Alma Master Studiorum · Università di Bologna

Scuola di Scienze
Corso di Laurea Magistrale in Informatica

Confronto simulativo tra architetture per la mobilità: analisi simulatore MIPv6 e confronto con ABPS

Relatore:
Chiar.mo Prof.
Vittorio Ghini

Presentata da:
Giovanni Sitta

Sessione III
Anno Accademico 2014/2015

Ringraziamenti

Desidero ringraziare tutti coloro che, in un modo o in un altro, mi hanno aiutato nella realizzazione della mia tesi.

Anzitutto ringrazio ovviamente il Prof. Vittorio Ghini, che ha supervisionato lo studio oggetto del presente elaborato.

Proseguo con il resto del personale dell'Università degli Studi di Bologna: docenti, assistenti e tecnici, che hanno sempre saputo rispondere alle mie esigenze in caso di dubbi e problemi, non solo nel periodo di tirocinio e di stesura della tesi, ma anche durante l'intero Corso di Studi.

Un ringraziamento speciale va poi al mio collega Edoardo Convertino, che ha collaborato con me nelle fasi iniziali di approfondimento delle architetture di rete esaminate, e nell'analisi dei relativi simulatori.

Vorrei infine ringraziare tutti gli altri miei compagni di corso, e le persone a me più care, la mia famiglia e i miei amici, per il supporto che mi hanno offerto in questi due anni di studi presso Unibo.

A tutti coloro che ho citato va la mia gratitudine, ma desidero precisare che ogni eventuale errore o imprecisione riscontrato è imputabile soltanto a me.

Indice dei contenuti

1. Introduzione **p. 1**
- 1.1. Presentazione del lavoro svolto p. 1
- 1.2. OMNeT++ e INET p. 2
- 1.3. Architetture di supporto alla mobilità p. 3
  - Concetti generali della mobilità
  - MIPv6
  - LISP
  - ABPS

2. MIPv6 – Mobile IPv6 **p. 9**
- 2.1. Panoramica del protocollo MIPv6 p. 9
- 2.2. Terminologia e concetti chiave di MIPv6 p. 11
- 2.3. Approfondimento sui meccanismi di MIPv6 p. 14
  - Return routability
  - Route optimization
  - Returning home

3. LISP – Locator/Identifier Separation Protocol **p. 17**
- 3.1. Panoramica del protocollo LISP p. 17
- 3.2. Terminologia e concetti chiave di LISP p. 19

4. ABPS – Always Best Packet Switching **p. 21**
- 4.1. Panoramica dell'architettura ABPS p. 21
- 4.2. Approfondimento sugli elementi dell'architettura ABPS p. 23
- 4.3. Terminologia e concetti chiave di ABPS p. 25

5. Simulatori **p. 27**
5.1. Lavoro effettuato con OMNeT++ e INET p. 27
5.2. Modifiche comuni p. 28
- Modello di attenuazione del segnale
- Calcolo della latenza
- Calcolo degli intervalli di indisponibilità

5.3. Simulatore MIPv6 p. 31
- Porting degli ostacoli
- Modifiche apportate e correzione problemi
- Scenario di prova – testMIPv6

5.4. Simulatore LISP p. 38
- Scenario di prova – testLISP

5.5. Simulatore ABPS p. 40
- Modifiche apportate e correzione problemi
- Scenario di prova – testABPS

6. Risultati sperimentali **p. 47**
6.1. Scenari urbani p. 47
6.2. VoWiFiNetworkCity - MIPv6 p. 48
6.3. VoWiFiNetworkCity - LISP p. 54
- CN-in-LISP-site
- CN-in-non-LISP-site

6.4. VoWiFiNetworkCity - ABPS p. 60
- External-Proxyserver
- ProxyServer-near-MN
- CN-near-ProxyServer

6.5. Considerazioni conclusive p. 71
6.6. Sviluppi futuri p. 74

- **Fonti bibliografiche e sitografia** **p. 77**

1. Introduzione

1.1 Presentazione del lavoro svolto

Il presente elaborato, redatto dal sottoscritto Giovanni Sitta, tratta il lavoro di studio, analisi e sperimentazione effettuato in conclusione al Corso di Laurea Magistrale in Informatica presso l'Università degli Studi di Bologna, svolto sotto la costante supervisione del Prof. Vittorio Ghini, e con la collaborazione nelle sue fasi iniziali del collega Edoardo Convertino.
Questo ha dapprima previsto un periodo di approfondimento di alcune architetture di supporto alla mobilità dei terminali di rete, in particolare di due protocolli allo stato dell'arte, *Mobile IPv6* (*MIPv6*) e *Locator/Identifier Separation Protocol* (*LISP*), e di una terza architettura sperimentale denominata *Always Best Packet Switching* (*ABPS*).
Sono stati in seguito esaminati tre simulatori, uno per ciascuna architettura di supporto alla mobilità studiata, realizzati come estensioni della libreria *INET* del framework *OMNeT++*, assicurandosi che fossero conformi alle specifiche del protocollo implementato (almeno entro i limiti di semplificazione rilevanti ai fini del lavoro), e correggendone eventuali problematiche, mancanze e anomalie in caso questi non le rispettassero.
Sono poi stati configurati alcuni scenari simulativi utilizzando le tre librerie, in prima battuta di natura molto semplice, utilizzati per verificare il corretto funzionamento dei simulatori in condizioni ideali, e successivamente più complessi, allestendo un ambiente di esecuzione più verosimile, dotato di un maggior numero di host connessi alla rete e di ostacoli per i segnali radio usati nelle comunicazioni wireless.
Tramite i risultati sperimentali ottenuti da queste simulazioni è stato infine possibili realizzare un confronto tra le prestazioni di MIPv6, LISP e ABPS.

1.2 OMNeT++ e INET

OMNeT++ è un framework simulativo, basato su componenti modulari programmate in linguaggio C++, principalmente utilizzato nella simulazione di reti [9].
Esso non è un simulatore in senso stretto, quanto più una piattaforma su cui poter implementare nuovi modelli di reti, o importare in maniera completamente modulare quelli già esistenti.
Questi sono definiti dall'utente tramite *NED* (*Network Description*), il linguaggio di descrizione della topologia della rete utilizzato dal simulatore, che presenta le seguenti caratteristiche principali:

- gerarchia di moduli e sotto-moduli;
- basato su componenti;
- permette l'uso di interfacce;
- ereditarietà dei moduli;
- struttura a package.

In particolare, il modello su cui è stato effettuato lo studio qui presentato è INET Framework [10], una delle principali librerie disponibili per OMNeT++, che consente di realizzare scenari simulativi per reti cablate, wireless e mobili.
Esso fornisce infatti i moduli relativi a svariati tra i principali protocolli di comunicazione di rete, andando così a creare un'implementazione di tutti i livelli dello stack ISO/OSI, compreso il livello fisico.
Sono inoltre già presenti in INET alcune configurazioni di reti d'esempio, che l'utente può utilizzare come punto di partenza da modificare, o semplicemente estendere.
Nel tempo, la comunità degli utenti ha sviluppato numerose estensioni di INET relative a protocolli e componenti di rete non ancora presenti nella versione base della libreria; tra queste sono presenti anche due dei simulatori che sono stati utilizzati nello studio:

- l'implementazione di MIPv6 è descritta in [4], ed è poi stata integrata nella release ufficiale a partire dalla versione 1.99.1;
- l'implementazione di LISP è descritta in [5] ed è basata su INET-20100323.

Allo stesso modo, il simulatore per ABPS è stato realizzato a partire da INET-20080920, e progressivamente aggiornato da svariati tirocinanti e tesisti che hanno precedentemente lavorato sotto la supervisione del Prof. Ghini; in particolare, l'analisi è stata effettuata sulla versione utilizzata dal collega Luca Regazzi nel proprio lavoro di tesi [6].

1.3 Architetture di supporto alla mobilità

• Concetti generali della mobilità

Quelle che sono state prese in esame durante lo svolgimento dello studio qui presentato sono solo alcune tra le numerose architetture di supporto alla mobilità esistenti.

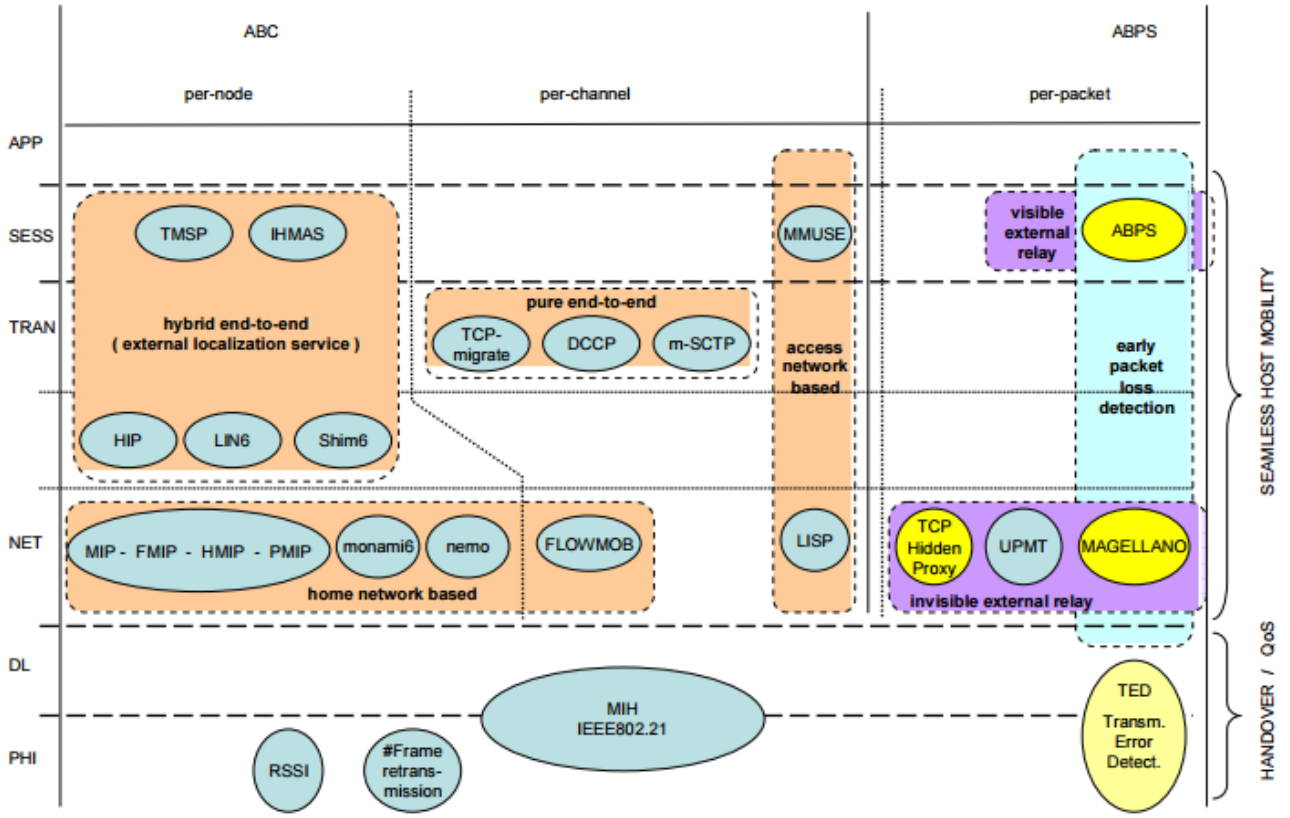

Fig. 1 – Architetture di supporto alla mobilità: state of the art

Tutte queste, per quanto sostanzialmente differenti nel proprio funzionamento e nelle proprie particolarità, presentano comunque svariati elementi comuni, in quanto legati alla natura stessa della mobilità dei terminali.

Prima di soffermarsi nello specifico sulle singole architetture, è dunque opportuno introdurre alcuni di questi concetti generali.

- ***Mobilità***

  Un dispositivo dotato di supporto alla mobilità deve essere in grado di mantenere attive le proprie comunicazioni anche durante gli spostamenti da una rete ad un'altra; differisce quindi dal concetto di *nomadicità*, che prevede più semplicemente di poter

effettuare comunicazioni da punti diversi, senza però richiederne la continuità.

- ***Mobile Node (MN)***

  Come “nodo mobile” si intende un qualunque dispositivo dotato di supporto alla mobilità, che si serve di tecnologie wireless per connettersi alla rete, siano queste a lungo raggio (tecnologie cellulari) o a corto raggio (principalmente Wi-Fi).

- ***Correspondent Node (CN)***

  Ai fini della discussione sulla mobilità, per correspondent node si intende un qualunque terminale che ha instaurato una comunicazione con un mobile node, sia esso un nodo fisso o a sua volta mobile.

- ***Multi-Homing***

  Un nodo mobile è fornito di supporto per il multi-homing in caso questo sia dotato di due o più interfacce di rete (Network Interface Controller, NIC), indipendentemente da quale tipo di tecnologia di comunicazione queste supportino; l'esempio più comune è la presenza di un'interfaccia Wi-Fi e di una che usi tecnologie cellulari (es. UMTS).

- ***Handover***

  Procedura che coinvolge il cambiamento del canale attualmente utilizzato dalla comunicazione da un nodo mobile, in genere come conseguenza dei suoi spostamenti.

  - *Handover orizzontale:* avviene tra due reti che fanno uso della stessa tecnologia (es. due reti Wi-Fi).
  - *Handover verticale :* coinvolge NIC che sfruttano tecnologie differenti (es. una rete cellulare e una rete Wi-Fi).
  - *Handover di Layer 2:* passaggio da una connessione link-layer ad un'altra (es. passaggio ad un nuovo access point).
  - *Handover di Layer 3:* avviene in seguito ad un handover di L2 quando sul nuovo link viene rilevato un prefisso di rete differente dal precedente, ed è quindi richiesto di ottenere un nuovo indirizzo (es. connessione ad un altro router di accesso in seguito al cambio di access point).

- ***Firewall e Network Address Translation (NAT)***

  Un firewall è un apparato di rete che filtra tutti i pacchetti in entrata e in uscita da e

verso un host, applicando regole che contribuiscono alla sicurezza della comunicazione.

NAT è una tecnica di modifica degli indirizzi e le porte dei pacchetti in transito, implementata su router o firewall; a questi viene assegnato un indirizzo IP pubblico, quello del gateway della rete, e viene generato un mapping *porta-IP* che associa la porta del gateway utilizzata nella comunicazione con l'esterno all'indirizzo dell'host situato nella sottorete locale: il gateway potrà quindi reindirizzare i pacchetti in arrivo su tale porta all'host corrispondente.

Un firewall è detto *simmetrico* in caso permetta ad un nodo esterno di contattare un nodo interno soltanto se precedentemente è già avvenuta una comunicazione nella direzione opposta, ovvero dal nodo protetto dal firewall verso quello situato in un'altra rete.

In caso due host entrambi protetti da firewall simmetrici tentino di comunicare, questi falliranno nel tentativo di contattare l'altro, in quanto i due firewall bloccheranno le comunicazioni avviate dall'esterno; la soluzione in questo caso è fare uso di un *relay* esterno, un nodo raggiungibile tramite un indirizzo pubblico e non protetto da firewall (e quindi contattabile con successo da parte di entrambi gli host), attraverso il quale dovranno transitare tutti i pacchetti inviati in ciascuna direzione durante il corso della comunicazione.

- ***Localization service***

  Meccanismo che mantiene in memoria il mapping tra l'attuale posizione di un nodo mobile e il suo identificatore univoco; quando il MN effettua uno spostamento, lo comunica al servizio di localizzazione (*registration phase*) in modo che questo aggiorni le proprie informazioni riguardo alla sua posizione; quando il CN intende iniziare una comunicazione con un MN, o riprenderne una in seguito al cambio di indirizzo di quest'ultimo, potrà contattare il localization service (*lookup phase*) per ottenere l'indirizzo da contattare.

• MIPv6

Realizzato dalla Internet Engineering Task Force (IETF), Mobile IP (MIP) è un protocollo di comunicazione progettato come estensione di supporto alla mobilità per Internet Protocol (IP), che permette agli host che ne fanno uso di spostarsi da una rete ad un'altra pur mantenendo un indirizzo IP permanente.
Inizialmente nato come Mobile IPv4 (MIPv4), che lavora ovviamente con IPv4, ne è stata poi realizzata anche una versione compatibile con le reti IPv6, chiamata appunto Mobile IPv6 (MIPv6).

MIPv6 è l'architettura di supporto alla mobilità su cui è stato maggiormente incentrato lo studio qui presentato. Seguirà dunque una discussione sul protocollo, che non vuole certamente essere considerata come esaustiva, soprattutto tenendo conto di come molti aspetti e dettagli non siano stati effettivamente trattati durante lo studio, in quando non rilevanti ai fini degli scopi prefissati; ad esempio, gli elementi riguardanti la sicurezza del protocollo sono state in gran parte omesse dalla trattazione, in quando non concretamente presenti all'interno delle simulazioni per questioni di semplificazione.
Gli aspetti che hanno invece mostrato maggiore rilevanza per quanto riguarda il lavoro effettuato con i simulatori verranno approfonditi all'interno del capitolo 2:

- il funzionamento generale del protocollo è trattato nella sezione 2.1, così come alcune delle differenze che sono nate nel passaggio da IPv4 a IPv6;
- nella sezione 2.2 vengono esaminati i principali termini e concetti relativi a MIPv6;
- la sezione 2.3 tratta infine più nel dettaglio alcuni meccanismi specifici utilizzati da MIPv6 nella gestione della mobilità.

• LISP

Realizzato dal LISP Working Group della Internet Engineering Task Force, Locator/ID Separation Protocol è un protocollo di livello network basato sul concetto di separazione dello spazio di indirizzamento, che attualmente è formato dall'insieme degli indirizzi IP, in due spazi differenti:

- il primo, quello dei *locator*, è utilizzato per determinare il punto di accesso alla rete corrente di un client;
- il secondo, quello degli *identifier*, è usato per distinguere univocamente un client dagli altri.

La divisione delle funzionalità di instradamento e identificazione, al momento entrambe affidate agli indirizzi IP, favorirebbe la mobilità dei terminali, che rimarrebbero così individuabili anche a seguito di spostamenti in altre reti.
Concettualmente, sia locator che identifier possono essere elementi di natura arbitraria, compresi gli stessi indirizzi IP, permettendo quindi un'integrazione graduale del protocollo all'interno delle infrastrutture di rete già esistenti, senza richiedere particolari modifiche e senza il bisogno di attendere un “flag day”.

Non essendo l'oggetto centrale del lavoro effettuato, LISP sarà trattato in maniera più sommaria rispetto a Mobile IPv6; essendo comunque uno dei protocolli con cui è stato confrontato MIPv6 durante le simulazioni, a questo verrà dedicato il capitolo 3:

- la sezione 3.1 descrive il funzionamento generale di LISP;
- la sezione 3.2 ne esamina la terminologia e ne ricapitola brevemente i concetti principali.

• ABPS

Progettato presso il Dipartimento di Informatica dell'Università degli Studi di Bologna dal Prof. V. Ghini, il Prof. S. Ferretti e il Prof. F. Panzieri, Always Best Packet Switching (ABPS) è un'architettura di supporto alla mobilità concepita come alternativa a quelle esistenti, e che rispetto a queste tenta di applicare un approccio di maggiore efficienza nell'utilizzo di tutte le interfacce di rete disponibili sul terminale mobile.

Correntemente, il criterio comune è infatti quelli di scegliere una singola interfaccia come quella effettivamente utilizzata, e in caso questa dovesse disconnettersi per un qualche motivo (le cause possono spaziare dalla perdita del segnale, alle interferenze, al semplice allontanamento eccessivo del nodo mobile dal punto di accesso alla rete) tutte le comunicazioni attualmente in corso verrebbero interrotte, in attesa che questa ritorni operativa, o che si inizi ad utilizzarne un'altra al suo posto; questo implica inevitabilmente una nuova fase di configurazione della comunicazione, con tutti gli overhead che essa comporta.

ABPS monitora invece costantemente tutte le interfacce di rete presenti, ed opera con granularità di pacchetto nella selezione di quale sia la più adatta da impiegare in quel momento per lo scambio di dati, andando a minimizzare il più possibile i tempi di attesa dovuti a cadute della connessione e riconfigurazioni in seguito a cambi di rete, e favorendo dunque la continuità nelle comunicazioni dei terminali mobili che ne fanno uso.

Always Best Packet Switching verrà trattato nel capitolo 4, che è strutturato come segue:

- alla struttura e al funzionamento generale dell'architettura ABPS è dedicata la sezione 4.1;
- alcuni elementi specifici e di particolare importanza dell'architettura sono poi approfonditi nella sezione 4.2;
- i relativi termini e concetti chiave sono infine ricapitolati nella sezione 4.3.

