Taller de Wireshark

Teoría de las Comunicaciones

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

03.09.2014

Objetivos

- Presentar Wireshark: Lo mejor que te paso en la vida.
- Trabajar el concepto de paquete (o frame) de datos.
- Visualizar y manipular en forma explícita los paquetes.
- Analizar estadísticamente una Red Local (LAN).

¿Qué es Wireshark?

- Wireshark es un capturador de paquetes/protocolos de red (aka: sniffer).
- Ademas, parsea paquetes capturados por una interfaz y los muestra con un alto grado de detalle.
- Se usa fundamentalmente como herramienta de diagnóstico de networking: es un "debugger" de la red.
- El mejor amigo del administrador de red, analista de seguridad, programador, hacker, etc.
- Es libre, abierto y gratis.

Algunas definiciones

- ¿NIC? Network Interface Controller (wlan0, eth0, lo, prueben haciendo ifconfig).

```
$ ifconfig
eth0:   flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
ether 3c:92:0e:33:4b:01  txqueuelen 1000  (Ethernet)
RX packets 0  bytes 0 (0.0 B)
RX errors 0  dropped 0  overruns 0  frame 0
TX packets 0  bytes 0 (0.0 B)
TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Algunas definiciones, cont.

Modo promiscuo

lo que significa que los paquetes con MAC destino ajena no se descartan. Suben hasta el kernel para que podamos consumir las tramas. **Igual veríamos mensajes broadcast, multicast y unicast.**

Modo monitor

lo que permite capturar tráfico por la WNIC, ya estemos asociados o no con el AP o la red Ad-Hoc, sin que este sea descartado.

Algunas definiciones, cont. 2

capabilites

Starting with kernel 2.2, Linux divides the privileges traditionally associated with superuser into distinct units, known as capabilities, which can be independently enabled and disabled. Capabilities are a per-thread attribute.

