Questa è la Dedica*:*
ognuno può scrivere quello che vuole,
anche nulla . . .

Introduzione

Questa è l'introduzione.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Cos'è un kernel

Il kernel rappresenta il nucleo di un sistema operativo e racchiude tutte le funzioni principali del sistema stesso come gestione della memoria, gestione delle risorse, lo scheduling e il file system. Le applicazioni in esecuzione nel sistema possono richiedere particolari servizi al kernel tramite chiamate di sistema (system call) senza accedere direttamente alle risorse fisiche. L'accesso diretto all'hardware può risultare anche molto complesso pertanto il kernel implementa una o più astrazioni dell'hardware, il cosiddetto Hardware Abstraction Layer. Queste astrazioni servono a nascondere la complessità e a fornire un'interfaccia pulita ed omogenea dell'hardware sottostante.
I kernel si possono classificare in quattro categorie:

- kernel monolitici, un unico aggregato di procedure di gestione mutuamente coordinate e astrazioni hardware

- micro kernel, semplici astrazioni dell'hardware gestite e coordinate da un kernel minimale, basate su un paradigma client/server, e primitive di message passing

- kernel ibridi simili a micro kernel con la sola differenza di eseguire alcune componenti del sistema in kernel space per questione di efficienza

- exo kernel non forniscono alcuna astrazione dell'hardware sottostante ma soltanto una collezione di librerie per mettere in contatto applicazioni con le risorse fisiche

1.1 Il kernel Linux

Nell'aprile del 1991 Linus Torvalds, uno studente finlandese di informatica, comincia a sviluppare un semplice sistema operativo chiamato Linux. L'architettura del kernel sviluppato da Torvalds è di tipo monolitico a discapito di una struttura più moderna e flessibile come il micro kernel. Sebbene oggi il kernel possa essere compilato in modo da ottenere un'immagine binaria ridotta al minimo e i driver caricabili da moduli esterni, l'architettura originaria è chiaramente visibile: tutti i driver infatti devono avere una parte eseguita in kernel mode, anche quelli per cui ciò non sarebbe affatto necessario (ad esempio i driver dei file system). Attualmente il kernel Linux è distribuito con licenza GNU General Public License e in continua evoluzione grazie a una vastissima comunità di sviluppatori da ogni parte del mondo che contribuiscono al suo sviluppo. Il kernel Linux trova larghissima diffusione: infatti grazie alla sua flessibilità viene utilizzato dai personal computer ai grandi centri di calcolo, dai nuovi sistemi embedded agli smartphone. Il sistema mobile più diffuso al mondo Android si basa su una versione lightweight del kernel Linux.

Capitolo 2

Lo stack di rete

In questo capitolo si vuole presentare la suite di protocolli utilizzata nel progetto di tesi. Verrà descritto brevemente il modello ISO/OSI per poi soffermarsi con particolare attenzione sulla tecnologia Wi-Fi e sui protocolli toccati dal progetto di tesi. ISO/OSI è uno standard che definisce un modello composto su più livelli: ogni *layer* si occupa di uno specifico aspetto delle comunicazioni di rete fornendo delle funzionalità a livello superiore e sfruttando le astrazioni fornite dal livello immediatamente inferiore. In questo modo è possibile ridurre la complessità non banale delle comunicazioni di rete.

2.1 Il modello ISO/OSI

Il modello OSI (Open System Interconnection) è uno standard per le reti di calcolatori stabilito da ISO (International Standard Organization) che definisce l'architettura logica di rete come una struttura a strati composta da una pila di protocolli di comunicazione di rete suddivisa in 7 livelli, i quali insieme espletano in maniera logico-gerarchica tutte le funzionalità della rete. Ciascun layer racchiude in sé a livello logico uno o più aspetti fra loro correlati della comunicazione fra due nodi di una rete. I layers vanno dal livello fisico (quello del mezzo fisico, ossia del cavo o delle onde radio) fino al livello

delle applicazioni, attraverso cui si realizza la comunicazione di alto livello. Come già accennato ciascun livello fornisce servizi e funzionalità al livello superiore utilizzando le astrazioni fornite dal livello inferiore. Ma vediamo brevemente le funzioni di ciascun livello all'interno dello stack ISO/OSI.

Physical Layer Si occupa di trasmettere dati non strutturati attraverso un mezzo fisico e di controllare la rete, gli hardware che la compongono e i dispositivi che permettono la connessione. In questo livello vengono decisi diversi aspetti legati al mezzo fisico come ad esempio le tensioni scelte per rappresentare i valori logici dei bit trasmessi, la durata in microsecondi del segnale che identifica un bit, la modulazione e la codifica utilizzata e l'eventuale trasmissione simultanea in due direzioni (duplex).

Datalink Layer Questo livello si occupa in primis di formare i dati da inviare attraverso il livello fisico, incapsulando il pacchetto proveniente dallo strato superiore in un nuovo pacchetto provvisto di un nuovo header (intestazione) e tail (coda). Questa frammentazione dei dati in specifici pacchetti è detta framing e i singoli pacchetti sono chiamati *frame*. Il livello Datalink effettua inoltre un controllo degli errori e delle perdite di segnale in modo tale da far apparire, al livello superiore, il mezzo fisico come una linea di trasmissione esente da errori di trasmissione.

Network Layer Si occupa di rendere i livelli superiori indipendenti dai meccanismi e dalle tecnologie di trasmissione usate per la connessione e prendersi carico della consegna a destinazione dei pacchetti. È responsabile del *routing* ovvero della scelta ottimale del percorso di rete da utilizzare per garantire la consegna delle informazioni dal mittente al destinatario, scelta svolta dal router attraverso dei particolari algoritmi di routing e tabelle di routing. È responsabile inoltre della conversione dei dati nel passaggio fra una rete ed un'altra con diverse caratteristiche, come il protocollo di rete utilizzato: si deve occupare quindi di tradurre gli indirizzi di rete, valutare

la necessità di frammentare i pacchetti dati se la nuova rete ha una diversa Maximum Transmission Unit (MTU) e di valutare la necessità di gestire diversi protocolli attraverso l'impiego di gateway. L'unità dati fondamentale è il pacchetto o *datagram*.

Transport Layer Permettere un trasferimento di dati trasparente e affidabile (implementando anche un controllo degli errori e delle perdite) tra due host. È il primo livello realmente end-to-end, cioè da host sorgente a destinatario. Si occupa di stabilire, mantenere e terminare una connessione, garantendo il corretto e ottimale funzionamento della sottorete di comunicazione nonché del controllo della congestione: evitare che troppi pacchetti dati arrivino allo stesso router contemporaneamente con effetto di perdita di pacchetti stessi. A differenza dei livelli precedenti, che si occupano di connessioni tra nodi contigui di una rete, il Transport Layer si occupa solo del punto di partenza e di quello finale. Si occupa anche di effettuare la frammentazione dei dati provenienti dal livello superiore in pacchetti, detti "segmenti" e trasmetterli in modo efficiente ed affidabile usando il livello rete ed isolando da questo i livelli superiori. Inoltre, si preoccupa di ottimizzare l'uso delle risorse di rete e di prevenire la congestione.

Session Layer Consente di aggiungere, ai servizi forniti dal livello di trasporto, servizi più avanzati, quali la gestione del dialogo (mono o bidirezionale), la gestione del token (per effettuare mutua esclusione) o la sincronizzazione (inserendo dei checkpoint in modo da ridurre la quantità di dati da ritrasmettere in caso di gravi malfunzionamenti). Si occupa anche di inserire dei punti di controllo nel flusso dati: in caso di errori nell'invio dei pacchetti, la comunicazione riprende dall'ultimo punto di controllo andato a buon fine.

Presentation Layer Si occupa di trasformare i dati forniti dalle applicazioni in un formato standardizzato e offrire servizi di comunicazione comuni, come la crittografia, la compressione del testo e la riformattazione.

Application Layer Il livello Applicazione è quello più vicino al livello utente fornisce quindi un interfaccia tra le applicazioni e lo stack di rete sottostante che si occupa dell'invio di messaggi. I protocolli di livello applicazione si occupano quindi dello scambio di informazioni tra apps in esecuzione sull'host sorgente e quello destinatario della comunicazione.

2.2 ISO/OSI vs TCP/IP

TCP/IP sviluppato inizialmente dal dipartimento della difesa americano e utilizzato nei primi computer UNIX-based attualmente è lo *standard de facto* per tutte le comunicazioni internet.
TCP/IP, come l'OSI model, è strutturato su più livelli alcuni molto simili per caratteristiche e funzionalità a quelli di ISO/OSI: TCP/IP accorpa in un unico layer funzionalità contenute su più livelli del modello OSI.

TCP/IP è l'approccio utilizzato in ogni tipo di comunicazione internet. L'obbiettivo di ISO/OSI invece è quello fornire uno standard da usare come guideline per la definizione di protocolli e applicazioni internet.

I layer nello stack TCP/IP sono quattro e sono così organizzati rispetto al modello OSI come illustrato nella figura **??**.

