

# Sniffing di rete

Federico Fergnani

# Sniffing

Da Wikipedia:

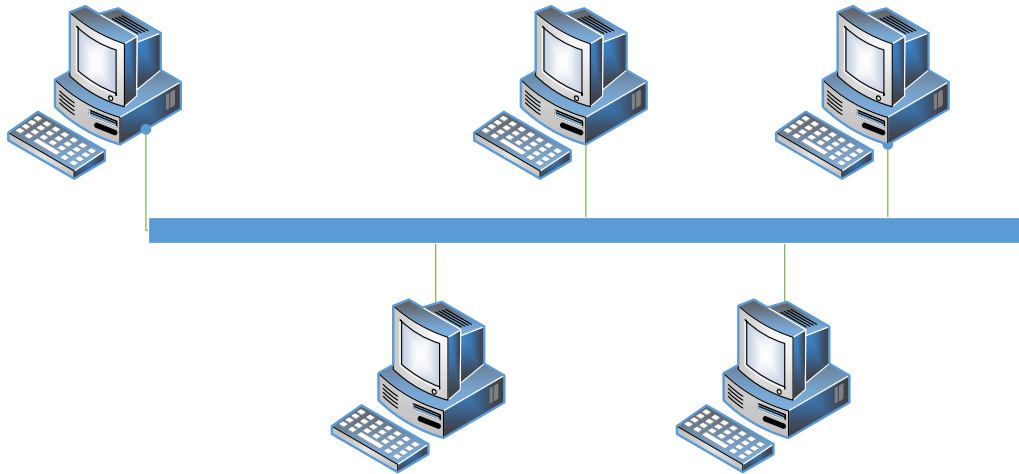
- ❑ Si definisce **sniffing** (dall'inglese, odorare), in informatica e nelle telecomunicazioni, l'attività di intercettazione passiva dei dati che transitano in una rete telematica.
- ❑ Tale attività può essere svolta sia per scopi legittimi (ad esempio l'analisi e l'individuazione di problemi di comunicazione o di tentativi di intrusione) sia per scopi illeciti contro la sicurezza informatica (intercettazione fraudolenta di password o altre informazioni sensibili).
- ❑ I prodotti software utilizzati per eseguire queste attività vengono detti **sniffer** ed oltre ad intercettare e memorizzare il traffico offrono funzionalità di analisi dello stesso.
- ❑ Gli **sniffer** intercettano i singoli pacchetti, decodificando le varie intestazioni di livello data-link, rete, trasporto, applicativo. Inoltre possono offrire strumenti di analisi che analizzano ad esempio tutti i pacchetti di una connessione TCP per valutare il comportamento del protocollo di rete o per ricostruire lo scambio di dati tra le applicazioni.

# Software di sniffing

Tra le soluzioni più popolari e diffuse si possono sicuramente segnalare:

- Tcpdump
- Tshark
- Wireshark (GUI)

# Cosa è possibile intercettare



- ❑ I tool indicati nella slide precedente utilizzano la scheda di rete della macchina su cui sono in esecuzione (PC, Server, ...) per "intercettare" il traffico di rete
- ❑ Normalmente quindi è possibile intercettare i pacchetti che transitano nello stesso segmento di rete (o dominio di collisione) di quello della macchina che si usa per l'analisi.

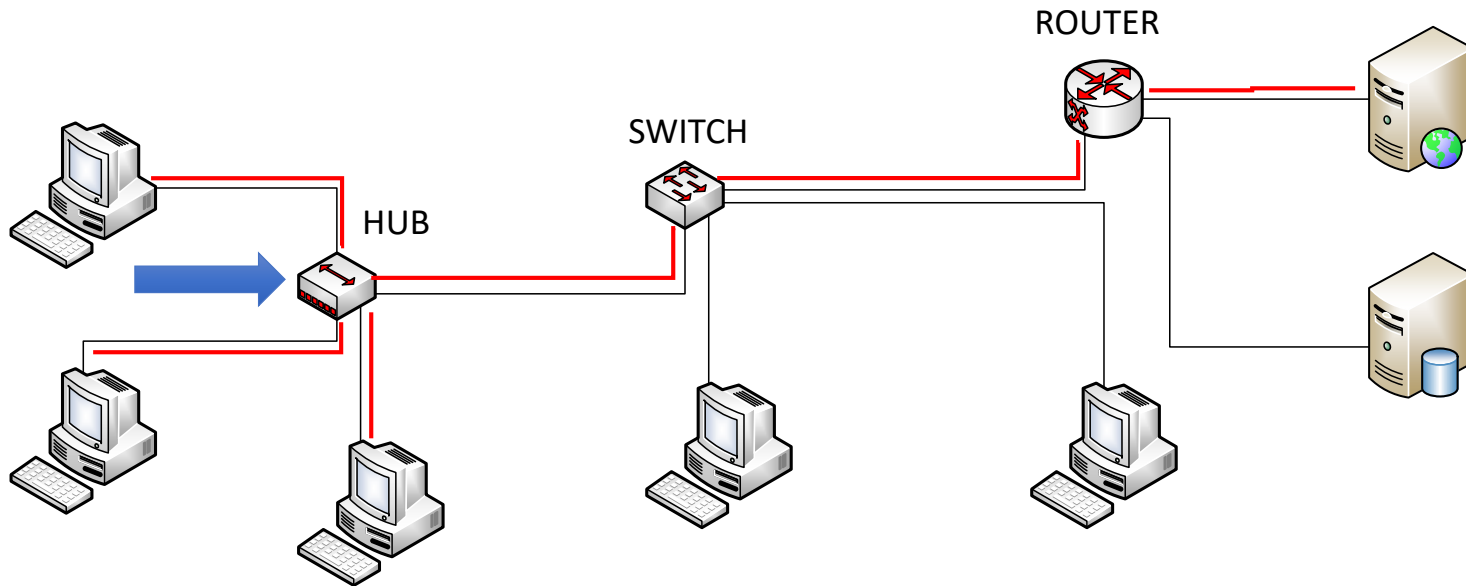
# Modalità promiscua

In condizioni normali una interfaccia di rete rende disponibili al sistema operativo

- ❑ I frame indirizzati all'interfaccia stessa (che hanno come MAC address di destinazione quello dell'interfaccia)
- ❑ Frame broadcast (nelle reti Ethernet hanno come destinatario `FF:FF:FF:FF:FF:FF`) usati ad esempio dall' Address Resolution Protocol (ARP) e dal Neighbor Discovery Protocol per determinare l'indirizzo IP associato ad un indirizzo MAC
- ❑ Una parte dei frame previsti per le trasmissioni multicast

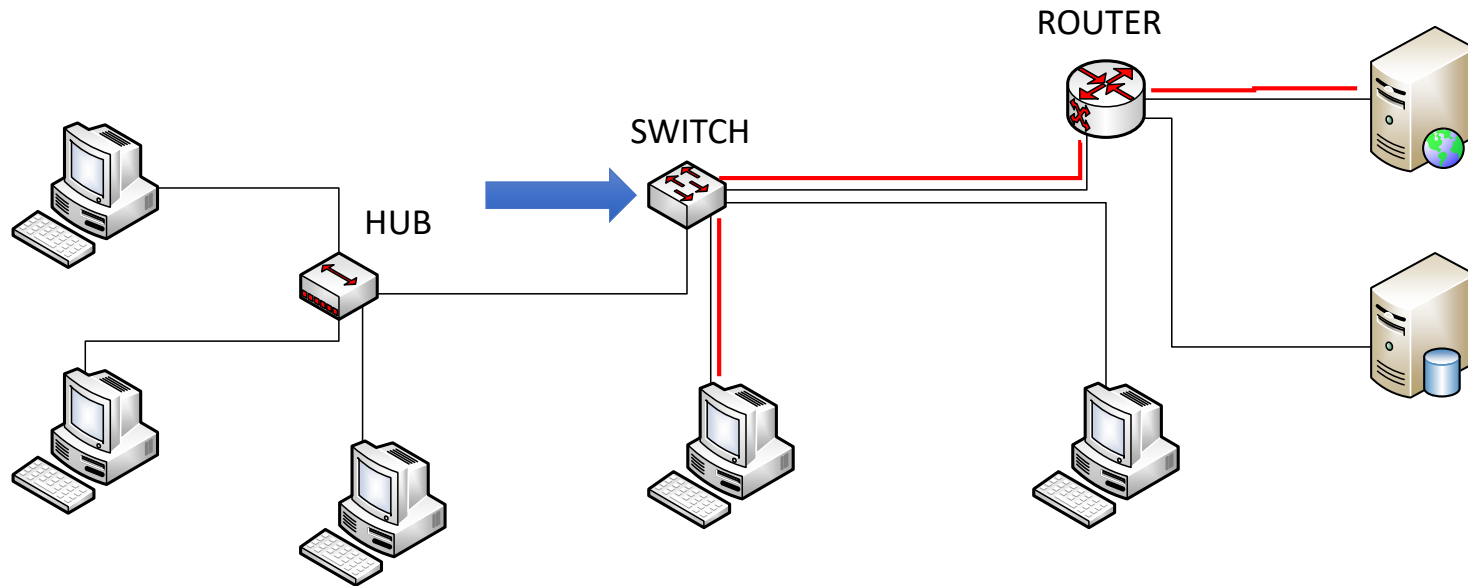
La maggior parte delle schede di rete però supporta una diversa modalità di funzionamento, denominata **modalità promiscua**, nella quale rende disponibile al S.O. tutti i frame che è in grado di leggere **nel segmento di rete in cui si trova**, anche quelli destinati a interfacce con un diverso MAC address

# Identificazione del segmento di rete



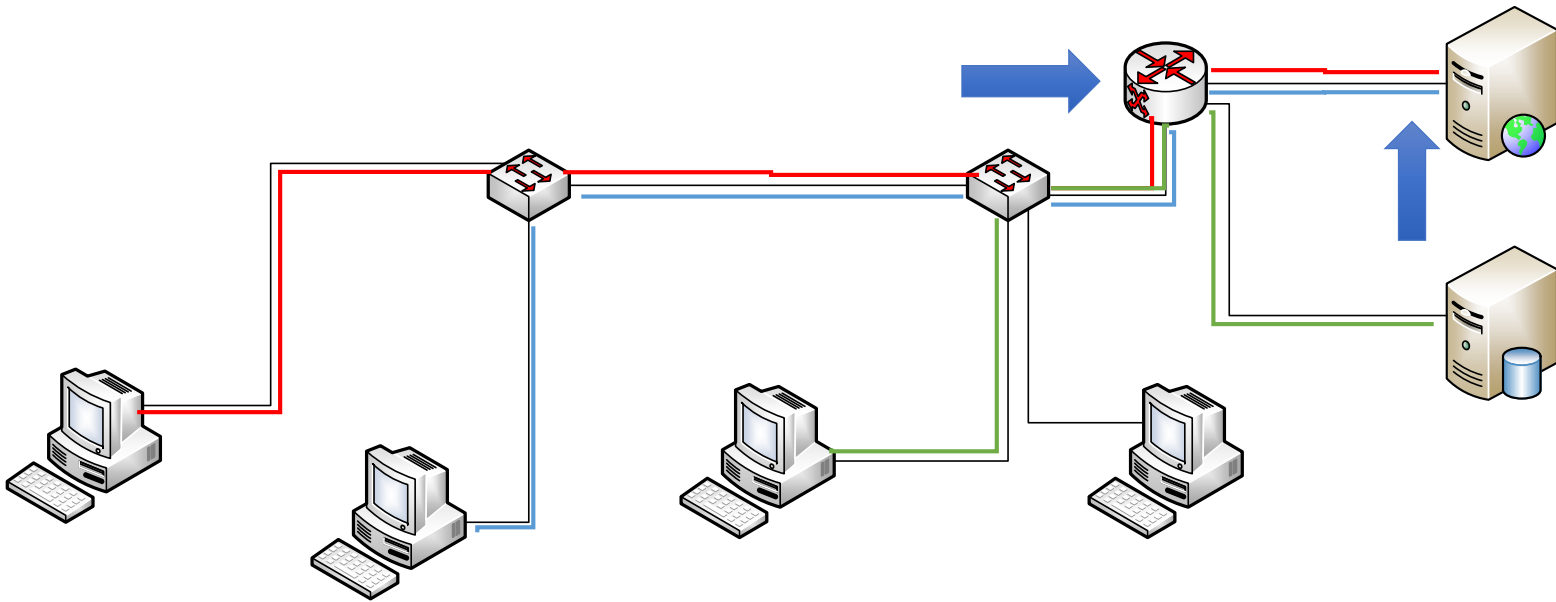
- ❑ Gli HUB replicano il traffico di rete ricevuto da una porta su tutte le altre, per cui tutti gli apparati collegati sono nello stesso **segmento di rete** (o **dominio di collisione**)
- ❑ Ormai sono considerati (giustamente) un device obsoleto, da evitare di utilizzare ove possibile (visto anche il costo assolutamente comparabile di mini-switch dalle prestazioni nettamente superiori).

