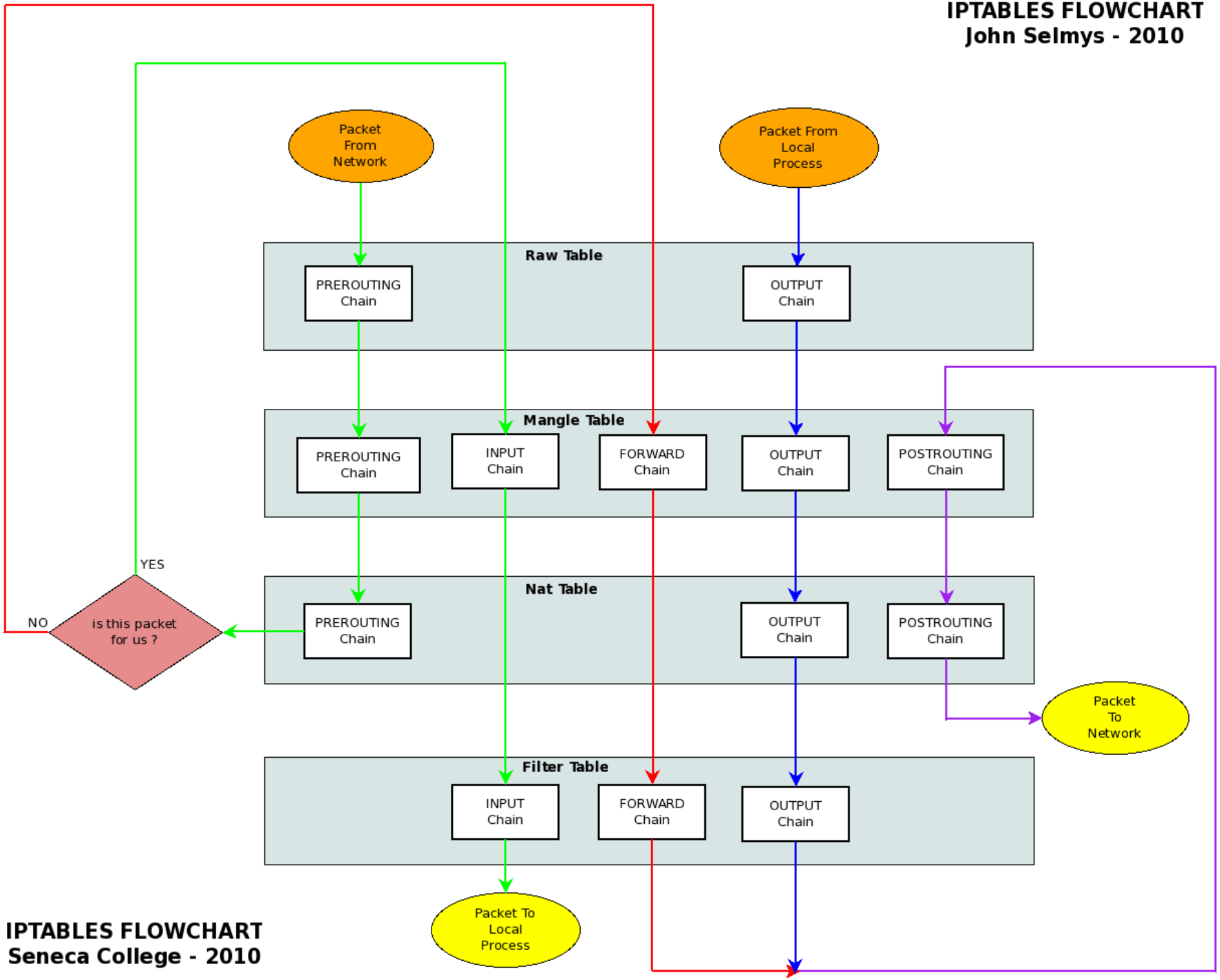
Part A
A Gentle Introduction to iptables

iptables is a command, used by system administrators, to manage the network packet filtering subsystem within the Linux kernel

iptables is used for IPV4 networks
ip6tables is used for IPV6 networks

iptables can be used to build firewalls, perform network address translation and to log network activity