CAP_NET_ADMIN

Permite

- Allow interface configuration
- Allow modification of routing tables
- Allow setting promiscuous mode

Algunas definiciones, cont. 3

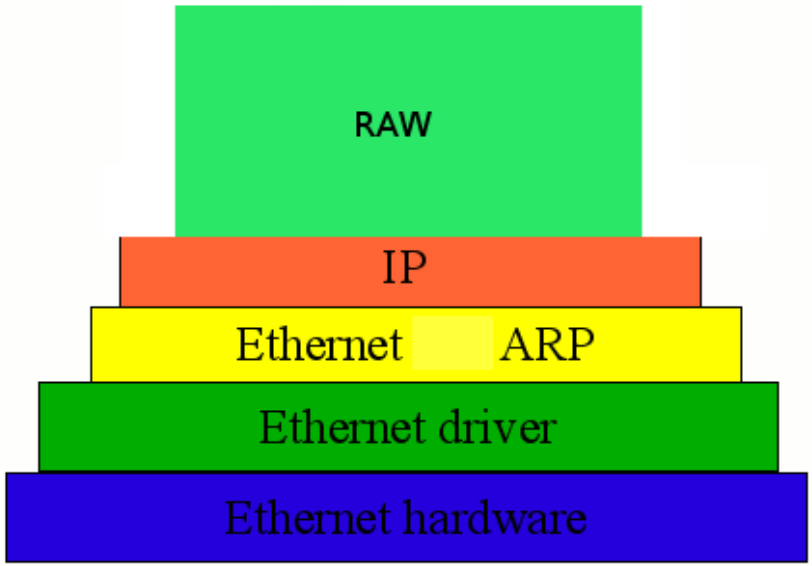
CAP_NET_RAW

Permite emitir:

- **Raw** frames permiten escribir los headers de la capa física
- **Packet** frames obtienen los parámetros de la capa física

Ambos permiten escribir frames con los headers de capa 2 en adelante.

Raw packets?



Captura de paquetes, pero... ¿cómo?

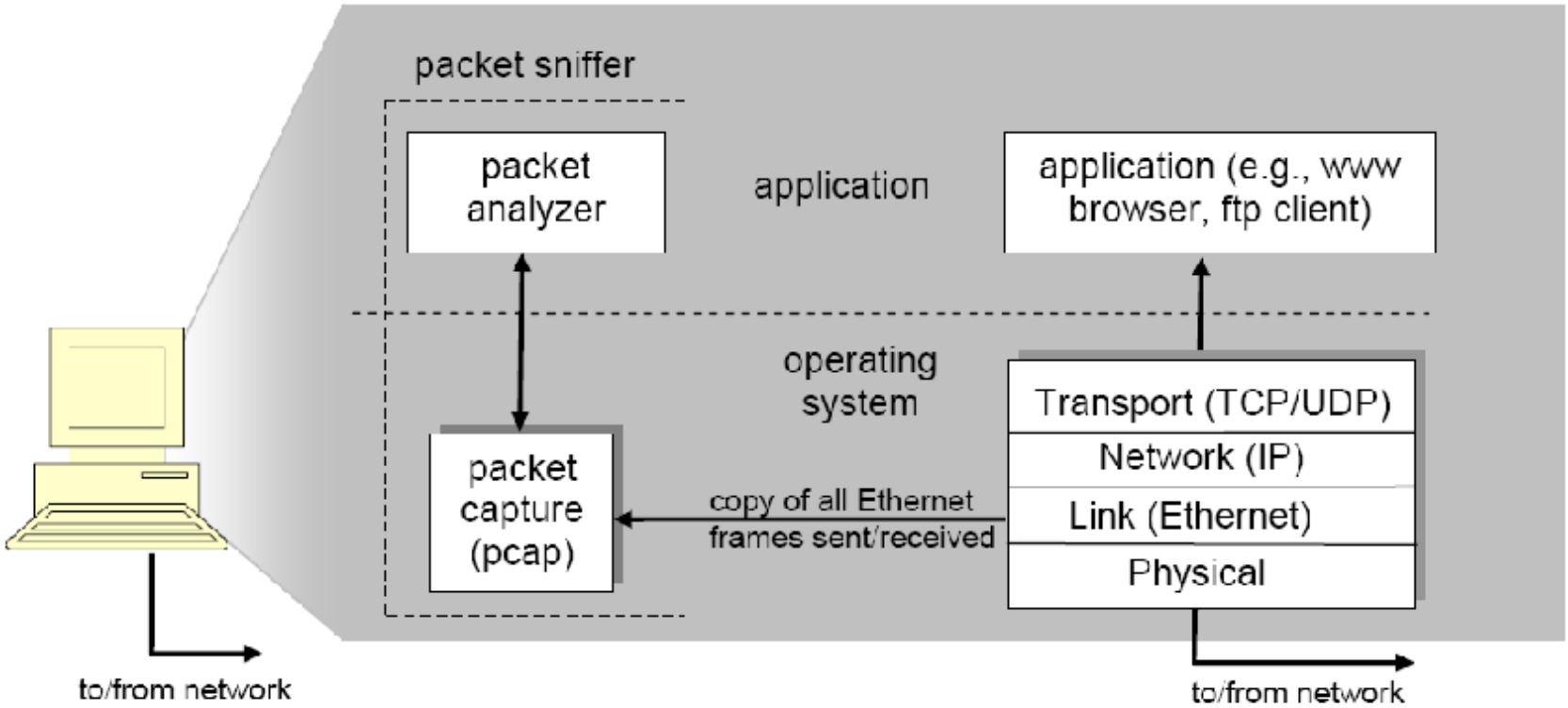


Figure 1: Packet sniffer structure

Para leer mas: http://www.tcpdump.org/faq.html

Escenarios

Local

- loopback
- eth, wlan, etc

Red local

- Atrás de un hub. Todos los mensajes se floodean.
- Atrás de un switch. No podemos ver mensajes ajenos. (Salvo que...)

¿Dónde estamos parados?

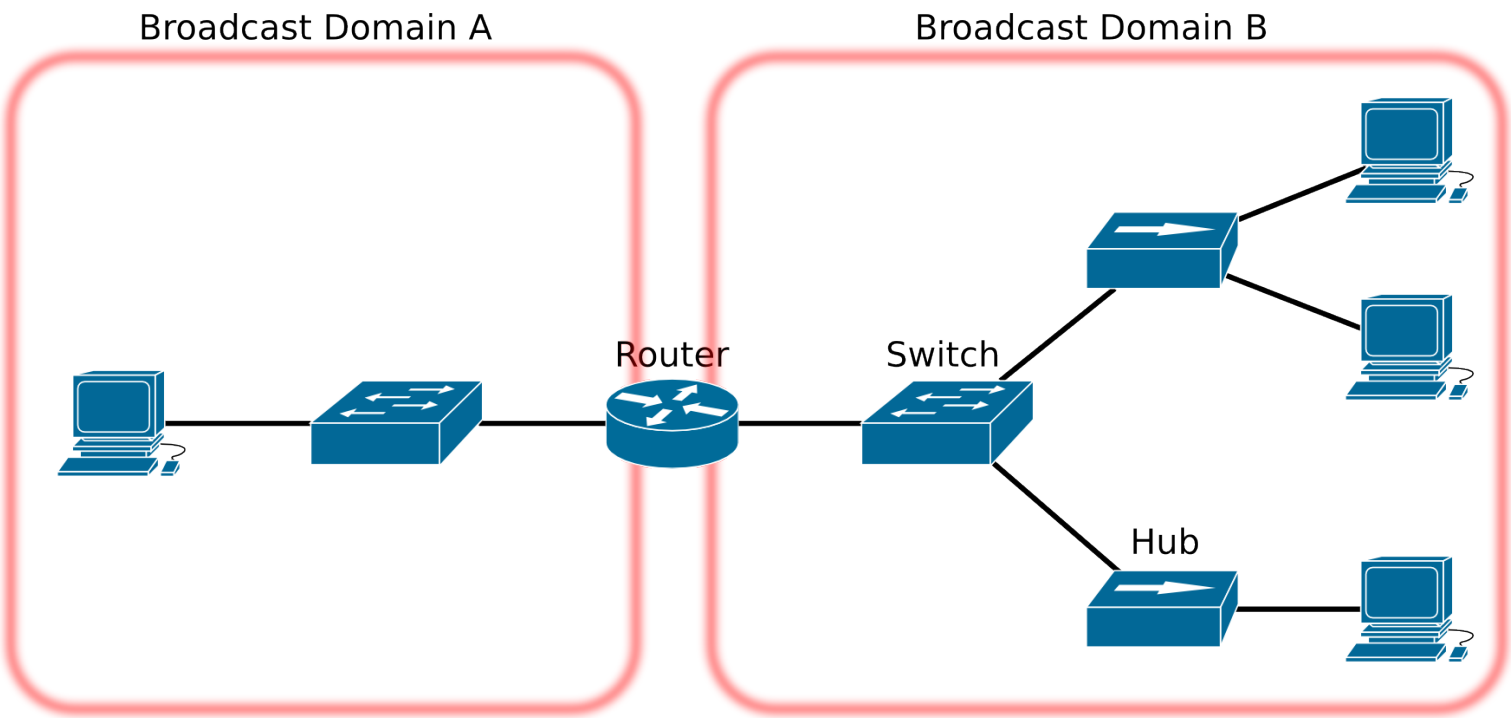
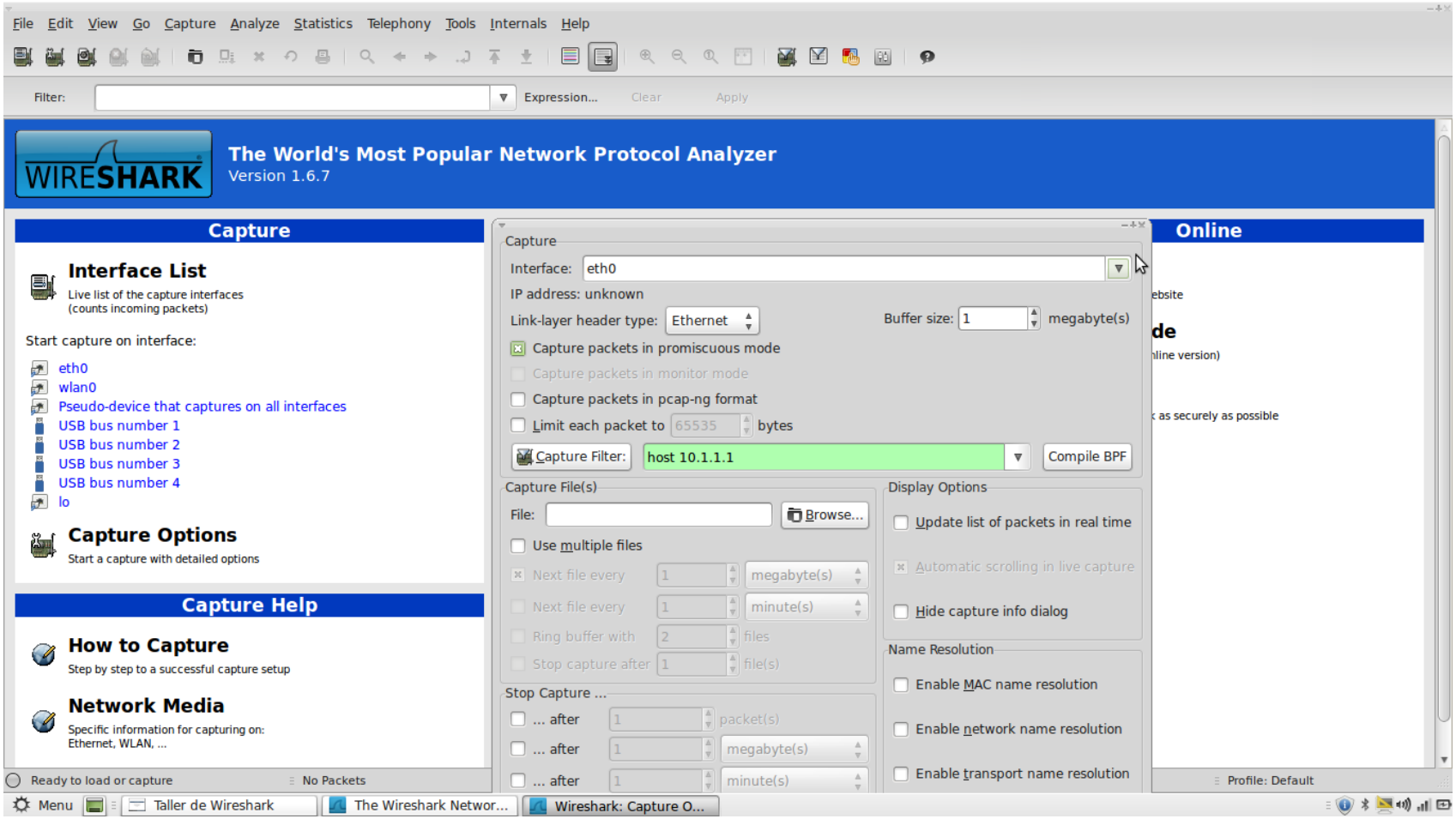


Figura: Mismo dominio de broadcast, mismo segmento de red

Wireshark 1


File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Clear
Apply
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Version 1.6.7
Capture
Interface List
Live list of the capture interfaces
(counts incoming packets)
Start capture on interface:
eth0
wlan0
Pseudo-device that captures on all interfaces
USB bus number 1
USB bus number 2
USB bus number 3
USB bus number 4
lo
Capture Options
Start a capture with detailed options
Capture Help
How to Capture
Step by step to a successful capture setup
Network Media
Specific information for capturing on:
Ethernet, WLAN, ...
Capture
Interface: eth0
IP address: unknown
Link-layer header type: Ethernet
Buffer size: 1 megabyte(s)
Capture packets in promiscuous mode
Capture packets in monitor mode
Capture packets in pcap-ng format