2.3 Wi-Fi

Wi-Fi indica una tecnologia che consente a calcolatori collocati su di una stessa WLAN (Wireless Local Area Network) di comunicare senza fili attraverso specifiche frequenze di onde radio. Sempre più dispositivi dispongono di interfacce di rete Wi-Fi dai laptop, gli smartphone fino agli elettrodomestici di ultima generazione che possono essere così interconnessi entro un certo

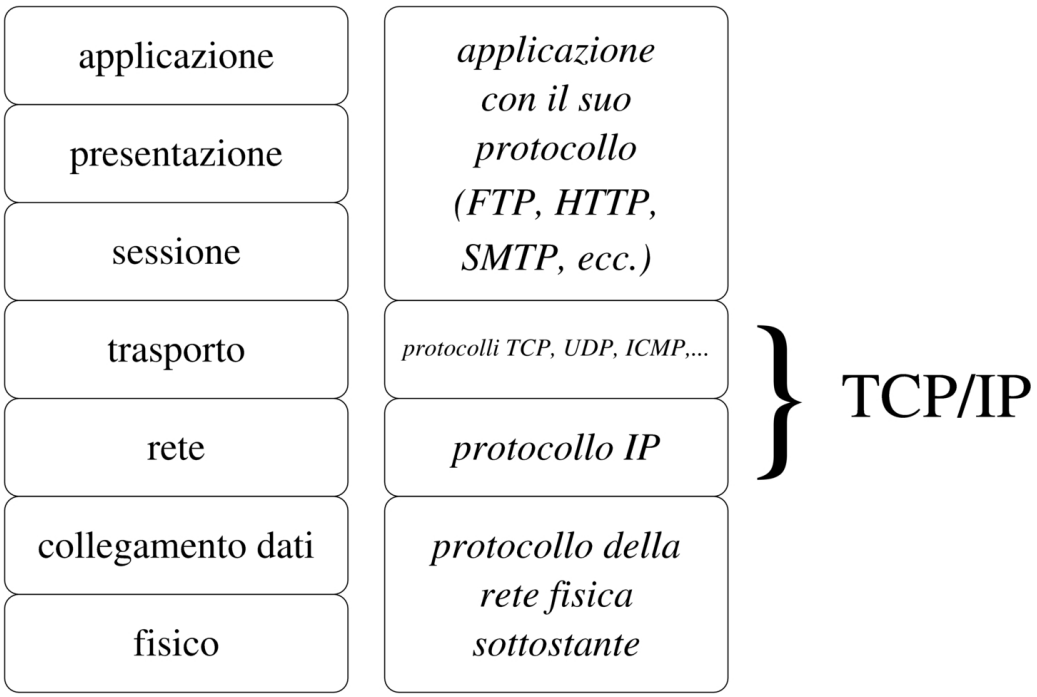

Figura 2.1: Stack di rete ISO/OSI e TCP/IP

raggio di copertura. La tecnologia Wi-Fi può essere usata per fornire connettività internet ai dispositivi presenti nel raggio di copertura della WLAN se la Local Area Network è connessa a internet.

2.3.1 WLAN architecture and types

L'architettura di una WLAN è caratterizzata da diverse componenti.

Stazioni In una WLAN ciascun dispositivo munito di Wireless Network Interface Controllers (WNICs) e che quindi può comunicare senza fili è detto stazione. Vi sono due categorie di stazioni:

- Access Points (AP) ovvero dispositivi elettronici che ricevono/trasmettono segnali radio da/verso nodi mobili equipaggiati con schede di rete Wi-Fi.

- Clients tutti i nodi mobili che possono essere equipaggiati con una wireless network interface come ad esempio laptops, smartphones, workstations.

Basic service set Il Basic Service Set (BSS) è un insieme di tutte le stazioni che possono comunicare tra loro. Ciascun BSS ha un proprio identificativo detto BSSID che corrisponde al indirizzo MAC dell'access point che serve i diversi clients per quella BSS.
Vi sono due tipi di BSS:

- Indipendent BSS (IBSS) ovvero una *rete ad-hoc* caratterizzata dall'assenza di un access point. Questo tipo di BSS non può quindi essere interconnesso con altri Basic Service Set.
- Infrastructure BSS caratterizzati dalla presenza di un access point, un BSS di questo tipo può essere connesso con altri Basic Service Set.

Extended Service Set Un Extended Service Set (ESS) è un insieme di BSS interconnessi tra loro. Gli access points in un ESS sono connessi tra loro da un *Distribution System*. Ciascun EES è caratterizzato da una stringa identificativa lunga al massimo 32 byte detta SSID.

Distribution System Il Distribution System (DS) interconnette tra loro gli access point di diversi EES. Un access point può essere principale, di inoltro o remoto. Un access point principale è collegato tipicamente alla rete cablata. Un access point di inoltro trasmette i dati fra le stazioni remote e principali. Un access point remoto accetta i collegamenti dai client senza fili e li passa a quelli di inoltro o quelli principali.

Esistono due tipologie di rete WLAN che differiscono dalle modalità di comunicazione:

- Infrastructure, i nodi comunicano tra loro attraverso a una base station che funge da wireless access point.

- Reti ad-hoc, ovvero una rete dove le stazioni possono comunicare peer-to-peer (P2P) senza alcun access point. Questo viene realizzato tramite IBSS.

2.4 IEEE 802.11

IEEE 802.11 è uno standard di trasmissione per reti WLAN operanti su frequenze 2.4, 3.6, 5 e 60 GHz che definisce un interfaccia di comunicazione base per comunicazione Wi-Fi. Le specifiche definite nello standard 802.11 si focalizzano sul livello fisico e MAC del modello ISO/OSI. Il sistema di numerazione 802.11 è dovuto a IEEE che utilizza 802.x per indicare una famiglia di standard per le comunicazione di rete tra cui lo standard *Ethernet* (IEEE 802.3). Per tanto IEEE 802.11 si adegua perfettamente agli altri standard 802.x per reti locali wired e le applicazioni che lo utilizzano non dovrebbero notare nessuna differenza logica, una degradazione delle performance invece è tuttavia possibile. IEEE 802.11b è stato il primo protocollo largamente utilizzato seguito da 802.11a, 802.11g, 802.11n, and 802.11ac. Vi sono altri standard nella famiglia 802.11 (c-f, h, j) che sono per lo più piccole modifiche, estensioni o correzioni alle precedenti specifiche.
Per quanto concerne le performance lo stream data rate può arrivare fino a 780 Mbit/s in 802.11ac grazie anche alla tecnologia MIMO (Multiple-Input and Multiple-Output) che consente di aumentare la capacità del canale trasmissivo usando più trasmittenti e ricevitori sfruttando così il fenomeno del *multipath-propagation* ovvero un segnale radio può raggiungere un ricevitore attraverso diversi percorsi, *path*.

2.4.1 Physical Layer in 802.11

Come già detto IEEE 802.11 utilizza le bande di frequenza 2.4, 3.6, 5 e 60 GHz e a *livello fisico* vengono usate delle tecniche di modulazione *half-duples*: in particolare viene utilizza la Orthogonal Frequency-Division Multiplexing (OFDM) che utilizza un numero elevato di sotto-portanti ortogonali tra di

loro, oppure quella chiamata Direct Sequence Spread Spectrum (DSSS), che è una tecnologia di trasmissione a banda larga nella quale ogni bit viene trasmesso come una sequenza ridondante di valori, detti chip, rendendola così più resistente ad eventuali interferenze.

2.4.2 Media Access Control

Media Access Control (MAC) è un *sublayer* del livello *Data Link* del modello OSI. MAC fornisce meccanismi di indirizzamento e di controllo di accesso al canale che consentono a nodi mobili di comunicare attraverso una rete con medium condiviso. L'hardware che implementa MAC è detto *media access controller*. Le funzioni principali del MAC sono quindi quelle di regolamentare l'accesso al mezzo fisico, frammentazione dati in frame e riconoscimento frame stessi e controllo degli errori.

2.4.3 CSMA/CA

Carrier Sense Multiple Access with Collision Avoidance è un protocollo di accesso multiplo in cui i nodi cercano di evitare a priori il verificarsi di collisioni in trasmissione. Questo approccio è l'ideale per tipologie di reti nella quale non risulta possibile (oppure poco affidabile e dispendioso) rilevare un'avvenuta collisione.
Quando un nodo vuole effettuare una trasmissione ascolta il canale *(listen-before-talk)* (LBT): se il canale risulta *idle* il nodo aspetta un certo *DISF time (Distribuited Inter Frame Space)* trascorso il quale, se il canale risulta ancora libero, comincerà a trasmettere. A termine della trasmissione il nodo sorgente aspetterà per certo intervallo *SISF (Short Inter Frame Space)*, più piccolo di DISF, la ricezione di un *ACK*. Se il nodo sorgente non riceve alcun ACK ritrasmetterà il messaggio per un certo numero di volte.
Per tutta la durata della trasmissione e per la durata dello SISF time le altre stazioni, trovando il canale occupato, non avvieranno altre comunicazioni evitando così collisioni. La durata dello SISF inferiore a quello dell'intervallo

di DISF assicura che nessuna stazione comincerà una trasmissione prima della ricezione dell'eventuale messaggio di acknowledgment da parte del nodo che ha appena concluso la trasmissione.
Nel caso in cui una stazione volesse trasmettere e rileva il canale occupato attenderà per un certo intervallo di tempo casuale, detto intervallo di *back-off*, prima di riprovare a trasmettere. L'intervallo di back-off è realizzato per mezzo di un timer che decrementa il valore di un contatore, inizializzato con il valore dell'intervallo, solamente nei periodi di inattività del canale, ovvero quando non vi sono trasmissioni, il valore del contatore resterà invece invariato durante i periodi di trasmissione da parte di altre stazioni (*frozen back-off*). Quando il valore del contatore raggiungerà lo zero la stazione effettuerà un nuovo tentativo di trasmissione. Questo meccanismo di accesso al mezzo è detto *Basic Access Mechanism*

2.4.4 Il problema dei nodi nascosti

Il problema dei nodi nascosti in una rete wireless si ha quando un nodo all'interno della rete è visibile da un Access Point ma non da tutte le altre stazioni collegate al medesimo AP. Questo può comportare una serie di problemi per quanto riguarda il controllo dell' accesso al mezzo.

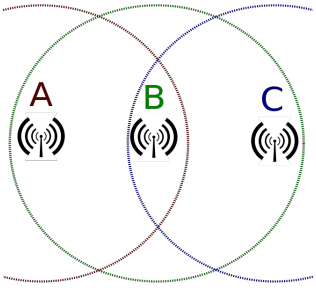

Figura 2.2: Il problema dei nodi nascosti

Come si può vedere dalla figura **??** la stazione A e la C sono nel raggio di copertura della stazione B. Per qualche motivo (come può essere la distanza o un ostacolo) A e C non possono comunicare direttamente e quindi non

possono nemmeno rilevare (*sensing*) la portante trasmessa dall'altra stazione verso la stazione centrale B. In particolare si possono quindi verificare delle collisioni quando sia A che C, rilevando il canale libero, effettuano in contemporanea una trasmissione verso B.
Per ovviare a questo problema IEEE 802.11 definisce un meccanismo opzionale che introduce due tipi di pacchetti di controllo, in particolare:

- *RTS* (Request To Send), quando un nodo vuole trasmettere, prima di inviare il frame vero e proprio, invia al destinatario un pacchetto di tipo RTS contenente destinatario del messaggio, mittente e durata della trasmissione che seguirà.

