Università degli Studi di Trento
Corso di Laurea Specialistica in Ingegneria delle Telecomunicazioni – AA 2007/08
Wireless Networks

RELAZIONE DI LABORATORIO

Laboratorio #3
CATTURA ED ANALISI DEL TRAFFICO GENERATO SU UNA RETE WIRELESS MEDIANTE IPERF E WIRESHARK

Gruppo A5
Davide Perina (128698)
Nicola Speri (129144)
Stefano Testi (128697)
Michele Vincenzi (128689)

1. INTRODUZIONE

In questo esperimento sono stati analizzati la struttura dei pacchetti e le procedure di comunicazione dello standard 802.11, mediante una procedura di cattura del traffico. Lo studio si applica quindi sia ad un *BSS* (modalità infrastrutturata), che ad un *IBSS* (modalità *ad-hoc*), entrambi realizzati in ambito *indoor*.
Le misure sono state eseguite in un ambiente "rumoroso", ovvero in un ambiente in cui erano presenti altri AP che quindi hanno dato luogo ad interferenze a causa dell'utilizzo dello stesso canale o di canali adiacenti (e quindi parzialmente sovrapposti al canale oggetto del test).

Infine è stato effettuato un paragone tra i risultati ottenuti mediante la cattura del traffico (**Wireshark**) e quelli ottenuti mediante un software di misura del traffico (**Iperf**).

2. SETUP SPERIMENTALE

I test sono stati realizzati utilizzando vari setup sperimentali.

Il primo setup (TEST A) è costituito da tre laptop:

- un laptop con server Iperf (tabella 1) connesso all'*AP* (tabella 3) mediante cavo *Ethernet*;
- un laptop con un client Iperf (tabella 1);
- un laptop che esegue lo *sniffing* (tabella 2).

Le caratteristiche dei vari laptop sono riassunte nelle seguenti tabelle.

MODELLO	Dell Latitude 110l
OS	GNU/Linux Debian 3.1 Sarge
PROCESSORE	Intel Pentium® M - Modello 725 (1.7 GHz, 400 FSB) con Hyper-Threading
RAM	Memoria DDR 333 SDRAM 512MB 333MHz Dual Channel Shared
SCHEDA WIRELESS	Intel® Pro Wireless 2200 802.11b/g

Tabella 1

MODELLO	HP Pavilion dv4378EA
OS	GNU/Linux Ubuntu 7.10
PROCESSORE	Intel Pentium® M - Modello 740 (1.73 GHz)
RAM	Memoria DDR2 SDRAM 1024MB
SCHEDA WIRELESS	Intel® Pro Wireless 2200 802.11b/g

Tabella 2

AP	Cisco Aironet 1310
FIRMWARE	12.3(8)JEA3
FUNZIONI SUPPORTATE DAL FIRMWARE	Multiple SSID (fino a 16), per ogni SSID: VLAN, authenticazione via *MAC* address, 802.1x, WPA, numero massimo di stazioni fissabile

Tabella 3

Lo schema relativo al primo setup è mostrato in figura 1:

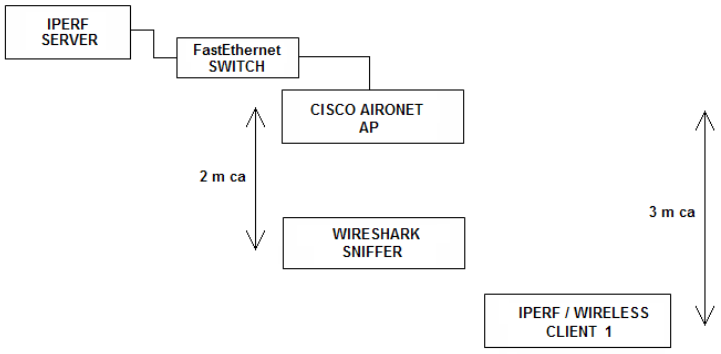


Figura 1

In questo specifico contesto, il tool Iperf è stato usato per generare uno *stream* di dati *UDP* a varie velocità. Per lo *sniffing* dei pacchetti è stato utilizzato il software open-source Wireshark (noto anche con il precedente nome di **Ethereal**): un analizzatore di pacchetti di rete che effettua una cattura in tempo reale e permette quindi una successiva analisi *off-line*.

Il secondo setup sperimentale (TEST B) è simile al primo, con l'unica differenza dell'aggiunta di un ulteriore laptop con client Iperf (caratt. tabella 1), per simulare il comportamento della rete in presenza di due stazioni che generano traffico da e verso l'*AP*. Lo schema è il seguente:

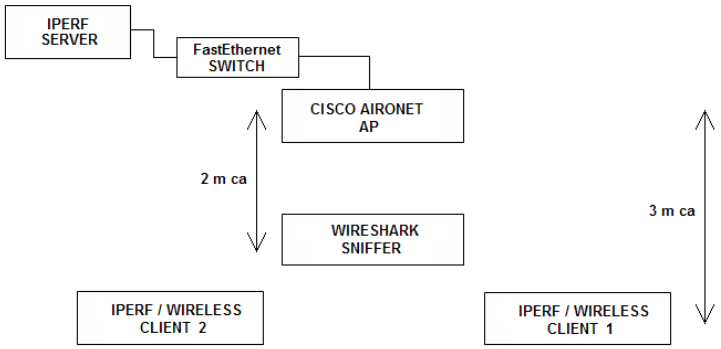


Figura 2

Per quanto concerne il terzo setup (TEST C), sono stati utilizzati due laptop (caratt. tabella 1) per la creazione di una rete *ad-hoc*: il primo configurato come server Iperf, il secondo come client Iperf. Come per il primo setup, è stato utilizzato un laptop (caratt. tabella 2) in qualità di *sniffer* per la cattura dei pacchetti come mostrato in figura 3.

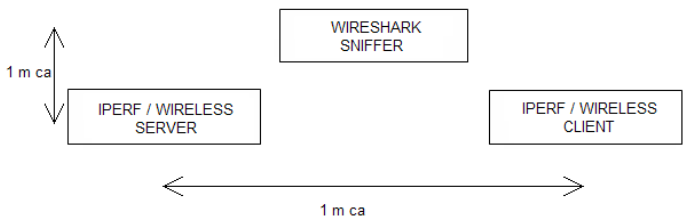


Figura 3

Infine l'ultimo setup (TEST D) è derivato dal primo, con l'aggiunta di un ulteriore *sniffer* (caratt. tabella 1, con la variante del sistema GNU/Linux Backtrack 4.0 Beta Live CD al posto di GNU/Linux Debian). Questo test è stato realizzato per verificare la possibile perdita, durante la fase di cattura degli *sniffer*, di alcuni pacchetti dello *stream* di dati generato. La configurazione viene riportata in figura 4.

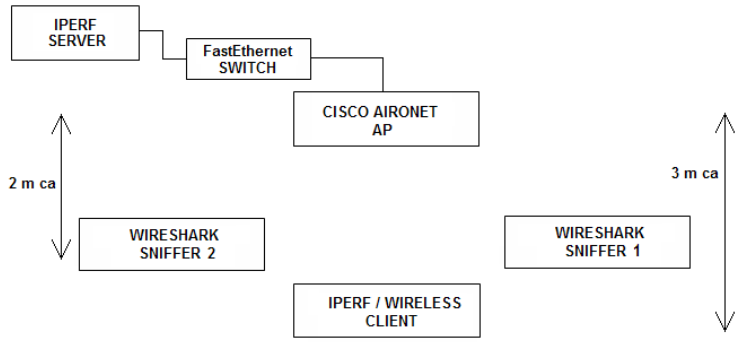


Figura 4

I test eseguiti sono quindi di 4 tipologie:

- **TEST A**: 1 *sniffer*, 1 singola stazione client Iperf con velocità impostata dall'*AP*:
  - **A1**: 802.11g velocità 54Mbps
  - **A2**: 802.11g velocità 24Mbps
  - **A3**: 802.11g velocità 12Mbps
- **TEST B**: 1 *sniffer*, 2 stazioni client Iperf con velocità impostata dall'*AP* a 54 Mbps
- **TEST C**: 1 *sniffer*, rete *ad-hoc* tra 2 stazioni (1 server Iperf e 1 client Iperf) con velocità impostata a 54 Mbps
- **TEST D:** 1 stazione client Iperf, 2 *sniffer* con velocità impostata dall'*AP* a 54 Mbps.

I **test A** sono stati effettuati per analizzare l'intervallo di tempo tra i pacchetti e quindi valutare di quanto si discosta il *throughput* misurato mediante Iperf da quello determinato da Wireshark. Tale analisi dipende però dalla particolare *NIC* del dispositivo che opera come *sniffer*, in quanto è possibile che il tool per la cattura dei pacchetti non riesca ad essere abbastanza veloce nell'acquisizione di tutto il traffico transitante nel canale, causando quindi delle "perdite" di pacchetti (intese come pacchetti rilevati correttamente a destinazione ma non visualizzati dal software).

Il **test B** è molto simile al precedente, con l'eccezione di due client collegati al server Iperf e uno *sniffer* che cattura i pacchetti di entrambi. Le considerazioni sono analoghe alle precedenti, con la differenza che il confronto del *throughput* viene effettuato sul traffico aggregato misurato dai due differenti software.

Nel **test C** si passa dalla rete infrastrutturata alla rete *ad-hoc*. In questo esperimento si intende confrontare i risultati ottenuti dallo *sniffing* in quest'ultima topologia di rete con quelli ottenuti dal test **A1** (rete infrastrutturata alla stessa *datarate*).

Per quanto riguarda il **test D** si è ritornati alla configurazione infrastrutturata in cui è presente un solo client Iperf, ma è stato aggiunto un ulteriore *sniffer*. Questo tipo di test è stato svolto per osservare le eventuali differenze nell'acquisizione del traffico catturato, che potrebbero verificarsi a causa dell'utilizzo di due NIC diverse, ma anche a causa delle differenze hardware e software dei due laptop (*live CD* piuttosto che sistema operativo residente su disco).