Limit each packet to 65535 bytes
Capture Filter: host 10.1.1.1
Compile BPF
Capture File(s)
File:
Browse...
Use multiple files
Next file every 1 megabyte(s)
Next file every 1 minute(s)
Ring buffer with 2 files
Stop capture after 1 file(s)
Display Options
Update list of packets in real time
Automatic scrolling in live capture
Hide capture info dialog
Name Resolution
Enable MAC name resolution
Enable network name resolution
Enable transport name resolution
Stop Capture ...
... after 1 packet(s)
... after 1 megabyte(s)
... after 1 minute(s)
Online
Ready to load or capture
No Packets
Profile: Default
Menu
Taller de Wireshark
The Wireshark Networ...
Wireshark: Capture O...

Filtros

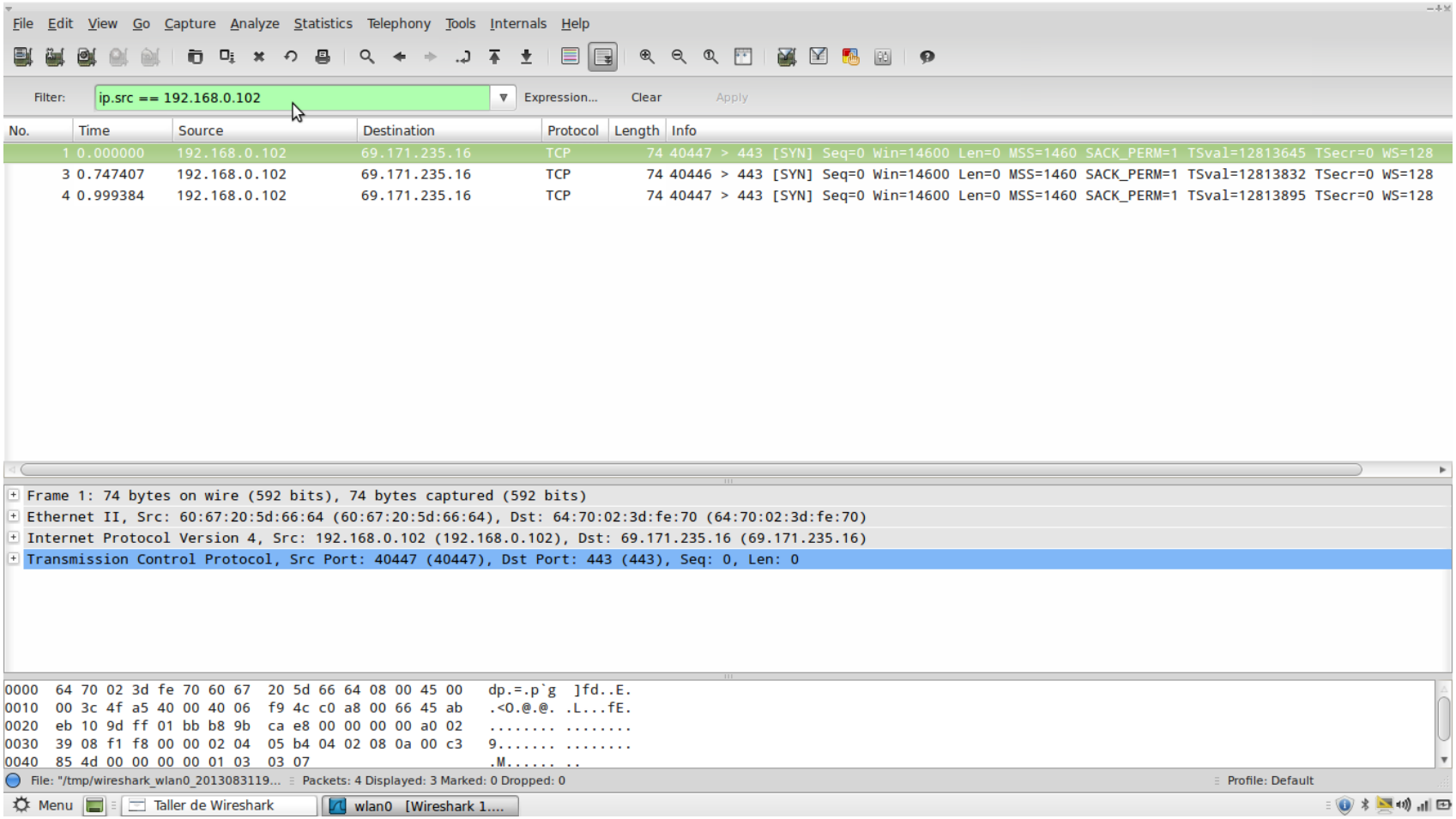
- Es demasiada información, necesitamos poder manejarla.

Ejemplos

- **broadcast ethernet**: eth.dst == FF:FF:FF:FF:FF:FF
- **ethernet type**: eth.type == 0xFFFF (2 bytes)
- **ether src ehost**: eth.src == 90:4c:e5:bb:e0:d6
- **ip src**: ip.src == 192.168.1.1
- **ip protocol**: ip.proto == 1
- etc. Ver secciones Expression y Filter en la barra de filtro.

Recomendado: http://biot.com/capstats/bpf.html

Wireshark 2


File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: ip.src == 192.168.0.102
Expression... Clear Apply
No. Time Source Destination Protocol Length Info
1 0.000000 192.168.0.102 69.171.235.16 TCP 74 40447 > 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=12813645 TSecr=0 WS=128
3 0.747407 192.168.0.102 69.171.235.16 TCP 74 40446 > 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=12813832 TSecr=0 WS=128
4 0.999384 192.168.0.102 69.171.235.16 TCP 74 40447 > 443 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=12813895 TSecr=0 WS=128
