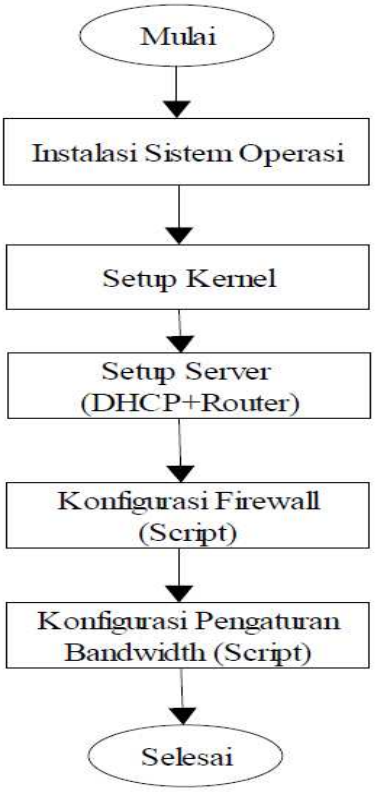
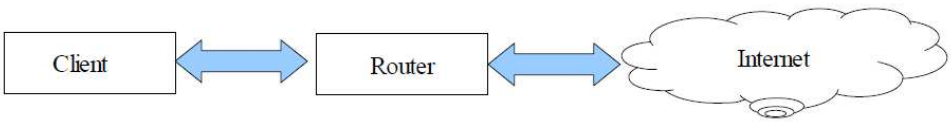
BAB III
PERANCANGAN SISTEM

Bab ini akan memaparkan bagaimana sistem dirancang sedemikian rupa sehingga menjadi sebuah PC *router* yang mampu membagi *bandwidth* dengan sama rata dan adil, dapat dilihat dari diagram dibawah ini.



Gambar 3.1 *Diagram Perancangan Secara Umum.*

Lalu perancangan jaringan secara umum dan alur paket yang di kirim maupun yang di terima dari *client* ke *server*, atau sebaliknya, dapat di lihat pada diagram di bawah ini.



Gambar 3.2 *Diagram Perancangan Jaringan Secara Umum.*

3.1 Instalasi Sistem

3.1.1 Perangkat Keras yang Digunakan

Tabel 3.1 *Perangkat Keras yang digunakan.*

Hardware	Jumlah	Spesifikasi
Server	1	a. Asus b. Ram 1.5 GB c. Harddisk 40 GB d. 100 Mbps Nic Via e. 100 Mbps Nic Realtak
Client	3	PC atau Notebook
Switch	1 Buah	Switch 100 Mbps D-Link
Kabel UTP	4 Meter	Belden Australia

3.1.2 Perangkat Lunak yang Digunakan

- Gentoo GNU/linux 2008 sebagai sistem operasi GNU/linux.
- FreeBSD 7.1 sebagai sistem operasi FreeBSD.
- Iproute sebagai NAT (*Network Address Translation*) di GNU/linux.
- Ipnat sebagi NAT di FreeBSD.
- Dnsmasq sebagai *server* ip (dhcp) di GNU/linux.
- Dhcpcd sebagai *server* ip (dhcp) di FreeBSD.
- IPerf merupakan perangkat lunak yang digunakan untuk mengukur performansi jaringan.
- ping, yaitu perangkat lunak yang digunakan untuk mengukur *response time* dan *paket loss.*

3.1.3 Instalasi Sistem Operasi Gentoo GNU/linux

Gentoo sebagai sistem operasi berbasis GNU/linux yang dijadikan penelitian dalam tugas akhir ini adalah gentoo release 2008 (proses instalasi ada di halaman lampiran B).

3.1.4 Instalasi Sistem Operasi FreeBSD

FreeBSD sebagai sistem operasi pengembangan dari unix yang digunakan dalam penelitian tugas akhir ini adalah FreeBSD release 7.1 (proses instalasi ada pada halaman lapiran B).

3.2 Konfigurasi, Kompilasi dan Instalasi Kernel

3.2.1 Menggunakan Kernel GNU/linux

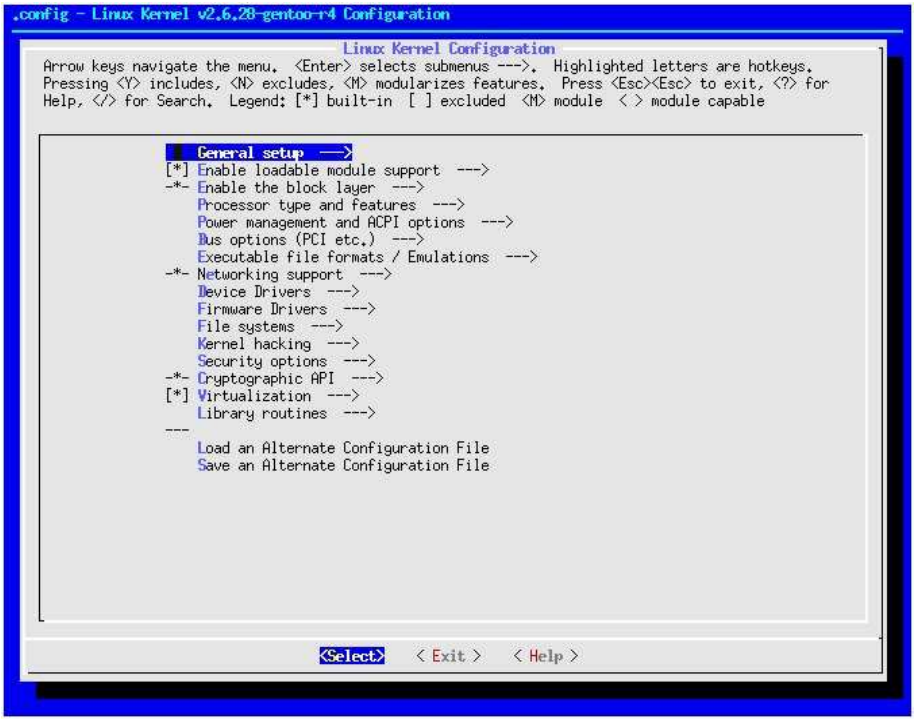
Agar Semua kebutuhan yang kita inginkan dalam sebuah sistem operasi yang akan dijadikan *server* maka modul-modul yang tidak diperlukan dalam *server* sebaiknya kita abaikan saja, karena dengan demikian kernel akan menjadi lebih optimal ketika melakukan tugas-tugasnya.