IPTABLES FLOWCHART
John Selmys - 2010



IPTABLES FLOWCHART
Seneca College - 2010

Four Tables

- **filter**: used to filter packets (default)
- **nat**: used for network address translation
- **mangle**: used for specialized packet alteration
- **raw**: used mainly for configuring exemptions from connection tracking

List All Rules in All Chains in a Table

- iptables -t filter -L
  - same as iptables -L
- iptables -t nat -L
- iptables -t mangle -L
- iptables -t raw -L

What is a Chain?

Each chain is a list of rules which can match a set of packets. Each rule specifies what to do with a packet that matches. This is called a ‘target’, which may be a jump to a user-defined chain in the same table.

List All Rules in One Chain in a Table

- iptables -t filter -L INPUT
- iptables -t raw -L OUTPUT
- iptables -t nat -L POSTROUTING
- iptables -t mangle -L PREROUTING

Default Chains

- filter: INPUT, FORWARD, OUTPUT
- nat: PREROUTING, OUTPUT, POSTROUTING
- mangle: PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING
- raw: PREROUTING, OUTPUT

Chain Management

- append a rule to a chain in a table, -A
- insert a rule into a chain in a table, -I
- delete a rule from a chain in a table, -D
- replace a rule in a chain in a table, -R
- flush all rules from a chain in a table, -F
- create a new (user) chain in a table, -N
- delete a user chain from a table, -X
- rename a user chain in a table, -E

More Chain Management

- list the rules of a chain in a table, -L
- print the rules of a chain in a table, -S
- zero packet and byte counts of a chain, -Z
- set the policy of a chain in a table, -P

Examples

- iptables -t mangle -D INPUT 3
- iptables -F OUTPUT
- iptables -X MYCHAIN
- iptables -t filter -N MYCHAIN
- iptables -E MYCHAIN YOURCHAIN
- iptables -S

Chain Policy

- is the default behavior to take on a packet if there was no rule that matched it
- and only built-in (non-user) chains can (must) have policies

Setting the Policy on a Chain

- iptables -t filter -P INPUT ACCEPT
  - same as iptables -P INPUT ACCEPT
- iptables -P FORWARD DROP

Allowable Policies

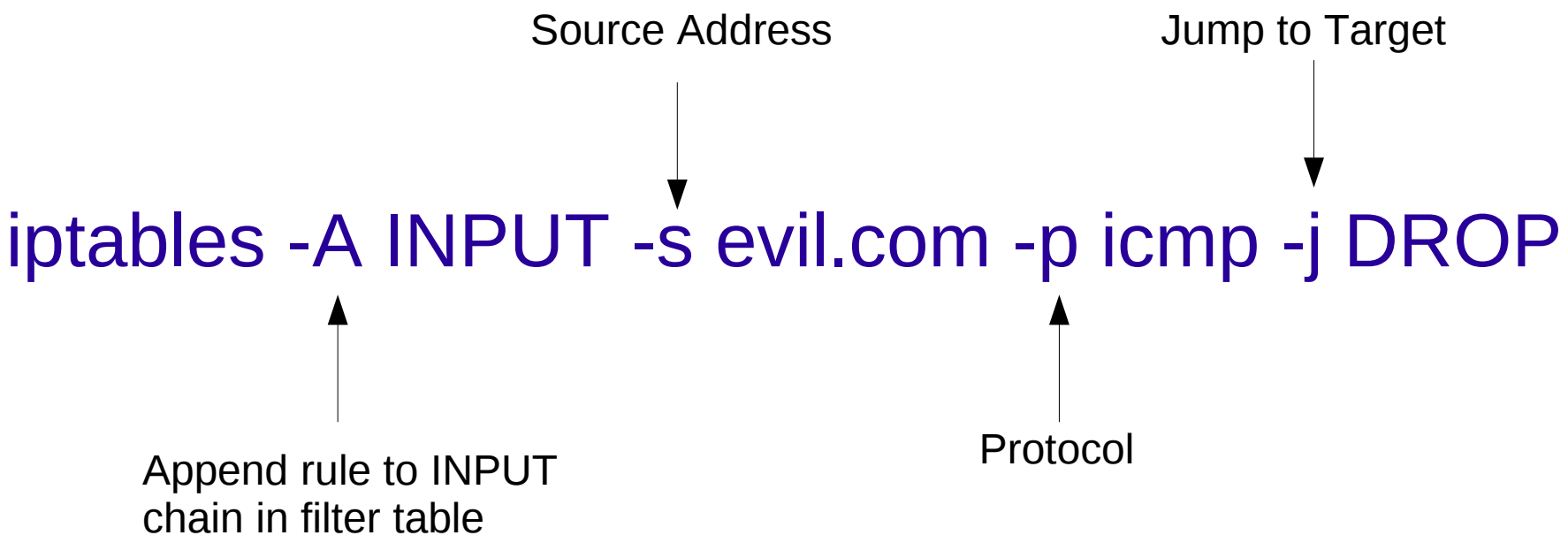
- ACCEPT: let the packet through
- DROP: drop the packet into the bit bucket