Per consentire la cattura mediante il software Wireshark, i laptop sui cui esso è stato posto in esecuzione sono stati configurati in modalità *Monitor*: questa configurazione permette di eseguire lo *sniffing* dei pacchetti in modo completamente passivo, potendoli catturare sul canale wireless specificato senza dover associarsi all'*AP* o alla rete *ad-hoc*. In questa modalità quindi non è possibile trasmettere alcun pacchetto perché non si è eseguita la fase di associazione. Se la scheda di rete wireless si trova in modalità *monitor* non viene verificata la correttezza del *CRC* contenuto nei vari pacchetti ricevuti, e quindi è possibile che alcuni di essi siano corrotti.

Per poter iniziare i test, sono state quindi configurate le schede wireless dei laptop atti alla cattura dei pacchetti in *Monitor Mode*, nel modo seguente:

```
iwconfig eth0 mode monitor channel 11
```

i cui parametri utilizzati hanno il seguente significato:

`eth0`	Nome dell'interfaccia di rete wireless
`mode monitor`	Modalità di *sniffing* dei pacchetti sul canale wireless
`channel 11`	Canale sul quale l'interfaccia di rete si pone in ascolto

Successivamente è necessario avviare il tool Iperf in modalità server su una stazione e in modalità client sull'altra, con le stesse modalità riportate nelle esperienze precedenti, ma con l'eccezione di una durata inferiore per il test, data l'elevata mole di dati che viene catturata dal software Wireshark anche in brevi intervalli di tempo. La durata del test è stata scelta quindi di soli 10 secondi.
Una volta avviata la fase di cattura, Wireshark memorizza tutti i pacchetti che è in grado di ricevere, mostrando la cronologia di arrivo e varie informazioni esportabili in un formato di file dedicato.

Come menzionato nelle precedenti relazioni, dopo un'attenta analisi dei canali occupati, effettuata con il software **Netstumbler**, si è deciso di utilizzare il **canale numero 11** (2451-2473 MHz, fc=2462 MHz) in quanto è risultato essere quello a minima interferenza. Infatti, come accennato nella parte introduttiva, l'ambiente in cui sono stati effettuati i test è caratterizzato dalla presenza di altri *AP* operativi sullo stesso canale o su canali adiacenti che potrebbero causare interferenze di tipo elettromagnetico, perdita di pacchetti e quindi riduzione del *throughput*. Per questo l'ambiente di test viene definito "rumoroso", facendo quindi riferimento ad emissioni di altre reti 802.11.

3. RICHIAMI TEORICI

Come accennato precedentemente, in questa esperienza l'obiettivo è l'analisi del traffico generato in una rete wireless conforme allo standard 802.11. Oltre a tale tipo di rete, i software utilizzati in questo test operano anche su altre tipologie di reti (ad esempio Ethernet).

Generalmente l'operazione di intercettazione dei dati transitanti su una rete viene denominata "*sniffing*". Tale procedura può essere svolta sia per scopi legittimi (ad esempio per l'individuazione di malfunzionamenti all'interno della rete stessa), sia a scopi illeciti (lettura delle informazioni inviate sul canale contenenti password o altri dati sensibili).

Mediante l'analisi del traffico transitante sul canale è possibile studiare quali sono le procedure di comunicazione che avvengono tra le varie stazioni (o gli *Access Point*) all'interno della rete. In particolare si riescono a distinguere i veri e propri dati dall'*overhead* che viene aggiunto ai vari pacchetti per garantire una corretta comunicazione sul canale.
Infatti, tramite l'utilizzo di *sniffer* come Wireshark si riescono ad intercettare informazioni contenute nell'*header* dello strato fisico dello standard 802.11, nell'*header MAC*, nell'*header LLC*, nell'*header IP* e *UDP*. In questa esperienza si focalizzerà l'attenzione sull'*header* dello strato *MAC* e, marginalmente su qualche parametro dell'*header* dello strato fisico (ampiamente descritto nell'esperienza #1).

Come indicato nello standard, la struttura di un frame *MAC* è la seguente:

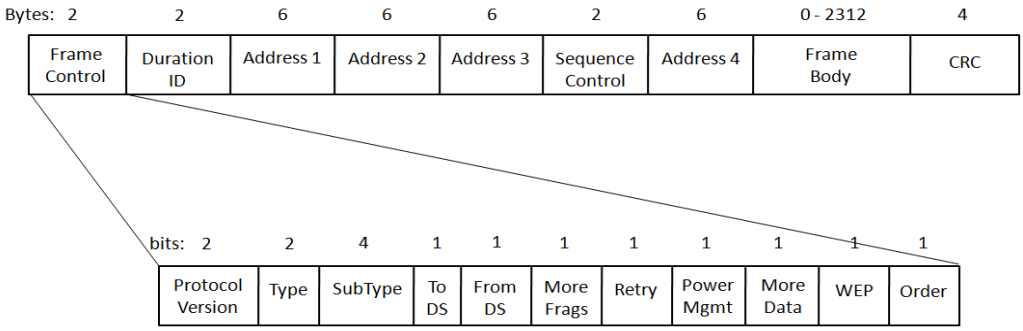


Figura 5

Si può osservare che la lunghezza complessiva dell'*header* di un frame *MAC* è di 34 Bytes ed è costituito dai seguenti campi:

1. Il campo *Frame Control* contiene informazioni relative ad alcuni parametri della trasmissione: esso infatti è costituito dai seguenti sotto-campi:

   - **Protocol Version**: questo campo indica la versione/revisione del protocollo. Un dispositivo che riceve un frame con una versione di protocollo differente da quella da esso supportata, scarta il frame senza indicarlo alla stazione trasmittente o al livello *LLC*.

   - **Type** e **SubType:** indicano la funzione del frame. Esistono tre tipologie di frame: *Control*, *Data* e *Management*

frame. I vari tipi e sottotipi di frame sono specificati nello standard 802.11.

- **ToDS:** questo bit è impostato a 1 nei frame di tipo *Data* destinati all'*AP* e che dovranno essere inoltrati al *DS* (*Distribution System*); è impostato a 0 in tutti gli altri casi.

- **FromDS:** questo sotto-campo è settato a 1 nei frame di tipo *Data* provenienti dal *DS*; è impostato a 0 in tutti gli altri frame.

- **More Fragments:** è impostato a 1 in tutti i frame di tipo *Data* o *Management* che vengono frammentati; è settato a 0 in tutti gli altri frame.

- **Retry:** è settato a 1 in tutti i frame di tipo *Data* o *Management* per indicare una ritrasmissione di pacchetti. Una stazione ricevente utilizza questa indicazione per eliminare i frame duplicati (dovuti ad esempio alla perdita del pacchetto stesso o dell'*ACK* associato). Tale campo è impostato a 0 in tutti gli altri casi.

- **Power Management:** il suo valore indica la modalità in cui la stazione si troverà dopo la trasmissione di un frame. Tale campo sarà pari a 1 se la stazione entrerà nella modalità *Power-Save mode* e pari a 0 se la stazione entrerà nella modalità *active-mode*. Questo campo è sempre settato a 0 nei frame trasmessi da un *AP*.

- **More Data:** è usato per indicare ad una stazione che si trova nella modalità *power-save mode* della presenza di ulteriori frame nel buffer dell'*AP*, pronti per la trasmissione. Questo campo è pari a 1 per indicare l'esistenza di almeno un frame addizionale diretto alla stessa stazione, mentre viene impostato a 0 in tutti gli altri frame. Tale campo inoltre può assumere il valore 1 anche nei frame di tipo *Data* trasmessi da una stazione nella modalità *Contention Free* (ossia quando il *Point Coordinator* effettua il *polling*) in risposta ad un *CF-Poll* per indicare che la stazione ha ancora almeno un frame da trasmettere memorizzato nel buffer.

- **WEP:** specifica se viene utilizzato l'algoritmo crittografico *WEP*. Se impostato a 1, il campo *Frame Body* contiene informazioni crittate con l'algoritmo *WEP*, mentre se impostato a 0 indica che per i dati contenuti all'interno del frame non si utilizza questo tipo di crittazione. Tale campo può assumere il valore 1 solamente nei frame di tipo *Data* o frame di tipo *Management* con sottotipo *Authentication*.

- **Order:** se settato a 1, questo campo indica che per la trasmissione del frame è stata utilizzata la *Strictly-Ordered service class* mentre è impostato a 0 per tutti gli altri frame.

2. Il campo *Duration ID* viene utilizzato nei frame di tipo *Control* e sottotipo *Power Save* (*PS-Poll*) per indicare l'identificativo di associazione (*AID – Association Identity*) della stazione che ha trasmesso il frame. L'*AID* è un identificativo che viene assegnato dall'*AP* alla stazione durante la fase di associazione. In tutti gli altri frame tale campo viene utilizzato per aggiornare il *Network Allocation Vector* (*NAV*), in quanto contiene il tempo previsto per il quale il canale rimarrà occupato a causa della trasmissione del frame (questo valore varia dal tipo/sottotipo di frame inviato).

3. I campi di indirizzo contengono (a seconda del tipo di frame) gli indirizzi *MAC* delle entità presenti nella rete, come indicato in tabella 4.

ToDS	FromDS	Address1	Address2	Address3	Address4
0	0	DA	SA	BSSID	N/A
0	1	DA	BSSID	SA	N/A
1	0	BSSID	SA	DA	N/A
1	1	RA	TA	DA	SA

Tabella 4

DA: *Destination Address* (indirizzo *MAC* dell'entità o delle entità finali a cui si vuole inviare l'*MSDU*);
SA: *Source Address* (indirizzo *MAC* dell'entità o delle entità che hanno generato l'*MSDU*);
BSSID: *BSS Identifier* identificativo di una particolare *BSS* (generalmente nelle reti con infrastruttura è l'indirizzo *MAC* dell'interfaccia wireless dell'*AP*);
TA: *Transmitter Address* (indirizzo *MAC* della stazione che ha trasmesso il frame sul mezzo wireless);
RA: *Receiver Address* (indirizzo *MAC* della stazione intermedia alla quale si deve inviare l'*MSDU* per far in modo che arrivi alla stazione di destinazione).