- Konfigurasi Kernel GNU/linux

Pertama kali setelah melakukan instalasi maka masuk kedalam konfigurasi kernel, disini saya menggunakan kernel 2.6.30-Gentoo-r6. Untuk melakukan konfigurasi kernel di GNU/linux berada pada directory `/usr/src/GNU/linux`, maka kita harus pindah kedalam *direktori* tersebut

```
cd /usr/src/GNU/linux
/usr/src/GNU/linux# make menuconfig
```

Setelah perintah di atas di *enter* maka akan nampak sebuah aturan GUI untuk mengkonfigurasi kernel pada lingkungan GNU/linux seperti gambar dibawah ini :



Gambar 3.3 *Tampilan GUI Konfigurasi Kernel GNU/linux.*

Setelah masuk ke dalam konfigurasi kita pilih modul-modul apa saja yang dibutuhkan pada sistem *server* kita. Karena kita membutuhkan untuk *server* seperti router, dhcp *server*, iptables, teknik antrian maka kita terfokus pada bagian *networking*. Berikut adalah modul-modul yang di aktifkan:

```
Networking  --->
  Networking options  --->
    [*] Network packet filtering (replaces ipchains)  --->
      IP: Netfilter Configuration  --->
        <*> Connection tracking (required for masq/NAT)
        <*> Userspace queueing via NETLINK
        <*> IP tables support (required for filtering/masq/NAT)
        <*>   limit match support
        <*>   IP range match support
        <*>   MAC address match support
        <*>   Packet type match support
        <*>   netfilter MARK match support
        <*>   Multiple port match support
        <*>   TOS match support
        <*>   recent match support
        <*>   ECN match support
        <*>   DSCP match support
        <*>   AH/ESP match support
        <*>   LENGTH match support
        <*>   TTL match support
        <*>   tcpmss match support
        <*>   Helper match support
        <*>   Connection state match support
        <*>   Connection tracking match support
        <*>   Owner match support
```

```
        <*>   Packet filtering
        <*>     REJECT target support
        <*>   LOG target support
        <*>   ULOG target support
        <*>   TCPMSS target support
        <*>   Full NAT
        <*>     MASQUERADE target support
        <*>     REDIRECT target support
        <*>     NETMAP target support
        <*>     SAME target support
        <*>   Packet mangling
        <*>     TOS target support
        <*>     ECN target support
        <*>     DSCP target support
        <*>     MARK target support
        <*>     CLASSIFY target support
        <M>   raw table support (required for NOTRACK/TRACE)
        <M>     NOTRACK target support
        <*> ARP tables support
        <*>   ARP packet filtering
        <*>   ARP payload mangling
    QoS and/or fair queueing  --->
      <M>   HTB packet scheduler
      <M>   SFQ queue
      [*]   QoS support
      [*]     Rate estimator
      [*]   Packet classifier API
    <M> Firewall based classifier
    [*] Traffic policing (needed for in/egress)
```

Gambar 3.4 *Konfigurasi kernel GNU/linux.*

- Kompilasi dan Instalasi Kernel

Setelah melakukan konfigurasi seperti yang dilakukan diatas maka waktunya untuk mengkompilasi dan instalasi, bagian ini sangat penting karena modul-modul yang telah diaktifkan akan di simpan sesuai pada tempatnya misalnya untuk kartu jaringan maka akan modulnya akan disimpan pada /etc/network , dll.

Untuk melakukan kompilasi bisa dilakukan secara terpisah atau bersamaan dengan instalasi kernel. Dengan mengetikan perintah berikut

```
/usr/src/GNU/linux#make && make modules_install
```

- `Make` yang pertama adalah proses dari kompilasi.
- `Make modules_install` adalah proses dari instalasi kernel.

3.2.2 Menggunakan Kernel FreeBSD

Sama halnya dengan kernel GNU/linux, konfigurasi dan instalasi kernel menjadi bagian yang penting dalam system FreeBSD dalam menjalankan sebuah *server.*

- Konfigurasi Kernel FreeBSD

Setelah proses instalasi FreeBSD selesai, saatnya untuk konfigurasi kernel, Untuk melakukan konfigurasi kernel pada FreeBSD, kita harus pindah *directory* terlebih dahulu ke /usr/src/sys/i386/conf, dan konfigurasi kernel berada pada *file GENERIC,* jika akan menambah atau mengurangkan konfigurasi kernel salin terlebih dahalu *file* GENERIC menjadi nama *file* yang lain misalnya ROUTER.

```
cd /usr/src/sys/i386/conf
cp GENERIC ROUTER
nano –w ROUTER
```

Setelah perintah terakhir dilakukan maka akan keluar tampilan seperti pada gambar 3.5.

```
machine         i386
cpu             I686_CPU
ident           TES

# To statically compile in device wiring instead of /boot/device.hints
#hints          "GENERIC.hints"         # Default places to look for
devices.

options         SCHED_4BSD              # 4BSD scheduler
options         INET                    # InterNETworking
options         INET6                   # IPv6 communications protocols
options         FFS                     # Berkeley Fast Filesystem
options         SOFTUPDATES             # Enable FFS soft updates support
options         UFS_ACL                 # Support for access control lists
options         UFS_DIRHASH             # Improve performance on big directories
options         MD_ROOT                 # MD is a potential root device
options         NFSCLIENT               # Network Filesystem Client
options         NFSSERVER               # Network Filesystem Server
options         NFS_ROOT                # NFS usable as /, requires NFSCLIENT
options         MSDOSFS                 # MSDOS Filesystem
options         CD9660                  # ISO 9660 Filesystem
options         PROCFS                  # Process filesystem (requires PSEUDOFS)
options         PSEUDOFS                # Pseudo-filesystem framework
options         GEOM_GPT                # GUID Partition Tables.
options         COMPAT_43               # Compatible with BSD 4.3 [KEEP THIS!]
options         COMPAT_FREEBSD4         # Compatible with FreeBSD4
options         SCSI_DELAY=15000        # Delay (in ms) before probing SCSI
options         KTRACE                  # ktrace(1) support
options         SYSVSHM                 # SYSV-style shared memory
options         SYSVMSG                 # SYSV-style message queues
options         SYSVSEM                 # SYSV-style semaphores
options         _KPOSIX_PRIORITY_SCHEDULING # POSIX P1003_1B real-time
extensions
options         KBD_INSTALL_CDEV  # install a CDEV entry in /dev
options         AHC_REG_PRETTY_PRINT    # Print register bitfields in debug
                                        # output.  Adds ~128k to driver.
options         AHD_REG_PRETTY_PRINT    # Print register bitfields in debug
                                        # output.  Adds ~215k to driver.
options         ADAPTIVE_GIANT          # Giant mutex is adaptive.

device              apic            # I/O APIC

# Bus support.  Do not remove isa, even if you have no isa slots
device              isa
```