2. MIPv6 – Mobile IPv6

2.1 Panoramica del protocollo MIPv6

Mobile IPv6 è un protocollo di rete che permette ad un terminale di mantenere attiva una comunicazione durante lo spostamento tra reti IPv6.
Senza uno specifico supporto alla mobilità per IPv6, i pacchetti destinati ad un nodo mobile non sarebbero normalmente in grado di raggiungerlo mentre questo si trova in una rete diversa dalla sua cosiddetta rete domestica (*home network*).
Per poter continuare le proprie comunicazioni anche dopo essersi spostato, un nodo mobile potrebbe cambiare il proprio indirizzo IP ogni volta che raggiunge una nuova rete, ma ciò non gli permetterebbe comunque di mantenere attive le proprie connessioni a livello di trasporto e superiori.
MIPv6 permette a un nodo mobile di spostarsi da un link a un altro senza cambiare il proprio *home address*: i pacchetti ad esso diretti vengono reindirizzati verso il suo attuale punto di connessione ad Internet, e anche dopo essersi spostati su di un nuovo link è possibile continuare le comunicazioni con gli altri nodi, siano questi fissi o a loro volta mobili; gli spostamenti avvengono quindi in maniera trasparente rispetto alle applicazioni e ai protocolli di layer di trasporto e superiori.
Mobile IPv6 tenta di risolvere i problemi di gestione della mobilità a livello network, analogamente ad esempio a come i meccanismi di handover tra access point lavorino invece a livello datalink. In ogni caso, è bene sottolineare che MIPv6 non si pone come obiettivo quello di risolvere qualunque problema nell'ambito generale della mobilità e delle reti wireless, quali:

- gestire link con connettività unilaterale o raggiungibilità parziale, come il problema del terminale nascosto, in cui un host non è visibile solo ad alcuni dei router presenti;
- controllo degli accessi sul link visitato dal nodo mobile;
- forme di gestione della mobilità gerarchiche o locali (simili a varie soluzioni esistenti a livello datalink);
- supporto per applicazioni adattive;
- mobilità dei router;
- service discovery, scoperta di dispositivi sulla rete e dei servizi offerti da questi;

- distinguere pacchetti persi a causa di errori piuttosto che per la congestione della rete.

Il funzionamento generale di Mobile IPv6 è basato, come naturale aspettarsi, su quello di Mobile IPv4, il protocollo analogo che opera sulle reti IPv4; si può comunque anticipare come siano presenti alcune differenze sostanziali tra i due protocolli, dovute ai miglioramenti offerti da IPv6, che verranno discusse in seguito.

Come per Mobile IPv4, i nodi MIPv6 fanno uso di due indirizzi:

- l'*home address*, ovvero l'indirizzo che gli era stato assegnato all'interno della propria rete domestica (home network), mantenuto invariato anche a seguito di spostamenti in altre reti (foreign networks);
- il *care-of address*, ovvero un nuovo indirizzo ottenuto in seguito allo spostamento verso una nuova rete, e che gli permette di essere identificati all'interno di questa.

Indipendentemente dalla rete corrente alla quale è collegato il nodo mobile, i pacchetti diretti ad esso possono continuare essere inoltrati all'home address: gli spostamenti del nodo mobile all'esterno della rete domestica sono quindi trasparenti rispetto ai livelli superiori dello stack protocollare. Ovviamente, se invece il nodo mobile si trova nel proprio home network, i pacchetti indirizzati al suo home address saranno inoltrati utilizzando meccanismi di routing convenzionali.
Il care-of address rappresenta invece la posizione corrente del nodo mobile al di fuori della propria rete domestica: i pacchetti indirizzati al suo home address dovranno quindi poi essere re-inoltrati al suo care-of address.
Questo meccanismo viene attuato da un nodo presente all'interno della rete domestica, chiamato home agent, al quale il terminale mobile ha registrato il suo attuale care-of address appena lo ha ottenuto. L'home agent intercetta i pacchetti destinati all'home address, e li incapsula creando un tunnel verso il care-of address precedentemente registrato. Ogni volta che il nodo mobile visita una nuova rete, esso registra il nuovo care-of address al suo home agent; tale operazione è detta binding.

Mobile IPv6 condivide numerose caratteristiche con la controparte IPv4, ma, basandosi su IPv6, tra i due protocolli di supporto alla mobilità sono ovviamente presenti delle sostanziali differenze:

- la più importante è che non vi sia la necessità di un foreign agent come in Mobile IPv4; questo è un router con funzionalità specifiche, che in MIPv4 è indispensabile dover installare in una rete per abilitare la connessione di nodi mobili e permettergli di ottenere

correttamente un care-of address; MIPv6 può invece operare in ogni rete senza alcun particolare supporto da parte del router locale, evitando modifiche all'infrastruttura e facilitandone così la diffusione;

- la modalità di route optimization è una parte fondamentale del protocollo, mentre in Mobile IPv4 questa era presente soltanto come estensione non-standard; ci si aspetta dunque che per MIPv6 essa possa essere attuata su scala globale, tra tutti i MN e CN;
- i meccanismi di Neighbor Unreachability Detection di IPv6 assicurano la raggiungibilità simmetrica tra il nodo mobile e il router di default nella sua posizione corrente;
- i pacchetti inviati all'esterno della rete domestica in MIPv6 sono inoltrati utilizzando un routing header IPv6 anziché utilizzando l'incapsulamento, riducendo l'overhead rispetto a MIPv4;
- MIPv6 fa uso della Neighbor Discovery IPv6 invece del protocollo ARP; questo gli permette di essere indipendente dal livello datalink, aumentando la robustezza del protocollo.

2.2 Terminologia e concetti chiave di MIPv6

- ***Home network***

  La rete "domestica" del nodo mobile, quella a cui è abitualmente connesso e tramite la quale è sempre rintracciabile.

- ***Home subnet prefix***

  Il prefisso di rete corrispondente all'home address di un nodo mobile.

- ***Home link***

  Il link su cui è definito l'home subnet prefix di un nodo mobile.

- ***Home address (HoA)***

  Indirizzo di rete assegnato al nodo mobile in maniera permanente, e che persiste anche a seguito dei suoi spostamenti all'interno di altre reti. Permette alle comunicazioni di raggiungere il MN anche quando questo utilizza un punto di accesso ad Internet diverso dal proprio home link.

- ***Foreign network***

  Una qualunque rete utilizzata dal MN che non sia il suo home network.

- ***Foreign subnet prefix***

  Il prefisso di rete corrispondente ad una qualunque rete utilizzata dal MN che non sia il suo home network.

- ***Foreign link***

  Un qualunque link utilizzato dal MN, all'interno di un foreign network, che non sia il suo home link.

- ***Care-of address (CoA)***

  Indirizzo di rete che viene associato al nodo mobile quando questo visita un foreign network; il suo prefisso di rete è un foreign subnet prefix.

- ***Home agent (HA)***

  Router sull'home link di un MN, al quale ha registrato il suo attuale care-of address.

- ***Binding (HoA ↔ CoA)***

  L'associazione tra l'home address di un MN ed un suo care-of address, compresa la durata che questa dovrà avere.

- ***Registration (Binding Update to HA/CN)***

  Operazione tramite la quale viene salvato un binding tra HoA e CoA di un nodo mobile; avviene tramite un messaggio di Binding Update, inviato dal MN al proprio home agent oppure al correspondent node con cui sta comunicando.

- ***Return routability***

  Procedura che permette al CN di verificare che il MN sia effettivamente raggiungibile tramite l'indirizzo che dichiara come proprio care-of address, oltre che tramite il proprio home address. Consiste in una fase di autenticazione basata sullo scambio di chiavi.

- ***Route optimization***

  Modalità di invio dei pacchetti che non richiede di transitare attraverso l'home agent, rendendo dirette le comunicazioni tra mobile node e correspondent node; per verificare che questa possa essere effettivamente attuata, è richiesto che sia stata precedentemente eseguita la procedura di return routability tra i due endpoint.

- ***Returning home***

  Procedura con cui il mobile node comunica al proprio home agent di essere rientrato nella propria home network; l'HA può quindi smettere di inoltrare al suo precedente

care-of address i pacchetti diretti al MN, in quanto quest'ultimo è nuovamente in grado di ricevere dati utilizzando il proprio home address.

- ***Messaggi di binding***
  - ***Binding Update***

    Utilizzato da un nodo mobile per notificare un binding al proprio home agent ("home registration") o ad un correspondent node.
  - ***Binding Acknowledgement***

    Conferma di avvenuta ricezione di una Binding Update, in caso fosse stata richiesta, o notifica in caso si siano verificati problemi durante la stessa.
  - ***Binding Refresh Request***

    Utilizzato da un correspondent node per richiedere ad un mobile node di aggiornare il binding tra i due, tipicamente quando questo sta per scadere.
  - ***Binding Error***

    Utilizzato dal correspondent node e dall'home agent per notificare un errore legato alla mobilità al mobile node.

- ***Strutture dati***
  - ***Binding Cache***

    Ogni home agent e correspondent node possiede una cache locale in cui cui memorizza i binding effettuati con i mobile node e la durata residua degli stessi.
  - ***Binding Update List***

    Mantenuta dai mobile node, contiene una entry per ciascun binding effettuato o che si sta cercando di stabilire con un altro nodo, sia per un home agent che per un correspondent node, e la durata residua dello stesso.
  - ***Home Agents List***

    Mantenuta dagli home agent, contiene una entry per ciascun altro home agent connesso allo stesso link.

2.3 Approfondimento sui meccanismi di MIPv6

• Return routability

Dopo aver registrato un nuovo care-of address presso il proprio home agent tramite una Binding Update (home registration), un mobile node può fare lo stesso anche con il correspondent node che intende contattare (correspondent registration), così come con quelli cui stava già precedentemente comunicando, memorizzati nella sua Binding Update List. Questo sarebbe ovviamente ideale per sfruttare tutti i vantaggi comportati dalla route optimization.
Prima di fare ciò, tuttavia, occorre assicurarsi che una comunicazione diretta tra i due endpoint sia effettivamente possibile, e inoltre che venga stabilita una connessione sicura.
Si ricorre dunque alla procedura di return routability, che consiste nello scambio dei seguenti quattro messaggi:

- *Home Test Init (HoTI)*, inviato dal MN al CN passando per l'HA;
- *Home Test (HoT)*, inviato in risposta all'HoTI dal CN al MN, anch'esso passando dall'HA;
- *Care-of Test Init (CoTI)*, inviato dal MN direttamente al CN;
- *Care-of Test (CoT)*, inviato in risposta al CoTI dal CN direttamente al MN.

Questi vengono utilizzati per effettuare uno scambio di chiavi e instaurare così una comunicazione sicura tra CN e MN; ciò avviene tramite l'uso dei tre elementi seguenti: i cookie, i keygen token e la binding management key.

- ***Cookie***

  Numero random utilizzato durante la procedura di return routability per evitare lo spoofing delle comunicazioni da parte di malintenzionati; in particolare, si fa uso dei due cookie seguenti:

  - *home init cookie*, inviato al CN passando per l'HA nel messaggio di Home Test Init, e che deve essere restituito nel messaggio di Home Test;
  - *care-of init cookie*, inviato direttamente al CN con il messaggio di Care-of Test Init, e che deve essere restituito nel messaggio di Care-of Test.

- ***Keygen token (care-of keygen token/home keygen token)***

  Numero fornito dal CN durante la procedura di return routability per permettere al MN di generare la Binding management key. In particolare, vengono utilizzati i due seguenti token:

  - *home keygen token*, inserito dal CN nel messaggio di CoT;

- *care-of keygen token*, inserito dal CN nel messaggio di HoT.

- ***Binding management key***

  Generata dal mobile node a partire dai keygen token inviatigli dal correspondent node durante la procedura di return routability, e utilizzata per autorizzare i futuri messaggi di binding che il MN invierà al CN.

• Route optimization

Mobile IPv6 permette anche una tipologia di comunicazione tra il nodo mobile e il correspondent node che non prevede il coinvolgimento dell'home agent, denominata route optimization; questa alternativa consente l'utilizzo di un percorso più breve per l'inoltro dei pacchetti, e riduce inoltre la congestione all'home agent.

Per poter utilizzare tale modalità di invio dei pacchetti, il mobile node deve prima assicurarsi che il correspondent node abbia una entry nella propria Binding Cache che gli permetta di gestire correttamente i pacchetti anche se questi non stati inoltrati dall'home agent.

Il MN controlla dunque se nella propria Binding Update List sia presente una entry che soddisfi le seguenti condizioni:

- l'indirizzo sorgente dei pacchetti che il MN intende inviare corrisponde all'home address presente nella entry;
- l'indirizzo di destinazione è quello del CN memorizzato nella entry;
- uno dei care-of address correnti del MN corrisponde al CoA nella entry;
- la entry indica che il binding è avvenuto con successo;
- il tempo di validità residuo del binding è maggiore di zero.

Quando il MN individua una entry che soddisfi tutte queste condizioni, questo sa che il CN ha una una entry corrispondente al medesimo binding nella propria Binding Cache, e può dunque procedere a comunicare con esso in maniera diretta; per fare ciò, dovrà costruire i propri pacchetti nella maniera seguente:

- generare il pacchetto nella stessa maniera che impiegherebbe se si trovasse all'interno della propria rete domestica, ovvero con l'home address come indirizzo sorgente, e i checksum dei layer superiori calcolati di conseguenza;
- inserire l'opzione Home Address, copiando il valore dell'indirizzo sorgente del pacchetto;
- modificare infine l'indirizzo sorgente sovrascrivendolo con il care-of address del MN.

In questo modo sarà l'indirizzo inserito nell'opzione Home Address ad essere effettivamente utilizzato dal CN nelle operazioni in cui tradizionalmente è impiegato il campo Source Address dell'header IPv6, così come sarà l'HoA ad essere passato ai protocolli di layer superiore e alle applicazioni in esecuzione sul CN che lo richiedono.

• Returning home

Quando il mobile node si connette ad un link avente come prefisso di rete il suo home subnet prefix, questo sa di essere tornato nuovamente nella propria rete domestica. Ciò significa che il MN è nuovamente in grado di ricevere comunicazioni attraverso il proprio home address, senza più dover dipendere dall'inoltro dei pacchetti ad un care-of address da parte dell'home agent, e che quest'ultimo deve esserne dunque informato.
Questo avviene tramite l'invio di una Binding Update di de-registrazione all'HA da parte del MN, che consiste in una home registration con lifetime pari a zero.
A questo punto, il tunnel di comunicazione che passava attraverso l'home agent cesserà di essere operativo, e sarà nuovamente il mobile node stesso a gestire i pacchetti in arrivo che hanno come destinazione il suo home address.

3. LISP – Locator/ Identifier Separation Protocol

3.1 Panoramica del protocollo LISP

LISP (Locator/Identifier Separation Protocol) è un protocollo di livello network concepito per rispondere ai problemi legati alla scalabilità e alla flessibilità dei meccanismi di indirizzamento e routing su Internet.
Il concetto chiave alla base di LISP è quello della separazione degli indirizzi IP tradizionali in due nuovi spazi di indirizzamento:

- *Endpoint Identifier* (*EID*), usato per distinguere univocamente un host, indipendentemente dalla topologia del network in cui si trova, ma che non fornisce funzionalità di routing a livello globale;
- *Routing Locator* (*RLOC*), topologicamente assegnato ai punti di accesso ai network, e utilizzato per l'indirizzamento e l'inoltro dei pacchetti.

La divisione di Locator e Identifier permette di far fronte al problema dell'esaurimento dello spazio di indirizzi IPv4, garantendo migliore scalabilità per i sistemi di routing, e semplifica la connessione di un nuovo host a una rete: questo sarà sempre identificato dal proprio Endpoint Identifier, indipendentemente dal network in cui si trova al momento (che sarà invece globalmente indirizzabile tramite uno o più Routing Locator), favorendo così la mobilità dei terminali.
Sia EID che RLOC sono sintatticamente identici ai normali indirizzi IP, differenziandosi da questi invece nella semantica con cui vengono usati; gli end-system possono quindi continuare ad inviare i propri pacchetti come hanno sempre fatto, mentre sarà la rete LISP sottostante ad occuparsi di individuare il destinatario e fargli arrivare correttamente le informazioni inviate dal mittente.
Per fare in modo che ciò avvenga, quando un host tenta di contattarne un altro (situato in un differente dominio), i suoi pacchetti vengono intercettati dal gateway LISP della rete del mittente, che funge da ingress tunnel router (ITR) nel creare un tunnel di comunicazione con il gateway LISP della rete del destinatario, detto egress tunnel router (ETR).
Il tunneling LISP avviene tramite l'aggiunta di un header esterno ai pacchetti del mittente da parte dell'ITR, che usa il proprio RLOC come indirizzo sorgente, e quello dell'ETR come indirizzo di destinazione; il routing dei pacchetti può quindi essere gestito normalmente dalla rete (in quanto gli RLOC sono globalmente indirizzabili), fino a raggiungere l'ETR, che si occuperà della rimozione

dell'header esterno e dell'inoltro del pacchetto al destinatario originale, situato nel suo dominio e individuabile tramite il proprio EID.

Alla base del meccanismo appena descritto, ovviamente, risiede la necessità di un servizio di mapping tra EID ed RLOC, che fornisca agli ITR la possibilità di venire a conoscenza di quale ETR da contattare per poter raggiungere l'host destinatario.
LISP non impone un singolo, specifico servizio di mapping, ma prevede invece la possibilità di interfacciarsi con diversi database di mapping, servendosi di due entità chiamate *map server* (*MS*) e *map resolver* (*MR*):

- gli ETR registrano al proprio map server le corrispondenze EID-to-RLOC relative a tutti gli host a cui fanno capo;
- gli ITR si rivolgeranno ai map resolver per venire a sapere quale RLOC corrisponda all'EID dell'host che devono contattare; questi inoltreranno la richiesta al sistema di mapping che si sta utilizzando, dove un server autoritativo risponderà fornendo l'EID corrispondente;
  i MS e i MR possono essere situati in nodi separati, così come all'interno degli ITR ed ETR stessi; gli ITR possono inoltre mantenere una cache locale di mapping, in modo da ridurre il carico di lavoro del sistema di mapping e velocizzare le comunicazioni tra domini LISP.

Per quanto riguarda le comunicazioni tra domini LISP e non-LISP, invece, potrebbero presentarsi problemi dovuti all'utilizzo degli EID, in quanto non globalmente indirizzabili: in caso un nodo non abilitato a comunicare tramite LISP ottenesse (ad esempio tramite DNS) l'EID di un host situato in una sottorete LISP, i pacchetti inviati a quest'ultimo verrebbero semplicemente scartati dai router di bordo di un dominio non-LISP; analogamente, un pacchetto indirizzato ad una rete non-LISP potrebbe essere scartato in caso provenisse da un host LISP, e avesse quindi un EID come indirizzo sorgente. Per ovviare a ciò, si può far uso di due tipi di entità aggiuntive:

- un proxy-ITR (PITR) segnala pubblicamente gli EID a cui fa capo tramite BGP advertisement, in modo che questi diventino globalmente indirizzabili, e si occupa poi di inoltrare i pacchetti ad allo stesso modo di un normale ITR;
- i pacchetti destinati a nodi non-LISP vengono incapsulati dagli ITR e inviati a un proxy ETR (PETR), che potrà decapsularli e inoltrarli al destinatario, aggirando così i meccanismi di filtraggio basati sull'indirizzo del mittente.

3.2 Terminologia e concetti chiave di LISP

- ***LISP site***

  Sottorete separata dalle altre da uno o più router LISP ad essa associati, utilizzati come tramite dagli host di tale rete per comunicare con l'esterno.

- ***Routing Locator (RLOC)***

  Indirizzo Ipv4 o IPv6 di un Egress Tunnel Router, globalmente raggiungibile e quindi normalmente utilizzabile nell'instradamento dei pacchetti; è inoltre l'output ottenuto da un'operazione di EID-to-RLOC mapping lookup.

- ***Endpoint ID (EID)***

  Valore numerico utilizzato per identificare univocamente un host; sintatticamente identico ad un indirizzo IPv4 o IPv6, ed ottenibile in maniera analoga (ad esempio tramite DNS); viene allocato a partire da un blocco che come prefisso di rete l'EID-Prefix associato al sito LISP in cui si trova l'host.

- ***EID-Prefix***

  Prefisso di rete assegnato ad un sito LISP, ed utilizzato nell'allocazione degli EID dei suoi host; vengono associati agli RLOC degli ETR per creare un database di mapping globale.

- ***Ingress Tunnel Router (ITR)***

  Router situato all'interno di un LISP site, che si occupa di instradare verso l'esterno i pacchetti generati dagli host di tale sito, creando un tunnel di comunicazione con l'ETR del sito in cui si trova il destinatario; per fare ciò, l'ITR incapsula i pacchetti con un header IP esterno, in cui l'indirizzo sorgente è uno dei suoi RLOC, e l'indirizzo di destinazione è il risultato della relativa operazione di EID-to-RLOC mapping lookup.

- ***Egress Tunnel Router (ETR)***

  Router che accetta pacchetti LISP incapsulati, in cui l'indirizzo di destinazione dell'header IP esterno è uno dei suoi RLOC, procedendo poi a decapsularli ed inoltrarli al destinatario dell'header IP successivo, normalmente un host all'interno del sito LISP a cui è associato.

- ***EID-to-RLOC Database***

  Database globale e distribuito, di natura relativamente statica, che contiene il

mapping tra gli EID-Prefix dei siti LISP e gli RLOC degli ETR associati a tali siti. Ogni ETR tipicamente possiede un frammento del database, ovvero i mapping relativi agli EID-Prefix che si trovano “oltre” tale router.

- ***EID-to-RLOC Cache***

  Tabella utilizzata dagli ITR per memorizzare i mapping EID-to RLOC; differisce dal database vero e proprio in quanto maggiormente dinamica, di dimensioni molto ridotte e solamente locale per l'ITR.

- ***Map Server (MS)***

  Componente dell'infrastruttura di rete che memorizza i mapping EID-to RLOC comunicatigli dagli ETR, e che li pubblica all'interno del database di mapping globale.

- ***Map Resolver (MR)***

  Componente dell'infrastruttura di rete che gestisce le richieste degli ITR, inoltrandole al database di mapping, per rispondere con l'RLOC richiesto, o con un messaggio di errore in caso l'indirizzo richiesto non facesse parte del namespace degli EID.

- ***Proxy-ITR (PITR)***

  Router utilizzato come ITR da parte di siti non-LISP per comunicare con destinazioni situate all'interno di siti LISP.

- ***Proxy-ETR (PETR)***

  Router utilizzato come ETR da parte di siti LISP per comunicare con destinazioni situate all'interno di siti non-LISP.