4. Il campo *Sequence Control* è a sua volta costituito da due sottocampi: *Fragment Number* e *Sequence Number*. Il primo indica il numero di frammento di un determinato frame (è impostato a 0 se non viene utilizzata la frammentazione) mentre il secondo indica il numero di frame in modulo 4096. Quest'ultimo campo viene utilizzato per evitare eventuali duplicazioni che si potrebbero verificare nelle ritrasmissioni.

5. Il campo *Frame Body* contiene le informazioni associate al tipo e sottotipo di frame in cui sono incapsulate.

6. Il campo *CRC* è costituito da 32 bit, che servono per la verifica della presenza di eventuali errori all'interno del frame stesso. Il calcolo del valore di controllo (*FCS*) viene effettuato su tutti i campi del *MAC header*, compreso il *Frame Body*.

La struttura dell'*header* di livello fisico dello standard 802.11 è già stata analizzata nel dettaglio nella relazione #1. Occorre ricordare che, in generale, l'*overhead* generato dal livello fisico è costituito dal cosiddetto *PLPC preamble* e *PLPC header* che non verranno trattati nel dettaglio.

4. RISULTATI SPERIMENTALI

In questo paragrafo viene visualizzata la struttura del traffico intercettato nella rete 802.11 dallo *sniffer*.

La prima considerazione da evidenziare è come in realtà l'ambiente sia effettivamente "rumoroso": assieme al traffico generato da Iperf, sul canale è presente anche il traffico di altre reti wireless, come dimostra la figura 6, relativa ad una cattura isolata sul canale con il software Iperf inattivo.

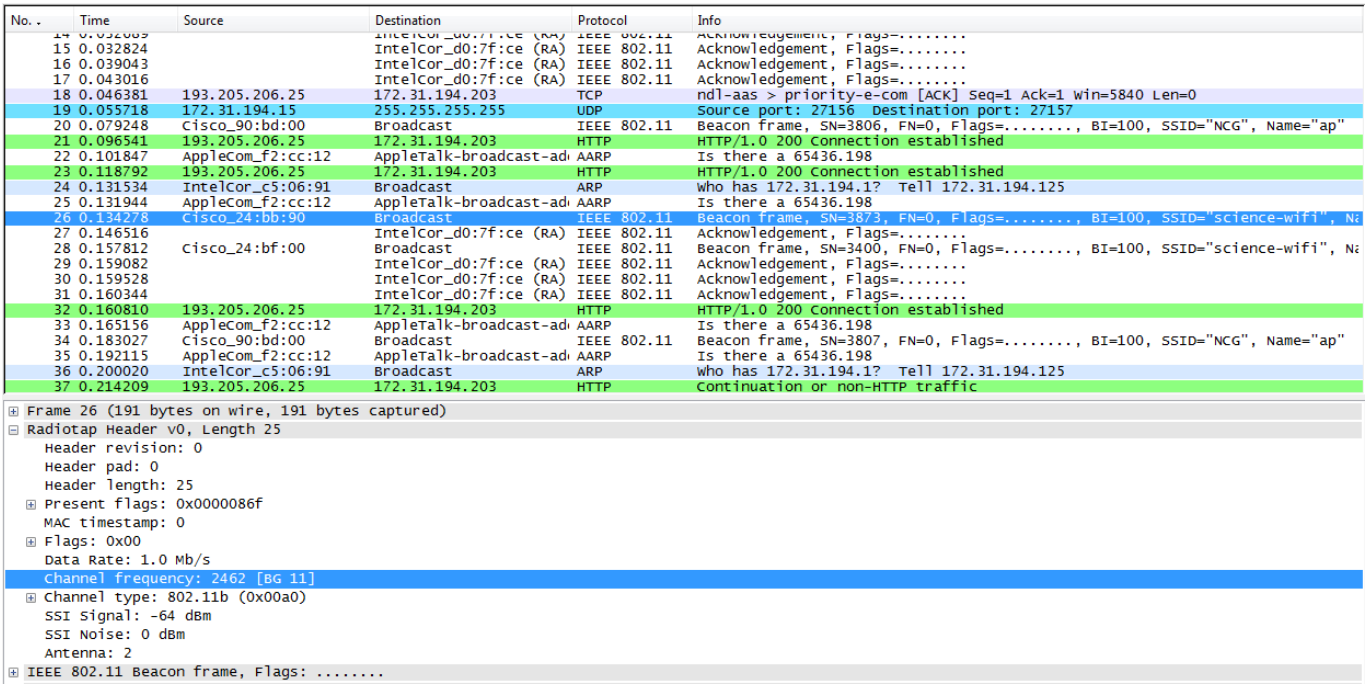


Figura 6

In particolare si è evidenziato un *frame beacon* (pacchetto 26) relativo ad un *AP* della rete di facoltà, operante sul canale numero 11; infatti come si può osservare dalla colonna *Info*, il campo *SSID* del *beacon* contiene il nome "science-wifi". E' utile notare come la velocità di trasmissione del *beacon* (1Mbps) sia la più bassa possibile, in modo tale che tutte le stazioni presenti all'interno del *BSS* possano rilevarlo.

Tutte le misurazioni effettuate di seguito si riferiscono a trasmissioni nelle quali non è stato utilizzato né l'invio dei pacchetti *RTS/CTS*, né la frammentazione, come del resto accadeva nelle precedenti sperimentazioni.
Inoltre, il *throughput* calcolato da Wireshark è stato determinato andando a filtrare i frame intercettati tramite l'indirizzo *MAC* delle stazioni sorgente e di destinazione. Questo filtraggio è stato effettuato per considerare solamente il flusso *UDP* generato dal client Iperf e quindi per eliminare tutti i frame che erano già presenti sul canale e relativi a reti esterne a quelle utilizzate per le prove in questione.

La prima tipologia di test riguarda l'intercettazione del traffico relativo ad una rete con infrastruttura dove è presente una sola stazione (client Iperf).
Tale tipologia di test è stata ulteriormente suddivisa in base alle varie velocità di funzionamento impostate dall'*AP* (test A1, A2, A3). Per motivi di sinteticità si è scelto di mostrare solamente il test A1, relativo alla velocità di 54Mbps, in quanto dai test A2 e A3 non si sono rilevate sostanziali differenze a parte le velocità di trasmissione.

Verrà descritto il livello fisico nel test A1 tenendo presente però che esso non varia per i test successivi, mentre il livello MAC verrà presentato all'interno del test B per una rete infrastrutturata e nel test C per una rete *ad-hoc*.

***4.1 TEST A1*:** 1 *sniffer*, 1 client Iperf con velocità impostata dall'*AP* a 54Mbps

Livello fisico

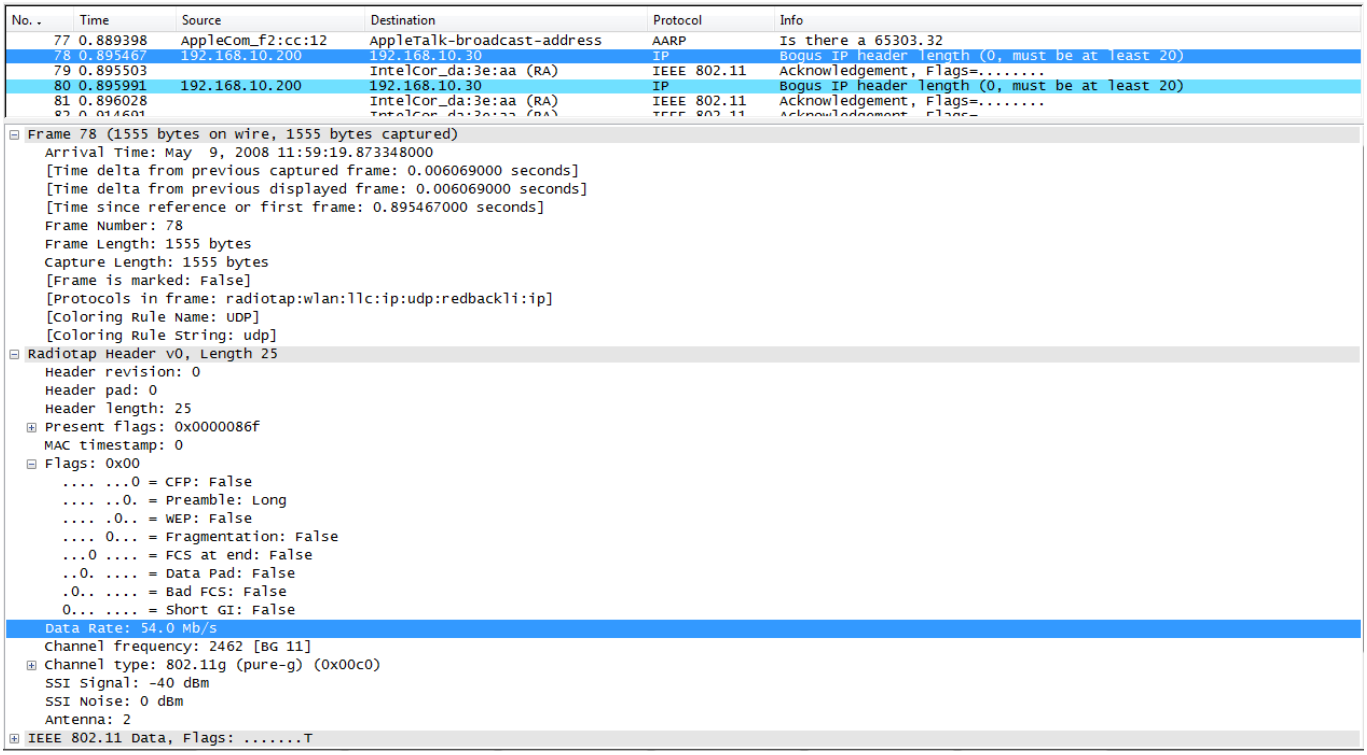


Figura 7

Come si può osservare dalla figura 7, la *Data Rate* nominale alla quale vengono inviati i dati è esattamente 54Mbps. Tale parametro però, come specificato nelle relazioni precedenti, si riferisce solamente al *PSDU* del livello fisico dello standard 802.11, in quanto l'*header* ed il preambolo vengono trasmessi entrambi ad una velocità di 1Mbps se si utilizza la versione lunga, oppure il preambolo a 1Mbps e l'*header* a 2Mbps se si utilizza la versione corta.