- *CTS* (Clear To Send), quando un nodo riceve un pacchetto di tipo RTS risponde con un pacchetto di tipo CTS che contiene essenzialmente le stesse informazioni contenute nel frame di tipo RTS; quando il nodo mittente avrà ricevuto il frame CTS potrà cominciare l'inoltro del frame effettivo precedentemente annunciato tramite il rispettivo RTS.

I pacchetti RTS e CTS vengono inoltrati a tutte le stazioni comprese quindi anche quelle nascoste al mittente che si metteranno in attesa per tutta la durata della trasmissione come specificato dai frame di controllo.
Questo meccanismo non è del tutto esente da collisioni. Infatti le collisioni possono ancora avvenire durante lo scambio dei pacchetti di controllo: ad esempio due stazioni mandano contemporaneamente una Request To Send. Nonostante ciò la probabilità di collisione risulta essere più bassa e meno significativa rispetto all'approccio che non fa uso dei pacchetti RTS/CTS in quanto i frame di controllo hanno una dimensione molto ridotta (up to 2347 bytes).
Se una stazione vuole trasmettere un pacchetto di dimensione inferiore al frame di controllo il messaggio verrà inoltrato immediatamente senza prima generare il corrispettivo RTS.

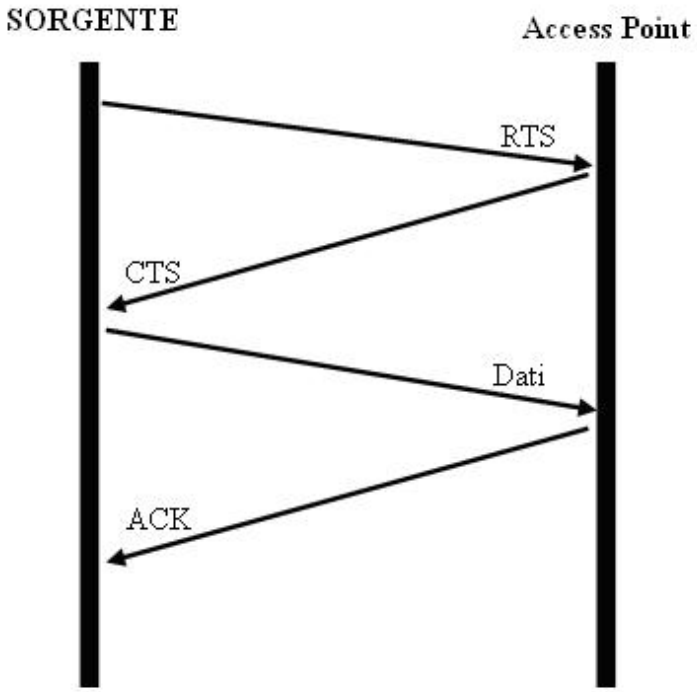

Figura 2.3: *Four-way handshake* via RTS/CTS

2.4.5 IEEE 802.11 frame

IEEE 802.11 definisce tre tipologie di frame:

- DATA, contengono meramente dati.
- CTRL, servono per facilitare l'interscambio di data frame tra le stazione; appartengono a questa categoria i frame RTS, CTS e ACK.
- MGMT, frame utili al mantenimento della comunicazione; i *beacon frame* (frame inviato periodicamente da un AP per annunciare al sua presenza e il suo SSID) appartengono a questa categoria.

Ciascun frame è composto da un *MAC header*, un *payload* e un *frame check sequence (FCS)*.

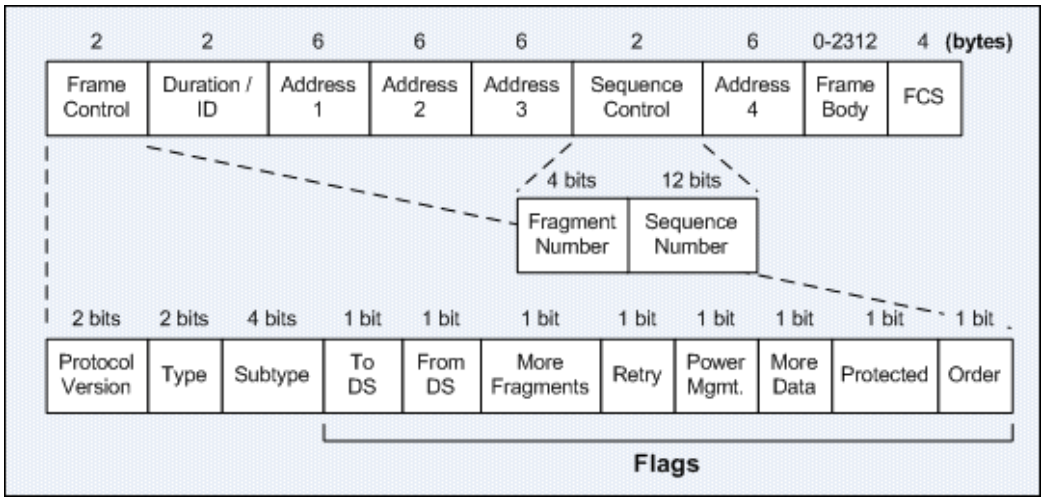

Figura 2.4: IEEE 802.11 frame

MAC header I primi due byte del MAC header contengono un campo molto interessante, il *frame control*. Il frame control contiene diversi sottocampi:

- Protocol Version, ovvero due bit rappresentanti la versione del protocollo, posto sempre a zero; altri valori sono riservati per un uso futuro.

- Type, due bit identificanti appunto il tipo di frame.

- Subtype, quattro bit che identificano il sottotipo del frame; ad esempio come beacon è un sottotipo di MGMT.

- ToDS & FromDS, ciascun campo occupa un bit e indica se un data frame è diretto o proviene da un distribution system. I frame di tipo CTRL e MGMT hanno entrambi i flag settati a zero.

- More Fragments, bit settato quando un pacchetto è viene frammentato in più frame per la trasmissione; tutti i pacchetti con eccezione dell'ultimo inviato avranno questo flag settato.

- Retry, indica se un frame è stato oggetto di ritrasmissione; utile per eliminare eventuali frame duplicati.

- Power Management: indica il *power management state* (ovvero se la stazione è in power-save state o meno) del mittente settato dopo la trasmissione. Gli AP non setteranno mai questo bit in quanto sempre attivi per la gestione delle connessioni.

- More Data, questo bit indica che c'è almeno un pacchetto disponibile; settato dagli AP per facilitare le stazioni in power-save mode.

- Protected, indica se il payload del frame è stato cifrato o meno.

- Order, questo bit è settato solamente quando i frame sono inviati in ordine uno dietro l'altro; spesso ciò non avviene per motivi di performance.

Gli altri campi contenuti nell'header MAC sono la durata di trasmissione, gli indirizzi MAC (*source*, *destination*, *transmitter* e *receiver*) e il *Sequence Control*.
Il campo Sequence Control è composto da due byte usato per identificare l'ordine dei frame spediti. I primi 4 bit corrispondono al *fregmentation number* e gli ultimi 12 bit sono il *sequence number*. Il fragmentation number indica il numero di ogni pacchetto precedentemente frammentato, il sequence number, invece, è un valore modulo 4096 assegnato a un frame e rimane costante per ogni ritrasmissione o per ciascun fragment di quel pacchetto.

Payload Il Payload ha dimensione variabile da 0 a 2304 byte; contiene informazioni provenienti dai livelli di rete superiori.

Frame check sequence (FCS) Spesso detto anche *Cyclic Redundancy Check* (CRC) permette di verificare l'integrità di un frame appena ricevuto: quando un frame sta per essere spedito la stazione sorgente calcola questo valore e lo appende al frame IEEE 802.11. Quando un nodo riceve il frame

ricalcola l'FCS sulla base dei dati ricevuti e lo confronta con il valore contenuto nel *trailer* del pacchetto; se i due valori coincidono il frame non ha subito delle modifiche durante la trasmissione.
Il campo FCS occupa gli ultimi quattro byte del frame IEEE 802.11

2.4.6 Security in IEEE 802.11

Data la sempre più larga diffusione delle reti Wi-Fi e dalla natura del loro segnale (è molto difficile controllare quale dispositivo riceve il segnale radio) la sicurezza è un aspetto molto importante e assolutamente da non sottovalutare in 802.11. Nel corso degli anni sono stati sviluppati e proposti diversi approcci per rendere le reti WLAN sempre meno sensibili a intercettazioni e attacchi da parte di terzi.

Access Control List Un primo banale approccio è quello dell'*access control list*. L' Access Point mantiene una lista degli indirizzi MAC autorizzati alla comunicazione: l' AP riceve messaggi provenienti solo dai clients presenti nell'access control list. Qualsiasi messaggio proveniente da una stazione non presente nella lista sarà ignorato.
Questo approccio presenta due grandi difetti. Innanzitutto fornisce solamente una politica di controllo degli accessi senza fornire nessun meccanismo di protezione sui dati trasmessi. Inoltre questo approccio può essere facilmente raggirato tramite una tecnica di *MAC spoofing*. In particolare tramite un software di *wireless network analysis* è possibile monitorare il traffico delle rete WLAN vicine e quindi captare informazioni sensibili da eventuali messaggi trasmessi in chiaro: data la mancanza di confidenzialità nei messaggi trasmessi su reti che adottano esclusivamente la politica dell'Access Control List come protezione è possibile quindi risalire ad indirizzi MAC autorizzati alla comunicazione. A questo punto è possibile modificare l'indirizzo MAC dell'interfaccia di rete (possibile sia in ambiente UNIX che Windows) per impersonare un altro client della rete WLAN.

