

# Tutorial IPTables

Lukman HDP/s3trum (lukman\_hdp@yahoo.com)

August 5, 2003

---

*Tulisan ini ditujukan untuk memberikan pengetahuan dasar mengenai pemfilteran paket menggunakan IPTables pada Linux. Tulisan ini bersifat general yang menjelaskan secara umum bagaimana sintaks IPTables dibuat. Beberapa (banyak?) bagian dari tulisan diambil dari official site IPTables. Tidak ada copyright apapun dalam dokumen ini, anda bebas menyalin, mencetak, maupun memodifikasi (dengan menyertakan nama penulis asli). Kritik, koreksi, saran dan lain-lain silahkan dialamatkan ke email tersebut di atas. Semoga bermanfaat.*

---

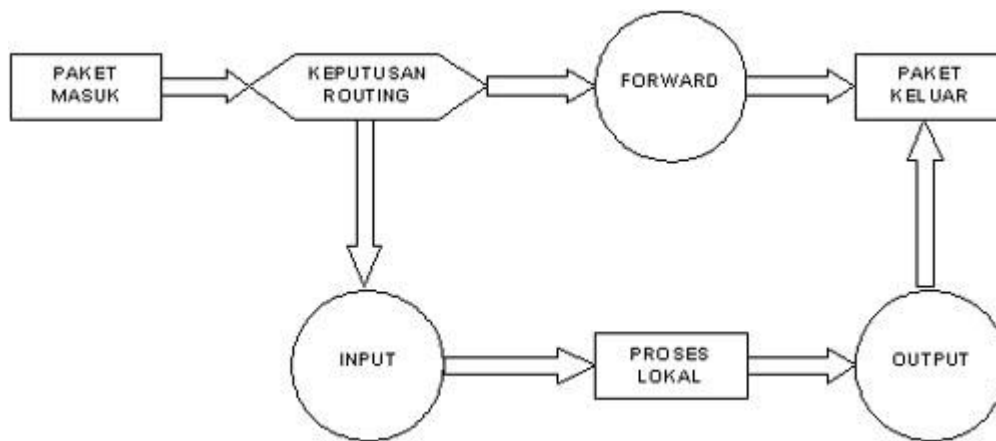
## 1. Persiapan

Sebelum mulai, diharapkan pembaca sudah memiliki pengetahuan dasar mengenai TCP/IP karena hal ini merupakan dasar dari penggunaan IPTables. Ada (sangat) banyak resource yang mendokumentasikan konsep dasar tentang TCP/IP, baik itu secara online maupun cetak. Silahkan googling untuk mendapatkannya.

Hal berikutnya yang harus anda persiapkan adalah sebuah komputer yang terinstall Linux. Akan lebih baik jika komputer anda memiliki 2 buah *network interface card*, sebab bisa menjalankan fungsi *packet forwarding*. Disarankan anda menggunakan linux dengan kernel 2.4 ke atas, karena (setahu saya) linux dengan kernel 2.4 ke atas sudah memiliki dukungan IPTables secara default, sehingga anda tidak perlu mengkompilasi ulang kernel anda. Bagi anda yang menggunakan kernel 2.2 atau sebelumnya, anda harus melakukan kompilasi kernel untuk memasukkan dukungan IPTables. Silahkan lihat tutorial [Kompilasi kernel 2.4.x di Linux](#) oleh mas Asfik.

## 2. Pendahuluan

IPTables memiliki tiga macam daftar aturan bawaan dalam tabel penyaringan, daftar tersebut dinamakan rantai firewall (*firewall chain*) atau sering disebut *chain* saja. Ketiga chain tersebut adalah INPUT, OUTPUT dan FORWARD.



Pada diagram tersebut, lingkaran menggambarkan ketiga rantai atau chain. Pada saat sebuah paket sampai pada sebuah lingkaran, maka disitulah terjadi proses penyaringan. Rantai akan memutuskan nasib paket tersebut. Apabila keputusannya 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.