Gambar 3.5 *Konfigurasi kernel FreeBSD.*

karena yang dibutuhkan pada kernel adalah sebagai *server* yang dapat mengatur *bandwidth*, maka aktifkan yang dibutuhkan saja dengan cara menghilagkan tanda kress (#), sebagai salah satu cara dalam mempercepat mengakses sistem operasi.

- Kompilasi dan Instalasi kernel

Setelah memberikan konfigurasi kernel pada *file* yang telah dimodifikasi, kernel siap untuk dikompilasi, langkah-langkah dalam kompilasi dan instalasi kernel dalam lingkungan FreeBSD, sedikit sama dengan lingkungan GNU/linux namun dalam FreeBSD kernel harus dikonfigurasi terlebih dahulu agar sistem dapat mengenali *file* yang telah dimofidikasi.

```
config ROUTER
cd . ./compile/ROUTER
make depand
make && make install
```

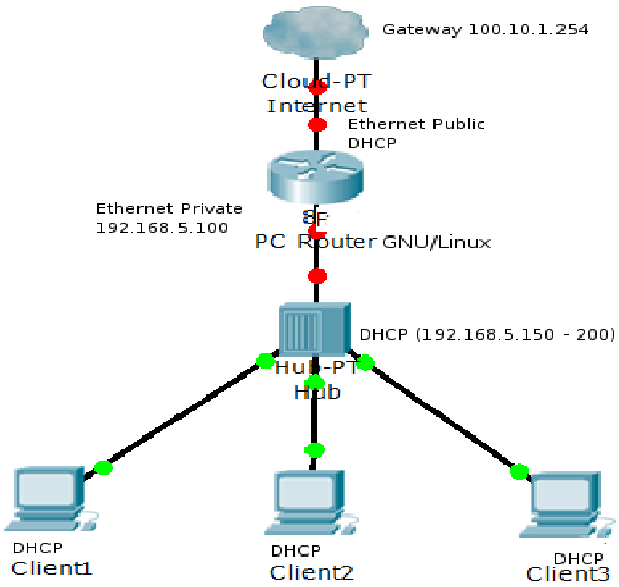
3.3 Konfigurasi Server

Setelah mengaktifkan modul-modul kernel yang dibutuhkan dalam pembuatan pc sebagai *server*, maka selanjutnya konfigurasi pc agar memberikan sumber daya ke *client*, dalam hal ini lebih diutamakan pada bagian *router* dan DHCP *server.*

3.3.1 Menggunakan Kernel GNU/linux

Sebelum memulai mengkonfigurasi PC sebagai *router* yang dibutuhkan adalah minimal memiliki 2 NIC (*Network Interface Card*) dan beberapa konfigurasi pada jaringan internet yang digunakan. Untuk lebih mudah penulis definisikan terlebih dahulu NIC yang digunakan.

- NIC 0 / *Ethernet* 0 / eth0 : adalah NIC yang terhubung ke internet
- NIC 1 / *Ethernet* 1 / eth1 ; adalah NIC yang terhubung ke LAN
- Topologi yang digunakan adalah topologi *star* (lihat gambar 3.6)



Gambar 3.6 *Topologi yang digunakan pada kernel GNU/linux.*

- Eth0 diberikan IP Address : DHCP
- Eth1 diberikan IP Address : 192.168.5.100
- Dengan *default Gateway* (ADSL) : 100.10.1.254

Router yang digunakan adalah *router* dinamis yaitu *router* yang dikonfigurasi secara otomatis dalam penerimaan atau pengiriman IP dari ISP. Setelah itu konfigurasi terlebih dahulu kartu jaringan yang digunakan agar sesuai dengan yang telah di definisikan dan sesuai dengan topologi yang digunakan.

Untuk konfigurasi kartu jaringan (NIC) pada distibusi GNU/linux Gentoo berada pada *file* `/etc/conf.d/net`. Berikut adalah hasil konfigurasi dari *file* Net.

```
dns_domain_lo="tk.unikom.ac.id"
config_eth0=("dhcp")
routes_eth0=("default gw 100.10.1.254")

config_eth1=("192.168.5.100 broadcast 192.168.5.255 netmask
255.255.255.0")
```

Setelah itu dilanjutkan untuk mengkonfigurasi DHCP *server* agar *client* dapat dengan mudah dalam mendapat akses ke jaringan luar. Di sini saya menggunakan aplikasi tambahan dalam DHCP *server*, yaitu adalah dnsmasq, dnsmasq bekerja selain memberikan ip ke *client* dnsmasq juga memberikan DNS atau sebagai DNS *server* juga. ke *client* yang didapat dari PC *router*. Untuk itu maka perlu di install dnsmasq

```
emerge dnsmasq
```

```
vim /etc/dnsmasq.conf
```

Setelah install selesai konfigurasi *file* dnsmasq yang berada pada /etc/dnsmasq.conf berikut adalah konfigurasinya.

```
dhcp-range=192.168.0.150,192.168.0.250,72h
interface=eth0
```

Setelah semua *client* dalam LAN dapat berkomunikasi dengan *router* lokal dan sesama *client* yang lain, maka dibutuhkan konfigurasi NAT (*Network Address Translation*) agar semua *client* dapat mengkases jaringan luar. Berikut adalah konfigurasi NAT dengan bantuan iptables :