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: 60:67:20:5d:66:64 (60:67:20:5d:66:64), Dst: 64:70:02:3d:fe:70 (64:70:02:3d:fe:70)
Internet Protocol Version 4, Src: 192.168.0.102 (192.168.0.102), Dst: 69.171.235.16 (69.171.235.16)
Transmission Control Protocol, Src Port: 40447 (40447), Dst Port: 443 (443), Seq: 0, Len: 0
0000 64 70 02 3d fe 70 60 67 20 5d 66 64 08 00 45 00 dp.=.p`g]fd..E.
0010 00 3c 4f a5 40 00 40 06 f9 4c c0 a8 00 66 45 ab .<O.@.@. .L...fE.
0020 eb 10 9d ff 01 bb b8 9b ca e8 00 00 00 00 a0 02 ....
0030 39 08 f1 f8 00 00 02 04 05 b4 04 02 08 0a 00 c3 9....... ....
0040 85 4d 00 00 00 00 01 03 03 07 .M...... ..
File: "/tmp/wireshark_wlan0_2013083119... Packets: 4 Displayed: 3 Marked: 0 Dropped: 0
Profile: Default
Menu Taller de Wireshark wlan0 [Wireshark 1....

Algunas precauciones

Algunas precauciones

- Todavía no vimos IP.
- No hablamos de routing.
- Nos estamos adelantando un poco.

Ethernet - MAC Address

- *Media Access Control Address*.

Ethernet - MAC Address

- *Media Access Control Address*.
- Identificador de una interfaz de red.

Ethernet - MAC Address

- *Media Access Control Address*.
- Identificador de una interfaz de red.
- 6 octetos

Ethernet - MAC Address

- *Media Access Control Address*.
- Identificador de una interfaz de red.
- 6 octetos
- 3 de OUI (Organization Unique Identifier) `standards.ieee.org/develop/regauth/oui/public.html`

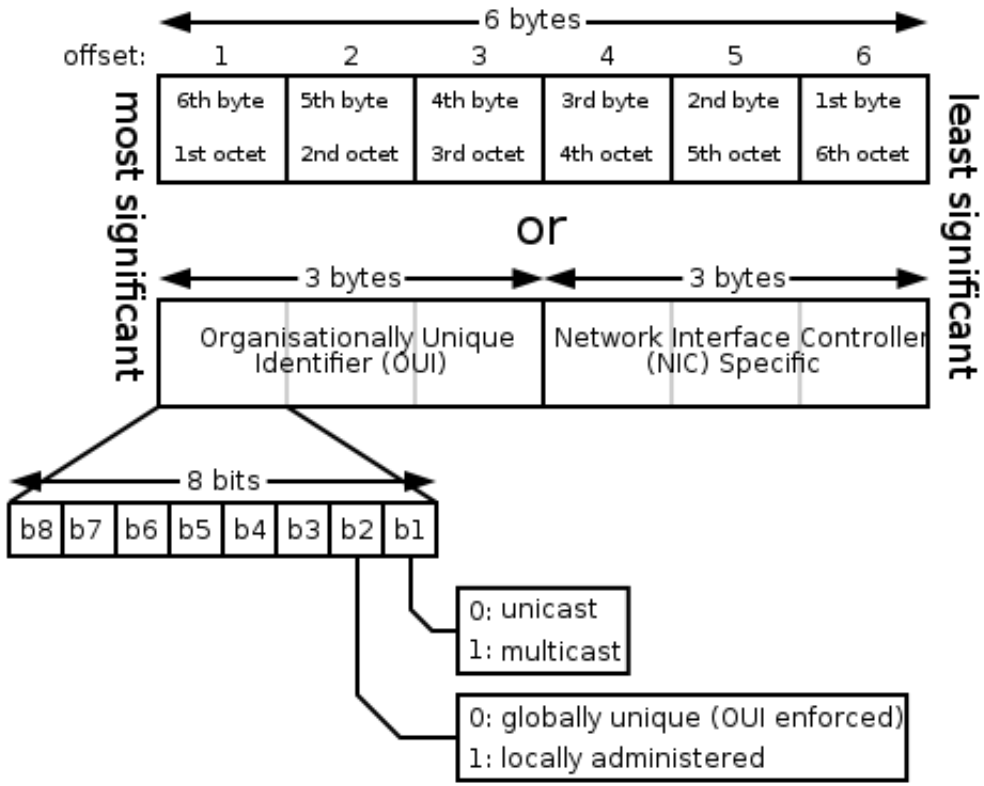
Ethernet - MAC Address

- *Media Access Control Address*.
- Identificador de una interfaz de red.
- 6 octetos
- 3 de OUI (Organization Unique Identifier) `standards.ieee.org/develop/regauth/oui/public.html`
- 3 de NIC (Network Interface Controller)

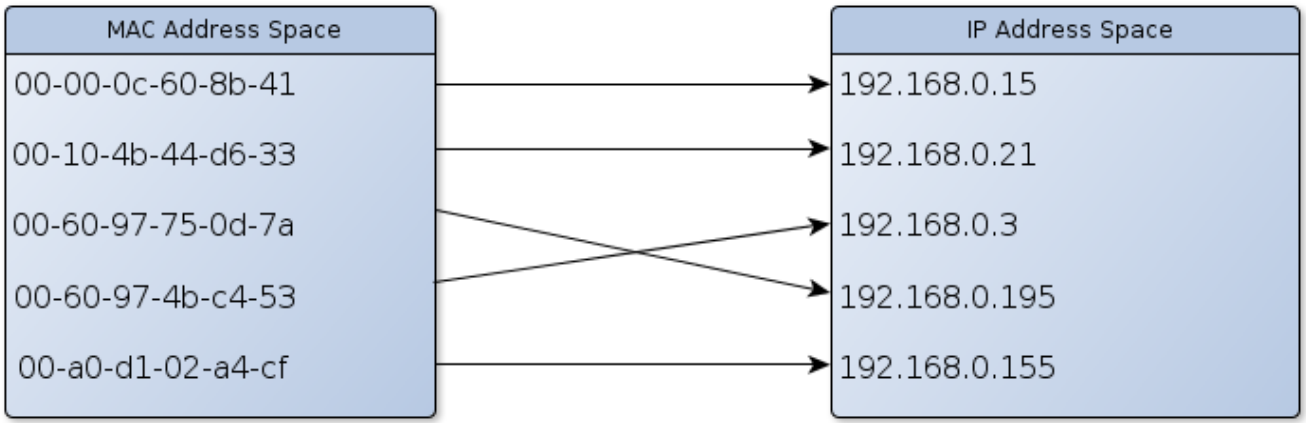
Ethernet - MAC Address

- *Media Access Control Address*.
- Identificador de una interfaz de red.
- 6 octetos