# Identificazione del segmento di rete



- ❑ Gli switch sono in grado di segmentare la rete tra tutti gli apparati collegati, per cui inviano ad ogni apparato solo il traffico Ethernet che gli è destinato
- ❑ Quindi, un PC collegato ad uno switch potrà rilevare solo il traffico diretto alla sua scheda di rete ed il traffico broadcast, anche utilizzando la modalità promiscua

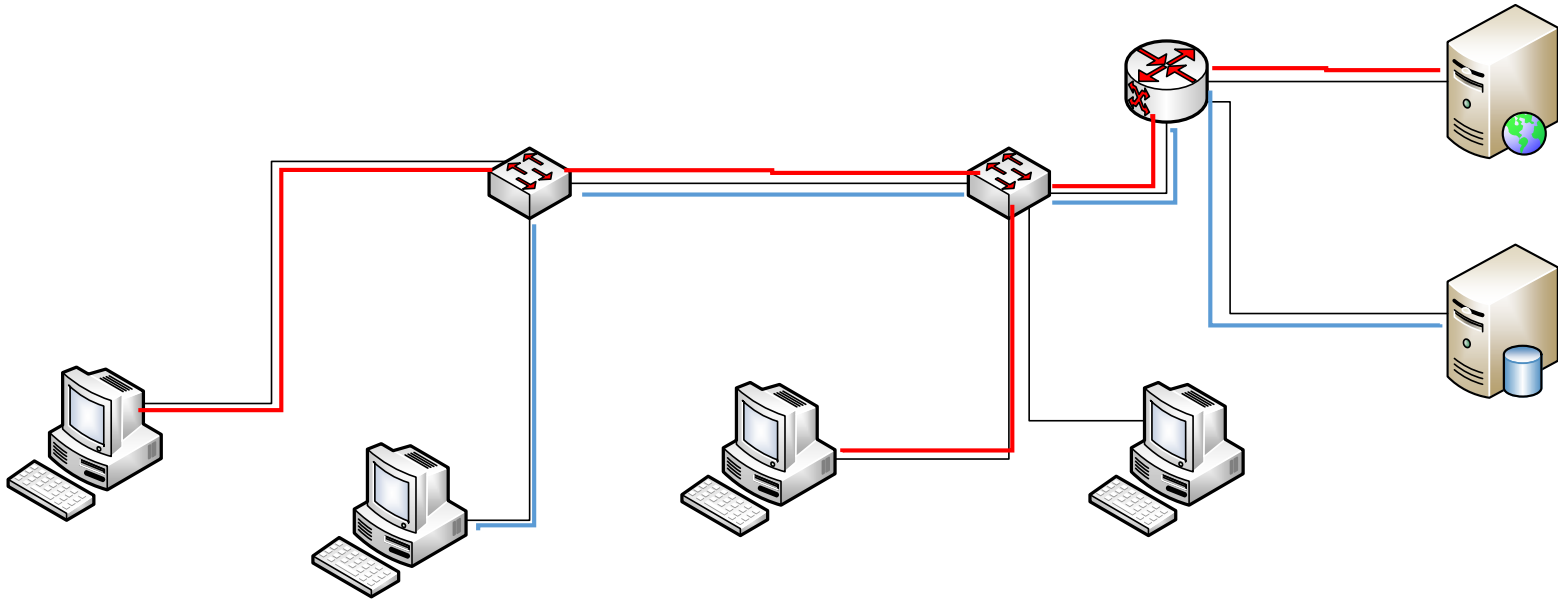
# Identificazione del segmento di rete



- Risulta di maggiore utilità ai fini della diagnostica della rete o dei servizi lo sniffing del traffico effettuato su di un server o su di un router
  - Sul server è possibile "intercettare" tutto il traffico scambiato con tutti i client
  - Sul router è possibile intercettare tutto il traffico che lo attraversa

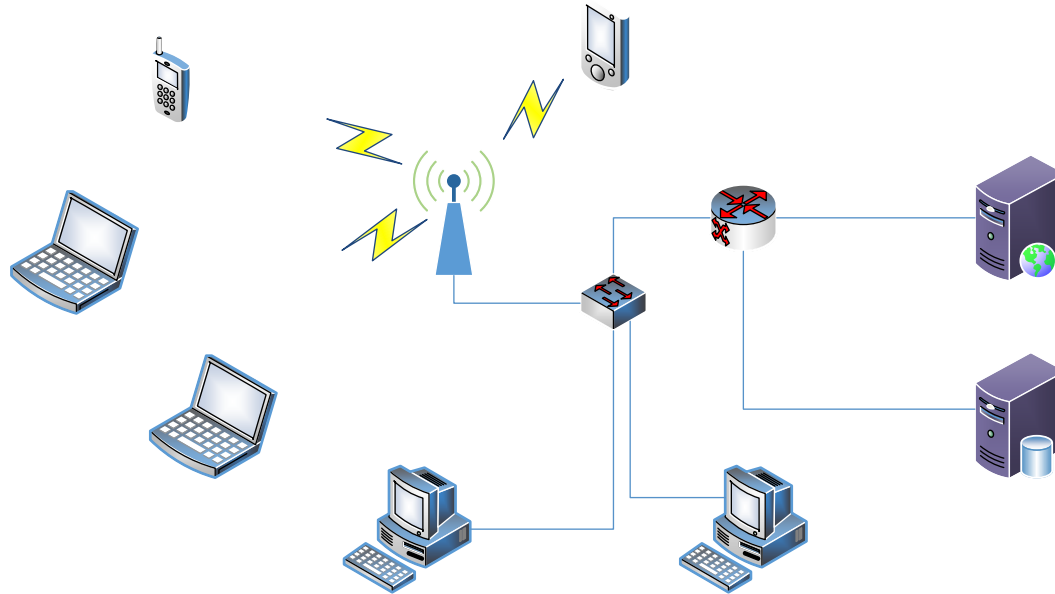


# Replicazione porte



- ❑ Esiste anche la possibilità, per uno switch, di replicare tutto il traffico destinato ad una delle sue porte, su di un'altra.
- ❑ Questa modalità viene solitamente attivata, sugli apparati che la supportano, per scopi di monitoraggio e diagnostica
- ❑ È possibile "forzare" l'instradamento di parte del traffico di rete destinato ad una delle porte verso una porta diversa utilizzando tecniche di ARP-poisoning / ARP-spoofing

# Identificazione del segmento di rete

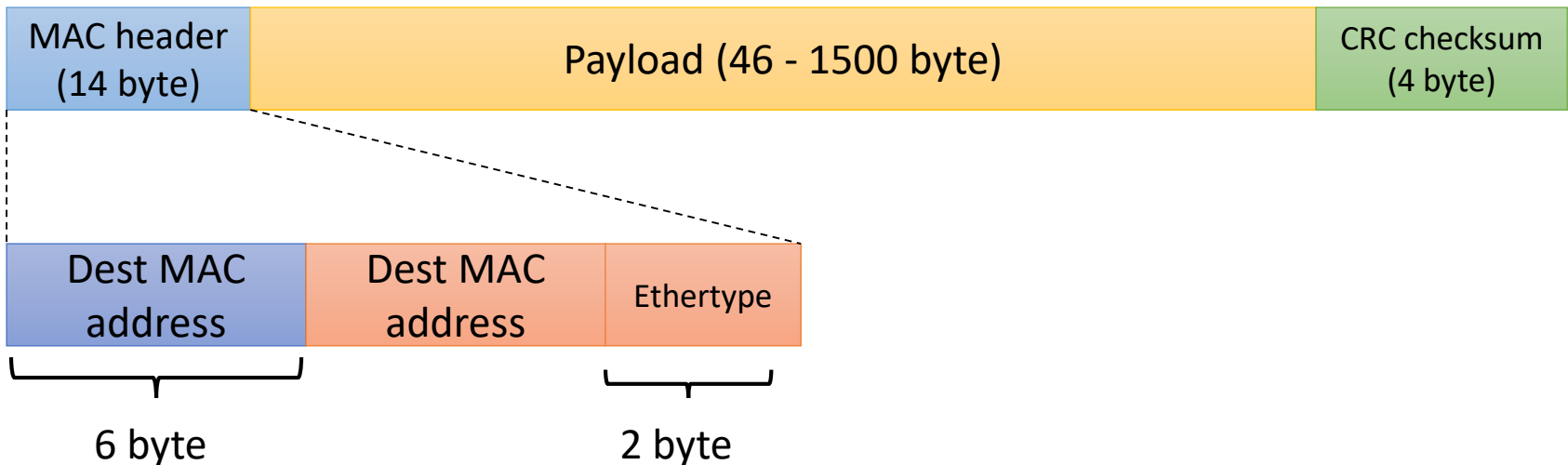


❑ Per quello che riguarda le reti WiFi invece, essendo la comunicazione wireless, per sua natura, propagata su di un mezzo condiviso, risulta molto più semplice fare sniffing del traffico di rete veicolato da un Access Point, in questo caso il limite maggiore è rappresentato dalle schede wireless, di cui solo una parte minoritaria supporta la modalità promiscua (o monitor mode) e dall'eventuale cifratura dei dati

❑ <https://wiki.wireshark.org/CaptureSetup/WLAN>

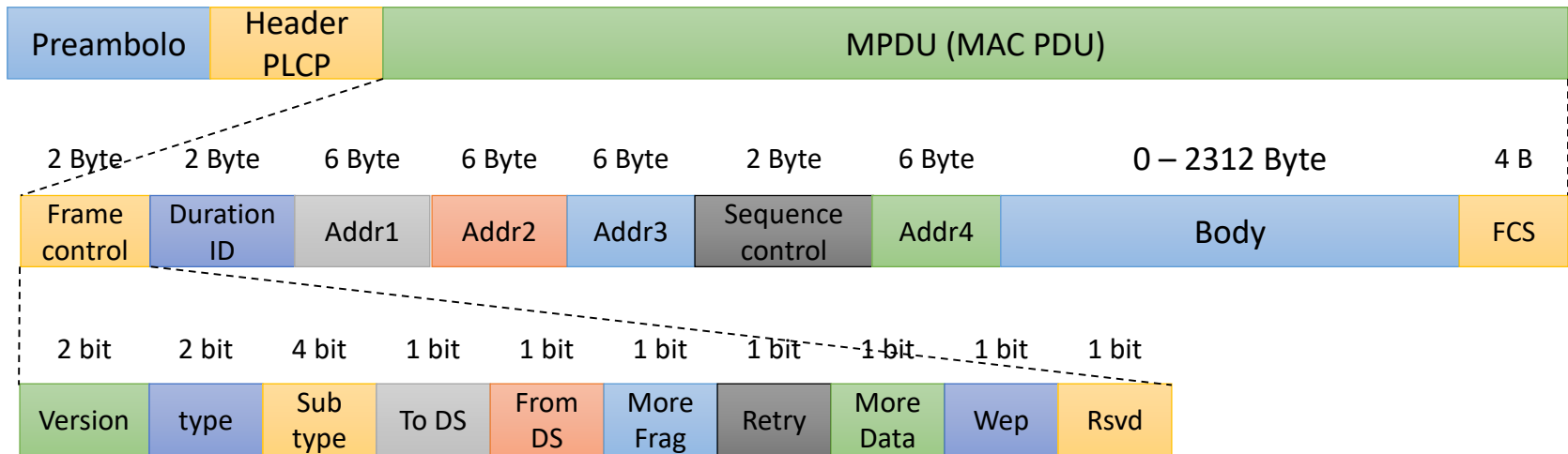
❑ <https://wiki.wireshark.org/HowToDecrypt802.11>

# Cosa si "cattura": Ethernet Frame



- ❑ Al livello "data link" del modello ISO-OSI, nel caso di una connessione basata sul protocollo Ethernet (IEEE 802.3 ) quello che viene analizzato dal software di sniffing è il frame Ethernet
- ❑ Nel payload del frame sono poi contenuti tutti i protocolli di livello superiore (al cui livello più esterno tipicamente troveremo il protocollo IP, anche se non è l'unico possibile)