Pertama hapus semua aturan

```
iptables -F
iptables -t nat -F
```

Setup aturan-aturan *default* untuk menangani *traffic* yang tidak cocok

```
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
```

Definisi kartu jaringan

```
export LAN=eth0
export WAN=eth1
```

Kemudian kunci layanan kita agar hanya dapat digunakan di LAN

```
iptables -I INPUT 1 -i ${LAN} -j ACCEPT
iptables -I INPUT 1 -i lo -j ACCEPT
iptables -A INPUT -p UDP --dport bootps -i ! ${LAN} -j
REJECT
iptables -A INPUT -p UDP --dport domain -i ! ${LAN} -j
REJECT
```

(Opsional) Ijinkan akses ke *server* ssh kita dari WAN

```
iptables -A INPUT -p TCP --dport ssh -i ${WAN} -j
ACCEPT
```

Tolak paket-paket TCP/UDP ke *port-port* khusus

```
iptables -A INPUT -p TCP -i ! ${LAN} -d 0/0 --dport
0:1023 -j DROP
iptables -A INPUT -p UDP -i ! ${LAN} -d 0/0 --dport
0:1023 -j DROP
```

Terakhir tambahkan aturan untuk NAT

```
iptables -I FORWARD -i ${LAN} -d 192.168.0.0/255.255.0.0 -j
DROP
iptables -A FORWARD -i ${LAN} -s 192.168.0.0/255.255.0.0 -j
ACCEPT
iptables -A FORWARD -i ${WAN} -d 192.168.0.0/255.255.0.0 -j
ACCEPT
iptables -t nat -A POSTROUTING -o ${WAN} -j MASQUERADE
```

Setelah membuat aturan NAT maka setting *file sysctl* yang berada pada *directory* `/etc/sysctl.conf` lalu ubah atau tambahkan konfigurasi menjadi seperti di bawah ini.

```
net.ipv4.ip_forward = 1
net.ipv4.conf.default.rp_filter = 1
```

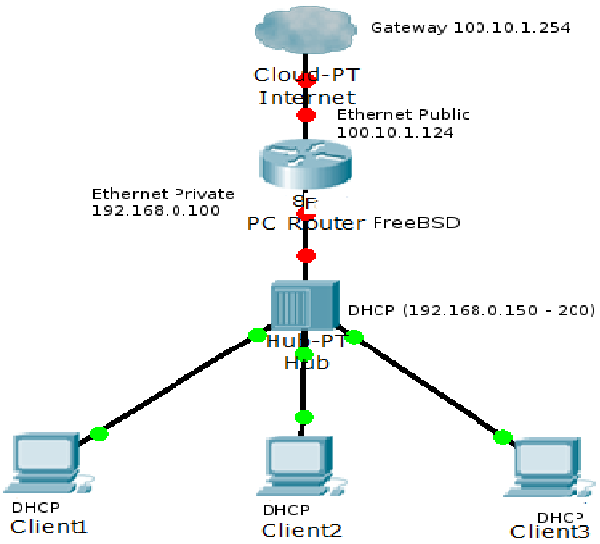
Untuk melihat aturan iptables yang telah di buat ketikan perintah `iptables –L` dan `iptables –L –t nat`, setelah itu *save* iptables agar setiap komputer *restart* kernel akan terus mengenali aturan yang sebelumnya. Dengan menggunakan perintah `/etc/init.d/iptables save`, lalu `rc-update add iptables default`

3.3.2 Menggunakan Kernel FreeBSD

Sama halnya seperti yang dilakukan di kernel GNU/linux setelah dicek kartu jaringan NIC yang dikenal oleh system FreeBSD bukan *Ethernet* (eth) melainkan vr0 dan rl0. Agar segalanya terlihat mudah saya definisikasn terlebih dahulu.

- NIC 0 / vr0 : merupakan NIC yang terhubung ke internet
- NIC 1 / rl0 : merupakan NIC yang terhubung ke LAN
- Vr0 di setting dengan IP address 100.10.1.124

- Rl0 di setting dengan IP address 192.168.0.100
- Dengan *default gateway* sesuai dengan IP Address ADSL yaitu 100.10.1.254
- Dengan menggunakan topologi *star* (lihat gambar 3.7)



Gambar 3.7 *Topologi yang digunakan pada kernel FreeBSD.*

PC *router* berbasis kernel FreeBSD ini mengunakan *router* statik, *router* statik yaitu *router* yang dikonfigurasi secara statis dari IP addressnya hanya untuk *router*. Lalu mengkonfigurasi kartu jaringan dan membuat PC menjadi sebuah *server* berikut adalah langkah-langkah yang dilakukan.

Konfigurasi sistem dan kartu jaringan bisa di lihat di *directory* /etc/rc.conf, buka *file* rc.conf dengan perintah `nano –w /etc/rc.conf` lalu konfigurasikan kartu jaringan yang digunakan.

```
ifconfig_rl0=”inet 192.168.0.100 netmask 255.255.255.0"
ifconfig_vr0=”inet 100.10.1.124  netmask 255.255.255.0"
defaultrouter=”100.10.1.254"
```

Setelah kartu jaringan di beri IP *address*, langkah selanjutnya adalah membuat PC menjadi s*erver*, buka kembali *file rc.conf* yang sebelumnya dibuka karena karakter dari FreeBSD menempatkan konfigurasi sistem dapat dikenal

hanya dalam satu *file* yaitu *rc.conf*. setelah di buka maka tambahkan konfigurasi berikut ini ke dalam *file rc.conf.*

```
gateway_enable=”YES”
inetd_enable=”YES”
sendmail_enable=”NONE”
ipnat_enable=”YES”
ipnat_rules=”/etc/ipnat.rules”
dhcpd_ifaces=”rl0"
dhcpd_enable=”YES”
```

penjelasan dari konfigurasi diatas adalah :

- gateway_enable : mengaktifkan PC sebagai *gateway.*
- inetd_enable : mengaktifkan PC sebagai penyedia layanan internet.
- sendmail_enable= menonaktifkan pesan yang dikirim dari sistem kepada *user.*
- ipnat_enable = mengaktifkansalah satu *firewall* ipnat.
- ipnat_rules = mengarahkan aturan *firewall* ke *file* ipnat.rules dengan aturan seperti dibawah ini.

  ```
  Map vr0 192.168.0.0/24 -> 0.0.0.0/32 portmap tcp/udp 40000:65000
  map vr0 192.168.0.0/24 -> 0.0.0.0/32
  ```

- dhcpd_ifaces = DHCP *server* diaktifkan pada kartu jaringan rl0.
- dhcpd_enable = mengaktifkan *server* DHCP.