Think of a chain's policy (DROP or ACCEPT) as the last rule in the chain.

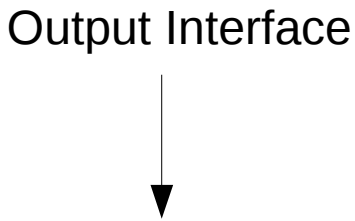
Rules

- a rule specifies criteria for a packet and a target
- if the packet does not match, the next rule in the chain is the examined
- if it does match, then the next rule is specified by the value of the target

Example
(drop pings from a host)



Example
(IP Masquerading)



iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

Example
(Port Forwarding)

```
iptables -t nat -A PREROUTING -p tcp --dport 80
-j REDIRECT --to-port 8080
```

Example
(Outgoing SSH Traffic)

iptables -A OUTPUT -p tcp -dport 22 -o eth0 -j ACCEPT

iptables -A INPUT -i eth0 -m state --state ESTABLISHED -j ACCEPT

Example
(Drop All http Packets from Subnet)

```
iptables -A INPUT -p tcp -dport 80
-s 121.57.31.0/24 -d cs.senecac.on.ca -j DROP
```

Example
(Set QoS on FTP)

Quality of Service

```
iptables -t mangle -A POSTROUTING -p tcp
--sport 20 -j TOS --set-tos 8
```

Type Of Service

Maximum Throughput

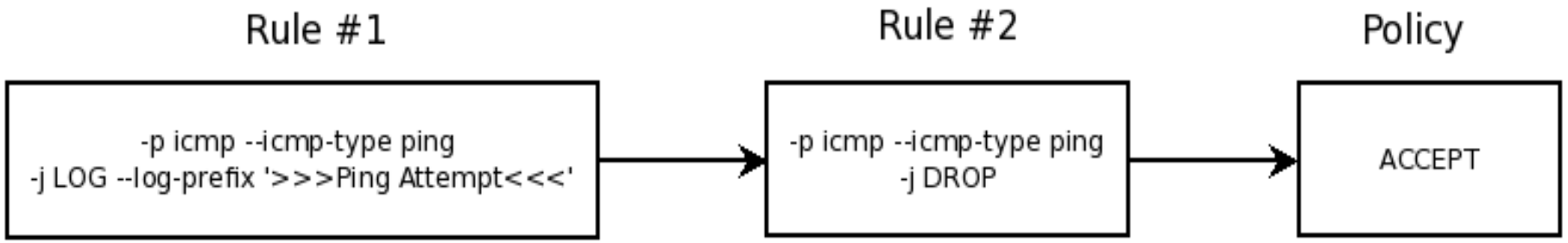
Example
(Logging/Dropping Ping Attempts)

```
iptables -P INPUT ACCEPT

iptables -I INPUT -p icmp --icmp-type ping
-j DROP

iptables -I INPUT -p icmp --icmp-type ping
-j LOG --log-prefix '>>>Ping Attempt<<<'
```

