Alma Mater Studiorum · Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea Triennale in Informatica

SIMULAZIONE DI TECNOLOGIE WIRELESS A LUNGO RAGGIO SU OMNET++

Tesi di Laurea in Architettura degli Elaboratori

Relatore:
Chiar.mo Prof.
Vittorio Ghini

Presentata da:
Enrico Succi

Sessione II
Anno Accademico 2009/2010

Introduzione

Negli ultimi anni, grazie alla sempre più rapida evoluzione e diffusione di dispositivi mobili, come *smartphone* e *netbook*, insieme alla ormai sempre maggiore diffusione di punti d'accesso (AP) a reti **wireless** pubbliche nei centri urbani, nonché l'introduzione, spesso sperimentale, di reti WiMAX nel territorio, si è resa realistica l'ipotesi di un passaggio verso sistemi di connettività per dispositivi mobili che sfruttino sia tecnologie **WiFi** (IEEE 802.11a/b/g/n) che Mobile WiMAX (IEEE 802.16e) . In particolare, risulta interessante la possibilità di utilizzare applicazioni **Voice over Internet Protocol** (VoIP) su questo tipo di reti, in contesti di mobilità su scala urbana. Allo scopo di mantenere un livello di **Quality of Service** (QoS) accettabile per applicazioni di telefonia in questo tipo di scenari, è stato realizzato il meccanismo **Robust Wireless Medium Access** (RWMA) nel kernel del sistema operativo open-source GNU/Linux.

Come per molte nuove tecnologie, anche per il meccanismo RWMA, sono richiesti dei test, spesso troppo complessi o costosi per essere eseguiti con dispositivi reali, per verificarne le condizioni di applicabilità, le prestazioni e l'affidabilità, per di più se si vogliono usare infrastrutture di rete differenti, come WiFi e WiMAX; per questo motivo si procede a sviluppare una simulazione del meccanismo, una versione virtuale, che tenga conto dei fattori rilevanti per quanto riguarda gli indici di prestazione che si è scelto di esaminare.

Fine ultimo è quello di costruire uno scenario di simulazione che riproduca il comportamento di una serie di tecnologie wireless in un contesto cittadino, nel quale si intende valutare le prestazione di un sistema di gestione di interfacce multiple dedicato al voice over ip (VoIP). Nello specifico, in ques-

ta tesi si è cercato di aggiungere allo strumento di simulazione l'utilizzo di una rete geografica di tipo WiMAX. Introdurre questa rete ha comportato la soluzione di svariati problemi, a partire dalla gestione di stack differenti IPv4 e IPv6, fino alla gestione della connessione fra le due reti.

Il contributo del lavoro è stato quindi quello di studiare quali fossero gli strumenti di simulazione disponibili per WiMAX, valutare come questi potessero essere integrati all'interno dello scenario simulativo fornitoci e definire l'impostazione di massima dell'architettura delle due diverse reti. Il tutto è stato svolto in collaborazione con il tesista Riccardo Ravaioli, con il quale è stata effettuata la fase di implementazione e di testing. L'autore della tesi, in particolare, si è occupato anche della gestione del dual stack all'interno dei nodi interessati.

Il resto del documento di tesi è strutturato nel seguente modo: nel capitolo 1 si descrive lo scenario in cui ci si è collocati, nel capitolo 2 si tracciano gli obiettivi di tesi, nel capitolo 3 si descrivono gli strumenti a disposizione, nel capitolo 4 si illustrano le scelte di progettazione, nel capitolo 5 si mettono in evidenza alcune note implementative, nel capitolo 6 si presentano istruzioni per la compilazione, risultati di test e valutazioni sul lavoro svolto e infine al capitolo 7 spetta il ruolo di conclusione del documento.

Indice

1 Scenario **9**

1.1 Infrastruttura 9

1.1.1 Wireless Access Point 10

1.1.2 Terminologia 10

1.1.3 Schede di rete wireless 11

1.2 Il protocollo IEEE 802.11 12

1.2.1 IEEE 802.11 b/g 12

1.2.2 I frame 802.11 13

1.2.3 I frame di gestione 15

1.2.4 I frame di controllo 16

1.3 Protocolli 17

1.3.1 Voice Over Internet Protocol 17

1.3.2 Realtime Transport Protocol 18

1.3.3 Session Initiation Protocol 18

1.3.4 User Datagram Protocol 19

1.3.5 Internet Protocol 21

2 Obiettivi **25**

2.1 Protocollo WiMAX (802.16e) 26

2.1.1 Infrastruttura WiMAX 27

2.1.2 Mobile WiMax 28

2.2 Obiettivi della tesi 30

2.3 Scelte implementative 30

2.4 Ambiente simulativo 31

3 Framework **33**
3.1 OMNeT++ . . . 33
3.1.1 Moduli . . . 34
3.1.2 Gate . . . 35
3.1.3 Messaggi . . . 35
3.1.4 Comunicazione tra livelli di protocollo . . . 36
3.1.5 Il linguaggio NED . . . 36
3.1.6 I file .ini . . . 37
3.1.7 OMNEST . . . 38
3.2 INET . . . 38
3.2.1 Protocolli . . . 38
3.2.2 Moduli comuni . . . 39
3.2.3 Architettura di una NIC IEEE 802.11 . . . 40
3.2.4 I file .irt . . . 41
3.2.5 Le simulazioni . . . 43
3.3 Numbat[15] . . . 43
3.3.1 Principi del progetto . . . 44
3.3.2 Architettura di Subscriber e Base Station . . . 45
3.3.3 Mobility model . . . 47

4 Progettazione **49**
4.1 Connessione fra reti IPv4 e IPv6 . . . 49
4.2 Dual Stack . . . 51
4.3 Gestione DHCP . . . 52
4.4 File . . . 54

5 Note implementative **55**
5.1 Integrazione di Numbat in Inet . . . 55
5.2 Gestione di src e dest address . . . 56
5.3 Gestione del movimento . . . 57

6 Test e Valutazioni **59**
6.1 Compilare il codice . . . 59
6.2 Simulazioni . . . 60

6.2.1 La simulazione NetworkCity 61
6.2.2 Alcuni risultati 62
6.3 Valutazione e sviluppi futuri 63
6.3.1 IPv6 di Inet 64
6.3.2 Restyling dei componenti 64
6.3.3 Divisione fra proxy server e secondo end point 65
6.3.4 Protocollo DHCP 65
6.3.5 Politica di scelta dell'interfaccia 66
6.3.6 Modalità di esecuzione "run" 66

7 Conclusioni **67**

Elenco delle figure

2.1 Possibile scenario di utilizzo 25
2.2 Architettura base scenario WiMax 27

3.1 Architettura di una WNIC in INET 40
3.2 Numbat 44

4.1 Dual Stack 50
4.2 Struttura del Mobile Host e del Proxy Server 52

6.1 Menù di scelta della simulazione 60
6.2 Latenza dei pacchetti ricevuti dal nodo mobile sulla connessione WiMax 63
6.3 Latenza dei pacchetti ricevuti dal Proxy Server sulla connessione WiMax 63

Capitolo 1

Scenario

In un contesto moderno quale quello di una grande o media città, è ormai assumibile come dato di fatto che in un dato punto siano presenti una o più reti wireless. Anche se la grande maggioranza di queste sono private e presumibilmente protette, è ormai d'uso comune da parte di comuni o luoghi di ritrovo pubblici (quali alberghi, aeroporti, parchi, biblioteche, ecc), mettere a disposizione reti aperte al pubblico.

L'accesso a queste reti **WLAN**[1] è possibile grazie a qualsiasi dispositivo che sfrutti la tecnologia wireless. La più diffusa è quella basata su specifiche **IEEE 802.11**, comunemente conosciuta come **WI-FI**[2].

1.1 Infrastruttura

Una delle più comuni modalità di installazione, è chiamata **Infrastructure Basic Service Set**[3], e consiste in un insieme di **nodi** (Station e Wireless Access Point) che sono utilizzati per comunicare all'interno di una stessa BSS.

[1]Wireless Local Area Network (lett: Rete locale senza fili). Il termine solitamente indica una qualsiasi rete di dispositivi che non utilizzano dei collegamenti via cavo per connettesi alla rete.

[2]Wi-Fi, abbreviazione di Wireless Fidelity. Termine che indica dispositivi che possono collegarsi a reti locali senza fili (WLAN) basate sulle specifiche IEEE 802.11

[3]Basic Service Set (BSS) è un termine usato per descrivere una collezione di dispositivi che comunicano all'interno di WLAN (può non includere Wireless Access Point). Ne esistono di due tipi: Independent Basic Service Set e Infrastructure Basic Service Set.

Solitamente il processo attraverso il quale le station comunicano tra loro o con l'esterno è piuttosto semplice e consiste nella station che invia il suo flusso dati ad un AP, che poi si occupa di inoltrarlo al dispositivo di destinazione (all'interno della rete o all'esterno).

La tesi userà **questo tipo** di infrastruttura nelle proprie simulazioni.

Una WLAN Infrastructure Basic Service Set invece, è una rete wireless (definita in modalità Ad-Hoc) che rende possibile collegare in modo indipendente più postazioni wireless tra loro senza nessun dispositivo centrale che funga da tramite. Il sistema IBSS non è adatto ad una rete numerosa e concentrata, a causa della sovrapposizione dei segnali ed i problemi che ne seguono.

1.1.1 Wireless Access Point

Un **Wireless Access Point** (abbreviato in WAP o AP) è un dispositivo, facente parte di una rete IEEE 802.11, che si presenta come un **punto di interconnessione** tra una station ed una rete, solitamente (ma non necessariamente) su cavo, quale ad esempio Ethernet.

Può essere costituito da un qualsiasi dispositivo wireless opportunamente configurato oppure da uno appositamente dedicato.

Gli AP moderni possono anche essere configurati per essere usati come:

1. Router (Instradatore): apparato che esegue l'interconnessione di reti locali multiprotocollo. Gestisce l'instradamento dei messaggi attraverso due reti.

2. Bridge (Ponte): dispositivo che interconnette due reti locali eterogenee, o ne suddivide una in più sottoreti interconnesse (viste all'esterno come una sola).

3. Client

1.1.2 Terminologia

Hotspot Nel caso gli Access Point siano pubblici, vengono definiti hotspot.

BSA La Basic Service Area è l'area teorica all'interno della quale ciascun membro di una BSS è in grado di comunicare.

Station Station (spesso abbreviato come STA) è un termine che indica qualsiasi dispositivo contenente un MAC (Medium Access Control) conforme allo standard IEEE 802.11 ed una interfaccia a livello fisico adatta a comunicare su una rete wireless. Solitamente all'interno di un contesto wireless i termini client wireless, station e nodo sono intercambiabili.

1.1.3 Schede di rete wireless

Una Wireless Network Interface Controller (WNIC) è una scheda di rete capace di connettersi ad una rete su mezzo radio[4].

Lavora sui livelli OSI[5] 1 e 2 ed è costruita secondo lo standard IEEE 802.11.

Alcuni parametri tipici di una scheda wireless sono:

- La banda (in Mb/s): da 2 Mbit/s a 54 Mbit/s
- La potenza di trasmissione (in dBm)
- Gli standard supportati (es: 802.11 b/g/n, ecc.)

Una scheda wireless può operare in due modalità:

1. Infrastructure
2. Ad-hoc

Nella prima modalità, la scheda ha bisogno di un AP come intermediario. Tutti i dati (di tutte le station) sono trasferiti usandolo come nodo di interconnessione. Per connettersi ad SSID[6] protette, tutte le station devono essere a conoscenza della chiave.

[4]Usa un'antenna per comunicare attraverso microonde.

[5]Open Systems Interconnection Reference Model è una descrizione astratta a livelli per l'organizzazione delle reti di comunicazioni.

[6]Il Service Set Identifier consiste in una serie di caratteri ASCII e rappresenta il nome della rete WLAN.

Nella modalità Ad Hoc invece gli AP non sono richiesti, essendo le WNIC in grado di comunicare direttamente con le altre station. Tutti i nodi devono comunque essere sullo stesso canale. L'architettura implementata nella tesi richiede che la scheda operi secondo la prima modalità.

1.2 Il protocollo IEEE 802.11

Sviluppato dal gruppo 11 dello IEEE 802 LAN/MAN Standards Committee (LMSC), il protocollo IEEE 802.11 definisce uno standard per le reti WLAN sulle frequenze 2.4, 3.6 e 5 GHz.

Tra i protocolli definiti nelle specifiche, il primo ad essere stato largamente utilizzato fu il b, seguito dal g ed infine dallo n. La tesi tratterà questi protocolli solo per quanto riguarda le sezioni pertinenti al lavoro svolto, senza soffermarsi su altri particolari comunque fondamentali dello standard, ma di interesse marginale per capire il lavoro svolto.

1.2.1 IEEE 802.11 b/g

Come anticipato, i protocolli b e g sono tra i più diffusi a livello civile, ed entrambi utilizzano lo spettro di frequenze sui 2,4 Ghz.

E' importante specificare che i range dei dispositivi, come quelli che verranno segnalati in seguito, possono variare in base all'ambiente in cui si trovano. Metallo, acqua e in generale ostacoli solidi riducono drasticamente la portata del segnale. Inoltre sono suscettibili a interferenze di altri apparecchi che operano sulle stesse frequenze (come forni a microonde o telefoni cordless).

Lo stesso vale per la velocità di trasferimento, i cui valori massimi sono raggiungibili solo vicino alla fonte del segnale.