Setelah mengkonfigurasi *file rc.conf* langkah selanjutnya adalah mengkonfigurasi *file sysctl.conf, file* ini berfungsi untuk memberikan akses kepada *client* berikut adalah konfigurasinya.

```
net.inet.ip.fw.verbose=1
net.inet.ip.fw.verbose_limit=100
net inet.ip.forwarding=1
```

langkah terakhir adalah membuat konfigurasi DHCP *server* yang berfungsi setiap *client* yang terhubung kedalam jaringan LAN akan mendapatkan IP Address secara otomatis, karena *server* memberikan IP Address ke *client* secara langsung. Sebelumya install terlebih dahulu paket DHCP *server*nya melalui *port* berikut adalah langkah-langkahnya.

```
/usr/port/net/isc-dhcp3-server
```

```
make install clean
```

Setelah instalasi selesai, konfigurasi *file* dhcpcd.conf yang berada pada *directory* `/usr/local/etc/dhcpd.conf`, buka *file* tersebut lalu masukan konfigurasi dibawah ini.

```
Option domain-name-servers 192.168.1.1;
option subnet-mask 255.255.255.0;

default-lease-time 3600;
max-lease-time 86400;
ddns-update-style none;

subnet 192.168.0.0 netmask 255.255.255.0 {
range 192.168.0.120 192.168.0.200;
option routers 192.168.0.100;
}
```

Setelah semuanya selesai di konfigurasi maka cek *ipnat* nya pertama lakukan perintah `ipnat -C` lalu `ipnat -f /etc/ipnat.rules` lalu jalankan `/usr/local/sbin/dhcpd rl0` setelah itu *restart* sistemnya.

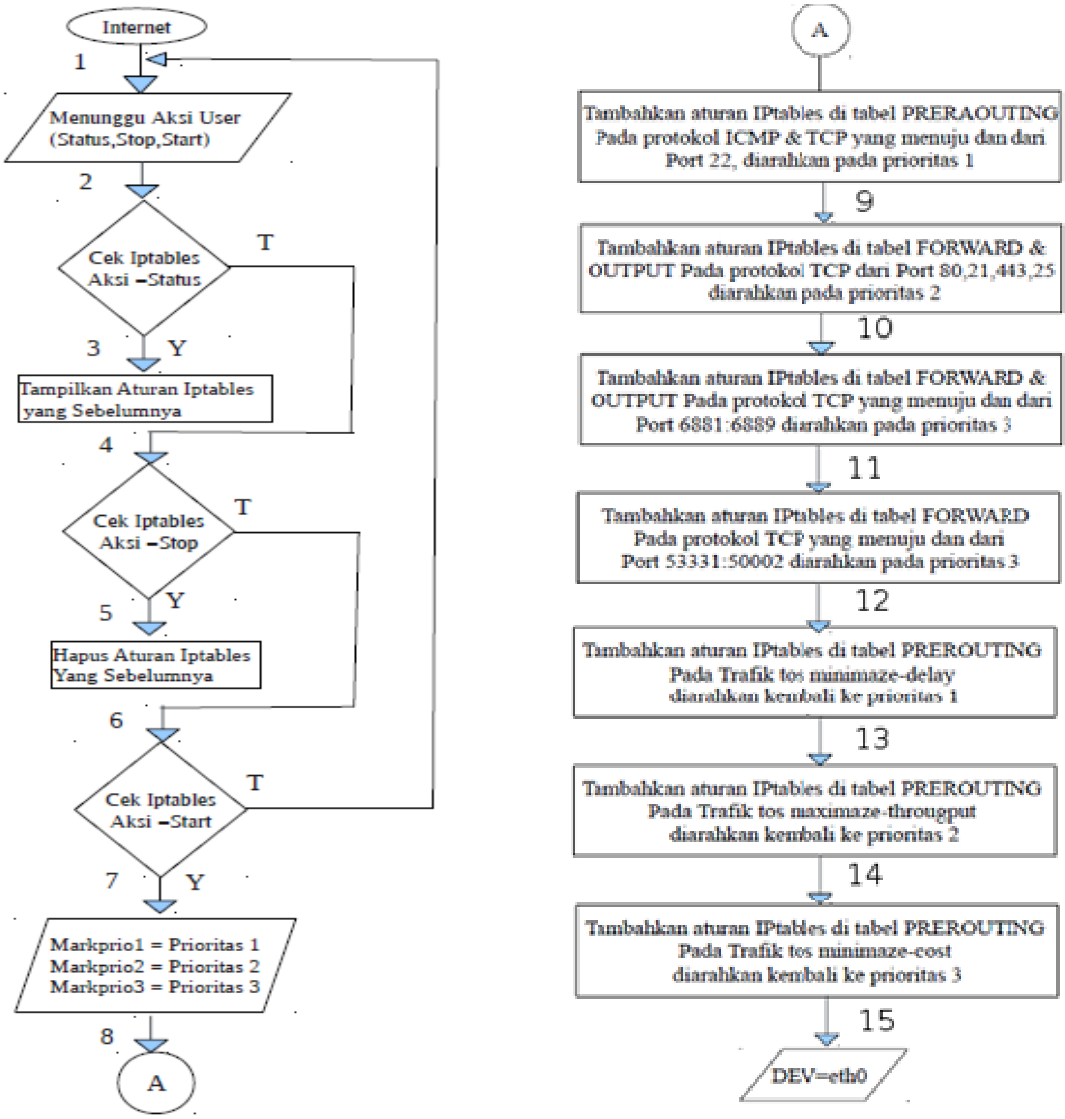
3.4 Konfigurasi Firewall

Setelah PC berhasil menjadi sebuah *router* langkah selanjutnya adalah mengkonfigurasi *firewall*, namun *firewall* pada bagian ini bukan untuk membuka akses pada *client* atau seperti halnya dalam konfigurasi PC menjadi *router* , namun *firewall* disini berfungsi sebagai *firewall* yang membantu pelayanan bersama teknik antrian dalam membuat sistem PC *router* yang dapat mengatur *bandwidth* dengan adil dan merata pada setiap *client* baik ketika *browsing*, *download* dan *upload*. Sehingga teknik antrian dan *firewall* saling berkaitan.

