BAB II

TINJAUAN PUSTAKA

2.1. Kerangka Teoritik

2.1.1. *Web Filtering*

Web *Filtering* adalah sebuah sistem yang membuat pengelola jaringan bisa mengontrol penggunaan koneksi Internet yang ada, pengontrolan tersebut bisa berupa pemblokiran terhadap situs-situs tertentu sesuai dengan kebijkan pemilik jaringan. (Cisco, 2013: 1). Situs-situs yang biasanya di blokir adalah situs-situs yang bermuatan negatif atau situs-situs lain yang dianggap dapat mengganggu kondisi jaringan maupun konsentrasi dari pengguna jaringan itu sendiri. Teknologi yang banyak digunakan untuk *web filtering* di tingkat jaringan lokal adalah *Proxy Server*. Sebuah web filtering dapat memperbolehkan atau menolak akses terhadap url atau domain yang spesifik berdasarkan pada :

1. *Whitelist* dan *Blacklist*

   Dalam pengaturan ini nama-nama *website* akan dimasukan kedalam sebuah daftar lalu saat penguna menuju ke suatu *website* melalui *browser* koneksi akan dialihkan ke *proxy server* kemudian dicocokan dengan daftar nama *website* yang ada. Pada pengaturan yang menggunakan *whitelist* jika nama *website* yang dituju pengguna cocok dengan salah satu *website* yang ada dalam daftar *whitelist* maka koneksi pengguna akan diteruskan ke Internet sehingga pengguna dapat mengakses situs tersebut namun jika tidak cocok maka permintaan akan ditolak dan halaman *error* akan diberikan ke *browser* pengguna. Sebaliknya saat pengaturan menggunakan sistem *blacklist* jika

website yang ingin dikunjungi pengguna cocok dengan salah satu *website* dalam daftar *blacklist* maka koneksi tersebut akan ditolak dan halaman *error* akan diberikan kepada pengguna namun jika tidak cocok maka koneksi tersebut akan diteruskan ke Internet sehingga pengguna bisa mengakses situs yang dimaksud.

2. Kategori

   Dalam pengaturan ini *website-website* yang ada di kategorikan ke dalam kategori-kategori tertentu seperti berita, sosial media, pendidikan, dewasa, dan lain lain. Sehingga nantinya seorang pengelola jaringan bisa memilih untuk memblokir atau memperbolehkan pengguna yang menuju ke situs yang sudah dikategorikan sebelumnya.

3. Reputasi

   Dala pengaturan ini setiap *website* diberikan skor sesuai keinginan pengelola jaringan skor inilah yang nantinya akan menjadi acuan diperbolehkan atau tidaknya pengguna untuk mengakses suatu *website*, semisal jika rentang skor dari 0 s.d. 10 maka pengelola jaringan bisa mengatur untuk memblokir akses *website* dengan reputasi kurang dari 6.

2.1.2.1. Jenis-Jenis Filtering

Terdapat beberapa jenis filtering untuk membatasi akses kepada para pengguna (Winanda, 2013), yaitu:

1. *Content-limit* (*or filtered*) ISP

   Pada jenis ini, penyedia layanan Internet yang menawarkan akses ke hanya satu set bagian konten internet. Orang yang berlangganan pelayanan jenis ini tunduk

pada pembatasan yang disediakan. Jenis filter ini berguna untuk melakukan pembatasan segala konten yang diberikan kepada pengguna.

2. *Network-based filtering*

   Pada jenis ini pemfilteran dapat disesuaikan. Penyaringan perangkat lunak termasuk didalamnya ialah fungsi untuk pencegahan kehilangan data. Semua pengguna yang tunduk pada kebijakan akses berdasarkan jaringan ini didefinisikan oleh lembaga.

3. *Search-engine Filters*

   Pada jenis ini, pemfilteran dilakukan terhadap URL atau link yang dirasa tidak pantas. Jika pengguna tahu URL yang sebenarnya merupakan sebuah *website* yang menampilkan eksplisit konten orang dewasa, mereka memiliki kemampuan untuk akses konten tersebut tanpa menggunakan mesin pencari.