- 3 de OUI (Organization Unique Identifier) `standards.ieee.org/develop/regauth/oui/public.html`
- 3 de NIC (Network Interface Controller)
- Intel Corporate: 00:1c:c0:fa:55:cc

Ethernet - MAC Address cont.



¿Perdón?


MAC Address Space
00-00-0c-60-8b-41
00-10-4b-44-d6-33
00-60-97-75-0d-7a
00-60-97-4b-c4-53
00-a0-d1-02-a4-cf
IP Address Space
192.168.0.15
192.168.0.21
192.168.0.3
192.168.0.195
192.168.0.155

¿Qué es ARP?

- La sigla: *Address Resoution Protocol*.
- Es un protocolo que, en esencia, permite mapear direcciones de nivel de red a direcciones físicas.
- Clave e indispensable en el funcionamiento de las redes modernas.
- Especificado en el RFC 826 (circa 1982).
- No está limitado a IP + Ethernet: la especificación es general.

¿Por qué lo queremos?

- Para hablar con una máquina remota (e.g., el servidor web de Google), mis datos van a tener que moverse primero por mi red local.
- Mi *router* será el encargado de recibirlos y mandarlos por donde corresponda.
- ¡Estos dispositivos de nivel de red necesitan también transmitir sobre la capa de enlace subyacente (e.g., Ethernet, 802.11, etc.)!

En definitiva, no es más que esto

1. Para hablar con Google, sé que debo dirigirme, por ejemplo, a la dirección IP `173.194.42.19` (ya veremos cómo).

En definitiva, no es más que esto

1. Para hablar con Google, sé que debo dirigirme, por ejemplo, a la dirección IP `173.194.42.19` (ya veremos cómo).
2. Mi máquina es astuta y sabe (ya veremos cómo) que, para ello, debe primero comunicarse con mi router, cuya IP es `192.168.1.1`.

En definitiva, no es más que esto

1. Para hablar con Google, sé que debo dirigirme, por ejemplo, a la dirección IP `173.194.42.19` (ya veremos cómo).
2. Mi máquina es astuta y sabe (ya veremos cómo) que, para ello, debe primero comunicarse con mi router, cuya IP es `192.168.1.1`.