4. ABPS – Always Best Packet Switching

4.1 Panoramica dell'architettura ABPS

Always Best Packet Switching (ABPS) è un'architettura di supporto alla mobilità concepita per permettere al dispositivo mobile che la utilizza di sfruttare costantemente tutte le proprie interfacce di rete disponibili, decidendo al momento dell'invio di ogni singolo pacchetto quale sia la più adatta da utilizzare.

I dispositivi mobili di ultima generazione sono dotati di più Network Interface Cards (NICs), che gli permettono di accedere ad Internet tramite l'uso di diverse tecnologie (es. connessioni 3G e reti Wi-Fi); tuttavia, tali sistemi sono comunque limitati a doverne scegliere soltanto una, e continuare ad utilizzarla per inviare dati durante il corso delle proprie comunicazioni.

È evidente come questa politica non sia la migliore in termini di efficienza, rispetto ad un'ideale possibilità di sfruttare più interfacce contemporaneamente per una parallelizzazione nel trasferimento dei dati.

Vi sono poi numerose situazioni in cui un nodo mobile si troverà di fronte ad un degrado nella qualità delle proprie comunicazioni, causato da disturbi e interferenze nel segnale, che per le trasmissioni wireless sono molto più frequenti e significative che per quelle cablate: piuttosto che attendere la caduta della connessione, sarebbe preferibile sfruttare le altre interfacce per cercare preventivamente una nuova rete a cui passare, che garantisca una migliore qualità di servizio.

La natura stessa della mobilità implica inoltre l'inevitabile presenza di scenari in cui diventi impossibile mantenere attiva la connessione con un punto di accesso ad Internet, per il semplice motivo che gli spostamenti del nodo mobile possono portarlo fuori dal raggio di copertura della rete in cui si trovava. Per le tecnologie tradizionali, questo implica necessariamente un'interruzione della comunicazione con il correspondent node, in attesa di potersi connettere ad un nuovo punto di accesso, pur disponendo di altre interfacce di rete che teoricamente potrebbero sostituire quella attualmente in uso, idealmente in maniera sufficientemente rapida da mantenere attiva la comunicazione tra i due endpoint.

L'architettura ABPS è stata ideata per tentare di rispondere efficacemente a tali problematiche. Questa consiste essenzialmente di due componenti principali:

- un *proxy server*, ovvero un nodo fisso e liberamente contattabile tramite un indirizzo pubblico (non mascherato da firewall), che funge da relay per il mobile node, abilitando le comunicazioni da/verso quest'ultimo indipendentemente da eventuali fw o sistemi NAT;
- un *proxy client*, una componente software in esecuzione sul nodo mobile, che si occupa di gestire le comunicazioni con il proxy server tramite un tunnel multi-path costruito a partire da tutte le interfacce di rete del MN.

L'implementazione dei protocolli necessari al funzionamento di ABPS avviene quindi soltanto nei due proxy, senza coinvolgere le infrastrutture di rete già esistenti; non sono dunque richieste modifiche a queste ultime, che il nodo mobile potrà utilizzare liberamente a seguito dei propri spostamenti.

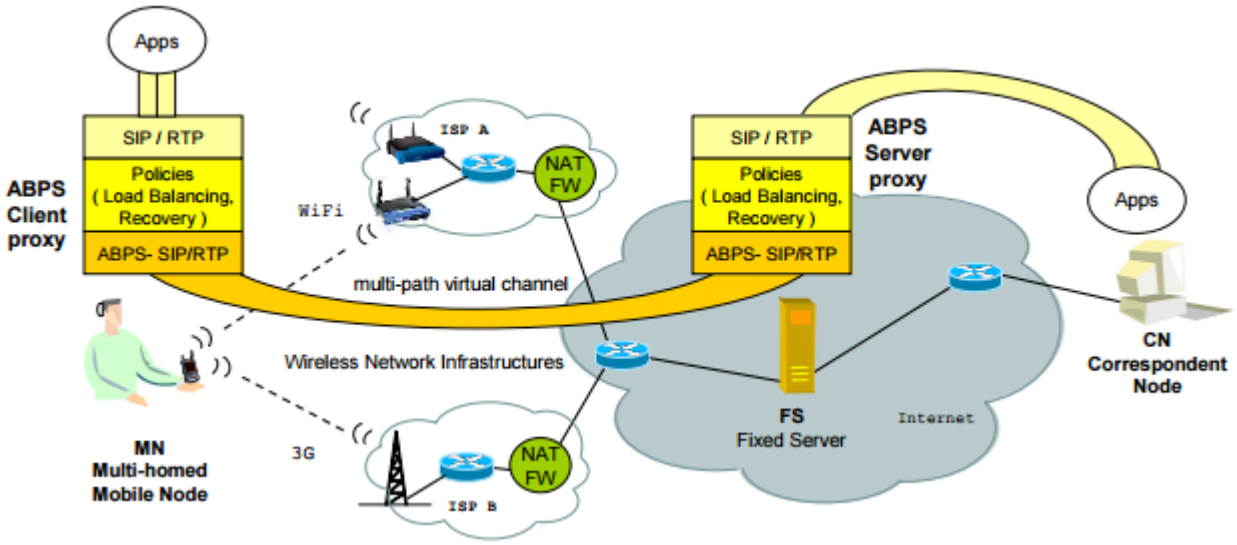

Fig. 2 – L'architettura di ABPS

ABPS è stato progettato cercando di soddisfare i tre seguenti requisiti principali:

- *garantire la continuità delle comunicazioni*;

  questo avviene sfruttando il supporto del proxy server, che continua a fungere da intermediario per lo scambio di dati tra correspondent node e mobile node, anche a seguito degli spostamenti di quest'ultimo verso nuove reti, e permettendo di aggirare le limitazioni imposte dalla presenza di eventuali firewall e NAT;

- *abilitare l'utilizzo simultaneo di tutte le NIC disponibili*;

  come anticipato, l'idea alla base di ABPS è quella di poter sfruttare al meglio tutte le interfacce di rete disponibili sul dispositivo mobile; più nel dettaglio, ciò avviene determinando quale NIC sia la più adatta da utilizzare per inviare ciascun pacchetto;
- *minimizzare la latenza tramite scorciatoie nei messaggi di segnalazione*;

  ABPS fa uso di alcune estensioni di Session Initiation Protocol (SIP) e Real Time Protocol (RTP), che introducono alcuni campi aggiuntivi necessari ad identificare univocamente e in sicurezza il mittente di un messaggio, anche quando questo cambia indirizzo IP, e favorendo quindi il supporto quindi alla mobilità dei terminali.

4.2 Approfondimento sugli elementi dell'architettura ABPS

Quello che differenzia Always Best Packet Switching dalle altre architetture di supporto alla mobilità allo stato dell'arte è l'approccio che questo ha verso le NIC multiple del mobile node, nel tentativo di sfruttarle tutte il più possibile.
Diventa dunque necessario servirsi di un modulo software di monitoraggio di tali interfacce, detto appunto *monitor* del nodo mobile, che si assicuri che funzionino tutte correttamente, e che informi il sistema in caso una di queste non sia più utilizzabile; tutto questo avviene ovviamente in background, e in maniera completamente trasparente rispetto alle applicazioni in esecuzione sul MN.
Al contempo, si ricorre ad un meccanismo cross-layer denominato *Transmission Error Detector* (*TED*), che fornisce al sistema un riscontro in merito all'avvenuta o mancata trasmissione di ciascun pacchetto all'access point wireless a cui è allacciato il nodo mobile, in modo da poter effettuare una valutazione con granularità di pacchetto dell'efficienza dell'interfaccia di rete attualmente in uso.
Questo compito è affidato ad un componente chiamato *UDP Load Balancer* (*ULB*), che sfrutta le indicazioni fornite dal monitor e da TED per valutare le prestazioni prestazioni delle NIC, e si occupa di selezionarne un'altra in caso la qualità delle comunicazioni degradi al di sotto della soglia minima tollerata. Tale meccanismo è infatti in grado di ritrasmettere i pacchetti persi servendosi di una NIC differente dalla precedente, senza comportarne la duplicazione a livello applicazione.
Una volta eletta la nuova interfaccia da adoperare, questa rimpiazzerà la precedente tramite un'operazione di handover.

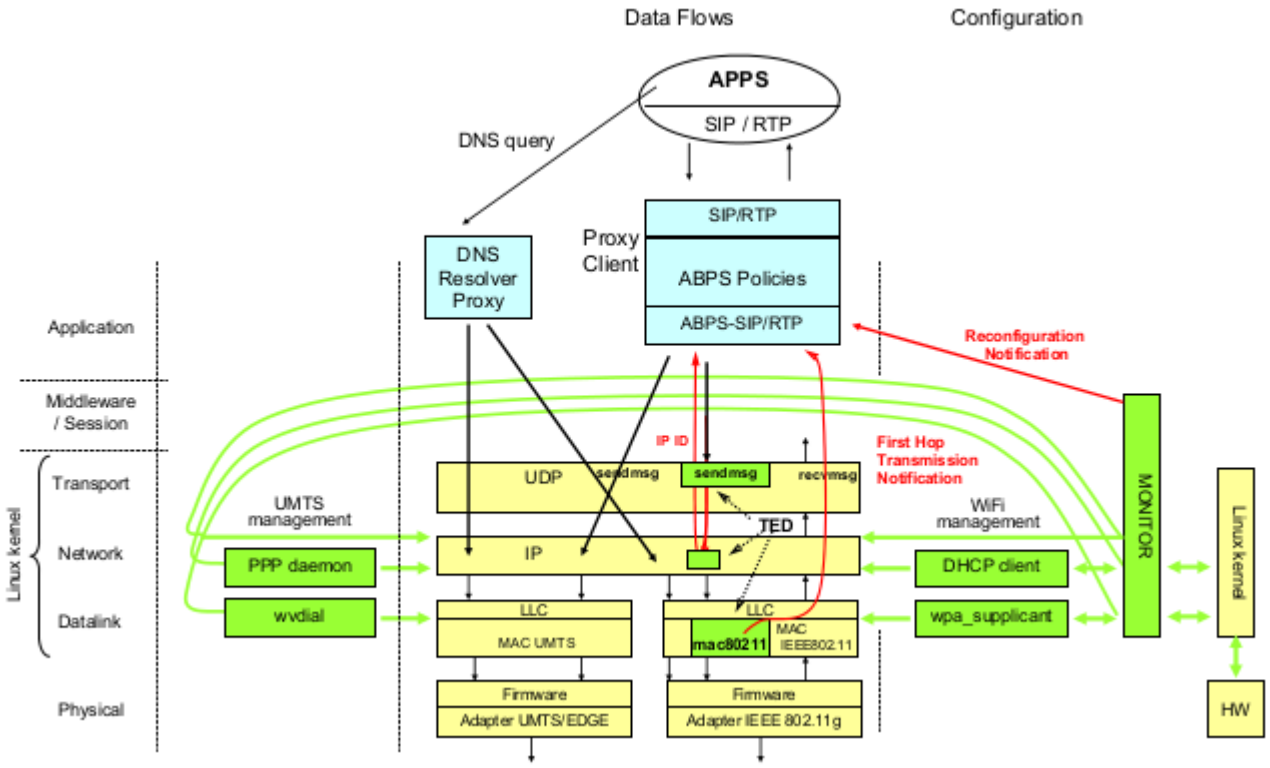

Fig. 3 – Struttura delle componenti di ABPS impiegate sul MN

Una semplice metrica impiegata per eleggere quale NIC utilizzare prevede di dare la priorità alle interfacce Wi-Fi, in quanto aventi un consumo energetico minore rispetto a quelle che fanno uso di tecnologie a lungo raggio, per poi controllare che il pacchetto dati abbia effettivamente raggiunto l'access point, servendosi della notifica inviata da TED; in caso di errori di trasmissione, viene effettuato un nuovo tentativo ricorrendo ad un'interfaccia differente (ad esempio una che faccia uso di UMTS, lo standard di comunicazione cellulare Universal Mobile Telecommunications System). Tale meccanismo software è comunque liberamente estensibile, permettendo l'utilizzo di diverse metriche nell'elezione dell'interfaccia, comprese quelle più classiche nella misurazione della QoS della rete, come larghezza di banda e qualità del segnale (ad esempio il Received Signal Strength Indicator, RSSI), così come soluzioni di altro genere.
Il proxy client e il proxy server collaborano dunque sfruttando il load balancing per massimizzare il throughput, e per minimizzare il numero di pacchetti persi e gli la durata degli intervalli di indisponibilità dovuti agli handover, oltre che per ridurre i costi economici in termini di consumo energetico.

4.3 Terminologia e concetti chiave di ABPS

- ***Proxy server***

  Elemento software in esecuzione su un nodo fisso contattabile pubblicamente e non coperto da firewall, agisce da relay per le comunicazioni tra MN e CN in modo che queste possano aggirare le limitazioni di rete imposte da NAT e fw, oltre che da punto fisso per permettere al MN di essere sempre raggiungibile anche in seguito a spostamenti verso altre reti.

- ***Proxy client***

  Elemento software in esecuzione sul nodo mobile, mantiene aperta la comunicazione con il proxy server attraverso un tunnel multi-path che sfrutta tutte le interfacce di rete disponibili sul MN; questo è gestito dai seguenti sotto-componenti software:

  - *Transmission Error Detector (TED)*
  - *UDP Load Balancer (ULB)*
  - *Monitor*

- ***Transmission Error Detector (TED)***

  Informa il Load Balancer se ciascun pacchetto inviato ha raggiunto l'access point a cui è allacciato il MN o meno.

- ***UDP Load Balancer (ULB)***

  Decide come comportarsi riguardo alle ritrasmissioni di pacchetti persi in seguito alle notifiche ricevute da TED, compresa quale interfaccia di rete utilizzare.

- ***Monitor***

  Configura, controlla e valuta il funzionamento di tutte le interfacce di rete presenti sul MN, comunicando al Load Balancer quali di queste dovessero diventare inattive.

5. Simulatori

5.1 Lavoro effettuato con OMNeT++ e INET

Per misurare l'efficienza delle tre architetture di supporto alla mobilità, e metterne in evidenza il comportamento in situazioni realistiche, sono state configurate alcune simulazioni che prevedono lo spostamento di un nodo mobile lungo un percorso preimpostato, attraversando diverse reti Wi-Fi all'interno di un ambiente cittadino, in cui i muri degli edifici sono rappresentati da ostacoli per le onde elettromagnetiche.
È stato dunque necessario importare all'interno dei simulatori di LISP e MIPv6 i moduli OMNeT++ relativi a tali ostacoli, già presenti nel simulatore ABPS, in modo da poter successivamente riprodurre il medesimo scenario urbano.
Ciò è stato effettuato a seguito di una analisi iniziale dei simulatori, dalla quale è emerso che una semplice unione di tutti i loro componenti all'interno di un unico workspace non sarebbe stata possibile, a causa di molteplici problemi di compatibilità tra le diverse versioni di INET dalle quali essi originavano; non si esclude ovviamente la fattibilità di una soluzione che preveda l'integrazione e la gestione della compatibilità di tutti e tre i simulatori in uno unico, ma si è ritenuto che attuarla avrebbe richiesto una mole di lavoro troppo onerosa per concludersi entro i tempi previsti.
Si è dunque optato per concentrarsi sull'integrazione delle componenti relative agli ostacoli incontrati dai segnali radio all'interno dei due simulatori di LISP e MIPv6, apportando le opportune modifiche agli altri moduli già presenti in maniera da garantirne la compatibilità.
Per tutti e tre i simulatori sono state inoltre richieste sostanziali modifiche al codice sorgente per correggere numerose problematiche e comportamenti anomali che impedivano un corretto funzionamento o andavano a intaccare i risultati sperimentali ottenuti.

5.2. Modifiche comuni

• Modello di attenuazione del segnale

I segnali radio utilizzati durante le simulazioni nelle comunicazioni wireless subiscono due tipi di attenuazione:

- dovuta alla distanza percorsa (*path loss*);
- dovuta agli ostacoli incontrati (*penetration loss*).

La prima viene calcolata tramite il modello impiegato di default da INET, e non ha subito modifiche; l'attenuazione dovuta agli ostacoli, in particolare quelli utilizzati per rappresentare i muri presenti nello scenario urbano, era inizialmente gestita invece da una formula importata dal simulatore ABPS. Questa, tuttavia, presentava alcune anomalie nei valori ricavati, e in particolare sembrava comportare un aumento della potenza del segnale piuttosto che una riduzione in caso quest'ultimo attraversasse più di un ostacolo.
La formula ha quindi subito una semplificazione, in maniera tale che ogni ostacolo incontrato (dopo aver confermato che si trovi effettivamente sul percorso tra i due endpoint) dia il proprio contributo all'attenuazione totale subita, che viene applicata dopo una conversione da dBm a mW:

```
foreach(ostacolo incontrato nel percorso del segnale radio)
      totalPenetrationLoss += obstacle.getPenetrationLoss();
totalPenetrationLoss = dBm_to_mW(totalPenetrationLoss);
rcvdPower *= totalPenetrationLoss;
```

• Calcolo della latenza

La latenza nella ricezione dei pacchetti durante le comunicazioni tra mobile node e correspondent node viene misurata semplicemente come la differenza tra il tempo di simulazione a cui il pacchetto arriva al destinatario e il tempo di simulazione al quale questo è stato generato. Questa viene poi registrata all'interno di un vettore di risultati, che può essere analizzato e mostrato su grafico tramite gli strumenti messi a disposizione da OMNeT++ stesso, oppure esportato su programmi esterni.

```
latencyInVector.recordWithTimestamp(simTime(),
                                    simTime() - msg->getCreationTime());
```

• Calcolo degli intervalli di indisponibilità

Vengono inoltre misurati gli intervalli di indisponibilità nella comunicazione, dovuti agli spostamenti e ai conseguenti cambi di rete effettuati dal mobile node, secondo i due criteri seguenti:

- sul mobile node, per misurarne l'indisponibilità in fase di ricezione;
- sul correspondent node, per misurare l'indisponibilità del MN in fase di invio.

In entrambi i casi, il principio che si è scelto di utilizzare per individuare un handover è quello di misurare l'intervallo che trascorre fra i due tempi della simulazione ai quali stati sono stati inviati, o rispettivamente ricevuti, l'ultimo pacchetto con un certo indirizzo e il primo pacchetto con un nuovo indirizzo.

Il correspondent node utilizzato nelle simulazioni è un nodo fisso, e dunque ovviamente non soggetto ad operazioni di handover che ne precludano la raggiungibilità durante le comunicazioni; nonostante questo, la seconda misurazione è stata comunque effettuata sul CN, in quanto è semplice constatare come esista in realtà una dualità speculare tra le misure effettuate sul mobile node e quelle effettuate sul correspondent node; in particolare:

- quando il MN non è in grado di inviare dati a causa degli handover, ovviamente in CN non riesce a ricevere nulla, ed il CN subisce dunque l'effetto dell'indisponibilità di invio del MN, riuscendo anch'esso a misurarne la durata;
- allo stesso modo, mentre il MN non è in grado di ricevere dati, i pacchetti inviati dal CN ovviamente non lo raggiungeranno con successo; si sarebbe dunque potuto pensare di effettuare sul CN anche il calcolo di questa indisponibilità di ricezione del MN, ad esempio utilizzando un messaggio di ack o comunque un riscontro per l'avvenuta ricezione sul MN.

L'indisponibilità del MN in fase di ricezione consiste infatti nel periodo che trascorre tra due eventi consecutivi di ricezione di dati provenienti dal CN, nel mezzo dei quali il MN ha però cambiato il proprio indirizzo sorgente nelle comunicazioni col CN tramite un'operazione di handover.
Se, all'evento di ricezione del pacchetto con indice *n*, l'interfaccia del MN ha un indirizzo differente rispetto a quando è avvenuto l'evento di ricezione con indice *n - 1*, è segno che nel frattempo è avvenuto un handover; da questo ne è conseguito un periodo di riconfigurazione durante il quale il MN non è stato raggiungibile, di durata approssimabile appunto all'intervallo tra gli eventi di ricezione di indice *n - 1* e *n*.

```
if(!lastSenderAddress.equals(receivedPacket->getSrcAddr())) {
      senderHandoverDowntimes.record(simTime().dbl() - lastReceiveTime);
      lastSenderAddress.set(receivedPacket->getSrcAddr());
}
lastReceiveTime = simTime().dbl();
```

Analogamente, lo stesso ragionamento può essere applicato sul correspondent node, seppur in maniera speculare, tra due eventi di ricezione consecutivi *m - 1* e *m*: nel caso questi siano causati da due pacchetti inviati dal MN, ma abbiano un indirizzo sorgente differente, si può inferire che tra l'uno e l'altro sia stato effettuato un handover, e i tempi di simulazione ai quali sono avvenuti possono nuovamente essere utilizzati nel calcolo di un'approssimazione dell'intervallo di indisponibilità del mobile node in fase di invio.

```
if(!lastDestinationAddress.equals(receivedPacket->getDestAddr())) {
      receiverHandoverDowntimes.record(simTime().dbl() - lastReceiveTime);
      lastDestinationAddress.set(receivedPacket->getDestAddr());
}
lastReceiveTime = simTime().dbl();
```

5.3 Simulatore MIPv6

• Porting degli ostacoli

Inizialmente è stato effettuato un primo porting dei moduli relativi agli ostacoli utilizzando una precedente versione di INET, la 1.99.1, in quanto questa offriva già il supporto per Mobile IPv6, ed era più simile a quella da cui è stato sviluppato il simulatore per LISP, ovvero INET-20100323. Ciò ha semplificato notevolmente il procedimento di modifica, in quanto nelle release successive sarebbe invece stato richiesto adattarsi ai significativi cambiamenti che sono stati effettuati alla struttura generale dei moduli che implementano lo stack ISO/OSI all'interno di INET, in particolare per quanto riguarda la rappresentazione a livello fisico delle reti wireless.

