

# Optimización del ancho de banda (Introducción al Firewall de Linux)



Christian Benvenuti

christian.benvenuti@libero.it

La Paz, Bolivia, 11-15/5/2009



# Before we start ...

- Are you familiar with *iptables* or firewalling in general?
- Is there anything specific that you would like to learn on firewalling?
- Are you comfortable with IP addresses, L4 port numbers, well known port numbers, L3/L4 protocol headers, etc ?
- Do you know how to check and/or change the kernel configuration? (including upgrading the kernel)

# This is not “Learn iptables in 2h”

- In this class I will not show you the best and latest and most efficient list of *iptables* rules that you can use to protect your network and reduce or limit bad uses of the bandwidth ...
- ... but I'll give you the instruments to define such ruleset. I'll show you the architecture of iptables/netfilter so that you can decide (and fully understand) your own configuration based on your exact needs.

# Agenda

- Survey
- Quick introduction to firewalls
  - Classification of firewall types
  - Most common firewalls (SW & HW)
- *iptables*/Netfilter
  - Filtering, Mangling, Connection tracking, NAT, ...
  - Examples
  - *iptables* & GUI
- Exercises

# Yes/No Linux/Firewall: why?

- For those that are NOT using a Linux firewall, what are the reasons?
  - Missing features?
  - Not performing well enough?
  - Hard to configure?
  - ...
- For those using a Linux firewall:
  - How do you configure it?
    - Command line
    - GUI (which one?)
    -
  - What do you like the most of it?

- For those using a firewall:
  - what are the features that you find more useful?
  - what are the features that you find more difficult to understand/configure?

# Quick introduction to firewalls: the Role

- Firewalls are not the solution, but represent an important component of the solution, which includes ...
  - Intelligence into the network/infrastructure
    - **Firewalls**, Intrusion Detection Systems, etc
  - Intelligence into the hosts
    - Authentication/Authorization Systems, Filesystem security, etc
  - Intelligence into the applications
    - Proxies, SSL, etc
  - User education
  - ...

# Quick introduction to firewalls: Classification

- **Cheap**, Fairly priced, toooooo expensive
  - Selling price Vs Total cost of ownership
- Hardware, **Software**, Hybrid
- Stateless, **Stateful**

# How must/should it be?

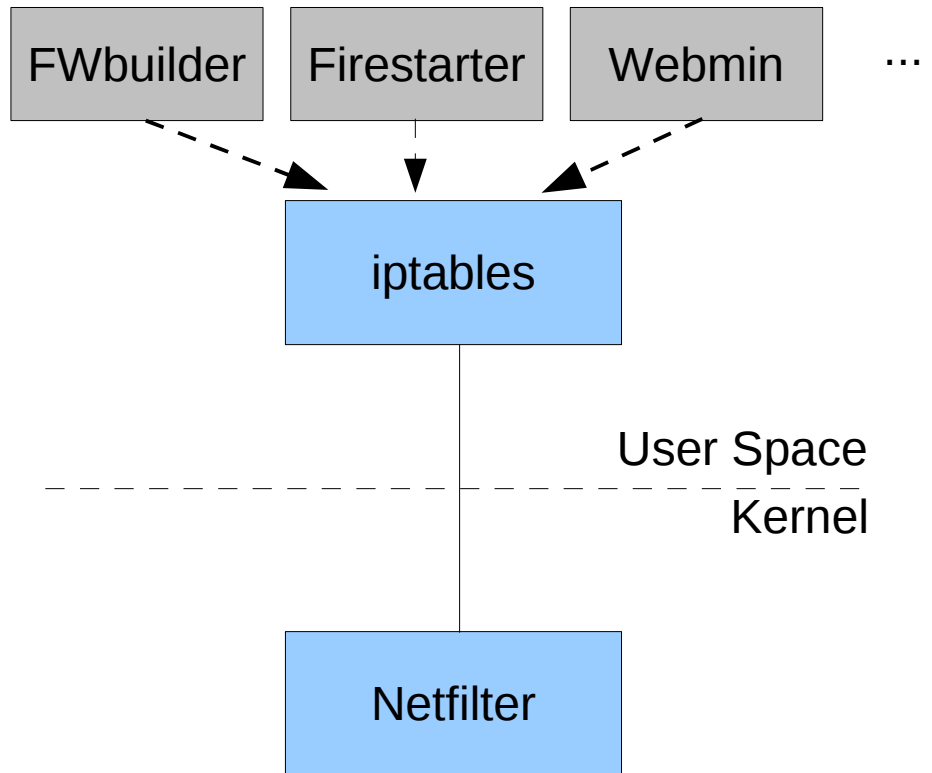
- **MUST BE:** Able to work on modern networks and handle modern issues.
  - Like anti-viruses ... firewalls are useless if they do not allow you to run your preferred protocols and applications, or they do not protect you against modern network security issues/problems/attacks.
- **SHOULD BE:** Well documented and actively supported

# Most common firewalls

- Specialized Hardware Devices (Juniper, Cisco, Nokia, ...)
- Personal Firewalls
  - Free Personal Firewalls
    - MacOS, WinXP/Vista, **Linux**, {Free,Open,Net}BSD, ...
  - Commercial Personal Firewalls
    - Panda, Norton, CheckPoint, ...

# ... and finally ... iptables/Netfilter

- Open Source
- Pretty modular
- Actively supported
- Enough documented
- Runs on LINUX



# When you configure a firewall ...

(the same applies to most net services, but security services can't afford a misconfiguration ...)

- ... you must know what you are doing
- You must know how to verify whether the configuration is correct and does what it is supposed to do.
  - Tools like *nmap* can be very useful
- etc

# Here is the plan ...

- Get familiar with the core components of Netfilter:
  - Connection Tracking (CT)
  - Network Address Translation (NAT)
  - Filtering
  - Mangling
- Get familiar with the kernel configuration of Netfilter
- Get familiar with the *iptables* command and its syntax
- Examples/Exercises
- Lab
- Intro to GUIs

# Core Netfilter Components

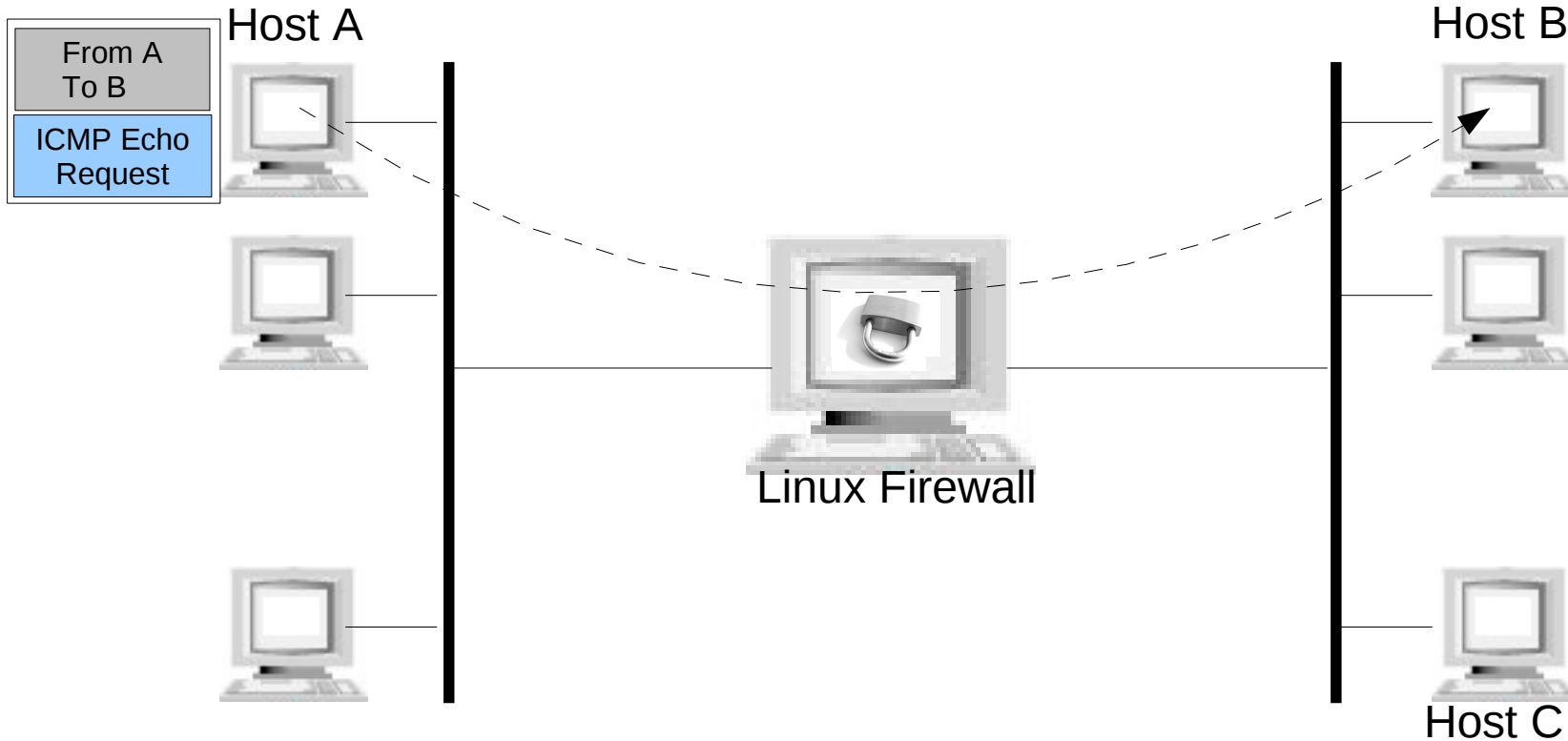
- **Connection Tracking**
- Network Address Translation (NAT)
- Filtering
- Mangling

