (6) Lokasi terpusat untuk sumber daya jaringan. (7) Potensial sebagai katalog dimana dapat menyimpan beragam data; misalnya dokumentasi, profil, gambar/foto, dan sebagainya. (8) Meningkatkan manajemen data, konsistensi data, dan keamanan dalam pengaksesan data. (9) Menyediakan media penyimpanan dan pencarian untuk aplikasi dan layanan data dalam struktur hirarki [5].

Jika dibandingkan dengan *database*, direktori LDAP memiliki beberapa kelebihan, diantaranya adalah sebagai berikut: (1) Lebih fleksibel dan mempunyai skala yang lebih besar. (2) Menangani berbagai ragam data dengan baik. (3) Memfasilitasi replikasi (duplikat data untuk *back up*) sebagai kehandalan. (4) Mendukung manajemen distribusi data yang telah tersimpan. (5) Lebih mudah dan murah untuk dikelola. Direktori LDAP dapat diimplementasikan untuk berbagai keperluan, diantaranya sebagai berikut: (1) Layanan direktori informasi data suatu perusahaan, institusi pendidikan, maupun organisasi. (2) Profil pengguna atau anggota suatu aplikasi website. (3) *Directory Enabled Networks* (DEN) jaringan cerdas yang menyimpan informasi mengenai komponen jaringan dan tersedia dalam layanan direktori. (4) Kendali akses dan proses autentikasi [1].

Cara Kerja LDAP

Secara teknis, *LDAP* adalah sebuah protokol untuk mengakses ke servis direktori *X.500*, yang merupakan direktori servis yang diatur oleh *OSI*. Awalnya, client *LDAP* mengakses *gateway* ke servis direktori *X.500*. *Gateway* ini menjalankan *LDAP* di antara *client* dan *gateway*, dan menjalankan Protokol Akses Direktori (*Directory Access Protocol* / *DAP*) *X.500* antara *gateway* dan *X.500* server. *DAP* adalah sebuah protokol kelas berat yang beroperasi melalui tumpukan protokol *OSI* secara penuh dan memerlukan pemrosesan yang sangat signifikan dari sumber daya komputasi. *LDAP* didesain untuk beroperasi melalui *TCP/IP* dan menyediakan sebagian besar dari fungsi *DAP* dengan biaya yang sangat lebih rendah.

Service direktori *LDAP* berdasarkan model *client-server*. Satu atau lebih server *LDAP* membentuk pohon (*tree*) direktori informasi. *Client* terkoneksi ke *server* dan mengajukan pertanyaan. *Server* merespon dengan jawaban dan/atau dengan *pointer*, ke arah mana *client* dapat mendapat tambahan informasi (khususnya ke *server LDAP* yang lain). Gambar 2 menunjukkan proses koneksi dari *client* ke *server LDAP* pertama dan *server LDAP* kedua yang ditunjuk oleh *server LDAP* pertama [1].



Gambar 2. Proses Koneksi Dari Client ke Server LDAP Pertama dan Kedua

Tidak masalah pada *server LDAP* yang mana seorang *client* akan terkoneksi. *Client* tersebut akan mendapat informasi yang sama dari *server* direktori berupa sebuah nama yang direpresentasikan ke satu *LDAP server* sebagai entri referensi yang akan menunjuk ke *server LDAP* lainnya. Ini ciri khas penting bagi servis direktori global seperti *LDAP*. Servis direktori *LDAP* menyediakan proteksi keamanan, yang dapat diset pada saat orang akan melihat informasi diharuskan untuk melewati proses authentifikasi atau login terlebih dahulu. Sehingga orang yang tidak terautentifikasi identitasnya, tidak berhak untuk melihatnya [5].

LDAP Server Pada Lingkungan Unix Dan Linux

Pada lingkungan sistem operasi *Unix* dan *Linux* terdapat berbagai macam *LDAP server* misalnya: (1) IBM Directory Server dan Sun One Directory Server sebagai contoh produk proprietary atau tidak free. (2) University of Michigan LDAP server dan OpenLDAP server sebagai contoh produk yang free. Hal ini akan membahas penggunaan *OpenLDAP server*, karena software ini telah menjadi *LDAP server* standar pada berbagai distribusi *Linux* besar seperti *Red Hat*, *SUSE*, *Mandrake*, maupun *Debian*. *OpenLDAP server* dibuat berdasarkan pada versi terakhir dari *University of Michigan LDAP Server*.