WEP (Wired Equivalent Privacy) WEP (Wired Equivalent Privacy) è stato il primo protocollo di sicurezza definito nello standard IEEE 802.11. L'obbiettivo di WEP è quello di garantire confidenzialità e integrità del traffico trasmesso in maniera wireless. Il nome è dovuto al fatto che WEP è stato pensato per fornire confidenzialità sui dati trasmessi paragonabile a quella delle reti cablate.
WEP sfrutta il cifrario a chiave simmetrica RC4 con chiave a 64 o 128 bit. La chiave WEP utilizzata è la concatenazione di due valori: il primo dinamico detto Initialization Vector (IV) e la seconda parte statica corrispondente alla chiave segreta condivisa. Il vettore di inizializzazione è una sequenza di 24 bit generata casualmente al momento dell'invio del frame da parte dell'interfaccia di rete (per ogni trasmissione verrà generato un IV in quanto RC4 è un *cifrario a flusso*). A seconda della lunghezza della WEP key la chiave segreta condivisa sarà quindi lunga 40 bit nel caso di una WEP key di 64 bit oppure 104 bit nel caso di una chiave a 128 bit.
Al momento dell'invio di un frame la stazione sorgente genera il vettore di inizializzazione e lo concatena alla shared key. Una volta che la WEP key è stata formata viene data in pasto all'algoritmo di cifratura RC4 per produrre una stringa pseudo-casuale della lunghezza pari ai dati da trasmettere. Una volta generata la stringa pseudo-random quest'ultima viene posta in XOR dalla scheda di rete con i dati da trasmettere: il risultato assieme al vettore di inizializzazione in chiaro sarà appeso a un header IEEE 802.11 e trasmesso verso il destinatario del messaggio.
Quando il nodo destinatario riceve il messaggio cifrato come prima cosa legge l'IV lo concatena alla shared key e calcola la psudo-random string via RC4 (data la stessa WEP key la stringa pseudo-casuale generata sarà sempre uguale). Il risultato ottenuto viene posto in XOR con i dati cifrati contenuti nel frame ottenendo così il testo in chiaro.
A partire dal 2003 questo approccio non è più considerato sicuro a causa dalle numerose falle presenti in WEP e dalla facilità con cui RC4 può essere violato.

WPA (Wi-Fi Protected Access) Una volta scoperte le falle che affliggevano WEP è iniziato lo sviluppo del protocollo IEEE 802.11i, un nuovo standard considerato pienamente sicuro. Nel frattempo viene rilasciato dalla Wi-Fi Alliance WPA (Wi-Fi Protected Access) che soddisfa molte delle linee guide di IEEE 802.11i Il WPA è caratterizzato da tre componenti principali:

- TKIP (Temporal Key Integrity Protocol), è la componente che più va a sostituire la logica di WEP risolvendo la maggior parte delle sue vulnerabilità. Una delle innovazioni più importanti è quella che ogni messaggio trasmesso viene cifrato con una chiave diversa in modo tale da non esporre la chiave principale.
  Molte funzioni di crittografia sono built-in nell'hardware di rete per tanto, non essendo possibile un aggiornamento software, per rendere compatibile a pieno WPA con il precedente hardware IEEE 802.11 il nuovo standard sfrutta alcune delle feature usate anche da WEP: in particolare anche WPA fa utilizzo di RC4. WPA inoltre utilizza un meccanismo di *key hierarchy* ovvero la chiave principale (Pairwise Master Key) viene utilizzata per generare chiavi temporanee come le sessione key, group keys etc etc. In particolare WPA sfrutta RC4 in modo diverso rispetto a WEP ovvero RC4 viene utilizzato per generare una chiave temporanea a partire dalla shared key anzichè per cifrare direttamente il messaggio. La prima chiave a essere generata è la *session key* che sarà poi utilizzata come seme per la generazione delle future *per-packet key.*
  Ciascuna per-packet key, lunga 104 bit, è generata da una funzione hash che calcola un digest a partire dall'indirizzo MAC sorgente, il vettore di inizializzazione (che in WPA è stato esteso da 24 a 48 bit ed è implementato come un contatore, emphsequence counter, per evitare *replay attack*) e la session key. Una volta ottenuta la per-packet key le operazioni di cifratura e decifratura sono identiche a quelle di WEP con la sola differenza che il vettore di inizializzazione è sostituito con i 16 bit meno significativi del IV di WPA e con un dummy byte inserito

in mezzo.
TKIP risulta quindi essere un sistema di cifratura a 128 bit a chiave dinamica molto più sicuro rispetto al sistema adottato da WEP che prevedeva 24 bit dinamici con una chiave di 40 o 104 bit statica.

- MIC (Message Integrity Code) detto anche *Michael* è una funzione hash con chiave pensata per proteggere l'integrità di un pacchetto. Il valore calcolato a partire dall'intero pacchetto non criptato è un digest di 8 byte. La funzione hash utilizzata da MIC è una funzione progettata per poter essere eseguita su dispositivi con capacità di calcolo scarse come possono essere appunto le schede di rete. A causa del bassa capacità di calcolo la funzione hash utilizzata fornisce protezione pari a un algoritmo di cifratura con con una chiave a 20 bit considerato lo standard per una bassa protezione. Per compensare quindi una bassa protezione WPA introduce una serie di contromisure per proteggere la rete da eventuali modifiche dei pacchetti da parti di terzi (packets modification attack) ovvero quando la rete intercetta un pacchetto compromesso disabilità il collegamento wireless con gli host coinvolti per 60 secondi ed ogni dispositivo compromesso deve richiedere forzatamente una nuova session key.
  Il pericolo causato da questo tipo di contromisura che un malintenzionato può intenzionalmente forgiare pacchetti invalidi in modo tale che l'access point adotti questo approccio di continuo generando così un denial of service.

- 802.1x Port based Network Access Control è un protocollo per il controllo degli accessi in una rete wireless basato sul controllo delle porte di accesso alla rete (switch e/o access point). Questo protocollo divide la rete in tre entità principali ovvero *supplicant*, cioè il client che vuole connettersi alla rete, l'*authenticator* ovvero il punto di accesso alla rete dove il supplicant vuole fisicamente connettersi (tendenzialmente uno switch o un access point che collega il nodo alla rete) e l'*authentication*

server il cui compito è quello di validare l'accesso alla rete del client. In una rete che adotta 802.1x un nodo deve prima autenticarsi prima di poter accedere e comunicare nella rete wireless. Uno switch o un access point accetta come traffico proveniente da un client non autentificato soltanto messaggi di autenticazione di tipo EAP (Extensible Authentication Protocol) bloccando qualsiasi altra forma di traffico fino a che il client non effettua l'accesso con successo. 802.1x è inoltre responsabile della generazione e della consegna della session key una volta che il nodo si è autentificato con successo.

IEEE 802.11i (a.k.a. WPA2) La più sostanziale differenza tra WPA e IEEE 802.11i (WPA2) è che 802.11i adotta AES (Advanced-Encription Standard) per cifrare i frame. AES è un algoritmo di cifratura a blocchi che rappresenta lo stato dell'arte per quel che riguarda algoritmi di cifratura. L'unico inconveniente è che le schede di rete che supportano esclusivamente WEP non possono essere aggiornate via software per supportare AES e quindi IEEE 802.11i. Se una rete volesse adottare IEEE 802.11i tutte le stazioni dovrebbero montare hardware di rete compatibile con WPA2.

2.5 Livello Network

Qui in seguito verranno presentati i principali protocolli utilizzati a livello rete ovvero il terzo livello dello stack ISO/OSI.

2.5.1 IPv4

IPv4 (Internet Protocol version 4), come descritto nel RFC 791 dell'IETF del 1981, è un protocollo connectionless per l'uso su reti a commutazione di pacchetto, come ad esempio Ethernet. È un protocollo di tipo *best-effort* ovvero non viene garantita l'effettiva consegna o se i pacchetti saranno recapitati nel giusto ordine oppure duplicati. I pacchetti scambiati a livello network sono detti *datagram*.

Indirizzi IPv4 IPv4 utilizza indirizzi a 32 bit (suddivisi in quattro gruppi da un byte tramite la *decimal dotted notation*) per identificare univocamente un singolo host . Molti degli indirizzi IP sono però riservati per scopi particolari (reti domestiche o indirizzi di *multicast*) facendo si che l'effettivo *pool* di indirizzi IP disponibili sia piuttosto ridotto in confronto alla diffusione degli ultimi anni dei dispositivi che possono connettersi alla rete.
Un indirizzo IP, ad esempio, può essere 192.168.1.102 (indirizzo IP classe C); in un indirizzo IP, a seconda della classe e dello scopo, i bit più significativi vengono utilizzati per identificare la rete quelli meno significativi, invece, sono utilizzati per indirizzare direttamente i singoli host. Il numero di bit utilizzati per individuare la rete o gli host dipende dalla classe dell'indirizzo IP e da come è stata progettata la rete stessa.

IPv4 header Un datagram IP è formato da un header e da una porzione dati. La dimensione massima del datagram può essere di 65535 byte.

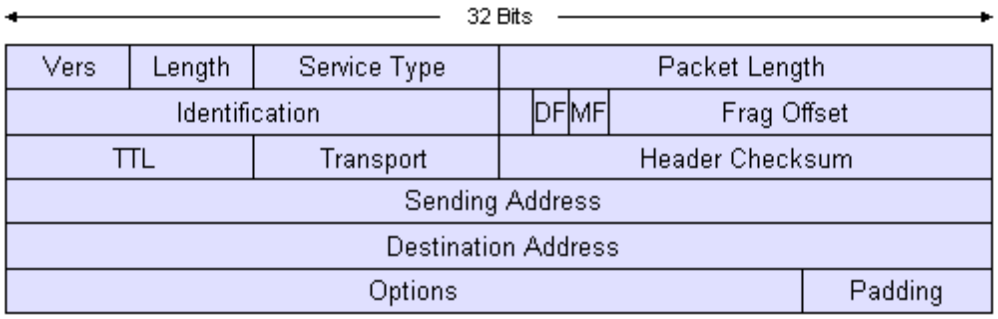

Figura 2.5: Formato dell'header IPv4

I campi dell'header IP sono:

- **Version**, 4 bit indicanti la versione del protocollo IP del pacchetto; in IPv4 questo campo assume il valore 4.
- **Internet Header Length** (IHL), 4 bit indicanti la lunghezza dell'header IP (in word da 32 bit); l'header può avere dimensione variabile a seconda se il campo *Options* è settato o meno.