3. Acá entra en juego ARP: mi máquina **pregunta** en mi red Ethernet local quién tiene registrada la IP `192.168.1.1`.

En definitiva, no es más que esto

1. Para hablar con Google, sé que debo dirigirme, por ejemplo, a la dirección IP `173.194.42.19` (ya veremos cómo).
2. Mi máquina es astuta y sabe (ya veremos cómo) que, para ello, debe primero comunicarse con mi router, cuya IP es `192.168.1.1`.
3. Acá entra en juego ARP: mi máquina **pregunta** en mi red Ethernet local quién tiene registrada la IP `192.168.1.1`.
4. Y mi router **responde** diligentemente con su dirección física, `c0:c1:c0:0b:8f:2a`.

En definitiva, no es más que esto

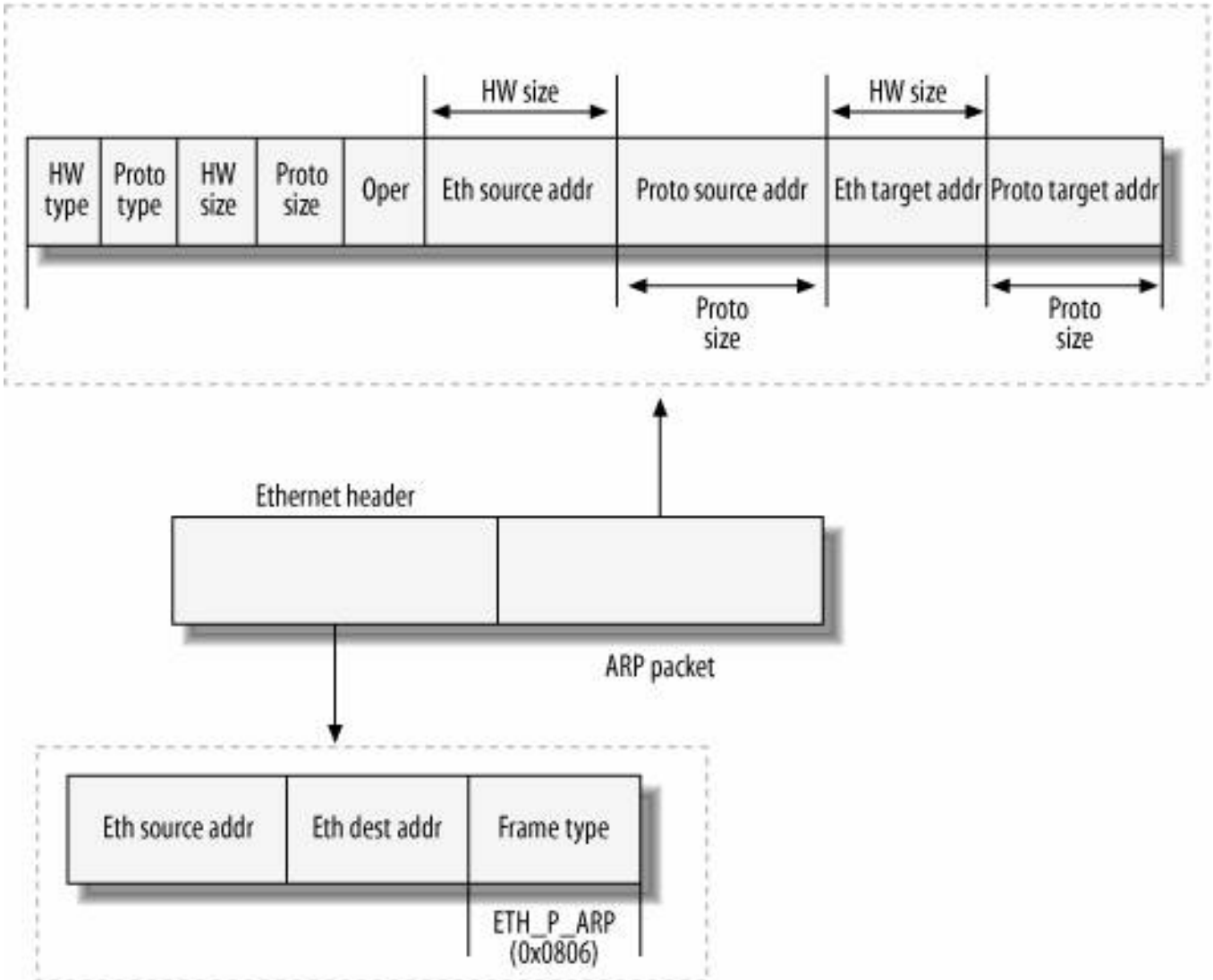
1. Para hablar con Google, sé que debo dirigirme, por ejemplo, a la dirección IP `173.194.42.19` (ya veremos cómo).
2. Mi máquina es astuta y sabe (ya veremos cómo) que, para ello, debe primero comunicarse con mi router, cuya IP es `192.168.1.1`.
3. Acá entra en juego ARP: mi máquina **pregunta** en mi red Ethernet local quién tiene registrada la IP `192.168.1.1`.
4. Y mi router **responde** diligentemente con su dirección física, `c0:c1:c0:0b:8f:2a`.

$\Rightarrow$ Mi máquina envía una trama Ethernet a esta MAC que encapsula un datagrama IP cuyo destino es el servidor de Google.

Tecnicismos varios

- La pregunta ARP consiste en un mensaje **broadcast** sobre la red local.
  - Recordar que no se propaga más allá de la red local!
- La respuesta, en cambio, es **unicast**.
- Optimización: se implementa una caché para guardar las direcciones resueltas (o conocidas).
  - Las entradas se agregan al resolver o bien al observar un pedido de otra máquina.
  - Cada entrada tiene un tiempo de expiración para evitar problemas.