IEEE 802.11 b Questo protocollo è stato ratificato nell'Ottobre del 1999. Utilizza il CSMA/CA[7] come metodo di trasmissione delle informazioni. Ha

[7] Il Carrier Sense Multiple Access con Collision Avoidance (lett: Accesso multiplo con rilevazione di portante a eliminazione di collisione) è un protocollo di trasmissione che evita le contese, cioè i tentativi di più stazioni di accedere contemporaneamente alla rete.

una capacità massima di 11Mb/s, un throughput[8] di circa 5 Mb/s e una portata massima all'esterno (in assenza di ostacoli) di circa 90-100 metri[1].

IEEE 802.11 g Nel Giugno 2003 è stata ratificato un secondo protocollo, chiamato g. È completamente compatibile con il b, ma le sue prestazioni sono nettamente maggiori. Fornisce una banda teorica di 54Mb/s, un throughput di circa 22 Mb/s e la stessa portata massima[2].

IEEE 802.11 n Lo standard è stato finalizzato l'11 Settembre 2009[**?**] e la sua pubblicazione è prevista per Ottobre. Compatibile con il b, il protocollo promette di essere molto più potente dei suoi predecessori, con un throughput di 144Mb/s ed una banda teorica di 600Mb/s. Come il g ed il b, la portata in spazi aperti prevista è di 90-100 metri.

1.2.2 I frame 802.11

Il protocollo 802.11 definisce come **frame** il tipo di pacchetto utilizzato sia per la trasmissione dei dati che per il controllo del canale di comunicazione. Ne esistono tre tipi:

- Dati (Data Frame): Usati per la trasmissione dei dati
- Gestione (Management Frame): Servono a scambiarsi informazioni
- Controllo (Control Frame): Gestiscono la trasmissione dei dati

Il primo tipo è quello che si occupa del trasporto vero e proprio dei dati. Gli ultimi due invece servono a creare e gestire il canale, oppure si occupano di gestire la trasmissione dei pacchetti Dati (questi tipi di frame non vengono inoltrati oltre il MAC[9] ai livelli superiori).

Per quanto riguarda la struttura di un frame, scenderemo nei particolari solo dei primi sedici bit, ovvero il **campo controllo** (presente in tutti e tre i tipi di frame). Questo è a sua volta suddiviso in undici sotto-campi (quando non specificato si assuma che siano di dimensione 1 bit):

[8] Quantità di dati trasmessa in un determinato intervallo di tempo.
[9] Media Access Control (MAC). E' un sotto-livello del livello Data Link.

1. Versione del Protocollo (2 bit) - Identificano il protocollo usato (es: 802.11g, 802.11b, ecc.)

2. Tipo (2 bit) - Specificano il sottotipo del frame: Gestione, Controllo o Dati

3. Sottotipo (4 bit) - Specificano il tipo del frame: RTS, CTS, ACK, ecc

4. Al DS - Inizializzato a 1 se il frame è diretto al sistema di distribuzione

5. Dal DS[10] - Inizializzato a 1 se il frame proviene dal sistema di distribuzione

6. Altri Frammenti (More frag) - Valorizzato con 1 solo se seguono altri frammenti appartenenti allo stesso datagram

7. Ripetizione (Retry) - Inizializzato a 1 se questo frammento è la ripetizione di un frammento precedentemente trasmesso. Aiuta l'AP nella eliminazione dei frammenti duplicati

8. Risparmio energia (Pwr mgt) - Inizializzato a 1 se al termine del frame l'interfaccia del mittente entrerà nella modalità di basso consumo . Gli AP non configurano mai questo bit

9. Altri dati (More Data) - Inizializzato a 1 se il mittente ha altri frame per il dispositivo

10. WEP – Utilizzato solo se il campo Dati è stato crittografato con l'algoritmo WEP[11]

11. Ordinati (Order) - Se è richiesto il metodo di trasmissione "strict ordering". I frame e i frammenti non sono sempre mandati in ordine perché spesso questo causa rallentamenti nella trasmissione

[10] È complementare di Al DS: uno dei due deve essere necessariamente valorizzato ed esclude l'altro.

[11] Wired Equivalent Privacy. E' un algoritmo ormai non più utilizzato per gestire la confidenzialità nelle reti wireless che implementano il protocollo IEEE 802.11.

Per quanto riguarda la comprensione del lavoro svolto, più che trattare i frame per il trasporto dei dati (DATA), ci concentreremo sui frame di gestione utilizzati per la configurazione, mantenimento e rilascio del canale di comunicazione, e quelli di controllo.

1.2.3 I frame di gestione

Tra i frame di gestione segnaliamo:

Authentication Frame Il processo di autenticazione è il meccanismo attraverso il quale un AP accetta o rigetta l'identità di una WNIC. L'interfaccia inizia il processo mandando un Frame di Autenticazione (Authentication frame) che contiene la sua identità all'AP. In una rete aperta (non criptata), la sequenza richiede solo l'invio di due frame: uno dalla scheda wireless, e uno di risposta (positiva o negativa) dall'AP.

Deauthentication Frame Un dispositivo invia un frame di de-autenticazione (deauthentication frame) a un AP quando desidera terminare una comunicazione. L'AP libera la memoria allocata e rimuove la scheda dalla tabella delle associazioni.

Association Request Frame Una interfaccia inizia il processo di associazione mandando uno di questi frame all'AP. Il frame contiene informazioni sul WNIC e l'SSID della rete a cui desidera associarsi. Se l'AP decide di accettare la richiesta, alloca le risorse per la scheda e manda un Association response frame.

Association Response Frame Contiene la risposta dell'AP ad un Association request frame. Se la risposta è positiva, trasmette anche informazioni aggiuntive (es: Association ID).

Disassociation frame Viene inviato se si vuole terminare una associazione.

Reassociation request frame Se una WNIC si allontana da un AP ed entra nel raggio di uno con un segnale più potente, invia un frame di questo tipo a quest'ultimo. Il nuovo AP coordina l'invio dei dati che possono essere ancora nel buffer del vecchio, e aspetta comunicazioni dal nodo.

Reassociation response frame Simile all'Association Response Frame, contiene la risposta alla richiesta ricevuta e le informazioni aggiuntive per la WNIC.

Beacon frame Un AP manda periodicamente dei beacon frame per annunciare la sua presenza a tutte le schede wireless nel proprio raggio di azione. Trasporta informazioni quali: timestamp, SSID, ecc. Le interfacce cercano continuamente beacon su tutti i canali radio disponibili, con lo scopo di trovare il migliore a cui associarsi.

Probe request frame Viene inviato dalla WNIC quando si richiedono informazioni riguardo gli AP disponibili nel proprio range.

Probe response frame Risposta dell'AP alla richiesta precedente, contiene tutte le informazioni necessarie a portare avanti una connessione.

1.2.4 I frame di controllo

Per ultimi abbiamo i frame di controllo, che si occupano della gestione dello scambio di dati. Ne esistono di tre tipi:

- Acknowledgement Frame (ACK)
- Request to Send Frame (RTS)
- Clear to Send Frame (CTS)

RTS e CTS Questi due tipi di frame implementano un meccanismo per ridurre le collisioni tra AP e station nascoste. Una nodo, prima di mandare i dati, invia un frame RTS come primo passo di un handshake. Una station

risponde a un frame RTS con un frame CTS dando il permesso di inviare dati. Questo tipo di gestione include un periodo di tempo in cui tutte le stazioni, tranne quella che ha richiesto il permesso, lasciano libero il canale di comunicazione.

ACK Dopo aver ricevuto un frame dati senza errori, l'AP invia un frame ACK al WNIC del mittente. Se il la scheda, a seguito di una trasmissione, non riceve un ACK entro un certo periodo di tempo, considera il frammento perso, e si appresta a inviarlo nuovamente.

1.3 Protocolli

1.3.1 Voice Over Internet Protocol

Come già anticipato, la tesi si occupa di trasmissioni Voice over IP. Il VoIP è un protocollo che permette l'invio e la ricezione di trasmissioni audio (analogico) codificate digitalmente.

L'utilizzo di una rete IP piuttosto che la rete telefonica standard, permette di comunicare con chiunque sia collegato alla stessa rete ed abbia una applicazione compatibile con la nostra a costo zero (a parte quello della connessione).

Solitamente la comunicazione consiste nello scambio di messaggi di piccole dimensioni tra due client. Chi invia, si occupa di convertire la voce in segnali digitali (una serie di bit) che verranno successivamente compressi attraverso un algoritmo al fine di ottenere pacchetti voce. Il ricevente, dai pacchetti ricostruisce la comunicazione voce e la riproduce all'utente.

Per poter funzionare, è richiesto quindi un tipo di protocollo che permetta di trasportare la voce sotto forma di dati e ne permetta la corretta riproduzione.

1.3.2 Realtime Transport Protocol

Il protocollo RTP[12] è la scelta più comune per questo tipo di utilizzo. Trattandosi di un protocollo di livello applicazione, il mittente ed il destinatario sono client VoIP.

Il protocollo di trasporto scelto per i pacchetti RTP è solitamente UDP (su IP). La sua scelta, piuttosto che protocolli più robusti come TCP, è dovuta al fatto che UDP contiene meno informazioni sulla rilevazione di errori e sulla verifica della trasmissione, riducendo quindi il carico del pacchetto sulla rete.

Ogni piccolo frammento della comunicazione viene incapsulato in tre pacchetti (RTP, UDP e IP) dove gli header contengono le seguenti informazioni:

- RTP: Sequenza del pacchetto, timestamp e le informazioni necessarie alla corretta riproduzione del messaggio.
- Header UDP: Contiene le porte del mittente e della destinazione più semplici controlli sull'integrità del pacchetto.
- Header IP: Contiene l'indirizzo del mittente e del destinatario.

Se il protocollo RTP si occupa di garantire una buona qualità della voce, quello UDP si occupa della buona qualità della comunicazione, non imponendo al sistema di tenere un ordine stretto nella ricezione dei pacchetti.

I pacchetti vengono infatti accettati secondo l'ordine di arrivo e non quello di invio. Questo sistema permette alle applicazioni di non aspettare troppo a lungo in attesa di un pacchetto perso. Nel protocollo VoIP infatti, è spesso considerato accettabile perdere qualche pacchetto, ma non lo è avere un flusso dati in ricezione troppo lento.

1.3.3 Session Initiation Protocol

Il protocollo SIP è un protocollo del livello applicazione che assomiglia, in qualche modo, al protocollo HTTP, essendo basato su un modello richies-

[12]Realtime Transport Protocol (lett: Protocollo di trasporto Real-Time) è stato sviluppato dallo Audio-Video Transport Working Group, membro della IETF (Internet Engineering Task Force).

ta/risposta simile, pur essendo stato progettato per applicazioni di tipo piuttosto diverso per cui ha potenzialità abbastanza diverse da quelle di HTTP. Le funzionalità messe a disposizione da SIP si possono raggruppare in cinque categorie:

- Raggiungibilità dell'utente: determinare il dispositivo corretto con cui comunicare per raggiungere un certo utente;
- Disponibilità dell'utente: determinare se l'utente vuole prendere parte ad una particolare sessione di comunicazione oppure se è in grado di farlo;
- Potenzialità dell'utente: determinare i media utilizzabili e i relativi schemi di codifica;
- Instaurazione della sessione: stabilire i parametri della sessione, come i numeri di porta, che devono essere utilizzati da entrambe le parti coinvolte nella comunicazione;
- Gestione della sessione: un ampio Instaurazione della sessione: stabilire i parametri della sessione, come i numeri di porta, che devono essere utilizzati da entrambe le parti coinvolte nella comunicazione.spettro di funzioni, che comprendono il trasferimento di sessioni (per realizzare, ad esempio, il "trasferimento di chiamata", call forwarding) e la modifica dei parametri di sessione;

1.3.4 User Datagram Protocol

Lo User Datagram Protocol[13][10] (UDP) è un protocollo di livello trasporto connectionless[14] e stateless (lett: senza stato).

[13]Il protocollo e' stato definito nel 1980 da David P. Reed (RFC 768).

[14]I protocolli connectionless (lett: senza connessione) sono caratterizzati dalla particolarità di non configurare una connessione end-to-end dedicata tra due nodi prima di un invio.

La comunicazione è implementata trasmettendo i pacchetti, chiamati Datagram[15], dal mittente alla destinazione senza verificare lo stato stato della rete ne quello del ricevente.

Infatti UDP non gestisce nessun tipo di controllo di flusso, ne garantisce l'affidabilità della connessione o l'ordine di ricezione. I pacchetti possono non arrivare o arrivare duplicati, ma nessun tipo di notifica è inviata all'utente. Ognuno dei pacchetti è completamente indipendente dall'altro.

Grazie a questi accorgimenti UDP è veloce ed efficiente. Inoltre il protocollo supporta il broadcast (l'invio di un pacchetto a tutti i nodi del network) ed il multicasting (l'invio dei pacchetti a tutti i sottoscritti ad un servizio).

Il protocollo UDP è quindi:

- Inaffidabile – Quando un pacchetto viene inviato non c'è modo di sapere se è arrivato a destinazione.
- Non ordinato – Se due messaggi sono inviati allo stesso nodo, non c'è modo di sapere l'ordine di arrivo.
- Leggero – Non ci sono meccanismi di controllo, quindi il pacchetto ha un minore carico sulla rete rispetto ad altri.
- Senza controlli – Viene controllata l'integrità dei pacchetti solo all'arrivo e solo se completi.

Gli unici servizi che UDP implementa sono il multiplexing (grazie alle porte) e il controllo dell'integrità (grazie al checksum). Ogni altro tipo di controllo deve essere implementato dall'applicazione che lo utilizza ad un livello superiore.

Porte Le applicazioni UDP utilizzano dei socket datagram per gestire le connessioni. I socket legano (bind) l'applicazione a delle porte, che funzionano da punti di partenza/arrivo per i pacchetti. Una porta è una struttura identificata da un numero a 16 bit (quindi un valore che può spaziare tra 0 e 65,535) unico per ogni socket. La porta 0 è riservata.