INPUT chain of filter table



Adding a User Chain

```
iptables -N JOHN

iptables -A JOHN -p icmp --icmp-type ping -j LOG

iptables -A JOHN -p icmp --icmp-type ping -j DROP

iptables -I -j JOHN
```

Saving/Restoring Your Setup

iptables-save
    dump iptables rules to standard output

iptables-restore
    restore iptables rules from standard input

Build Your Own Firewall

place all your rules inside a BASH script and run it as root

have a look at Fedora's iptables init script in /etc/init.d/iptables

NetFilter

http://netfilter.org/

Part B
A Harsh Introduction to iptables

TCP Match Options

-p tcp -- dport 80
-p tcp -- sport 22
-p tcp -- syn
-p tcp -- tcp-flags ACK,FIN,SYN SYN
-p tcp -- tcp-option 8

UDP Match Options

-p udp --dport 53
-p udp --sport 53

ICMP Match Options

-p icmp --icmp-type ping

Match Option Modules

-m state --state ESTABLISHED,RELATED
    also NEW or INVALID
-m limit --limit 3/hour
-m mac --mac-source 00:11:22:33:44:55

Many More Match Option Modules

-m addrtype
-m ah
-m comment
-m connbytes
-m connlimit
-m conntrack
-m connmark
-m dccp
-m dscp
-m ecn
-m ecp
-m hashlimit
-m helper
-m iprange
-m length
-m limit
-m mark
-m multiport
-m owner
-m physdev
-m pkttype
-m policy
-m quota
-m rateest

etc. etc.

Target Options

-j ACCEPT
-j DROP
-j QUEUE
-j RETURN

Extended Target Modules

-j REJECT
-j LOG
    --log-level
    --log-ip-options
    --log-prefix
    --log-tcp-options
    --log-tcp-sequence

Listing Options

iptables -L

- -v (verbose)
- -n (numeric format)
- -t (table)
- -x (expand numbers)

Parameter Options

-s (source address)
-d (destination address)
-j (jump to target)
-i (input interface)
-o (output interface)
-p (protocol)
-f (match fragmented packets)
-c (clear counters on match)

Examples

iptables -A INPUT -s 0/0 -d 1.2.3.4 -m state --state NEW -p tcp --dport 80 -i eth0 -j ACCEPT

iptables A OUTPUT -d 0/0 -m state --state NEW -p tcp -m multiport --dport http,https -o eth0 -j ACCEPT

Load Balancing Example

-A PREROUTING -i eth0 -p tcp --dport 80 -m state --state NEW -m nth --counter 0 --every 4 --packet 0 -j DNAT --to-destination 192.168.0.1:80

-A PREROUTING -i eth0 -p tcp --dport 80 -m state --state NEW -m nth --counter 0 --every 4 --packet 1 -j DNAT --to-destination 192.168.0.2:80

-A PREROUTING -i eth0 -p tcp --dport 80 -m state --state NEW -m nth --counter 0 --every 4 --packet 2 -j DNAT --to-destination 192.168.0.3:80

-A PREROUTING -i eth0 -p tcp --dport 80 -m state --state NEW -m nth --counter 0 --every 4 --packet 3 -j DNAT --to-destination 192.168.0.4:80

Restricting Connections

iptables -A FORWARD -m state --state NEW -p tcp -m multiport --dport http,https -o eth0 -i eth1 -m limit --limit 50/hour --limit-burst 5 -j ACCEPT

Matching Data Strings

```
iptables -A FORWARD -m string --string '.com' -j DROP

iptables -A FORWARD -m string --string '.exe' -j DROP
```

Setting Transfer Quotas

iptables -A INPUT -p tcp -m quota --quota 2147483648 -j ACCEPT

iptables -A INPUT -j DROP

Time-Based Rules

iptables -A FORWARD -p tcp -m multiport --dport http,https -o eth0 -i eth1 -m time --timestart 12:30 --timestop 13:30 –days Mon,Tue,Wed,Thu,Fri -j ACCEPT