- **Type of Service** (TOS), ottetto che nelle specifiche iniziali del protocollo (RFC 791) davano la possibilità all'host mittente di specificare il modo e la priorità con cui il nodo destinatario doveva trattare il datagram. In realtà questo campo non è mai stato largamente utilizzato e negli ultimi e recentemente questi 8 bit sono stati ridefiniti e hanno la funzione di *Differentiated services* (DiffServ) nell'IETF e *Explicit Congestion Notification* (ECN) codepoints (RFC 3168) necessari per le tecnologie basate su streaming in tempo reale come ad esempio per il *Voice over IP* (VoIP).

- **Total Length**, 16 bit, indica la dimensione massima in byte dell'intero dagram IP (header e dati); tale lunghezza può variare da un minimo di 20 byte (header minimo e senza alcun dato) a un massimo di 65535 byte. In ogni momento un host deve poter gestire datagrammi di dimensione minima 576 byte mente, se necessario, possono frammentare datagram di dimensione maggiore.

- **Identification**, 16 bit, definito per identificare univocamente i vari frammenti in cui può essere suddiviso un datagram IP. Può essere anche utilizzato per valutare la presenza di datagram ridondanti (indirizzo IP sorgente più Identification identificano univocamente un pacchetto).

- **Flags** ovvero 3 bit utilizzati per il controllo del protocollo e la frammentazione dei datagrammi. Il primo bit detto *Reserved* è sempre settato a zero; vi sono poi DF (Don't Fragment) che se settato a 1 indica che il pacchetto non deve essere frammentato e MF (More Fragment) che se settato a 0 indica che è l'ultimo frammento (o il solo frammento originario) e tutti gli altri frammenti avranno quindi questo flag posto a 1.

- **Fragment Offset**, 13 bit, indica l'offset, misurato in blocchi da 8 byte, di un particolare frammento relativamente all'inizio del datagram IP

originario. L'offset massimo risulta pertanto essere 65526 byte che, sommato la dimensione dell'header IP, potrebbe eccedere la dimensione massima di 65535 byte prevista per un datagram IP.

- **Time To Live** (TTL), 8 bit, indica il *tempo di vita* di un pacchetto necessario per evitarne la persistenza indefinita sulla rete nel caso in cui non si riesca a recapitarlo al destinatario. Storicamente il TTL misurava i "secondi di vita" del pacchetto, mentre ora esso misura il numero di "salti" da nodo a nodo della rete: ogni router che riceve il pacchetto prima di inoltrarlo ne decrementa il TTL (modificando quindi anche il campo Header Checksum), quando questo arriva a zero il pacchetto non viene più inoltrato ma scartato. Tipicamente, quando un pacchetto viene scartato per esaurimento del TTL, viene automaticamente inviato un messaggio ICMP al mittente del pacchetto, specificando il codice di richiesta scaduta; la ricezione di questo messaggio ICMP è alla base del meccanismo di traceroute.

- **Protocol**, 8 bit, indica il codice associato al protocollo utilizzato nel campo dati del pacchetto IP.

- **Header Checksum**, 16 bit, è un campo usato per il controllo degli errori dell'header. Ad ogni hop, il checksum viene ricalcolato (secondo la definizione data in RFC 791) e confrontato con il valore di questo campo: se non c'è corrispondenza il pacchetto viene scartato. È da notare che non viene effettuato alcun controllo sulla presenza di errori nel campo Data deputandolo ai livelli superiori.

- **Source Address**, 32 bit, indirizzo IP associato all'host del mittente del datagram.

- **Destination Address**, 32 bit, indirizzo IP associato all'host del destinatario del datagram.

- **Options** contiene opzione facoltative e poco usate per usi più specifici del protocollo. La dimensione del campo Options deve essere multipla

di 32 bit altrimenti, per raggiungere tale scopo, vengono aggiunti dei bit privi di significato, **padding**.

2.5.2 NAT

NAT Network Address Translation è una tecnica che consente di mappare uno o più indirizzi IP di un *address space* in un indirizzo IP appartenente a uno spazio di indirizzi diverso: questo avviene molto semplicemente modificando le informazioni sugli indirizzi di rete presenti in un datagram IP in transito su di un dispositivo di routing. Questa tecnica è stata originariamente pensata per semplificare le operazioni di rerouting del traffico IP in una rete senza rinumerare ciascun host di quella rete: ad esempio hosts appartenenti a una rete privata possono essere mappati su di un unico indirizzo IP appartenente a uno spazio di indirizzamento differente (tendenzialmente un indirizzo IP pubblico). Questo meccanismo viene realizzato memorizzando in un dispositivo di routing delle *translation table* che mappano gli indirizzi IP non visibili dall'esterno in uno (o più) indirizzi IP pubblici modificando i datagram in uscita facendo sembrare dall'esterno che il mittente del pacchetto è il routing device. Quando viene ricevuto un pacchetto dall'esterno verso la rete interna privata il routing device, che mantiene le translation table, inoltrerà il pacchetto all'effettivo host destinatario.
Questa tecnica è molto utilizzata anche per preservare il consumo di indirizzi IPv4. Gli indirizzi IPv4 essendo formati da 32 bit possono generare un pool di 4.294.967.296 indirizzi: se si considerano i vari diversi indirizzi riservati e la sempre più diffusione di dispositivi che possono connettersi alla rete come laptop, smartphone e IoT questo può rappresentare un limite molto serio. Per tanto a partire dal 2004 è disponibile una nuova versione dell'internet protocol IPv6 che adotta indirizzi di dimensione pari a 128 bit.

2.5.3 ARP

Il protocollo ARP (Address Resolution Protocol) è usato per determinare, dato un indirizzo IP, il corrispettivo indirizzo hardware di livello data-link. Quando un host vuole spedire un datagramma ad un altro nodo conoscendo il suo indirizzo IP ma non quello fisico, fa una richiesta ARP in broadcast sulla rete di appartenenza, chi ha l?indirizzo richiesto risponde con il proprio indirizzo MAC.
Il corrispettivo protocollo ovvero che consente di risalire all'indirizzo IP di un host dato il suo MAC address è il protocollo RARP (Reverse Address Resolution Protocol).

2.5.4 Frammentazione

Per rendere il protocollo tollerante alle eventuali differenze di specifiche sottoreti è possibile effettuare la frammentazione dei datagram IP. In particolare ogni qual volta un datagram IP deve essere trasmesso attraverso un link con MTU (Maximum Transfer Unit, che rappresenta un limite superiore alla dimensione di un frame di livello data-link dipendente dall'hardware di rete) inferiore alla dimensione del pacchetto stesso questo verrà frammentato in più fragment che soddisfano il valore della MTU. Il router quindi preleverà il payload dal datagram IP e lo frammenterà in modo tale che il nuovo segmento dati più l'header IP possa essere contenuto come body di un frame di livello data-linke e facendo si che ogni payload del datagram abbia dimensione multipla di 8 byte (campo Offset in header IP). In tutti i frammenti tranne l'ultimo inviato, che in genere avrà anche una dimensione minore rispetto ai precedenti, avranno il flag MF (More Fragment) settato a 1.

2.5.5 Routing

Uno dei compiti caratteristici del livello rete è il *routing* ovvero decidere su quale interfaccia di rete o porta inoltrare un pacchetto ricevuto verso la

sua destinazione finale. In particolare i router mantengono delle tabelle di instradamento che possono essere popolate manualmente (*routing statico*, poco scalabile adatto esclusivamente per reti molto piccole) o popolate da appositi protocolli di routing che mantengono le tabelle di routing aggiornate a seconda del variare della topologia della rete. Vi sono due tipologie principali di protocolli di routing: *Distance Vector* e *Link State*. Nel protocollo Distance Vector ogni router misura la distanza (utilizzando una metrica che può tenere conto di diversi fattori) dai nodi adiacenti tramite le informazioni ricevute dai nodi vicini. In un protocollo Link State, invece, ciascun nodo acquisisce informazione sullo stato dei collegamenti adiacenti e inoltra queste informazioni a tutti gli altri nodi sulla rete attraverso un pacchetto di tipo Link State trasmesso tramite un algoritmo di *link state broadcast*.

2.5.6 IPv6

Internet Protocol version 6 è la versione più recente dell'Internet Protocol formalizzato nel 1998 da IETF a seguito della recente crescita esponenziale dei dispositivi che accedono ad internet. IPv6 adotta indirizzi a 128 bit fornendo così 2^{128} indirizzi ovvero circa 3,4 x 10^{38} indirizzi. I protocolli IPv4 e IPv6 non sono stati progettati per essere interoperabili complicando così la transizione verso l'ultima versione dell'Internet Protocol seppure siano stati ideati dei meccanismi di transizione da un protocollo verso l'altro. Tuttavia la migrazione completa verso IPv6 dovrebbe venire nel 2025 con la deprecazione definitiva di IPv4.
IPv6 oltre a estendere lo spazio di indirizzamento porta con se alcune nuove feature come la dimensione fissa dell'header a 40 byte, datagram non frammentabili dai router e la rimozione del checksum dall'header (ridondate in quanto presente in altri layer dello stack di rete). Inoltre è stati aggiunto il supporto nativo alla sicurezza (IPsec) e inseriti meccanismi di autoconfigurazione come ad esempio ARP.
Gli indirizzi IPv6, composti da 128 bit, sono rappresentati come 8 gruppi di 4 cifre esadecimali (8 word di 16 bit ciascuna) in cui le lettere vengono scritte

in forma minuscola. Ad esempio fe80:0000:0000:0000:a4f5:f1ff:fe49:8d93 rappresenta un indirizzo IPv6 valido. In un indirizzo IPv6 una sequenza di zeri contigui composta da 2 o più gruppi può essere contratta con la semplice sequenza *::* riducendo così l'indirizzo appena illustrato a fe80::a4f5:f1ff:fe49:8d93.