[15] Il termine Datagram solitamente si riferisce a pacchetti di un protocollo non affidabile.

Nel pacchetto UDP due campi sono dedicati alle porte: una del mittente e una del destinatario.

1.3.5 Internet Protocol

Internet Protocol[16][11] (IP) è il protocollo primario delle comunicazioni di rete. Definito nell'Internet Protocol Suite, ha il compito di trasportare pacchetti dal mittente alla destinazione basandosi solamente sul loro indirizzo.

Si occupa principalmente di gestire i metodi di indirizzamento e tutti i dettagli relativi al percorso (la consegna deve avvenire indipendentemente dalla struttura della rete e dal numero di sotto-reti presenti). Il livello del protocollo deve quindi conoscere la topologia di reti e delle sotto-reti per potere scegliere il giusto percorso attraverso di esse, soprattutto nel caso sorgente e destinazione siano in reti differenti.

I pacchetti del livello superiore vengono incapsulati in pacchetti chiamati datagram (come in UDP). Non è necessario stabile una connessione tra due nodi prima di comunicare (connectionless e stateless).

E' un protocollo di tipo best-effort, infatti non garantisce la consegna dei pacchetti a destinazione ne la correttezza dei dati, e condivide tutti i problemi di UDP:

- Corruzione dei dati
- Perdita dei pacchetti
- Pacchetti duplicati o non in ordine

Esistono due versioni del protocollo IP:

- Internet Protocol Version 4: Gli indirizzi sono a 32 bit, e per questo vi sono solo 4.294.967.296 indirizzi possibili. Un indirizzo consiste in una quadrupla di numeri separati da un punto (es: 192.168.0.1).
- Internet Protocol Version 6

[16]IETF RFC 791 pubblicato per la prima volta nel settembre 1981.

Header IP La struttura di un header IPv4 è la seguente:

1. Versione (4 bit) : specifica se Ipv4 o IPv6
2. Internet Header Length (4 bits)
3. Tipo di servizio (8 bits): anche conosciuto come QoS, descrive che priorità deve avere il pacchetto
4. Lunghezza (16 bit): viene espressa in bytes
5. Identificazione (16 bit): aiuta a ricostruire il pacchetto dai frammenti
6. Flag Reserved (1 bit): sempre valorizzato con 0
7. Don't fragment (1bit) [DF]: valorizzato con 1 se il pacchetto può essere frammentato
8. More Fragments (1 bit) [MF]: valorizzato con 1 se seguono altri frammenti del datagram
9. Fragment offset (13 bit)
10. Time to Live (8 bit) [TTL]: numero di hop[17] attraverso il quale il pacchetto può passare prima di essere scartato
11. Protocollo (8 bit): TCP, UDP, ICMP, ecc...
12. Checksum (16 bit)
13. Indirizzo IP Mittente (32 bit)
14. Indirizzo IP Destinatario (32 bit)

Un datagram IP se necessario può essere frammentato a seconda dell'MTU[18] del mezzo che si vuole utilizzare.

[17] Dispositivi di qualsiasi tipo (router, computer, ecc) attraverso i quali verrà instradato il pacchetto.

[18] Maximum transmission unit (lett: Unità massima di trasmissione) è la dimensione massima che può avare un pacchetto per viaggiare su una data rete.

Per ricostruire un datagram IP frammentato si utilizzano principalmente due campi: il bit frammentazione e l'offset. Di ogni datagram IP (con lo stesso identificatore), bisogna raccogliere tutti i frammenti fino ad avere quello con il campo moreFrag uguale a 0. A quel punto i frammenti possono essere assemblati nell'ordine corretto utilizzando gli offset come discriminante.

Capitolo 2

Obiettivi

La situazione di partenza era un modello simulativo basato su Inet, in cui un end-system wireless mobile comunicava, usando un'applicazione VoIP, con un altro nodo wireless o mobile situato in una diversa rete. I due end-system operavano in uno scenario cittadino utilizzando un'architettura denominata RWMA(Robust Wireless Medium Access) che fornisce supporto per la QoS[1].

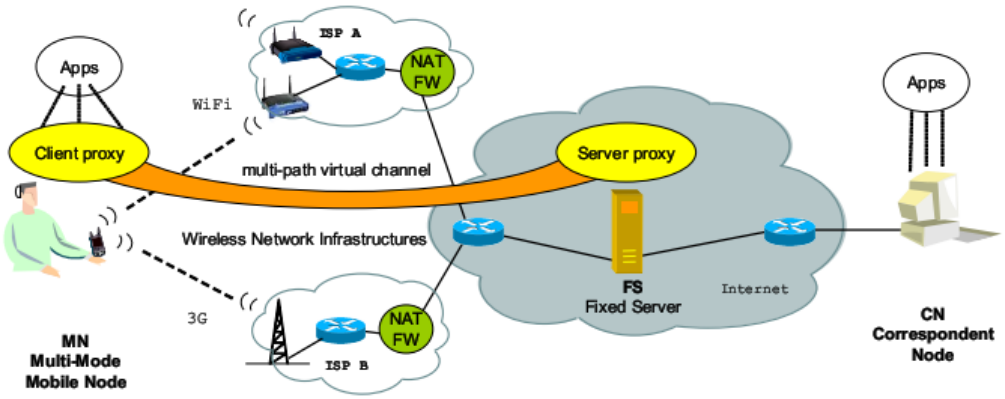

Figura 2.1: Possibile scenario di utilizzo

Si aveva quindi a disposizione network di tipo IPv4 e comunicazioni wireless di tipo WiFi (IEEE802.11x). Il nostro obiettivo è stato fornire al nodo

[1]Il temine Quality of Service (lett: Qualità del servizio) si riferisce a protocolli che si occupano di garantire determinati livelli di performance nelle trasmissioni dati.

mobile un'alternativa per la trasmissione wireless: un'ulteriore interfaccia, questa volta di tipo WiMAX (IEEE 802.16e), che potesse supplire a eventuali carenze di segnale WiFi.

2.1 Protocollo WiMAX (802.16e)

Worldwide Interoperability for Microwave Access (WiMAX)[16] è attualmente una delle tecnologie wireless più promettenti. La commissione 802 dello Institute of Electrical and Electronics Engineers (IEEE), responsabile per aver definito, tra l'altro, networking standard come Ethernet (802.3) e WiFi (802.11), ha pubblicato un insieme di standard che definiscono WiMAX. WiMAX IEEE 802.16-2004 (conosciuto anche come Revision D) è stato pubblicato nel 2004 per applicazioni di tipo fisso; 802.16 Revision E (che include la mobilità) è stato invece pubblicato nel luglio 2005. Esiste poi il WiMAX Forum, un corpo industriale costituito con lo scopo di promuovere lo standard IEEE 802.16 ed effettuare testing per la interoperabilità. Il WiMAX Forum ha adottato alcuni profili basati sugli standard 802.16 per interoperability testing e la "WiMAX certification".

WiMAX fornisce connessioni a banda larga con alto throughput su lunghe distanze e può essere usato in più modi: connessioni a banda larga dell'ultimo miglio, hotspot e connettività ad alta velocità per aziende. Offre connettività a velocità fino a 70 Mbps in una wireless metropolitan area network (WMAN), con base station capaci di coprire mediamente distanze fra i 5 e i 10 km.

WiMAX opera nelle bande di frequenza di 2.5GHz, 3.5GHz e 5.8GHz, tipicamente date in licenza da autorità governative e si basa su una tecnologia RF chiamata Orthogonal Frequency Division Multiplexing (OFDM), riuscendo così a trasferire dati in maniera molto efficiente quando l'onda portante ha ampiezza di almeno 5MHz. Sotto questa soglia, presenta prestazioni del tutto simili a sistemi 3G che usano CDMA.

Si tratta inoltre di una tecnologia su livello 1 (livello fisico) e livello 2 (livello MAC) che non definisce la connettività a livello 3 (livello network). IEEE lascia a terzi la possibilità di innovare e standardizzare i livelli superiori,

facendo sì che WiMAX sia pronto per essere connesso a una molteplicità di sistemi già presenti.

2.1.1 Infrastruttura WiMAX

Tipicamente, un sistema WiMAX consiste essenzialmente in due parti: - una Base Station WiMAX e un ricevitore WiMAX. Una Base Station è formata da due componenti: un apparato hardware interno e una torre WiMAX. In linea teorica, una base station può avere un raggio di copertura fino a 50 km, anche se considerazioni di tipo pratico limitano il raggio a non più di 10 km. Qualsiasi nodo tipo wireless all'interno dell'area di copertura è in grado di connettersi alla rete. Un ricevitore WiMAX altro non è che un nodo di qualunque tipo che presenta una interfaccia di tipo WiMAX. Più Base Station possono essere connesse fra loro tramite l'uso di connessioni backhaul, che permettono il roaming di un subscriber da una base station all'altra, in maniera simile a quanto già esiste nel mondo della telefonia mobile. Diverse opzioni di topologia e backhauling sono supportate dalle WiMAX Base Station: wireline backhauling (tipicamente su Ethernet), connessioni microwave Point-to-Point e WiMAX backhaul.

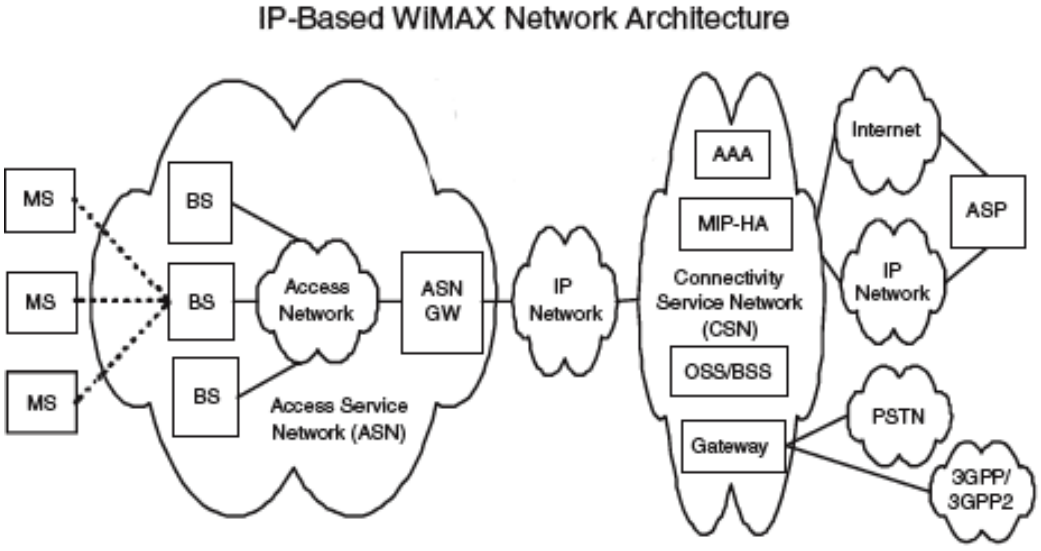

Figura 2.2: Architettura base scenario WiMax

2.1.2 Mobile WiMax

Senza alcun dubbio, WiMAX fisso, basato sullo Air Interface Standard 802.16-2004 ha dimostrato di essere una validissima alternativa, anche in termini economici, ai servizi di connessione via cavo e DSL. Nel dicembre 2005 IEEE ha ratificato lo standard 802.16e, che aggiunge le funzionalità necessarie a supportare la mobilità. Il WiMAX Forum sta attualmente definendo non solo profili di performance del sistema e di certificazione basati su IEEE 802.16e, ma anche l'architettura necessaria per implementare un end-to-end Mobile WiMAX network. Mobile WiMAX è una soluzione wireless a banda larga che consente la convergenza di network a banda larga mobili e fissi attraverso una tecnologia di accesso radio a banda larga coprente una vasta area, e una architettura di rete flessibile. La Mobile WiMAX Air Interface adotta Orthogonal Frequency Division Multiple Access (OFDMA) per migliori prestazioni multi-path in un ambiente che non sia line-of-sight. E' stato introdotto un OFDMA scalabile (SOFDMA) nel protocollo IEEE 802.16e per supportare bandwidth di canale scalabili da 1.25 fino a 20 MHz.

Mobile WiMAX permette a sistemi mobili di essere configurati bansandosi su un insieme di caratteristiche di base in comune, assicurando così le necessarie funzionalità a terminali fissi, mobile station e base station. Alcuni elementi della base station sono specificati come opzionali per fornire un'ulteriore flessibilità nell'utilizzo in scenari specifici, dove differenti configurazioni ottimizzate sulla capacità piuttosto che sulla copertura potrebbero risultare necessarie.

Alcune delle caratteristiche salienti di Mobile WiMAX sono:

- Data rate alti. L'inclusione della tecnologia di antenna MIMO (Multiple Input Multiple Output), assieme a meccanismi di sub-channelization flessibili, Advanced Coding e Modulation consentono a Mobile WiMAX di supportare in downlink data rate fino a 63Mbps a settore e picchi in uplink fino a 28Mbps a settore, in un canale da 10MHz.

- Quality of Service. (QoS). Uno degli obiettivi principali dell'architettura del livello MAC in IEEE 802.16e è implementare la QoS. Essa definisce Service Flows che mappano a DiffServ code points (DSCP),

consentendo così una Quality of Service basata su IP. Inoltre, i meccanismi di subchannelization offrono un meccanismo flessibile per uno scheduling ottimale delle risorse in spazio, frequenza e tempo, per ogni frame.