OpenLDAP mempunyai *daemon slapd* dan *slurpd*. *Daemon slapd* yang berdiri sendiri dapat dilihat sebagai sebuah servis direktori *X.500* kelas ringan. Ini tidak mengimplementasikan Protokol Akses Direktori kelas berat *X.500*. Sebagai direktori *server* kelas ringan, *slapd* hanya mengimplementasikan sebagian kecil dari model *X.500*. Sedangkan *daemon slurpd* digunakan untuk mereplikasi informasi direktori dari *daemon slapd*. *Client* tidak dapat meng-update informasi direktori yang ada pada *slurpd* secara langsung. *slurpd* akan mereferensi ke *slapd* jika ada permintaan untuk update informasi. Guna dari replikasi direktori menggunakan *slurpd* untuk memperingan beban pada *daemon* utama *slapd*, serta untuk redudansi pada saat *slapd* tidak berjalan. Jadi, mesin-mesin *client* akan tetap dapat mengakses informasi direktori melalui *slurpd* [2].

Pembahasan di atas merupakan penjelasan prinsip dasar *LDAP* secara garis besar sehingga diharapkan pembaca awam akan mudah untuk memahaminya. Jika pembaca ingin mendapatkan informasi lebih detail dapat membaca *OpenLDAP User Guide* di *www.opmldap.org*.

Proxy

Proxy merupakan istilah untuk sebuah mesin atau komputer yang berada di antara sebuah jaringan lokal dengan jaringan Internet. Komputer yang biasanya difungsikan sebagai *Proxy* juga berlaku sebagai *gateway*. *Proxy* bekerja dengan melayani berbagai macam *request* yang dikirimkan dari jaringan lokal di bawahnya untuk berhubungan dengan jaringan Internet. Selanjutnya *Proxy* juga berfungsi untuk meneruskan paket yang datang untuk membagikan data yang diterimanya dari Internet untuk diberikan kepada komputer klien yang ada di jaringan lokalnya sesuai dengan *request* yang dikirimkan oleh masing-masing komputer klien [3].

Artinya apabila komputer yang terdapat dalam sebuah jaringan lokal terhubung dengan Internet, maka sudah pasti akses ke Internetnya tidak akan bisa langsung, melainkan harus melewati mesin *Proxy* terlebih dahulu. Selanjutnya *Proxy Server* juga diberikan kewenangan untuk membuat keputusan apakah *request* dari komputer klien ini akan ditolak (*reject*) ataupun diteruskan (*forward*).

Berikut merupakan fungsi utama yang dimiliki oleh *Proxy Server* dalam suatu infrastruktur jaringan: (1) Connection Sharing : Dalam suatu jaringan lokal yang terhubung ke jaringan lain atau internet, pengguna tidak langsung berhubungan dengan jaringan luar atau internet, tetapi harus melewati suatu *gateway*, yang bertindak sebagai batas antara jaringan lokal (privat) dan jaringan luar (publik). *Gateway* ini sangat penting, karena jaringan lokal harus dapat dilindungi dengan baik dari bahaya yang mungkin berasal dari internet, dan hal tersebut akan sulit dilakukan bila tidak ada garis batas yang jelas antara jaringan lokal dan internet. *Gateway* juga bertindak sebagai titik dimana sejumlah koneksi dari pengguna lokal akan terhubung kepadanya, dan suatu koneksi ke jaringan luar juga terhubung kepadanya. Dengan demikian, koneksi dari jaringan lokal ke internet akan menggunakan sambungan yang dimiliki oleh *gateway* secara bersama-sama (*connection sharing*). Dalam hal ini, *gateway* adalah juga sebagai *proxy server*, karena menyediakan layanan sebagai perantara antara jaringan lokal dan jaringan luar atau internet. Singkatnya, 1 *IP public* dapat digunakan oleh banyak user, selain itu juga untuk melindungi jaringan dalam dari serangan luar. (2) Filtering : *Filtering*