Jalannya sebuah paket melalui diagram tersebut bisa dicontohkan sebagai berikut:

### **Perjalanan paket yang diforward ke host yang lain**

1. Paket berada pada jaringan fisik, contoh internet.
2. Paket masuk ke interface jaringan, contoh eth0.
3. Paket masuk ke chain PREROUTING pada table Mangle. Chain ini berfungsi untuk memangle (menghaluskan) paket, seperti merubah TOS, TTL dan lain-lain.
4. Paket masuk ke chain PREROUTING pada tabel nat. Chain ini berfungsi utamanya untuk melakukan DNAT (Destination Network Address Translation).
5. Paket mengalami keputusan routing, apakah akan diproses oleh host lokal atau diteruskan ke host lain.
6. Paket masuk ke chain FORWARD pada tabel filter. Disinilah proses pemfilteran yang utama terjadi.
7. Paket masuk ke chain POSTROUTING pada tabel nat. Chain ini berfungsi utamanya untuk melakukan SNAT (Source Network Address Translation).
8. Paket keluar menuju interface jaringan, contoh eth1.
9. Paket kembali berada pada jaringan fisik, contoh LAN.

### **Perjalanan paket yang ditujukan bagi host lokal**

1. Paket berada dalam jaringan fisik, contoh internet.
2. Paket masuk ke interface jaringan, contoh eth0.

3. Paket masuk ke chain PREROUTING pada tabel mangle.
4. Paket masuk ke chain PREROUTING pada tabel nat.
5. Paket mengalami keputusan routing.
6. Paket masuk ke chain INPUT pada tabel filter untuk mengalami proses penyaringan.
7. Paket akan diterima oleh aplikasi lokal.

### Perjalanan paket yang berasal dari host lokal

1. Aplikasi lokal menghasilkan paket data yang akan dikirimkan melalui jaringan.
2. Paket memasuki chain OUTPUT pada tabel mangle.
3. Paket memasuki chain OUTPUT pada tabel nat.
4. Paket memasuki chain OUTPUT pada tabel filter.
5. Paket mengalami keputusan routing, seperti ke mana paket harus pergi dan melalui interface mana.
6. Paket masuk ke chain POSTROUTING pada tabel NAT.
7. Paket masuk ke interface jaringan, contoh eth0.
8. Paket berada pada jaringan fisik, contoh internet.

## 3. Sintaks IPTables

```
iptables [-t table] command [match] [target/jump]
```

### 1. Table

IPTables memiliki 3 buah tabel, yaitu NAT, MANGLE dan FILTER. Penggunaannya disesuaikan dengan sifat dan karakteristik masing-masing. Fungsi dari masing-masing tabel tersebut sebagai berikut :

- a. NAT : Secara umum digunakan untuk melakukan Network Address Translation. NAT adalah penggantian field alamat asal atau alamat tujuan dari sebuah paket.
- b. MANGLE : Digunakan untuk melakukan penghalusan (mangle) paket, seperti TTL, TOS dan MARK.
- c. FILTER : Secara umum, inilah pemfilteran paket yang sesungguhnya.. Di sini bisa ditentukan apakah paket akan di-DROP, LOG, ACCEPT atau REJECT

### 2. Command

Command pada baris perintah IPTables akan memberitahu apa yang harus dilakukan terhadap lanjutan sintaks perintah. Umumnya dilakukan penambahan atau penghapusan sesuatu dari tabel atau yang lain.

Command	Keterangan
<b>-A</b> <b>--append</b>	Perintah ini menambahkan aturan pada akhir chain. Aturan akan ditambahkan di akhir baris pada chain yang bersangkutan, sehingga akan dieksekusi terakhir
<b>-D</b> <b>--delete</b>	Perintah ini menghapus suatu aturan pada chain. Dilakukan dengan cara menyebutkan secara lengkap perintah yang ingin dihapus atau dengan menyebutkan