Questa prima iterazione del porting degli ostacoli ha previsto diverse modifiche, delle quali le più importanti sono state le seguenti:

- *BasicMobility*, il modulo che controlla il movimento del nodo mobile nell'ambiente della simulazione, è diventato *MobilityBase*, che non è più un'interfaccia;
- allo stesso modo, il modulo *NullMobility*, che configura il dispositivo che lo utilizza come nodo stazionario, è diventato *StationaryMobility*;
- è stato necessario importare alcuni parametri dalla classe *BasicMobility* a *MobilityBase*, e di conseguenza modificarne la funzione di inizializzazione, in quanto non più presenti nella nuova classe, ma ancora utilizzati nelle funzioni di gestione della mobilità;
- altri parametri sono stati rinominati all'interno della nuova classe, ed è quindi stato richiesto modificare il codice per fare uso dei parametri analoghi (in particolare, *MobilityBase.lastPosition* ha sostituito *BasicMobility.pos*);
- varie modifiche aggiuntive alla classe *ChannelControl*, che si occupa della gestione delle comunicazioni wireless a livello fisico, tra cui l'inserimento di parametri e funzioni mancanti.

Dopo aver effettuato questo primo porting, tuttavia, si è ritenuto opportuno tentare di fare lo stesso anche su una versione più recente di INET, in quanto utilizzare un simulatore più aggiornato, nonostante i problemi di incompatibilità che questo avrebbe inizialmente comportato, avrebbe ragionevolmente aiutato ad ottenere risultati più verosimili.

L'operazione di porting è quindi stata ripetuta su quella che era la release più recente di OMNeT++, la 4.6, che utilizza INET 3.0.
In generale, i cambiamenti effettuati sono di natura analoga a quelli del porting precedentemente descritto, consistendo principalmente nell'importazione di parametri e funzioni mancanti, e nell'aggiornamento del codice in caso questo facesse riferimento a elementi che nella nuova versione sono stati rinominati o spostati.
Alcuni tra questi includono:

- i file *FWMath.h* (rinominato in *INETMath.h*) e *Coord.h*;
- la classe *BasicMobility*, rinominata in *IMobility*;
- la classe *NullMobility*, rinominata in *StationaryMobility*;
- *ObstacleMobility*, il modello di gestione della mobilità utilizzato dagli ostacoli per calcolare l'ostruzione dei segnali wireless;
- *SimpleObstacleLoss*, il modello di perdita del segnale dovuta agli ostacoli, che è stato implementato seguendo la struttura definita all'interno del simulatore, a partire dall'interfaccia *IObstacleLoss* messa a disposizione nelle versioni più recenti di INET.

• Modifiche apportate e correzione problemi

- A causa del modo in cui il simulatore gestisce il decapsulamento dei pacchetti, venivano a presentarsi degli errori durante la fase di return routability, in cui il CN tentava di utilizzare l'interfaccia sbagliata per inviare i messaggi di Home Test e Care-of Test.
  Più nel dettaglio, quando il CN riceve un pacchetto incapsulato, questi ne rimuove l'header esterno, e lo invia nuovamente a sé stesso tramite l'interfaccia di loopback; il problema nasceva dal fatto che era l'interfaccia da cui si ricevono HoTI e CoTI nella return routability ad essere utilizzata per rispondere con HoT e CoT: in caso i primi arrivassero dall'interfaccia di loopback, era proprio questa ad essere, erroneamente, selezionata per inviare i messaggi di risposta.
  Per risolvere questo problema, è stato aggiunto un parametro ai messaggi di Home Test Init e Care-of Test Init, settato dal MN, che indichi a quale destinatario fossero inizialmente indirizzati i messaggi originali, e che il CN possa consultare per individuare l'interfaccia corretta attraverso cui rispondere.
- L'incapsulamento dei pacchetti causava problemi anche in caso questi andassero a generare

messaggi ICMP durante la procedura di Neighbour Discovery. Questi contengono nel proprio corpo anche il pacchetto originale, che viene esaminato dall'host che lo riceve per estrarne il protocol number e determinare di conseguenza come gestirlo. Una volta ricevuti, questi non venivano tuttavia inoltrati correttamente, avendo un protocol number pari a 41, il valore riservato alla IPv6 Encapsulation, che non era riconosciuto come valido.
Si è dunque optato per inserire un controllo aggiuntivo, per il quale viene rimosso l'header esterno di incapsulamento IPv6 prima di generare il relativo messaggio ICMP, seguendo le direttive indicate in [7].

- Analogamente, i messaggi ICMP generati a partire da pacchetti con protocol number 135, il valore riservato al Mobility Header di MIPv6, causavano un errore durante l'esecuzione, in quanto questo non era riconosciuto come input valido per il mapping che determina a quale protocollo di trasporto inoltrare il messaggio.
  Si è dunque ricorso ad un controllo aggiuntivo che, per questioni di semplicità, emula banalmente il comportamento dell'applicazione alla ricezione di un errore ICMP, scartando banalmente il pacchetto.
- È stata effettuata una modifica ai due parametri utilizzati per la gestione del controllo di validità e di prossimità alla scadenza delle entry nelle Binding Update List; i time-to-live utilizzati in precedenza erano dei valori costanti, mentre nei controlli che li coinvolgono questi erano confrontati con il tempo di simulazione, che è invece un valore incrementale. Sono stati quindi convertiti tali parametri per renderli conformi al tempo di simulazione e correggere il confronto effettuato.
- Durante la procedura di Neighbour Discovery, era un presente un controllo per limitare l'invio di messaggi di Router Advertisement solamente verso un access point; questo è stato modificato per includere anche la possibilità di inoltrarli verso un hub o un bus ethernet, per dare la possibilità di includere anche tali entità di rete all'interno delle simulazioni.
- È stato disabilitato un parametro nel modulo WirelessHost6, che causava un qualunque nodo wireless di essere considerato come MN ai fini delle procedure di gestione della mobilità.
- Sono stati diminuiti i timer di attesa minima consentita tra l'invio di due diversi messaggi di HoTI e CoTI, in modo da permettere ad un nodo mobile di spostarsi con maggiore frequenza da una rete ad un'altra, ed effettuare la procedura di return routability senza che il CN sia ancora in attesa dei timeout precedenti.

- Sono state inoltre apportate altre varie modifiche di natura implementativa per la correzione di alcuni errori in fase di esecuzione, ad esempio controlli aggiuntivi su parametri a cui non sia stato specificato un valore.

• Scenario di prova – testMIPv6

Per testare l'implementazione del protocollo Mobile IPv6 e le relative modifiche effettuate, è stato realizzato un semplice scenario di prova, in cui il MN si sposta seguendo un percorso circolare in senso orario, e attraversando quattro reti wireless, dove ciascuna di queste è parzialmente sovrapposta alla precedente ed alla successiva in maniera tale da garantire sempre la presenza di copertura per il nodo mobile. Le sottoreti dello scenario, così come per tutte quelle negli scenari che seguiranno, sono interconnesse tramite link ad alta latenza (30ms) ad un router centrale, che rappresenta la rete globale.

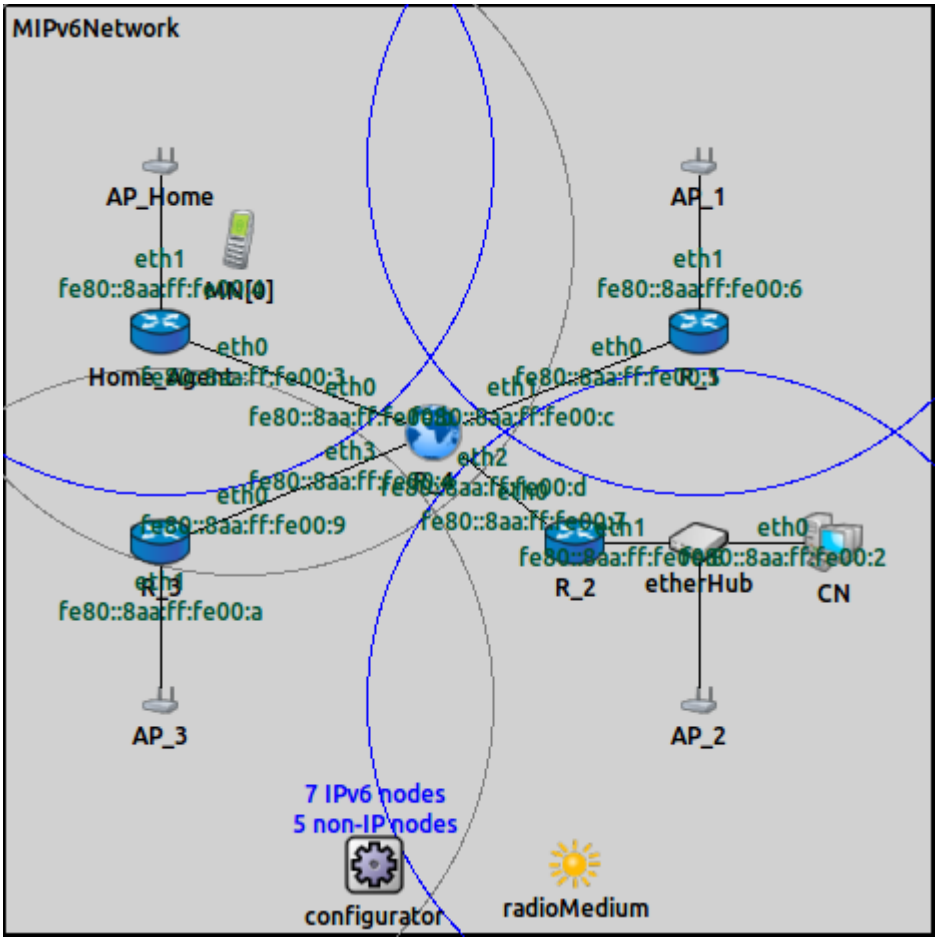

Fig. 4 – Struttura dello scenario di prova testMIPv6

I (principali) parametri utilizzati nella configurazione dei moduli utilizzati nei test effettuati con questo scenario sono stati i seguenti:

```
# = ================================================================
network = MIPv6Network
sim-time-limit = 1000s
*.total_mn = 1
*.total_cn = 1
**.neighbourDiscovery.minIntervalBetweenRAs = 0.03s #MinRtrAdvInterval (RFC 3775)
**.neighbourDiscovery.maxIntervalBetweenRAs = 0.07s #3 ;MaxRtrAdvInterval (RFC 3775)

**.wlan*.radio.receiver.sensitivity = -85dBm
**.carrierFrequency = 2.4GHz

# channel physical parameters
**.radioMedium.pathLossType = "SimplePathLoss"
*.radioMedium.mediumLimitCache.maxTransmissionPower = 2.5mW
*.radioMedium.mediumLimitCache.minReceptionPower = -85dBm
*.radioMedium.mediumLimitCache.minInterferencePower = -85dBm

# Access Point AP_Home ; AP_1 ; AP_2 ; AP_3 Parameters for EtherMAC
**.AP_Home.wlan*.mgmt.ssid = "HOME"
**.AP_Home.wlan*.mac.address = "10:AA:00:00:00:01
**.AP_1.wlan*.mgmt.ssid = "AP1"
**.AP_1.wlan*.mac.address = "10:AA:00:00:A1:01"
**.AP_2.wlan*.mgmt.ssid = "AP2"
**.AP_2.wlan*.mac.address = "10:AA:00:00:A2:01"
**.AP_3.wlan*.mgmt.ssid = "AP3"
**.AP_3.wlan*.mac.address = "10:AA:00:00:A3:01"

# mobility
**.MN[0].mobilityType = "CircleMobility"
**.MN[0].mobility.speed = 2mps
**.MN[0].mobility.r = 175m
**.MN[0].mobility.cx = 275m
**.MN[0].mobility.cy = 275m
**.MN[0].mobility.startAngle = -135deg

# wireless channels
**.AP_Home.wlan*.radio.channelNumber = 1
**.AP_1.wlan*.radio.channelNumber = 2
**.AP_2.wlan*.radio.channelNumber = 3
**.AP_3.wlan*.radio.channelNumber = 4
**.MN*.wlan*.radio.channelNumber = 0  # just initially -- it'll scan

# udp app
**.udpApp.typename = "UDPBasicApp"

**.MN[*].numUdpApps = 1
**.MN[*].udpApp[*].typename = "UDPMIPApp"
**.MN[*].udpApp[0].destinationNode = "CN"
**.MN[0].udpApp[0].destPort = 230
**.MN[0].udpApp[0].localPort = 230
**.MN[0].udpApp[0].startTime = 10s
**.MN[0].udpApp[0].messageLength = 512 byte

**.CN.numUdpApps = 1
**.CN.udpApp[*].typename = "UDPMIPApp"
**.CN.udpApp[0].destinationNode = "MN[0]"
**.CN.udpApp[0].destPort = 230
**.CN.udpApp[0].localPort = 230
**.CN.udpApp[0].startTime = 10s
**.CN.udpApp[0].messageLength = 512 byte
# = ================================================================
```

Mantenendo tale configurazione, sono poi state effettuate due differenti simulazioni, variando la frequenza di invio dei pacchetti da parte delle applicazioni in esecuzione su mobile node e correspondent node, che generano una trasmissione di dati comparabile a quella di un flusso multimediale, ad esempio una chiamata tramite voice-over-Wi-Fi.

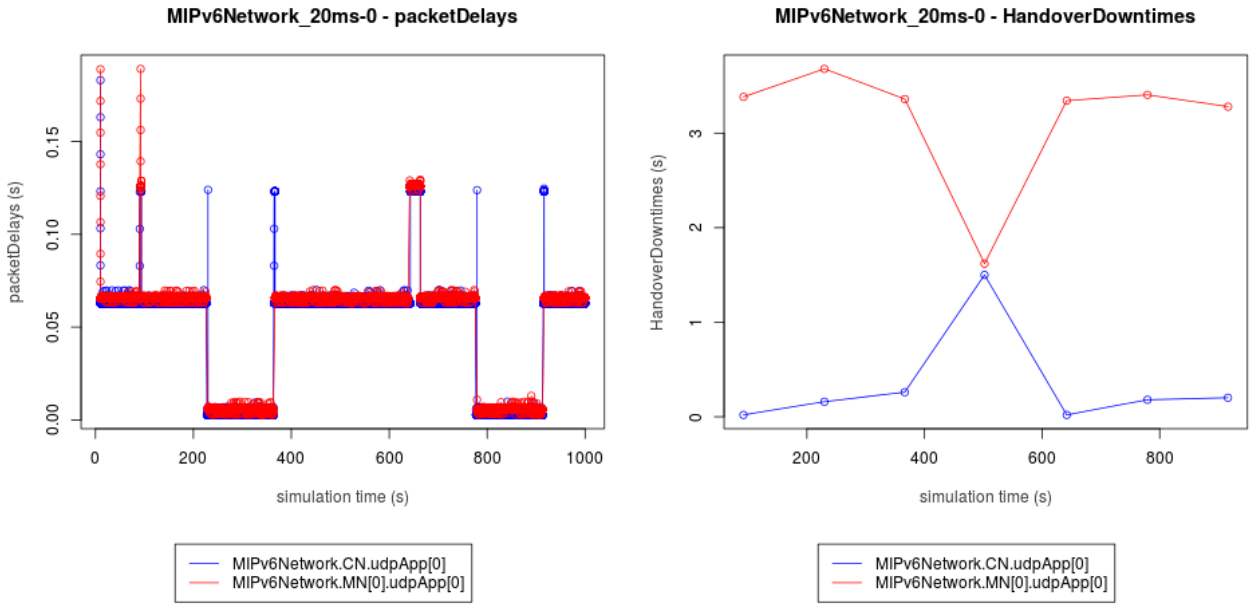

Fig. 5 – Latenza e intervalli di indisponibilità per la rete testMIPv6 (sendInterval = 20ms)

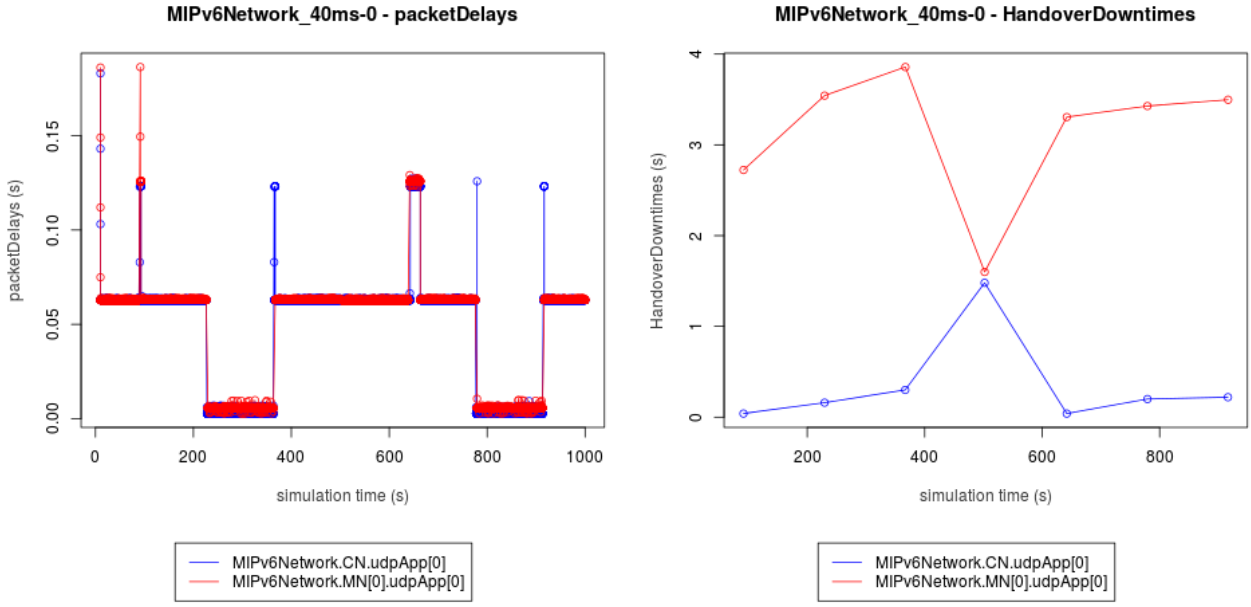

Fig. 6 – Latenza e intervalli di indisponibilità per la rete testMIPv6 (sendInterval = 40ms)

I relativi risultati in termini di latenza nella ricezione dei pacchetti e di intervalli di indisponibilità del mobile node dovuti agli handover sono riportati in *Fig. 5*, per un rate di invio di un pacchetto ogni 20ms, e in *Fig. 6*, per un rate di 40ms.
La linea blu mostra i valori calcolati sul correspondent node, relativi dunque alla latenza dei pacchetti inviatigli dal mobile node e agli intervalli in cui questo non è stato in grado di trasmettere dati a causa del passaggio da una rete alla successiva;
analogamente, la linea rossa si riferisce ai valori calcolati sul mobile node, ed è quindi relativa alla latenza dei pacchetti arrivati dal correspondent node, nel grafico di sinistra, e ai suoi intervalli di indisponibilità in fase di ricezione, per quello di destra.

I picchi all'interno nel grafico delle latenze evidenziano i passaggi da una rete alla successiva, ed appena dopo i 200 secondi di simulazione si può notare come la connessione del nodo mobile alla stessa rete del CN comporti ovviamente un abbattimento nei valori della latenza, che risale nuovamente appena prima dei 400 secondi, all'istante che corrisponde al successivo cambio di rete; attorno ai 500 secondi di esecuzione della simulazione il MN raggiunge nuovamente il punto iniziale del suo percorso, che si andrà poi a ripetere nei 500 secondi successivi, seguendo comportamenti simili a quelli esibiti durante il primo giro compiuto attorno allo scenario.
Il picco di latenza presente appena oltre i 600 secondi di simulazione è attribuibile ad un tentativo fallito di stabilire la route optimization tra MN e CN, che è stata instaurata soltanto dopo un breve periodo di trasmissione dei pacchetti avvenuta in maniera convenzionale, ovvero transitando attraverso il proxy server.
Il picco nei valori degli intervalli di indisponibilità del MN, rispettivamente minimo per quelli in ricezione e massimo per quelli in invio, coincide con il passaggio dalla quarta rete visitata alla rete domestica; questo è attribuibile alla procedura di returning home, che il nodo mobile attua per informare il suo home agent di non necessitare più del suo servizio di inoltro dei pacchetti inviati al proprio home address.
Si può infine notare come i comportamenti della rete rimangano costanti anche una volta raddoppiato l'intervallo che trascorre tra l'invio di due pacchetti consecutivi: la latenza della rete non viene intaccata dalla frequenza di invio dei pacchetti, che è invece presa in considerazione nella misurazione degli intervalli di indisponibilità, i quali sono tuttavia già abbastanza ampi da non avere una variazione sostanzialmente apprezzabile.

5.4 Simulatore LISP

Il lavoro svolto sul simulatore LISP è stato analogo a quello richiesto dal simulatore MIPv6: scoperta e analisi dei comportamenti anomali, correzione dei problemi che li causavano, porting e integrazione dei moduli relativi agli ostacoli già presenti nel simulatore ABPS.

• Scenario di prova - testLISP

Come per MIPv6, anche all'interno del simulatore LISP è stato realizzato un semplice scenario di prova, con cui testare il protocollo in condizioni ottimali (senza impedimenti dovuti ad ostacoli né al traffico di altri host).