L'informazione relativa al tipo di preambolo utilizzato è contenuta all'interno di uno dei *flag* di livello fisico, i quali servono a specificare i parametri della trasmissione, e quindi a garantire una corretta ricezione del frame stesso. In figura 7 si può notare che il preambolo utilizzato nella variante "g" dello standard 802.11 è in versione lunga.

A questo proposito occorre sottolineare che lo *sniffer* in questo esperimento visualizza un tipo di preambolo diverso da quello presentato nella relazione #1 e usato per il calcolo del *throughput* teorico. Ciò potrebbe essere dovuto sostanzialmente a due motivi:

- Si è notato che il software di *sniffing* in alcuni casi mostra informazioni relative ai pacchetti ricevuti incongruenti con lo standard. Ad esempio, in alcune misurazioni relative allo standard 802.11b il preambolo e l'*header* fisico dovrebbero essere in versione corta (e quindi con una lunghezza pari a 15 Byte) ma in realtà l'*overhead* aggiunto dal livello fisico e visualizzato dal software ha una durata di 25 Byte, sebbene il *flag* relativo al tipo di preambolo usato sia corretto e indichi la versione corta. Inoltre occorre notare che la lunghezza dell'*overhead* fisico in versione lunga dovrebbe comunque avere una durata di 24 Byte e non 25.
  Per questo, si potrebbe ipotizzare che lo *sniffer* interpreti in maniera errata alcune informazioni contenute nei pacchetti intercettati, specificando erroneamente una tipologia di trasmissione *DSSS-OFDM*.

- La seconda motivazione al problema di cui sopra potrebbe risiedere proprio nel tipo di modalità di trasmissione impiegata. Come accennato in precedenza, nell'802.11g sono presenti varie tipologie di trasmissione (*DSSS-OFDM*, *ERP-OFDM* e altre) e quindi le ipotesi adottate nei calcoli teorici sull'utilizzo della modalità *ERP-OFDM* potrebbero essere discordi con il tipo di *AP* e rete creata in fase di sperimentazione.

Per i suddetti motivi, viene quindi riportata una tabella riassuntiva con i *throughput* teorici relativi ad una modulazione *DSSS-OFDM* con preambolo sia in versione corta sia in versione lunga.

	802.11g 54Mbps DSSS-OFDM	
	Preambolo corto	**Preambolo lungo**
DIFS	50 μs	50 μs
CW	310 μs	310 μs
H. FISICO	96 μs	192 μs
SYNC	18 μs	18 μs
H. MAC + LLC	5.34 μs	5.34 μs
H. IP + UDP	4.15 μs	4.15 μs
DATI	217.78 μs	217.78 μs
SIFS	10 μs	10 μs
H. FIS. ACK	96 μs	192 μs
ACK	2.07 μs	2.07 μs
TOTALE	905.34 μs	1001.34 μs
THROUGHPUT TEORICO LIV. FISICO	**13.538 Mbps** (10.084 – 20.587)	**12.240 Mbps** (9.346 – 17.728)
THROUGHPUT TEORICO LIV. UDP	**12.990 Mbps** (9.676 – 19.753)	**11.744 Mbps** (8.968 – 17.010)

Tabella 5

Di seguito viene proposto un confronto tra i valori di *Data Rate* misurati dal client Iperf, i valori calcolati sulla sequenza di pacchetti intercettati dallo *sniffer* e i valori teorici. Relativamente allo sniffer si è considerato l'intervallo di tempo di interarrivo tra un frame *UDP* e l'altro, e, grazie ad una particolare funzionalità di Wireshark, si è mediato tale parametro sul numero di frame.

	WIRESHARK						**IPERF**						**TEORIA**		
Intervallo (secondi)	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**DSSS-OFDM preambolo corto**	**DSSS-OFDM preambolo lungo**	**ERP-OFDM**
Throughput misurato (Mbps)	22.8	22.0	22.3	23.0	22.1	**22.4**	23.4	22.8	23.0	23.6	23.2	**23.2**	**13.0**	**11.7**	**26.3**

Tabella 6

Si sottolinea che, al fine di effettuare un confronto accurato, l'istante iniziale di analisi per Wireshark (l'istante "0") è stato fatto coincidere con l'istante relativo alla rilevazione (da parte di Wireshark stesso) del primo pacchetto generato da Iperf. Gli intervalli di tempo sono pari a 2 secondi e, come ultimo campo, viene riportata una media del *throughput* sull'intero test (durata 10 secondi).

Dalla tabella 6 emerge che il *throughput* relativo al traffico catturato da Wireshark è inferiore del corrispondente traffico misurato da Iperf. Tale risultato sperimentale è compatibile con le ipotesi evidenziate in precedenza relative alla perdita, da parte del software di cattura, di diversi pacchetti che in realtà giungono correttamente a destinazione.
Inoltre, confrontando il *throughput* teorico di una modulazione *DSSS-OFDM* con il *throughput* medio misurato, si nota come i valori siano nettamente inferiori anche nel caso limite di finestra di contesa nulla, e quindi incongruenti con l'ipotesi di tale tipologia di trasmissione.
Per questo motivo è ragionevole supporre che il preambolo indicato da Wireshark sia errato, facendo propendere per la prima delle due ipotesi descritte precedentemente, ovvero l'utilizzo di una modulazione *ERP-OFDM*. Tuttavia, anche in questo ultimo caso si riscontra una discrepanza tra le prestazioni reali e quelle teoriche, attribuibile al fatto che il canale wireless comporta delle perdite e che il valore teorico rappresenta una stima del valor medio del *throughput* (*CW* = 15.5 *time slot*).
Per i confronti da qui in poi si è quindi scelto di comparare i dati misurati dallo *sniffer* con i valori ottenuti nel caso *ERP-OFDM*.

Inoltre, il software Wireshark mette a disposizione molte altre informazioni, quali ad esempio, l'indicazione dell'uso di una

crittografia dei dati basata su algoritmo *WEP* o *WPA*, della frammentazione, la visualizzazione della potenza del segnale ricevuto e della potenza del rumore (espressa in dBm).
In aggiunta si possono esplorare molti parametri relativi al mezzo trasmissivo, come ad esempio la frequenza e quindi il canale utilizzato, che in questo caso sono rispettivamente 2462MHz e 11.
Sono riportate anche tutte le specifiche relative alla modulazione.

Il successivo test riguarda l'aggiunta, rispetto al caso precedente, di un ulteriore client *iperf*.

***4.2 TEST B*:** 1 *sniffer*, 2 client Iperf con velocità impostata dall'*AP* a 54Mbps

Livello MAC - rete infrastrutturata

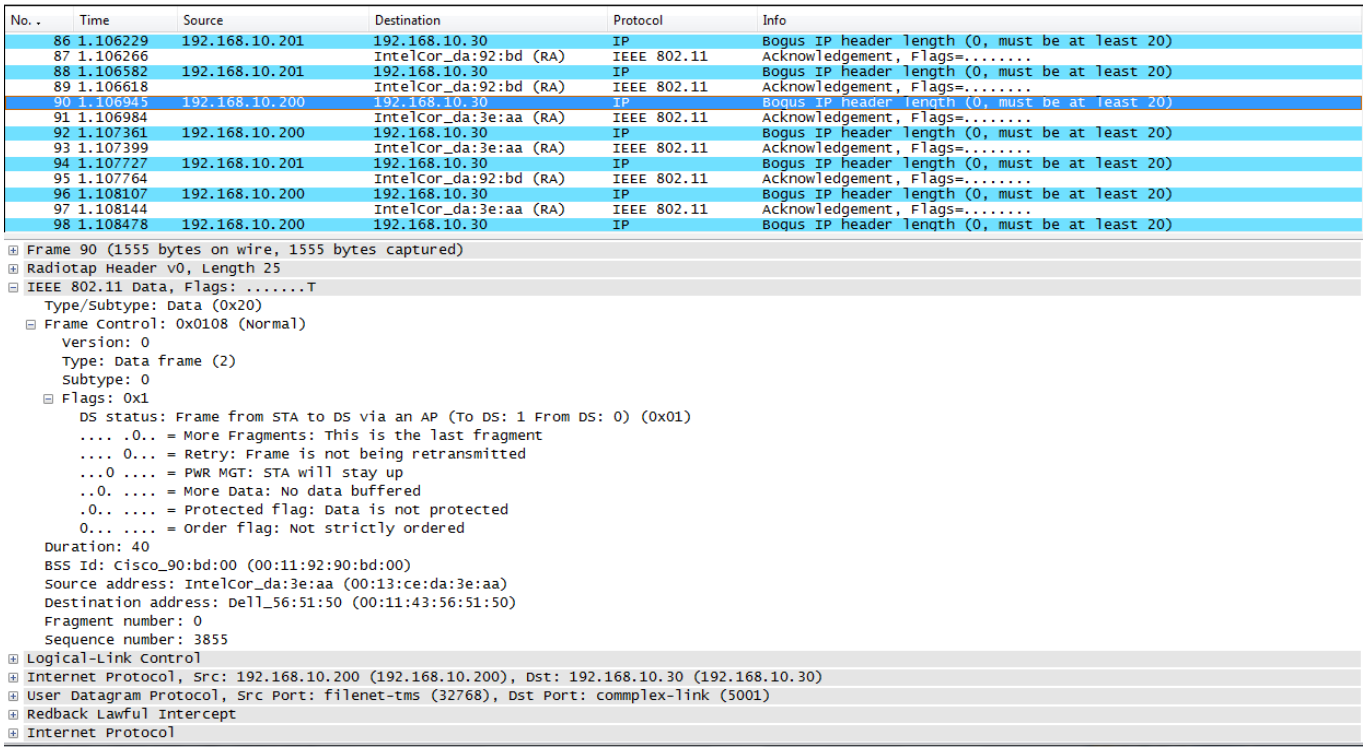


Figura 8

Come si può osservare dalla figura 8, in questo test sono presenti 2 client collegati allo stesso *AP*, che generano 2 *stream* separati di dati *UDP*.