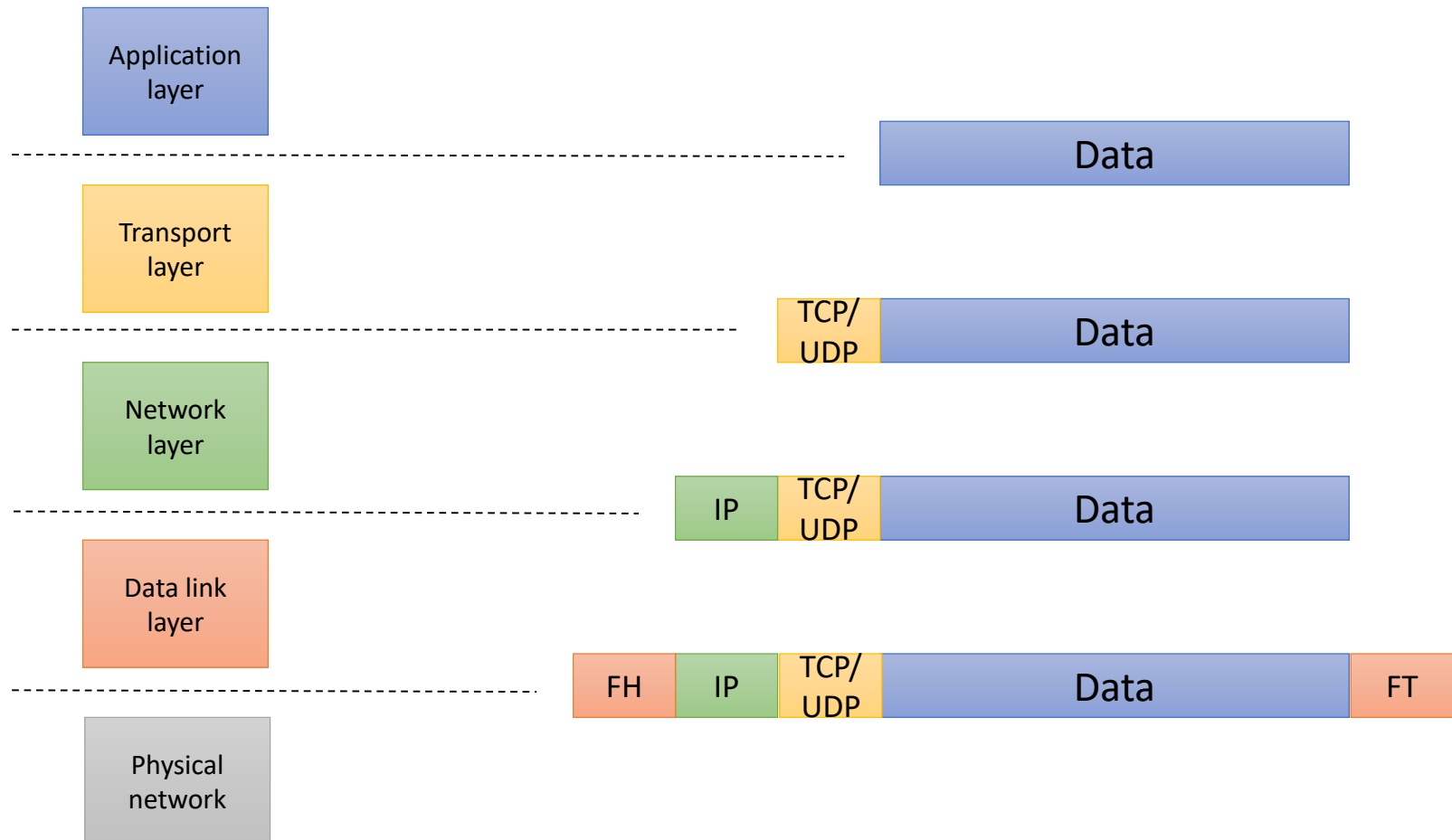
# Cosa si cattura: WiFi 802.11



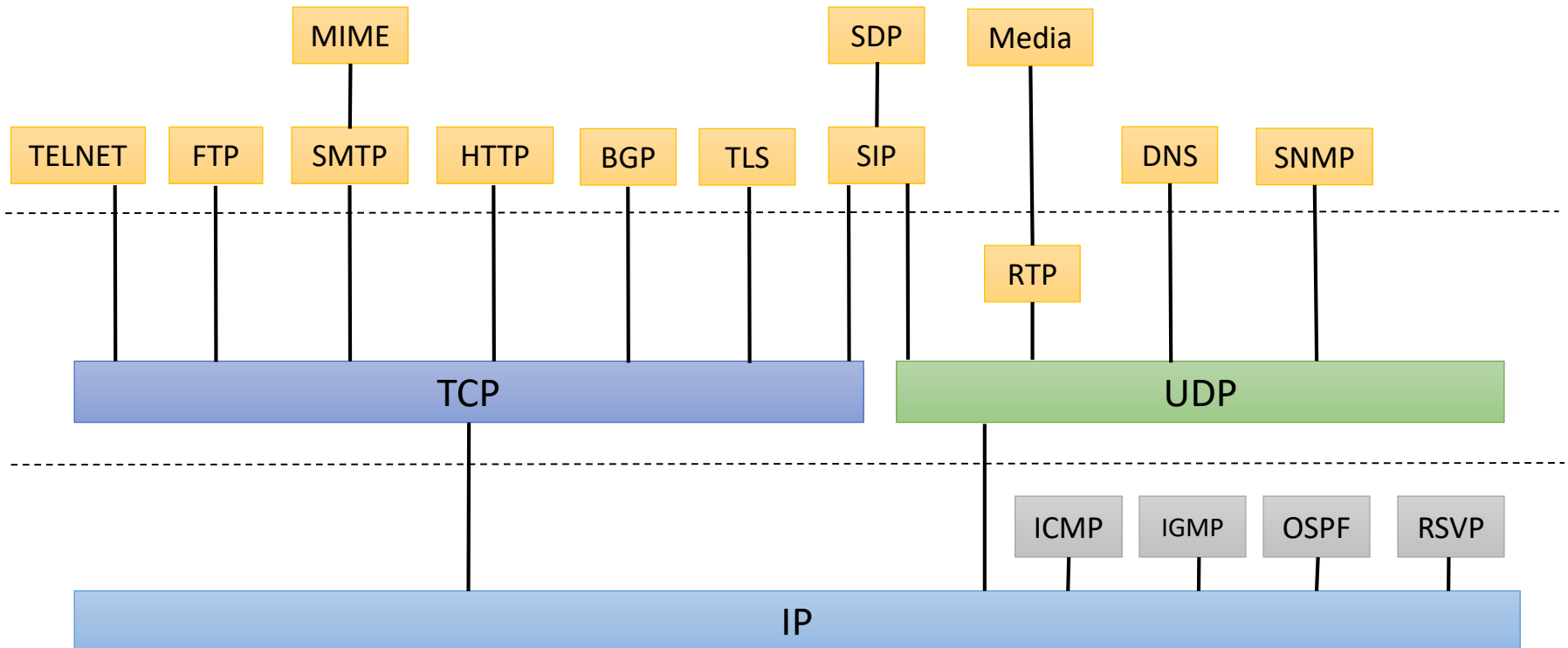
- ❑ L'utilizzo di un tool di sniffing in ambito WiFi porta alla cattura e all'analisi dei frame specifici del protocollo (in cui, come nel caso Ethernet, sono contenuti i protocolli di comunicazione dei livelli superiori dello stack ISO-OSI)

# Incapsulamento pacchetti

- ❑ I dati sono inviati allo stack di protocolli
- ❑ Ogni Layer aggiunge i propri header ai dati provenienti dal livello superiore



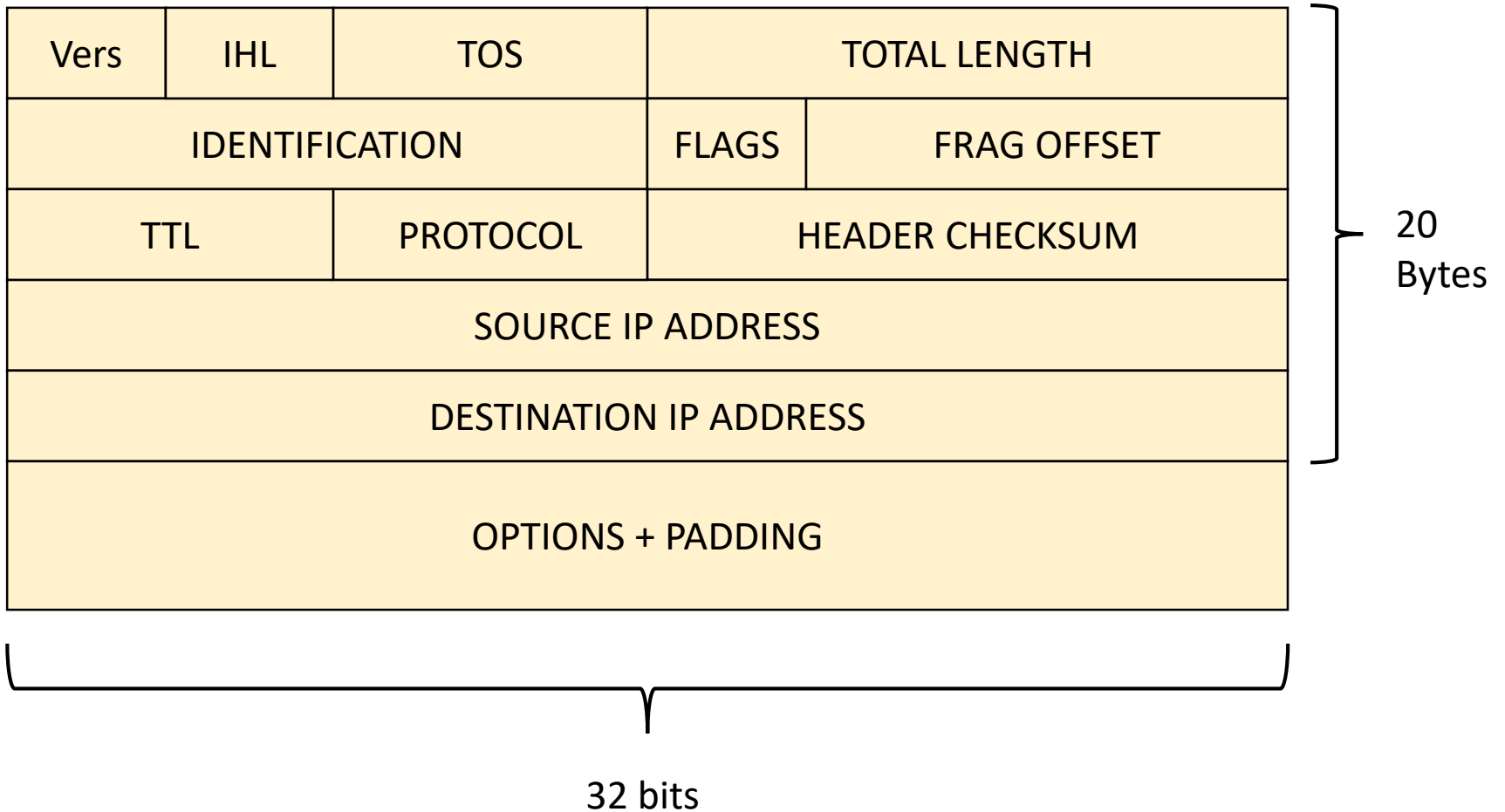
# Suite protocolli TCP/IP



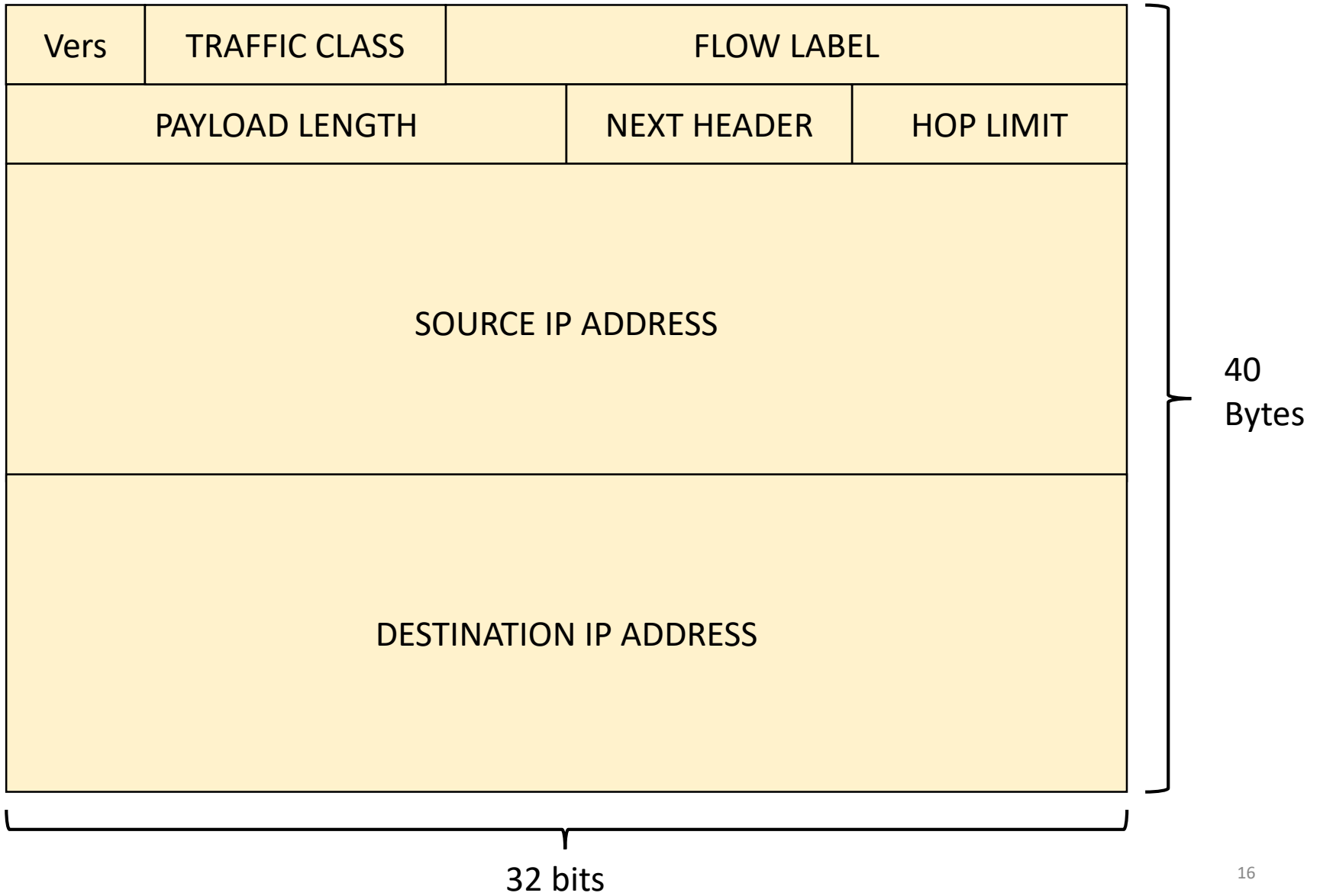
- BGP: border gateway protocol
- DNS: domain name system
- FTP: file trasfer protocol
- HTTP: hyper text transfer protocol
- ICMP: internet control message protocol
- IGMP: internet group management protocol
- IP: internet protocol
- MIME: multi-purpose internet mail extension
- OSFP: open shortest path first