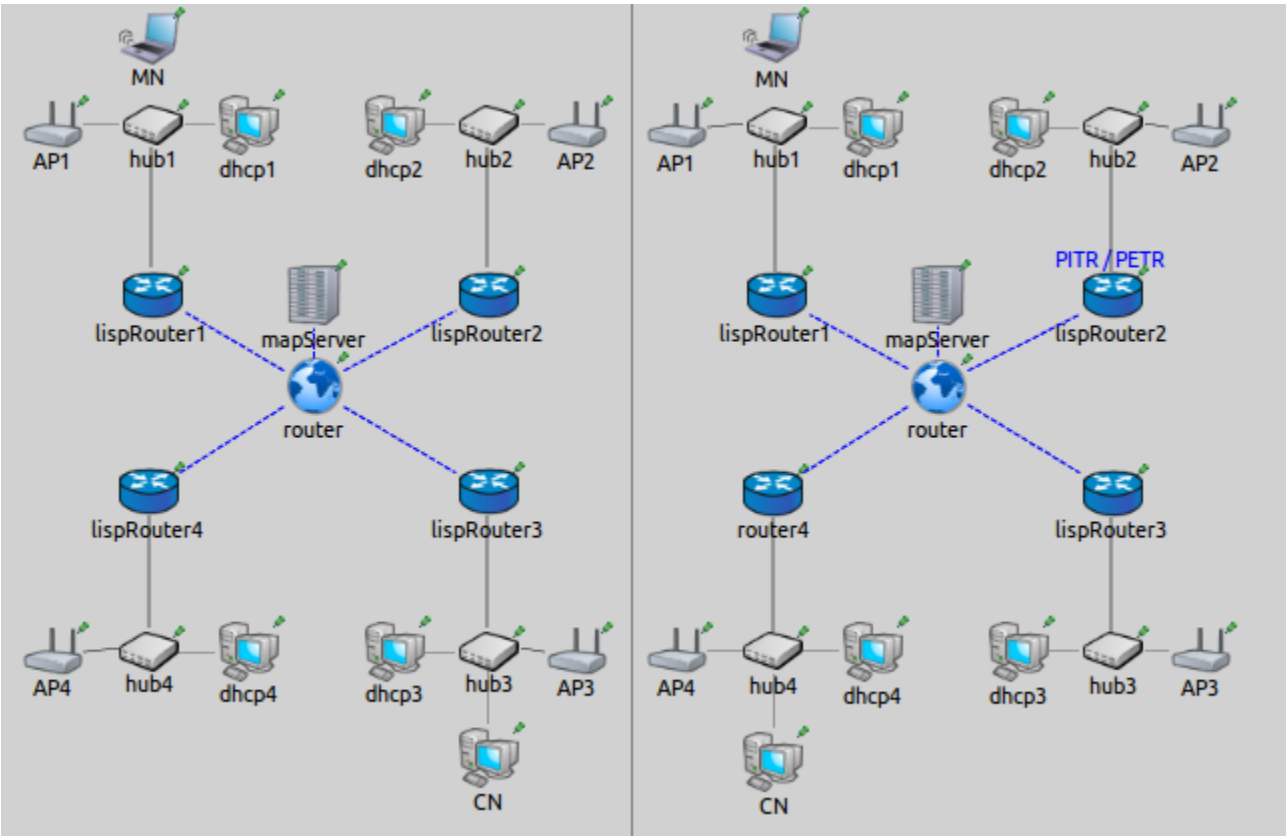

Fig. 7 – Struttura dello scenario di prova testLISP

La Fig. 7 riporta le due reti realizzate tramite questo scenario:

- nella configurazione mostrata a sinistra, il CN è situato in un sito LISP, dietro a *lispRouter4*;
- in quella di destra, il CN si trova invece in un sito non-LISP, dietro a *router4*, che è appunto un normale router senza alcuna funzionalità LISP; per instaurare la comunicazione sarà

dunque necessario servirsi di *lispRouter2*, che farà da proxy-ITR per i pacchetti inviati dal CN, e da proxy-ETR per quelli inviati dal MN.

Per un'analisi più dettagliata del simulatore LISP, così come dei risultati sperimentali ottenuti dagli scenari realizzati servendosi di esso, si rimanda alla tesi del collega E. Convertino, con cui si è collaborato nell'affrontare le fasi iniziali del lavoro di tesi, e che si è poi maggiormente concentrato sulle tematiche relative a Locator/Identifier Separation Protocol.

5.5 Simulatore ABPS

• Modifiche apportate e correzione problemi

- È stata apportata una modifica al modulo che gestisce il *Dynamic Host Configuration Protocol* (*DHCP*), per renderlo conforme alle specifiche indicate in [8].
  Questo infatti faceva precedentemente uso di un indirizzo preimpostato a cui rivolgersi durante la procedura di DHCP discovery; la ricerca di un server DHCP viene ora invece effettuata tramite una richiesta inviata in broadcast (con destinazione 255.255.255.255), come richiesto dal protocollo.
- Per rendere possibile la modifica appena descritta, è stato inoltre necessario abilitare l'invio dei pacchetti in broadcast, in quanto precedentemente l'access point scartava qualunque pacchetto destinato ad un indirizzo non presente nella lista delle interfacce ad esso associate; è stato dunque aggiunto un controllo che permettesse comunque l'inoltro anche per la destinazione di broadcast.
- Come ulteriore conseguenza dell'implementazione dei messaggi di broadcast, è stato necessario apportare delle modifiche anche ai moduli relativi ai router, in maniera che potessero gestire tali messaggi senza generare errori a tempo di esecuzione; per fare ciò è stato dunque creato un nuovo modulo, denominato *BroadcastRouter*, che si serve di due applicazioni dummy, *tcpSinkApp* e *udpSink*, per scartare semplicemente i pacchetti di DHCP discovery che questo riceve in caso sia connesso ad un access point wireless.
- Si sono presentati problemi con il protocollo *ARP* (*Address Resolution Protocol*) quando utilizzato dal MN in congiunzione al multihoming: un MN avente più interfacce di rete non faceva distinzioni su quale di queste avesse ricevuto una ARP request, esaminando semplicemente la lista degli indirizzi IP delle sue NIC, e rispondendo positivamente in caso una qualunque di queste avesse l'indirizzo richiesto.
  Ciò ovviamente comportava errori di comunicazione in caso la richiesta arrivasse su una certa interfaccia di indirizzo B, ma l'indirizzo richiesto fosse invece quello di una certa altra interfaccia A del MN. Si è optato dunque per aggiungere un controllo riguardo quale sia la NIC sulla quale si ricevono le ARP request, e assicurarsi di rispondere soltanto in caso sia effettivamente quella che stava cercando il mittente.
- Per rendere operativa quest'ultima modifica, è stato inoltre necessario differenziare le entry all'interno delle cache ARP dei nodi in base all'interfaccia a cui queste effettivamente si

riferiscono, onde evitare di aggiornare le informazioni memorizzate nella cache quando in realtà il pacchetto ARP è stato ricevuto da un'interfaccia differente da quella desiderata.

- I risultati ottenuti dalle simulazioni riguardo i tempi di latenza nelle comunicazioni tra MN e CN si erano inizialmente mostrati come inaspettatamente instabili; la causa è stata poi ricondotta all'inserimento all'interno del codice dei moduli che gestiscono il livello MAC di alcune procedure che dettavano degli arbitrari drop e burst drop dei pacchetti.
  Per semplicità, queste operazioni sono state semplicemente disabilitate, ristabilendo la regolarità che ci si aspettava nella misurazioni della latenza; si potrebbe eventualmente pensare di spostare tali procedure all'interno delle effettive configurazioni della simulazione, e di gestirle tramite parametri.
- L'applicazione originale è stata suddivisa in tre differenti moduli:
  - *ULB*, che implementa le funzionalità del load balancer;
  - *ABPSProxyServer*, in esecuzione sul nodo fisso utilizzato come proxy client nelle comunicazioni tra mobile node e correspondent node; decapsula i pacchetti in transito sul proxy server e li inoltra all'effettivo destinatario;
  - *ABPSUDPApp*, che estende ULB e invia pacchetti al destinatario specificato per simulare un flusso di dati tra i due host; per questioni di semplificazione, questa gestisce inoltre le procedure di registrazione al server proxy e di setup della comunicazione tra MN e CN, che sarebbero normalmente riservate ai protocolli SIP e RTP.
- Alla classe *VoWPacket*, che simula un pacchetto voice over Wi-Fi inviato dalle *ABPSUDPApp*, sono stati aggiunti due parametri per memorizzare id e porta degli effettivi destinatari della comunicazione, che saranno utilizzati dal proxy server per inoltrarli correttamente.
- Per abilitare la funzione di relay del proxy server, la classe *IP* è stata modificata in maniera tale da permettergli di utilizzare l'indirizzo del mobile node come source address dei pacchetti che inoltra.
- Le applicazioni in esecuzione su mobile node e correspondent node sono state infine configurate in maniera tale da iniziare a trasmettere dati solo dopo un timeout iniziale, in cui viene concesso alla rete il tempo necessario per finire di configurarsi (allacciamento agli access point, richiesta di indirizzi IP ai server DHCP, ecc.).

• Scenario di prova – testABPS

In maniera analoga a quanto fatto per MIPv6 e LISP, anche per ABPS è stato realizzato uno scenario di prova per testare il funzionamento del simulatore; in particolare, è stato fatto uso di due configurazioni di rete, che si differenziano in base alla posizione scelta il server proxy:

- nel primo caso, questo è esterno a tutte le sottoreti a cui si allaccia periodicamente il nodo mobile, ed è separato da queste da quella che è idealmente la backbone della rete globale (il router centrale, raggiungibile tramite link ad alta latenza);
- nel secondo, il server proxy si trova invece all'interno di una delle sottoreti che sarà visitata anche dal MN durante il suo percorso, e in particolare la stessa a cui è connessa anche il CN.

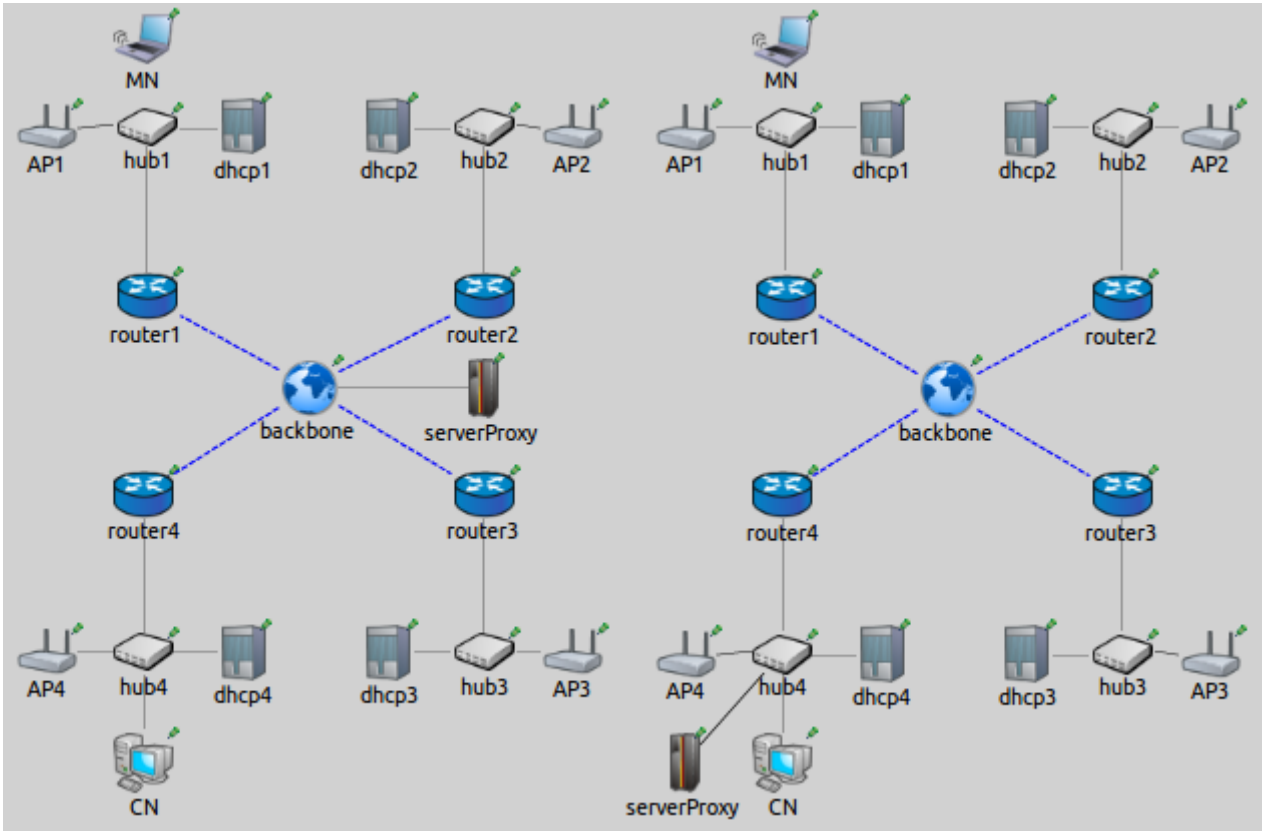

Fig. 9 – Struttura dello scenario di prova testABPS

Per entrambi i casi, i parametri principali sono stati settati come segue:

```
# ======================================================================
sim-time-limit = 1000s

# Access Points configuration
**.AP1.wlan.mac.address = "10:00:00:00:00:00"
**.AP2.wlan.mac.address = "20:00:00:00:00:00"
**.AP3.wlan.mac.address = "30:00:00:00:00:00"
**.AP4.wlan.mac.address = "40:00:00:00:00:00"
**.MN.wlan[0].mac.address = "0A:AA:00:01:00:01"
**.MN.wlan[1].mac.address = "0A:AA:00:01:00:02"
**.AP1.wlan.mgmt.ssid = "AP1"
**.AP2.wlan.mgmt.ssid = "AP2"
**.AP3.wlan.mgmt.ssid = "AP3"
**.AP4.wlan.mgmt.ssid = "AP4"

# Channel numbers configuration
**.MN.wlan[0].radio.channelNumber = 1
**.MN.wlan[1].radio.channelNumber = 2
**.AP1.wlan.radio.channelNumber = 1
**.AP2.wlan.radio.channelNumber = 2
**.AP3.wlan.radio.channelNumber = 3
**.AP4.wlan.radio.channelNumber = 4

# mobility
**.MN.mobilityType = "CircleMobility"
**.MN.mobility.speed = 2 mps
**.MN.mobility.startAngle = -135deg
**.MN.mobility.r = 175
**.MN.mobility.cx = 275
**.MN.mobility.cy = 275

**.startTime = 5s

# UDP app on mobile node
**.MN.numUdpApps = 1
**.MN.udpAppType = "ABPSUDPApp"
**.MN.udpApp[0].clientId = 2
**.MN.udpApp[0].receiverId = 3
**.MN.udpApp[0].destPort = 80
**.MN.udpApp[0].localPort = 20
**.MN.udpApp[0].clientDHCP = true
**.MN.udpApp[0].messageLength =  512 byte

# UDP app on correspondent node
**.CN.numUdpApps = 1
**.CN.udpAppType = "ABPSUDPApp"
**.CN.udpApp[0].clientId = 3
**.CN.udpApp[0].receiverId = 2
**.CN.udpApp[0].destPort = 20
**.CN.udpApp[0].localPort = 80
**.CN.udpApp[0].clientDHCP = true
**.CN.udpApp[0].messageLength =  512 byte

# Server proxy app
## App fixed
**.serverProxy.numUdpApps = 1
**.serverProxy.udpAppType = "ABPSProxyServerApp"
**.serverProxy.udpApp[0].clientId = 1

# DHCPs configuration
**.dhcp*.udpAppType = "ABPSUDPApp"
**.dhcp*.numUdpApps = 1
**.dhcp*.udpApp[0].messageLength = 0
**.dhcp*.udpApp[0].messageFreq = 0
```

```
**.dhcp1.udpApp[0].clientId = 101
**.dhcp2.udpApp[0].clientId = 102
**.dhcp3.udpApp[0].clientId = 103
**.dhcp4.udpApp[0].clientId = 104

**.dhcp1.udpApp[0].ipAddressesDHCP = "132.187.1.100 … 132.187.1.104" #ip addrs for DHCP client
**.dhcp2.udpApp[0].ipAddressesDHCP = "132.187.2.100 … 132.187.2.104" #ip addrs for DHCP client
**.dhcp3.udpApp[0].ipAddressesDHCP = "132.187.3.100 … 132.187.3.104" #ip addrs for DHCP client
**.dhcp4.udpApp[0].ipAddressesDHCP = "132.187.4.100 … 132.187.4.104" #ip addrs for DHCP client

**.radio.transmitterPower = 2.5mW
**.radio.sensitivity = -90dBm
**.radio.pathLossAlpha = 2
# ===================================================================
```

Per entrambe le collocazioni del proxy server sono poi state eseguite due simulazioni, variando nuovamente la frequenza di invio dei pacchetti da parte di MN e CN da 20ms a 40ms; i risultati ottenuti sono riportati nelle figure seguenti.

Le misurazioni effettuate, così come la loro rappresentazione, seguono gli stessi criteri utilizzati con la rete testMIPv6 (*vedi capitolo 5.3*).

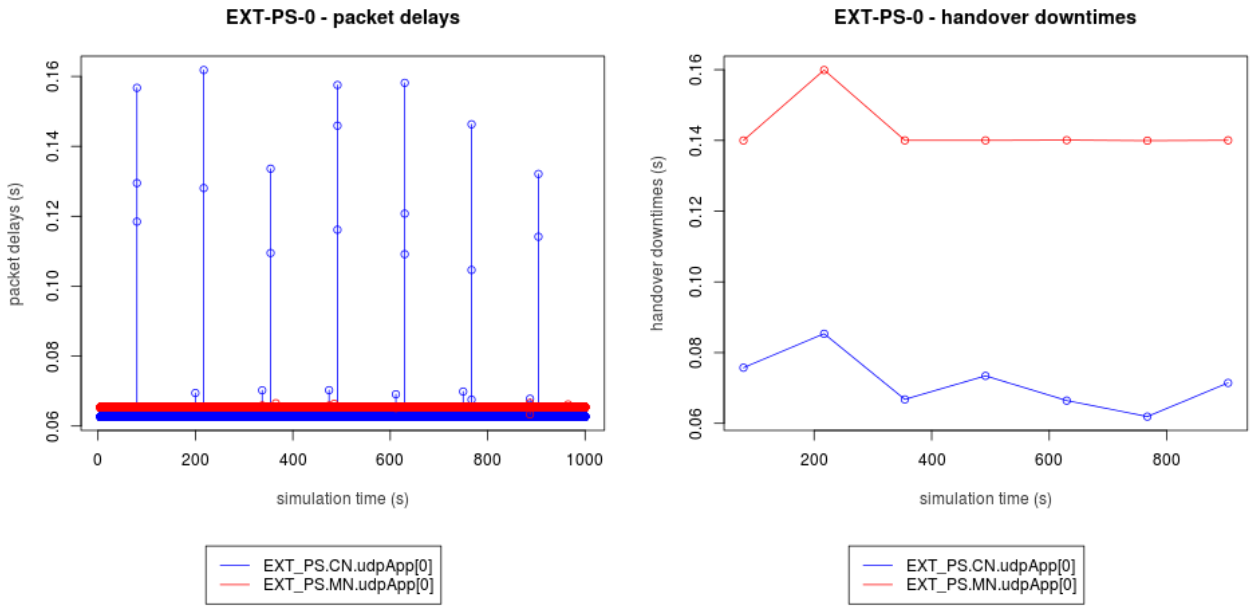

Fig. 10 – Latenza e intervalli di indisponibilità per la rete testABPS – proxy server esterno alle sottoreti (sendInterval = 20ms)

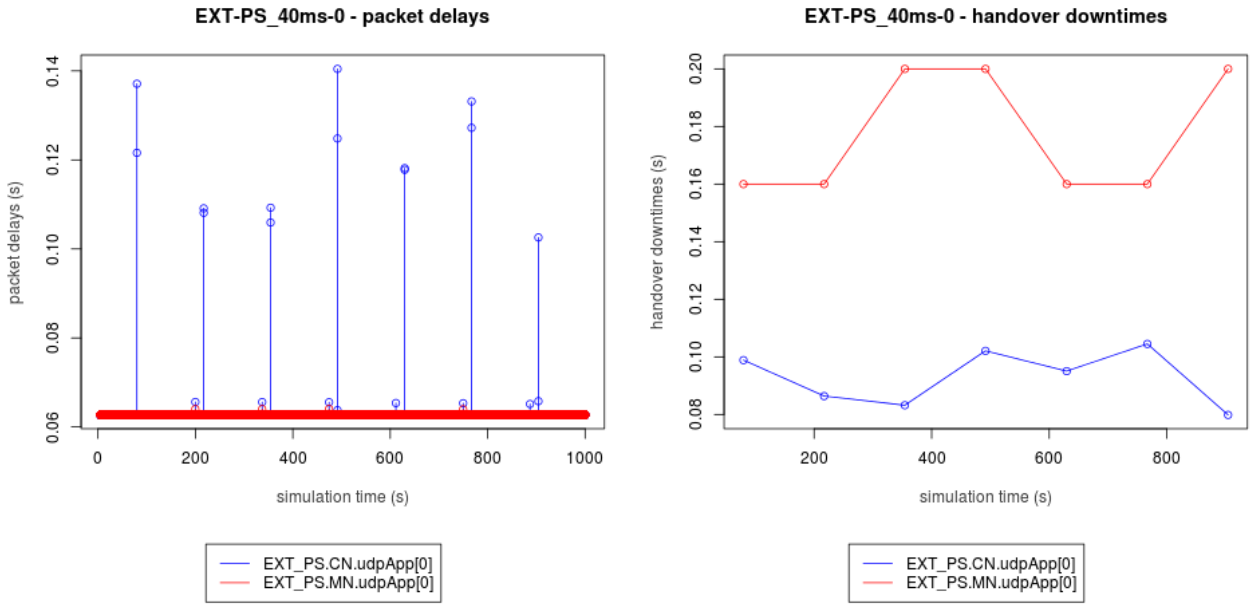

Fig. 11 – Latenza e intervalli di indisponibilità per la rete testABPS – proxy server esterno alle sottoreti (sendInterval = 40ms)