# Connection Tracking (1/5)

- Connection Tracking is what makes Netfilter stateful
  - Keeps track of all the 'connections' that traverse the firewall
  - Connection Tracking is not filtering, but:
    - It blocks illegal packets (we will see an example with ICMP)
    - It makes it possible for the administrator to decide whether to filter those packets that are likely to be illegal
  - It makes it possible to implement stateful NAT (i.e., NAT depends on conntrack)

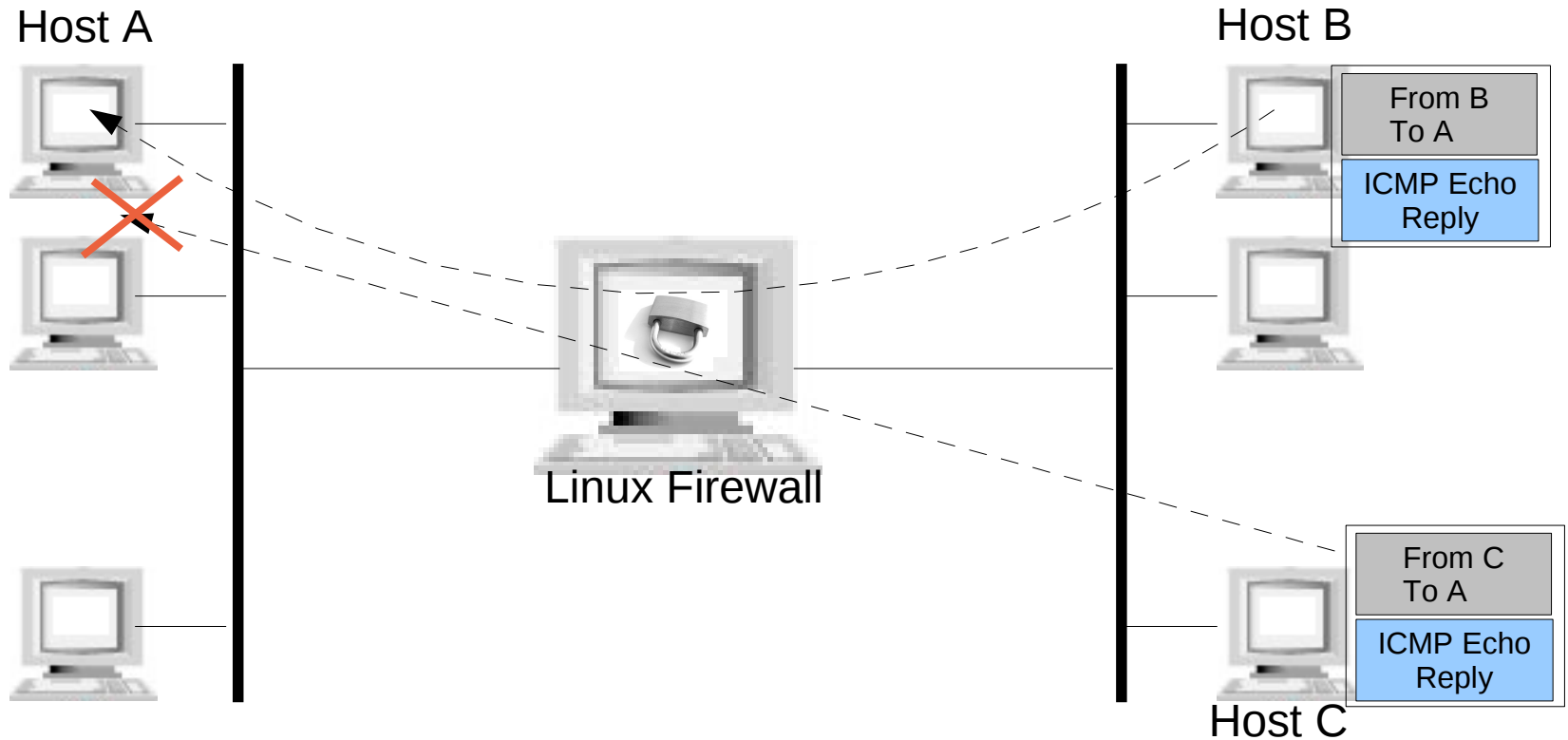
# Why is tracking connections useful?

(example of stateless FW ) (1/2)