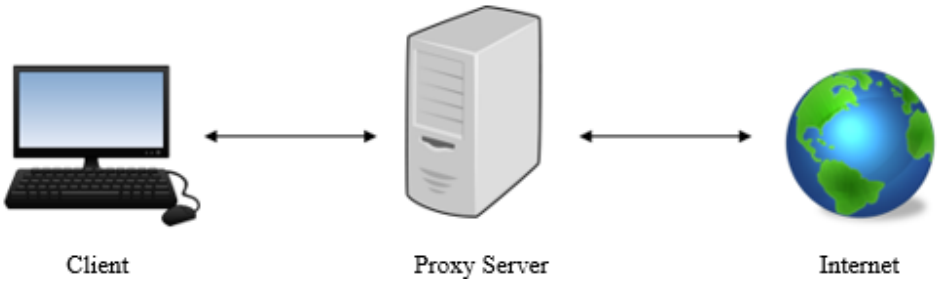
2.1.2.2. Teknologi Dalam Web Filtering

Dalam penelitian ini digunakan proxy server sebagai web filtering pada jaringan Fakultas Teknik Universitas Negeri Jakarta.

Proxy server adalah sistem komputer antara *client* yang meminta dokumen web dengan target *server* (sistem komputer lain) yang melayani dokumen (Kulbir Saini, 2011: 7). Sederhananya, *proxy server* memfasilitasi komunikasi antara *client* dan target *server* tanpa memodifikasi permintaan atau balasan dari permintaan tersebut. Ketika *client* memulai permintaan untuk sumber daya dari target *server*, maka *proxy server* menghubungkan *client* dan mewakili dirinya sebagai sebagai *client* ke targer *server* dan meminta sumber daya atas nama *client*. Jika jawaban diterima,

maka *proxy server* mengembalikan kepada *client* dan memberikan hasil dari permintaan *client* seolah *client*-lah yang telah berkomunikasi dengan target *server*.

Proxy server juga dapat menyaring permintaan berdasarkan aturan dan memungkinkan komunikasi terjadi hanya ketika permintaan tersebut telah divalidasi. Aturan-aturan tersebut berdasarkan *ip address client*, protokol, dokumen web, konten web, dan sebaginya.



Gambar 2.1 Skema *Proxy Sever*

Terkadang, *proxy server* bisa mengubah permintaan atau balasan, atau bahkan bisa menyimpan balasan sumber daya dari *web server* yang sama secara lokal untuk memenuhi permintaan dari *client* jika *client* meminta permintaan yang sama di waktu yang berbeda (teknik *reverse proxy*). *Proxy server* atau *web cache* akan mengecek salinan penyimpanan lokal dari dokumen web yang masih berlaku sebelum menyajikan salinan dari *cache* tersebut.

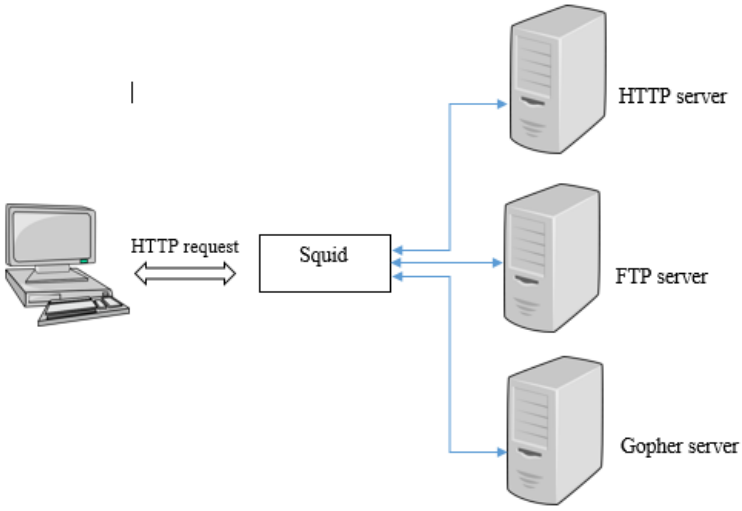
Biasanya *proxy server* digunakan untuk hal-hal sebagai berkut:

4. Mengurangi penggunaan *bandwidth*
5. Mengurangi waktu *load* halaman
6. Pemantauan lalu-lintas pelaporan *user* Internet terhadap *user* individu atau kelompok

7. Meningkatkan privasi *user* dengan tidak menampilkan *user* langsung ke Internet
8. Mendistribusikan beban *web server* yang berbeda untuk mengurangi beban pada *server* tunggal
9. Meningkatkan performa *web server*
10. Menyaring permintaan atau balasan menggunakan sistem pendeteksian *virus/malware*
11. Memuat keseimbangan lalu-lintas jaringan di beberapa koneksi Internet.

2.1.2. *Squid*

Squid adalah sebuah *service* yang digunakan sebagai *proxy server* dan *cache web*. *Squid* memiliki banyak kegunaan, mulai dari mempercepat server web dengan melakukan caching permintaan yang berulang-ulang, DNS, situs web, serta *caching* pencarian di dalam jaringan untuk sekelompok komputer yang menggunakan sumber daya jaringan yang sama hingga membantu keamanan dengan melakukan penyaringan lalu lintas (http://www.squid-cache.org/). Sebagai *proxy*, *squid* adalah media penghubung dalam transaksi web, yang akan menerima permintaan dari *client*, memproses permintaan tersebut, lalu meneruskannya ke alamat *server* yang dituju. Namun permintaan tersebut bisa saja direkam kedalam sebuah *log file*, ditolak, atau bahkan dimodifikasi sebelum diteruskan. Saat sebagai *cache*, *squid* menyimpan konten web dari permintaan *client* agar nanti bisa digunakan kembali, artinya saat ada *user* yang melakukan permintaan yang sudah ada dalam *cache* maka *squid* akan memberikan data tersebut dan tidak akan melakukan permintaan kembali ke alamat *server* tujuan yang sama.



Gambar 2.2 Skema *Squid Proxy Sever*

Kita bisa menonaktifkan fitur *cache* jika kita mau. Namun, penggunaan *squid* sebagai *proxy* tak bisa dinonaktifkan. Berikut adalah fitur dari squid :

a. Menggunakan *bandwidth* lebih sedikit saat *surfing* di Internet
b. Mengurangi waktu untuk me-*load* sebuah halaman web
c. Melindungi *host* pada jaringan lokal
d. Mengumpulkan statistik tentang lalu lintas Internet di sebuah jaringan
e. Mencegah *user* mengunjungi web-web tertentu
f. Memastikan hanya *user* yang memiliki hak yang dapat mengakses Internet
g. Memperkuat pengamanan privasi *user*
h. Mengurangi proses *load* pada web *server*
i. Mengkonversi permintaan HTTPS menjadi HTTP

2.1.2.1. *Caching Document*

Cache adalah sebuah tempat penyimpanan sementara berisi data salinan dari data lainnya. Pada *processor* terdapat *cache memory* yang berisi salinan instruksi atau data dari memori utama yang akan diproses oleh *processor*. Namun dalam *proxy server* juga terdapat sebuah *cache*, *cache* disini berfungsi sebagai penyimpan data laman web yang telah dikunjungi oleh *user* dalam suatu jaringan yang dinaungi *proxy server*. Kemudian, jika ada *user* lain dalam jaringan tersebut yang memberikan permintaan terhadap laman web yang sama maka *proxy server* akan memberikan laman web yang telah tersimpan dalam *cache* tanpa harus membuat permintaan lagi ke Internet sehingga dapat mempercepat akses laman web bagi *user* dan menghemat *bandwidth* Internet pada jaringan.