In questo esperimento si concentra l'attenzione sul contenuto dell'*header MAC,* in quanto l'*header* fisico non presenta sostanziali differenze rispetto al test precedente.
L'aggiunta di una stazione rispetto al caso precedente è stata effettuata con il mero scopo di mostrare gli effetti della condivisione del mezzo sul *throughput* aggregato e sui singoli flussi di dati, e per mostrare quanto le performance di una stazione dipendano strettamente dal comportamento delle altre.

La prima informazione che Wireshark mette a disposizione è il tipo/sottotipo di frame che si sta andando a visualizzare; nella prova di figura 8 vengono evidenziate le caratteristiche di un frame di dato inviato da uno dei due client.
Il campo sopra menzionato, oltre a contenere il tipo e sottotipo di frame, include anche la versione del protocollo e una serie di *flag*. Questi *flag*, come indicato nel capitolo 3, si suddividono in:

- *flag DS status* (*ToDS* e *FromDS*): 1 e 0, il frame è generato dal client Iperf e diretto all'*AP*;
- *flag More Fragment*: 0, nessuna frammentazione;
- *flag Retry*: 0, nessuna ritrasmissione;
- *flag PWR MGT*: 0, nessuna modalità di risparmio energetico impiegata (*power safe mode*);
- *flag More Data*: 0, nessun dato bufferizzato;
- *flag Protected Data*: 0, utilizzo di un algoritmo di crittazione WPA;
- *flag order*: 0, non viene impiegata la *Strictly-Ordered service class*.

In coda ai *flag* sopra descritti si può trovare il campo *Duration*, che in questo caso, essendo incapsulato in un frame di tipo *data*, contiene il tempo previsto per la trasmissione del frame stesso.

Successivamente vengono riportati i campi d'indirizzo:

- *BSS Id*: indirizzo *MAC* dell'interfaccia wireless dell'*AP* (campo *Address1* figura 5);
- *Source address*: indirizzo *MAC* della sorgente del frame ovvero delle stazioni sulle quali è in esecuzione il client Iperf (campo *Address2* figura 5);
- *Destination address*: indirizzo *MAC* della destinazione ovvero il server Iperf connesso all'*AP* (campo *Address3* figura 5);
- *Address4*: non usato, si utilizza per comunicazioni tra *AP* tramite *DS*.

Le ultime due informazioni riguardano il *fragment number* e il *sequence number*. Non avendo utilizzato la frammentazione, il *fragment number* è sempre impostato a 0, mentre il numero di frame indica il numero progressivo di trasmissione dello stesso.
Anche in questo caso si può effettuare un confronto tra i valori di *throughput* stimati dai due software e quelli teorici.

	WIRESHARK						**IPERF**						**TEORIA**
Intervallo (secondi)	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**ERP-OFDM**
Throughput misurato (Mbps) Stazione 1	11.0	12.4	12.0	12.4	11.7	**11.9**	11.2	12.0	11.3	11.8	11.8	**11.6**	-
Throughput misurato (Mbps) Stazione 2	12.3	12.7	13.4	12.8	13.2	**12.9**	12.7	12.5	12.5	12.5	12.5	**12.5**	-
Throughput aggregato (Mbps)	23.3	25.1	25.4	25.2	24.9	**24.8**	23.9	24.5	23.8	24.3	24.3	**24.1**	**26.3**

Tabella 7

Per le medesime considerazioni fatte nel test precedente, il valor medio teorico risulta essere più elevato rispetto ai *throughput* misurati dai due software.
Occorre sottolineare che in questo caso i risultati di *throughput* calcolati mediante Wireshark sono addirittura superiori a quelli misurati da Iperf. Un'analisi più approfondita ha mostrato che in realtà il traffico catturato da Wireshark spesso non rispecchia esattamente ciò che viene trasmesso sul canale. Accade infatti che, probabilmente a causa dell'elevato numero di pacchetti, una serie di essi venga interpretata in modo errato, come evidenziato in figura 9.

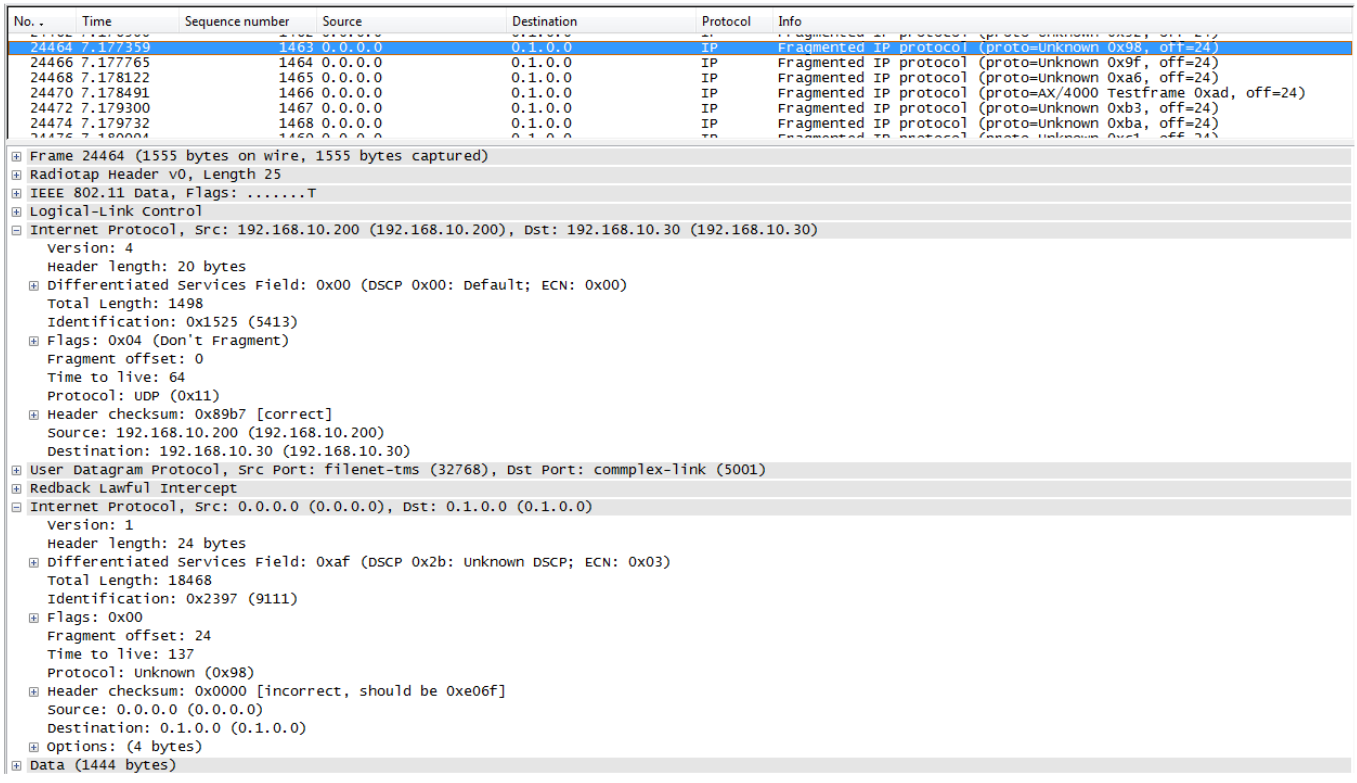


Figura 9

Dalla figura 9 si nota infatti che una serie consecutiva di pacchetti, evidenziati come "*Fragmented IP Protocol*", presenta campi anomali, come ad esempio due *header IP*.

Ciò ovviamente non rispecchia la trasmissione che avviene sul canale, ma è un'errata interpretazione di essa. Per tale motivo non è stato possibile correggere il calcolo del *throughput* di Wireshark, in quanto una serie di pacchetti, che in realtà giungono correttamente a destinazione, e che vengono misurati da Iperf, viene interpretata in modo erroneo. E' interessante notare come questo fenomeno si presenti sia sui laptop descritti in tabella 1 che sui laptop descritti in tabella 2, e con diversi sistemi operativi (sia con GNU/Linux Ubuntu 7.10 che con GNU/Linux Backtrack 4.0 Beta). Per questo i valori forniti dallo *sniffer* sono parzialmente inesatti e potrebbero non rispecchiare il vero *throughput* delle stazioni.

Di seguito viene presentato l'andamento sul canale del traffico generato da ogni singola stazione e catturato da Wireshark. Occorre notare che a causa di particolari impostazioni di visualizzazione dello *sniffer,* la quantità riportata in ordinata è espressa in bit/0.1sec e non bit/sec come normalmente avviene.

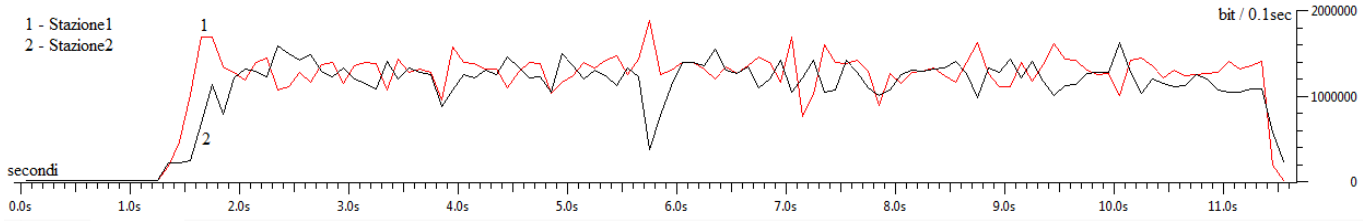


Figura 10

Si evidenzia come, in corrispondenza dei picchi di traffico generato da una stazione, corrispondano in generale degli abbassamenti del *throughput* generato dall'altra, e quindi come il *throughput* aggregato si mantenga all'incirca costante. Ciò avviene perché, come sottolineato nelle precedenti relazioni, accade spesso che ogni singola stazione che accede al mezzo subisca un calo di prestazioni, dovuto, ad esempio, a drastiche diminuzioni della qualità del canale. In corrispondenza di questi abbassamenti, l'altra stazione che accede al mezzo subisce degli incrementi di *throughput*, in quanto il canale, liberato momentaneamente dalla mole di traffico della prima stazione, risulta libero per la trasmissione della seconda, che incrementa quindi il suo *throughput* (v. figura 10 secondo 5.7). In ogni caso, avendo il canale una capacità limitata, la somma di tutti i flussi si mantiene all'incirca costante, a meno di peggioramenti contemporanei sui link di tutte le stazioni che determinano un calo del *throughput* aggregato (v. figura 10 secondo 3.8).