Pormenores del paquete



Pormenores del paquete (cont.)

- El campo **Oper** puede tomar los valores 1 (`who-has`) o 2 (`reply`).
- Observar que la cantidad de bits asingada a las direcciones depende del valor que tomen los campos **HW size** y **Proto size**.
- Dichos campos tienen un largo de 8 bits (i.e., direcciones con un máximo de $2^8 - 1 = 255$ bits).
- **HW type** y **Proto type** indican los protocolos de nivel de enlace y de nivel de red respectivamente involucrados en la comunicación.

Otro uso interesante

- Cuando una máquina bootea o se levanta una de sus interfaces, muchos SOs envían automáticamente un pedido ARP *gratuito*.
- En él, **Proto source addr** == **Proto target addr**.
- Objetivos:
  - Detectar IPs duplicadas en la red local: esto ocurre si se recibe una respuesta.
  - Actualizar la caché ARP de los otros hosts.

...y otro uso más: ARP Spoofing

Spoofing

1. To deceive.
2. To do a spoof of; satirize gently.

- De lo anterior se desprende que ARP es un protocolo **sin estado** y **sin seguridad**.
- La técnica de ARP spoofing se apoya precisamente en estas características.
- Idea: una máquina envía de la nada una respuesta ARP mapeando una IP objetivo con su propia MAC.
- $\Rightarrow$ todo el tráfico destinado a dicha IP va a ser recibido por ella.

Intro a Scapy

- Scapy es un framework de manipulación de paquetes.

Intro a Scapy

- Scapy es un framework de manipulación de paquetes.
- Permite crear paquetes, capturar paquetes, enviar paquetes, analizar paquetes, etc.

Intro a Scapy

- Scapy es un framework de manipulación de paquetes.
- Permite crear paquetes, capturar paquetes, enviar paquetes, analizar paquetes, etc.
- Orientado a capas. pkt = Ether() / IP() / TCP() nos genera un paquete TCP valido.

Transmitiendo