merupakan sebuah usaha pengamanan atau pembatasan sehingga dengan adanya *filtering* sebuah *proxy server* dapat mengamankan dan membatasi hak akses *client* pada jaringan privat. Jadi meskipun mula-mula dibuat sebagai *cache nonsecurity*, tujuan utama *proxy server* sekarang menjadi *firewalling*. *Proxy server* memperbarui *request* layanan pada jaringan eksternal atas nama client mereka pada jaringan *private*. Ini secara otomatis menyembunyikan identitas dan jumlah client pada jaringan internal dari jaringan eksternal. Karena posisi mereka di antara *client internal* dan *public server*, *proxy* juga dapat menyimpan *content* yang sering diakses dari jaringan publik untuk mengurangi akses ke jaringan publik tersebut. Kebanyakan implementasi nyata *proxy* security meliputi pemfilteran paket dan *Network Address Translation* untuk membangun *firewall* yang utuh. Teknologi tersebut dapat digabungkan dengan *proxy* untuk menghilangkan serangan yang terhadapnya *proxy* rentan. (3) Caching : *Caching* atau *Internet Object Caching* adalah suatu cara untuk menyimpan hasil permintaan *internet-object*. (seperti: data yang ada dari *HTTP*, *FTP*, dan *ghoper protocol*) untuk membuat sistem dekat dengan permintaan daripada ke sumber aslinya. *Web browser* dapat menggunakan lokal *squid cache* sebagai *proxy HTTP server*, ini akan mengurangi waktu akses seperti halnya penghematan bandwidth. Dengan kata lain sebuah *client* tidak harus melakukan kontak dengan *server* untuk meminta layanan akan tetapi *client* dapat mendapatkan layanan (data) yang sudah tersimpan pada *proxy server*, dengan hal ini maka akses akan semakin cepat [2].

Berikut ini terdapat beberapa manfaat yang bisa dilakukan dari sebuah mesin *Proxy* atau *Proxy Server*, yaitu: (1) Mempercepat akses ke situs atau website yang pernah dijalankan. (2) Memblokir situs atau content pornografi. (3) Menolak file-file tertentu, misalnya file video seperti *avi*, *wmv*, *mov*, dan sebagainya. (4) Menolak software, aplikasi atau script-script yang berbahaya, misalnya *virus*, *trojan*, *maleware*, dan sebagainya. (5) Menghilangkan *banner* dan sebagainya [4].

Proxy juga dapat melakukan kerja sama antar *Proxy Server* lainnya atau biasa disebut dengan *hirarki cache*, tujuannya agar kinerjanya lebih maksimal. Terdapat dua macam *hirarki cache* terkait dengan hubungan antar mesin *Proxy*, yaitu: (1) Parent (a) Berkewajiban untuk selalu memberikan objek Internet yang merupakan *request* dari komputer klien. (b) Apabila dalam server *parent* tidak memilikinya, komputer klien akan tetap menunggu sampai mencapai waktu time out. (c) Selanjutnya *browser* baru akan mengambil sendiri objeknya langsung sesuai dengan konfigurasi yang telah ditentukan sebelumnya. (2) Sibling (a) *Proxy Server* yang difungsikan sebagai *sibling* hanya akan melayani *request* komputer klien apabila memang dalam *Proxy Server* tersebut telah menyimpan objek yang diminta oleh komputer klien. (b) Jika tidak, maka *Proxy Server sibling* ini tidak mencarikan objek yang diminta ke situs yang bersangkutan [2].

Squid

Squid merupakan *software public domain* berbasis *UNIX* yang digunakan untuk *software* default *Proxy*. *Squid* juga disebut sebagai mesin *caching proxy* untuk klien web, seperti *HTTP*, *HTTPS*, *FTP*, *gopher* dan layanan sejenis lainnya. *Squid* mampu menurunkan konsumsi *bandwidth* sekaligus mempercepat waktu respons. Ini terwujud dengan melakukan *caching* halaman web dan menggunakan ulang halaman yang sering dikunjungi oleh pengguna. *Squid* memiliki setumpuk kendali akses yang dapat mendongkrak kecepatan server karena *squid* menangani semua *request* melalui sebuah proses I/O tunggal. *Squid* bekerja dengan cara menyimpan *meta data*, terutama pada objek yang sering diakses, mereka dijaga (di-cache) dalam memori (RAM) komputer [3].