Datagram IPv6 IPv6 specifica un nuovo header rispetto a IPv4 semplificandolo di molto eliminando alcuni campi poco usati e spostandoli in estensioni separate. Come già detto l'header IPv6 ha una dimensione fissa di 40 byte.

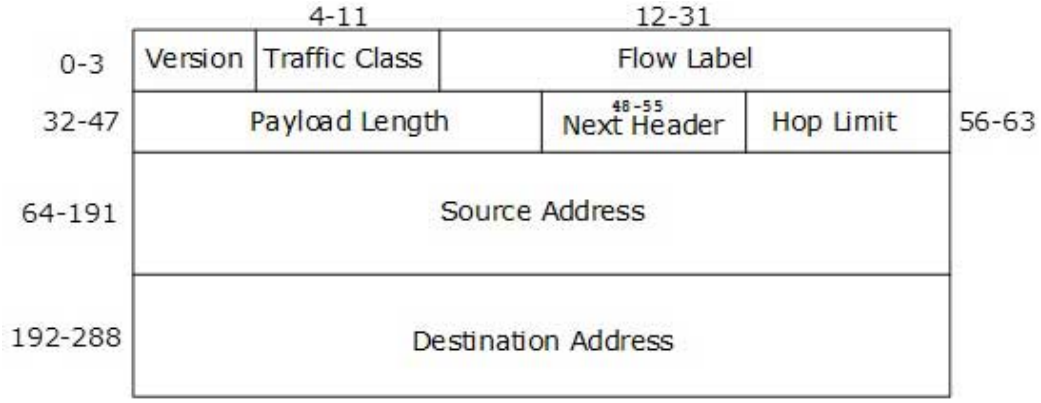

Figura 2.6: Header datagram IPv6

In particolare eventuali estensioni vengono inviate successivamente: il campo *next header* infatti definisce il tipo dell'header successivo nel caso siano inviate delle estensione altrimenti indica il protocollo del pacchetto incapsulato nel payload del datagram. È quindi possibile avere più di un estensione per ciascun datagram. Alcune tra le estensioni più comuni sono:

- **Hop-by-hop options** definisce un insieme arbitrario di opzioni intese per ogni hop attraversato.
- **Routing packet** definisce un metodo per permettere al mittente di specificare la rotta da segure per un datagramma.
- **Fragment packet** usato quando un datagramma viene frammentato.

- **No next header** indica che i dati nel payload del datagramma non sono incapsulati in nessun altro protocollo.
- **Destination options** definisce un insieme arbitrario di opzioni di interesse del solo destinatario.
- **Mobility options** usato per Mobile IPv6.
- **Altri protocolli (TCP, UDP, etc etc.)** indica il protocollo del pacchetto incapsulato nel payload.

2.6 Transport Layer

Il livello trasporto è il quarto livello dello stack di rete ISO/OSI che ha lo scopo di fornire un canale di comunicazione *end-to-end* tra processi residenti in host diversi.
Uno dei diversi protocolli di livello trasporto è il TCP (Transmission Control Protocol). TCP è un protocollo connection-oriented tramite il quale un flusso di dati inviato viene recapitato al destinatario senza errori. I frammenti dello stream vengono impacchettati e passati ai layer inferiori, sarà poi compito del ricevente provvedere a riassemblare ciascun pacchetto ed eventualmente, in caso di errori, chiederne il rinvio. TCP implementa inoltre un meccanismo per il *flow control*, non permette cioè ad una sorgente "veloce" di congestionare un ricevente "lento", assieme al *congestion control* che consente di evitare di congestionare i router che sono tra i due end-point.
Considerato lo scenario e il contesto di questa tesi il protocollo di livello trasporto che più è degno di approfondimento è il protocollo UDP.

Protocollo UDP Il protocollo UDP (User Datagram Protocol, RFC 768 INSERIRE NOTEEEE) è un protocollo di livello trasporto connection-less e best-effort cioè non assicura che un pacchetto sia effettivamente consegnato a destinazione. Applicazioni in esecuzione su host differenti possono scambiarsi messaggi, detti datagram, consegnandoli su appositi socket dedicati.

Essendo un protocollo inaffidabile una volta inviato un datagram verso la rete l'applicazione sorgente non potrà mai sapere se effettivamente è stato consegnato a destinazione o meno. Ciascun datagram inviato è indipendente dai restanti datagram inviati ed inoltre non viene effettuata alcun tipo di frammentazione a livello trasporto. Non viene effettuato alcun tipo di controllo nemmeno per quanto riguarda l'ordine con cui i pacchetti vengono ricevuti dall'end-system. Tutti questi motivi UDP è un protocollo molto leggero e veloce ideale per applicazioni che necessitano di interattività così come ad esempio servizi di streaming audio/video in real-time.
L'header UDP, visti i pochi controlli di cui si fa carico il protocollo, risulta essere molto snello introducendo così un overhead molto basso.

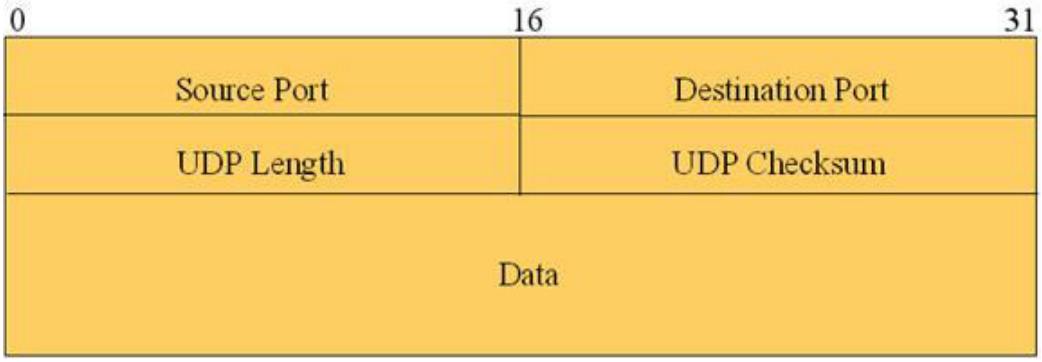

Figura 2.7: Header UDP

Nell'header UDP, oltre alle porte che indicano sostanzialmente le applicazioni (sorgente e destinataria) coinvolte nelle comunicazione, vi è il campo checksum. Questo campo è opzionale se la trasmissione avviene over IPv4 mentre obbligatorio se si utilizza IPv6 (header IPv6 non contiene campo sull'integrità datagram).

2.7 Application Layer

Il livello applicazioni si colloca nella parte più alta dello stack ISO/OSI e coinvolge tutta una serie di protocolli che si occupano di fornire servizi per i

processi delle applicazioni usate dagli utenti finali. I protocolli appartenenti a questo livello sono davvero numerosissimi e utilizzati dalle più disparate applicazioni come ad esempio FTP (File Transfer Protocol) o HTTP (HyperText Transfer Protocol) alla base del World Wide Web.
Se considerato il livello Application dello standard *de jure* TCP/IP questo include anche i livelli presentazione e sessione del modello OSI. Per quanto riguarda il livello applicazione si vuole ora approfondire i protocolli coinvolti nella tecnologia VoIP in quanto la applicazioni che sfruttano questa tecnologia possono toccare con mano l'oggetto di questa tesi.

2.7.1 VoIP

VoIP (Voice over IP) è una famiglia di tecnologie internet e protocolli di comunicazione per la distribuzione di comunicazioni vocali e sessioni multimediali attraverso il protocollo IP.
Il VoIP richiede due tipologie di protocolli di comunicazione in parallelo, una per il trasporto dei dati (pacchetti voce su IP), ed una per la segnalazione della conversazione che comprende la ricostruzione del frame audio, la sincronizzazione, l'identificazione del chiamante lo scambio di altri parametri di comunicazione così come indirizzp IP, porte e codec audio/video.
Per quanto riguarda la trasmissione dati la maggioranza delle implementazioni VoIP adottano il protocollo RTP (Real-time Transfer Protocol, basato su UDP). RTP viene usato assieme a RTPC (RTP Control Protocol) che monitora la *qualità del servizio* (QoS) inviando periodicamente delle statistiche ai partecipanti dello streaming multimediale senza però trasportare alcun tipo di dato relativo alla comunicazione vera e propria. È stato inoltre definito SRTP (Secure Real-time Transfer Protocol) che introduce crittografia, autenticazione e integrità in RTP.
Per quanto concerne i protocolli di segnalazione, invece, nel corso degli anni ne sono stati sviluppati diversi. H.323 definito dalla ITU (International Telecommunications Union) è stato uno dei primi protocolli pensati per il VoIP. H.323 nasce in ambito telefonico e delinea un'architettura completa

per lo svolgimento di conferenze multimediali. Comprende la definizione dei formati di codifica a livello applicativo, la definizione dei formati per la segnalazione e il controllo, per il trasporto dei flussi multimediali (audio, video e dati). H.323 definisce anche alcuni meccanismi legati agli aspetti relativi alla sicurezza.
Vi è poi SIP (Session Initiation Protocol) definito da IETF, operante a livello sessione del modello OSI, che gestisce in modo generale una sessione di comunicazione tra due o più entità, ovvero fornisce meccanismi per instaurare, modificare e terminare (rilasciare) una sessione. Fornisce funzionalità di instaurazione e terminazione di una sessione, operazioni di segnalazione , tono di chiamata, chiamata in attesa, identificazione del chiamante e molto altro ancora. Adotta un un pattern message/response e i messaggi principali di SIP sono:

- REGISTER, inviato da un client verso un server SIP che mantiene una mappa degli utenti SIP e la loro posizione (indirizzi IP).
- INVITE, utilizzato per inizializzare una sessione di comunicazione specificando l'identificativo dell'utente con la quale si vuole comunicare; inviato da un client verso un server SIP che risponde con l'indirizzo dell'altro end-node con la quale si vuole avviare la sessione assieme ad altri parametri di configurazione.
- re-INVITE, viene utilizzato quando uno dei parametri di configurazione cambia (ad esempio l'indirizzo IP).

SIP è più recente e viene largamente utilizzando riscontrando un maggiore successo di H.323.