- RSVP: resource reservation protocol
- RTP: real-time transport protocol
- SDP: session descrittione protocol
- SIP: session initiation protocol
- SMTP: simple mail transfer protocol
- SNMP: simple network management protocol
- TCP: transmission control protocol
- TLS: transport layer security
- UDP: user datagram protocol

# Header IPv4



# Header IPv6





# Header TCP

source port (16)		destination port (16)	
sequence number (32)			
acknowledgement number (16)			
d. o.(4)	reserved(6)	flags(6)	window(16)
checksum (16)		urgent pointer	
options (variabile)		padding (variabile)	
data(variabile)			

## Flags:

- URG (urgent),
- ACK (acknowledgment),
- PSH (push function),
- RST (close the connection),
- SYN (synchronize sequence numbers),
- FIN (end of data from sender)

# Header UDP

source port (16)	destination port (16)
length (16)	checksum (16)
data (variabile)	

- ❑ I pacchetti UDP non contengono informazioni di stato per la sessione di comunicazione.
- ❑ UDP è un protocollo stateless, senza ritrasmissioni, protezione contro la perdita dei dati o l'arrivo fuori ordine (se ne può occupare il protocollo di livello superiore)

# TCPDUMP

- ❑ TCPDUMP ( [www.tcpdump.org](http://www.tcpdump.org) ) è forse il più diffuso tool a riga di comando per l'analisi del traffico di rete
- ❑ È un tool indispensabile agli amministratori di rete per analizzarne il traffico, trovare eventuali problemi e risolverli
- ❑ Ma è anche un tool estremamente utile per studiare i protocolli di rete ed il loro comportamento

# TCPDUMP: opzioni comuni

- ❑ `-i any` : si mette in ascolto su tutte le interfacce di rete
- ❑ `-i eth0` : si mette in ascolto solo sull'interfaccia specificata (in questo caso `eth0`).
- ❑ `-D` : mostra la lista delle interfacce disponibili
- ❑ `-n` : non risolve gli hostname.
- ❑ `-nn` : non risolve hostname e port name.
- ❑ `-q` : limita la quantità di output (quiet).
- ❑ `-t` : NON stampa un timestamp per ogni pacchetto.
- ❑ `-tttt` : fornisce un timestamp in formato human-readable.
- ❑ `-X` : mostra il contenuto dei pacchetti sia in esadecimale che in ASCII.
- ❑ `-XX`: come `-X` ma mostra anche i dati contenuti nel pacchetto l'header a link level (tipicamente Ethernet).

# TCPDUMP: opzioni comuni

- ❑ `-A` : Mostra il contenuto dei pacchetti in ASCII. Il contenuto del pacchetto prima del payload (per esempio gli header IP e TCP) possono contenere caratteri non stampabili, il che fa sì che la prima parte del pacchetto contenga caratteri "strani".
- ❑ `-v` , `-vv` , `-vvv` : Aumenta la quantità di informazioni sui pacchetti stampata a video
- ❑ `-c <num>`: Cattura un numero `num` di pacchetti e poi si ferma.
- ❑ `-s <len>`: definisce la *snaplength* (dimensione di cattura) in byte. La dimensione di default dipende dalla versione di `tcpdump` (di solito almeno 96, nelle più vecchie 68), ma si può usare `-s0` per catturare l'intero pacchetto.
- ❑ `-e` : stampa l'header a link level per ogni pacchetto

# TCPDUMP: opzioni comuni

- ❑ `-S` : Stampa il numero di sequenza assoluto dei pacchetti TCP anziché quello relativo.
- ❑ I client su ognuno dei lati di una connessione TCP mantengono un numero a 32 bit, chiamato "*sequence number*", incluso in ogni pacchetto scambiato, che serve a tenere traccia dei dati scambiati e ad associare gli ACK al pacchetto corretto.
- ❑ Quando un host inizia una sessione TCP, il suo numero di sequenza iniziale è un numero casuale compreso tra 0 e 4.294.967.295. Gli analizzatori di traffico come tcpdump (ma anche Wireshark) mostrano un numero di sequenza **relativo**, calcolato a partire dal primo pacchetto scambiato, anziché il numero realmente scritto all'interno del pacchetto, a meno che non sia specificato diversamente (con l'opzione `-S` )

# TCPDUMP: output

- ❑ Tcpcmdump differenzia l'output prodotto in funzione del tipo di pacchetto catturato.
- ❑ Di default produce una riga di output per pacchetto, che di solito inizia con un timestamp, a cui segue l'indicazione del protocollo ed una serie di dati aggiuntivi significativi per il particolare protocollo

```
11:05:40.331077 ARP, Request who-has 10.15.0.6 tell man-10-15-2-10.cdt.man, length 46
```

```
11:06:35.085537 IP davide.cdt.man.17500 > 10.15.255.255.17500: UDP, length 131
```

```
11:09:07.422433 IP a23-6-112-130.deploy.static.akamaitechnologies.com.http > kali.37806: Flags [S.], seq 454838011, ack 1826699044, win 14480, options [mss 1460,sackOK,TS val 1342016717 ecr 3558140594,nop,wscale 7], length 0
```

# TCPDUMP: Espressioni

- ❑ L'uso di **espressioni** (o filtri) permette di filtrare il traffico catturato.
- ❑ Combinando più espressioni è possibile focalizzarsi solo sulla tipologia di traffico che realmente interessa, rendendo così tcpdump uno strumento molto potente e molto flessibile
- ❑ Esistono 3 tipologie principali di espressioni: `type`, `dir` e `proto`
  - ❑ Sono di tipo `type`: che identificano a quale tipo di oggetti si riferiscano le opzioni (es: `host`, `net`, `port`, `portrange`)
  - ❑ `dir` (direction): specificano la direzione di trasferimento dei frame (es: `src` e `dst`)
  - ❑ `proto` (protocol): restringono il match a un particolare protocollo. Sono molte, tra cui: `ip`, `ipv6`, `arp`, `tcp`, `udp`, `icmp` e molte altre
- ❑ Per una lista delle espressioni ci si può riferire alla pagina di manuale di `pcap-filter`
  - ❑ <https://www.tcpdump.org/manpages/pcap-filter.7.html>



# TCPDUMP: esempi di base

- ❑ Visualizzare le interfacce di rete disponibili

```
root@kali:~# tcpdump -D
```

- ❑ Visualizzare il traffico di rete su tutte le interfacce o su una specifica

```
root@kali:~# tcpdump -i any
```

```
root@kali:~# tcpdump -i eth0
```

Cosa succede a lanciare il comando su una macchina a cui si è connessi da remoto?

- ❑ Traffico di rete "raw": output verboso senza risoluzione di hostname o numero di porta, numeri di sequenza assoluti e timestamp in formato "human readable"

```
root@kali:~# tcpdump -ttttnnvvS
```

# TCPDUMP: esempi di base

- ❑ Visualizzare il traffico che coinvolge un particolare indirizzo IP:

```
root@kali:~# tcpdump host 1.2.3.4
```

- ❑ Filtrare il traffico in base a IP sorgente o destinazione:

```
root@kali:~# tcpdump src 1.2.3.4  
root@kali:~# tcpdump dst 1.2.3.4
```

- ❑ Filtrare i pacchetti in base alla rete:

```
root@kali:~# tcpdump net 1.2.3.0/24
```

- ❑ Visualizzare il traffico correlato con una determinata porta:

```
root@kali:~# tcpdump port 80  
root@kali:~# tcpdump dst port http
```

Notare come sia possibile utilizzare il nome del servizio normalmente associato ad una particolare porta anziché il numero (l'associazione nome/porta si può trovare nel file `/etc/services` )

# TCPDUMP: esempi di base

- ❑ Mostrare solo il traffico legato ad un particolare protocollo

```
root@kali:~# tcpdump icmp  
root@kali:~# tcpdump ip6
```

- ❑ Filtrare il traffico specificando un range di porte a cui si è interessati:

```
root@kali:~# tcpdump portrange 21-23
```

# TCPDUMP: scrivere la cattura su file

- ❑ Risulta estremamente utile poter scrivere su file i pacchetti catturati:
  - ❑ Molto spesso la cattura si fa su un server remoto, su cui non è disponibile un ambiente grafico, ed è comodo poi trasferire il file in locale ed analizzare il traffico di rete con un tool diverso ( es: Whireshark )
  - ❑ In una cattura "live" l'output scorre molto velocemente sul monitor e può essere molto difficile individuare eventuali anomalie, problemi o individuare le stringhe che si stavano cercando, per cui è più pratico salvare il traffico ed analizzarlo con calma in un secondo momento

# TCPDUMP: scrivere la cattura su file

```
root@kali:~# tcpdump -s 0 -w file_cattura1
root@kali:~# tcpdump -i eth0 -s 0 -w file_cattura2
root@kali:~# tcpdump port 80 -w file_cattura3
```

- ❑ Se non si specifica lo snaplength (opzione `-s <len>` ), tcpdump catturerà solo un numero di byte fissato per default per ogni pacchetto (la dimensione dipende anche dalla versione di tcpdump). La dimensione di default permette di catturare sia gli header IP che TCP/UDP.
- ❑ I file scritti da tcpdump sono conosciuti come file PCAP (PCAP) e il loro formato è interpretabile da decine di applicazioni, tra cui altri analizzatori di traffico, Intrusion Detection System e molti altri, tra cui anche tool malevoli per l'individuazione e la decifrazione delle password

# TCPDUMP: lettura della cattura da file

- ❑ Così come può scrivere la cattura su di un file, tcpdump è in grado di leggere un file in formato PCAP e visualizzarne il contenuto

```
root@kali:~# tcpdump -r file_cattura
```

- ❑ Visualizzando il contenuto di un file è possibile applicare gli stessi filtri che si utilizzerebbero nella cattura

```
root@kali:~# tcpdump port 80 -r file_cattura
```

- ❑ Ovviamente il filtraggio avviene sui pacchetti salvati nel file, per cui se nel salvataggio era stato usato un filtro diverso non sarà possibile ottenere i pacchetti del flusso originale

# TCPDUMP: combinazione di filtri

- ❑ Nell'uso pratico l'impostazione di un solo filtro non è quasi mai sufficiente a visualizzare solo il traffico a cui si è realmente interessati, ma la vera potenza di tcpdump è proprio nel combinare più filtri fino a riuscire ad isolare solo quello che si sta realmente cercando.
- ❑ Ci sono tre modi per combinare i filtri, e risultano familiari a chiunque abbia studiato programmazione:
  - ❑ AND: `and` oppure `&&`
  - ❑ OR: `or` oppure `||`
  - ❑ NOT: `not` oppure `!`

# TCPDUMP: esempi di filtri combinati

- ❑ Catturare i pacchetti che coinvolgono l'IP 10.15.8.5 e che hanno come porta di destinazione la 22

```
root@kali:~# tcpdump host 10.15.8.5 and dst port 22
```

- ❑ Catturare i pacchetti che provengono dall'IP 10.15.8.5 e che sono diretti alla porta 80

```
root@kali:~# tcpdump -i eth0 -tttt src 10.15.8.5 and dst port 80
```

- ❑ Visualizzare il traffico che va dalla rete 192.168.0.0/16 alla 10.10.0.0/16 in formato esadecimale, senza risoluzione dei nomi, e con un livello aggiuntivo di verbosità

```
root@kali:~# tcpdump -nvX src net 192.168.0.0/16 and dst net 10.10.0.0/16
```



# TCPDUMP: esempi di filtri combinati

- ❑ Visualizzare tutto il traffico diretto a 10.15.0.244 che non sia ICMP

```
root@kali:~# tcpdump dst 10.15.0.244 and not icmp
```

- ❑ È possibile costruire filtri più complessi ma, se è necessario inserire delle parentesi, è necessario raggruppare le opzioni usando ' ' o " ", oppure, in alternativa, far precedere le parentesi da un carattere di escape "\ (" , "\ ) ". Ad esempio per filtrare il traffico che ha come sorgente l'IP 10.10.1.1 e come porta destinazione 80 oppure 22 si può scrivere:

```
root@kali:~# tcpdump 'src 10.10.1.1 and (dst port 80 or 22) '  
root@kali:~# tcpdump src 10.10.1.1 and \(dst port 80 or 22\)
```