Fitur Squid

Squid, sebagai *software proxy* berlisensi *open source* yang dilengkapi dengan segudang fitur yang ditawarkan, juga mendukung *SSL*, *extensive access control*, dan *logging request* yang lengkap. Dengan menggunakan *Internet Cache Protocol* yang ringan, *squid cache* dapat disusun dalam format *hierarkis* atau *mesh* untuk penghematan *bandwidth* tambahan. Secara garis

besar, *squid* terdiri dari program server utama *squid*, program *DNS* (Domain Name System) *lookup*, program *dns server* (*Squid* sekarang mengimplementasikan protocol DNS secara default), beberapa program opsional untuk penulisan ulang *request* dan pembentukan *otentikasi*, serta sejumlah *tools* manajemen [4].

Hubungan Squid Dengan SSL, HTTPS, TLS

Mulai versi 2.5, *squid* menambahkan fitur yang dapat menghentikan sambungan/koneksi *SSL*, yang kemungkin hanya digunakan dalam situasi tertentu. Untuk mengaktifkan fitur ini, jalankan perintah *configure* dengan menambahkan parameter **--enable-ssl** ketika melakukan instalasi *squid* pertama kali. *Squid* juga dapat mendukung protokol terenkripsi ini melalui proses *tunneling* trafik di antara klien dan server. Dalam hal ini, *squid* melakukan *relay* pada bit-bit terenkripsi yang hilir-mudik di antara klien dan server. Normalnya, ketika browser melintasi sebuah URL *https, squid* akan melakukan salah satu proses berikut: (1) Browser membuka koneksi *SSL* secara langsung ke server asal. (2) Browser melakukan *tunnelling* terhadap *request* tersebut dengan *squid*, melalui metode *request CONNECT*. Metode *CONNECT* adalah jalan untuk melakukan *tunneling* setiap koneksi yang melintasi mesin *HTTP proxy*, di mana mesin *proxy* ini hanya melepas byte-byte di antara klien, dan server tidak akan mengerti atau mampu menginterpretasi isi di dalamnya [4].

I. Metodologi

Pada pengamatan yang dilakukan selama melakukan penelitian pada lingkungan kampus tempat yang dipilih oleh penulis sebagai tempat penelitian, ternyata ditemukan beberapa website yang selalu diakses oleh pengguna dilingkungan kampus tersebut, sehingga pada pengujian nantinya akan dibuatkan suatu skenario pengujian yang didalamnya berisi penjelasan tentang bagaimana sistem yang dibangun dapat menolak akses pengguna terhadap website yang sudah dikumpulkan pada pengamatan yang dilakukan sebelumnya.

Daftar Website Pengujian

Daftar website pengujian disini berisi beberapa website yang sering diakses oleh pengguna, dan daftar website akan diklasifikasi menjadi dua yaitu website pertemanan dan website permainan.

1. Website Pertemanan
   Website atau situs pertemanan berikut merupakan website yang paling sering diakses oleh pengguna, sehingga website inilah yang digunakan pada pengujian penelitian ini untuk membuktikan apakah sistem yang dibangun dapat mem-filter website tersebut sesuai dengan ketentuan otorisasi yang ditentukan. Berikut website pertemanan yang digunakan:
   a. Facebook, http://www.facebook.com.
   b. Twitter, http://www.twitter.com.
   c. Instagram, http://www.instagram.com.
   d. Google+, https://plus.google.com.
   e. Foursquare, https://foursquare.com.
   f. Flickr, http://www.flickr.com.

2. Website Permainan
   Website atau situs permainan berikut merupakan website yang paling sering diakses oleh pengguna, sehingga website inilah yang digunakan pada pengujian penelitian ini untuk membuktikan apakah sistem yang dibangun dapat mem-filter website tersebut sesuai dengan ketentuan otorisasi yang ditentukan. Berikut website permainan yang digunakan:
   a. Point Blank, http://pb.gemscool.com.
   b. Ayo Dance, http:// www.ayodance.com.
   c. Rising Force, http://rf.lytogame.com.
   d. Perfect World, http://perfectworld.lytogame.com.
   e. Ragnarok Online, http://ragnarok.lytogame.com.
   f. Aion, http://aion-idgs.com.