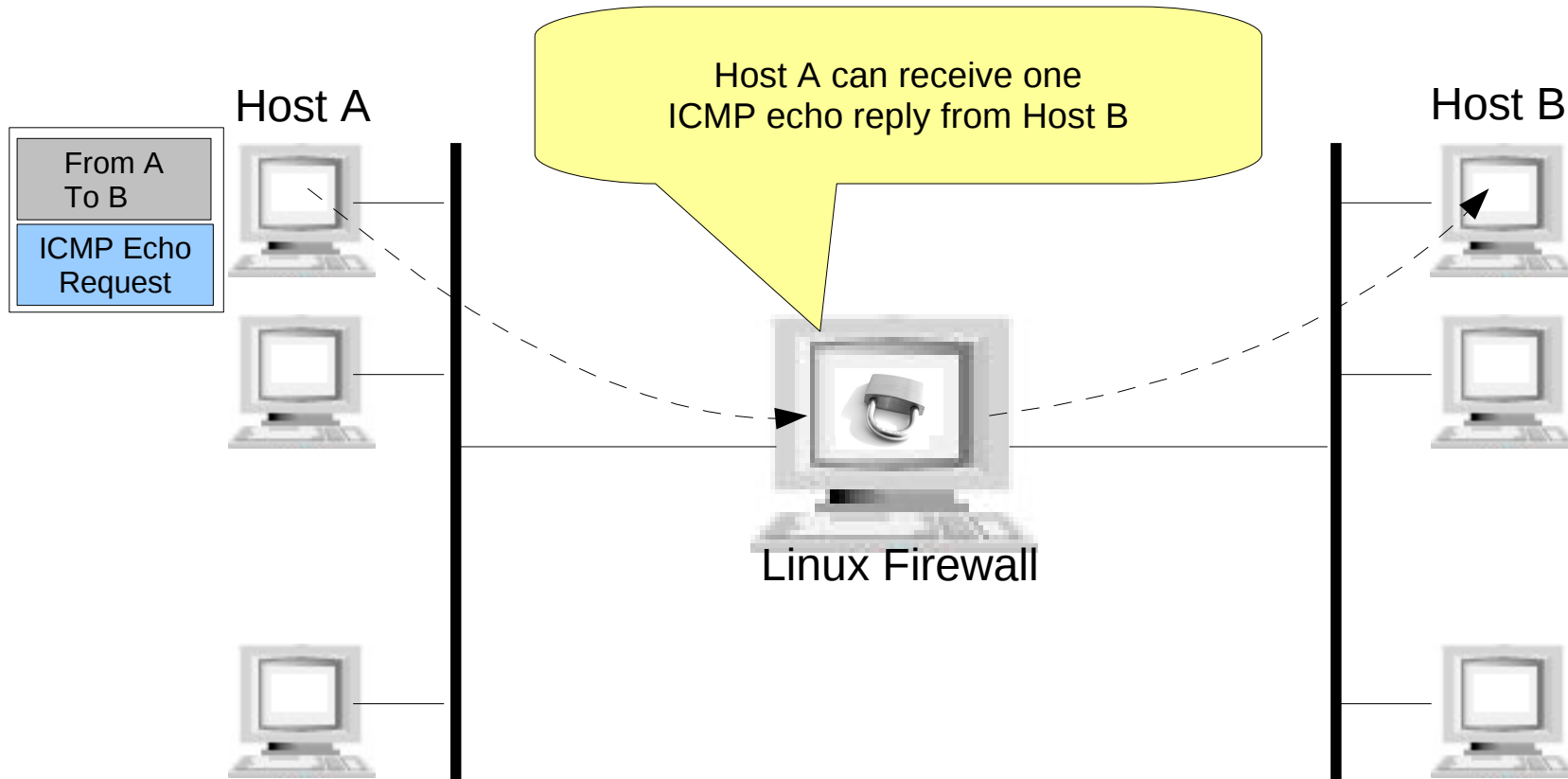


# Why is tracking connections useful?

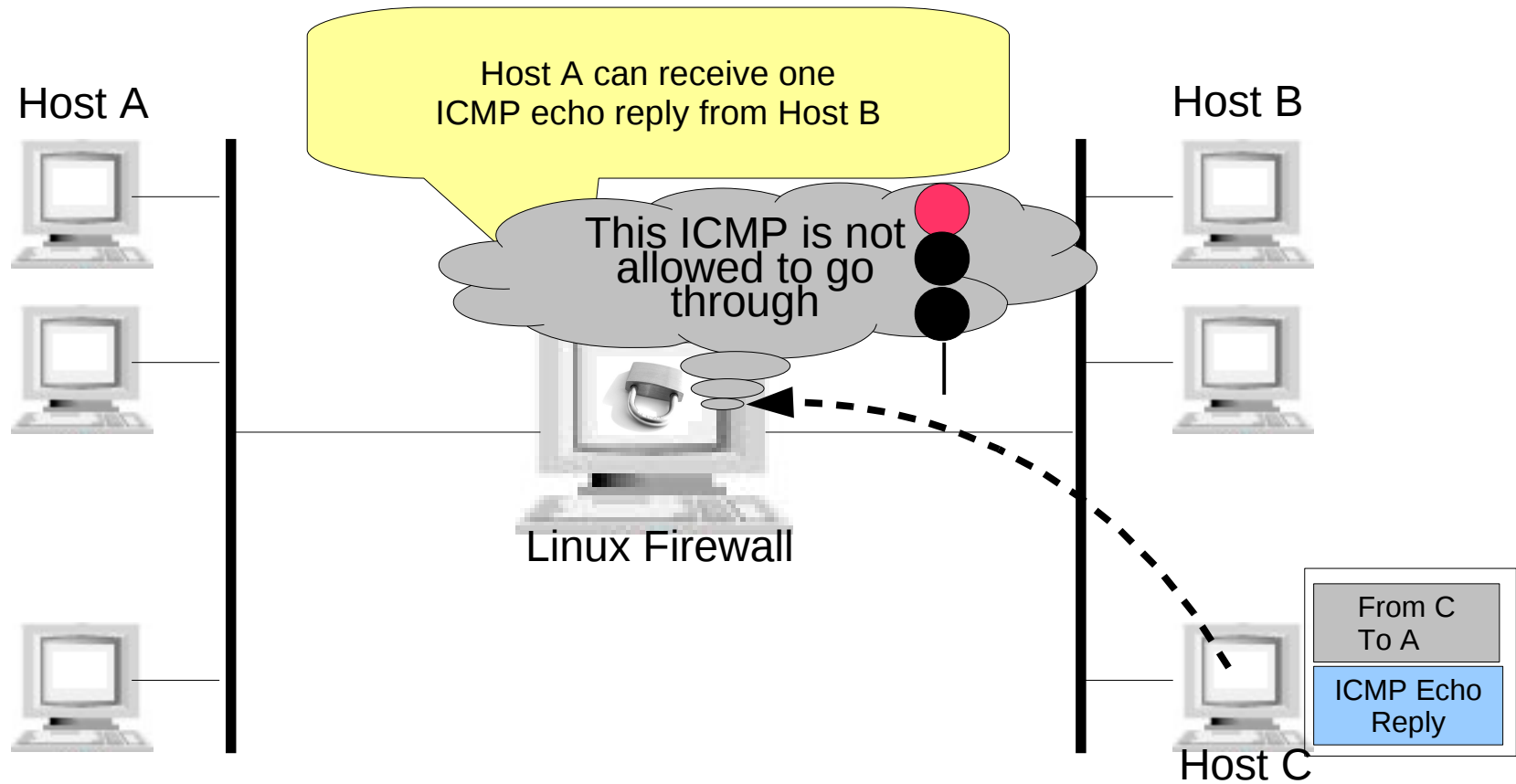
(example of stateless FW ) (2/2)



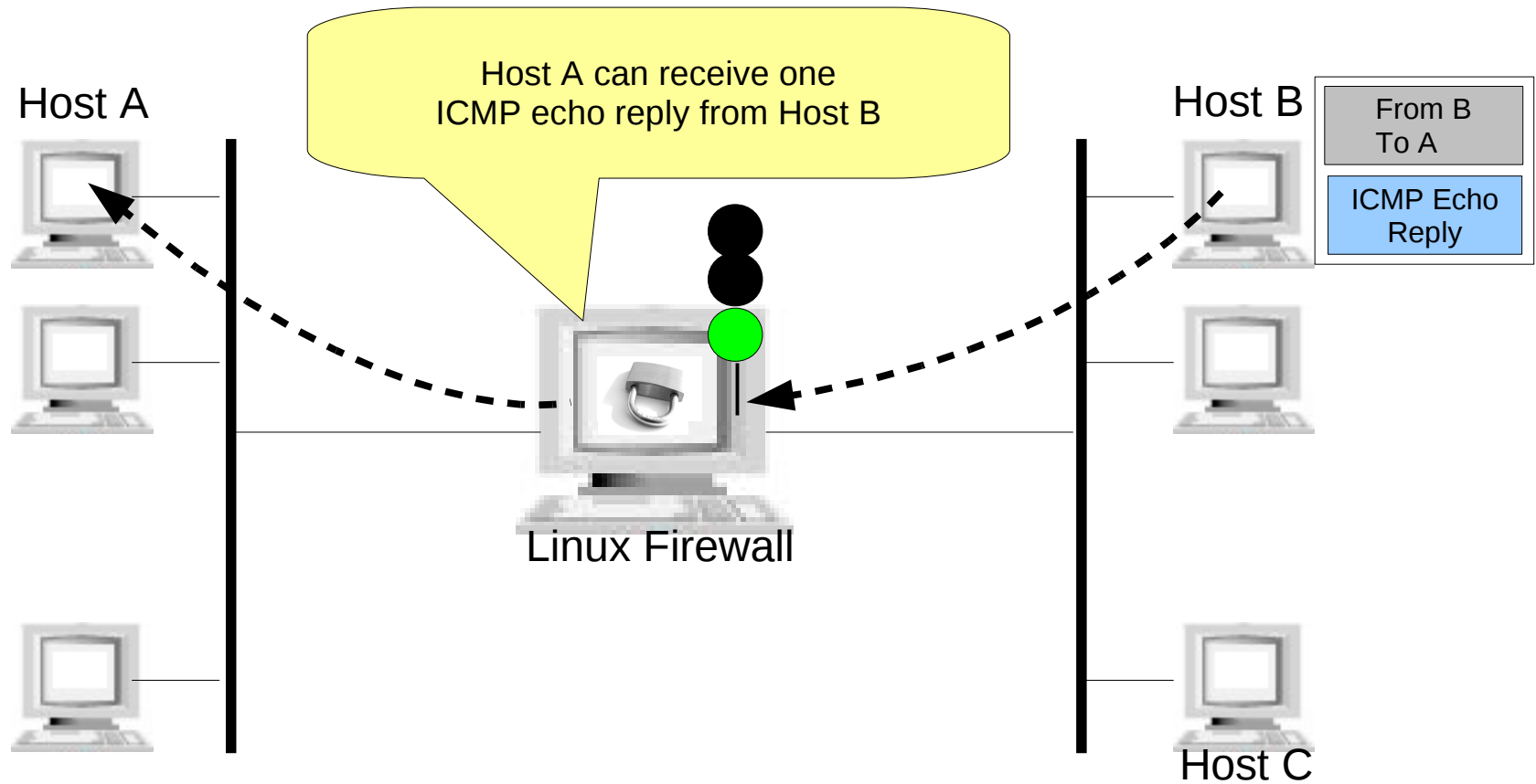
# Connection Tracking in action (1/4)



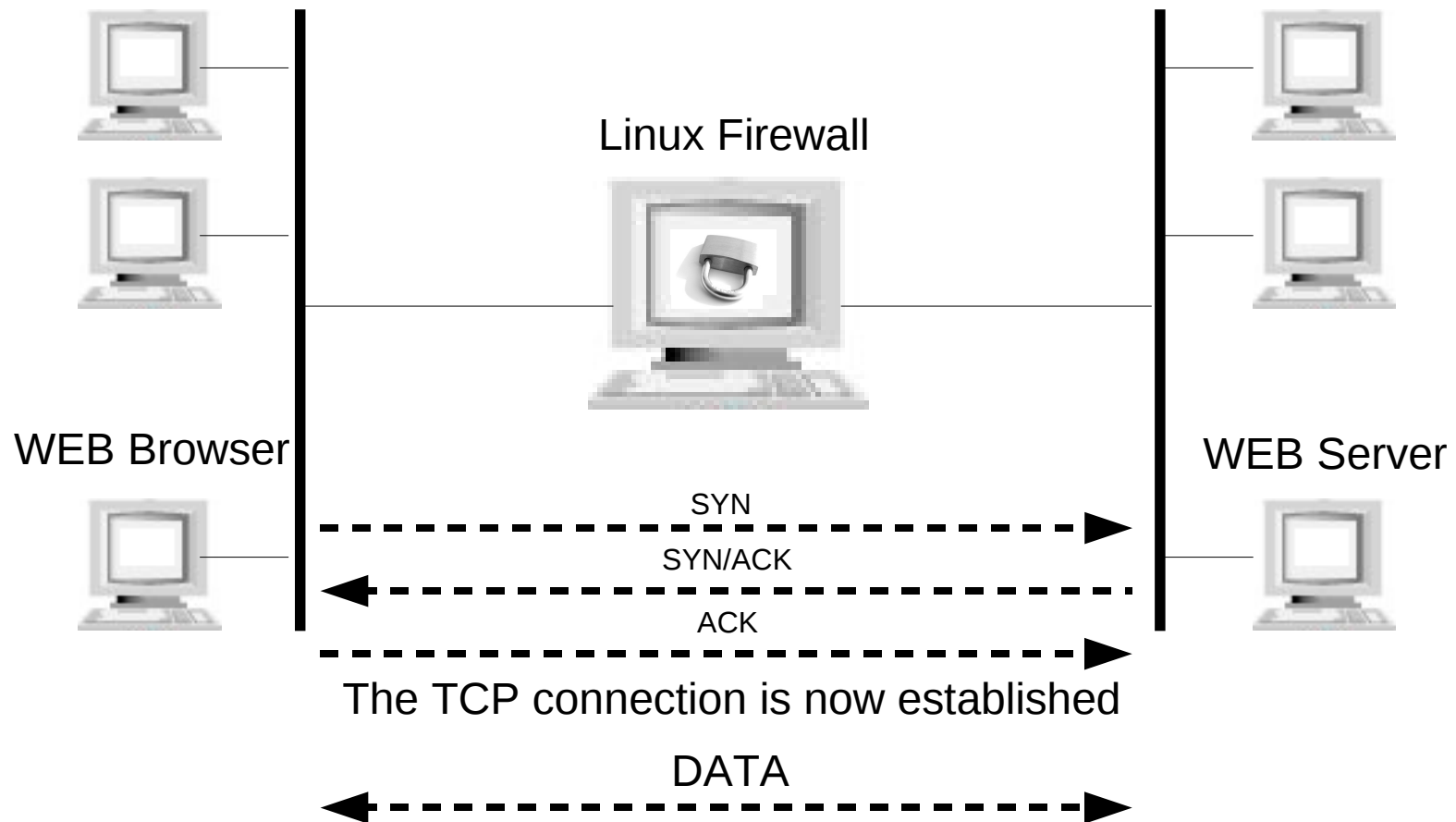
# Connection Tracking in action (2/4)