Squid melakukan metode *caching document*, sehingga dokumen-dokumen tersebut dapat disajikan jika *user* melakukan *request* (Saini, 2011:46). Squid menggunakan memori utama (RAM) dan *harddisk* untuk menyimpan dan melakukan metode *caching document* ini. *Caching* merupakan proses yang kompleks, akan tetapi *squid* bisa menangani hal tersebut dengan mulus dan memperlihatkannya menggunakan *squid.conf* sehingga *user* dapat mengendalikanmya dan mengetahui seberapa banyak dan apa saja yang sebaiknya diprioritaskan ketika *caching* sedang pada puncak tertinggi.

2.1.2.2. *Access Control List*

Access Control List (ACL) adalah sebuah metode yang berfungsi untuk melakukan *packet filtering* (Saini, 2011:39). Dengan menerapkan ACL, maka seorang *network administrator* dapat menentukan paket-paket yang dapat diakses

melalui *proxy server*. Dalam *squid*, ACL akan dikombinasikan penggunaannya menggunakan perintah *http_access* untuk menentukan paket-paket tersebut kan ditolak atau diterima masuk kedalam sistem. Sistem *acces control list* adalah dasar dalam pembuatan *filtering* pada aplikasi *squid*.

2.1.3. *Iptables*

Iptables merupakan program yang digunakan untuk memasukkan dan menghapus isi tabel filter paket, termasuk memelihara dan memeriksa tabel filter paket pada kernel (inti sistem operasi). *Linux* merupakan OS (*Operating System*) yang sudah memiliki kemampuan filter paket di dalam kernelnya. Dengan menggunakan *iptables*, apapun yang tertulis di dalam tabel akan hilang jika melakukan *reboot system*.

Secara sederhana iptables berfungsi sebagai firewall yang ada pada windows, hanya saja iptables digunakan pada linux (3D4 Telkom 2004, 2014:83).

Gambar 2.3 Skema *Firewall*

Iptables sebagai *firewall* memiliki beberapa fungsi, yaitu:

1. *Service Control* (kendali terhadap layanan)

   Teknik ini berdasarkan pada tipe-tipe layanan yang digunakan di Internet dan menentukan boleh atau tidaknya layanan tersebut diakses oleh pengguna. Biasanya *firewall* akan mengecek ip address dan port yang digunakan baik pada protokol TCP dan UDP, bahkan bisa dilengkapi perangkat lunak untuk

proxy yang akan menerima dan menterjemahkan setiap *request* dari suatu layanan sebelum mengizinkannya. Bahkan bisa jadi perangkat lunak pada *server* itu sendiri, seperti layanan web ataupun *mail*.

2. *Direction Control* (kendali terhadap arah)

   Teknik ini berdasarkan pada arah dari berbagai *request* terhadap layanan yang akan dikenali dan diizinkan melewati *firewall*. Arah yang digunakan seperti INPUT dan PREROUTING yang mendefinisikan koneksi atau paket yang akan masuk kedalam sistem, OUTPUT dan POSTROUTING yang menandakan paket atau koneksi yang akan mninggalkan sistem serta FORWARD yang berada diantara kedua arah tersebut.

3. *User Control* (kendali terhadap pengguna)

   Teknik ini berdasarkan layanan yang dapat dijalankan oleh pengguna layanan layanan tersebut eperti HTTP, SMTP, SFTP, SSH, VPN, SeSH, ICMP dan layanan-layanan lainnya, iptables memberikan pengelola jaringan kuasa pengguna mana yang bisa menggunakan layanan apa saja. Sehingga pengelola bisa mencegah koneksi yang tak sesuai dengan kebijakan yang ada pada jaringannya.

4. *Behavior Control* (kendali terhadap perlakuan)

   Teknik ini berdasarkan pada seberapa banyak suatu layanan itu digunakan. Contohnya pengelola jaringan bisa saja membuat firewall dapat memfilter email yang masuk ke jaringannya untuk menanggulangi / mencegah spam.