Skenario Pengujian

Skenario pengujian dibuat dengan tujuan supaya konsep penelitian yang dilakukan bisa terpenuhi dengan baik. Di bawah ini adalah beberapa skenario yang dilakukan pada penelitian ini:

1. Komputer yang berperan sebagai *proxy server* dan *LDAP server* adalah komputer *gateway*.
2. Komputer *proxy*/*gateway* memiliki dua *interface network*, yaitu *eth0* dan *eth1*, eth0 terhubung langsung dengan modem

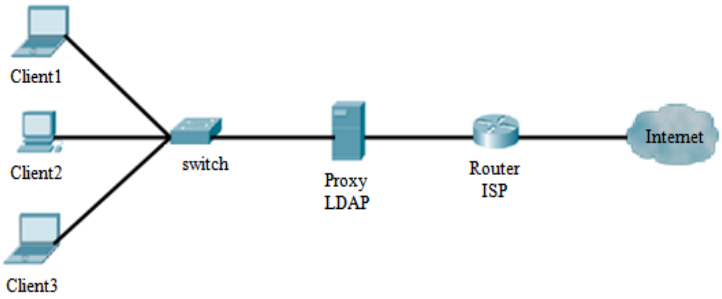
ADSL , dan *eth1* terhubung ke jaringan lokal dengan *IP* 10.134.11.254.

3. Agar tidak ada pengguna yang dapat mengakses internet (*http*) secara langsung maka pada *gateway* diterapkan kebijakan *firewall* yang memblok semua akses *http* (*port 80*) secara langsung dari *LAN* ke internet. Hal ini dimaksudkan agar pengguna harus mengatur *connection setting* pada *web client* atau *browser* dengan setingan *manually use proxy*. Pada *web client* atau *browser*, *proxy* diseting ke *IP* 10.134.11.254 yang merupakan *IP address* milik *proxy server* dan *port 3128*.
4. Skenario otentikasi, *proxy* hanya mengijinkan akses internet, dengan ketentuan harus *login* terlebih dahulu menggunakan *user account* yang terdaftar di *LDAP* (*openldap*).
5. Skenario grup, grup pada *LDAP* terdiri dari grup Dosen dan Mahasiswa yang masing-masing grup tersebut memilik hak akses terhadap penggunaan internet yang tidak sama.
6. Skenario otorisasi, *proxy* akan memblok koneksi atau akses ke sejumlah website yang terdapat pada subbab 3.1.1 daftar website pengujian, jika yang mengakses adalah user yang tergabung dalam grup Mahasiswa, sedangkan untuk user yang tergabung dalam grup Dosen, maka bebas mengakses.

Perencanaan Kebutuhan Sistem

Kebutuhan sistem disini meliputi kebutuhan hardware maupun software yang digunakan pada masing-masing komputer sesuai dengan konsep desain jaringan yang telah dibuat. Berikut perencanaan kebutuhan sistem:

1. Desain Jaringan
   Pengujian pada penelitian ini dilakukan sesuai dengan skenario pengujian yang sudah dijelaskan pada subbab sebelumnya, yaitu terdiri dari *client* sebagai komputer pengakses, *proxy server* sebagai filtering akses yang dilakukan oleh komputer *client*, dan sebuah *router* yang digunakan untuk menghubungkan ke jaringan internet.



Gambar 3. Desain Jaringan

Jadi artinya, pada desain jaringan gambar 3 di atas, pengguna akan mengakses internet melalui komputer *client*. Semua permintaan internet pada komputer *client* akan melewati komputer *Proxy* untuk diseleksi atau difilter terlebih dahulu berdasarkan otorisasi yang dilakukan. Jika otorisasi yang diberikan oleh komputer *client* diterima oleh *Proxy* dengan LDAP, maka permintaan tersebut akan diteruskan untuk dapat mengakses internet. Jika tidak, maka pengguna tidak dapat mengakses internet melalui komputer *client*.