Berikut akan dijelaskan pada system yang berbeda yaitu dalam GNU/linux dan FreeBSD.

3.4.1 Menggunakan Kernel GNU/linux (Iptables)

Script *firewall* ini disimpan pada *directory* `/etc/init.d/`, lalu ubah *runlevel-nya* agar setiap PC *router restart* script ini tetap berjalan. Berikut adalah diagram alir konfigurasi *firewall* yang mendukung teknik antrian HTB yang berjalan pada kernel GNU/linux. (Script dari IPtables berada pada lampiran B).



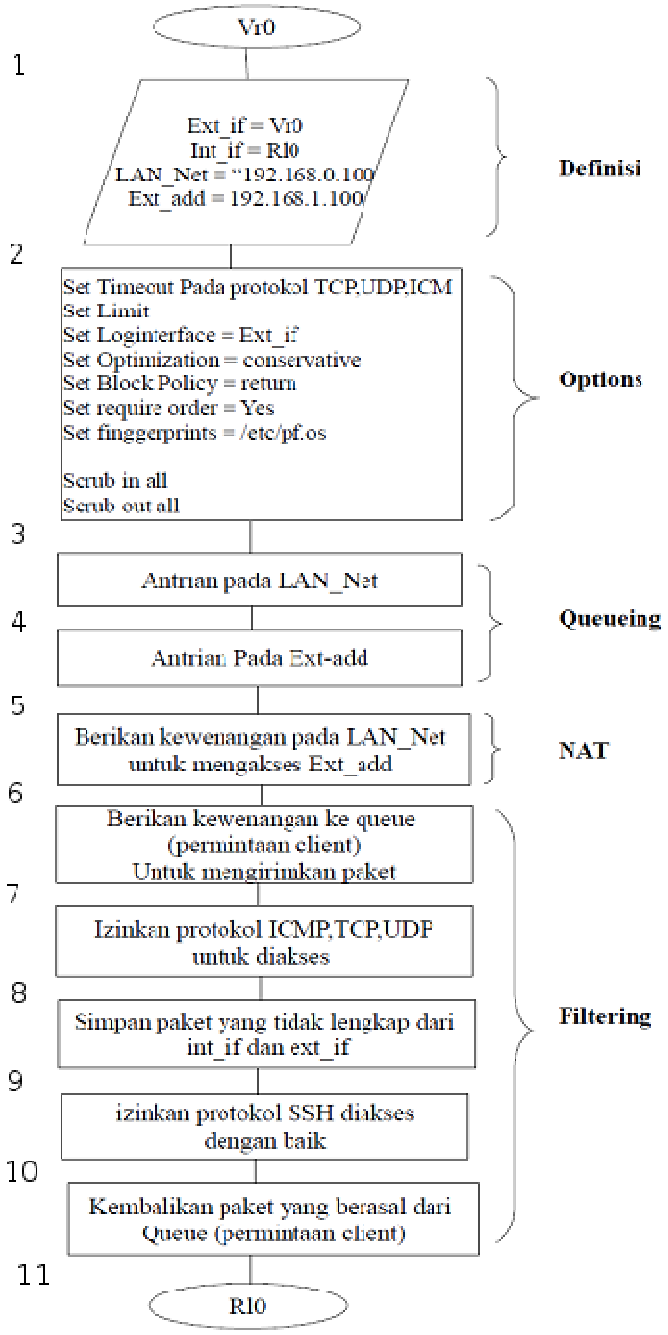
Gambar 3.8 *Diagram Alir dari perancangan firewall iptables.*

Penjelasan pada diagram alir perencanaan *firewall iptables* dalam kaitan pembagian *bandwidth*. Yaitu untuk menyaring paket berdasarkan prioritas :

- Point 1, paket diterima oleh *ethernet* eth1
- Point 2, *firewall* menunggu untuk di mulai, jika menanyakan status maka masuk ke Point 3 yaitu tampilakan aturan *firewall* sebelumnya.
- Point 4, *firewall* kembali menunggu untuk dimulai, jika dicek saat melihat *firewall* sebelunya, maka masuk ke point 5 untuk menghapus aturan *firewall* sebelunya.
- Point 6, *firewall* diaktifkan.
- Point 7, *firewall* memberikan inisialisai untuk prioritas antrian, disini dapat dilihat ada 3 prioritas antrian.
- Point 8, menambahkan aturan *iptables* di tabel PREROUTING pada protokol ICMP dan TCP yang menuju dan dari *port* 22 diarahkan kedalam prioritas 1.
- Point 9, tambahkan aturan *firewall* di tabel FORWARD dan OUTPUT pada protokol TCP dari *port* 80,21,443,25 diarahkan pada prioritas 2.
- Point 10, tambahkan aturan *firewall* di tabel FORWARD dan OUTPUT pada protokol TCP yang menuju dari *port* 6881:6889 diarahkan pada prioritas 3.
- Point 11, tambahkan aturan *firewall* di tabel FORWARD pada protokol TCP yang menuju dari port 53331:50002 diarahkan pada prioritas 3.
- Point 12, tambahkan aturan *firewall* di tabel PREROUTING pada trafik tos minimize-delay diarahkan untuk prioritas 1.
- Point 13, tambahkan aturan *firewall* di tabel PREROUTING pada trafik tos minimize-througput diarahkan untuk prioritas 2.
- Point 14, tambahkan aturan *firewall* di tabel PREROUTING pada trafik tos minimize-cost diarahkan untuk prioritas 3.
- Point 15, semua aturan diberlakukan semua paket yang datang dari *ethernet* eth1 (paket yang masuk dari LAN)

3.4.2 Menggunakan Kernel FreeBSD (PF)

Untuk konfigurasi *firewall* yang menunjang dalam pengaturan *bandwidth* di sistem operasi freebsd scriptnya berhubungan langsung dengan *trafik shapping* yang berada dalam satu *file* (script) yang berada pada *directory* `/etc/pf.conf`, berikut adalah diagram alir dari keseluruhan script. (script berada pada lampiran B).