# Connection Tracking in action (3/4)



# Connection Tracking in action (4/4)



# Connection Tracking

- Often referred to as 'conntrack'
- Modular design that allows you to add support for new Transport Protocols and new Applications easily

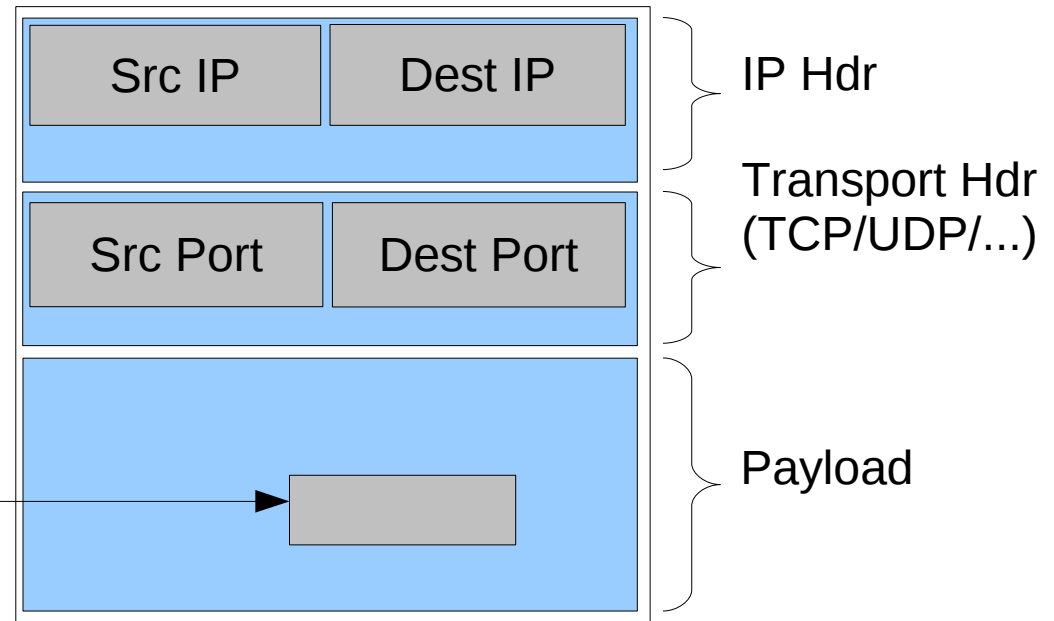
# Core Netfilter Components

- Connection Tracking
- **Network Address Translation (NAT)**
- Filtering
- Mangling

# Network Address Translation (1/3)

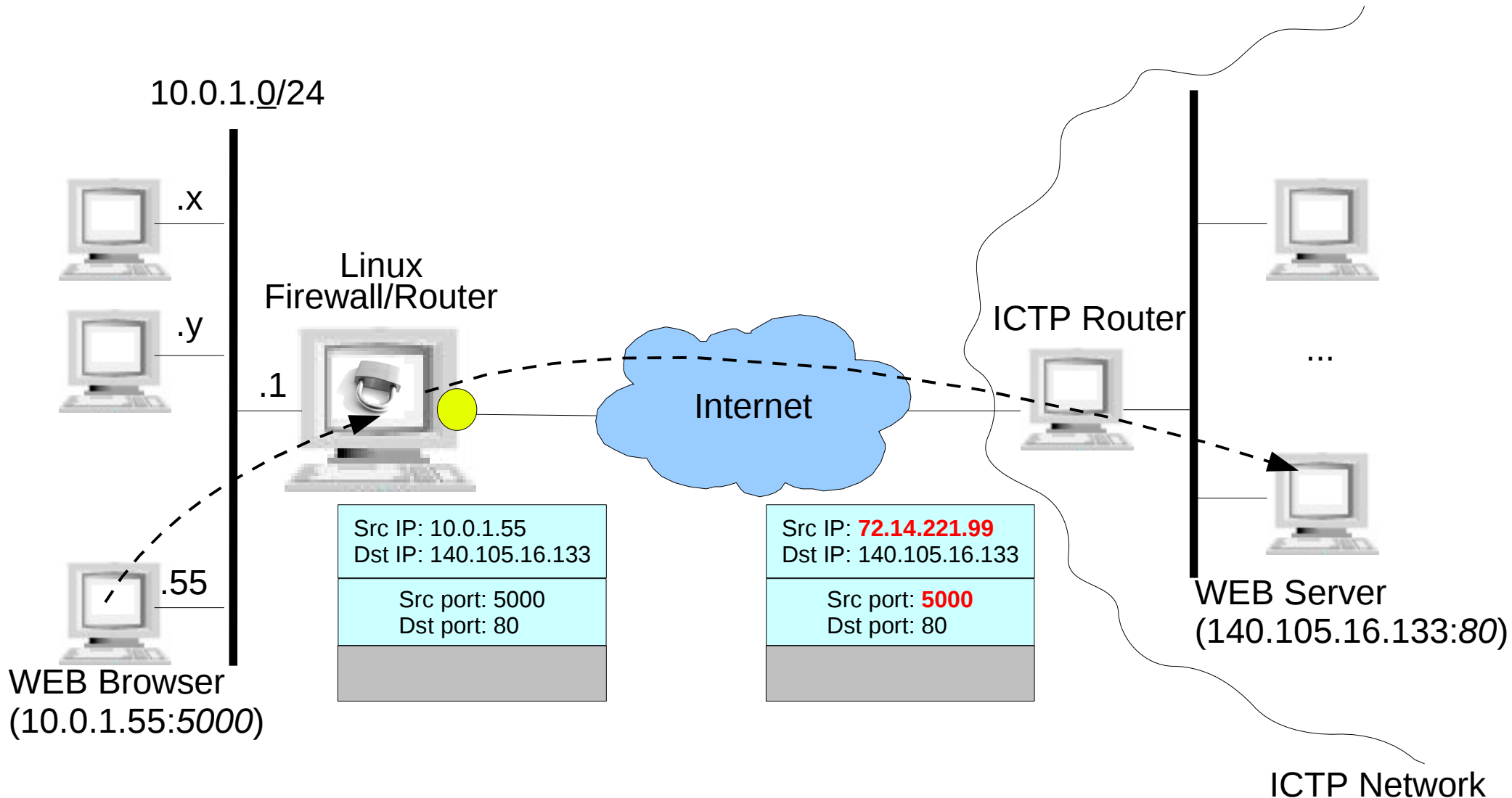
- Source NAT (SNAT)
  - Masquerading/PAT is just a special case
- Destination NAT (DNAT)
- Redirect
- ...