# TCPDUMP: flag TCP

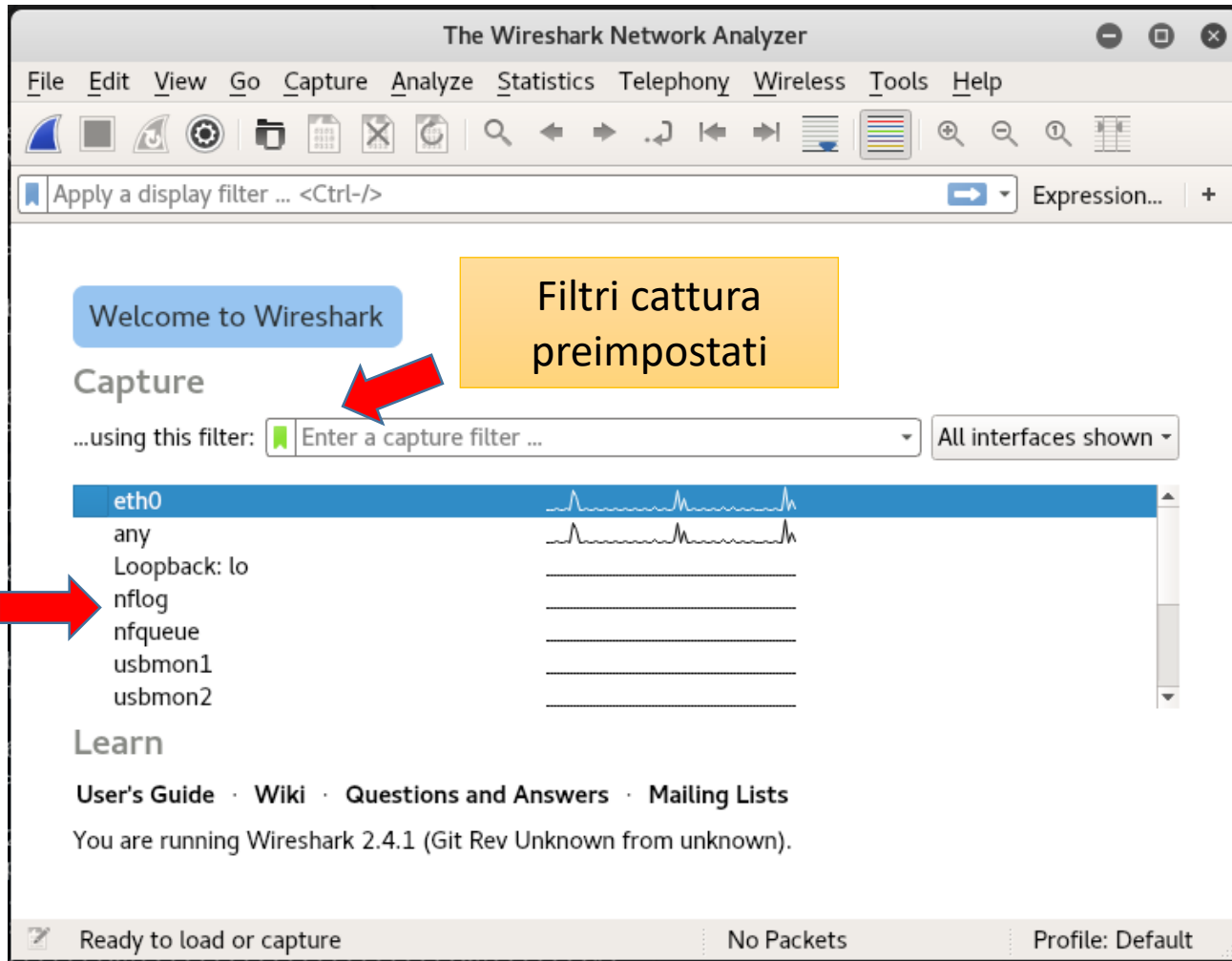
- ❑ È possibile costruire filtri in grado di mostrare solo determinati pacchetti di una connessione TCP (ad esempio solo i pacchetti SYN) attraverso un'apposita opzione applicabile al filtro di protocollo TCP

```
root@kali:~# tcpdump 'tcp[tcpflags] == tcp-syn'  
root@kali:~# tcpdump 'tcp[tcpflags] == tcp-ack'  
root@kali:~# tcpdump 'tcp[tcpflags] == tcp-rst'  
root@kali:~# tcpdump 'tcp[tcpflags] == tcp-fin'
```

# Wireshark

- ❑ Whireshark ( [www.wireshark.org](http://www.wireshark.org) ) è molto probabilmente il più diffuso tool ad interfaccia grafica per l'analisi del traffico di rete
- ❑ È in grado sia di effettuare una cattura dei pacchetti in real-time che analizzare il contenuto di un file in formato PCAP (lo stesso di tcpdump)
- ❑ Wireshark include filtri, codifica a colori e una serie di strumenti aggiuntivi che permettono di analizzare in dettaglio il traffico di rete, il comportamento dei protocolli standardizzati e di ispezionare i singoli pacchetti.
- ❑ Wireshark è disponibile per tutti i sistemi operativi più diffusi ( Unix/Linux, OSX, Windows )