Iptables mengizinkan *user* untuk mengontrol sepenuhnya jaringan melalui paket

ip dengan sistem *linux* yang diimplementasikan pada kernel *Linux* (Syarif, 2008). Sebuah kebijakan atau *Policy* dapat dibuat dengan *iptables* sebagai polisi lalulintas jaringan. Sebuah *policy* pada *iptables* dibuat berdasarkan sekumpulan peraturan yang diberikan pada kernel untuk mengatur setiap paket yang datang.

Pada iptables ada istilah yang disebut dengan Ipchain yang merupakan daftar aturan bawaan dalam Iptables. Ketiga chain tersebut adalah :

1. INPUT

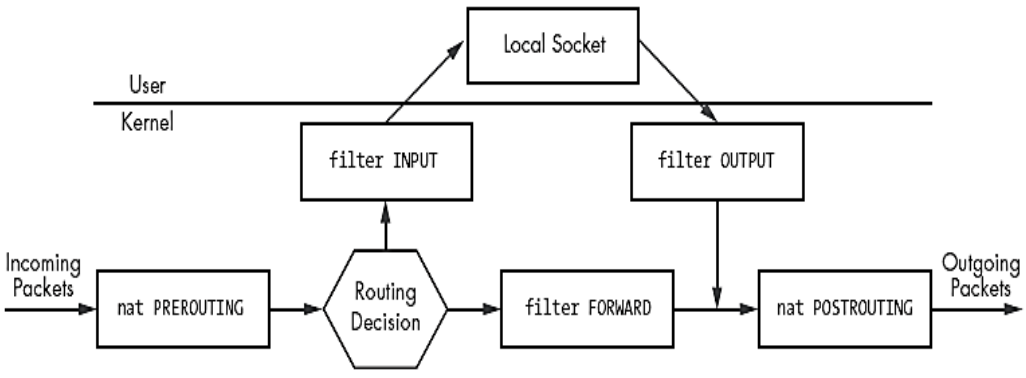
   Rantai INPUT dilalui oleh paket tujuan pada sistem linux setelah melalui perhitungan (paket yang ditujukan untuk lokal soket).

2. OUTPUT

   Rantai Output adalah paket yang dihasilkan dari proses sistem linux itu sendiri.

3. FORWARD

   Rantai FORWARD mengatur paket yang diarahkan melalui sistem linux, yaitu ketika iptables/firewall yang digunakan untuk menghubungkan jaringan satu dengan yang lain dan untuk paket antara dua jaringan harus melalui firewall.



Gambar 2.4 Diagram Paket *Iptables*

Pada saat sebuah paket sampai pada salah satu persegipanjang diantara IPchains, maka disitulah terjadi proses penyaringan. Rantai akan memutuskan nasib paket tersebut. Apabila keputusannnya adalah DROP, maka paket tersebut akan di-drop. Tetapi jika rantai memutuskan untuk ACCEPT, maka paket akan dilewatkan melalui diagram tersebut.

Sebuah rantai adalah aturan-aturan yang telah ditentukan. Setiap aturan menyatakan "jika paket memiliki informasi awal (header) seperti ini, maka inilah yang harus dilakukan terhadap paket". Jika aturan tersebut tidak sesuai dengan paket, maka aturan berikutnya akan memproses paket tersebut. Apabila sampai aturan terakhir yang ada, paket tersebut belum memenuhi salah satu aturan, maka kernel akan melihat kebijakan bawaan (default) untuk memutuskan apa yang harus dilakukan kepada paket tersebut. Ada dua kebijakan bawaan yaitu default DROP dan default ACCEPT.

Agar isi tabel tidak hilang dan dapat diarahkan ke suatu tempat penyimpanan tertentu misalnya file, maka digunakan program iptable-save. Selain iptable-save, ada juga program iptables-restore yang digunakan untuk memulihkan tabel dari suatu tempat penyimpanan tertentu termasuk file. Dari program tersebut, iptables lah yang bertindak sebagai program intinya.