This requires the help of application helpers (modules)



# Network Address Translation (2/3)

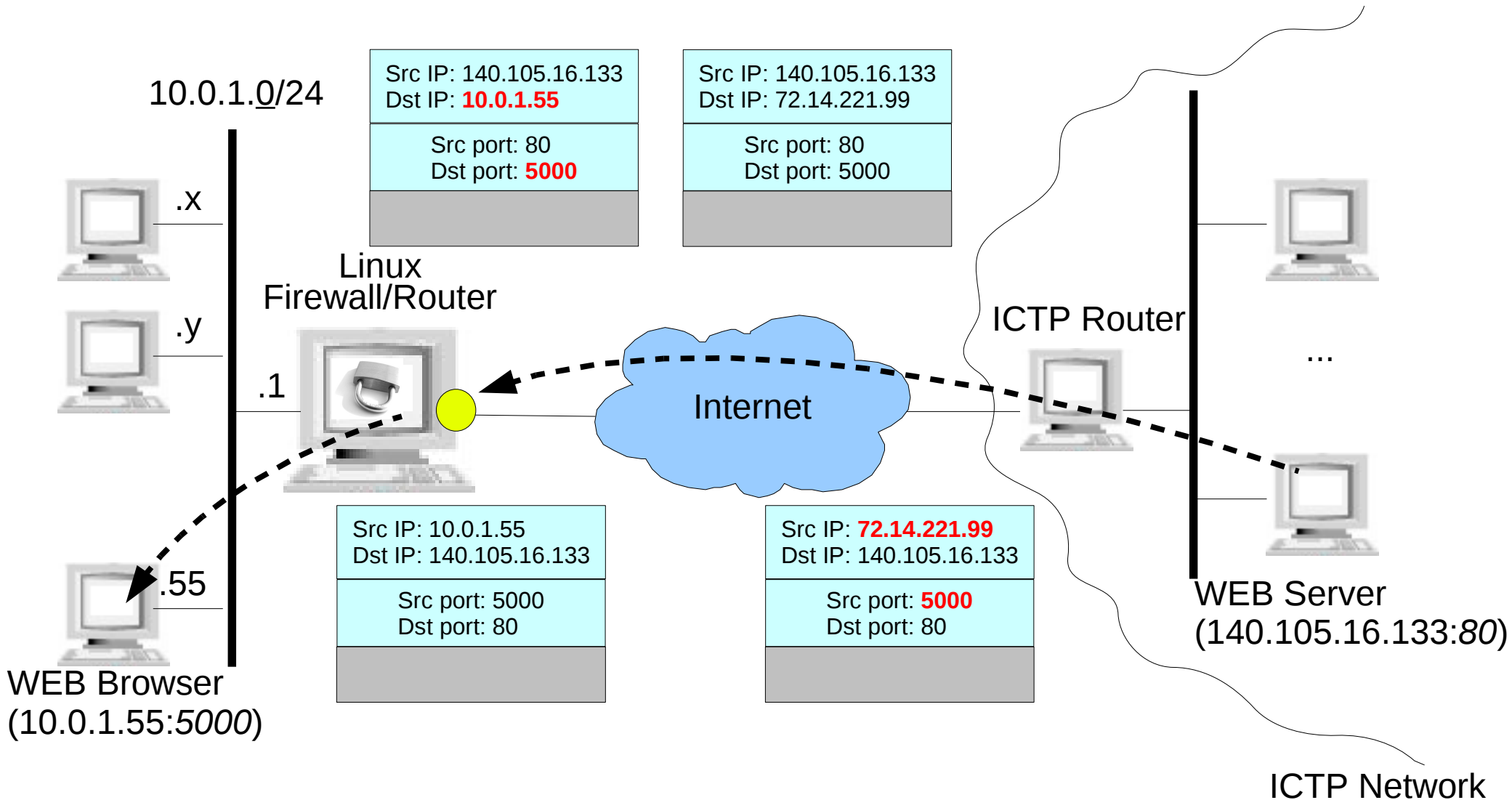
## The famous MASQUERADE target



● Interface with public IP address 72.14.221.99 that is masqueraded

# Network Address Translation (3/3)

## The famous MASQUERADE target



● Interface with public IP address 72.14.221.99 that is masqueraded

# Core Netfilter Components

- Connection Tracking
- Network Address Translation (NAT)
- **Filtering**
- Mangling

# Filtering (1/3)

- Filtering is the most common task a firewall is used for.
  - Stateless firewalls only provide stateless filtering (i.e., no connection tracking means that neither stateful filtering nor stateful NAT are available).
- You can filter based on almost any field of the network protocol stack headers (and also on the payload)
- You can filter based on external (context) factors too, such as the user that generates the traffic, the bandwidth usage, etc.

# Filtering (2/3)

- The configuration of a filtering firewall consists of two main parts:
  - Default policy
  - Exceptions to the default policy

# Filtering (3/3)

## Firewall rules, aka ACLs

- An ACL must include at least the following two pieces of information:
  - The traffic to match
  - What to do with the traffic that matches

# Core Netfilter Components

- Connection Tracking
- Network Address Translation (NAT)
- Filtering
- **Mangling**

# Mangling

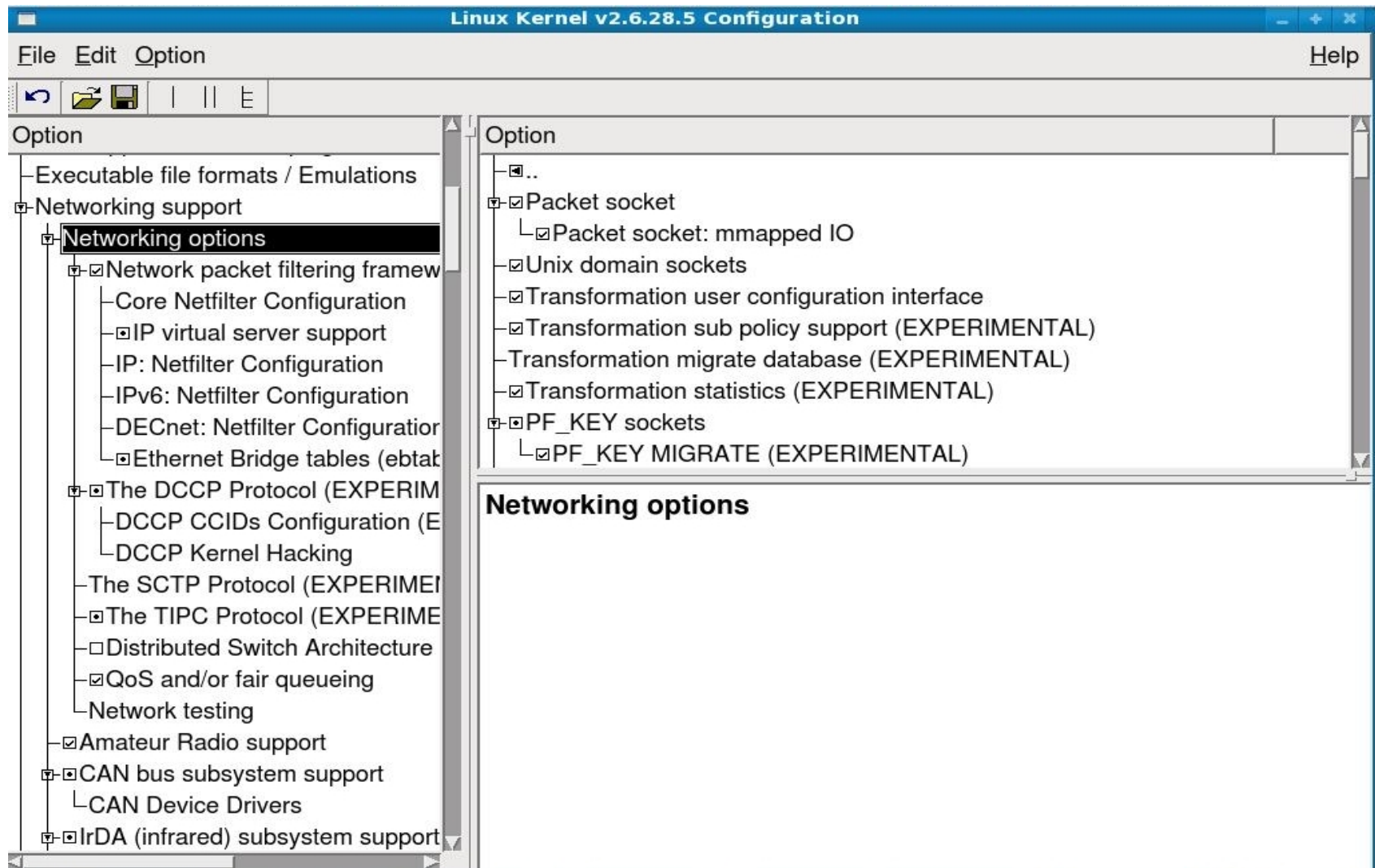
- Mangling provides two main options:
  - The ability to change the content of specific parts (i.e., header fields) of a data packets in order to influence/change the way the latter will be treated on their path to destination
    - TTL, TOS, DSCP, ...
  - The ability to assign to the data packets some sort of tags that can be used by other (kernel) applications.
    - Examples of consumers are the routing tables and Traffic Control.

# Here is the plan ...

- Get familiar with the core components of Netfilter:
  - Connection Tracking (CT)
  - Network Address Translation (NAT)
  - Filtering
  - Mangling
- **Get familiar with the kernel config of Netfilter**
- Get familiar with the *iptables* command and its syntax
- Examples/Exercises
- Lab
- Intro to GUIs

# Configuring the kernel

- Networking
  - Networking options
    - Network packet filtering framework (Netfilter)



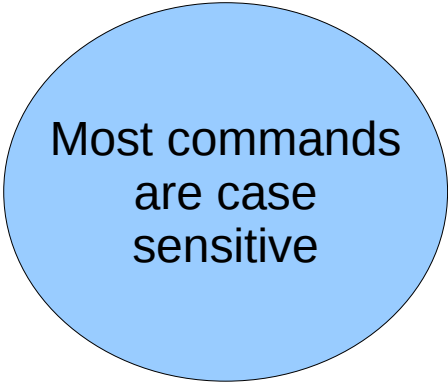
# Here is the plan ...