	nomor baris dimana perintah akan dihapus.
<b>-R</b> <b>--replace</b>	Penggunaannya sama seperti <b>--delete</b> , tetapi <i>command</i> ini menggantinya dengan entry yang baru.
<b>-I</b> <b>--insert</b>	Memasukkan aturan pada suatu baris di chain. Aturan akan dimasukkan pada baris yang disebutkan, dan aturan awal yang menempati baris tersebut akan digeser ke bawah. Demikian pula baris-baris selanjutnya.
<b>-L</b> <b>--list</b>	Perintah ini menampilkan semua aturan pada sebuah tabel. Apabila tabel tidak disebutkan, maka seluruh aturan pada semua tabel akan ditampilkan, walaupun tidak ada aturan sama sekali pada sebuah tabel. <i>Command</i> ini bisa dikombinasikan dengan option <b>-v</b> (verbose), <b>-n</b> (numeric) dan <b>-x</b> (exact).
<b>-F</b> <b>--flush</b>	Perintah ini mengosongkan aturan pada sebuah chain. Apabila chain tidak disebutkan, maka semua chain akan di- <i>flush</i> .
<b>-N</b> <b>--new-chain</b>	Perintah tersebut akan membuat chain baru.
<b>-X</b> <b>--delete-chain</b>	Perintah ini akan menghapus chain yang disebutkan. Agar perintah di atas berhasil, tidak boleh ada aturan lain yang mengacu kepada chain tersebut.
<b>-P</b> <b>--policy</b>	Perintah ini membuat kebijakan default pada sebuah chain. Sehingga jika ada sebuah paket yang tidak memenuhi aturan pada baris-baris yang telah didefinisikan, maka paket akan diperlakukan sesuai dengan kebijakan default ini.
<b>-E</b> <b>--rename-chain</b>	Perintah ini akan merubah nama suatu chain.

### 3. Option

Option digunakan dikombinasikan dengan command tertentu yang akan menghasilkan suatu variasi perintah.

Option	Command Pemakai	Keterangan
<b>-v</b> <b>--verbose</b>	<b>--list</b> <b>--append</b> <b>--insert</b> <b>--delete</b> <b>--replace</b>	Memberikan output yang lebih detail, utamanya digunakan dengan <b>--list</b> . Jika digunakan dengan <b>--list</b> , akan menampilkan K (x1.000), M (1.000.000) dan G (1.000.000.000).
<b>-x</b> <b>--exact</b>	<b>--list</b>	Memberikan output yang lebih tepat.
<b>-n</b> <b>--numeric</b>	<b>--list</b>	Memberikan output yang berbentuk angka. Alamat IP dan nomor port akan ditampilkan dalam bentuk angka dan bukan hostname ataupun nama aplikasi/servis.
<b>--line-number</b>	<b>--list</b>	Akan menampilkan nomor dari daftar aturan. Hal ni akan mempermudah bagi kita untuk melakukan modifikasi aturan, jika kita mau meyisipkan atau menghapus aturan dengan nomor

		tertentu.
<code>--modprobe</code>	<code>All</code>	Memerintah IPTables untuk memanggil modul tertentu. Bisa digunakan bersamaan dengan semua <i>command</i> .

#### 4. Generic Matches

Generic Matches artinya pendefinisian kriteria yang berlaku secara umum. Dengan kata lain, sintaks generic matches akan sama untuk semua protokol. Setelah protokol didefinisikan, maka baru didefinisikan aturan yang lebih spesifik yang dimiliki oleh protokol tersebut. Hal ini dilakukan karena tiap-tiap protokol memiliki karakteristik yang berbeda, sehingga memerlukan perlakuan khusus.

Match	Keterangan
<code>-p</code> <code>--protocol</code>	Digunakan untuk mengecek tipe protokol tertentu. Contoh protokol yang umum adalah TCP, UDP, ICMP dan ALL. Daftar protokol bisa dilihat pada <code>/etc/protocols</code> .  Tanda inversi juga bisa diberlakukan di sini, misal kita menghendaki semua protokol kecuali icmp, maka kita bisa menuliskan <code>--protokol ! icmp</code> yang berarti semua kecuali icmp.
<code>-s</code> <code>--src</code> <code>--source</code>	Kriteria ini digunakan untuk mencocokkan paket berdasarkan alamat IP asal. Alamat di sini bisa berbentuk alamat tunggal seperti 192.168.1.1, atau suatu alamat network menggunakan netmask misal 192.168.1.0/255.255.255.0, atau bisa juga ditulis 192.168.1.0/24 yang artinya semua alamat 192.168.1.x. Kita juga bisa menggunakan inversi.
<code>-d</code> <code>--dst</code> <code>--destination</code>	Digunakan untuk mencocokkan paket berdasarkan alamat tujuan. Penggunaannya sama dengan <code>match -src</code>
<code>-i</code> <code>--in-interface</code>	<i>Match</i> ini berguna untuk mencocokkan paket berdasarkan interface di mana paket datang. <i>Match</i> ini hanya berlaku pada chain INPUT, FORWARD dan PREROUTING
<code>-o</code> <code>--out-interface</code>	Berfungsi untuk mencocokkan paket berdasarkan interface di mana paket keluar. Penggunaannya sama dengan <code>--in-interface</code> . Berlaku untuk chain OUTPUT, FORWARD dan POSTROUTING