Selain rantai yang telah disebutkan di atas ada juga rantai seperti RETURN, MIRROR atau REJECT, RETURN sendiri berfungsi untuk mengembalikan paket ke superset rantai di atasnya dan masuk ke baris aturan berikutnya, jika pada subrantai. Tapi jika pada rantai utama seperti INPUT, maka paket dikembalikan kepada kebijakan *default*. Pada MIRROR iptables akan membalik *source address* dan *destination address* target ini bekerja pada rantai INPUT, FORWARD dan

PREROUTING atau rantai buatan yang melalui chain tersebut. Sedangkan REJECT berfungsi untuk menolak paket maupun menelusuri paket lebih jauh walau terlihat mirip dengan DROP tapi sebenarnya keduanya berbeda perbedaan itu terdapa pada apa yang diperikan pada *user* yang melakukan permintaan jika menggunakan DROP maka *user* tidak akan mengetahui apa-apa yang diterima hanya berupa pesan *request time out* saat melakukan ping namun jika memakai REJECT maka *user* akan menerima pesan penolakan berupa *destination host unreachable*.

2.1.4. *Debian Linux*

Debian adalah sebuah *GNU/Linux distribution* yang *non-commercial distribution*. Walaupun *non-commercial*, debian termasuk sistem operasi yang lengkap, mulai dari perangkat lunak, sistem untuk instalasi dan manajemen, semua berdasarkan kernel linux dan perangkat lunak yang bebas (Hertzog, 2013: 2). Bukan hanya sistem operasi yang gratis, kualitas debian akan dikembangkan dengan perawatan yang besar agar menjadi kernel linux yang layak, yang akan membawa debian menjadi cukup kredibel untuk bersaing dengan ditribusi komersial utama.

Debian memiliki sistem pemaketan sendiri (*.deb) yang akan dipaketkan ke dalam distro *debian* dan harus menuruti *Debian Free Software Guidelines* Selain itu, paket-paket tersebut harus melalui 3 fase penyeleksian paket yakni *stabeling, testing,* dan *unstable*. Untuk melewati ketiga fase tersebut, sampai mendapat predikat *software stable* biasanya memakan waktu yang cukup lama. Isu yang yang banyak didengungkan orang terhadap distro ini adalah masalah kestabilan, sehingga

tidak mengherankan jika distro ini banyak digunakan sebagai dasar pembuatan distro lain.

2.1.5. Penelitian Yang Relevan

Adapun penelitian mengenai *web filtering* yang berkaitan dengan penggunaan *squid* sebagai *proxy server* adalah sebagai berikut:

1. Penelitian Sandy Arjuni yang berjudul *Perancangan Dan Implementasi Proxy Server Dan Manajemen Bandwidth Menggunakan Linux Ubuntu Server (Studi Kasus di Kantor Manajemen PT.Wisma Bumiputera Bandung).* Penelitian ini menggunakan squid yang digunakan sebagai proxy untuk pemblokiran situs dengan pembatasan akses hanya dengan domain. Sistem ini juga melakukan manajemen bandwidth secara umum kepada *client* yang menggunakannya serta mengimplementasikan metode autentikasi kepada *client*, sehingga untuk kedepannya dapat diterapkan sistem *log in* ketika *client* akan menggunakan Internet. Sistem sangat efektif dalam melakukan pembatasan akses dengan melakukan pemblokiran situs melalui proxy squid. Dari hasil pengujian yang dilakukan prosentase keberhasilannya adalah 100%.
2. Penelitian Dendy Trisna Pasuruan yang berjudul *Implementasi Squid Proxy Server Pada PC Router Berbasis Linux Mandriva di UPT SDN Mandaranrejo.* Penelitian ini digunakan untuk untuk memblokir situs porno dan manajemen bandwidth. Pengujian dilakukan dengan cara browsing ke Internet untuk menguji transparan proxy dan filterisasi situs porno, mengukur bandwidth klien dengan bandwidth meter untuk menguji manajemen bandwidth. Dari hasil pengujian alat diperoleh data sebagai berikut: klien tidak dapat mengakses

Internet tanpa melalui proxy, bandwidth yang diterima oleh masing-masing klien sebesar 6 KB/s, klien relatif tidak dapat mengakses situs porno.