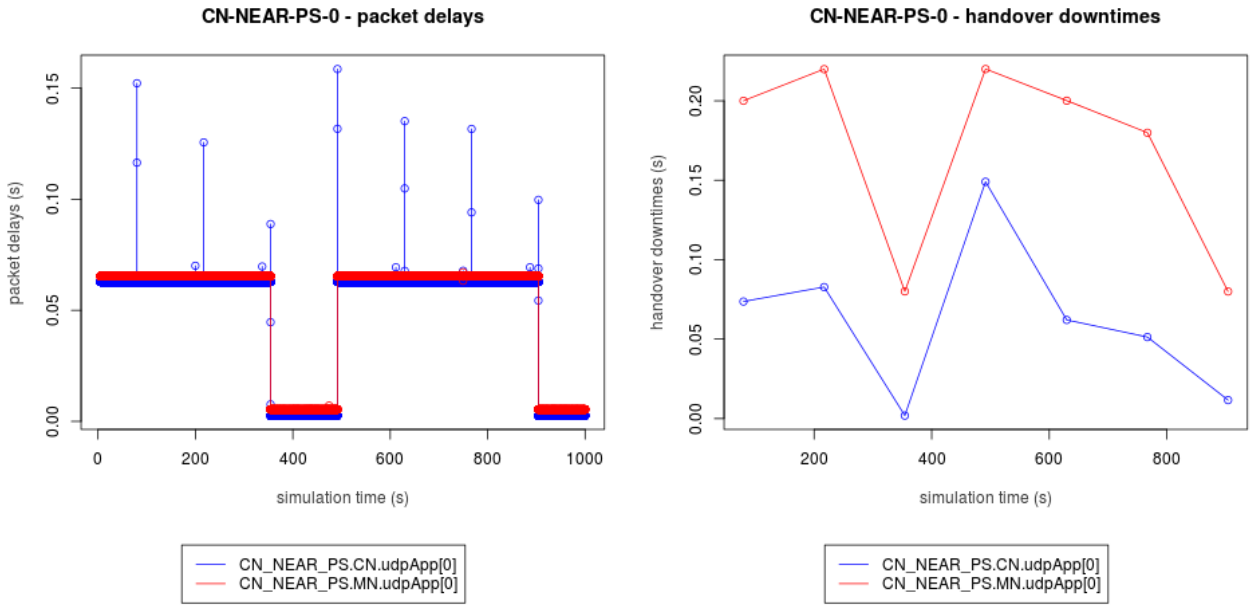

Fig. 12 – Latenza e intervalli di indisponibilità per la rete testABPS – proxy server nella sottorete del correspondent node (sendInterval = 20ms)

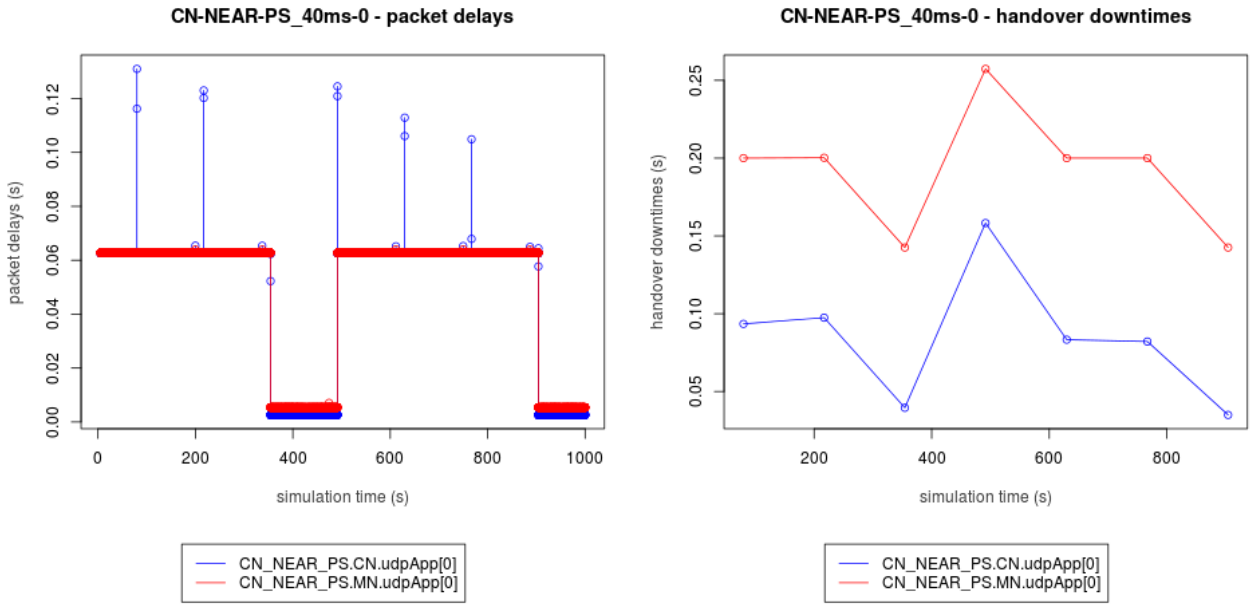

Fig. 13 – Latenza e intervalli di indisponibilità per la rete testABPS – proxy server nella sottorete del correspondent node (sendInterval = 40ms)

Le *Fig. 10* e *11* mostrano, come prevedibile, un andamento tendenzialmente regolare per quanto riguarda le misurazioni di latenza, salvo ovviamente gli occasionali picchi in corrispondenza dei cambi di rete da parte del nodo mobile, data la natura prettamente simmetrica dello scenario.
Più interessante notare invece come gli intervalli di indisponibilità subiscano questa volta un incremento decisamente più significativo al raddoppiare del periodo tra l'invio di due pacchetti consecutivi, in quanto tale parametro agisce in maniera diretta sulla misurazione.
Nelle *Fig. 12* e *13* emerge invece come evidente l'effetto della presenza del proxy server in una delle sottoreti; una volta che questa viene raggiunta dal mobile node (dopo tre operazioni di handover, come segnalato anche dai picchi di latenza), le comunicazioni diventano strettamente di carattere locale, in quanto anche il correspondent node si trova al suo interno: non dovendo più passare per la backbone, i ritardi nello scambio dei pacchetti subiscono una riduzione notevole, che si riflette anche sul tempo richiesto dall'handover, con cui è approssimata la durata dell'indisponibilità del MN.

6. Risultati sperimentali

6.1 Scenari urbani

Per fornire un riscontro più verosimile del funzionamento dei simulatori esaminati, questi sono stati testati all'interno di uno scenario urbano, con un maggiore numero di elementi interconnessi, e in cui i muri degli edifici sono rappresentati tramite ostacoli che vanno ad attenuare i segnali radio delle comunicazioni wireless.
Questo era già presente nel simulatore di ABPS, e per effettuare un confronto tra i protocolli è stato dunque necessario realizzarlo anche all'interno dei simulatori di MIPv6 e LISP, importando i moduli relativi agli ostacoli ed alla loro attenuazione del segnale.

La figura seguente mostra il percorso intrapreso dal nodo mobile attraverso l'ambiente cittadino:

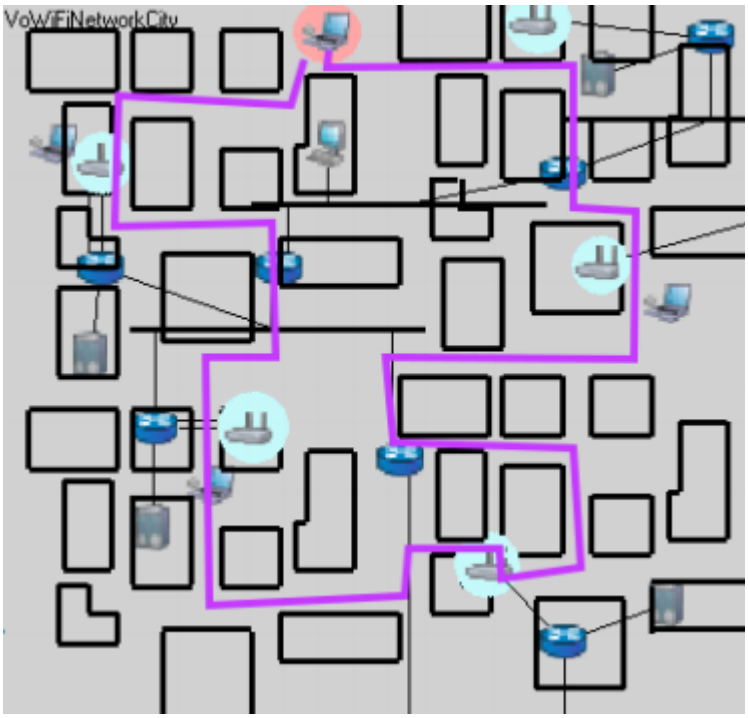

Fig. 14 – Percorso del nodo mobile attraverso lo scenario urbano

6.2 VoWiFiNetworkCity – MIPv6

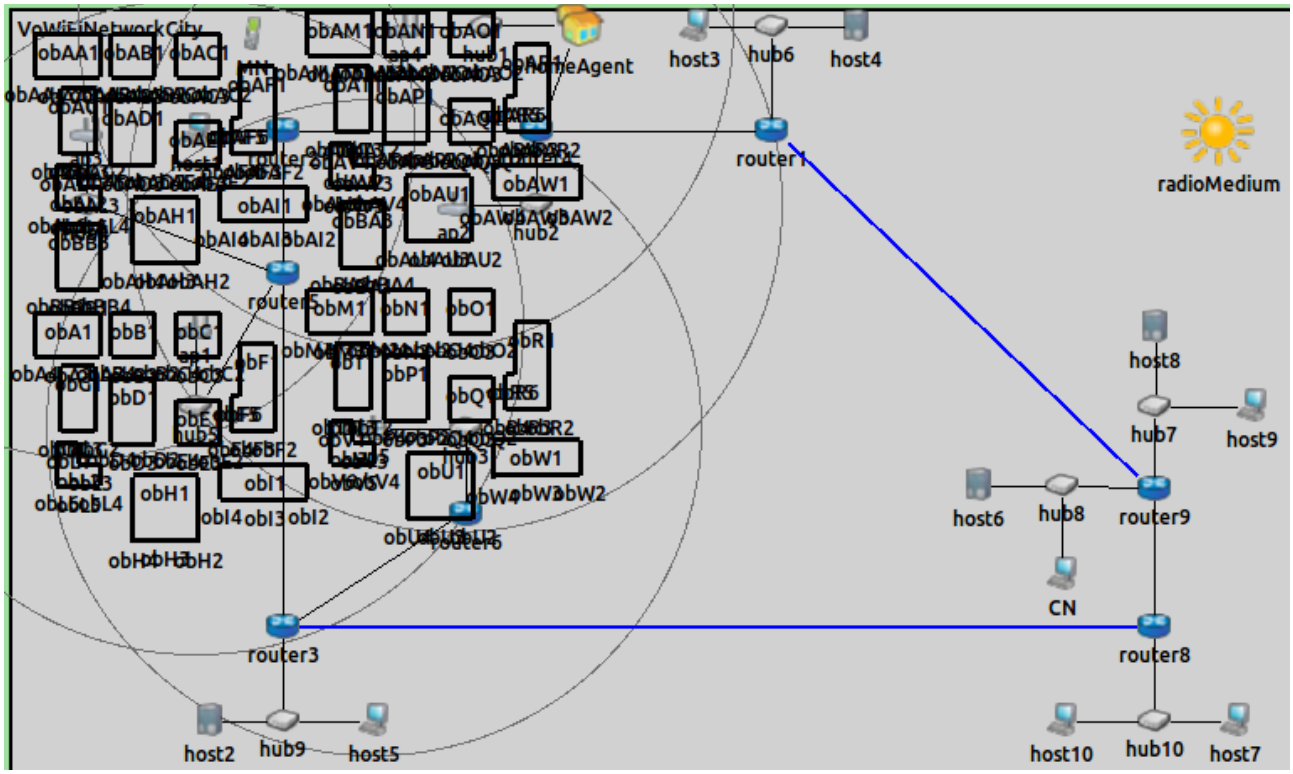

Fig. 15 – Scenario urbano VoWiFiNetworkCity – MIPv6

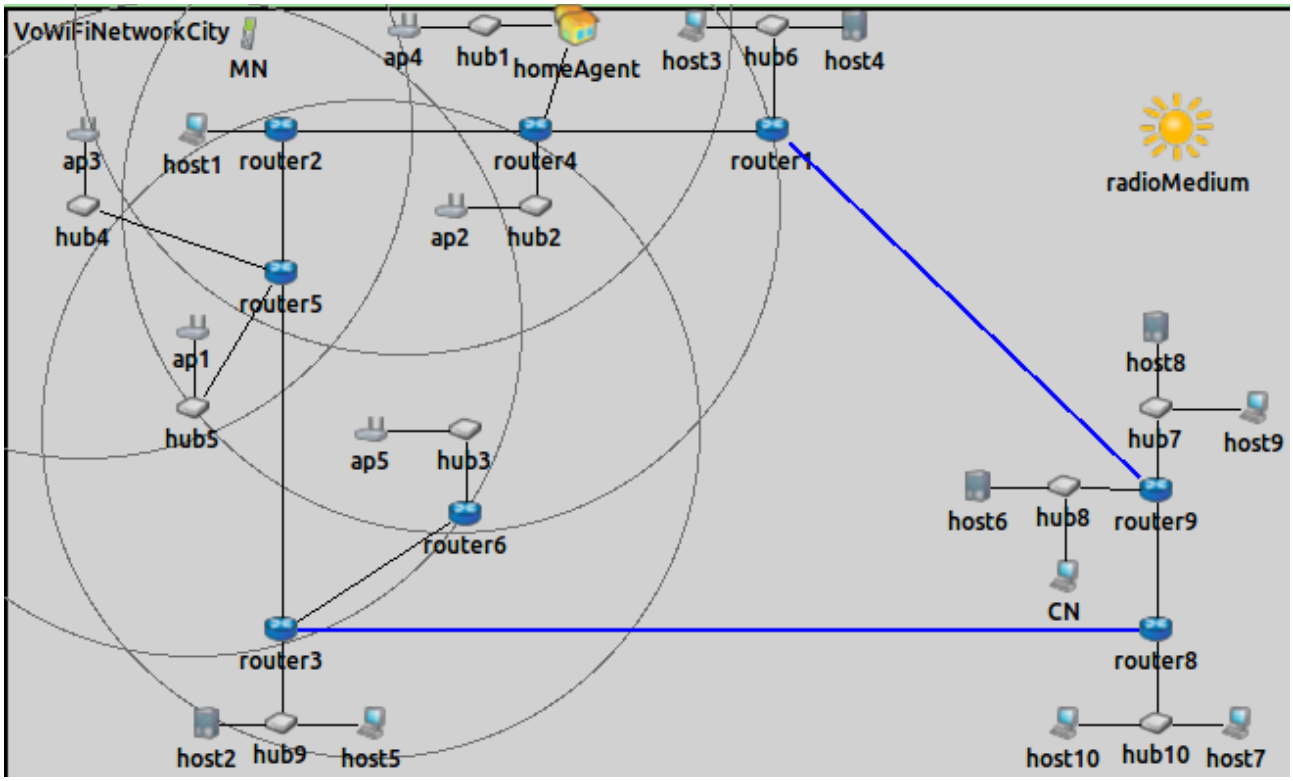

Fig. 16 – Infrastruttura di rete dello scenario urbano VoWiFiNetworkCity – MIPv6 (senza ostacoli)

Lo scenario realizzato all'interno del simulatore MIPv6 è riportato in *Fig. 15*.
La *Fig. 16* fornisce invece una visione della sola infrastruttura di rete, dopo che sono stati rimossi gli ostacoli; oltre che fornire una visione più chiara e meno ostruita delle connessioni tra gli elementi della rete, tale configurazione è stata effettivamente utilizzata per testare i protocolli in caso di assenza di ostruzioni del segnale.

I link evidenziati in blu sono link ad alta latenza (30ms, rispetto a 0.1µs degli altri), utilizzati per separare la sezione di destra, in cui è situato il correspondent node, da quella di sinistra, in cui si sposta il mobile node.
È in quest'ultima che è situato ovviamente anche l'home agent, in quanto è alla sua rete (la home network) che il nodo mobile is connetterà inizialmente, prima di procedere agli eventuali cambi di rete dovuti ai propri spostamenti attraverso lo scenario.

I principali parametri utilizzati per le simulazioni effettuate all'interno di tale ambiente urbano sono stati settati nella maniera seguente:

```
# ======================================================================
sim-time-limit = 3600s
**.ob*.penetrationLoss = -2dBm

# Accesspoint Configurations
**.AP1.wlan.mac.address = "10:00:00:00:00:00"
**.AP2.wlan.mac.address = "20:00:00:00:00:00"
**.AP3.wlan.mac.address = "30:00:00:00:00:00"
**.AP4.wlan.mac.address = "40:00:00:00:00:00"
**.AP5.wlan.mac.address = "50:00:00:00:00:00"
**.AP1.wlan.mgmt.ssid = "AP1"
**.AP2.wlan.mgmt.ssid = "AP2"
**.AP3.wlan.mgmt.ssid = "AP3"
**.AP4.wlan.mgmt.ssid = "AP4"
**.AP5.wlan.mgmt.ssid = "AP5"
**.MN.wlan[0].mac.address = "0A:AA:00:01:00:01"
**.MN.wlan[1].mac.address = "0A:AA:00:01:00:02"

# Radio Configurations
**.MN.wlan[0].radio.channelNumber = 1
**.MN.wlan[1].radio.channelNumber = 2
**.AP1.wlan.radio.channelNumber = 1
**.AP2.wlan.radio.channelNumber = 3
**.AP3.wlan.radio.channelNumber = 6
**.AP4.wlan.radio.channelNumber = 9
**.AP5.wlan.radio.channelNumber = 11

**.radio.bitrate = 54Mbps
**.radio.sensitivity = -68dBm
**.radio.transmitterPower = 32.0mW

# Agent
**.MN.wlan[*].agent.channelsToScan = "1 2 3 4 5 6 7 8 9 10 11"
**.wlan*.agent.probeDelay = 0.1s
```

```
# mobility
**.MN.mobilityType = "TurtleMobility"
**.MN.mobility.turtleScript = xmldoc("constPath.xml", "pre//nohtml//movement")

### UDP Apps ###
# MN ABPS UDP application
**.MN.numUdpApps = 1
**.MN.udpAppType = "ABPSUDPApp"
**.MN.udpApp[0].clientId = 1
**.MN.udpApp[0].receiverId = 2
**.MN.udpApp[0].destPort = 20
**.MN.udpApp[0].localPort = 80
**.MN.udpApp[0].clientDHCP = true
**.MN.udpApp[0].messageFreq = 100ms
**.MN.udpApp[0].startTime = 5s
**.MN.udpApp[0].messageLength =  512 byte

# CN ABPS UDP application
**.CN.numUdpApps = 1
**.CN.udpAppType = "ABPSUDPApp"
**.CN.udpApp[0].destPort = 80
**.CN.udpApp[0].localPort = 20
**.CN.udpApp[0].messageFreq = 100ms
**.CN.udpApp[0].startTime = 5s
**.CN.udpApp[0].clientDHCP = true
**.CN.udpApp[0].clientId = 2
**.CN.udpApp[0].receiverId = 1
**.CN.udpApp[0].messageLength =  512 byte

# Server proxy app
**.proxyServer.numUdpApps = 1
**.proxyServer.udpAppType = "ABPSProxyServerApp"
**.proxyServer.udpApp[0].clientId = 1

## DHCP server configuration
**.dhcp*.udpAppType = "ABPSUDPApp"
**.dhcp*.numUdpApps = 1

**.dhcp*.udpApp[0].acksRequired = false
**.dhcp*.udpApp[0].messageLength = 0
**.dhcp*.udpApp[0].messageFreq = 0
**.dhcp*.udpApp[0].startTime = 0

**.dhcp1.udpApp[0].clientId = 101
**.dhcp2.udpApp[0].clientId = 102
**.dhcp3.udpApp[0].clientId = 103
**.dhcp4.udpApp[0].clientId = 104
**.dhcp5.udpApp[0].clientId = 105

**.dhcp1.udpApp[0].ipAddressesDHCP = "128.96.4.141 … 128.96.4.149" #ip addrs for DHCP client
**.dhcp2.udpApp[0].ipAddressesDHCP = "128.96.4.11 … 128.96.4.19" #ip addrs for DHCP client
**.dhcp3.udpApp[0].ipAddressesDHCP = "128.96.6.11 … 128.96.6.19" #ip addrs for DHCP client
**.dhcp4.udpApp[0].ipAddressesDHCP = "128.96.5.11 … 128.96.5.19" #ip addrs for DHCP client
**.dhcp5.udpApp[0].ipAddressesDHCP = "128.96.5.141 … 128.96.5.149" #ip addrs for DHCP client

## Other app
**.host*.numUdpApps = 1
**.host*.udpAppType = "UDPBasicApp"

**.host1.udpApp[0].localPort = 1000
**.host1.udpApp[0].destinationNode = "host6"
**.host1.udpApp[0].destPort = 1000
**.host1.udpApp[0].messageLength = 100B
**.host1.udpApp[0].messageFreq = 122ms

**.host2.udpApp[0].localPort = 1000
**.host2.udpApp[0].destinationNode = "host7"
**.host2.udpApp[0].destPort = 1000
**.host2.udpApp[0].messageLength = 100B
```

```
**.host2.udpApp[0].messageFreq = 180ms

**.host3.udpApp[0].localPort = 1000
**.host3.udpApp[0].destinationNode = "host8"
**.host3.udpApp[0].destPort = 1000
**.host3.udpApp[0].messageLength = 100B
**.host3.udpApp[0].messageFreq = 156ms

**.host4.udpApp[0].localPort = 1000
**.host4.udpApp[0].destinationNode = "host9"
**.host4.udpApp[0].destPort = 1000
**.host4.udpApp[0].messageLength = 100B
**.host4.udpApp[0].messageFreq = 223ms

**.host5.udpApp[0].localPort = 1000
**.host5.udpApp[0].destinationNode = "host10"
**.host5.udpApp[0].destPort = 1000
**.host5.udpApp[0].messageLength = 100B
**.host5.udpApp[0].messageFreq = 100ms

**.host6.udpApp[0].localPort = 1000
**.host6.udpApp[0].destinationNode = "host1"
**.host6.udpApp[0].destPort = 1000
**.host6.udpApp[0].messageLength = 100B
**.host6.udpApp[0].messageFreq = 122ms

**.host7.udpApp[0].localPort = 1000
**.host7.udpApp[0].destinationNode = "host2"
**.host7.udpApp[0].destPort = 1000
**.host7.udpApp[0].messageLength = 100B
**.host7.udpApp[0].messageFreq = 180ms

**.host8.udpApp[0].localPort = 1000
**.host8.udpApp[0].destinationNode = "host3"
**.host8.udpApp[0].destPort = 1000
**.host8.udpApp[0].messageLength = 100B
**.host8.udpApp[0].messageFreq = 156ms

**.host9.udpApp[0].localPort = 1000
**.host9.udpApp[0].destinationNode = "host4"
**.host9.udpApp[0].destPort = 1000
**.host9.udpApp[0].messageLength = 100B
**.host9.udpApp[0].messageFreq = 223ms

**.host10.udpApp[0].localPort = 1000
**.host10.udpApp[0].destinationNode = "host5"
**.host10.udpApp[0].destPort = 1000
**.host10.udpApp[0].messageLength = 100B
**.host10.udpApp[0].messageFreq = 100ms
# ===================================================================
```

Sono poi state effettuate tre differenti simulazioni, aventi le seguenti caratteristiche:

- invio dei pacchetti ogni 20ms, con presenza di ostacoli;
- invio dei pacchetti ogni 40ms, con presenza di ostacoli
- invio dei pacchetti ogni 20ms, ostacoli non presenti.