4.3 TEST C: 1 *sniffer*, rete *ad-hoc* tra 2 stazioni (1 server Iperf e 1 client Iperf) con velocità impostata a 54 Mbps

Livello MAC - rete ad-hoc

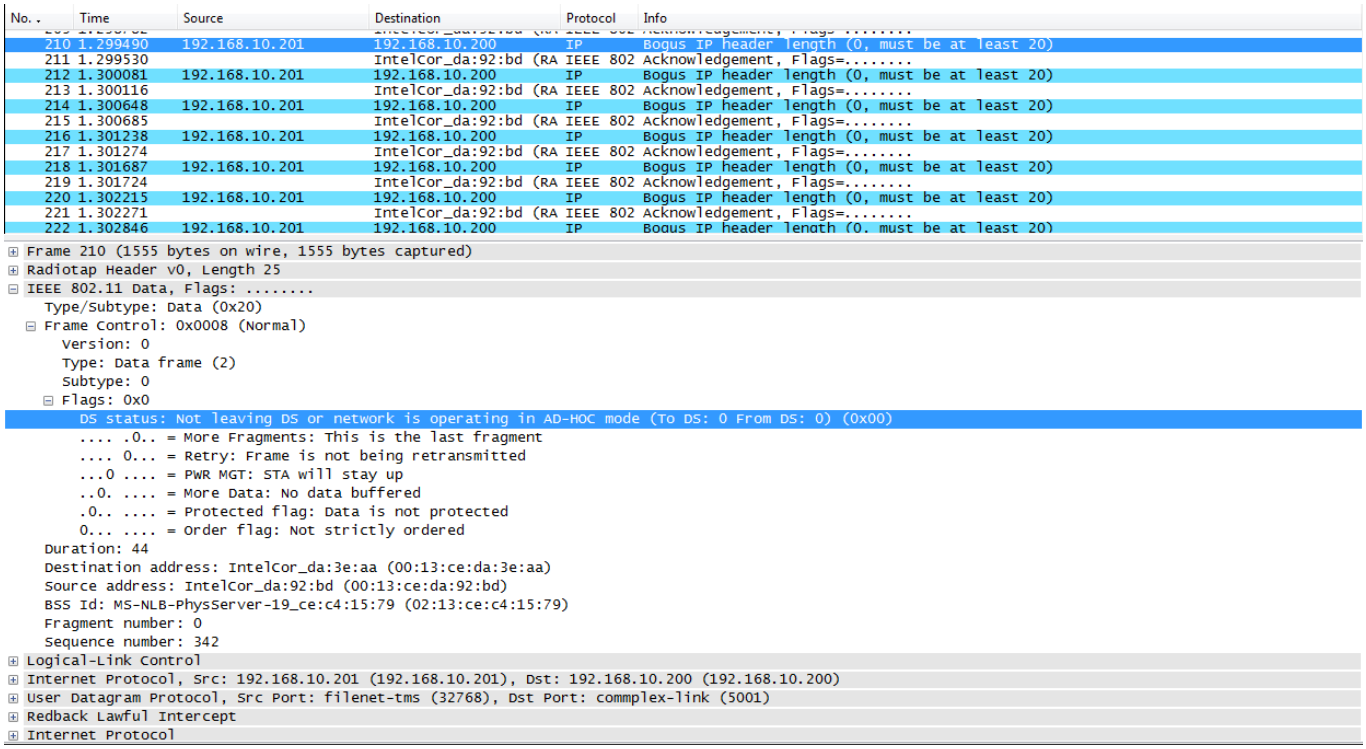


Figura 11

Il test C si riferisce ad una configurazione in cui si è creata un rete *ad-hoc* tra il client ed il server Iperf. L'*overhead* aggiunto dal livello fisico è il medesimo dei test precedenti.

Le variazioni rispetto ai casi di cui sopra si possono invece notare a livello *MAC*, in quanto i *flag* relativi allo stato del *DS* indicano che la rete è di tipo *ad-hoc*, e quindi i campi *ToDS* e *FromDS* contenuti all'interno del *Frame Control* sono impostati entrambi a 0, come specificato in tabella 4.

Oltre ai *flag* relativi al *DS*, anche i campi d'indirizzo cambiano ordine e significato:

- *Destination address*: indirizzo *MAC* della destinazione, ossia l'indirizzo della stazione su cui è attivo il server Iperf (campo *Address1* figura 5);
- *Source address*: indirizzo *MAC* della sorgente, ovvero l'indirizzo della stazione sulla quale è in esecuzione il client Iperf (campo *Address2* figura 5);
- *BSS Id*: identificativo dell'*IBSS,* è un indirizzo *MAC* generato in maniera randomica dalla stazione che ha creato l'*IBSS* stesso.

Di seguito viene riportata la struttura di un frame *ACK* relativo alla comunicazione tra le due stazioni all'interno della rete *ad-hoc*. Si ricorda che in ogni caso i frame di tipo *Acknowledgment* sono presenti anche in una rete infrastrutturata e, secondo lo standard, non presentano differenze tra i due casi.

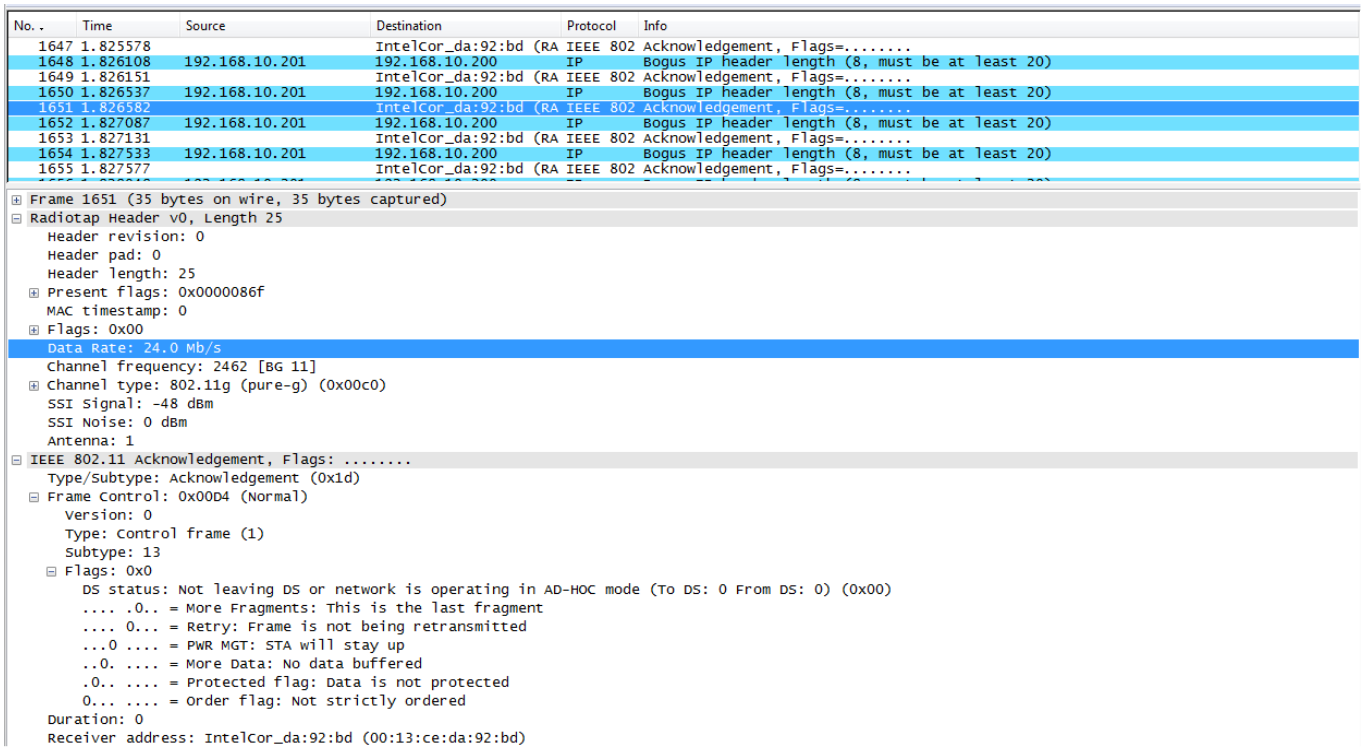


Figura 12

La struttura dell'*header* e del preambolo fisico è identica a quella dei frame di dato, ma si può notare come la lunghezza dell'intero *ACK* sia invece molto minore. Infatti, per non impattare troppo nella comunicazione tra le stazioni, il protocollo 802.11 prevede degli *ACK* con *header MAC* molto corti, che contengono solamente i campi *Frame Control* (2 Byte), *Duration* (2 Byte), *RA* (6 Byte) e *CRC* (4 Byte). Quindi un frame di questo tipo dovrebbe avere un *overhead* aggiunto dal livello *MAC* pari a 14 Byte, ai quali si vanno ad aggiungere i Byte del livello fisico.

Wireshark mostra una durata di *header MAC* pari a 10 Byte (35 Byte totali meno 25 Byte di *header* fisico) pertanto si suppone che non vengano considerati i 4 Byte di *CRC* di livello *MAC*. Questa problematica non è legata al particolare software di *sniffing*, ma direttamente al sistema operativo e ai *driver* dell'interfaccia di rete wireless, che non permettono la visualizzazione del *CRC* di livello *MAC* in quanto esso è un parametro utilizzato direttamente dalla *NIC* e successivamente scartato. Nonostante ciò, l'*ACK* ha una dimensione decisamente inferiore rispetto ad un frame di dato.

Si può osservare che i *flag* relativi alla tipologia di frame indicano che l'*ACK* è un frame di controllo di sottotipo *Acknowledgment*, mentre tutti gli altri *flag* sono settati a 0, come da specifiche nel caso di un pacchetto di controllo.

Il campo *Duration* dell'*ACK* è posto pari a 0, perché il *flag More Fragment* del frame di dati immediatamente precedente è impostato a 0.