- Scalabilità. A dispetto di una economia sempre più globalizzata, le risorse di spettro per quanto riguarda la banda larga wireless sono ancora allocate in modo piuttosto eterogeneo. La tecnologia Mobile WiMAX è di conseguenza stata pensata per potersi adattare in diverse channelization, da 1.25 a 20MHz, per essere conforme ai vari requisiti presenti nei diversi Paesi. Solo a lungo termine infatti è previsto il raggiungimento di un'armonizzazione di questi valori a livello mondiale. L'attuale situazione permette però di trarre vantaggio dai molteplici benefici di Mobile WiMAX nelle realtà geografiche più svariate: da un accesso ad internet a prezzo contenuto in contesti rurali, a una più capillare diffusione della banda larga mobile in aree metropolitane e suburbane.

- Sicurezza. Mobile WiMAX incorpora le funzionalità di sicurezza più avanzate attualmente in uso in sistemi ad accesso wireless, vale a dire: autenticazione con Extensible Authentication Protocol (EAP), crittazione autenticata con Advanced Encryption Standard (AES), e sistemi di protezione con Cipher-base message Authentication Code e (CMAC) e Hashed Message Authentication Code (HMAC).

- Mobility. Mobile WiMAX è in grado di supportare meccanismi di handover ottimizzati con latenze di meno di 50 ms per garantire che applicazioni real-time come Voice over Internet Protocol (VoIP) siano supportate in modo efficiente senza quindi degradazione nel segnale. Meccanismi di key management assicurano che anche durante l'handover la sicurezza del sistema rimanga invariata.

2.2 Obiettivi della tesi

Gli obiettivi che ci si è prefissati di raggiungere sono:

1. Aggiungere a quanto già presente nell'elaborato di Marco Ciaramella[19] una simulazione di un protocollo di trasmissione wireless a lungo raggio che permettesse così al nodo mobile di connettersi attraverso l'interfaccia della tecnologia scelta, in mancanza di segnale WiFi.

2. Poiché il modello simulativo di WiMAX utilizzato faceva uso e già disponeva di un protocollo IPv6 per la simulazione, mentre il nostro network WiFi era solo di tipo IPv4, è stato necessario valutare come integrare le due diverse reti per arrivare a una soluzione accettabile senza compromettere la validità dell'intero sistema.

3. Congiuntamente a un livello network di tipo IPv6, il progetto che abbiamo inserito e adeguatamente modificato prevedeva l'utilizzo del protocollo DHCPv6. E' stato nostro compito creare sia nel nodo mobile che nel fisso con il quale avviene la comunicazione VoIP una sorta di dual stack che permettesse la trasmissione dei dati nell'uno o nell'altro protocollo IP, e la configurazione dei livelli superiori, di trasporto e di applicazione, usando il corretto protocollo DCHP.

2.3 Scelte implementative

Al fine di raggiungere efficacemente gli obiettivi preposti sono state prese le seguenti decisioni:

- Sfruttando il fatto che la parte fissa della comunicazione VoIP è stata implementata direttamente sul proxy server, che fa quindi le veci del secondo end system, si è pensato di usare il proxy server stesso come punto di congiuntura fra le reti IPv4 e la nostra IPv6.

- Ai puri fini di creare una prima simulazione WiMAX all'interno del contesto fornitoci, è stata inserita una sola Base Station, la cui unica

cella copre l'intera area urbana. Questo ci ha permesso di concentrarci maggiormente sul parte di integrazione del codice e allo stesso tempo essere coerenti con la tecnologia di trasmissione a lungo raggio scelta, che consentirebbe comunque una copertura fino a 10km di raggio.

- Per gli stessi motivi di cui sopra, le politiche di gestione del movimento nei moduli relativi a WiMAX, per quanto semplici, sono state disabilitate: si assume che il nodo, una volta instaurata la connessione con la Base Station WiMAX sia in grado di mantenerla per tutta la durata della simulazione.

- Infine, la decisione di utilizzare il protocollo IPv4 o quello IPv6, da parte e del nodo mobile e di quello fisso, è basata nel primo caso in una scelta random degli indirizzi di destinazione disponibili, mentre nel secondo un apposito flag fa sì che vengano creati alternativamente pacchetti con indirizzi dell'uno o dell'altro tipo, congruentemente alle attuali configurazioni dhcp presenti.

2.4 Ambiente simulativo

Si è scelto di sviluppare l'applicazione utilizzando la piattaforma simulativa OMNeT++, che fornisce il sistema di gestione degli eventi, sulla quale è stato esteso il framework INET, che implementa i più diffusi protocolli di rete.

Capitolo 3

Framework

In questo capitolo verranno introdotti i concetti fondamentali necessari per comprendere il lavoro svolto nella tesi e le scelte implementative che verranno discusse nei capitoli successivi. Verrà spiegato cosa sono e come funzionano i framework OMNeT++[8] e INET[12], su cui si basa l'intera implementazione e verrà descritto per sommi capi il progetto di simulazione del protocollo WiMAX che si è scelto per l'integrazione.

3.1 OMNeT++

OMNeT++ è un framework modulare per lo sviluppo di simulazioni ad eventi discreti. Non è propriamente un simulatore, ma piuttosto una infrastruttura che fornisce i mezzi per scrivere delle simulazioni.

Il campo in cui sta subendo lo sviluppo maggiore è sicuramente quello delle reti per la comunicazione. Ad ogni modo, la sua architettura generica e flessibile gli permette di essere usato con successo anche in altri ambiti, tra i quali:

- Modellazione di protocolli generici
- Modellazione di sistemi multiprocessore o distribuiti
- Validazione di architetture hardware
- Valutazione delle performance di sistemi software

Più in generale, OMNeT++ può essere usato in qualsiasi contesto dove un approccio ad eventi discreti è applicabile, e le cui entità possono essere mappate in moduli che comunicano attraverso lo scambio di messaggi[1]. Il suo diffusissimo uso sia nella comunità scientifica che in quella industriale, lo rende un'ottima ed affidabile scelta per l'implementazione del meccanismo RWMA.

La versione di OMNeT++ a cui facciamo riferimento è la 4.0 (stabile) rilasciata il 27 Febbraio 2009[9].

3.1.1 Moduli

Le componenti fondamentali del framework sono delle classi riusabili chiamate **moduli**, scritte in C++ usando le librerie di OMNeT++.

I moduli sono strutturati in maniera gerarchica, fino a modellare una struttura il cui numero massimo di livelli non è definito. Quelli al livello più basso vengono chiamati **simple modules** (moduli semplici), e rappresentano delle entità e/o dei comportamenti. Il modulo di massimo livello invece viene chiamato **system module** (modello di sistema) e racchiude tutto il sistema nel suo complesso.

I moduli semplici vengono assemblati in componenti più grandi e complessi chiamati **compound modules** (moduli composti) o in **modelli** (chiamati anche network), attraverso l'uso di un linguaggio di alto livello chiamato **NED**. Un modello è di per sé un modulo composto.

I moduli semplici possono avere parametri che vengono principalmente utilizzati per passare dati di configurazione ai moduli semplici, o per parametrizzarne il comportamento (possono essere stringhe, numeri o valori booleani). All'interno di un modulo composto invece i parametri possono definire il numero di sotto moduli, gate o connessioni interne.

I parametri possono essere assegnati nei file NED, nei file di configurazione (con estensione ini) o possono essere chiesti interattivamente all'utente al lancio della simulazione. Possono riferirsi ad altri parametri o essere il frutto di calcoli su di essi.

[1] Tutto OMNeT++ è costruito sul meccanismo del Message Passing.

3.1.2 Gate

Solitamente i moduli semplici inviano e ricevono messaggi attraverso dei gate (associabili al concetto di porte), i quali sono in tutto e per tutto le interfacce di input ed output di un modulo.

Un gate può essere collegato con una **connessione** creata all'interno di uno stesso livello gerarchico, oppure può collegare un modulo semplice con uno composto. Le uniche connessioni vietate sono quelle tra diversi livelli gerarchici, perché andrebbero a minare la riusabilità del modello stesso. Data la struttura gerarchica, i messaggi viaggiano attraverso una catena di connessioni, per partire ed arrivare in un modulo semplice attraverso dei gate.

Una connessione che collega due moduli semplici viene anche definita **route** (rotta) o **link**.

Le connessioni possiedono tre parametri (tutti opzionali):

- Propagation delay: lasso di tempo di cui il pacchetto viene rallentato nel mezzo prima di essere consegnato.
- Bit error rate: probabilità che un bit venga trasmesso in maniera non corretta.
- Data rate (bit/s): viene usata per calcolare il tempo di trasmissione di un pacchetto.

3.1.3 Messaggi

All'interno della simulazione, i messaggi possono rappresentare frame, pacchetti di un network, job o qualsiasi altro tipo di struttura arbitraria. Hanno attributi standard (quali il timestamp) e possono arrivare da un altro modulo oppure dal modulo stesso, nel qual caso vengono definiti **self-message** (lett: auto-messaggi) il cui invio è gestito con dei timer.

Il "tempo locale di simulazione" di un modulo è strettamente legato ai messaggi. Questo infatti aumenta all'arrivo nel modulo di uno di essi.

Gli header dei pacchetti sono descritti nei **Message Definition File** (file semplici con estensione msg), scritti in una sintassi simile a C. Questi file

vendono tradotti in classi ed header C++ dal tool OMNeT++ **opp_msgc**. Per esempio, un nuovo pacchetto chiamato pacchetto.msg, dato in input al programma opp_msgc genera due file C++ chiamati rispettivamente: pacchetto_m.h e pacchetto_m.cc. Per poter creare ed utilizzare i pacchetti qui definiti all'interno della propria implementazione, è necessario includere l'header file del pacchetto nella propria classe.

Le classi generate in questo modo sono sottoclassi di cMessage (libreria OMNeT++).

3.1.4 Comunicazione tra livelli di protocollo

In OMNeT++ quando un protocollo di un livello superiore vuole mandare il pacchetto su un protocollo di livello inferiore, manda l'oggetto attraverso la connessione che li lega al modulo sottostante, che poi si occuperà di incapsularlo ed inoltrarlo.

Il processo inverso avviene quando un modulo di un livello inferiore riceve un pacchetto. In questo caso il modulo si occupa di mandarlo al livello superiore dopo averlo decapsulato.

Control Info A volte è necessario veicolare informazioni aggiuntive insieme al pacchetto. Questo tipo di informazioni viaggiano in un oggetto chiamato **control info**, che viene collegato al messaggio attraverso la chiamata setControlInfo() del pacchetto, e contiene informazioni ausiliarie che sono necessarie al livello a cui è diretto, ma che non sono da inoltrare ad altri moduli. Gli oggetti control info possono ovviamente essere definiti dall'utente, e sono sottoclassi di cObject (libreria OMNeT++).

3.1.5 Il linguaggio NED

Il termine NED fa riferimento a **Network Description** (lett. Descrizione del network), ed è un linguaggio di alto livello usato dall'utente per descrivere la struttura di un modello.

Ha una struttura gerarchica e flessibile (a package) simile a Java, per ridurre il rischio di name clashes tra moduli differenti. Un esempio è il

NEDPATH (simile al CLASSPATH di Java), che rende più facile specificare dipendenze tra moduli.

Altra particolarità di questo linguaggio è la sua struttura ad albero, del tutto equivalente ad XML[2]. Un file con estensione ned può quindi essere convertito in XML (o viceversa) senza perdita di dati, inclusi i commenti.

Il NED permette di creare moduli semplici, e successivamente connetterli ed assemblarli in moduli composti, di ottenere sottoclassi, aggiungere parametri, gate e nuovi sotto-moduli e di assegnare parametri esistenti a valori fissi o casuali.

Quando vengono modificati file ned non è necessaria una ricompilazione, essendo tutto gestito a run-time.

3.1.6 I file .ini

Questo tipo di file contiene la configurazione ed i dati di input per le simulazioni. I dati al suo interno sono raggruppati in sezioni il cui nome, racchiuso tra parentesi quadre (e dopo la parola chiave Config), indica l'identificatore univoco della simulazione. L'uso di wildcard e dell'ereditarietà tra le simulazioni permette una configurazione veloce di un gran numero di moduli contemporaneamente.

In questi file si fa riferimento ai parametri dei moduli attraverso il loro path completo o al loro "nome gerarchico". Quest'ultimo consiste in una lista di nomi di moduli separati dal "." (dal modulo di massimo livello al modulo contenente il parametro).

I parametri vengono assegnati attraverso l'operatore "=", e la loro risoluzione avviene nel seguente modo:

1. Se il parametro è già assegnato nel NED, questo non può essere sovrascritto

2. Se dopo l'operatore di assegnamento è presente un valore, questo viene assegnato al parametro

[2]XML (Extensible Markup Language (XML) è un formato di codifica per il testo semplice e flessibile, derivato da SGML (ISO 8879).

3. Se dopo l'operatore di assegnamento è presente l'identificatore "default", viene assegnato il valore di default per il tipo del parametro

4. Se dopo l'operatore di assegnamento è presente l'identificatore "ask", il valore da assegnare viene chiesto in modo interattivo all'utente

5. Se il parametro non viene assegnato ma ha un valore di default definito, questo viene assegnato

6. Se non si è in presenza di nessuno dei casi sopra citati, il parametro viene dichiarato "non assegnato" e verrà gestito a seconda della politica dell'interfaccia utilizzata

3.1.7 OMNEST

OMNEST è la versione commerciale di OMNeT++. Il framework infatti è libero solo per un uso accademico senza scopo di lucro. Per utilizzi commerciali è necessario acquistare una licenza OMNEST da Simulcraft Inc.

3.2 INET

Il framework INET è costruito al di sopra OMNeT++, e si basa sullo stesso concetto: moduli che comunicano attraverso l'invio di messaggi. È interamente open-source, e viene usato principalmente per la simulazione di comunicazioni di rete.