# Wireshark

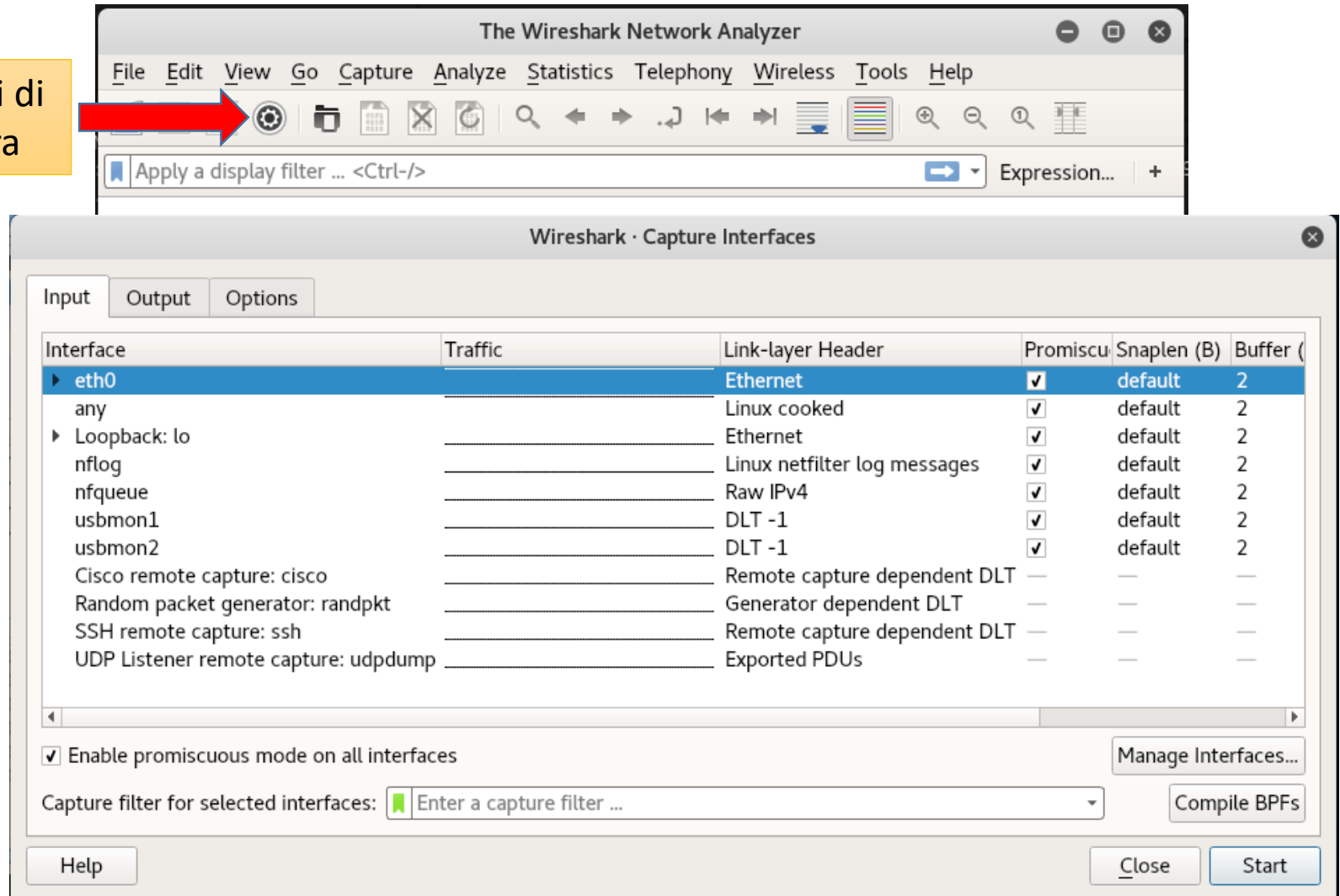


Interfacce disponibili

Filtri cattura preimpostati

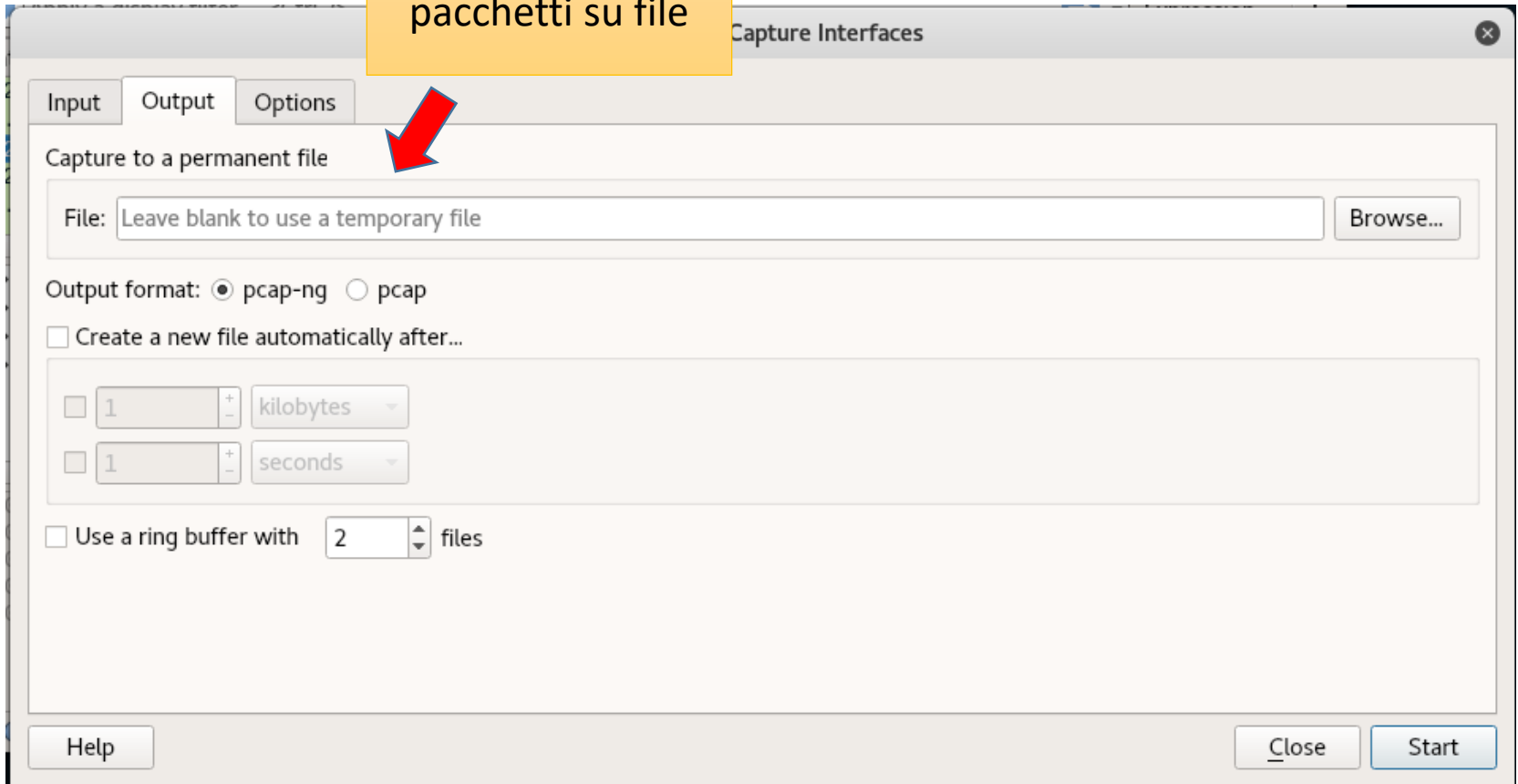
# Wireshark

Opzioni di  
cattura



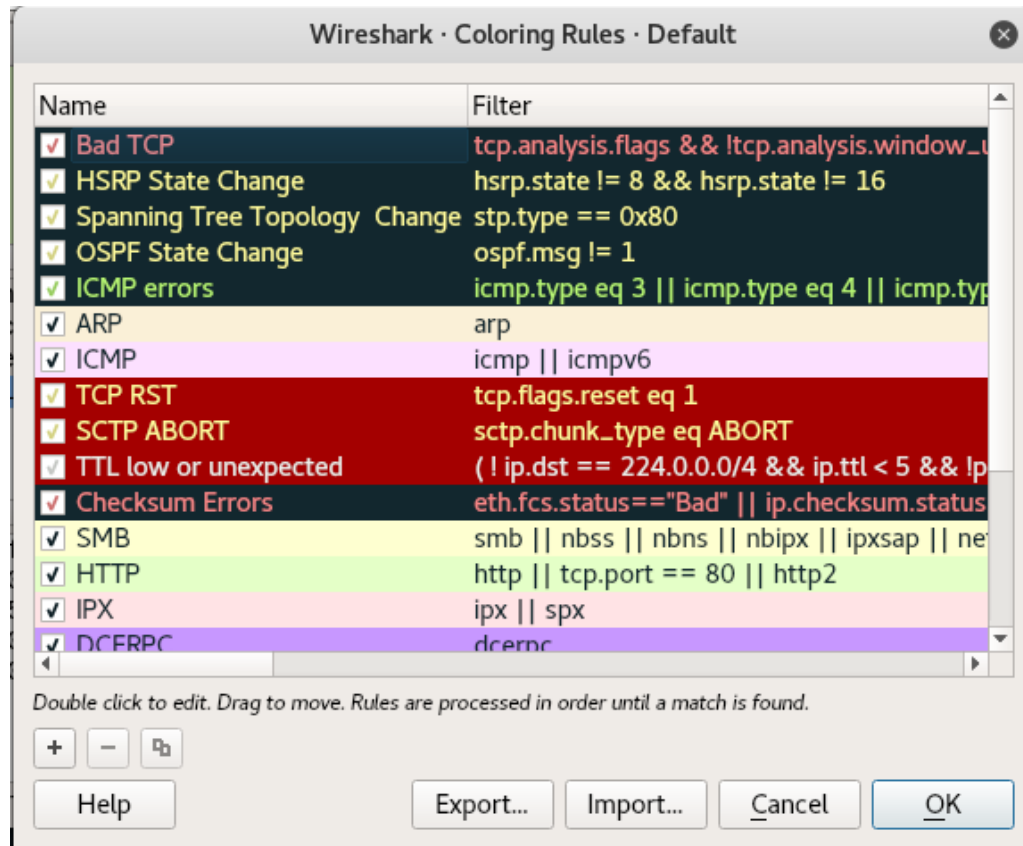
# Wireshark

Salvataggio dei  
pacchetti su file



# Wireshark

- ❑ Dal menu "View" è possibile consultare le "Coloring rules", ossia la legenda delle colorazioni usate per i pacchetti



# Wireshark: filtri

- ❑ Wireshark distingue 2 differenti tipologie di filtri:

- ❑ I filtri di cattura

- ❑ Hanno sintassi analoga a quelli di tcpdump

- ❑ Devono essere selezionati prima dell'inizio della cattura e non possono essere modificati durante la stessa

- ❑ Limitano la quantità di pacchetti che il tool visualizza (o salva su file)

- ❑ I filtri di visualizzazione

- ❑ Hanno una sintassi propria

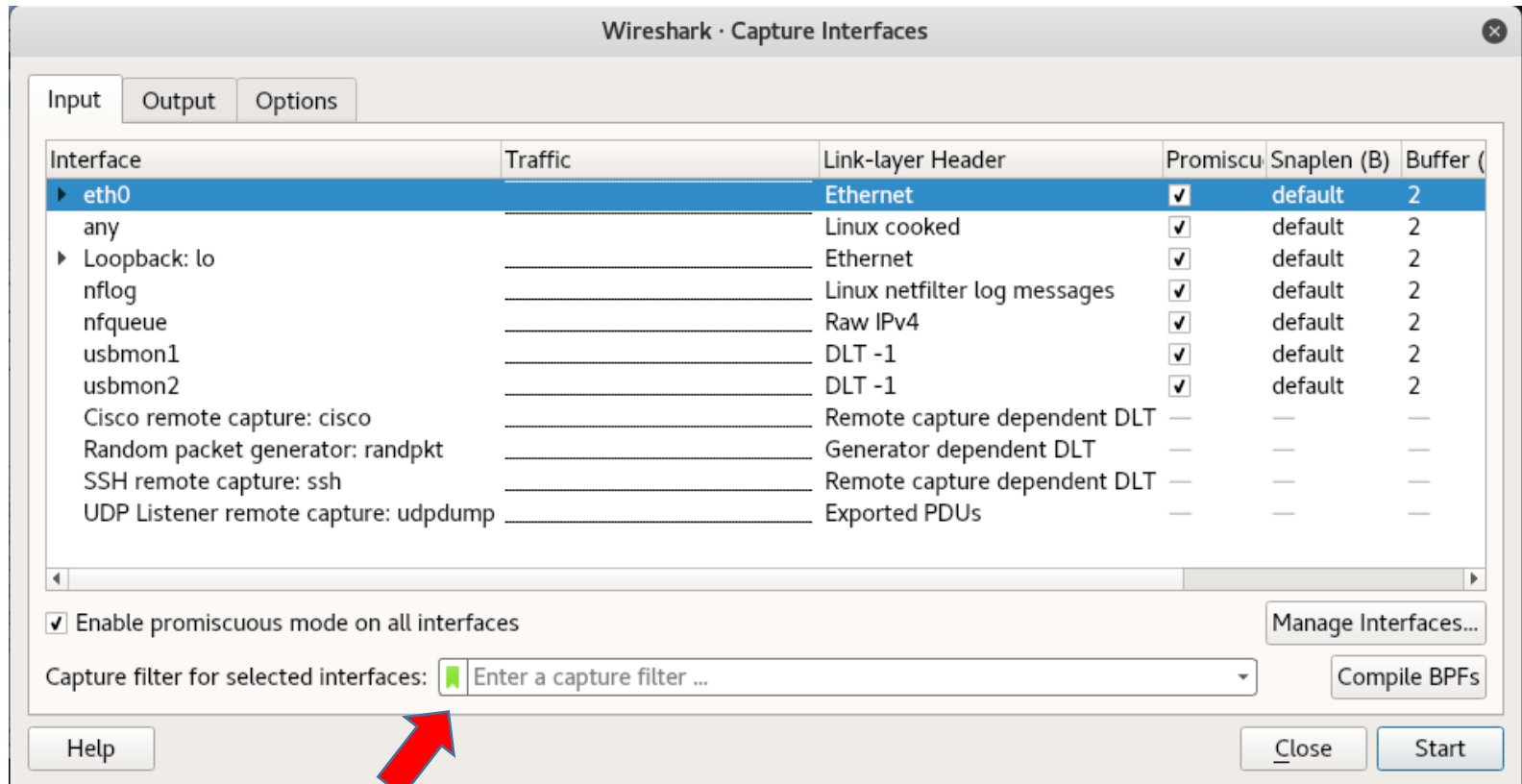
- ❑ Possono essere costruiti con un apposito editor fornito con Wireshark

- ❑ Possono essere modificati durante l'utilizzo

- ❑ Non influiscono sulla quantità di pacchetti catturati, ma solo sulla loro visualizzazione all'utente



# Wireshark: filtri di cattura



Impostazione  
filtro di cattura

# Wireshark: filtri di visualizzazione

❑ Sono documentati su

<http://wiki.wireshark.org/CaptureFilters>

❑ Possono essere usati analogamente ai filtri di cattura per specificare

❑ Protocolli

```
udp tcp icmp
```

❑ informazioni specifiche di un pacchetto

```
ip.dst == 10.10.10.1
```

```
ip.src == 192.168.1.1
```

❑ sono componibili con operatori logici

```
ip == 192.168.1.1 &&! ( tcp.port eq 80 or  
tcp.port eq 25)
```

# Wireshark: visualizzazione della cattura

Selezione pacchetto



Decodifica dei campi del protocollo



Contenuto del pacchetto



\*eth0 (tcp)

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.port == 80 Expression...

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.15.0.244	2.18.240.137	TCP	74	39862 → 80
2	0.001271429	2.18.240.137	10.15.0.244	TCP	74	80 → 39862
3	0.001312913	10.15.0.244	2.18.240.137	TCP	66	39862 → 80
4	0.002351629	10.15.0.244	2.18.240.137	HTTP	354	GET /succe
5	0.002970223	2.18.240.137	10.15.0.244	TCP	66	80 → 39862

Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0

Ethernet II, Src: PcsCompu\_b8:7d:19 (08:00:27:b8:7d:19), Dst: FujitsuT\_57:68:fa (00:19:99:57:68:fa)

Internet Protocol Version 4, Src: 10.15.0.244, Dst: 2.18.240.137

- 0100 .... = Version: 4
- .... 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 52
- Identification: 0x1a2d (6701)
- Flags: 0x02 (Don't Fragment)
- Fragment offset: 0
- Time to live: 64
- Protocol: TCP (6)

```
0000 00 19 99 57 68 fa 08 00 27 b8 7d 19 08 00 45 00  ...Wh... '.)...E.
0010 00 34 1a 2d 40 00 40 06 22 f9 0a 0f 00 f4 02 12  .4.-@.@. ".....
0020 f0 89 9b b6 00 50 3d 48 76 2c a2 94 5f c5 80 10  ....P=H v,.....
0030 00 e5 fd c4 00 00 01 01 08 0a 68 cc 2c eb 45 fa  .... .h.,.E.
0040 c0 77  .w
```

Version (ip.version), 1 byte      Packets: 91 · Displayed: 91 (100.0%) · Dropped: 0 (0.0%)      Profile: Default

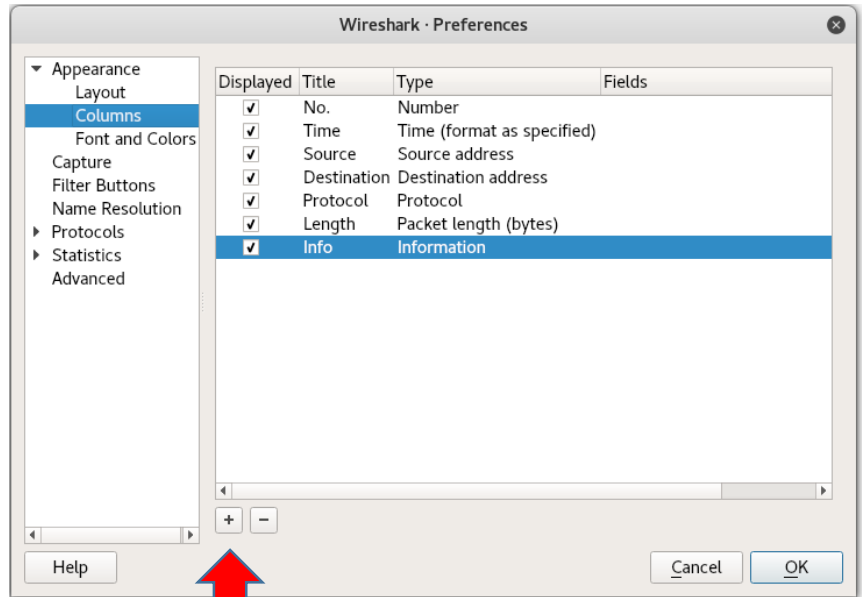
# Visualizzazione elementi aggiuntivi

Come impostazione di default wireshark visualizza:

- ID numerico
- Timestamp
- indirizzo IP sorgente e destinazione
- protocollo
- lunghezza del pacchetto (in byte)
- informazioni sul pacchetto (dipende dal protocollo analizzato, es: "TCP segment of a reassembled PDU")

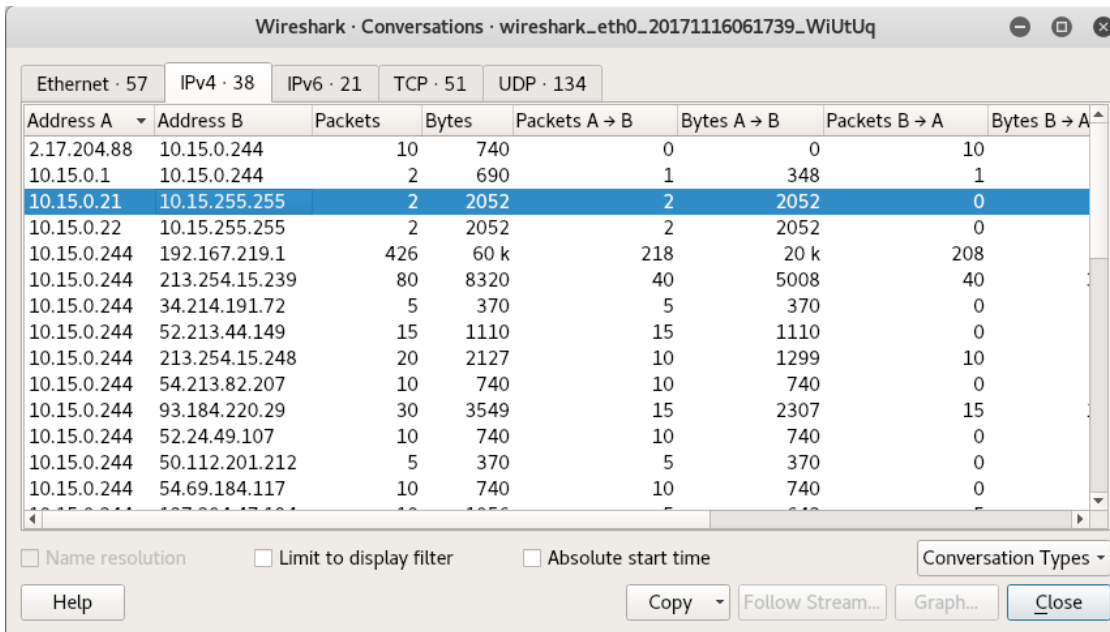
È possibile aggiungere ulteriori colonne per visualizzare ulteriori campi del pacchetto eventualmente anche non previsti (opzione "custom")