Infine si può evidenziare come venga utilizzato solamente un campo di indirizzo, il quale contiene l'indirizzo *RA* (*Receiver Address*), che a sua volta è copiato dall'*Address2* del frame di dati precedente.

Come mostrato nella figura 12, bisogna porre attenzione alla velocità di trasmissione dei 14 Byte di *ACK*, che, contrariamente a quanto considerato nei calcoli teorici della relazione #1 (54Mbps) è di 24Mbps. Tale aspetto si è notato in tutte le prove effettuate su una rete *ad-hoc* alla massima velocità permessa dal protocollo 802.11g, mentre non accade per le reti infrastrutturate. Tenendo conto di questo aspetto, sono stati quindi ricalcolati (v. tabella 8) i valori medi teorici per i tre differenti tipi di modulazione considerati in precedenza.
Alla luce dei nuovi calcoli, si nota che la variazione di *throughput* teorico è irrilevante e questo non spiega la bassa *Data Rate* ottenuta nella rete *ad-hoc*.
Si potrebbe quindi supporre che, data la vicinanza tra i valori misurati ed il *throughput* medio teorico per una modulazione *DSSS-OFDM* all'interno della rete *ad-hoc* considerata, le varie *NIC* utilizzino tale modulazione anziché l'*ERP-OFDM* precedentemente supposta per la rete con infrastruttura.

	WIRESHARK						**IPERF**						**TEORIA**		
Intervallo (secondi)	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**DSSS-OFDM preambolo corto**	**DSSS-OFDM preambolo lungo**	**ERP-OFDM**
Throughput misurato (Mbps)	12.4	15.5	16.7	14.6	18.0	**15.4**	12.1	14.2	16.0	12.8	16.6	**14.3**	**13.0**	**11.7**	**26.1**

Tabella 8

Come per il precedente test, anche in questo caso i *throughput* calcolati da Wireshark risultano maggiori di quelli misurati da Iperf, e anche qui si è verificata la presenza di una serie di pacchetti catturati in maniera errata da Wireshark.

Come evidenziato nel capitolo 3, la ritrasmissione di un frame, dovuta ad esempio ad un ostacolo interposto tra trasmettitore e ricevitore, oppure a causa di sfavorevoli condizioni del canale di comunicazione, è caratterizzata dalla presenza all'interno dell'*header MAC* del *flag Retry* settato a 1.
Mediante l'utilizzo di questo *flag* e del campo *Sequence number* (che, come evidenziato in figura 13, è invariato tra il frame originale e quello ritrasmesso) la stazione ricevente è in grado di identificare se un pacchetto è duplicato e, in tal caso, è in grado di scartarlo.

A puro scopo esplicativo, viene mostrata la struttura dell'*header MAC* di un frame ritrasmesso:

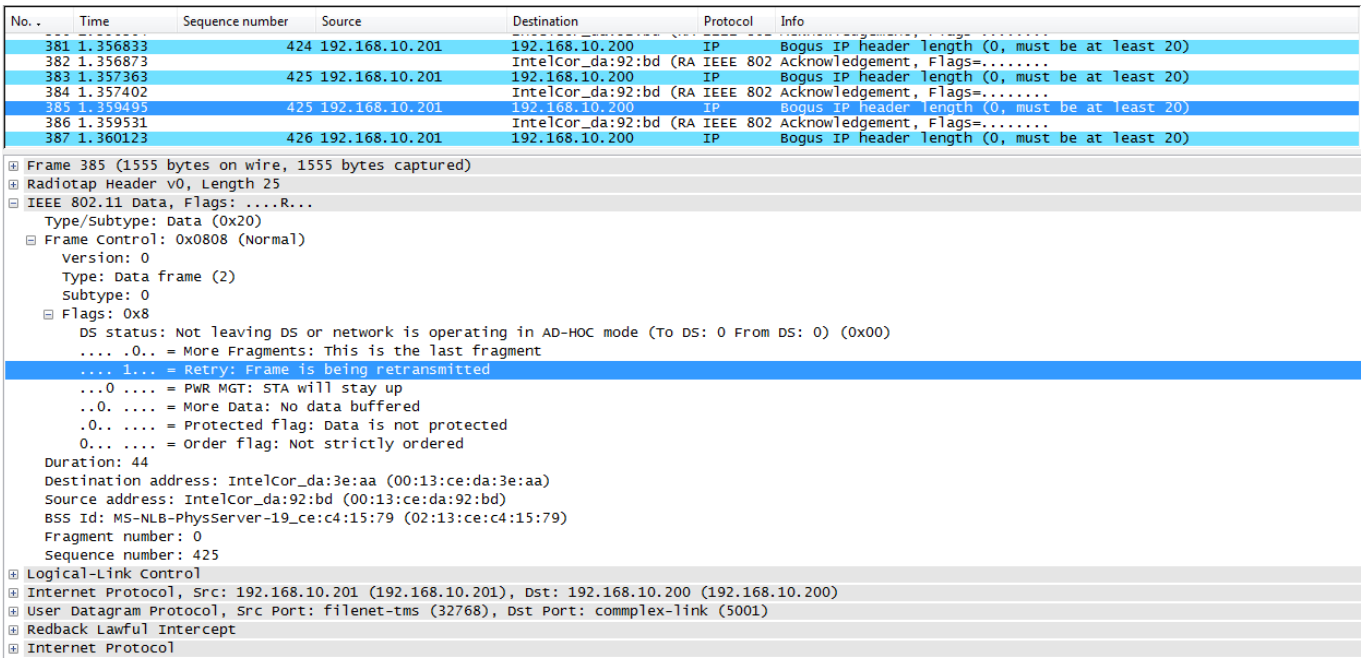
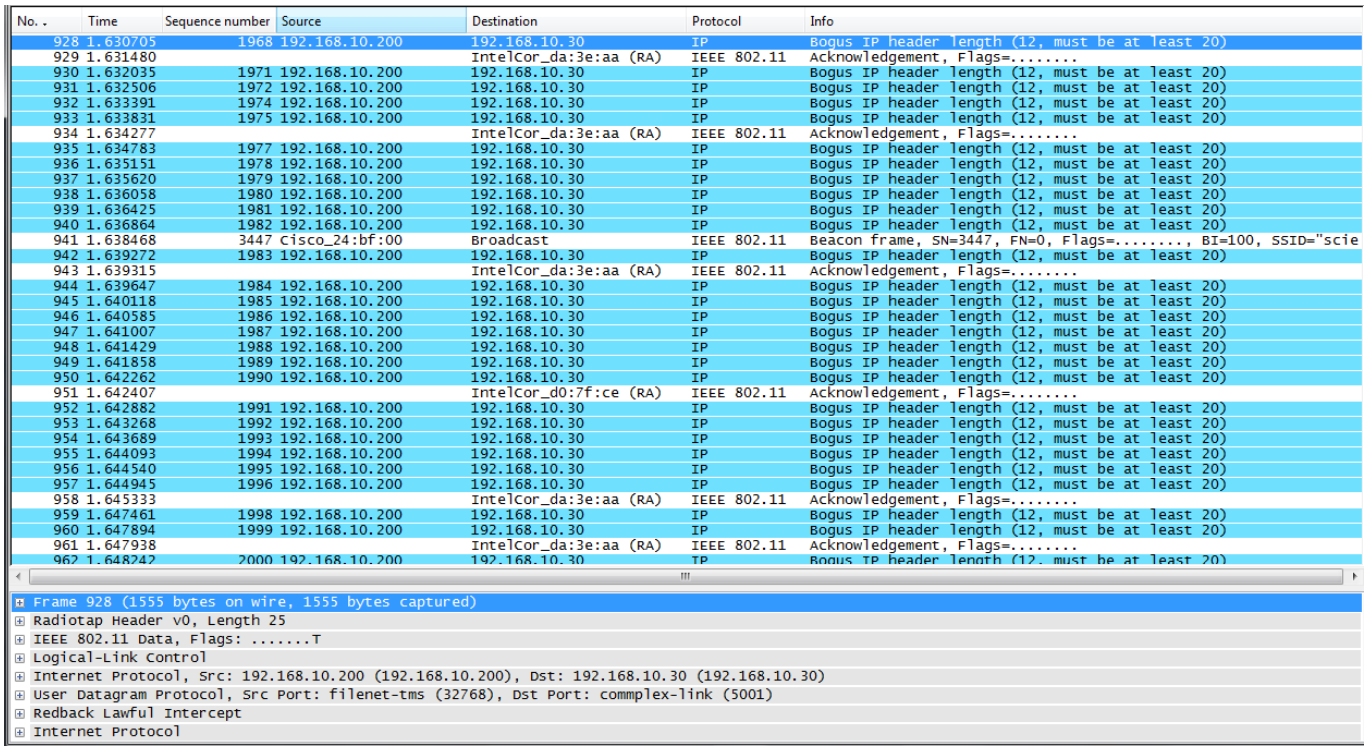