Contiene modelli per diversi protocolli tra i quali citiamo: UDP, TCP, SCTP, IP, IPv6, Ethernet, PPP, IEEE 802.11, MPLS, OSPF.

3.2.1 Protocolli

I protocolli sono rappresentati da moduli semplici la cui implementazione è contenuta in una classe che solitamente porta il loro nome. Le interfacce di rete (Ethernet, 802.11, ecc), invece sono solitamente dei moduli composti.

Questi moduli possono essere liberamente ricombinati per formare station o altri dispositivi a seconda delle necessità. Diversi tipi di host, router, switch,

ed Access Point sono già presenti all'interno del codice di INET nella cartella "nodes", ma ovviamente ne possono essere creati di nuovi su misura per il proprio scenario.

Non tutti i moduli semplici però implementano protocolli. Ci sono moduli che contengono informazioni (es: RoutingTable), gestiscono la comunicazione (es: Notification Board), l'auto-configurazione di un network (es: FlatNetworkConfigurator), il movimento dei nodi (es: LinearMobility), oppure gestiscono operazioni sui canali radio nelle comunicazioni wireless (es: ChannelControl).

3.2.2 Moduli comuni

Ci sono moduli che grazie all'importantissimo ruolo ricoperto, sono quasi indispensabili all'interno di host, router ed altri dispositivi di rete. Tra questi segnaliamo:

- InterfaceTable: questo modulo tiene memoria della tabella delle interfacce (eth0, wlan0, ecc) negli host. Non manda ne riceve messaggi ed è accessibile dagli altri moduli attraverso una semplice chiamata C++. Le schede di rete si registrano (inseriscono nella tabella) dinamicamente implementandone l'interfaccia.

- RoutingTable: questo modulo gestisce le routing table per IPv4 e viene acceduto dai moduli che sono interessati a lavorare con le rotte dei pacchetti (soprattutto IP). Ci sono funzioni per richiedere, aggiungere, cancellare, e trovare le rotte migliori per un dato indirizzo IP.

- NotificationBoard: permette ai moduli di comunicare secondo un modello publish-subscribe. Lavorando in questa modalità, quando un modulo genera un evento, questo viene notificato a tutti i moduli che lo hanno sottoscritto. Nessuno scambio di messaggi è richiesto.

- ChannelControl: necessario nelle simulazioni wireless, tiene traccia dei nodi e della loro posizione.

3.2.3 Architettura di una NIC IEEE 802.11

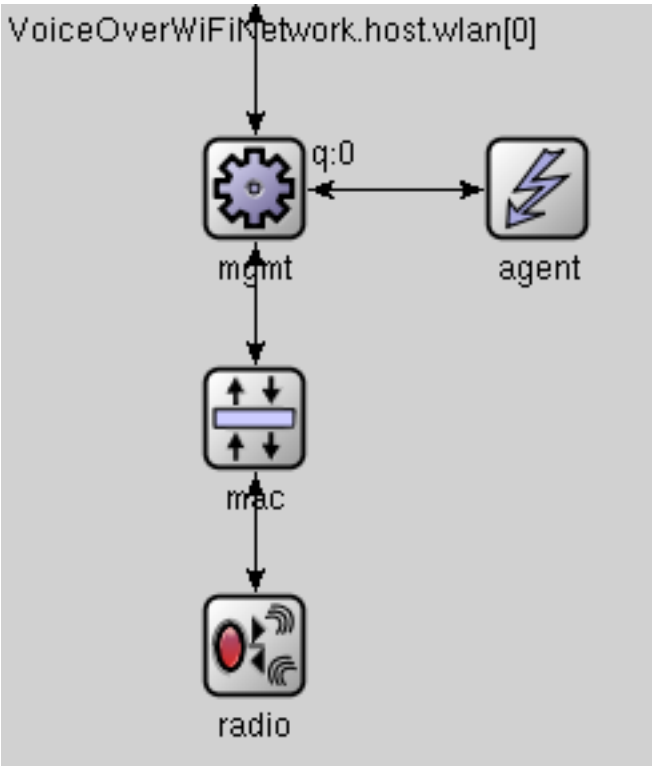

Figura 3.1: Architettura di una WNIC in INET

Una interfaccia NIC (Ieee80211) nel framework INET consiste nei seguenti quattro moduli composti tra loro (in ordine top-down):

1. Agent
2. Management
3. MAC
4. Livello fisico (radio)

L'agent L'agent è il modulo che si occupa di gestire il comportamento del livello a lui annesso (management). Si occupa di ordinare (attraverso messaggi command) a quest'ultimo di condurre operazioni quali la scansione dei canali radio, l'autenticazione o l'associazione con un Access Point. Il

livello management si limita ad eseguire questi comandi per poi riportare il risultato all'agent. Modificando o rimpiazzando un agent, si può modificare il comportamento stesso di uno STA ed implementare algoritmi o strategie necessari alla propria simulazione.

Il manager Il manager si occupa di incapsulare e decapsulare i messaggi per/dal MAC, e scambiare frame di gestione con altre station o AP. I frame Probe Request/Response, Authentication, Association Request/Response ecc, sono creati ed interpretati dal manager ma trasmessi e ricevuti attraverso il MAC. Durante la scansione è il manager che cambia periodicamente canale per raccogliere informazioni e ricevere beacon e probe response. Come l'agent, ha differenti implementazioni a seconda del suo ruolo.

Il livello MAC Il livello MAC si occupa della trasmissione dei frame secondo il protocollo CSMA/CA[3]. Riceve dati e management frame dai livelli alti, e li trasmette.

Il livello fisico Il livello fisico si occupa della trasmissione e ricezione dei frame. Modella le caratteristiche del canale radio e determina se un frame è stato ricevuto o no correttamente (ad esempio nel caso subisca errori a causa del basso potere del segnale o interferenze nel canale radio). I frame ricevuti correttamente sono passati al MAC.

3.2.4 I file .irt

I file di Routing hanno estensione irt, e vengono utilizzati per configurare il modulo RoutingTable (presente in tutti i nodi IP) prima del lancio di una simulazione.

Questi file possono contenere la configurazione delle interfacce di rete e delle rotte statiche (che verranno aggiunte alla routing table), entrambe opzionali.

[3]Carrier Sense Multiple Access (CSMA) è un protocollo MAC probabilistico nel quale un nodo verifica l'assenza di altro traffico prima di trasmettere su un canale condiviso.

Le interfacce sono identificate da nomi (es: ppp0, ppp1, eth0) originariamente definiti nel file NED.

Ogni file è suddiviso in due sezioni (che per struttura ricordano l'output dei comandi "ifconfig" e "netstat -rn" di Unix):

- Compresa tra le keyword "ifconfig" e "ifconfigend": Configurazione delle interfacce
- Compresa tra le keyword "route" e "routeend": Contiene le rotte statiche

Interfacce I campi accettati nel definire un'interfaccia sono:

- name: nome (e.g. ppp0, ppp1, eth0)
- inet_addr: indirizzo IP
- Mask: netmask
- Groups: gruppo Multicast
- MTU: MTU del canale
- Metric
- flag: BROADCAST, MULTICAST, POINTTOPOINT

Rotte I campi accettati nelle rotte sono:

- Destination: Indirizzo IP del destinatario (o default)
- Gateway
- Netmask
- Flags: H (host: rotta diretta per il router) o G (gateway, rotta remota attraverso un altro router)
- Metric
- Interface

3.2.5 Le simulazioni

Le simulazioni in OMNeT++/INET possono essere eseguite in diverse modalità. Mentre l'interfaccia grafica (default) è comoda per dimostrazioni ed il debugging, la versione da linea di comando è sicuramente quella più adatta per esecuzioni batch.

Durante una simulazione tutti i campi di una classe (se appositamente inizializzati) possono essere controllati, navigati e modificati.

Per eseguirne una, bisogna recarsi nella cartella dove è situato il file di configurazione (quello con estensione ini) e lanciare l'eseguibile di INET con il file come parametro.

- /PATH_TO_INET/run_inet <*fileDiConfigurazione.ini*>

Questo tipo di esecuzione lancia una interfaccia grafica. Le simulazioni definite nel file (nel caso ce ne siano più di una) possono essere scelte da un comodo menù a tendina nella finestra della simulazione. Mentre nel caso si voglia lanciare direttamente una simulazione specifica all'interno del file, si utilizza la chiamata:

- /PATH_TO_INET/run_inet -c <*nomeDellaSimulazione fileDiConfigurazione.ini*>

3.3 Numbat[15]

E' stata fatta un'attenta ricerca sulla rete per verificare se fosse già stato implementato almeno uno dei protocolli del tipo richiesto, ossia wireless a lungo raggio: WiMAX, UMTS o GPRS, sia in Omnet++ che in ns2 o ns3 (altre piattaforme di simulazione), per un'eventuale operazione di porting. La scelta naturale è stata WiMAX, essendo presente su Internet sia un modello per Omnet++ che uno per ns3. Per quanto riguarda gli altri due protocolli menzionati, la situazione non è apparsa altrettanto fortunata: mentre per ns2 o il più recente ns3 esistevano implementazioni, per Omnet++ nulla ancora era stato fatto.

Il progetto di simulazione di protocollo Mobile WiMAX scelto, *Numbat*[17], ossia "New Ubiquitous Mobility Basic Analysis Tool", è stato sviluppato dal dottorando Tomasz Mrugalski della Gdansk University of Technology (Polonia) e usa per la comunicazione e configurazione fra nodi mobili (Subscriber Stations), access point (Base Stations) e nodi fissi (Corresponding Nodes) una propria implementazione del protocollo IPv6 e DHCPv6.

3.3.1 Principi del progetto

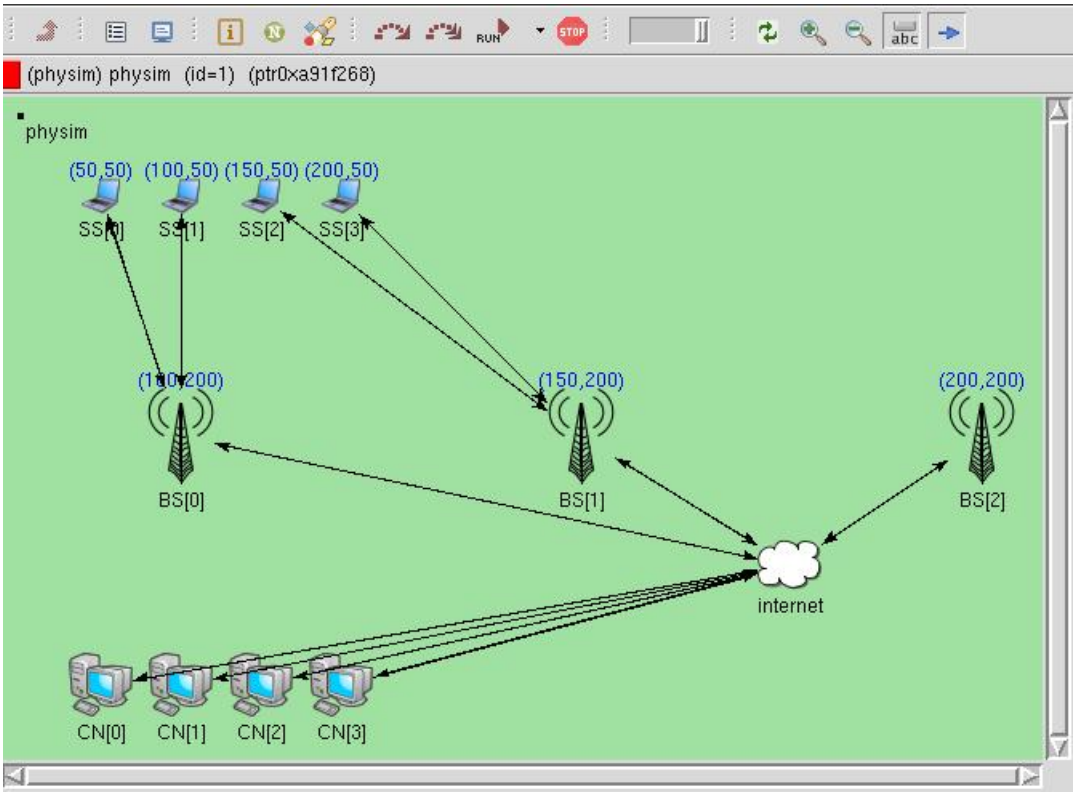

Figura 3.2: Numbat

Numbat è un ambiente di simulazione pensato e implementato con il fine di soddisfare obiettivi e requisiti specifici:

- Flessibilità – Offre un ambiente di simulazione di IPv6 per tutte quelle station in grado supportare 802.16e, ossia subscriber station fisse, subscriber station mobili e base station.

- Handover Oriented – Simula i meccanismi essenziali di IEEE 802.16e, come network entry e handover procedure, ma omette meccanismi non essenziali dal punto di vista della mobilità. Si avranno quindi, ad esempio, una implementazione banale della protezione crittografica per la comunicazione e una implementazione semplificata del canale radio.

- Implementazione dello stack IPv6 – Stateless autoconfiguration, Router Advertisements, DHCPv6 e Mobile IPv6. Non è necessaria una completa implementazione: solo quegli aspetti che riguardano o che incidono sulla mobilità sono stati presi in considerazione.

- Approccio Modulare - L'ambiente di Numbat è stato pensato per essere in constante sviluppo. Invece che simulare un sistema complesso, il tutto è stato diviso in più moduli di piccole dimensioni, ciascuno dei quali interagisce con i suoi "vicini" o con il suo "genitore". D'altro canto, per simulare entità più complesse, è sufficiente raggruppare moduli semplici in moduli composti.