- Get familiar with the core components of Netfilter:
  - Connection Tracking (CT)
  - Network Address Translation (NAT)
  - Filtering
  - Mangling
- Get familiar with the kernel config of Netfilter
- **Get familiar with the *iptables* command and its syntax**
- Examples/Exercises
- Lab
- Intro to GUIs

# Syntax of an *iptables* rule

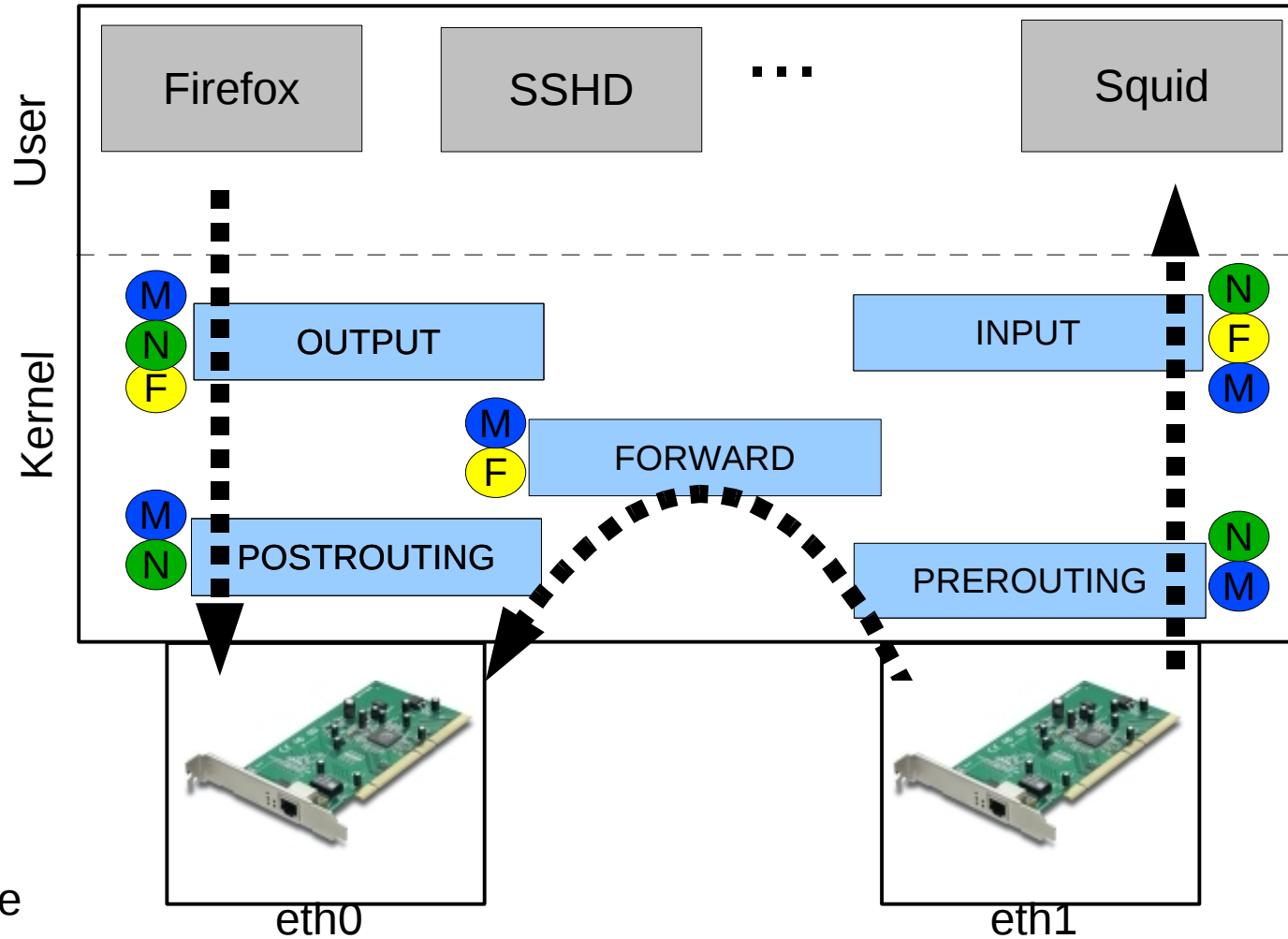
*iptables* -t <TAB> <OP> <HOOK> <MATCH> ... <MATCH> -j <TARGET>

- Type of rule
  - Table (filter, nat, mangle, raw)
- Are you adding, removing or modifying a rule?
  - Operation (A, D, I, R, ...)
- When to enforce the rule
  - Hook point (INPUT, OUTPUT, ...)
- What traffic to match with
  - Matches (many here, both implicit and explicit)
- What to do with the matching traffic
  - Target (ACCEPT, DROP, ...)



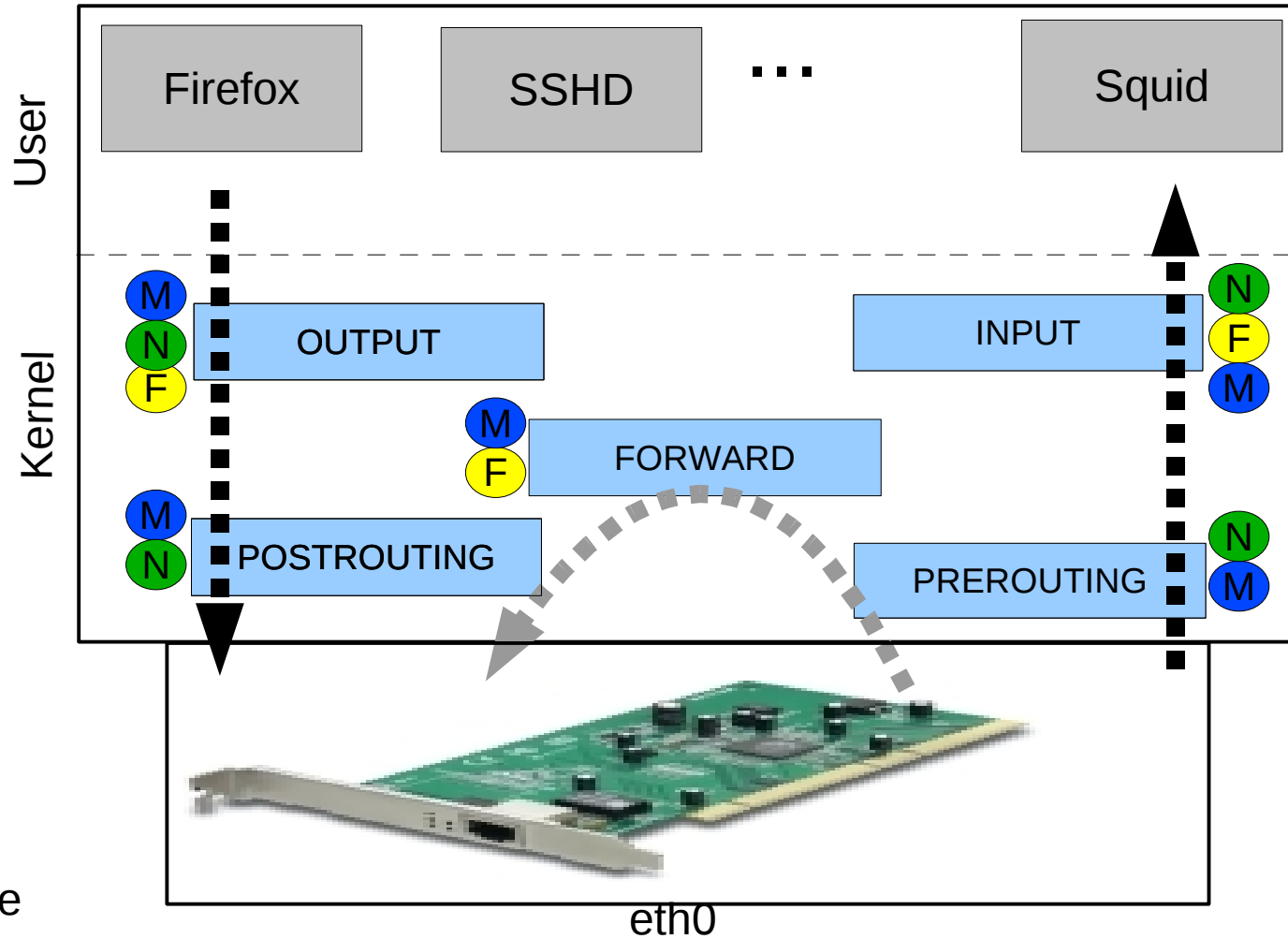
Most commands  
are case  
sensitive

# Tables and Hooks/Chains



(The Raw table is not shown in the figure)

# Tables and Hooks/Chains



- F Filter table
- M Mangle table
- N NAT table

(The Raw table is not shown in the figure)