Edit -> Preferences ->  
Columns -> Add



# Isolare una "network conversation"

- ❑ Una "network conversation" (comunicazione di rete) rappresenta il traffico scambiato da due specifici endpoint
- ❑ Utile per restringere le analisi tra due endpoint
- ❑ Si può ottenere con la definizione manuale di filtri (sia per la cattura che per la visualizzazione), ma si può usare anche l'apposito strumento guidato "conversations" che crea il filtro in maniera automatica (Statistics -> Conversations )



Wireshark · Conversations · wireshark\_eth0\_20171116061739\_WiUtUq

Ethernet · 57   IPv4 · 38   IPv6 · 21   TCP · 51   UDP · 134

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A
2.17.204.88	10.15.0.244	10	740	0	0	10	10
10.15.0.1	10.15.0.244	2	690	1	348	1	1
10.15.0.21	10.15.255.255	2	2052	2	2052	0	0
10.15.0.22	10.15.255.255	2	2052	2	2052	0	0
10.15.0.244	192.167.219.1	426	60 k	218	20 k	208	208
10.15.0.244	213.254.15.239	80	8320	40	5008	40	40
10.15.0.244	34.214.191.72	5	370	5	370	0	0
10.15.0.244	52.213.44.149	15	1110	15	1110	0	0
10.15.0.244	213.254.15.248	20	2127	10	1299	10	10
10.15.0.244	54.213.82.207	10	740	10	740	0	0
10.15.0.244	93.184.220.29	30	3549	15	2307	15	15
10.15.0.244	52.24.49.107	10	740	10	740	0	0
10.15.0.244	50.112.201.212	5	370	5	370	0	0
10.15.0.244	54.69.184.117	10	740	10	740	0	0

Name resolution    Limit to display filter    Absolute start time   Conversation Types ▾

Help   Copy ▾   Follow Stream...   Graph...   Close

# Timestamp e formati

The screenshot shows the Wireshark application window titled '\*eth0'. The 'View' menu is open, and the 'Time Display Format' submenu is also open. The main packet list shows several entries with their protocols and lengths. The packet details pane shows the selected packet's structure.

No.	Time	Source	Destination	Protocol	Length	Info
73	0.000000	192.168.1.1	192.168.1.2	ICMPv6	86	Neighbor Solicitation
74	0.000000	192.168.1.2	192.168.1.1	ARP	60	Who has 192.168.1.1
75	0.000000	192.168.1.1	192.168.1.2	ICMPv6	86	Neighbor Solicitation
76	0.000000	192.168.1.2	192.168.1.1	ARP	60	Who has 192.168.1.1
77	0.000000	192.168.1.1	192.168.1.2	UDP	1026	40082 → 30000

The 'Time Display Format' submenu is open, showing various options for displaying timestamps. The 'Date and Time of Day (1970-01-01 01:02:03.123456)' option is selected, with the keyboard shortcut 'Ctrl+Alt+1'.

Option	Keyboard Shortcut
Date and Time of Day (1970-01-01 01:02:03.123456)	Ctrl+Alt+1
Year, Day of Year, and Time of Day (1970/001 01:02:03.123456)	
Time of Day (01:02:03.123456)	Ctrl+Alt+2
Seconds Since 1970-01-01	Ctrl+Alt+3
Seconds Since Beginning of Capture	Ctrl+Alt+4
Seconds Since Previous Captured Packet	Ctrl+Alt+5
Seconds Since Previous Displayed Packet	Ctrl+Alt+6
UTC Date and Time of Day (1970-01-01 01:02:03.123456)	Ctrl+Alt+7
UTC Year, Day of Year, and Time of Day (1970/001 01:02:03.123456)	
UTC Time of Day (01:02:03.123456)	Ctrl+Alt+8
Automatic (from capture file)	
Seconds	
Tenths of a second	
Hundredths of a second	
Milliseconds	
Microseconds	
Nanoseconds	
<input type="checkbox"/> Display Seconds With Hours and Minutes	

# Analisi dei protocolli

- ❑ Wireshark offre delle funzionalità avanzate di analisi dei protocolli di rete, ad esempio rende possibile intercettare comunicazioni VoIP non cifrate

The screenshot shows the Wireshark interface for a capture file named 'dump\_voip.pcap'. The main pane displays a list of captured packets:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.15.250.20	10.15.1.250	SIP/SDP	1194	Request: INVITE sip:8998@10.15.1.250;user=ph
2	0.000680	10.15.1.250	10.15.250.20	SIP	560	Status: 401 Unauthorized
3	0.037164	10.15.250.20	10.15.1.250	SIP	401	Request: ACK sip:8998@10.15.1.250;user=phone
4	0.048931	10.15.250.20	10.15.1.250	SIP/SDP	1363	Request: INVITE sip:8998@10.15.1.250;user=ph
5	0.052610	10.15.1.250	10.15.250.20	SIP	538	Status: 100 Trying

The details pane for the selected packet (Frame 1) shows the following structure:

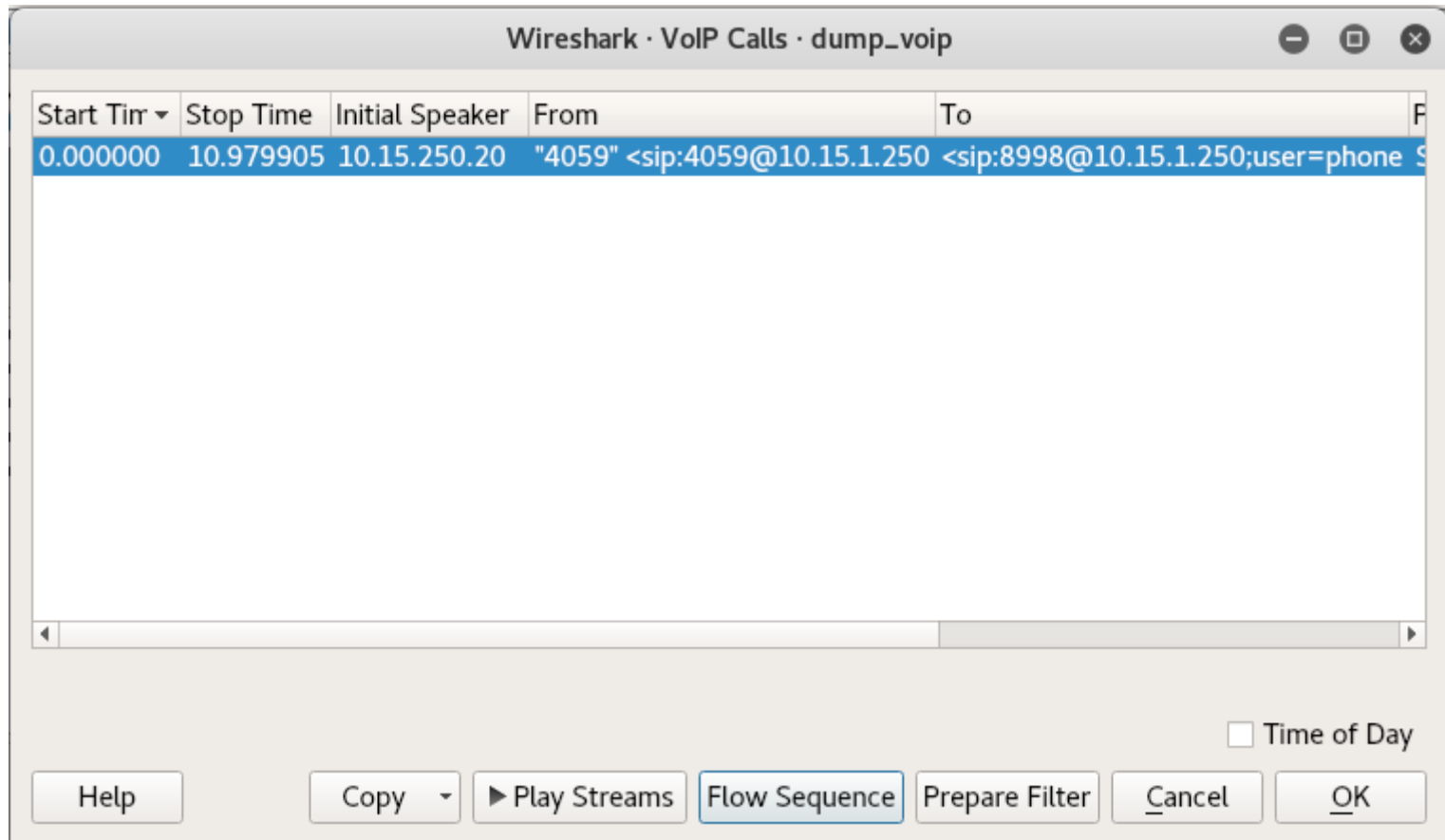
- Frame 1: 1194 bytes on wire (9552 bits), 1194 bytes captured (9552 bits)
- Ethernet II, Src: SnomTech\_74:2f:4a (00:04:13:74:2f:4a), Dst: HewlettP\_ff:cc:87 (00:17:a4:ff:cc:87)
- Internet Protocol Version 4, Src: 10.15.250.20, Dst: 10.15.1.250
- User Datagram Protocol, Src Port: 32768, Dst Port: 5060
- Session Initiation Protocol (INVITE)