- Flessibilità di configurazione. La definizione dei parametri della simulazione deve essere flessibile e facile da modificare.

- Versioni visuali e di testo - Durante le fasi di sviluppo, debugging e presentazione, può tornare utile usare una interfaccia grafica che permetta l'ispezione visuale di tutti gli elementi del network. Quando i parametri di simulazione sono customizzati, tuttavia, simulazioni complesse possono durare parecchio tempo prima di produrre risultati. Durante queste esecuzioni, la visualizzazione grafica rallenta in modo considerevole l'intero processo, tanto da rendere necessaria una interfaccia a linea di comando. Un ulteriore vantaggio è che quest'ultima modalità è utilizzabile anche in esecuzione da remoto.

3.3.2 Architettura di Subscriber e Base Station

IEEE 802.16e è un protocollo asimmetrico: una station agisce in maniera differente a seconda che sia una subscriber station o una base station. Tut-

tavia, esistono alcune somiglianze nella funzionalità che implementano: per esempio, entrambe le station possiedono scheduler che coordina la trasmissione e la ricezione dei messaggi di dati. Per questo motivo Base Station e Subscriber Station sono divise in moduli del tutto simili. Un ambiente di networking complesso solitamente ha a che fare con un gran numero di pacchetti dati e un minor numero di pacchetti di controllo. Per ragioni di performance, i pacchetti dati sono spesso gestiti utilizzando un fast path, a cui ci si riferisce normalmente come *data plane*. I messaggi di controllo necessitano di solito di una gestione aggiuntiva, facendo sì che la loro trasmissione e ricezione impieghi più tempo. Questa gestione di messaggi su slow path è detta *control plane*. Nel design di Numbat, control plane e data plane sono stati divisi e implementati separatamente. In seguito sono riportati i moduli presenti nel progetto:

- Modulo IPv6 – invia e riceve messaggi IPv6. Si tratta di un modulo composto che costituisce l'intero stack IPv6. E' formato da diversi sottomoduli: IPv6Gen (un generatore e analizzatore di traffico per IPv6), DHCPv6Cli (un client DHCPv6), RaSrv (Router Advertisement server/router), RaCli (Router Advertisement client), MobIPv6Mn (Mobile IPv6 mobile node) and MobIPv6Ha (Mobile IPv6 Home Agent).

- WMaxCS – Convergence Sublayer. Classifica i dati ricevuti in base alle connessioni presenti e li smista ai moduli di destinazione corrispondenti.

- WMaxCtrl – Implementa il control plane, cioè la logica che sta alla base della Base Station e della Subscriber station. Tutte le decisioni vengono prese qui. E' una istanziazione della Finite State Machine descritta più avanti.

- WMaxMAC – Rappresenta la parte principale del data plane. Questo è principalmente uno scheduler (per la trasmissione) e un dispatcher (per la ricezione).

- WMaxPHY – Simulazione del livello PHY (fisico) di IEEE 802.16e. Poiché le operazioni a livello fisico cadono al di fuori dei fini del progetto, la loro implementazione è triviale.

- WMaxRadio – Presente solo nella Base Station. E' una semplice simulazione del canale radio. Supporta due tipi di trasmissione: broadcast (un mittente che trasmette a tutti i possibili riceventi, ad esempio una base station e i suoi subscriber) e unicast (un mittente che trasmette a un ricevente, ad esempio una subscriber station e una base station).

Dal momento che le base station funzionano come relay e non generano da sole alcun traffico dati, subscriber aggiuntivi sono stati aggiunti nell'esempio di simulazione (physim.ned) per simulare traffico di background. Un subscriber fisso può essere configurato molto facilmente in modo che agisca da generatore o analizzatore di traffico. L'ambiente di Omnet++ include già una implementazione di una Finite State Machine (FSM), la cui interfaccia non offre però la necessaria flessibilità. Il difetto maggiore riscontrato dagli sviluppatori di Numbat è che non permette di rimanere sul medesimo stato, se non uscendo e rientrando. E' stata quindi sviluppato un framework per una nuova FSM.

3.3.3 Mobility model

Ci sono due possibili modalità per la simulazione della mobilità di un nodo mobile in Numbat:

- Location Based - Un subscriber mobile cambia di posizione ed effettua uno scanning periodico. Quando rileva la presenza di una base station più vicina di quella a cui è attualmente associato, inizia il meccanismo di handover. Questo modello è più realistico, ma richiede la pianificazione della posizione di ogni base station e subscriber station. Sono disponibili due movimenti: circolare o orizzontale.

- Time based - E' possibile definire che, qualunque sia la sua posizione, un subscriber inizi l'handover dopo un certo lasso di tempo. Il modello fornito è comunque una semplificazione, ma può dimostrarsi utile per quegli scenari che si concentrano sulla procedura stessa di handover.

Capitolo 4

Progettazione

Tutto il codice è stato interamente progettato e strutturato per lavorare sul framework OMNeT++ [8] (v. 4.0). Il progetto parte dall'implementazione del sistema RWMA realizzato da Fabrizio Sabatini[14] e Marco Ciaramella, e utilizza la versione di INET modificata per supportare schede wireless multiple su ogni singola station, ad opera di Piero Murphy [13]. A questo si è aggiunto il modello simulativo di Mobile WiMAX "Numbat", disponendone i sorgenti in maniera coerente con la gerarchia dei file di Inet, e lo si è modificato adeguatamente per fornire uno scenario il quanto più realistico.

L'implementazione dovrebbe risultare compatibile con versioni future di INET e OMNeT++, a meno di cambiamenti nelle interfacce delle classi.

4.1 Connessione fra reti IPv4 e IPv6

Come già accennato nel precedente capitolo, il progetto Numbat contiene, per precise scelte progettuali, una implementazione di IPv6 nativa, mentre la versione di Inet estesa a nostra disposizione fa uso di un livello network di tipo IPv4, essendo stata rilasciata solo negli ultimissimi tempi una versione ufficiale e completamente funzionante di un modello IPv6 per Omnet++. La coesistenza di un network IPv6 con uno IPv4 è generalmente risolta introducendo un componente chiamato NAT IPv6-IPv4 che si preoccupa di convertire indirizzi IPv6 in IPv4 e relativi pacchetti. Si è subito

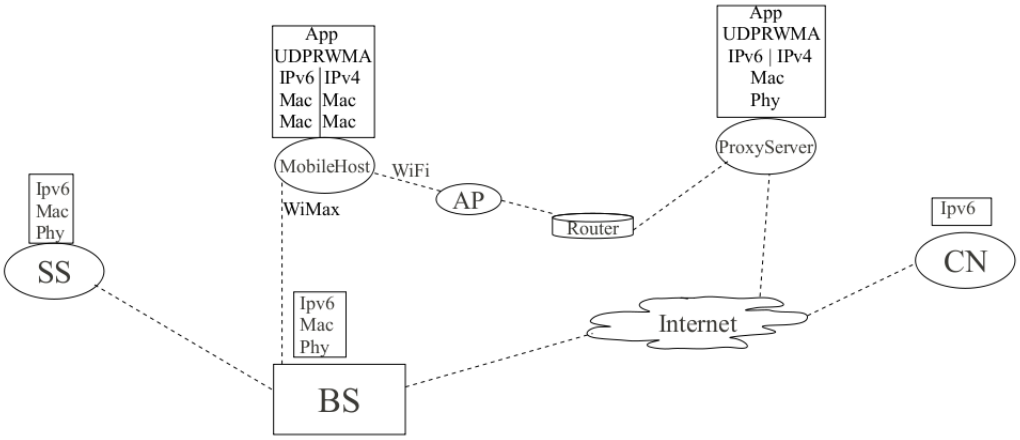

Figura 4.1: Dual Stack

notata l'assenza in Internet di un modello per Omnet++ già implementato che fungesse da NAT IPv6-IPv4 e che quindi permettesse agilmente la connessione fra le due reti. É immediatamente parso chiaro che la creazione del suddetto modello avrebbe necessitato una mole di lavoro di analisi, progettazione e implementazione non congruente con l'obiettivo finale e che, anzi, sarebbe potuto essere esso stesso oggetto di un progetto separato. Si è quindi proceduto con la ricerca di un qualche tipo di semplificazione che però non minasse al funzionamento il più possibile realistico del modello simulativo stesso. Si è perciò pensato di sfruttare il fatto che, in Inet esteso, il nodo mobile comunicasse già con il solo Proxy Server, che fungeva di fatto anche da end point per la comunicazione VoIP, per progettare il seguente scenario simulativo: poiché il nodo mobile conosce già in partenza l'indirizzo IPv4 del proxy server, nulla ci vieta di metterlo a conoscenza anche dell'indirizzo IPv6 dello stesso proxy server, che quindi disporrà di due indirizzi diversi a seconda del network con cui sta comunicando e che di conseguenza avrà il ruolo di punto di incontro fra le due diverse reti. Si noti inoltre il fatto che le classi implementate nei moduli di Inet per i protocolli di livello di trasporto e applicazione sono perfettamente compatibili sia con IPv6 che con IPv4.

4.2 Dual Stack

Per poter comunicare sia attraverso una delle interfacce WiFi che attraverso l'interfaccia WiMAX, il nodo mobile ha bisogno di gestire, per la prima opzione, messaggi di livello fisico e MAC relativi a 802.11x e network IPv4, mentre per la seconda, messaggi di livello fisico e MAC propri dell'implementazione di WiMAX e messaggi di livello network di tipo IPv6. Ciò risulta possibile con l'utilizzo di un dual stack che permetta quindi al nodo di poter avere a livello fisico, MAC e network due diversi tipi di protocolli, che entrano in azione a seconda delle operazioni richieste. Sono stati quindi inclusi nel modello composito rappresentante il nodo mobile i moduli relativi ai livelli PHY, MAC e IPv6, mentre per quanto riguarda il proxy server, è servito aggiungere solamente il livello IPv6. Per scelta progettuale, infatti, gli autori di Numbat hanno adottato una semplificazione per cui, una volta usciti dalla Base Station a livello IPv6 (questa, a tutti gli effetti, è un'ulteriore interpretazione), i pacchetti vengono instradati a destinazione senza più passare per i livelli sottostanti. I correspondent node (CS) dispongono infatti solo della componente IPv6. Inoltre, sempre in Numbat, non esistono livelli di stack sopra il livello network: la generazione iniziale e la ricezione finale dei messaggi dati, nonché la registrazione delle statistiche, avviene a livello IPv6, ossia il livello più alto nello stack presente. E' stato quindi necessario, per poter utilizzare l'applicazione VoIP sopra il suddetto stack, aggiungervi i due livelli sovrastanti che mancavano. I moduli a livello applicazione e trasporto, rispettivamente BasicAppForMultipleNics (BasicAppForServerProxy per il server), ULBRWMA (ULBRWMAserverproxy) e UDPRWMA (UDPRWMAserverproxy), facevano già uso di classi per la memorizzazione di indirizzi trasparente alla tipologia di IP a cui quest'ultimi facevano riferimento. E' stato sufficiente modificare opportunamente il livello IPv6, affinché non generasse più traffico, ma gestisse e inoltrasse i pacchetti ricevuti dai livelli superiori al livello MAC e viceversa.

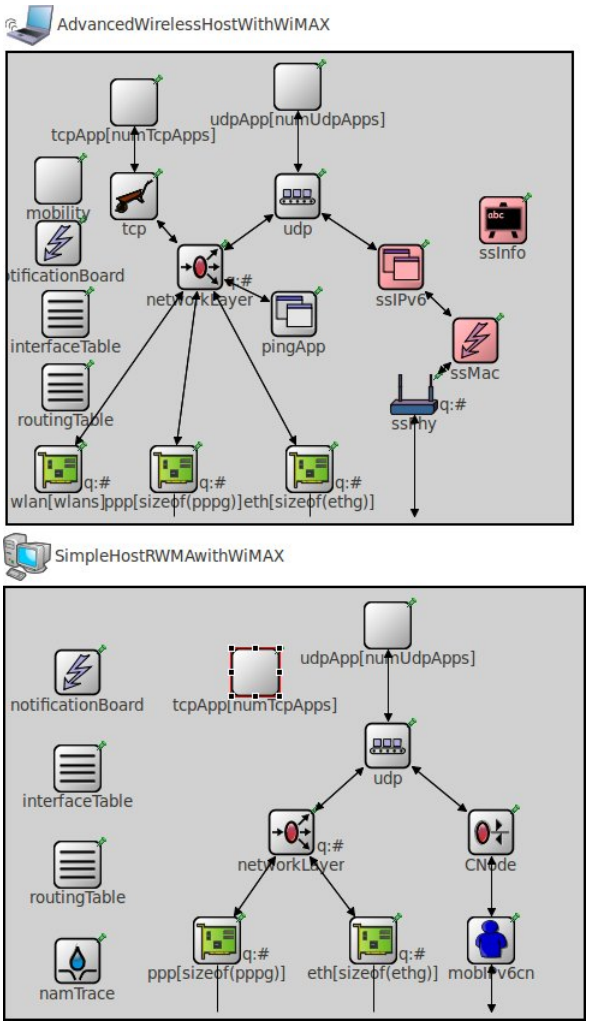

Figura 4.2: Struttura del Mobile Host e del Proxy Server

4.3 Gestione DHCP

Sia nel caso che il nodo mobile utilizzi un'interfaccia WiFi, sia che usi quella WiMAX, il suo indirizzo, IPv4 o IPv6, non potrà in alcun modo essere statico. La possibilità infatti di potersi muovere da una rete WiFi all'altra, cambiando quindi Access Point, così come nella rete WiMAX il nodo può cambiare posizione e agganciarsi a una diversa Base Station, rende necessaria un'assegnazione dinamica dell'indirizzo IP.