# Example 1

***iptables -t <TAB> <OP> <HOOK> <MATCH> ... <MATCH> -j <TARGET>***

- I would like to block ingress ICMP echo request messages

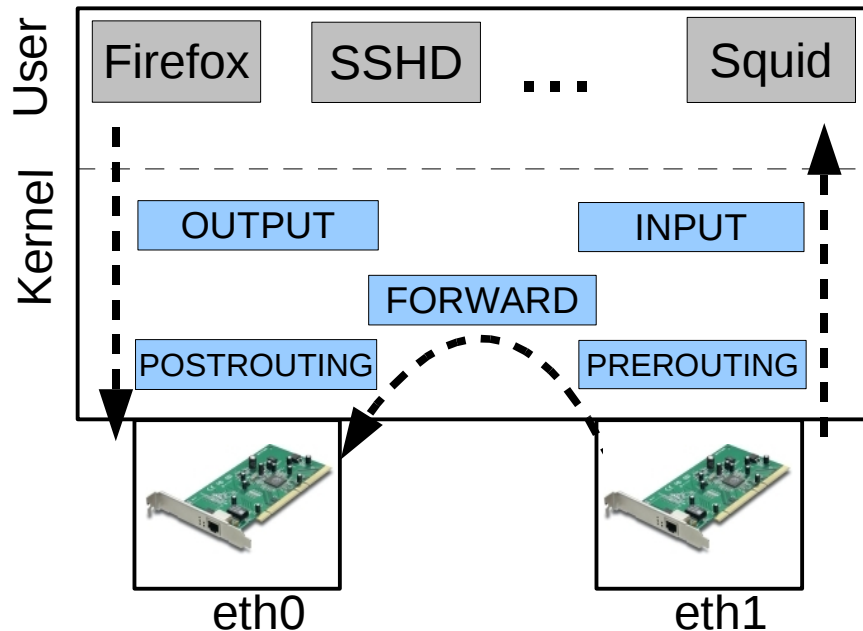


Table	-t filter
Operation	-A
Hook	INPUT
Match/es	-p icmp --icmp-type echo-request
Target	-j DROP

```
#iptables -t filter -A INPUT -p icmp --icmp-type echo-request -j DROP
```

# Example 2

`iptables -t <TAB> <OP> <HOOK> <MATCH> ... <MATCH> -j <TARGET>`

- I would like to let go through (i.e., allow) the traffic that is addressed to the WEB server with IP 192.168.3.1

From where? What interfaces? Is that IP local to the FW or remote?

Let's assume 1) the WEB server is reachable through eth0 (internal net) 2) the hosts are reachable through eth1 (external net)

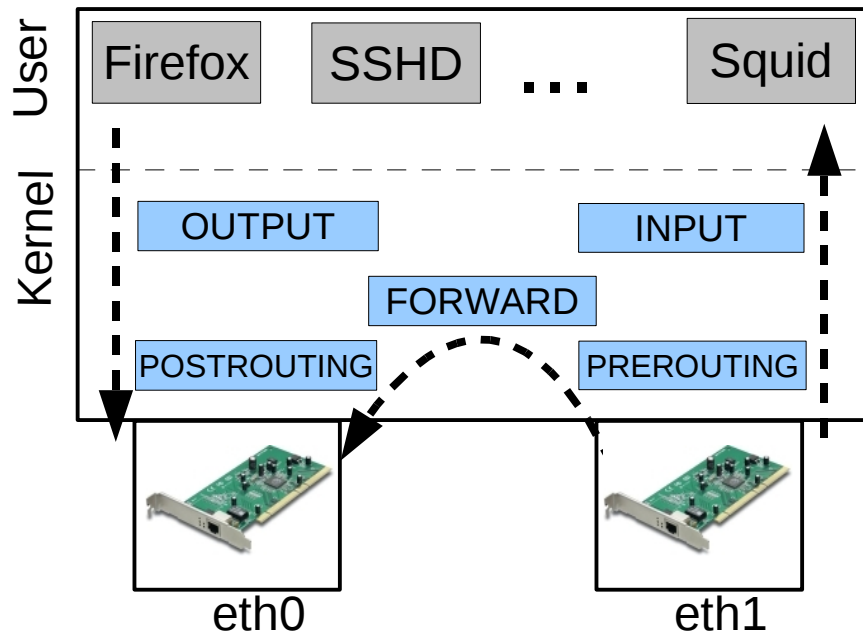


Table	-t filter
Operation	-A
Hook	FORWARD
Match/es	-p tcp --dport 80
Target	-j DROP

# Example 3

`iptables -t <TAB> <OP> <HOOK> <MATCH> ... <MATCH> -j <TARGET>`

- I would like to set the value of the DSCP field (in the IPv4 header) to 2 for the traffic that is transmitted out the interface *eth1*

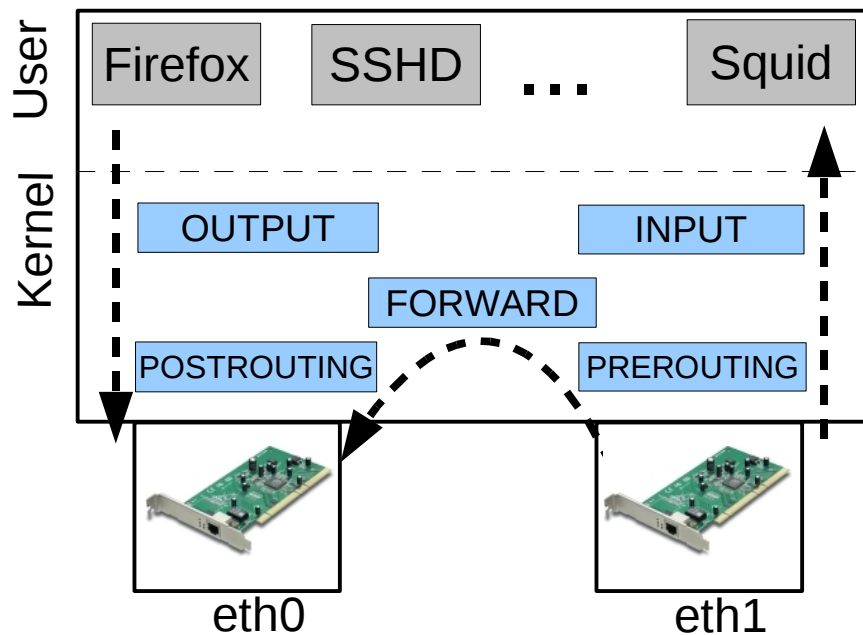


Table	-t mangle
Operation	-A
Hook	POSTROUTING or OUTPUT?
Match/es	-o eth1
Target	-j DSCP --set-dscp 2

# Example 4

`iptables -t <TAB> <OP> <HOOK> <MATCH> ... <MATCH> -j <TARGET>`

- I would like to masquerade *eth0*, but only for the traffic that originates in *eth1* (let's assume there were more than 2 interfaces)

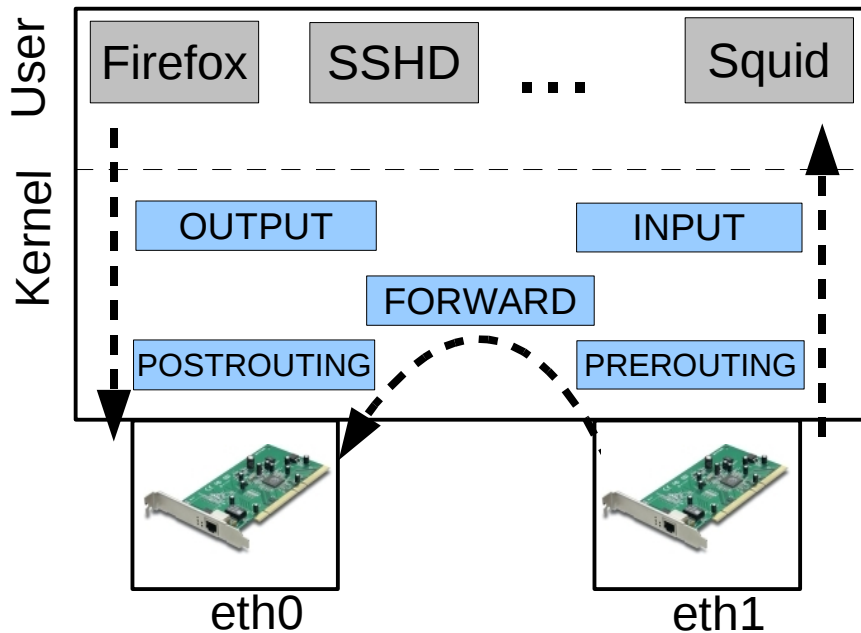


Table	-t nat
Operation	-A
Hook	POSTROUTING
Match/es	-i eth1 -o eth0 ?
Target	-j MASQUERADE