2. Kebutuhan Hardware
   Lingkungan pengujian pada penelitian ini dibangun pada dua lingkungan yang berbeda, yaitu menggunakan komputer fisik atau *Host Machine* dan komputer virtual atau *Guest Machine*. Selanjutnya kebutuhan hardware yang dibutuhkan akan dibedakan sesuai dengan kedua lingkungan yang digunakan, yaitu:
   A. Komputer Client 1 (Guest Machine)
      a. Prosesor : Dual Core @2,0 GHz.
      b. Memori : 312 MHz.
   B. Komputer Client 2 (Guest Machine)
      a. Prosesor : Dual Core @2,0 GHz.
      b. Memori : 312 MHz.
   C. Komputer Client 3 (Guest Machine)
      a. Prosesor : Dual Core @2,0 GHz.
      b. Memori : 312 MHz.
   D. Komputer Proxy dan LDAP (Host Machine)
      a. Prosesor : Dual Core @2,0 GHz.
      b. Memori : 2 GHz.
      c. Router ISP : Modem Router Wireless N.
      d. Internet Service Provider : Telkom Speedy.
      e. Hub Surecom 5 Port 10/100M.
      f. Kabel UTP dan socket RJ-45.

3. Kebutuhan Software
   Selain kebutuhan *hardware* yang sudah dijelaskan diatas, ada beberapa *software* yang dibutuhkan, antara lain:
   A. Komputer Client 1 (Guest Machine)
      a. Sistem Operasi : Linux Ubuntu LTS 12.04 Precise Pangolin.
      b. Browser : Mozilla Firefox.
   B. Komputer Client 2 (Guest Machine)
      a. Sistem Operasi : Linux Mint 13 Maya.
      b. Browser : Opera.
   C. Komputer Client 3 (Guest Machine)
      a. Sistem Operasi : Linux Blankon 9 Pattimura.
      b. Browser : Chromium.
   D. Komputer Proxy dan LDAP (Host Machine)
      a. Sistem Operasi : Linux Debian Squezee.
      b. Aplikasi Proxy : squid.
      c. Aplikasi LDAP : slapd.
      d. Aplikasi Firewall : iptable (default).

II. Hasil dan Pembahasan

Komputer *Proxy* dan sekaligus merupakan server *LDAP* adalah komputer yang bertugas untuk meneruskan koneksi internet dari komputer klien ke jaringan internet secara luas. Jadi artinya *Proxy* bisa dianggap sebagai komputer *gateway* yang bertugas menghubungkan jaringan internal dengan jaringan eksternal atau internet. Selain itu, tugas lain *Proxy* disini adalah melakukan filtering atau pembatasan akses situs-situs yang diakses oleh pengguna sesuai dengan otorisasi grup yang dikelola oleh server *LDAP*. Adapun tahapan-tahapan yang dilakukan supaya komputer *Proxy* dan *LDAP* dapat berjalan dan berfungsi seperti konsep yang diinginkan.

Proxy Sebagai Gateway

Berikut merupakan langkah-langkah yang dilakukan penulis dalam melakukan konfigurasi *Proxy* sebagai gateway pada penelitian ini, antara lain:

1. Pemasangan IP Address pada komputer *Proxy Server*:
   ```
   # vim /etc/network/interfaces
   ```
   masukkan perintah dibawah ini untuk IP Address computer *Proxy Server* address xxx.xxx.xxx.xxx netmask 255.255.255.0, keluar dan simpan konfigurasi pemasangan IP Address tersebut dengan menekan kombinasi *keyboard* "ESC", kemudian tekan ":" + wq. Selanjutnya lakukan restart konfigurasi jaringan
   ```
   # /etc/init.d/networking restart.
   ```

2. Aktifkan fungsi *IP Forwarding* untuk memfungsikan komputer *Proxy Server* menjadi komputer gateway
   ```
   # echo 1 >
/proc/sys/net/ipv4/ip_forward.
   ```

3. Aktifkan fungsi *IP Masquerade* untuk memfungsikan komputer gateway mampu meneruskan koneksi internet dari komputer klien ke jaringan internet
   ```
   # iptables –F
   # iptables –t nat –F
   # iptables -t nat -A POSTROUTING
-s xxx.xxx.xxx.0/24 -o eth0 -j
MASQUERADE.
   ```