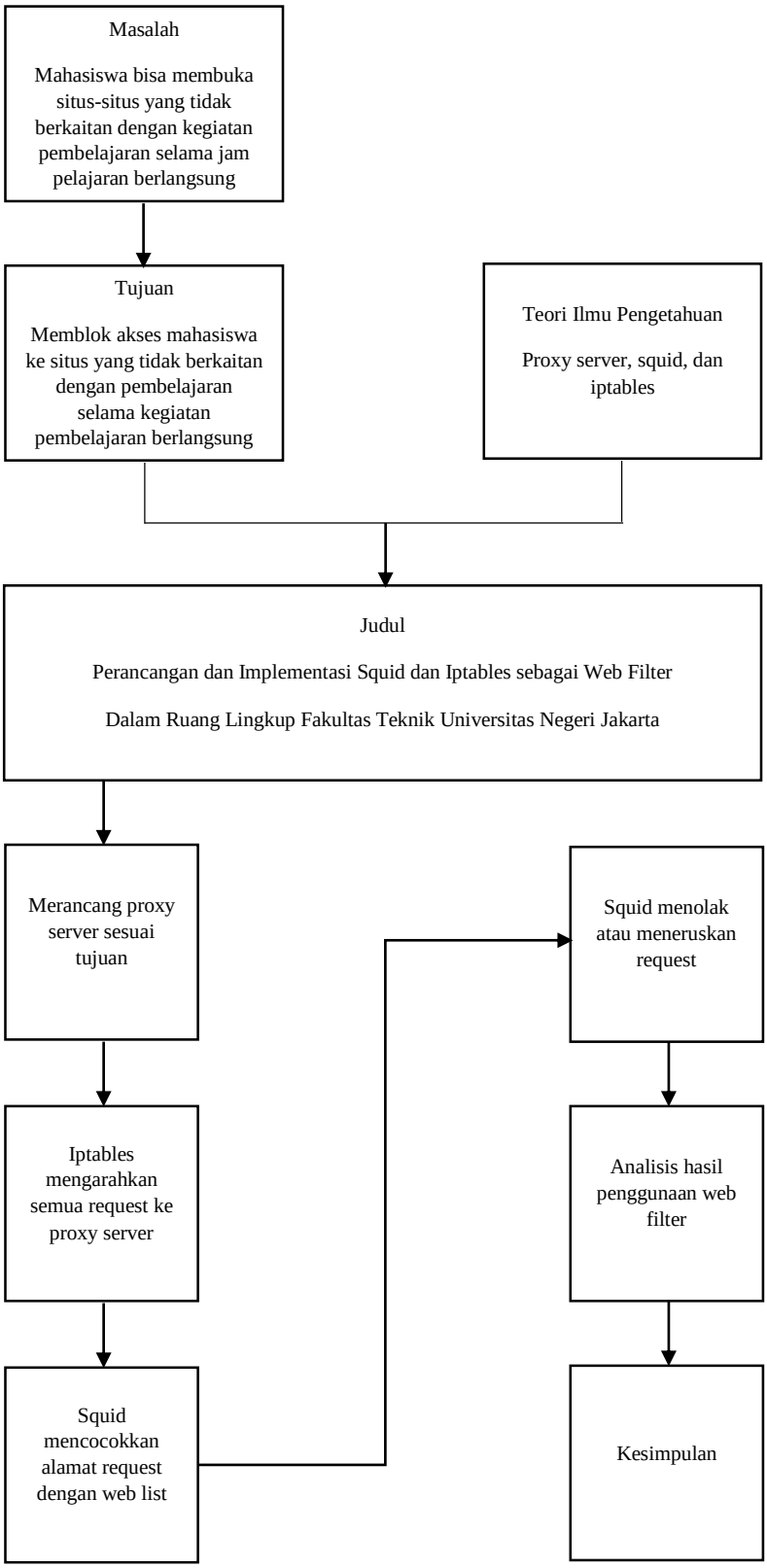
3. Penelitian Mokhamad Iklil Mustofa dkk yang berjudul *Implementasi Squid Proxy Untuk Mengontrol Penggunaan Internet Di Magistra Utama Semarang.* Berdasarkan sistem ini, tercipta penggunaan Internet yang terjadwal dengan baik dan aman karena situs-situs terlarang dapat diblokir, serta situs *video streaming* juga dapat diblokir karena mempengaruhi kecepatan akses Internet dan dikatakan hemat karena mengurangi kuota bandwidth yang diberikan oleh *ISP (Internet Services Provider).* Selain itu dikatakan terjadwal karena jam akses Internet untuk IP Address klien, dapat dibatasi dari jam 12.00 sampai dengan 15.00, sedangkan untuk manajer dan staf lebih bebaskan waktunya.
4. Penelitian Mohamad Lukman Pranoto yang berjudul *Implementasi VPN Dan Proxy Server Menggunakan FreeBSD Pada SMA Islam Hidayatullah*. Penelitian ini menggunakan VPN dan *Proxy server* pada *FreeBSD* yang dapat mempermudah admin dalam melakukan *accounting* dan manajemen *database client*. Sistem ini membatasi alamat web dan konten yang bersifat tidak mendidik dan mengandung unsur negatif (contoh situs pornografi) dapat difilter dan tidak bisa diakses oleh *client*. Kecepatan proses akses ke web akan lebih cepat karena konten yang sudah pernah dibuka (*history content*) akan tersimpan dalam *cache proxy* sehingga tidak perlu mengakses ulang dari *global network.*