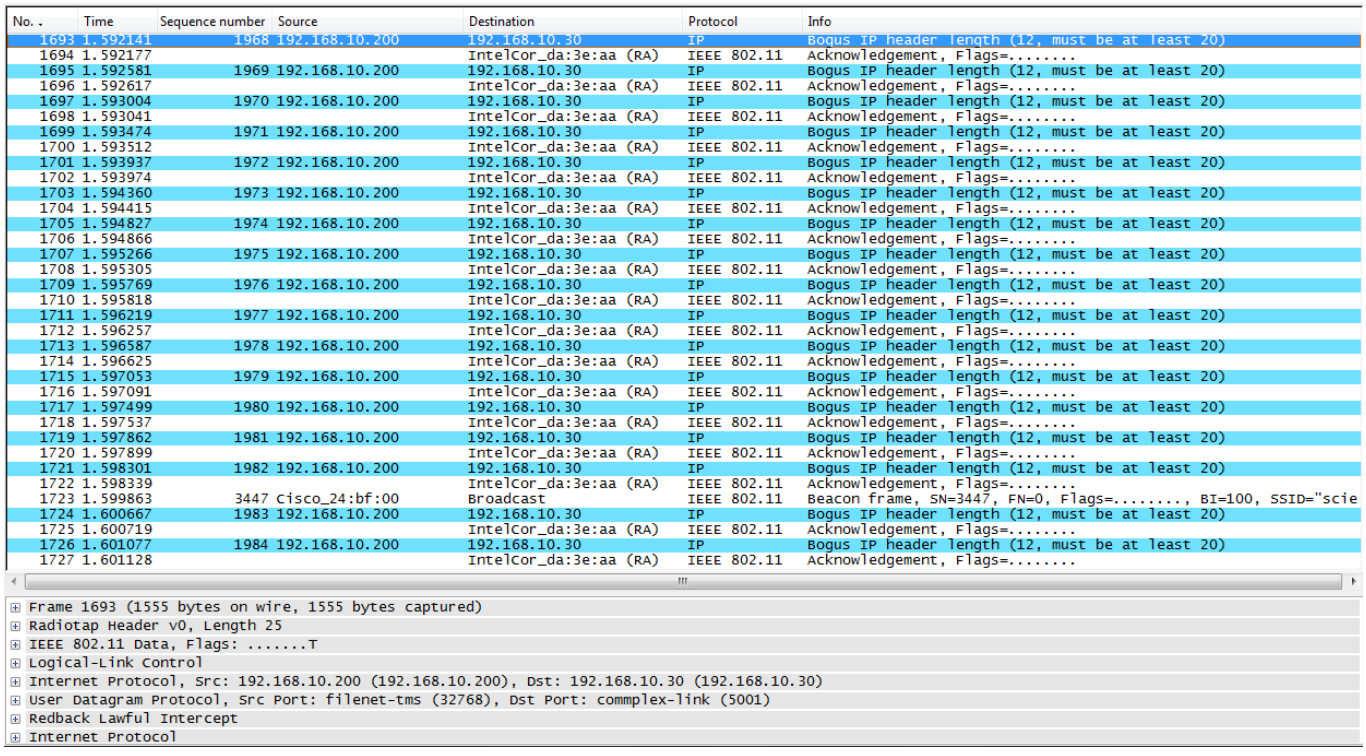
Figura 13

***4.3 TEST D*:** 1 singolo client Iperf, 2 *sniffer* con velocità impostata dall'*AP* a 54 Mbps

Per concludere, viene presentato il test relativo alla presenza di due diversi *sniffer* che intercettano il traffico di un singolo client Iperf all'interno di una rete con infrastruttura. Tale test è stata effettuato per analizzare il problema della perdita dei pacchetti cui è soggetto Wireshark.



(a) 1° sniffer



(b) 2° sniffer

Figura 14

Quanto ipotizzato nei capitoli precedenti viene effettivamente riscontrato in figura 14, dove si può notare come il software di *sniffing* fornisca informazioni differenti a seconda della "macchina" sulla quale è in esecuzione. Più precisamente, può capitare che nella visualizzazione Wireshark tralasci alcuni pacchetti andando così a compromettere la corretta determinazione delle medie delle velocità di trasferimento e quindi delle quantità di dati inviati.

La figura 14a si riferisce ad un'intercettazione eseguita con un laptop le cui caratteristiche sono riportate in tabella 1, con la variante d'utilizzo di un Sistema operativo GNU/Linux Backtrack 4.0 Beta, mentre in figura 14b è presentata una fase dello *sniffing* da parte di una stazione con caratteristiche identiche a quelle descritte in tabella 2.

Accanto ad ogni frame è indicato il *sequence number*, che non deve essere confuso con il valore della colonna "No." (*Number*), nella quale è contenuto il numero progressivo di cattura del pacchetto da parte dello specifico *sniffer*.

Qualora si utilizzi la frammentazione, grazie alla coppia (*sequence number*, *fragment number*) la stazione ricevente riesce ad identificare tutti i frammenti relativi allo stesso pacchetto, in quanto essi avranno lo stesso *sequence number* e diversi *fragment number*. Mediante la visualizzazione a video del *sequence number* inoltre è più facile per l'operatore individuare quali pacchetti siano stati tralasciati.

Nelle figure precedenti si può osservare che il frame avente come *sequence number* il valore 1968 e il rispettivo *ACK* di conferma (frame iniziali della lista) sono intercettati dai software di *sniffing* di entrambe le stazioni; in seguito però lo *sniffer* di figura 14a perde i successivi due frame di dati e i relativi *ACK*, al contrario della stazione sulla quale è in esecuzione il secondo *sniffer*.

In aggiunta, si osserva che il primo *sniffer* in questa fase dell'intercettazione tende a non rilevare gli *ACK* di molti frame successivi.

Di seguito vengono riassunte le medie calcolate sull'intero flusso *UDP* rilevato dai due *sniffer* e comparate con i valori forniti dal client Iperf.

	WIRESHARK						**IPERF**					
Intervallo (secondi)	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**	**0-2**	**2-4**	**4-6**	**6-8**	**8-10**	**0-10 (AVG)**
Throughput misurato (Mbps) da Sniffer 1	20.6	20.6	22.7	14.2	22.7	**20.1**	24.1	22.6	21.5	23.0	23.0	**22.8**
Throughput misurato (Mbps) da Sniffer 2	25.0	23.6	23.2	23.8	23.0	**23.7**						

Tabella 9

Si notano delle differenze nella misurazione del *throughput* operata dai due diversi *sniffer*, in quanto la quantità di pacchetti catturati è alquanto differente. Per questo motivo le misure fornite dagli *sniffer* devono essere considerate come misure indicative sulle quali è sempre presente un certo margine di inaffidabilità.

Effettuando una semplice analisi dei due listati riportati in figura 14 si può asserire che a causa delle perdite di molti frame e i relativi *ACK* del primo *sniffer*, molto probabilmente il valore di *throughput* più veritiero è quello fornito dal secondo, che infatti si avvicina maggiormente al valore indicato dal client Iperf.

Permane comunque il problema evidenziato precedentemente relativo ai valori di Wireshark, più elevati di quelli misurati da Iperf.

In figura 15 è riportato l'andamento temporale del traffico rilevato dallo *sniffer* 1 e dallo *sniffer* 2. Si nota come, per alcune finestre temporali (ad esempio tra 5sec e 7sec) gli andamenti siano pressoché simili, mentre in altri intervalli (ad esempio tra 8sec e 9sec) le rilevazioni siano piuttosto diverse. Questo perché lo *sniffer* 1 non ha catturato molti dei pacchetti trasmessi dal client Iperf, che invece sono stati intercettati dallo *sniffer* 2.

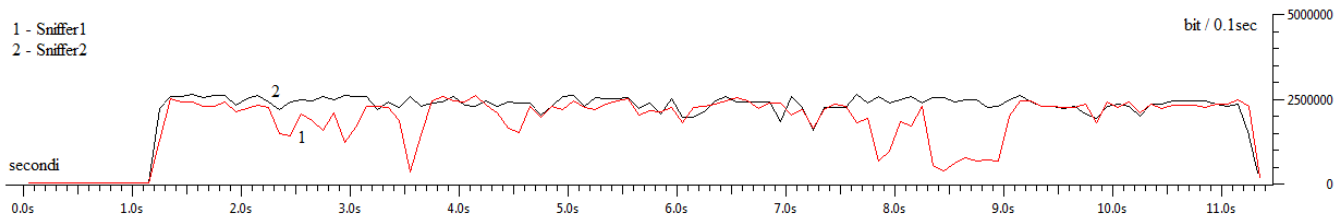


Figura 15

5. CONCLUSIONI

Mediante questa esperienza si è potuto verificare quali siano le procedure di comunicazione previste dal protocollo 802.11 relativo alle WLAN.

L'analisi ha permesso inoltre di comprendere la struttura dei due livelli più bassi dello *stack* protocollare (livello *MAC* e livello fisico) e delle varie tipologie di frame che vengono inviati sul canale di trasmissione. Essi sono stati esposti per mezzo dell'analisi dettagliata delle informazioni che tali layer aggiungono ai veri e propri dati, ad esclusione di alcuni campi del *MAC* layer, che non sono visualizzabili dal software impiegato nei test (ad esempio il campo relativo al *CRC*).

Inoltre questi test hanno evidenziato le differenze tra i frame scambiati dalle stazioni all'interno di una rete con infrastruttura ed una rete *ad-hoc*.

L'analisi del traffico generato all'interno della rete infrastrutturata ha mostrato alcune incongruenze con quanto atteso: ad esempio per la massima velocità permessa dal protocollo 802.11g (54Mbps) si è potuto notare che la trasmissione è stata interpretata dallo *sniffer* come *DSSS-OFDM*, quando invece tutti gli altri dati fanno supporre l'utilizzo di una modulazione *ERP-OFDM*.
Un'ulteriore incoerenza verificatasi nella rete *ad-hoc* alla velocità di 54Mbps è quella corrispondente alla *Data Rate* di trasmissione degli *ACK*, che contrariamente alle aspettative è avvenuta a 24Mbps.

Oltre a ciò, si è mostrato che i valori forniti da Wireshark sono affetti da una certa variabilità, che ha portato alla determinazione di medie relative al *throughput* che non rispecchiano fedelmente la realtà e che a volte sono piuttosto diverse da quelle misurate dal client Iperf. Questa complicazione è dovuta principalmente al tool di *sniffing* utilizzato, in quanto, come si è potuto dimostrare nell'ultimo test, è dipendente dal tipo di interfaccia di rete e sistema operativo sui quali viene eseguito. A causa di ciò, si può verificare la situazione nella quale il software di cattura tralascia alcuni pacchetti che in realtà transitano sul canale, oppure visualizza informazioni erronee non corrispondenti ai dati veri e propri.

Infine, come nelle esperienze precedenti, si è verificata una leggera discrepanza tra i valori medi teorici e il *throughput* misurato, imputabile alle perdite, alle non idealità del canale di trasmissione e alla reale variazione della finestra di contesa rispetto al valore medio di 15.5 *time slot* considerato nei calcoli.

La conclusione che si può trarre è che un software di cattura come Wireshark è indubbiamente utile nell'analisi e nella diagnostica di problematiche legate alla realizzazione e manutenzione di reti wireless. Tuttavia, le informazioni da esso fornite non possono e non devono essere ritenute sufficienti ai fini della completa ed effettiva comprensione del funzionamento della rete.