Capitolo 3

Scenario

Tutti i dispositivi mobili attualmente offrono diverse modalità di accesso ad internet basti pensare a uno smartphone che consente l'accesso a internet sia in modalità Wi-Fi che attraverso la telefonia mobile sfruttando tecnologie come il 3G fino al più recente 4G.
In questo capitolo si vuole descrivere un'architettura che consente a un user app che fornisce un servizio multimediale in esecuzione su di un mobile device *multi-homed*, ovvero che monta più di un'interfaccia di rete, di sfruttare tutte le sue NIC (Network Interface Card) in un contesto di mobilità. È interessante studiare uno scenario che fa uso di un'app multimediale (come può essere ad esempio un'applicazione per chiamate VoIP) in quanto sono app *real-time* e generano una grande mole di traffico. Le app appartenenti alle categorie dei servizi multimedial real-time vengono solitamente realizzate sfruttando il protocollo UDP: l'architettura presentata in questo capitolo fornisce un meccanismo in grado da rendere possibile l'inoltro di ciascun datagram UDP attraverso l'interfaccia di rete più adatta e disponibile al momento della trasmissione. Questa architettura è detta ABPS (Always Best Packet Switching) ma prima di descriverne i suoi principali aspetti verrà descritto lo stato dell'arte per quel che riguarda la gestione della mobilità dei nodi mobili.

3.1 Seamless Host Mobility & State Of the Art

Nel corso degli ultimi anni sono state sviluppate diverse architetture che consentono a un nodo mobile in movimento di avere accesso continuo a servizi di rete in particolare sono stati sviluppati diversi approcci che cercano di far si che device che montano più interfacce di rete possano effettuare un *handover* da un interfaccia di rete all'altra in maniera del tutto trasparente all'app utente in esecuzione, ovvero in maniera *seamless*. Per handover o handoff si intende il processo di cambio di interfaccia di telecomunicazione da parte di un dispositivo multi-homed (*vertical handoff*) oppure il cambio di punto di accesso mantenendo la stessa tecnologia di telecomunicazione (*horizontal handoff*, ad esempio cambio di AP all'interno di una stessa rete WLAN).
In generale una buona architettura per la seamless mobility dovrebbe essere responsabile di identificare univocamente ciascun nodo mobile permettendogli di essere raggiungibile dall'altro nodo coinvolto nella comunicazione (Correspondent Node che potrebbe essere anch'esso un nodo mobile) e dovrebbe, inoltre, monitorare la QoS fornita dalle diverse reti a cui il nodo mobile potrebbe connettersi in modo da prevedere la necessità di un handoff ed eventualmente eseguirlo in maniera *seamless* assicurando la piena continuità della comunicazione.
Vediamo ora una rassegna di tutte le soluzioni sviluppate finora per implementare meccanismi di seamless handoff in un contesto di un nodo mobile che attraversa reti eterogenee.

Implementazioni a livello network Tra le architetture presenti che lavorano a livello network vi è Mobile IP version 6 e le sue ottimizzazioni come ad esempio FMIP (Fast Handover Mobile IPv6), HMIP (Hierarchical Mobile IPv6) e PMIP (Proxy Mobile IPv6). Tutte queste architetture adottano un *Home Agent* ovvero un'entità aggiuntiva che opera all'interno della rete alla quale il nodo mobile appartiene. L'home agent ricopre il ruolo

di *location registry* ovvero un servizio *always on*: quando un nodo mobile cambia interfaccia di rete e quindi indirizzo IP lo comunica al location registry (*registration phase*) che tiene una mappa degli indirizzi. Quando un Correspondent Node vuole comunicare con il nodo mobile invia al location registry una *lookup phase* per ottenere l'indirizzo corrente del mobile node. Tutti i nodi coinvolti devono avere il supporto a IPv6: in particolare l'indirizzo attuale e l'identificativo univoco del nodo mobile sono trasmessi attraverso delle estensioni di IPv6. Il fatto che tutti i nodi debbano necessariamente supportare IPv6 rappresenta un limite di questo approccio architetturale. Un altro limite di questo approccio è che presso un home agent può essere registrato l'indirizzo di una sola interfaccia di rete per ogni nodo impedendo così un supporto al multihoming visto che la latenza introdotta dai numerosi messaggi di autenticazione procurerebbe un overhead insostenibile per la tipologia di comunicazione multimediale che dovrebbe essere veloce e snella.

Implementazioni tra livello rete e trasporto Esistono alcune possibili implementazioni di architetture per la seamless host mobility che introducono un nuovo layer posto tra il livello rete e quello trasporto: questa nuova astrazione dovrà essere aggiunta su tutti i nodi prendenti parte alla comunicazione. Alcuni esempi possono essere HIP (Host Identity Protocol) e LIN6 (Location Independet Addressing for IPv6). Il location registry si comporta in maniera simile a un server DNS mappando l'identificativo di un host e la sua attuale posizione restando però all'esterno della rete di appartenenza del nodo mobile. Un limite di questo approccio è nella necessità di modificare lo stack di rete di tutti i nodi coinvolti.

Implementazioni a livello trasporto Vi sono alcuni protocolli che operano a livello trasporto: ogni nodo coinvolto in una comunicazione si comporta in maniera pro-attiva fungendo da location registry informando direttamente il proprio Correspondent Node ogni qualvolta la configurazione di rete cambia. Il limite di questo approccio sta nel fatto che se entrambi i mobile end-system cambiano contemporaneamente gli indirizzi IP a seguito

di un handoff diventano mutuamente irraggiungibili. Inoltre, come nel precedente approccio, questo tipo di architettura richiederebbe una modifica delle applicazioni sia sul nodo mobile che sul suo correspondent node.

Implementazioni a livello Sessione Sono state progettato alcune soluzioni che operano a livello sessione come ad esempio TMSP (Terminal Mobility Support Protocol) che sfrutta un SIP server ausiliario collocato fuori dalla rete di un nodo mobile che funge da location registry che mappa ciascun identificativo SIP di utente al suo indirizzo IP attuale. Ogni nodo mobile esegue un client SIP che manda un messaggio di tipo REGISTER per aggiornare il suo indirizzo IP. I messaggi INVITE, al solito, sono utilizzati per avviare comunicazioni con altri nodi così come i messaggi di tipo re-INVITE. Gli approcci operanti a livello Session non sembrano essere particolarmente efficienti per i ritardi introdotti dal pattern message/response dei sistemi basati SIP. In letteratura vi sono altre soluzioni alcune parziali come IEEE 802.11e e IEEE 802.11r che però coinvolgono solamente la gestione dell'interfaccia di rete Wi-Fi (handover orizzontali) e soluzioni proprietarie come LISP di CISCO.

3.2 Always Best Packet Switching

Un'architettura progettata all'interno del Dipartimento di Informatica dell'Università di Bologna che supera molti dei limiti delle implementazioni precedentemente descritte è ABPS (Always Best Packet Switching). L'architettura ABPS è composta da due componenti principale:

- **fixed proxy server**, una macchina esterna alla rete in cui si trova il mobile node; munito di IP pubblico statico e fuori da qualsiasi firewall o NAT. Il fixed proxy server gestisce e mantiene tutte le comunicazione da un mobile node verso l'esterno e viceversa: nel caso di un handoff e quindi di una riconfigurazione delle interfacce di rete del nodo mobile il fixed proxy server nasconde questi cambiamenti al correspondent

node facendo si che la comunicazione continui in maniera del tutto trasparente.

- **proxy client**, in esecuzione su ogni mobile node, mantiene per ogni NIC una connessione verso il fixed proxy server. Applicazioni in esecuzione su un nodo mobile possono quindi sfruttare un multi-path virtuale creato tra il proxy client e il fixed proxy server per comunicare con il resto del mondo.

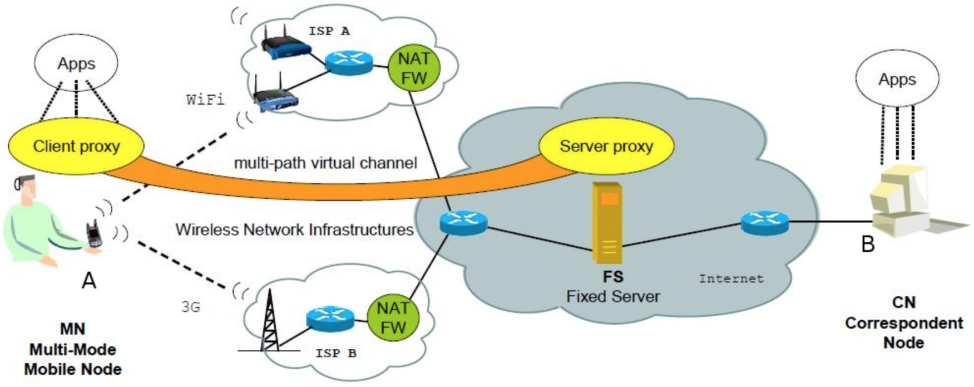

Figura 3.1: Architettura ABPS

Si immagini uno scenario in cui un nodo mobile A munito di più interfacce wireless (ad esempio NIC Wi-Fi e UMTS) stia intrattenendo una comunicazione VoIP con un altro nodo mobile B. Il nodo A implementa il meccanismo ABPS appena descritto quindi sul dispositivo è in esecuzione un ABPS proxy client che mantiene un canale di comunicazione con un ABPS fixed proxy. Supponiamo che il nodo A sta utilizzando l'interfaccia di rete Wi-Fi e quindi risulta essere connesso a una rete WLAN. Se avviene un handoff verso un'altra interfaccia di rete, ad esempio UMTS, per un calo delle prestazioni o per una perdita improvvisa del segnale dell'access point a cui è connesso il nodo mobile il cambio di configurazione di rete avviene in maniera del tutto trasparente al nodo B e alle applicazioni in esecuzione sul

nodo A. Il meccanismo descritto può inoltre decidere su quale interfaccia di rete inoltrare il singolo datagram UDP a seconda delle condizioni della rete a cui ciascuna NIC è connessa. In particolare vi è un monitoraggio del QoS per ciascuna interfaccia di rete wireless e se vi è il sospetto di una perdita di informazioni o di un ritardo di trasmissione un dato pacchetto può essere ritrasmesso su un'altra NIC. Ciascuna interfaccia di rete rimane configurata e attiva e pronta a essere utilizzata nel caso in cui l'interfaccia attualmente in uso abbia dei ritardi o delle perdite.