```
#! /usr/bin/env python
# arping2tex : arpings a network and outputs a LaTeX table as a result

import sys
if len(sys.argv) != 2:
    print "Usage: arping2tex <net>\n  eg: arping2tex 192.168.1.0/24"
    sys.exit(1)

from scapy.all import srp,Ether,ARP,conf
conf.verb=0
ans,unans=srp(Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(pdst=sys.argv[1]),
          timeout=2)

print r"\begin{tabular}{|l|l|}"
print r"\hline"
print r"MAC & IP\\"
print r"\hline"
for snd,rcv in ans:
    print rcv.sprintf(r"%Ether.src% & %ARP.psrc%\\")
print r"\hline"
print r"\end{tabular}"
```

Escuchando

```
#! /usr/bin/env python
from scapy.all import *
def monitor_callback(pkt):
    print pkt.show()


if __name__ == '__main__':
    sniff(prn=monitor_callback, filter = "arp", store = 0)
```

Trabajos Prácticos

¿Cómo son los Trabajos Prácticos?

- 3 Trabajos Prácticos (3 entregas)
  1. TP1: Wiretapping (Information Gathering)
  2. TP2: ICMP (Rutas en Internet)
  3. TP3: Programación en *raw sockets*
- Objetivos
  1. Experimentar con la red. No siempre es lo que parece.
  2. Hacer análisis acerca de los comportamientos no esperados.
  3. Enmarcar el análisis en un informe (o *tech rep*).

¿Qué esperamos que hagan?

- Que reflexionen sobre lo que es una red.
- Que se vayan con herramientas prácticas para hacer diagnóstico.
- Que entiendan los conceptos teóricos de una manera aplicada.
- Que entreguen informes rigurosos sobre lo que ustedes descubrieron.

Dinámica de presentación y entrega.

- 3 o 4 integrantes.
- Fechas de entrega por mail.
  1. TP1: 23/09/2014
  2. TP2: 21/10/2014
  3. TP3: 11/11/2014
- Pautas para los informes.
  1. Tener en cuenta la estructura de informe científico *(introducción, métodos, resultados, conclusiones).*
  2. El código no es tan importante.
  3. Ojo con las figuras. Que sean claras y tengan leyendas.
- Template (*recomendado*):
  `http://mocha-java.uccs.edu/ieee/`

¡A trabajar!: Primera consigna

(a) Implementar una *tool* para escuchar pasivamente en la red local.

(b) En base a los dos siguientes modelos de fuente de información propuestos:

$S_{dst} = \{s_1 \cdots s_n\}$ siendo s_i una IP que aparece como dirección destino en los paquetes ARP *who-has*

$S_{src} = \{s_1 \cdots s_n\}$ siendo s_i una IP que aparece como dirección origen en los paquetes ARP *who-has*

- Adapte la *tool* del inciso (a) para estimar las probabilidades de dichas fuentes en función de los paquetes ARP observados y calcular la entropía de cada fuente.
- Usando dicha *tool*, realizar capturas de paquetes ARP sobre distintas LANs (una por cada integrante de grupo, mínimo 3). *En la medida de lo posible, intentar capturar en una red que no sea controlada (en el trabajo, en un shopping, etc.)*

¡A trabajar!: Segunda consigna

Utilizando lo hecho en la consigna previa, realizar un análisis que permita encontrar nodos distinguidos, valiéndose principalmente de gráficos. Sugerimos, entre otros, histogramas de IPs solicitadas o grafos dirigidos de IPs con pesos en los nodos (donde existirá un eje entre la IP x y la IP y sii se observó un request ARP con source IP x y target IP y) y analizar que IPs son estadísticamente significativas en la LAN analizando la información de cada símbolo con respecto a la entropía de su respectiva fuente.

Referencias

- `http://www.tcpdump.org/papers/bpf-usenix93.pdf`
- `http://biot.com/capstats/bpf.html`
- **man capabilities**
- **man packet**