4. Tambahkan rule *iptables* untuk memblok akses *HTTP* dari *LAN*
   ```
   # iptables -A INPUT -p tcp -s
xxx.xxx.xxx.0/24 -dport 80 -j REJECT
   # iptables -A FORWARD -p tcp -s
xxx.xxx.xxx.0/24 -dport 80 -j REJECT.
   ```

5. Simpan konfigurasi rule *iptables* dan agar *service* berjalan otomatis ketika komputer *gateway log on*
   ```
   # chkconfig iptables on
   ```

Proxy Server

Berikut merupakan langkah-langkah yang dilakukan penulis dalam melakukan konfigurasi *Proxy Server* pada penelitian ini, antara lain:

1. Instalasi squid proxy
   ```
   # apt-get install squid
   ```

2. Secara default konfigurasi *squid* akan memblok semua koneksi dari jaringan *LAN* ke internet. Untuk itu diperlukan konfigurasi ulang file konfigurasi *squid* yang terdapat pada direktori *squid*
   ```
   # vim /etc/squid/squid.conf
   ```
   beberapa direktif atau parameter yang perlu di *setup*, diantaranya adalah:
   a. **visible_hostname:** direktif ini mendefinisikan nama komputer *proxy server* yang digunakan, nilai default parameter ini tidak

diset. Parameter ini dapat diisi dengan nama komputer yang dilengkapi dengan nama domain, `visible_hostname proxy.universitasku.ac.id`.

b. **http_port:** direktif ini menunjukkan nomor *port service squid*, `http_port 8080`.

c. **auth_param:** direktif ini digunakan untuk mendefinisikan parameter-parameter untuk berbagai skema otentikasi yang didukung oleh *squid*, beberapa skema otentikasi yang secara default didukung oleh *squid* di antaranya dapat diketahui dengan melihat isi direktori */usr/lib/squid*. Dalam skenario ini, penulis menggunakan skema otentikasi menggunakan *LDAP*, untuk itu modul yang digunakan adalah *squid_ldap_auth* dan konfigurasi parameter-parameter *squid_ldap_auth* yang digunakan adalah sebagai berikut:

- `auth_param basic program /usr/lib/squid/squid_ldap_auth -b dc=myldap,dc=com -f "cn=%s" -s sub -h localhost.`
- `auth_param basic credentialsttl 2 hours.`
- `auth_param digest children 5.`
- `auth_param basic realm squid proxy-caching web server.`
- `auth_param basic casesensitive off.`

d. **external_acl_type:** direktif ini menentukan *helper* program apa yang digunakan oleh *external acl*. Sesuai skenario, kita menggunakan *helper* program *squid_ldap_group*, sehingga konfigurasinya seperti berikut ini: `external_acl_type ldap_group %LOGIN /usr/lib/squid/squid_ldap_group -b dc=myldap,dc=com -f "(&(cn=%a) (memberUid=%v) (objectClass=posixGroup))" -s sub -h localhost.`

e. **acl:** direktif ini digunakan untuk mendefinisikan *access control list*. Agar sesuai dengan scenario yang sudah dibuat, maka perlu didefinisikan beberapa *acl*, seperti berikut ini:

- `acl haruslogin proxy_auth REQUIRED`
- `acl groupDosen external ldap_group dosen`
- `acl groupMhs external ldap_group mahasiswa`
- `acl dosenweb dstdomain .youtube.com .keepvid.com`

f. **http_access:** direktif ini menunjukkan rule yang akan diterapkan pada suatu *access control list*. Sesuai scenario, rule harus didefinisikan dan dituliskan di bagian bawah seluruh pendefinisian *acl* seperti berikut ini:

- `http_access allow groupDosen haruslogin`
- `http_access deny dosenweb`
- `http_access allow groupMhs haruslogin`

3. Setelah konfigurasi *squid* selesai dilakukan, lakukan pengaktifan *service squid* dan pastikan *service squid* diaktifkan otomatis ketika komputer diaktifkan kembali

```
# /etc/init.d/squid start, # chkconfig squid on
```

LDAP Server

Berikut merupakan langkah-langkah yang dilakukan penulis dalam melakukan konfigurasi *LDAP Server* pada penelitian ini, antara lain:

1. Instalasi LDAP Server

```
# apt-get install slapd
```

2. Mengedit file konfigurasi *LDAP Server*, yaitu file *slapd.conf* yang terletak pada direktori */etc/openldap*. Ada beberapa parameter yang perlu ditentukan nilainya agar sesuai dengan skenario yang sudah dibuat, di antaranya sebagai berikut:

a. `database bdb`
b. `suffix "dc=myldap,dc=com"`

c. `rootdn "cn=admin,dc=myldap,dc=com"`

d. `rootpw` rahasiabanget

3. Mengaktifkan *service ldap* dan memastika agar *ldap* diaktifkan secara otomatis ketika komputer dihidupkan
```
# /etc/init.d/sladp start
# chkconfig slapd start
```

4. Menambahkan entri data ke *LDAP Server*, untuk itu terlebih dahulu buat file baru dengan nama *data.ldif* pada direktori home LDAP.
```
# touch data.ldif
```
buka file *data.ldif* dengan menggunakan aplikasi *text editor*
```
# vim /etc/openldap/data.ldif
```
menambahkan perintah-perintah berikut kedalam file *data.ldif*
```
a. dn: dc=myldap,dc=com
b. objectClass: top
c. objectClass: dcObject
d. objectClass: organization
e. dc: myldap
f. o: Universitasku Tercinta
g. dn: ou=groups,dc=myldap,dc=com
h. ou : groups
i. objectClass: organizationalUnit
j. objectClass: top
```

5. Menambahkan file *data.ldif* ke dalam database direktori *LDAP* dengan cara berikut
```
# ldapadd -x -D "cn=admin,dc=myldap,dc=com" -f data.ldif -W
```

III. Simpulan

Kesimpulan yang dapat diambil dari penelitian perancangan sistem filtering akses website dengan menerapkan otorisasi *LDAP* grup pada *Proxy* Server adalah:

1. Penggunaan internet sebagai fasilitas kampus dapat dimanfaatkan dengan baik sesuai kebutuhan dikarenakan pengguna hanya bisa mengunjungi website-website yang bukan dalam kategori permainan dan hiburan semata.
2. Sistem yang dibangun mampu memfilter atau mengontrol penggunaan internet berdasarkan grup atau atau klasifikasi tertentu sesuai dengan kebutuhan pengguna.

Saran yang diberikan untuk penelitian perancangan sistem filtering akses website dengan menerapkan otorisasi *LDAP* grup pada *Proxy* Server untuk kelanjutan penelitian berikutnya adalah pada penelitian ini, sistem dibangun pada lingkungan Linux sebagai sistem operasinya yang dikenal sangat handal digunakan sebagai sistem operasi dalam jaringan. Namun tidak menutup kemungkinan sistem yang dibangun pada penelitian ini dapat diimplementasikan pada lingkungan Windows, OSX, dan Solaris sebagai sistem operasi jaringannya. Sehingga mampu diketahui performa dari masing-masing lingkungan kerja terhadap operasi dari sistem yang dibangun.

IV. Daftar Pustaka

[1] Fitri, R., & Hidayat, S. (2006). *Integrasi Mekanisme Autentikasi Aplikasi Web Server Dengan Metode LDAP*. Jurnal Teknologi, No. 1, Hal. 61-71.

[2] Munawaroh, S. (2007). *Penyaringan Akses Internet Menggunakan Squid di Linux*, Jurnal Teknologi Informasi Dinamik, Vol. XII, No. 1, Hal. 56-66.

[3] Prasetya, N. I. (2009). Rancang Bangun Honeypot Sebagai Alat Bantu Pendeteksian Serangan Pada Sistem Jaringan Komputer UPN Veteran Jatim. Jurnal Pelita, Vol. 2, No. 2, Hal. 135-148.

[4] Rafiudin, R. (2009). *Squid Koneksi Anti Mogok*. Yogyakarta: Andi Publisher.

[5] Rudy, Riechie, & Gunadi, O. (2009). Integrasi Aplikasi Menggunakan Single Sign On Berbasiskan Lightweight Directory Access Protocol. Jakarta: Universitas Bina Nusantara.