# iptables

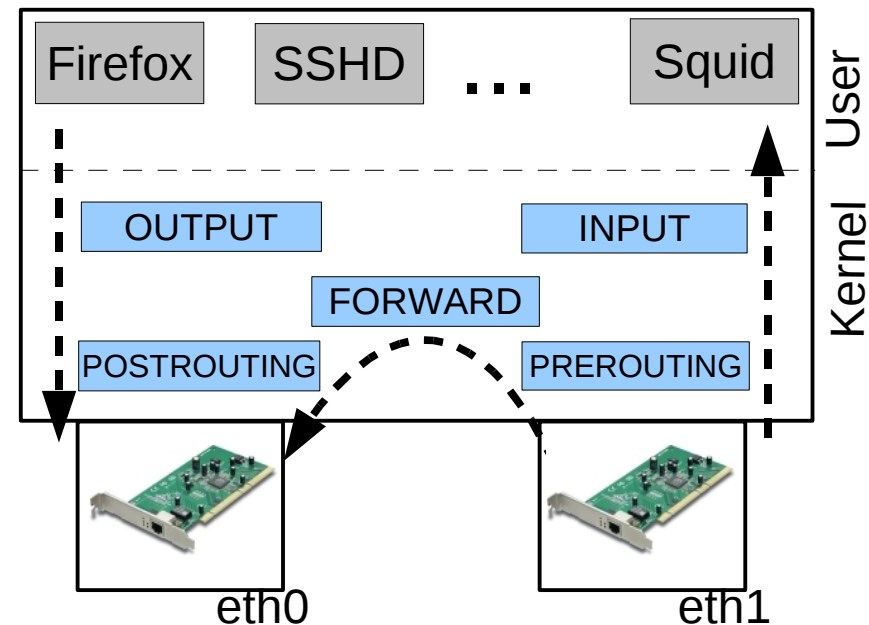
- Setting the default policy for a chain/hook  
`iptables [-t Table] -P <HOOK> {ACCEPT, DENY}`
- Creating a new (user) chain  
`iptables [-t Table] -N <chain_name>`
- Deleting a (user) chain  
`iptables [-t Table] -X <chain_name>`

## Table

{**filter**, nat, mangle, raw}

Default  
table

User chain 1  
User chain 2  
...



# iptables

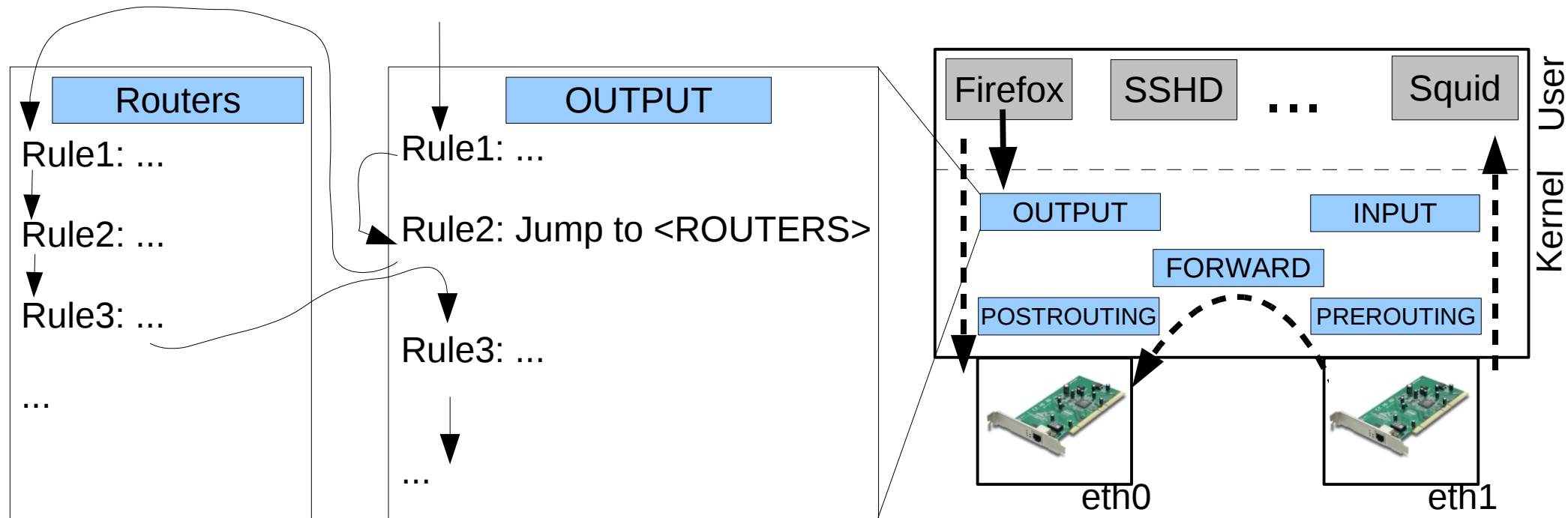
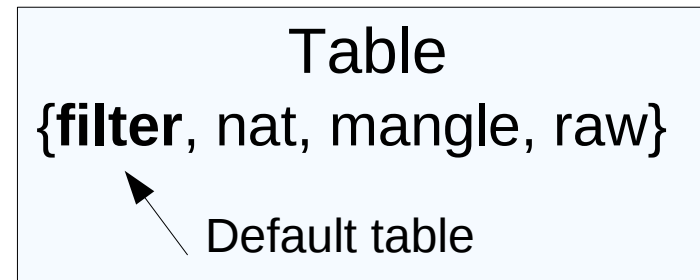
- Creating a new (user) chain

*iptables -t filter -N Routers*

- Adding rules to the new Routers chain

*iptables -t filter -A Routers ...*

- There is no default policy for user chains



# Basic commands

(see *man <command>* for the options)

- iptables
- iptables-save
- iptables-restore

Fedora systems: *service iptables status | start | stop | restart*

# iptables

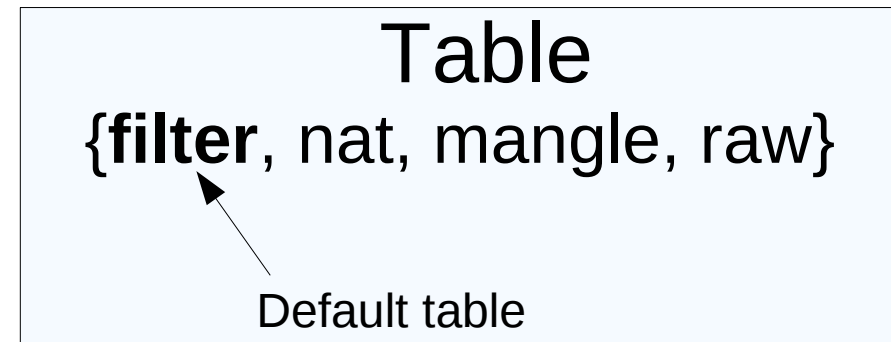
- Checking the current ruleset

`iptables [ -t Table ] [-v] -L`

Verbose (i.e., more details)

- Flush the current ruleset

`iptables [-t Table] -F`



# Save and restore the current ruleset

- iptables-save [> filename]
- iptables-restore [< filename]

# Example1: *iptables-save* on a FC10 system (basic security).

```
# iptables-save
# Generated by iptables-save v1.4.1.1 on Tue Feb 17 09:03:40 2009

*filter

:INPUT ACCEPT [0:0]

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [926:1353750]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited

COMMIT
```

See “*man iptables-save*” and test the `-c` and `_t` options

# Before we start, let's keep in mind that:

- The order of the rules is very important
- The firewall is configured differently depending on the default policy
- The rules you type in with the *iptables* command are applied right away, but ...
  
- We are going to exercise (mainly) on the configuration of a Firewall, not on the configuration of a Host (the typical requirements change).
- For each policy/requirement, there may be more than one correct solution. The solutions may differ with regards to their efficiency and/or adaptability to network topology changes and/or firewall/hosts hardware config (i.e., IPs, NIC names, MAC addresses, etc)

# A few commands you may want to get familiar with ...

- netstat -tupan
- lsof -i
- nmap
- ...

I highly recommend you playing with them and learning about the various options they provide.

# Logging

- LOG vs ULOG
- A matching LOG rule does not interrupt the firewall lookup
- Example of LOG:

```
#iptables -t filter -A FORWARD -p tcp --dport 23 -j LOG --log-prefix "Telnet: "
```

# GUI apps for configuring iptables/Netfilter

- Firewall Builder
  - <http://www.fwbuilder.org>
  - FC10 Menu: System-->Administration-->Firewall Builder
- Fedora Core 10
  - system-config-firewall
  - FC10 Menu: System-->Administration-->Firewall
- Firestarter
  - <http://www.fs-security.com>
- Webmin
  - <http://www.webmin.com/>
-

# Conclusion

- Now that you are (supposed to be) familiar with *iptables*/Netfilter
  - Define your requirements (what to allow, what to deny, what to limit, etc)
    - Online you can find good documentation on what a general good security policy is.
      - We can also sit down and define it TOGETHER.
  - Determine whether *iptables*/Netfilter can help you enforce your policy.
    - For example, does *iptables*/Netfilter support those protocols that you would like to allow/deny/limit/etc ?
    - Is there any feature that you have seen available (and useful) on other firewalls but that *iptables*/Netfilter does not support?

# Documentation

- `man {iptables, iptables-save, iptables-restore}`
- <http://www.netfilter.org/documentation>
  - In particular <http://iptables-tutorial.frozentux.net/iptables-tutorial.html>

# Copyright



- This presentation is released under the Creative Common License:
  - Attribution, Noncommercial, Share Alike 2.5
  - (<http://creativecommons.org/licenses/by-nc-sa/2.5/>)
- Attribution
  - You must attribute the work in the manner specified by the author or licensor.
- Noncommercial.
  - You may not use this work for commercial purposes.
- Share Alike.
  - If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.