Gambar 3.9 *Diagram Alir dari perancangan firewall PF.*

Penjelasan pada diagram alir perencanaan *firewall* PF dalam kaitan pembagian *bandwidth*. Yaitu untuk menyaring paket berdasarkan prioritas :

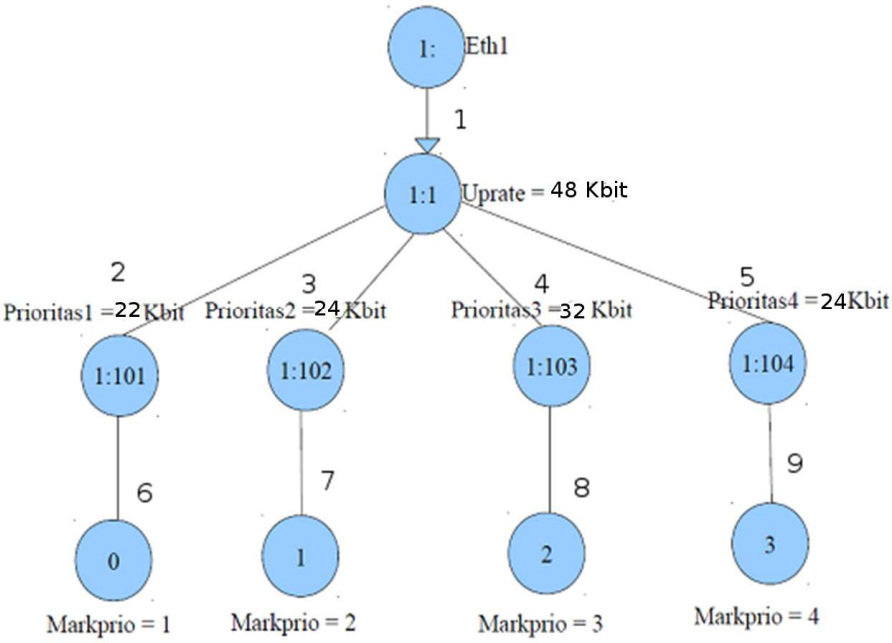
- Point 1, proses dimulai dari vr0 yaitu *ethernet* yang terhubung ke internet. Untuk diproses *firewallnya*, namun sebelumnya didefinisikan terlebih dahulu ethernet *external* beserta IP *address*, dan *ethernet* internal beserta IP *address.*
- Point 2, setelah didefinisikan maka harus masuk kebagian *options*, untuk menseting *firewall* PF.
- Point 3, masuk ke dalam *firewall* antrian untuk bagian lokal
- Point 4, masuk ke dalam *firewall* antrian untuk bagian yang ke internet
- Point 5, memberikan hak kepada *ethernet* lokal untuk dapat mengakses *ethernet* yang mengarah ke internet
- Point 6, memberikan akses ke *client* untuk masuk ke dalam antrian agar dapat mengirimkan paket.
- Point 7, memberikan akses untuk protokol ICMP,UDP,TCP agar dapat diakses *client.*
- Point 8, menyimpan paket yang tidak lengkap pada *ethernet* lokal dan *ethernet* dari internet.
- Point 9, mengizinkan protokol SSH untuk diakses dengan baik.
- Point 10, mengembalikan paket yang diminta oleh *client* melalui *client.*
- Point 11, aturan *firewall* dan pembagian *bandwidth* berlaku untuk *ethernet* lokal (rl0).

3.5 Konfigurasi Traffik Bandwidth

Pada konfigurasi kali ini yang menentukan apakah *bandwidth* yang di terima setiap *client* dengan *bandwidth* yang sama atau merata. Dengan bantuan *firewall* konfigurasi ini dapat bekerja secara optimal.

3.5.1 Menggunakan Kernel GNU/linux (HTB)

Teknik antrian HTB memberikan kita fasilitas pembatasan trafik pada setiap level maupun klasifikasi, bandwidth yang tidak terpakai bisa digunakan oleh klasifikasi yang lebih rendah. Kita juga dapat melihat HTB seperti suatu struktur organisasi dimana pada setiap bagian memiliki wewenang dan mampu membantu bagian lain yang memerlukan. Berikut adalah skema antrian dan diagram alir dari script HTB. (script berada pada lampiran B).



Gambar 3.10 *Diagram Alir dari teknik antrian HTB.*

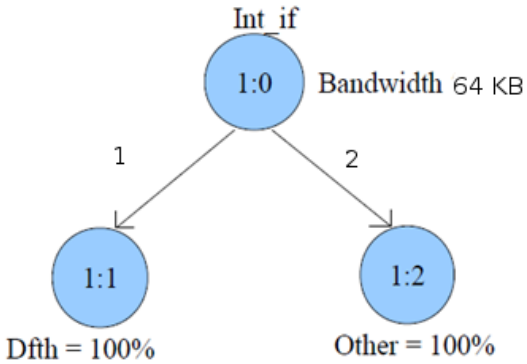
Penjelasan dari diagram alir dalam perancangan pembagian bandwidth menggunakan teknik antrian HTB :

- Pada eth1 yang merupakan ethernet untuk jaringan lokal (LAN).

- Eth1 membuat sebuah kelas dengan identitas 1:1 dengan besar *bandwidth* 56 Kbit yaitu *uprate* atau besar *bandwidth* yang di dapat oleh *client.*
- 1:1 mempunyai 4 *sub class* dengan identitas 1:101,1:102,1;103 dan 1:104 dengan *sub class default* adalah 1:103.
- Dari tiap-tiap *sub class* memiliki masing-masing satu *leaf* atau kelas yang tidak memiliki *sub class* lagi. Namun langsung dimasukan ke dalam prioritas yang ditentukan.
- Setiap kelas yang tidak terpakai *bandwidthnya* akan digunakan oleh kelas yang aktif.