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000 00 17 a4 ff cc 87 00 04 13 74 2f 4a 08 00 45 a0 ..... .t/J..E.
0010 04 9c 00 00 40 00 40 11 25 85 0a 0f fa 14 0a 0f ....@.@. %.....
0020 01 fa 80 00 13 c4 04 88 9f 61 49 4e 56 49 54 45 ..... .aINVITE
0030 20 73 69 70 3a 38 39 39 38 40 31 30 2e 31 35 2e ..... sip:899 8@10.15.
0040 31 2e 32 35 30 3b 75 73 65 72 3d 70 68 6f 6e 65 1.250;us er=phone
0050 20 53 49 50 2f 32 2e 30 0d 0a 56 69 61 3a 20 53 SIP/2.0 ..Via: S
0060 49 50 2f 32 2e 30 2f 55 44 50 20 31 30 2e 31 35 IP/2.0/U DP 10.15
```

# Intercettazione chiamate VoIP

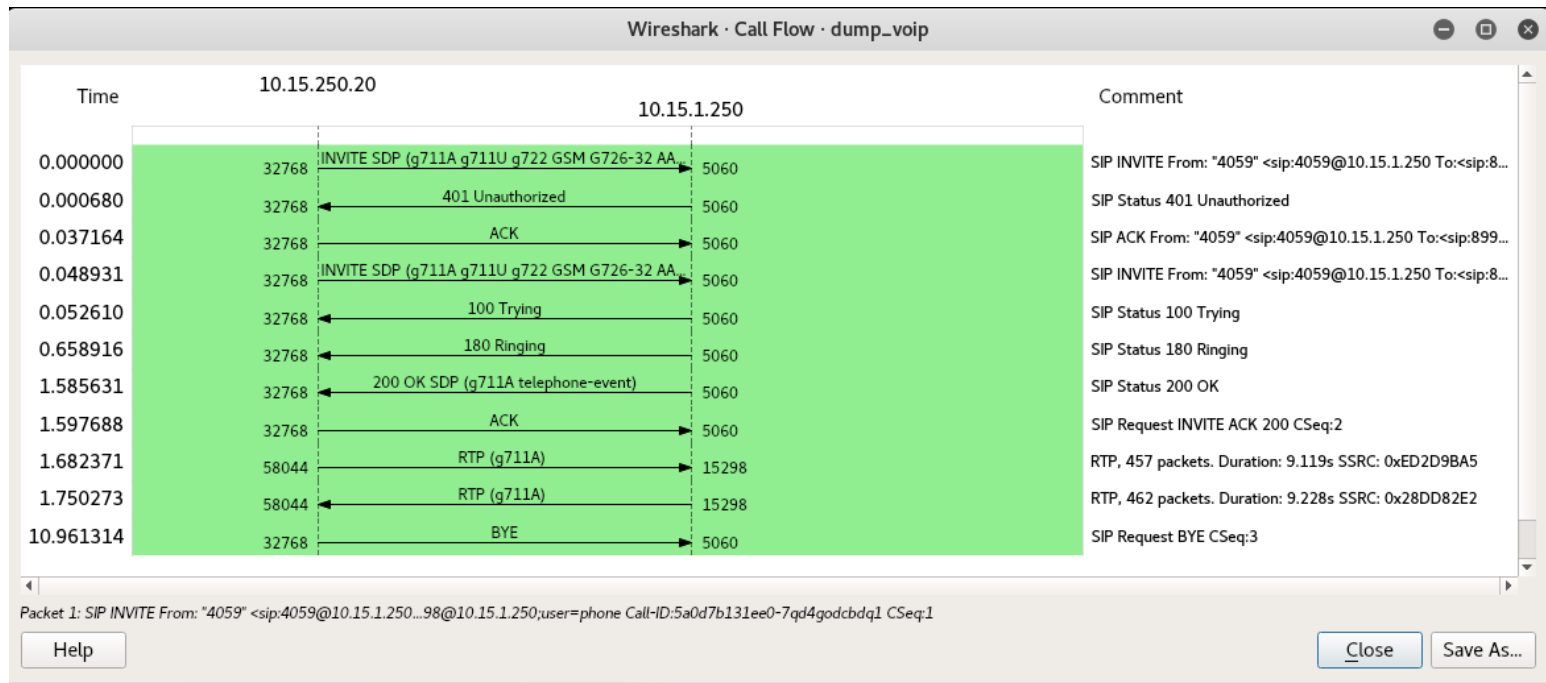
- ❑ Nel menu "Telephony" si possono individuare le chiamate VoIP intercettate





# Intercettazione chiamate VoIP

- ❑ Wireshark è in grado di visualizzare il flusso delle comunicazioni del protocollo SIP



# Intercettazione chiamate VoIP

- Allo stesso modo è in grado di visualizzare un'analisi completa delle statistiche di funzionamento del flusso audio su RTP

The screenshot shows the Wireshark RTP Stream Analysis window for a dump of VoIP traffic. The window is titled "Wireshark · RTP Stream Analysis · dump\_voip". It displays statistics for two RTP streams: one from 10.15.1.250:15298 to 10.15.250.20:58044 (Forward) and another from 10.15.250.20:58044 to 10.15.1.250:15298 (Reverse).

**Forward Stream Statistics:**

- SSRC: 0x28dd82e2
- Max Delta: 34.75 ms @ 256
- Max Jitter: 2.57 ms
- Mean Jitter: 0.50 ms
- Max Skew: -9.76 ms
- RTP Packets: 462
- Expected: 462
- Lost: 0 (0.00 %)
- Seq Errs: 0
- Start at: 1.750273 s @ 16
- Duration: 9.23 s
- Clock Drift: -31 ms
- Freq Drift: 7973 Hz (-0.34 %)

**Reverse Stream Statistics:**

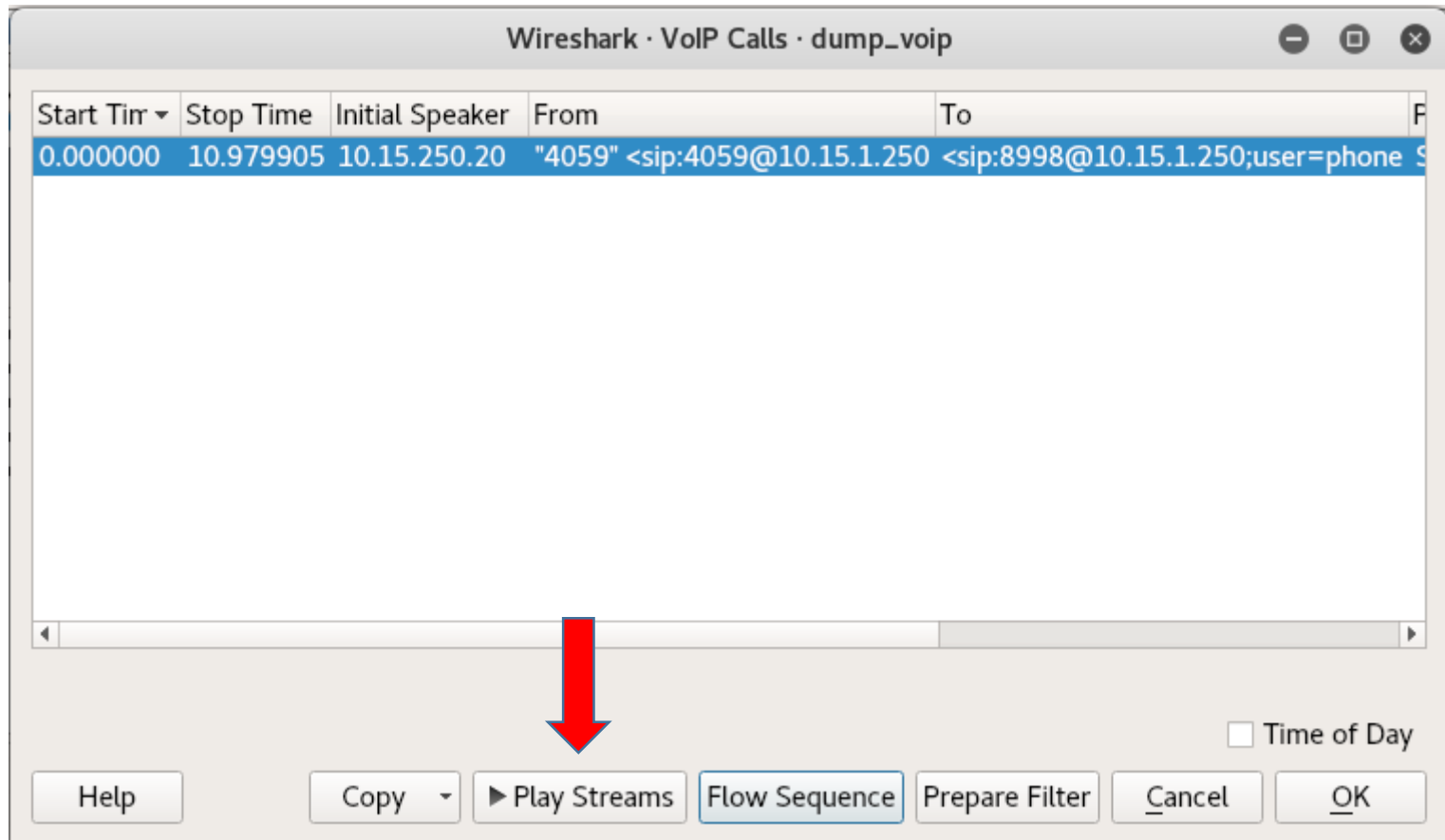
- SSRC: 0x00000000
- Max Delta: 0.00 ms @ 0
- Max Jitter: 0.00 ms
- Mean Jitter: 0.00 ms
- Max Skew: 0.00 ms
- RTP Packets: 0
- Expected: 1
- Lost: 1 (100.00 %)
- Seq Errs: 0
- Start at: 0.000000 s @ 0
- Duration: 0.00 s
- Clock Drift: 0 ms
- Freq Drift: 1 Hz (0.00 %)

**Packet List Table:**

Packet	Sequence	Delta (ms)	Jitter (ms)	Skew	Bandwidth
16	57405	0.00	0.00	0.00	
18	57406	15.06	0.31	4.94	
20	57407	20.01	0.29	4.93	
22	57408	20.00	0.27	4.93	
24	57409	20.02	0.26	4.90	
26	57410	19.98	0.24	4.93	
28	57411	19.98	0.23	4.95	
30	57412	20.04	0.22	4.91	
32	57413	19.97	0.20	4.94	
34	57414	20.01	0.19	4.93	
36	57415	19.99	0.18	4.94	
38	57416	20.00	0.17	4.94	
40	57417	19.99	0.16	4.95	
42	57418	20.01	0.15	4.94	
44	57419	20.00	0.14	4.94	
46	57420	19.99	0.13	4.95	
48	57421	20.00	0.12	4.95	
50	57422	20.00	0.12	4.95	
52	57423	20.00	0.11	4.94	
54	57424	20.00	0.10	4.94	
56	57425	20.15	0.11	4.79	
58	57426	19.83	0.11	4.95	
60	57427	20.01	0.10	4.95	
62	57428	20.00	0.10	4.95	
64	57429	20.01	0.09	4.94	
66	57430	19.99	0.09	4.96	
68	57431	20.01	0.08	4.95	
70	57432	20.02	0.08	4.93	
72	57433	19.99	0.07	4.95	
74	57434	20.03	0.07	4.91	
76	57435	19.97	0.07	4.94	
78	57436	20.00	0.06	4.95	

# Intercettazione chiamate VoIP

- ❑ Wireshark può lanciare un player multimediale per riprodurre il contenuto vocale della telefonata



# TSHARK

- ❑ tshark ( <https://www.wireshark.org/docs/man-pages/tshark.html> ) è sostanzialmente la versione a riga di comando di wireshark, di cui condivide parte del codice (sono parte dello stesso progetto)
- ❑ tshark si comporta sostanzialmente come tcpdump
  - ❑ Può leggere e scrivere i file in formato PCAP
  - ❑ Può applicare filtri con la stessa sintassi di tcpdump
  - ❑ Alcune delle opzioni di tcpdump hanno lo stesso significato in tshark ( purtroppo non tutte però )
- ❑ Ha alcune funzionalità aggiuntive che lo rendono uno strumento ancora più potente di tcpdump
- ❑ Nella maggior parte delle distribuzioni linux non è preinstallato, ma è comunque pacchettizzato nei repository ufficiali di tutte le distro più diffuse

# TSHARK: opzioni

- ❑ `-i any` : si mette in ascolto su tutte le interfacce di rete
- ❑ `-i eth0` : si mette in ascolto solo sull'interfaccia specificata (qui eth0).
- ❑ `-D` : mostra la lista delle interfacce disponibili
- ❑ `-n` : non risolve gli hostname.
- ❑ `-q` : limita la quantità di output (quiet).
- ❑ `-t <formato>` : setta il formato di visualizzazione del timestamp del pacchetto, alcuni dei formati disponibili sono:
  - ❑ `a`: assoluto nella timezone locale senza data
  - ❑ `ad`: assoluto con indicazione della data
  - ❑ `d`: delta del pacchetto rispetto alla cattura precedente
  - ❑ `r`: relativo al primo pacchetto catturato
  - ❑ `u`: timestamp UTC senza data
  - ❑ ...
- ❑ `-V` : verbose, aumenta la quantità di dettagli stampata a video
- ❑ `-O <protocol>[, <protocol>]` : come `-V` ma limitato alla lista di protocolli forniti ad argomento. Una lista dei protocolli disponibili si può ottenere con `tshark -G protocols`

# TSHARK: opzioni

- ❑ `-x` : mostra il contenuto completo dei pacchetti (in esadecimale e ASCII)
- ❑ `-c <num>`: limita la cattura a `num` pacchetti
- ❑ `-w <nome_file>`: scrive la cattura sul file `nome_file` in formato PCAP
- ❑ `-r <nome_file>`: legge il file `nome_file` in formato PCAP
- ❑ `-f "<filtro>"` : permette di specificare un filtro di cattura, nel formato PCAP, formato anche di più espressioni connesse con operatori logici. Deve essere racchiuso tra doppi apici (es: `tshark -f "multicast or broadcast"` )
  - ❑ I filtri di cattura non funzionano nell'analisi di traffico di rete letto da file (opzione `-r <nome_file>` )
  - ❑ Si possono inserire più filtri con l'opzione `-f`, tshark li combinerà tra loro usando un `&&`

# TSHARK: opzioni

- ❑ `-Y "<filtro>"`: permette di specificare un filtro di visualizzazione. Il significato e la sintassi sono analoghi a quelli utilizzati da Wireshark (es: `tshark -Y "ip.addr == 192.168.8.10"`)
  - ❑ I filtri di visualizzazione non sono compatibili con la scrittura su file (opzione `-w <nome_file>` )
  - ❑ Si possono utilizzare nella stessa cattura sia filtri di cattura (`-f`) che filtri di visualizzazione (`-Y`), a patto che non ci siano opzioni di lettura o scrittura su file. In questo caso la visualizzazione a schermo corrisponde alla combinazione in `&&` dei due filtri

# TSHARK: opzioni

- ❑ `-T <formato>`: setta il formato di visualizzazione dell'output (es: `json`, `ps`, `tab`, `text`, `fields`, ...)
- ❑ se come opzione di formato viene usato `"fields"`, allora è necessario specificare uno o più parametri di "elaborazione" utilizzando l'opzione `-e`, tshark visualizzerà solo i campi specificati. Ad esempio per visualizzare solo numero di ordine del pacchetto catturato, gli indirizzi IP e la dimensione dello stesso si può usare:

```
root@kali:~# tshark -T fields -e frame.number -e ip.addr  
-e ip.len
```

- ❑ la lista completa dei parametri di elaborazione disponibili si può ottenere con `tshark -G fields`
- ❑ Non si possono combinare più parametri con operatori logici
- ❑ Non si può usare `-T fields ...` se è già presente l'opzione `-x`