Nel lavoro portato a termine da Marco Ciaramella, data l'assenza di una

versione già implementata di DHCP all'interno delle librerie di Inet, è stata creata una versione semplificata che, posta sopra il livello UDP, a stretto contatto quindi con l'applicazione - e non come è solito a livello IP - fosse in grado di consentire una configurazione automatica dei nodi.

In numbat, d'altro canto, i moduli di implementazione del protocollo DHCPv6 si trovano come di consueto a livello network, ma un loro inserimento dentro l'ambiente simulativo di inet ha implicato una riconsiderazione del loro funzionamento. In Inet, infatti, già a livello applicazione il modulo che implementa il load balancer deve essere a conoscenza dell'indirizzo del proprio nodo e del nodo destinazione. E' nello specifico nel modulo ULBRWMA (ULBRWMAproxyserver per il server) che, all'interno delle UDP Control Info del messaggio applicazione che viene poi incapsulato in UDP, vengono inseriti gli indirizzi IP (versione 4 o versione 6) di mittente e destinazione. La soluzione adottata risolve il problema nel seguente modo: all'effettivo assegnamento di un indirizzo al nodo mobile, una volta che quest'ultimo ha effettuato tutte le operazioni del caso, viene inviato un messaggio di tipo noto (*IPv6AddrNotify*) ai livelli superiori, e il modulo ULBRWMA registrerà questo nuovo indirizzo, potendo così cominciare le trasmissioni di dati anche utilizzando la rete IPv6 a disposizione.

Nello scenario da noi implementato è stata posta una sola Base Station di tipo WiMAX: questo sia perché avrebbe comunque ragionevolmente coperto l'intera area urbana, sia per motivi semplificativi. Questo significa che l'assegnamento dell'indirizzo IPv6 avverrà solo all'inizio della simulazione e di seguito il nodo mobile continuerà a essere connesso alla base station.

Sul lato proxy server, invece, si è fatto sì che non venissero spediti messaggi sulla rete IPv6, se non all'arrivo di un messaggio di BindingUpdate proveniente dal nodo mobile, con il quale viene informato il corresponding node, nel nostro caso il proxy server, dell'indirizzo assunto dall'altro end point.

4.4 File

Per implementare il sistema sono stati inseriti o modificati diversi file. Verranno ora elencati a seconda del loro PATH (e di conseguenza per attinenza di livello OSI), segnalando con N quelli creati in Numbat, M quelli modificati in Numbat, con U i file *non* modificati in Numbat e con I quelli già esistenti in Inet e che hanno subito modifiche.

- src/applications/udpapp/ : DHCPPacket.msg/cc/h (I)
- src/applications/udpapp/ : ULBRWMAServerProxy.cc/h (I), ULBRWMA.cc/h (I)
- src/applications/udpapp/ : UDPBasicAppForServerProxy.cc/h (I) , UDPBasicAppForMultipleNics.cc/h (I), UDPBasicApp (I)
- src/transport/udp/ : UDPRWMA.cc/h
- src/networlayer/numbatIPv6 : DHCPv6.cc/h (U), Internet.cc/h (U), ipv6disp.cc/h (M), ipv6msg.msg (M), ipv6ModifiedNode.cc/h (N), ipv6node.cc/h (M), mip6.cc/h (M), ra.cc/h (U).
- src/linklayer/ieee80216e/mac/: wmaxctrl.cc/h (U), wmaxmac.cc (U), wmaxmsg.msg (U), wmaxcs.cc/h (U)
- src/linklayer/ieee80216e/ : wmaxphy.cc/h (U), wmaxradio.cc/h (U)
- src/numbatUtil/: fsm.cc/h (U), hoinfo.h (U), ipv6.cc (U), logger.cc (U), Portable.h (U), ssinfo.h (U), ssinfo.cc/h (U), mih.msg (U).
- src/nodes/wimax: AdvancedHostWithWiMAX.ned (N), BS.ned (U), common.ned (U), SimpleHostRWMAWithWiMAX.ned (N), SS.ned (U).s

Capitolo 5

Note implementative

L'implementazione proposta utilizza i moduli di Inet sviluppati da Piero Murphy [13], Fabrizio Sabatini [14] e Marco Ciaramella. La rete e i nodi vengono configurati tramite i file di configurazione city1.ned e city1.ini.

5.1 Integrazione di Numbat in Inet

Si sono subito riscontrati problemi di compatibilità fra Numbat e Inet, non tanto nel momento di connessione dei due network, ma fin da quando si è tentato di distribuire il codice di Numbat in modo coerente rispetto alla gerarchia dei moduli di Inet. Una intensa e oculata attività di debugging con gdb, congiuntamente a valgrind, ha dimostrato come, in maniera del tutto sorprendente, la compilazione da parte di Omnet++ generasse errori piuttosto grossolani in alcune chiamate a metodi di certi messaggi. Nello specifico, al momento delle chiamate dei metodi che assegnavano indirizzi IP di mittente e destinatario e marcavano il messaggio come BindingUpdate (si trattava quindi dei metodi setter: getDstIP(), getSrcIP(), setBindingUpdate()), l'esecuzione del codice saltava ad istruzioni di altri moduli e non al corpo del metodo della classe desiderato. Questo portava, al momento dell'incapsulamento dei primi pacchetti IPv6, a un inevitabile segmention fault.

Dopo svariati tentativi, la soluzione ottenuta per superare questo ostacolo è stata quella di introdurre un tipo di pacchetto differente per ogni tipo di messaggio che veniva spedito, distaccandoci quindi dalla versione originale, che prevedeva lo stesso tipo di *packet*, a cui veniva semplicemente assegnato in fase di istanziazione un nome differente per ogni messaggio, che fosse di dati, bindingUpdate, Routing Advertisement o Ack. Abbiamo così modificato il file ipv6msg.msg introducendo 4 nuovi pacchetti che estendevano il già presente e nativo pacchetto IPv6. Cosi' facendo non solo abbiamo risolto il nostro problema iniziale ma si è data anche una più chiara idea di quello che succedeva durante la creazione e trasmissione dei pacchetti nella rete.

Non è escluso che problemi simili si possano ripresentare in futuro in uno sviluppo o integrazione del progetto "Numbat", dal momento che il reale motivo di quanto accaduto, se non legato a questioni di compatibilità fra le versioni di Omnet++ usate, è tuttora sconosciuto.

5.2 Gestione di src e dest address

Abbiamo già accennato al fatto che ai livelli UDP e Applicazione vengano utilizzate classi che permettano di salvare indirizzi IP, indipendentemente dal fatto che siano IPv4 o IPv6. Si tratta di IPvXAddress, che ci ha permesso quindi di mantenere i moduli UDPRWMA, ULBRWMA e ULBRWMAserver-proxy senza sostanziali modifiche nel loro comportamento. Sul nodo mobile è stato possibile gestire la politica di decisione su quale tipo di network IP usare utilizzando le stesse funzioni del metodo, apportando solo quelle modifiche necessarie affinché la scelta fosse coerente con l'indirizzo destinazione appena scelto, ma soprattutto con l'attuale disponibilità di un indirizzo IPv6. Sull'altro end system, quello in cui è collassato il server, le operazioni usate per ottenere i propri indirizzi, oltre che essere abbastanza macchinose, facevano uso di strutture che non comprendevano la scelta di un indirizzo che non fosse IPv4. E' stato perciò necessario adottare una soluzione poco elegante che imponesse la scelta della tipologia di network non in base a fattori random, come quindi era la politica nel nodo mobile, ma sulla base di un flag che alternativamente facesse fluire il traffico verso l'una o l'altra rete.

5.3 Gestione del movimento

La gestione del movimento della subscriber station in Numbat, come già descritto in precedenza, prevede traiettorie circolari o rettilinee per il nodo mobile e l'aggancio di quest'ultimo a una diversa base station in base o a una vicinanza maggiore rispetto alla base station attuale o al trascorrere di un determinato periodo di tempo dopo il quale, per puro esercizio di handover, il nodo si collega a una diversa base station. Non era infatti obiettivo dei progettisti di Numbat implementare una gestione della mobilità avanzata, come quella che si può trovare in Inet, quanto simulare un meccanismo di handover il più possibile simile alla realtà. In Inet, infatti, sono stati sviluppati da tesisti passati modelli che simulano il movimento all'interno di un contesto in cui sono presenti ostacoli di varia natura, ostacoli la cui presenza influisce sulla potenza del segnale WiFi.

Per poter comunque avere una prima simulazione di uno scenario in cui un nodo mobile avesse a disposizione due diverse interfacce di diversa tecnologia, è stata assunta una semplificazione: è stato disabilitato del tutto il movimento proprio dei moduli di Numbat, rimanendo quindi con il ben più raffinato modello di movimento presente in Inet.

Un possibile sviluppo futuro consisterà nel mantenere i moduli di mobilità di Inet, aggiungendo però chiamate alle funzioni di movimento di Numbat, cosicché queste ultime non provino ad aggiornare la posizione del nodo nella grafica utilizzata, ma aggiornino internamente la posizione del nodo in questione, così da lasciare inalterata e pienamente funzionante la politica di handover scelta.

Parallalelamente, sarebbe utile includere una gestione del decadimento del segnale in presenza di ostacoli, come già presente nella versione estesa di Inet, e che in Numbat era stata omessa in toto.

Capitolo 6

Test e Valutazioni

In questo capitolo verrà spiegato come poter compilare il codice ed eseguire le simulazioni presenti. Prerequisito necessario affinché le operazioni che verranno introdotte vadano a buon fine, è avere sul proprio sistema il framework OMNeT++ correttamente compilato e configurato. Infine, nella seconda parte verrà data una valutazione sul lavoro svolto e consigli sui possibili sviluppi per raffinare la simulazione.

6.1 Compilare il codice

Per compilare il codice (ad esempio a seguito di una modifica dei file), bisogna invocare lo script **makeandcopy**, che si occupa di compilare ed infine copiare l'eseguibile nella cartella di destinazione.

Se, invece, si creano dei nuovi file, il makefile va aggiornato lanciando il comando **make -f makemakefiles** e poi invocare lo script precedente per avere il sistema funzionante.

Un possibile problema che si può riscontrare nel lancio del comando sopra citato, è dato dalla presenza della documentazione del codice all'interno della cartella **doc**. Nel caso il comando fallisca, si può spostare la documentazione fuori dalla cartella principale e riprovare nuovamente.

Se ad essere modificati sono solo file ned (come nel caso si intenda usare un modulo al posto di un'altro) o file ini, non è necessario ricompilare, ma

basta lanciare nuovamente la simulazione.

Alternivamente, è possibile la compilazione attraverso l'interfaccia grafica di Omnet++, sicuramente più facile e per la quale non sono necessarie le accortezze di cui sopra.

In ultimo, eventuali operazioni di debugging sono effettuabili sia con l'apposito CDT (C/C++ Development Tooling) di Eclipse, già integrato in Omnet++, sia attraverso la più tradizionale linea di comando lanciando gdb.

6.2 Simulazioni

La simulazione creata per testare il sistema, si trova nella cartella /examples/wimax/, nel file ini: `city1.ini`

Per la simulazione del'ambiente urbano è stata creata una configurazione nel file `city1.ini` chiamata NetworkCity, che fa riferimento al **network** definito nel file city1`.ned`.

Per facilitarne il lancio, è stato creato, all'interno della cartella, uno script che evita all'utente di dover specificare il percorso dell'eseguibile di INET. Quindi, per eseguire una simulazione, basta lanciare il comando:

`./run <inifile.ini>`

Eseguendo la simulazione col comando `./run city1.ini` viene mostrata una finestra contenente la configurazione **NetworkCity**:

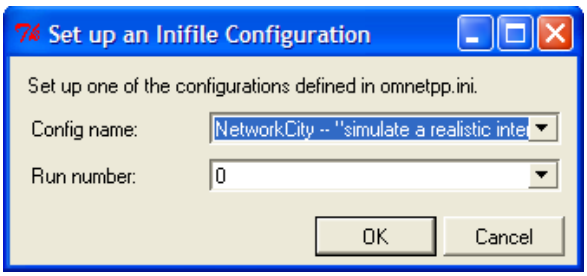

Figura 6.1: Menù di scelta della simulazione

6.2.1 La simulazione NetworkCity

Lo scenario **NetworkCity** è costituito dalle due reti IPv4, wireless e wired, in cui sono presenti i vari host, più la nostra rete IPv6 formata da alcune subscriber station che comunicano con il loro corresponding node, una base station e il componente "Internet", che funge da instradatore dei pacchetti ai nodi fissi. Il dispositivo wireless mobile si muove lungo un percorso predefinito, passando tra un certo numero di ostacoli disposti così da rappresentare il perimetro di edifici in un contesto urbano. Nella rete dell'host mobile sono inoltre presenti 5 Access Point, collocati in modo tale da fornire una copertura disomogenea dello scenario, facendo sì che l'host mobile passi per aree con intensità di segnale diversa, e una Base Station che il nodo mobile utilizza a sua discrezione.

Nella simulazione si mostra il comportamento nella rete costruita in presenza di più host comunicanti, dove il nodo mobile comunica con il proxy server attraverso un'interfaccia WiFi e l'interfaccia WiMAX.