#### 5. Implicit Matches

Implicit Matches adalah match yang spesifik untuk tipe protokol tertentu. Implicit Match merupakan sekumpulan rule yang akan diload setelah tipe protokol disebutkan. Ada 3 Implicit Match berlaku untuk tiga jenis protokol, yaitu TCP matches, UDP matches dan ICMP matches.

##### a. TCP matches

Match	Keterangan
<b>--sport</b> <b>--source-port</b>	<p><i>Match</i> ini berguna untuk mencocokkan paket berdasarkan port asal. Dalam hal ini kita bisa mendefinisikan nomor port atau nama <i>service</i>-nya. Daftar nama <i>service</i> dan nomor port yang bersesuaian dapat dilihat di <code>/etc/services</code>.</p> <p><b>--sport</b> juga bisa dituliskan untuk range port tertentu. Misalkan kita ingin mendefinisikan range antara port 22 sampai dengan 80, maka kita bisa menuliskan <b>--sport 22:80</b>.</p> <p>Jika bagian salah satu bagian pada range tersebut kita hilangkan maka hal itu bisa kita artikan dari port 0, jika bagian kiri yang kita hilangkan, atau 65535 jika bagian kanan yang kita hilangkan. Contohnya <b>--sport :80</b> artinya paket dengan port asal nol sampai dengan 80, atau <b>--sport 1024:</b> artinya paket dengan port asal 1024 sampai dengan 65535. <i>Match</i> ini juga mengenal inversi.</p>
<b>--dport</b> <b>--destination-port</b>	<p>Penggunaan <i>match</i> ini sama dengan <i>match --source-port</i>.</p>
<b>--tcp-flags</b>	<p>Digunakan untuk mencocokkan paket berdasarkan <i>TCP flags</i> yang ada pada paket tersebut. Pertama, pengecekan akan mengambil daftar <i>flag</i> yang akan diperbandingkan, dan kedua, akan memeriksa paket yang di-<i>set</i> 1, atau <i>on</i>.</p> <p>Pada kedua <i>list</i>, masing-masing entry-nya harus dipisahkan oleh koma dan tidak boleh ada spasi antar entry, kecuali spasi antar kedua <i>list</i>. <i>Match</i> ini mengenali <code>SYN, ACK, FIN, RST, URG, PSH</code>. Selain itu kita juga menuliskan <code>ALL</code> dan <code>NONE</code>. <i>Match</i> ini juga bisa menggunakan inversi.</p>
<b>--syn</b>	<p><i>Match</i> ini akan memeriksa apakah flag <code>SYN</code> di-<i>set</i> dan <code>ACK</code> dan <code>FIN</code> tidak di-<i>set</i>. Perintah ini sama artinya jika kita menggunakan <i>match --tcp-flags SYN,ACK,FIN SYN</i></p> <p>Paket dengan <i>match</i> di atas digunakan untuk melakukan <i>request</i> koneksi TCP yang baru terhadap server</p>

## b. UDP Matches

Karena bahwa protokol UDP bersifat *connectionless*, maka tidak ada flags yang mendeskripsikan status paket untuk membuka atau menutup koneksi. Paket UDP juga tidak memerlukan *acknowledgement*. Sehingga *Implicit Match* untuk protokol UDP lebih sedikit daripada TCP.

Ada dua macam *match* untuk UDP:

```
--sport atau --source-port  
--dport atau --destination-port
```

### c. ICMP Matches

Paket ICMP digunakan untuk mengirimkan pesan-pesan kesalahan dan kondisi-kondisi jaringan yang lain. Hanya ada satu implicit match untuk tipe protokol ICMP, yaitu :

```
--icmp-type
```

## 6. Explicit Matches

### a. MAC Address

Match jenis ini berguna untuk melakukan pencocokan paket berdasarkan MAC source address. Perlu diingat bahwa MAC hanya berfungsi untuk jaringan yang menggunakan teknologi ethernet.

```
iptables -A INPUT -m mac -mac-source 00:00:00:00:00:01
```