2.2. Kerangka Berpikir

Internet kini sudah menjadi bagian penting dari kehidupan yang seakan tidak bisa dipisahkan lagi dari kehidupan sehari-hari, beragam situs dapat dibuat dan diakses oleh seluruh orang lewat Internet. Walau begitu tidak semua situs memilliki muatan yang positif, ada beberapa situs yang memiliki muatan negatif dan mungkin tak sesuai dengan keadaan *user* yang membuka situs tersebut.

Setelah melihat keadaan tersebut, penulis membangun sebuah *proxy server* dimana *proxy server* tersebut dibantu dengan *iptables* yang akan bertindak untuk mengambil seluruh permintaan *user* dan meneruskannya ke Internet, lalu *proxy server* ini juga akan menyimpan berbagai laman situs yang pernah dikunjungi oleh *user* sebelumnya hingga nanti. Jika ada *user* yang membuat permintaan yang sama, maka *proxy server* akan memberikan laman situs yang telah ia simpan tanpa harus mengakses Internet lagi. Hal ini ditunjang oleh beberapa penelitian terdahulu yang telah dilakukan dan ditunjang pula oleh teori yang diambil dari beberapa buku terkait web filtering menggunakan squid dan iptables.

Namun yang menjadi perhatian penulis adalah kemampuan *proxy server* untuk meneruskan atau menolak permintaan dari *user* dengan memanfaatkan *squid* sebagai *proxy server* dan *web cache* serta pemanfaatan *iptables* untuk membatasi web dengan protokol https . Penulis juga akan memanfaatkannya untuk memblok akses *user* ke situs-situs yang bermuatan negatif dengan cara membuat list alamat situs yang bermuatan negatif lalu mencocokannya dengan alamat yang dimasukan *user*. Jika sama, permintaan akan ditolak oleh *squid* dan *iptables.* Dan jika tidak sama, permintaan akan diteruskan oleh *squid* atau mengambil data yang telah tersimpan di direktori *proxy server*.



Gambar 2.5 Rancangan Teoritis