La configurazione utilizzata è costituita da:

- Periodo di tempo simulato: $3600s$
- Canali
  - AP: statico, scelto con distribuzione uniforme nell'intervallo $[1, 11]$
  - Host mobile: scansiona tutti i canali tra 1 e 11
- Host Mobile:
  - Applicazione: **UDPBasicAppForMultipleNics**
  - Dimensione dei messaggi: $100B$
  - Frequenza dei messaggi: $40ms$
  - Indirizzo IPv4 temporaneo: 64.64.64.64
  - Indirizzo IPv4 del server DHCP: 128.128.128.128
  - Indirizzo IPv6 del server proxy: 3100 :: 1
- Server Proxy:

- Applicazione: **UDPBasicAppForServerProxy**

- Dimensione dei messaggi: $100B$

- Frequenza dei messaggi: $40ms$

- Server DHCP:

  - Applicazione: **UDPBasicAppForServerProxy**

  - Indirizzo IP del client: 64.64.64.64

  - Indirizzi IP per i client DHCP (sottorete wireless 1):

  `128.96.4.132 128.96.4.133 128.96.4.134 128.96.4.135 128.96.4.136`

  - Indirizzi IP per i client DHCP (sottorete wireless 2):

  `128.96.4.4 128.96.4.5 128.96.4.6 128.96.4.7 128.96.4.8`

  - Indirizzi IP per i client DHCP (sottorete wireless 3):

  `128.96.6.3 128.96.6.4 128.96.6.5 128.96.6.6 128.96.6.7`

  - Indirizzi IP per i client DHCP (sottorete wireless 4):

  `128.96.5.6 128.96.5.7 128.96.5.8 128.96.5.9 128.96.5.10`

  - Indirizzi IP per i client DHCP (sottorete wireless 5):

  `128.96.5.132 128.96.5.133 128.96.5.134 128.96.5.135 128.96.5.136`

  - Proprio indirizzo IPv6: 3100 :: 1

- Mobilità

  - Modulo: **TurtleMobility**

  - Percorso: descritto nel file `path.xml`

  - Velocità: scelta con distribuzione uniforme nell'intervallo $[1m/s, 2m/s]$ per ogni segmento del percorso

6.2.2 Alcuni risultati

Per valutare l'effettiva ricezione dei pacchetti IPv6 da parte del nodo mobile e del proxy server, sono stati effettuati test che consentissero di analizzare

i tempi di latenza in entrambe le direzioni. I risultati si sono rivelati simili dall'una e dall'altra estremità della comunicazione.

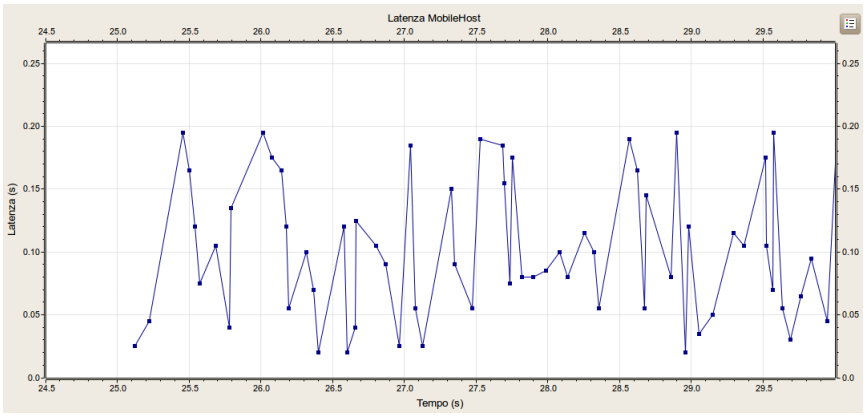

Figura 6.2: Latenza dei pacchetti ricevuti dal nodo mobile sulla connessione WiMax

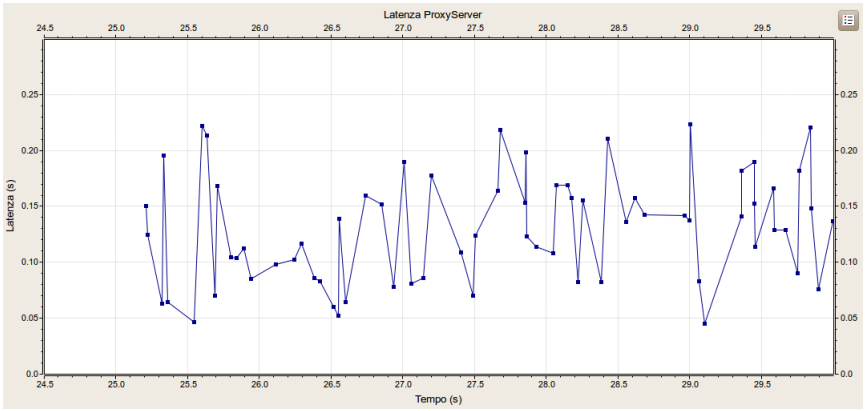

Figura 6.3: Latenza dei pacchetti ricevuti dal Proxy Server sulla connessione WiMax

6.3 Valutazione e sviluppi futuri

La simulazione creata non è che una prima approssimazione di ciò che accade in un contesto reale, dove un nodo mobile comunica usando le interfacce a disposizione con un nodo fisso attraverso una applicazione VoIP. Lo scenario creato riesce pienamente a testare sia la comunicazione nella rete IPv4, a cui il

nodo mobile invia pacchetti quando decide di usare una delle interfacce WiFi di cui è fornito, sia la comunicazione nella rete IPv6, usata invece quando si sceglie l'interfaccia WiMAX. Le semplificazioni assunte sono state descritte nei capitoli precedenti. Per ottenere una simulazione ancora più precisa, sarebbe necessario apportare alcune modifiche, che di seguito illustriamo.

6.3.1 IPv6 di Inet

Nel nostro lavoro si è utilizzato il modello di Mobile IPv6 già presente nel progetto di Numbat, modello che approssima il comportamento del livello network senza però contenere l'intero insieme delle funzionalità proprie dello standard Mobile IPv6. Si potrebbe quindi pensare di utilizzare il recentissimo modello denominato "xMIP6", sviluppato da Faqir Zarrar Yousaf della Dortmund University of Technology e Christian Bauer del German Aerospace Center, ora divenuto il modello ufficiale per Inet.

Alternativamente, nel caso si voglia utilizzare la connessione WiMAX all'interno di una rete IPv4, è necessario rendere compatibile il livello MAC di Numbat con i formati che i moduli di Inet specificano e gestiscono.

6.3.2 Restyling dei componenti

Per una maggiore flessibilità e coerenza con i componenti situati nello scenario di Inet, è pensabile di ovviare a quelle semplificazioni nella struttura dei nodi presenti in Numbat, fra cui:

- Eliminazione del livello IPv6 nella Base Station: come nel caso reale, una Base Station dovrebbe contenere solo fino al livello 2 (MAC) dello stack e lasciare ad altri componenti il compito di inoltrare il pacchetto al network a cui è connessa. In Inet questo è il ruolo di una relayUnit che, essendo di grado di comprendere i messaggi a livello MAC inviatele dall'access point, incapsula adeguatamente ogni frame, perché questo possa essere poi inoltrato alla componente ethernet, che tramite un router ad essa collegato riuscirà a instradare il messaggio al network giusto.

- Separazione fra Base Station e DHCP server: in Numbat, DHCPv6 e moduli della base station sono collassati nello stesso modulo composito. Sulla falsa riga di quanto accade in Inet, sarebbe opportuno introdurre un componente DHCPv6 separato, che sia responsabile della sottorete in cui è posizionato.

- Eliminazione del componente Internet: in Numbat, il ruolo di instradamento dei pacchetti al nodo fisso corrispondente è svolto da un componente denominato Internet, che riceve pacchetti di tipo IPv6 o dalla Base Station o dai nodi fissi e li inoltra a destinazione dopo aver consultato una tabella interna. Si tratta chiaramente di una semplificazione: la soluzione ottimale consisterebbe nell'eliminare questo modulo e inserire, anche nella rete IPv6, router e reti ethernet come già accade in Inet.

- Restyling del nodo fisso: con le modifiche appena descritte, diventa possibile usare i livelli 1 e 2 del nodo fisso anche quando si spediscono pacchetti IPv6, cosa ora impossibile. Spetterà quindi al router la gestione di pacchetti con livello network IPv4 o IPv6.

6.3.3 Divisione fra proxy server e secondo end point

Dato che si vorrebbe simulare un situazione reale in cui due nodi comunicano tramite proxy server, sarebbe consigliabile introdurre il secondo end-system nella rete, al momento emulato dal proxy server stesso, in modo da effettuare test più affidabili. L'introduzione del secondo end-sytem, pur non essendo immediata, non comporterebbe sostanziali modifiche ai moduli esistenti.

6.3.4 Protocollo DHCP

L'implementazione data del protocollo DHCP, come già illustrato, è solo una semplificazione di quella originale. Sarebbe utile modificarla in modo che rispetti le specifiche reali del protocollo. In particolare l'ideale sarebbe rendere il DHCP uns servizio indipendente dalle altre applicazioni in esecuzione nel nodo, separandolo cioè dalle funzionalità del load balancer.

6.3.5 Politica di scelta dell'interfaccia

Allo stato attuale, nodo mobile e proxy server scelgono il tipo di interfaccia da usare rispettivamente in modo random e in maniera alternata. Un naturale sviluppo sarebbe quello di privilegiare l'utilizzo dell'interfaccia WiMAX solo nel caso in cui il segnale WiFi non è presente o è non sufficientemente potente.

6.3.6 Modalità di esecuzione "run"

Per qualche motivo, l'esecuzione della simulazione in semplice modalità run genera segmentation fault nell'ambiente grafico di *tkenv* dopo che qualche decina di migliaia di eventi sono stati portati a termine. L'errore riportato su console è il seguente: *tkenv.cc#925:null or malformed pointer*. Questa anomalia rispetto all'esecuzione e di Inet e di Numbat è stata riscontrata sin dal primo momento in cui si è provato ad eseguire il codice di Numbat all'interno di Inet. La scarsa documentazione a riguardo ci ha impedito di risalire alla causa dell'errore. Si tratta comunque di un problema legato alla rappresentazione grafica della simulazione e non alla simulazione in sè: è sufficiente far partire il tutto in modalità *fast* o *express* per disabilitare l'ambiente grafico e far proseguire la simulazione senza interruzioni.

Capitolo 7

Conclusioni

Nostro compito è stato quello di costruire uno strumento simulativo in grado di contemplare reti WiFi e reti wireless a lungo raggio, in modo che un nodo mobile potesse agilmente scegliere, in base alle condizioni in cui si trovava, quale rete utilizzare per le comunicazioni. Dopo uno studio sugli strumenti utilizzabili, si è deciso di partire da un simulatore di reti wimax denominato *numbat*. Durante l'integrazione di Numbat all'interno di inet si sono subito riscontrati problemi di compatibilità, probabilmente dovuti alle diverse versioni di Omnet++ nelle quali i due progetti sono stati sviluppati. Il risultato del lavoro ha permesso di condurre le simulazioni come richiesto, nonostante alcuni aspetti possano essere migliorati, fra cui in primis l'adattamento del modello di movimento di Numbat a quello di Inet e l'inserimento di un modello simulativo di IPv6 più completo.

Il comportamento dei moduli sviluppati si è comunque dimostrato essere come da aspettative.

Bibliografia

[1] IEEE Std. 802.11b, "Higher-Speed Physical Layer (PHY) extension in the 2.4 GHz band", IEEE Standard for Information Technology, 1999

[2] IEEE Std. 802.11g, Further Higher-Speed Physical Layer Extension in the 2.4 GHz Band , IEEE Standard for Information Technology, 2003

[3] IEEE Std. 802.11n, Website, 2009, <http://standards.ieee.org/announcements/ieee802.11n 2009amendment ratified.html>

[4] IEEE Std. 802.11n, "Higher Throughput Improvements using MIMO", IEEE Standard for Information Technology, 2007

[5] IEEE Std. 802.11e, Website, 2008, <http://standards.ieee.org/getieee802/download/802.11e-2005.pdf>

[6] IEEE Std. 802.11e http://standards.ieee.org/getieee802/download/802.16e-2005.pdf

[7] ANSI/IEEE Std 802.11, 1999 Edition

[8] OMNet++ staff, OMNet++ v4.0 User Manual, <http://www.omnetpp.org/doc/manual/usman.html>

[9] OMNet++ v4.0, <http://www.omnetpp.org/omnetpp>

[10] J. Postel, "RFC768 - User Datagram Protocol", Website, 1980, <http://www.faqs.org/rfcs/rfc768.html>

[11] Information lifornia, Sciences "RFC791 Institute - University Internet Protocol", of Southern Websiste, Ca- 1981, <http://www.faqs.org/rfcs/rfc791.html>

[12] INET staff, INET Framework Model Documentation, Websiste, <http://inet.omnetpp.org/doc/INET/neddoc/index.html>

[13] Piero Murphy, "Uno Strumento Simulativo per Architetture VoIP per Dispositivi Mobili Multihomed", Università degli Studi di Bologna,Non pubblicato

[14] Fabrizio Sabatini, "Un simulatore per un protocollo di QoS per reti wireless", Università degli Studi di Bologna, Non pubblicato

[15] Numbat, <http://klub.com.pl/numbat/>

[16] Sassan Ahmadi, "Introduction to Mobile WiMAX Radio Access Technology: PHY and MAC Architecture", <http://vivonets.ece.ucsb.edu/ahmadiUCSB_abstract_Dec7.pdf>

[17] Tomasz Mrugalski and Jozef Wozniak, "Numbat – extensible simulation environment for mobile, IPv6 capable IEEE 802.16 stations", <https://klub.com.pl/numbat/doc/numbat-atnac2007.pdf>

[18] J.G. Andrews, A. Ghosh, R.Muhamed, "Fundamentals of WiMAX: understanding broadband wireless networking", Prentice Hall

[19] Marco Ciaramella,"Scenario simulativo per VoIP da dispositivi mobili basato su proxy server SIP/RTP", Non pubblicato