### b. Multiport Matches

Ekstensi Multiport Matches digunakan untuk mendefinisikan port atau port range lebih dari satu, yang berfungsi jika ingin didefinisikan aturan yang sama untuk beberapa port. Tapi hal yang perlu diingat bahwa kita tidak bisa menggunakan port matching standard dan multiport matching dalam waktu yang bersamaan.

```
iptables -A INPUT -p tcp -m multiport --source-port 22,53,80,110
```

### c. Owner Matches

Penggunaan match ini untuk mencocokkan paket berdasarkan pembuat atau pemilik/owner paket tersebut. Match ini bekerja dalam chain OUTPUT, akan tetapi penggunaan match ini tidak terlalu luas, sebab ada beberapa proses tidak memiliki owner (??).

```
iptables -A OUTPUT -m owner --uid-owner 500
```

Kita juga bisa memfilter berdasarkan group ID dengan sintaks --gid-owner. Salah satu penggunaannya adalah bisa mencegah user selain yang dikehendaki untuk mengakses internet misalnya.

### d. State Matches

Match ini mendefinisikan state apa saja yang cocok. Ada 4 state yang berlaku, yaitu NEW, ESTABLISHED, RELATED dan INVALID. NEW digunakan untuk paket yang akan memulai koneksi baru. ESTABLISHED digunakan jika koneksi telah tersambung dan paket-paketnya merupakan bagian dari koneksi tersebut. RELATED digunakan untuk paket-paket yang bukan bagian dari koneksi tetapi masih berhubungan dengan koneksi tersebut, contohnya adalah FTP data transfer yang menyertai sebuah koneksi TCP atau UDP.

INVALID adalah paket yang tidak bisa diidentifikasi, bukan merupakan bagian dari koneksi yang ada.

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED
```

## 7. Target/Jump

Target atau jump adalah perlakuan yang diberikan terhadap paket-paket yang memenuhi kriteria atau match. Jump memerlukan sebuah chain yang lain dalam tabel yang sama. Chain tersebut nantinya akan dimasuki oleh paket yang memenuhi kriteria. Analoginya ialah chain baru nanti berlaku sebagai prosedur/fungsi dari program utama. Sebagai contoh dibuat sebuah chain yang bernama tcp\_packets. Setelah ditambahkan aturan-aturan ke dalam chain tersebut, kemudian chain tersebut akan direferensi dari chain input.

```
iptables -A INPUT -p tcp -j tcp_packets
```

Target	Keterangan
-j ACCEPT --jump ACCEPT	Ketika paket cocok dengan daftar <i>match</i> dan target ini diberlakukan, maka paket tidak akan melalui baris-baris aturan yang lain dalam chain tersebut atau chain yang lain yang mereferensi chain tersebut. Akan tetapi paket masih akan memasuki chain-chain pada tabel yang lain seperti biasa.
-j DROP --jump DROP	Target ini men- <i>drop</i> paket dan menolak untuk memproses lebih jauh. Dalam beberapa kasus mungkin hal ini kurang baik, karena akan meninggalkan <i>dead socket</i> antara <i>client</i> dan <i>server</i> .  Paket yang menerima target DROP benar-benar mati dan target tidak akan mengirim informasi tambahan dalam bentuk apapun kepada client atau server.
-j RETURN --jump RETURN	Target ini akan membuat paket berhenti melintasi aturan-aturan pada chain dimana paket tersebut menemui target RETURN. Jika chain merupakan <i>subchain</i> dari chain yang lain, maka paket akan kembali ke <i>superset chain</i> di atasnya dan masuk ke baris aturan berikutnya. Apabila <i>chain</i> adalah chain utama misalnya INPUT, maka paket akan dikembalikan kepada kebijakan default dari <i>chain</i> tersebut.
-j MIRROR	Apabila kompuuter A menjalankan target seperti contoh di atas, kemudian komputer B melakukan koneksi http ke komputer A, maka yang akan muncul pada browser adalah website komputer B itu sendiri. Karena fungsi utama target ini adalah membalik <i>source address</i> dan <i>destination address</i> .  Target ini bekerja pada chain INPUT, FORWARD dan PREROUTING atau chain buatan yang dipanggil melalui chain tersebut.

Beberapa target yang lain biasanya memerlukan parameter tambahan:

### a. LOG Target

Ada beberapa option yang bisa digunakan bersamaan dengan target ini. Yang pertama adalah yang digunakan untuk menentukan tingkat log. Tingkatan log yang bisa digunakan adalah debug, info, notice, warning, err, crit, alert dan emerg. Yang kedua adalah -j LOG --log-prefix yang digunakan untuk memberikan string yang tertulis pada awalan log, sehingga memudahkan pembacaan log tersebut.