3.5.2 Menggunakan Kernel FreeBSD (CBQ)

CBQ memungkinkan *sharing bandwidth* antar kelas dan mengatur pemakaian *bandwidth* jaringan yang dialokasikan untuk tiap user, pemakaian *bandwidth* yang melebihi nilai set akan dipotong (*shaping*), dan juga dapat diatur untuk *sharing* dan meminjam *bandwidth* antar *class* jika diperlukan. Script teknik antrian ini berada dalam satu dengan *firewall* PF berbeda dengan yang berada di sistem operasi GNU/linux, berikut adalah skema dari teknik antrian pada sistem operasi FreeBSD. Antrian yang disetting pada NIC yang menuju LAN : (script berada pada lampiran B).

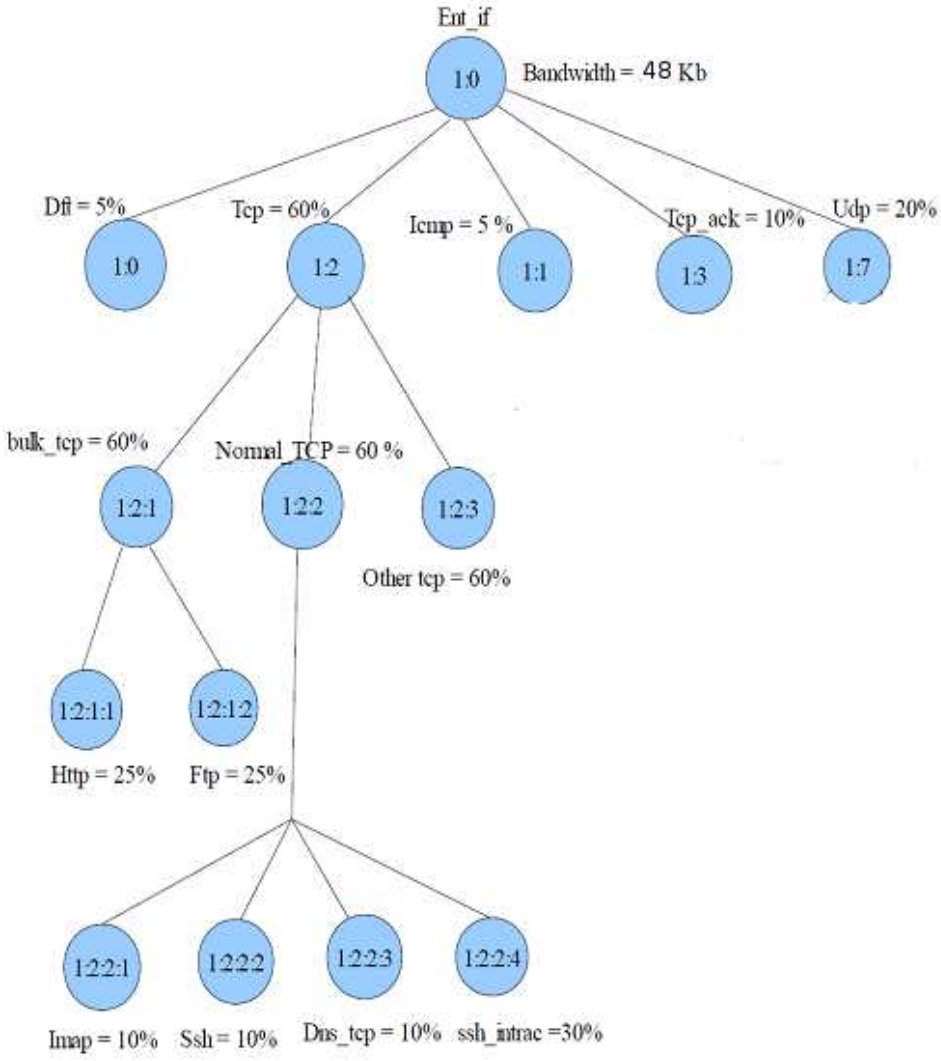


Gambar 3.11 *Diagram Alir dari teknik antrian CBQ bagian internal.*

Penjelasan dari diagram alir yang menggunakan CBQ bagian internal adalah sebagai berikut:

- *Ethernet* internal (rl0) diberikan *bandwidth* 64 Kbit dan mempunyai 2 *child*.
- *Child* pertama yaitu dfth dengan skala 100% sama artinya 64 Kbit
- *Child* kedua yaitu other dengan besar *bandwidth* 64 Kbit. Keduanya digunakan untuk proses jaringan atau pertukaran data pada lokal (LAN).

Antrian yang disetting pada NIC yang menuju internet :



Gambar 3.12 *Diagram Alir dari teknik antrian CBQ bagian Eksternal.*

Penjelasan dari diagram alir yang menggunakan CBQ bagian eksternal adalah sebagai berikut:

- *Ethernet* eksternal (vr0) diberikan besar *bandwidth* sebesar 48 Kbit, yang artinya ethernet eksternal tidak melebihi kuota dari *ethernet* internal.
- *Ethernet* vr0 mempunyai lima *child*, yaitu dft diberi kuota 5%, lalu tcp diberi 60%, icmp besarnya 5%, tcp ack sebesar 10%, dan udp sebesar 10% semua besaran kuota pada *child* berasal dari besar *bandwidth* pada *ethernet* vr0.
- Dari semua *child* yang dimiliki oleh *ethernet* vr0, *child* yang memiliki *child* lagi hanya tcp, sedangkan yang lain berdiri sendiri.
- Tcp selain menjadi *childnya* dari vr0, tcp juga mempunyai tiga *child* yang memiliki besar *bandwidth* sama dengan parentnya (tcp) yaitu bulk_tcp, normal_tcp, dan other_tcp dengan kuota sama-sama 60%.
- Dari ketiga *child* tcp, hanya other_tcp yang tidak mempunya *child*, yang mempunya *child* adalah bulk_tcp dengan *child* http dan tcp lalu normal_tcp mempunyai *child* imap, ssh, dns_tcp, dan ssh_intrac.
- Bulk_tcp menjadi *parent* bagi http dan ftp dengan kuota untuk http sebesar 25% begitu juga dengan *child* ftp.
- Normal_tcp menjadi *parent* bagi imap, ssh, dns_tcp, dan ssh_intrac. Dengan besar kuota bagi imap , ssh ,dan dns_tcp sama-sama sebesar 10% sedangkan kuota untuk ssh_intrac sebesar 30%.