I risultati così ottenuti sono riportati nei grafici seguenti.

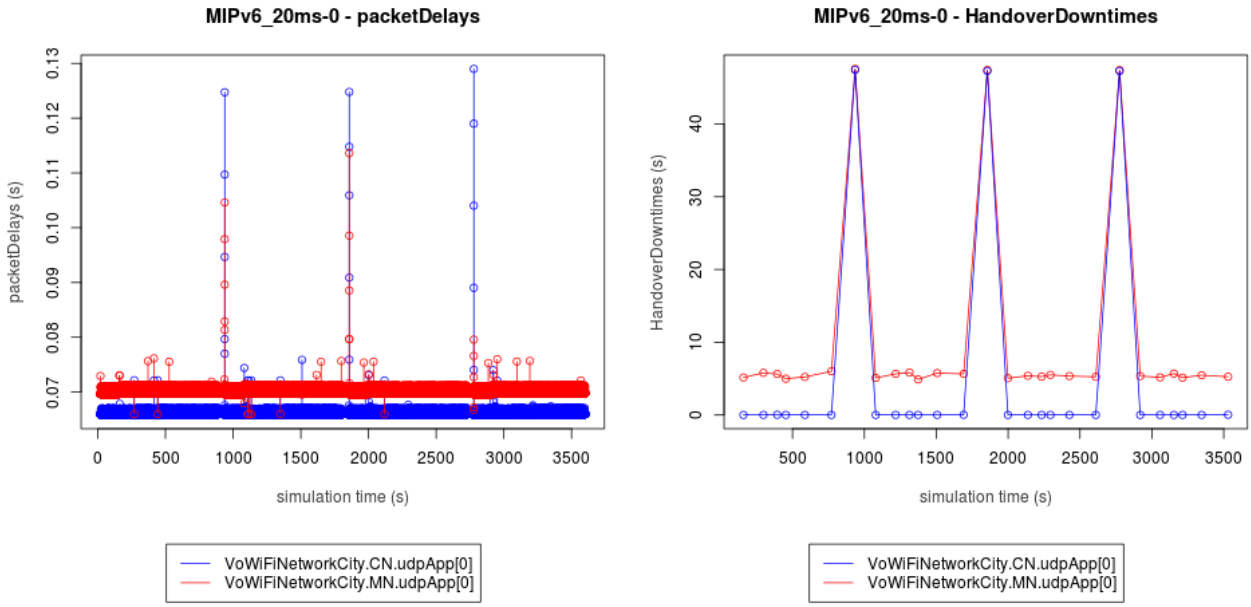

Fig. 17 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – MIPv6 (sendInterval = 20ms)

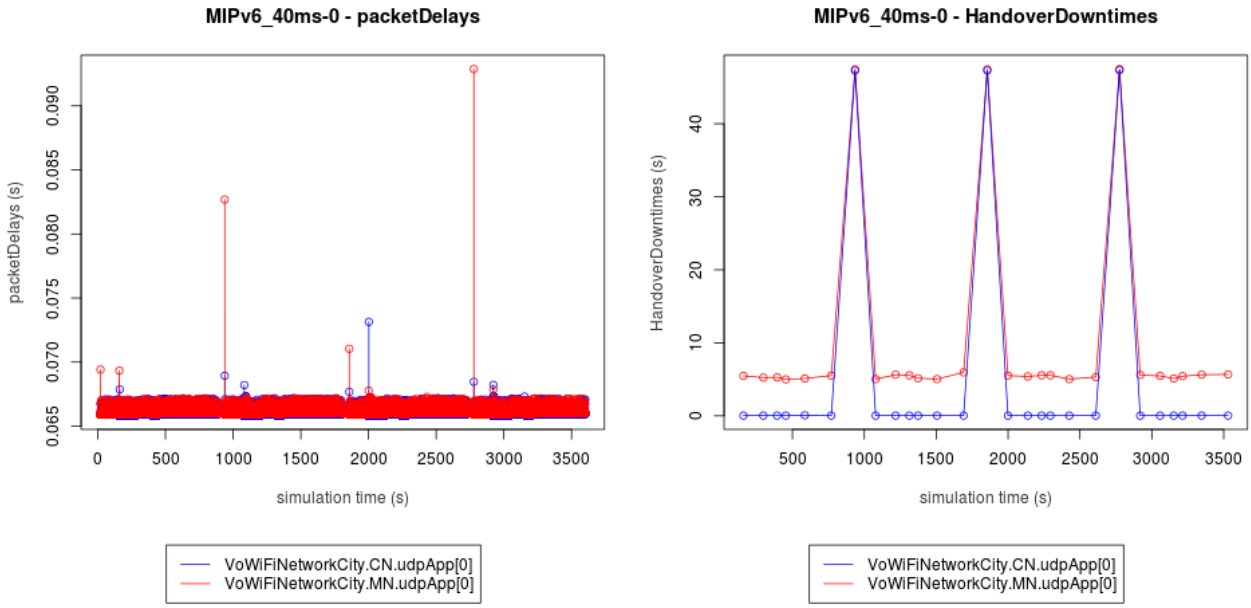

Fig. 18 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – MIPv6 (sendInterval = 40ms)

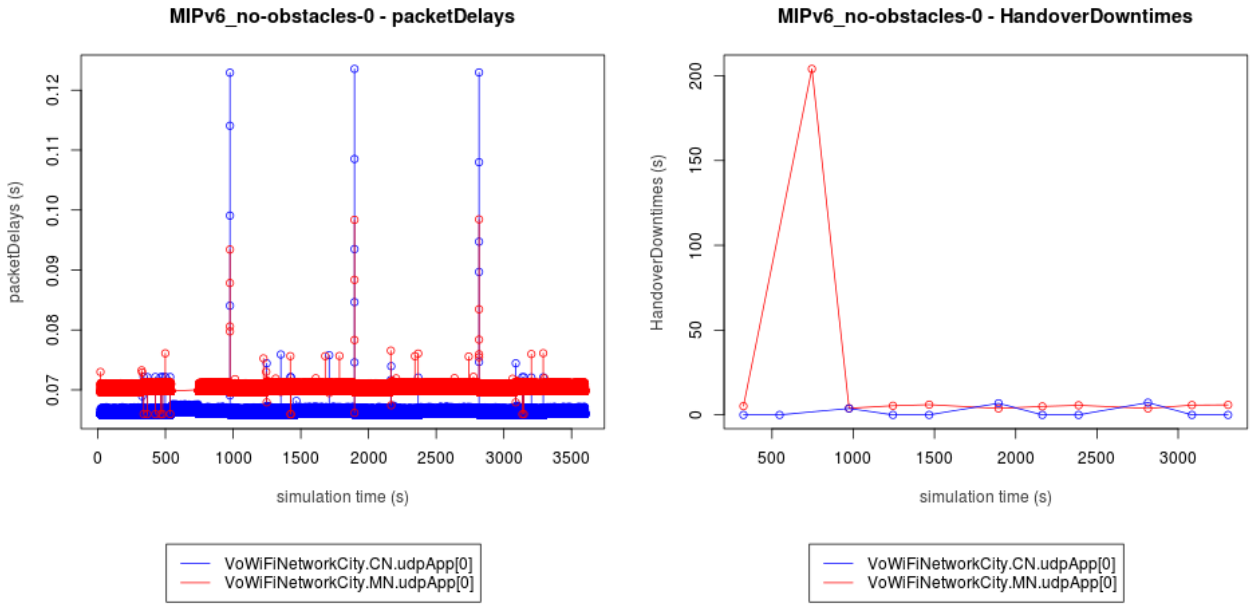

Fig. 19 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – MIPv6 (sendInterval = 20ms; ostacoli non presenti)

Le *Fig. 17* e *18* mostrano un andamento piuttosto regolare dei valori della latenza, che si stabilizza attorno ad un valore medio, tranne che per tre picchi appena prima dei 1000, 2000 e 3000 secondi di simulazione, rispettivamente, ai quali corrispondono tre intervalli di indisponibilità del MN di durata considerevole; questi sono causati dalla conformazione della rete, che presenta un punto in cui non vi è sufficiente copertura per potersi connettere ad un access point, raggiunto a questi tre istanti dal nodo mobile mentre ripete il suo percorso.
La *Fig. 19* evidenzia invece come cambia il comportamento della rete in seguito alla rimozione degli ostacoli:

- il CN continua a ricevere senza problemi pacchetti dal MN durante l'intera durata della simulazione, non essendo presenti zone prive di copertura dovute all'attenuazione imposta dagli ostacoli;
- il MN presenta il medesimo comportamento soltanto a partire dalla seconda ripetizione del proprio percorso attorno allo scenario; durante il suo primo giro presenta invece un intervallo di indisponibilità in ricezione di durata considerevole, dovuto con ogni probabilità ad una qualche anomalia nell'implementazione del protocollo MIPv6 che non è però ancora stato possibile individuare, e che richiederà ulteriori approfondimenti.

6.3 VoWiFiNetworkCity – LISP

Il medesimo scenario è stato realizzato anche all'interno del simulatore LISP dal collega E. Convertino; vengono qui riportate le due configurazioni di rete che sono state testate (gli ostacoli sono stati omessi), oltre al confronto tra i risultati ottenuti utilizzando i protocolli MIPv6 e LISP.

• CN-in-LISP-site

In questa configurazione, il correspondent node è situato all'interno di un sito LISP, e in particolare utilizza *lispRouter4* come punto di connessione alla rete LISP, tramite la quale può comunicare con il map server e raggiungere la sottoreti in cui si trova al momento il MN.
I link tratteggiati evidenziati in blu hanno una latenza di 30ms, mentre quelli grigi di 60ms.

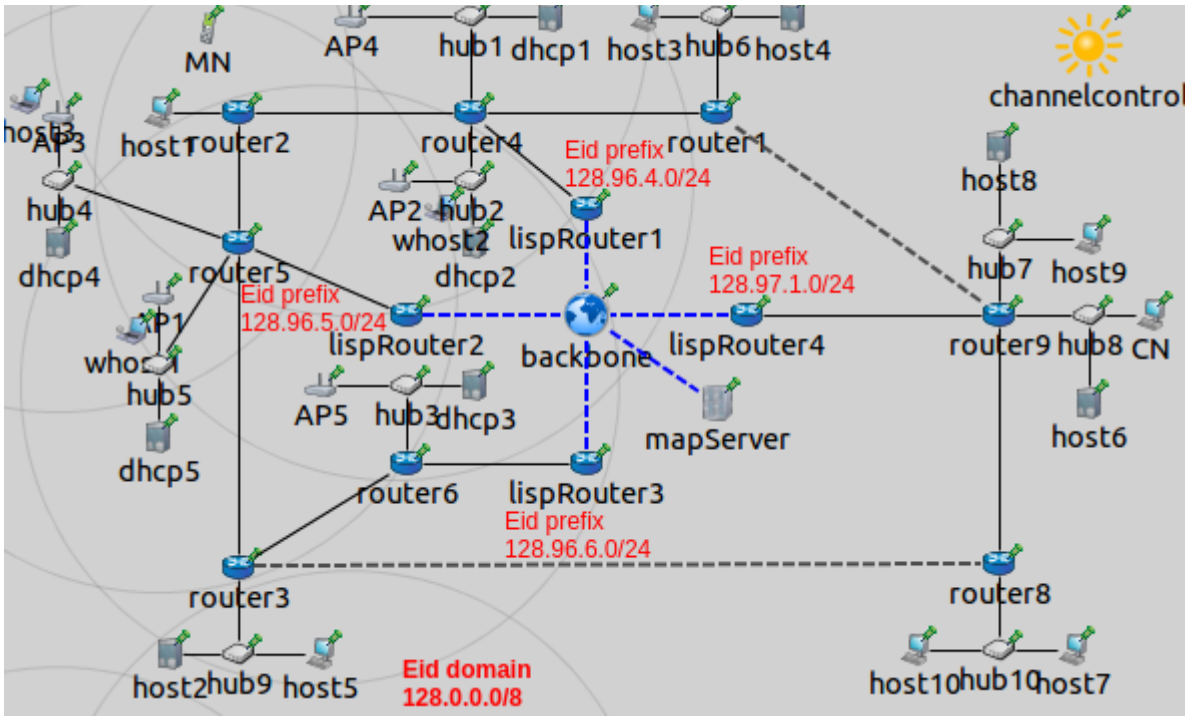

Fig. 20 – Scenario urbano VoWiFiNetworkCity – LISP – CN-in-LISP-site

Per permettere un confronto il più equo possibile, sono stati utilizzati gli stessi parametri (dove possibile) delle simulazioni effettuati nell'ambiente MIPv6, tramite le medesime configurazioni:

- invio dei pacchetti ogni 20ms, con presenza di ostacoli;
- invio dei pacchetti ogni 40ms, con presenza di ostacoli
- invio dei pacchetti ogni 20ms, ostacoli non presenti.

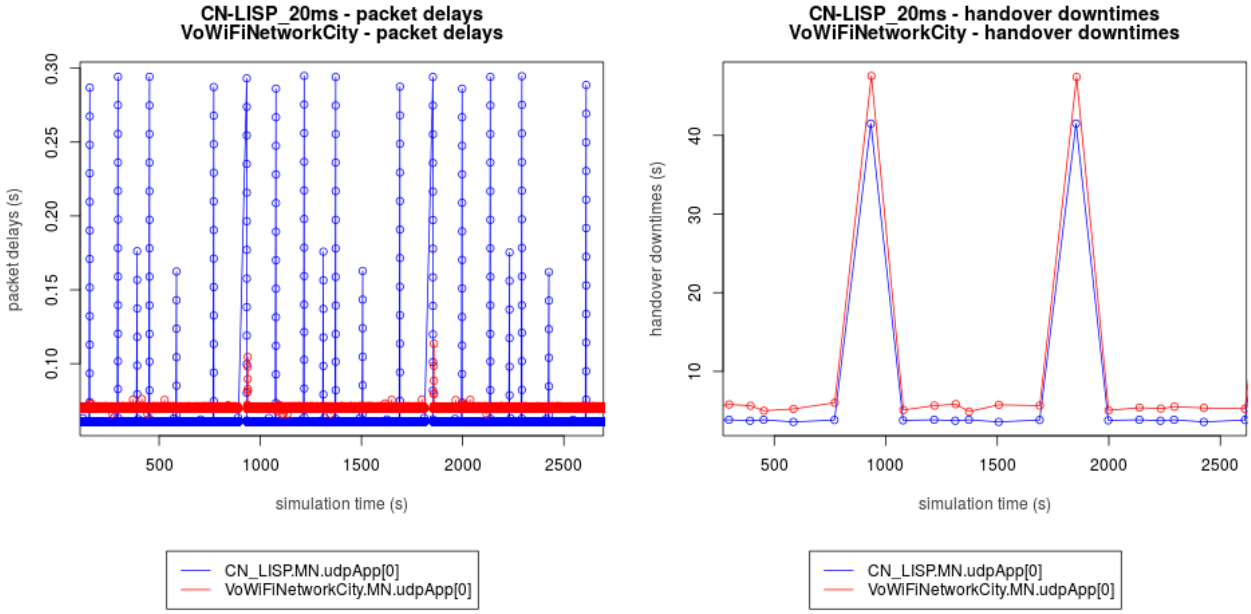

Fig. 21 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra LISP[CN-in-LISP-site] e MIPv6 (sendInterval = 20ms)

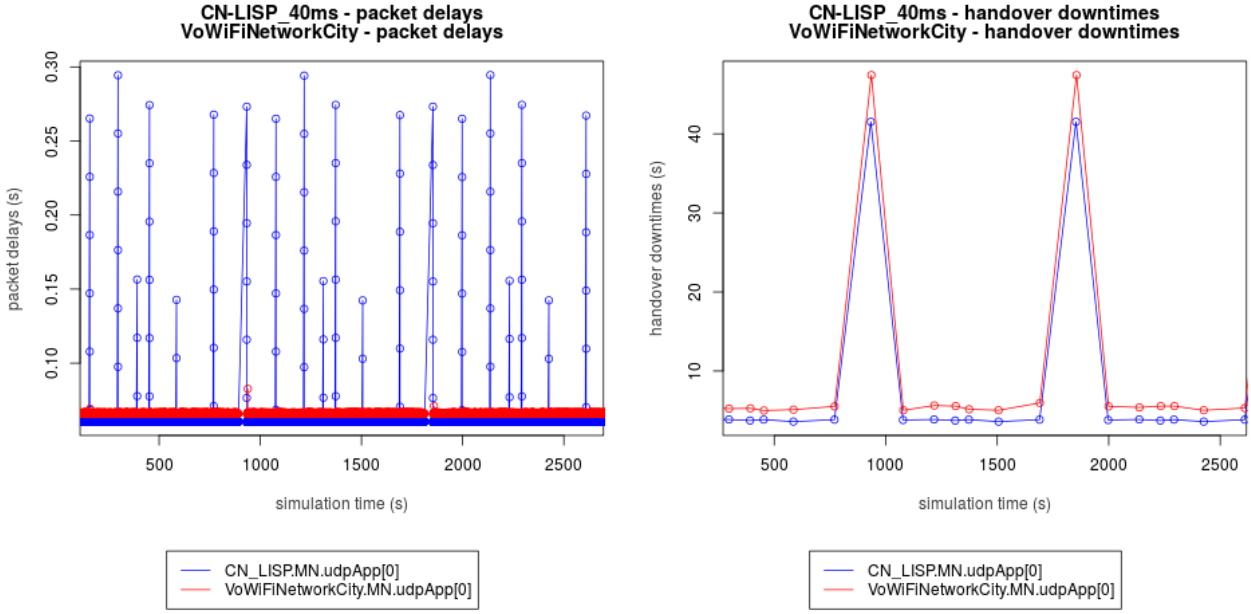

Fig. 22 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra LISP[CN-in-LISP-site] e MIPv6 (sendInterval = 40ms)

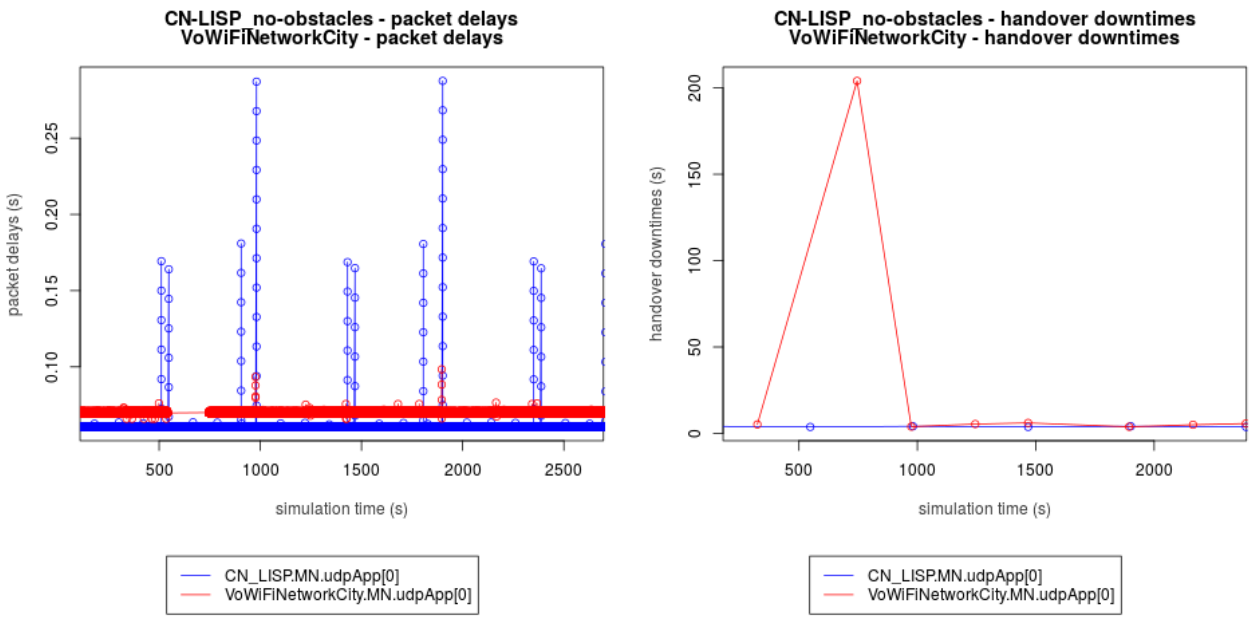

Fig. 23 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra LISP[CN-in-LISP-site] e MIPv6 (sendInterval = 20ms; ostacoli non presenti)

Per semplificarne l'esposizione, dati mostrati nelle *Fig. 21*, *22* e *23* si riferiscono solamente ai pacchetti ricevuti dal MN: il CN è in ogni caso un nodo fisso, e in quanto tale ragionevolmente più semplice da raggiungere; si è dunque preferito concentrarsi invece sul comportamento e sull'indisponibilità in ricezione del nodo mobile.