```
iptables -A FORWARD -p tcp -j LOG --log-level debug
iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT Packets"
```

## **b. REJECT Target**

Secara umum, REJECT bekerja seperti DROP, yaitu memblokir paket dan menolak untuk memproses lebih lanjut paket tersebut. Tetapi, REJECT akan mengirimkan error message ke host pengirim paket tersebut. REJECT bekerja pada chain INPUT, OUTPUT dan FORWARD atau pada chain tambahan yang dipanggil dari ketiga chain tersebut.

```
iptables -A FORWARD -p tcp -dport 22 -j REJECT --reject-with icmp-host-unreachable
```

Ada beberapa tipe pesan yang bisa dikirimkan yaitu icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable, icmp-protocol-unreachable, icmp-net-prohibited dan icmp-host-prohibited.

## **c. SNAT Target**

Target ini berguna untuk melakukan perubahan alamat asal dari paket (Source Network Address Translation). Target ini berlaku untuk tabel nat pada chain POSTROUTING, dan hanya di sinilah SNAT bisa dilakukan. Jika paket pertama dari sebuah koneksi mengalami SNAT, maka paket-paket berikutnya dalam koneksi tersebut juga akan mengalami hal yang sama.

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 194.236.50.155-194.236.50.160:1024-32000
```

## **d. DNAT Target**

Berkebalikan dengan SNAT, DNAT digunakan untuk melakukan translasi field alamat tujuan (Destination Network Address Translation) pada header dari paket-paket yang memenuhi kriteria match. DNAT hanya bekerja untuk tabel nat pada chain PREROUTING dan OUTPUT atau chain buatan yang dipanggil oleh kedua chain tersebut.

```
iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 --dport 80 -j DNAT --to-destination 192.168.0.2
```

## **e. MASQUERADE Target**

Secara umum, target MASQUERADE bekerja dengan cara yang hampir sama seperti target SNAT, tetapi target ini tidak memerlukan option --to-source. MASQUERADE memang didesain untuk bekerja pada komputer dengan koneksi yang tidak tetap seperti dial-up atau DHCP yang akan memberi pada kita nomor IP yang berubah-ubah.

Seperti halnya pada SNAT, target ini hanya bekerja untuk tabel nat pada chain POSTROUTING.

```
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

## f. REDIRECT Target

Target REDIRECT digunakan untuk mengalihkan jurusan (redirect) paket ke mesin itu sendiri. Target ini umumnya digunakan untuk mengarahkan paket yang menuju suatu port tertentu untuk memasuki suatu aplikasi proxy, lebih jauh lagi hal ini sangat berguna untuk membangun sebuah sistem jaringan yang menggunakan transparent proxy. Contohnya kita ingin mengalihkan semua koneksi yang menuju port http untuk memasuki aplikasi http proxy misalnya squid. Target ini hanya bekerja untuk tabel nat pada chain PREROUTING dan OUTPUT atau pada chain buatan yang dipanggil dari kedua chain tersebut.

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j REDIRECT --to-port 3128
```

Tutorial Squid bisa dilihat di [Instalasi Squid, Banner Filter, Porn Filter, Limit Bandwith, Transparan Proxy](#) buatan mas Hanny.

## 4. Penutup

Demikian dasar-dasar dari IPTables beserta komponen-komponennya. Mungkin anda masih agak bingung tentang implementasi dari apa yang telah dijelaskan di atas. Insya Allah dalam tulisan yang akan datang, saya akan memberikan beberapa contoh kasus jaringan yang menggunakan IPTables. Yea.. may I have enough power to do it :)

## 5. Change Log

5 Agustus 2003

- Penulisan pertama dokumen ini

## 6. Referensi

1. [www.netfilter.org](http://www.netfilter.org)
2. Manual page iptables
3. Beberapa sumber yang lain, tapi saya lupa :)