Qui di seguito viene illustrato un esempio di una possibile implementazione dell'architettura ABPS per il mantenimento di un servizio VoIP basato sui protocolli SIP e RTP/RTCP.
Come si può vedere dalla figura **??** il nodo mobile oltre alla user app VoIP mantiene attivo un client proxy: i pacchetti trasmessi dall'app vengono intercettati dal proxy client mantenendo attiva un'interfaccia di rete virtuale settata come default gateway. Per ogni interfaccia di rete reale il proxy client inizializza e mantiene attivo un socket per ogni protocollo di comunicazione e segnalazione.

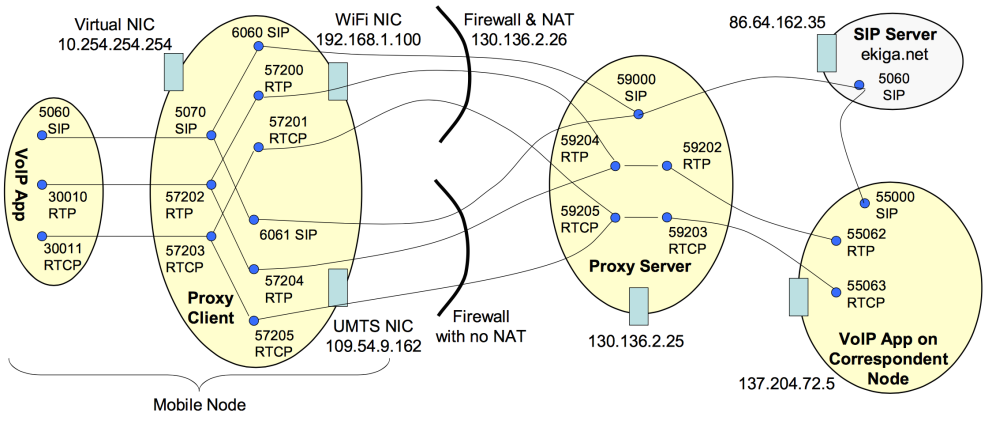

Figura 3.2: Architettura ABPS per una comunicazione VoIP via protocollo SIP e RTP/RTCP

Il fixed proxy server regola il traffico VoIP in invio o ricezione dal mobile node esponendo verso un SIP server (ad esempio ekiga.net) e il correspondent node un indirizzo IP pubblico e statico e delle porte associate a ciascun protocollo VoIP. La comunicazione tra il client proxy e il fixed server proxy può fare uso di un'estensione del protocollo SIP per lo scambio di ulteriori parametri di configurazione (ad esempio identificativo per individuare ciascun proxy client che comunica con ABPS proxy server).

3.2.1 Architettura

Vediamo ora brevemente le caratteristiche principali delle componenti dell'architettura Always Best Packet Switching.

Fixed proxy server Sul fixed proxy server è presente un modulo software chiamato *Policies Module*. Il compito del Policies Module, molto semplicemente, è quello di valutare su quale interfaccia di rete inoltrare un messaggio diretto verso un mobile node: verrà utilizzato l'indirizzo IP mittente dell'ultimo pacchetto ricevuto dal proxy client.

Mobile node L'architettura per supportare ABPS su un nodo mobile risulta essere piuttosto complessa. È composta da:

- **Monitor**, il suo compito principale è quello di monitorare e configurare le diverse interfacce di rete wireless presenti sul nodo mobile. Ogniqualvolta una NIC viene configurata o disabilitata il Monitor invia una notifica al proxy client di tipo *Reconfiguration Notification*.

- **TED (Transmission Error Detector)**, è il componente più importante del sistema; si occupa di monitorare l'invio dei datagram UDP trasmessi dall'app VoIP e di notificare al client proxy se il pacchetto è stato consegnato o meno all'access point. Una volta inviato un certo pacchetto, attraverso la scheda di rete Wi-Fi, TED valuterà il relativo

ACK proveniente dall'access point e tramite la notifica *First-hop Transmission Notification* notificherà al proxy client lo status di consegna di quel pacchetto. TED è implementato in maniera cross-layer nello stack di rete del kernel Linux. TED e la sua implementazione saranno largamente descritti nel capitolo successivo.

- **Wvdial** si tratta di un modulo che implementa Transmission Error Detector per UMTS.

- **Proxy client**, il cui ruolo principale è quello di inoltrare il traffico di una user app verso il ABPS fixed proxy server.
  Il proxy client al suo interno implementa il modulo *ABPS Policies* che implementa una serie di politiche e meccanismi in base alle notifiche provenienti dall'altre componenti in esecuzione sul nodo mobile e dirette verso il proxy client. Le tipologie di notifiche che possono essere ricevute sono quelle precedentemente accennate. Quando il proxy client riceve una notifica di tipo Reconfiguration Notification per ogni nuova interfaccia di rete segnalata come attiva viene creato un nuovo socket e associato a tale interfaccia; viceversa quando un'interfaccia viene segnalata come disabilitata (a seguito ad esempio di un errore di trasmissione) il relativo socket associato viene chiuso.
  Un'altro tipo di notifica che un proxy client può ricevere è proveniente dal TED: in questo l'ULB (UDP Load Balancer), un altro modulo implementato all'interno del proxy client, valuterà in base alla notifica se ritrasmettere un dato datagram e attraverso quale interfaccia di rete.
  L'ultimo tipo di notifica che un proxy client può ricevere è quella proveniente dal protocollo ICMP: semplicemente questa notifica segnala al proxy client che il fixed proxy server (o la porta sul proxy server con la quale si vuole comunicare) è *unreachable* attraverso l'interfaccia di rete attualmente in uso.
  Queste tre tipologie di notifiche permettono al proxy client di stabilire se un determinato pacchetto debba essere ritrasmesso (eventualmen-

te attraverso un?altra interfaccia wireless) o definitivamente scartato. Permettono inoltre di realizzare un opportuno algoritmo per la selezione dinamica dell?interfaccia di rete che offre maggiori garanzie di trasmissione.

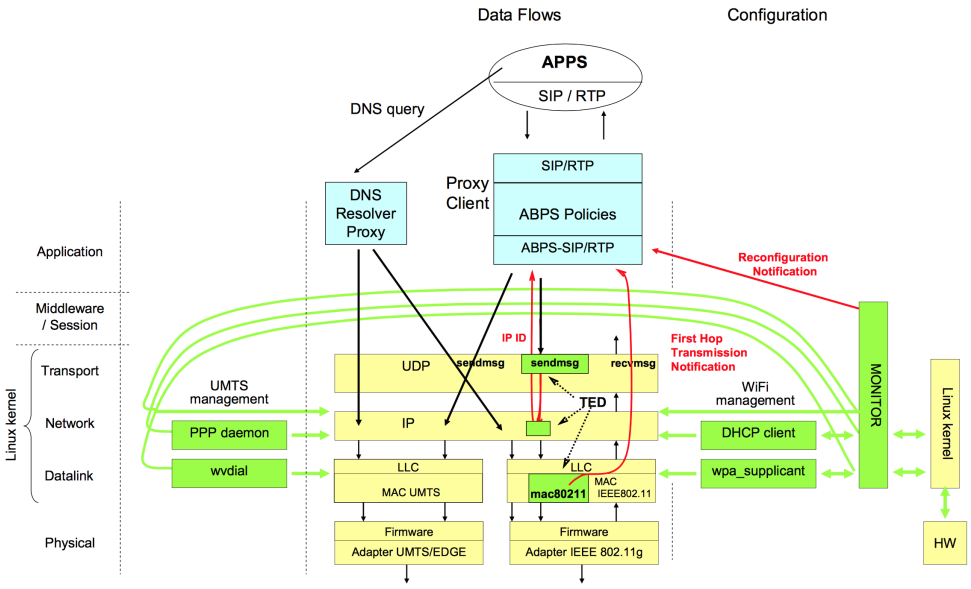

Figura 3.3: Infrastruttura Mobile Node in ABPS

3.3 Considerazioni finali

In questo capitolo abbiamo visto una breve panoramica di Always Best Packet Switching e delle sue funzionalità di base. Abbiamo visto come senza modificare l'infrastruttura di rete è possibile inoltrare il flusso di dati proveniente da un certa applicazione utente su una piuttosto che su un'altra interfaccia di rete di un nodo mobile a seconda dello stato in cui si trova la rete.

Nel caso di una riconfigurazione di rete ABPS abbatte eventuali overhead introdotti da approcci come quello basato su SIP e può far fronte a handoff verticali senza alcun ritardo in quanto, come visto, ciascuna interfaccia di

rete viene mantenuta attiva e pronta all'utilizzo. L'utilizzo di ciascuna NIC è ottimizzato valutando pacchetto per pacchetto su quale interfaccia di rete questo dovrebbe essere inoltrato valutando le condizioni della rete e le QoS desiderate. Inoltre ABPS consente a un nodo mobile di comunicare verso l'esterno anche in presenza di NAT o firewall.
ABPS può essere utilizzato in qualsiasi contesto VoIP.

Conclusioni

Queste sono le conclusioni.
In queste conclusioni voglio fare un riferimento alla bibliografia: questo è il mio riferimento [**?**, **?**].

Appendice A

Prima Appendice

In questa Appendice non si è utilizzato il comando: `\clearpage{\pagestyle{empty}\cleardoublepage}`, ed infatti l'ultima pagina 8 ha l'intestazione con il numero di pagina in alto.

Appendice B

Seconda Appendice

Bibliografia

[1] Primo oggetto bibliografia.

[2] Secondo oggetto bibliografia.

[3] Terzo oggetto bibliografia.

[4] Quarto oggetto bibliografia.

Ringraziamenti

Qui possiamo ringraziare il mondo intero!!!!!!!!!!
Ovviamente solo se uno vuole, non è obbligatorio.