I risultati ottenuti evidenziano come le prestazioni di questa configurazione della rete LISP siano mediamente migliori di quelle offerte da MIPv6, sia in termini di durata degli intervalli di indisponibilità del MN e che di latenza nella ricezioni dei pacchetti.
Per quanto riguarda quest'ultima, tuttavia, si nota chiaramente come le riconfigurazioni a seguito di un cambio di rete comportino dei picchi di durata notevole, in confronto alla latenza media ed ai picchi presentati da MIPv6; questi sono dovuti al comportamento tenuto dai router LISP, che consiste nel conservare i pacchetti che non sono riusciti ad inoltrare (a causa del cambio di rete del MN), e che saranno poi inviati una volta risolta con successo la procedura di mapping: si può infatti notare come questi abbiano una disposizione estremamente regolare, dove lo scarto di latenza tra un picco e il successivo corrisponde all'intervallo compreso tra l'invio di due pacchetti consecutivi.

• CN-in-non-LISP-site

In questa configurazione alternativa, il correspondent node è invece situato all'interno di un sito non LISP, e *lispRouter4* viene utilizzato come proxy-ITR e proxy-ETR per permettergli di comunicare con il MN.

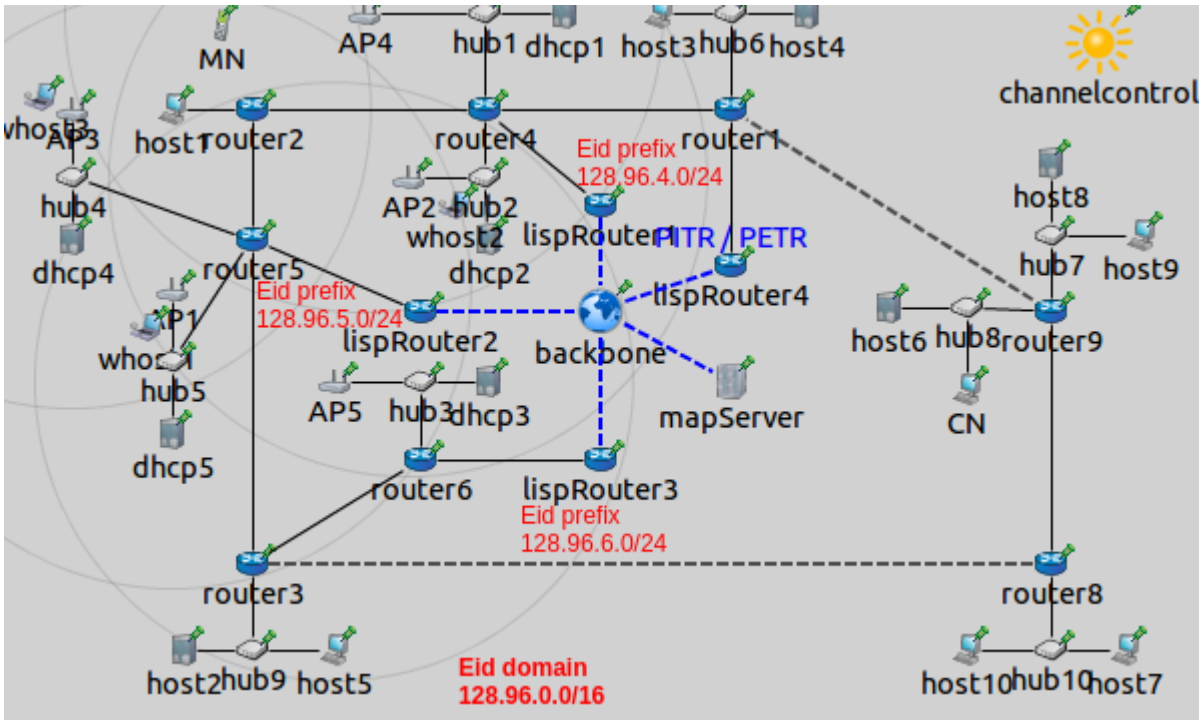

Fig. 24 – Scenario urbano VoWiFiNetworkCity – LISP – CN-in-non-LISP-site

Per testare questa configurazione, sono stati utilizzati i medesimi parametri, e sono state eseguite le tre simulazioni analoghe a quelle della configurazione precedente.
I risultati ottenuti da queste sono mostrati nelle *Fig. 25*, *26* e *27*, dove vengono nuovamente confrontati con quelli relativi allo scenario urbano del simulatore MIPv6.

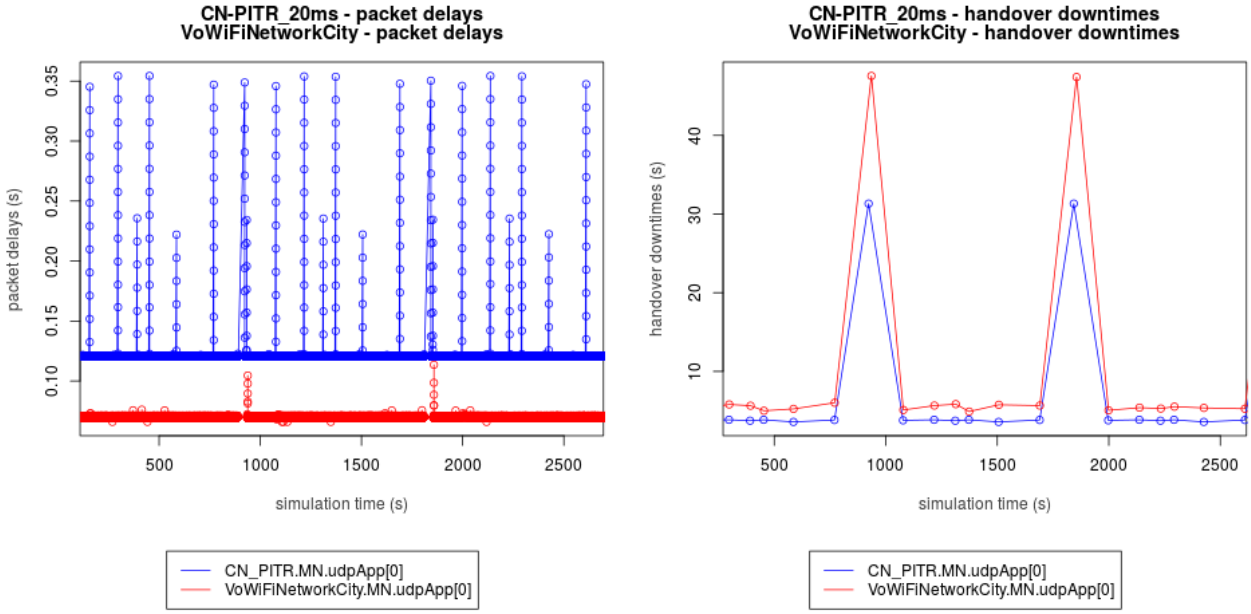

Fig. 25 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra LISP[CN-in-non-LISP-site] e MIPv6 (sendInterval = 20ms)

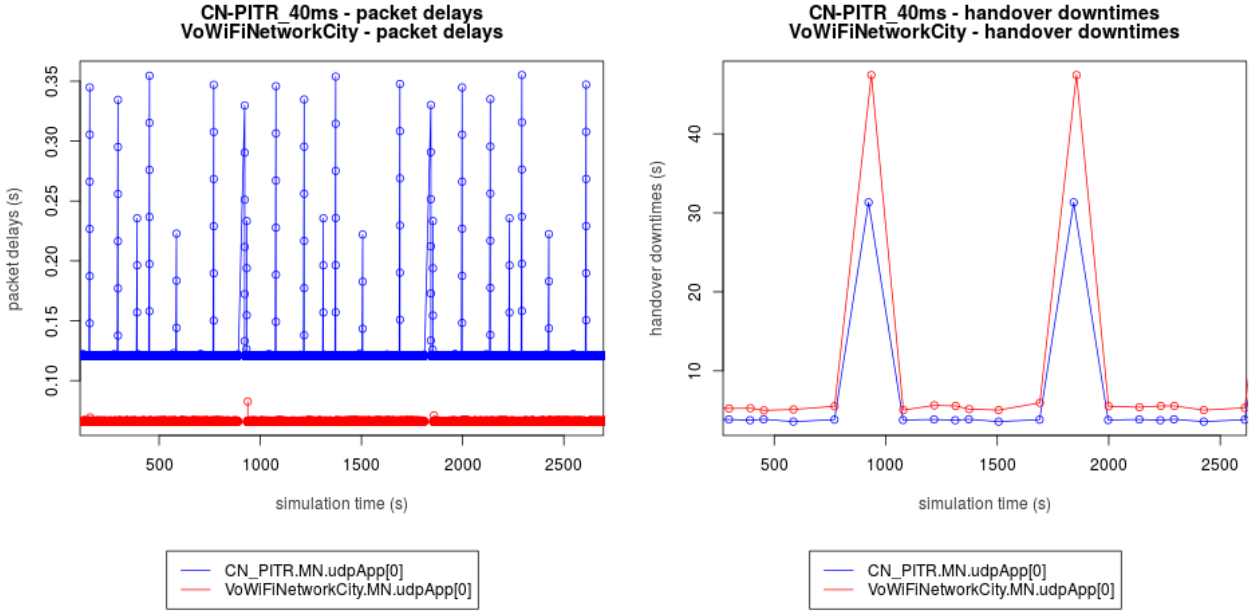

Fig. 26 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra LISP[CN-in-non-LISP-site] e MIPv6 (sendInterval = 40ms)

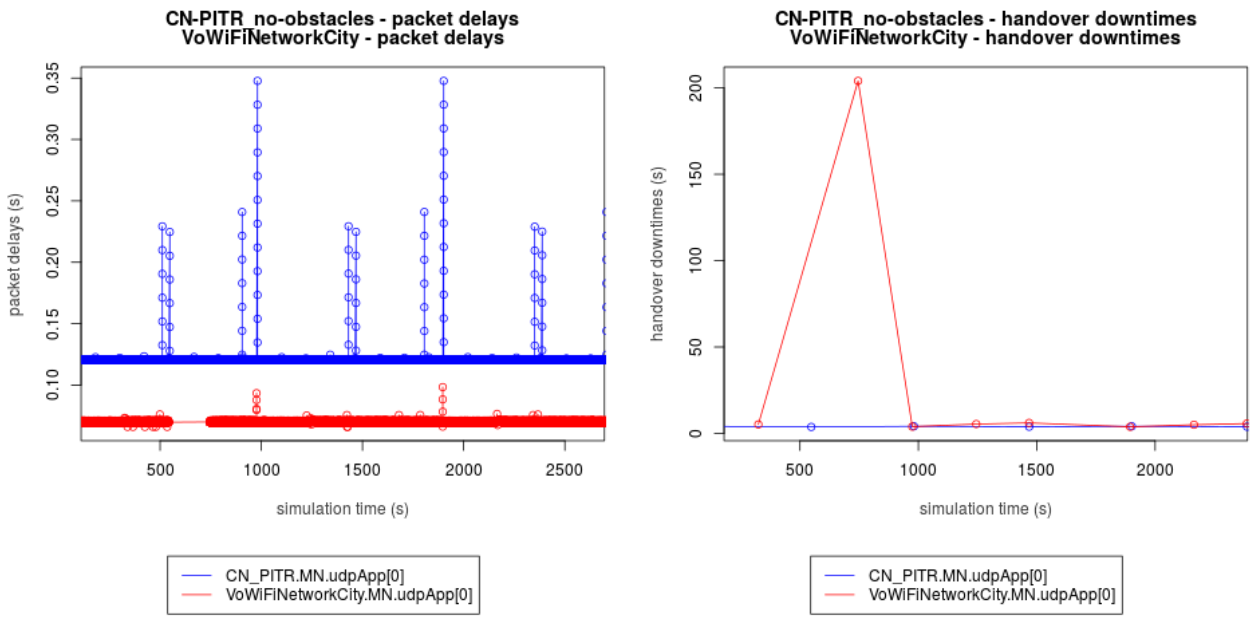

Fig. 27 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra LISP[CN-in-non-LISP-site] e MIPv6 (sendInterval = 20ms; ostacoli non presenti)

Nelle *Fig. 25* e *26* si presentano nuovamente i buchi di copertura dello scenario dovuti all'attenuazione del segnale da parte degli ostacoli, mentre non è ovviamente il caso della *Fig. 27*, relativa alla configurazione in cui gli ostacoli sono stati rimossi; in questa è nuovamente presente l'handover di durata anomala per MIPv6, di cui già discusso in precedenza (*vedi capitolo 6.3, Fig. 19*), e che non sembra però comparire nella simulazione effettuata con LISP: questo lascia pensare, come anticipato, che tale anomalia sia da ricondursi puramente all'implementazione del protocollo MIPv6, piuttosto, che alla conformazione della rete (come invece accadeva nel caso dei buchi di copertura evidenziati dai due grafici precedenti, che si manifestano per entrambi i protocolli).
Per quanto riguarda invece la latenza, vengono nuovamente mostrati i picchi dovuti al comportamento dei router LISP, come per la configurazione precedente; le performance di LISP sono però in questo caso mediamente inferiori a quelle di MIPv6, a causa della necessità di far transitare tutti i pacchetti attraverso il proxy-ITR/proxy-ETR (in base al verso della comunicazione) per poter raggiungere con successo il proprio interlocutore, aggiungendo così ritardi dovuti all'attraversamento di un maggior numero di hop all'interno della rete LISP.

6.4 VoWiFiNetworkCity – ABPS

Per la simulazione di ABPS nello scenario urbano sono stati realizzati tre scenari, che differiscono in base alla posizione in cui inserire il proxy server, ma aventi in ogni caso i seguenti parametri comuni:

```
# ======================================================================
sim-time-limit = 3600s
**.radio.transmitter.power = 32mW
**.wlan*.radio.receiver.sensitivity = -68dBm
**.carrierFrequency = 2.4GHz
**.ob*.penetrationLoss = -2dBm
**.neighbourDiscovery.minIntervalBetweenRAs = 0.03s #MinRtrAdvInterval (RFC 3775)
**.neighbourDiscovery.maxIntervalBetweenRAs = 0.07s #MaxRtrAdvInterval (RFC 3775)

# channel physical parameters
**.radioMedium.obstacleLossType = "SimpleObstacleLoss"
**.radioMedium.pathLossType = "SimplePathLoss"
*.radioMedium.mediumLimitCache.maxTransmissionPower = 32.0mW
*.radioMedium.mediumLimitCache.minReceptionPower = -82dBm
*.radioMedium.mediumLimitCache.minInterferencePower = -82dBm

### Access Point Configs
**.ap1.wlan*.mgmt.ssid = "AP1"
**.ap2.wlan*.mgmt.ssid = "AP2"
**.ap3.wlan*.mgmt.ssid = "AP3"
**.ap4.wlan*.mgmt.ssid = "AP4"
**.ap5.wlan*.mgmt.ssid = "AP5"

#  Radio
#**.mobilehost.wlan[0].radio.channelNumber = 1
**.MN*.wlan*.radio.channelNumber = 0
**.ap1.wlan*.radio.channelNumber = 1
**.ap2.wlan*.radio.channelNumber = 3
**.ap3.wlan*.radio.channelNumber = 6
**.ap4.wlan*.radio.channelNumber = 9
**.ap5.wlan*.radio.channelNumber = 11

# Agent
*.MN.wlan*.agent.channelsToScan = "1 2 3 4 5 6 7 8 9 10 11"
**.wlan*.agent.activeScan = true
**.wlan*.agent.probeDelay = 0.1s
**.wlan*.agent.authenticationTimeout = 5s
**.wlan*.agent.associationTimeout = 5s

# mobility
**.MN.mobilityType = "TurtleMobility"
**.MN.mobility.turtleScript = xmldoc("path_const.xml", "pre//nohtml//movement")

### UDP Apps ###
**.udpApp[0].startTime = 20s

**.MN.numUdpApps = 1
**.MN.udpApp*.typename = "UDPMIPApp"
**.MN.udpApp[0].localPort = 80
**.MN.udpApp[0].destPort = 80
**.MN.udpApp[0].destinationNode = "CN"
**.MN.udpApp[0].messageLength = 1024B
**.MN.udpApp[0].sendInterval = 40ms

**.CN.numUdpApps = 1
**.CN.udpApp*.typename = "UDPMIPApp"
**.CN.udpApp[0].localPort = 80
```

```
**.CN.udpApp[0].destPort = 80
**.CN.udpApp[0].destinationNode = "MN"
**.CN.udpApp[0].messageLength = 1024B
**.CN.udpApp[0].sendInterval = 40ms

**.host*.numUdpApps = 1
**.host*.udpApp*.typename = "UDPMIPApp"

**.host1.udpApp[0].localPort = 1000
**.host1.udpApp[0].destinationNode = "host6"
**.host1.udpApp[0].destPort = 1000
**.host1.udpApp[0].messageLength = 100B
**.host1.udpApp[0].sendInterval = 122ms

**.host2.udpApp[0].localPort = 1000
**.host2.udpApp[0].destinationNode = "host7"
**.host2.udpApp[0].destPort = 1000
**.host2.udpApp[0].messageLength = 100B
**.host2.udpApp[0].sendInterval = 180ms

**.host3.udpApp[0].localPort = 1000
**.host3.udpApp[0].destinationNode = "host8"
**.host3.udpApp[0].destPort = 1000
**.host3.udpApp[0].messageLength = 100B
**.host3.udpApp[0].sendInterval = 156ms

**.host4.udpApp[0].localPort = 1000
**.host4.udpApp[0].destinationNode = "host9"
**.host4.udpApp[0].destPort = 1000
**.host4.udpApp[0].messageLength = 100B
**.host4.udpApp[0].sendInterval = 223ms

**.host5.udpApp[0].localPort = 1000
**.host5.udpApp[0].destinationNode = "host10"
**.host5.udpApp[0].destPort = 1000
**.host5.udpApp[0].messageLength = 100B
**.host5.udpApp[0].sendInterval = 100ms

**.host6.udpApp[0].localPort = 1000
**.host6.udpApp[0].destinationNode = "host1"
**.host6.udpApp[0].destPort = 1000
**.host6.udpApp[0].messageLength = 100B
**.host6.udpApp[0].sendInterval = 122ms

**.host7.udpApp[0].localPort = 1000
**.host7.udpApp[0].destinationNode = "host2"
**.host7.udpApp[0].destPort = 1000
**.host7.udpApp[0].messageLength = 100B
**.host7.udpApp[0].sendInterval = 180ms

**.host8.udpApp[0].localPort = 1000
**.host8.udpApp[0].destinationNode = "host3"
**.host8.udpApp[0].destPort = 1000
**.host8.udpApp[0].messageLength = 100B
**.host8.udpApp[0].sendInterval = 156ms

**.host9.udpApp[0].localPort = 1000
**.host9.udpApp[0].destinationNode = "host4"
**.host9.udpApp[0].destPort = 1000
**.host9.udpApp[0].messageLength = 100B
**.host9.udpApp[0].sendInterval = 223ms

**.host10.udpApp[0].localPort = 1000
**.host10.udpApp[0].destinationNode = "host5"
**.host10.udpApp[0].destPort = 1000
**.host10.udpApp[0].messageLength = 100B
**.host10.udpApp[0].sendInterval = 100ms
# ===================================================================
```

Ciascuno è stato poi testato utilizzando nuovamente queste combinazioni di parametri aggiuntivi:

- invio dei pacchetti ogni 20ms, con presenza di ostacoli;
- invio dei pacchetti ogni 40ms, con presenza di ostacoli
- invio dei pacchetti ogni 20ms, ostacoli non presenti.

Negli schemi delle tre configurazioni, che verranno ora esaminati, gli elementi più importanti sono stati evidenziati con i colori seguenti:

- verde per il mobile node;
- rosso per il correspondent node;
- blu per proxy-server.

• External-Proxyserver

In questa prima configurazione, il proxy-server è connesso alla backbone della rete (i link tratteggiati sono nuovamente ad alta latenza, 30ms per quelli in blu, 60ms per quelli in grigio), all'esterno dunque sia delle sottoreti visitate dal mobile node che da quella in cui è situato il correspondent node.

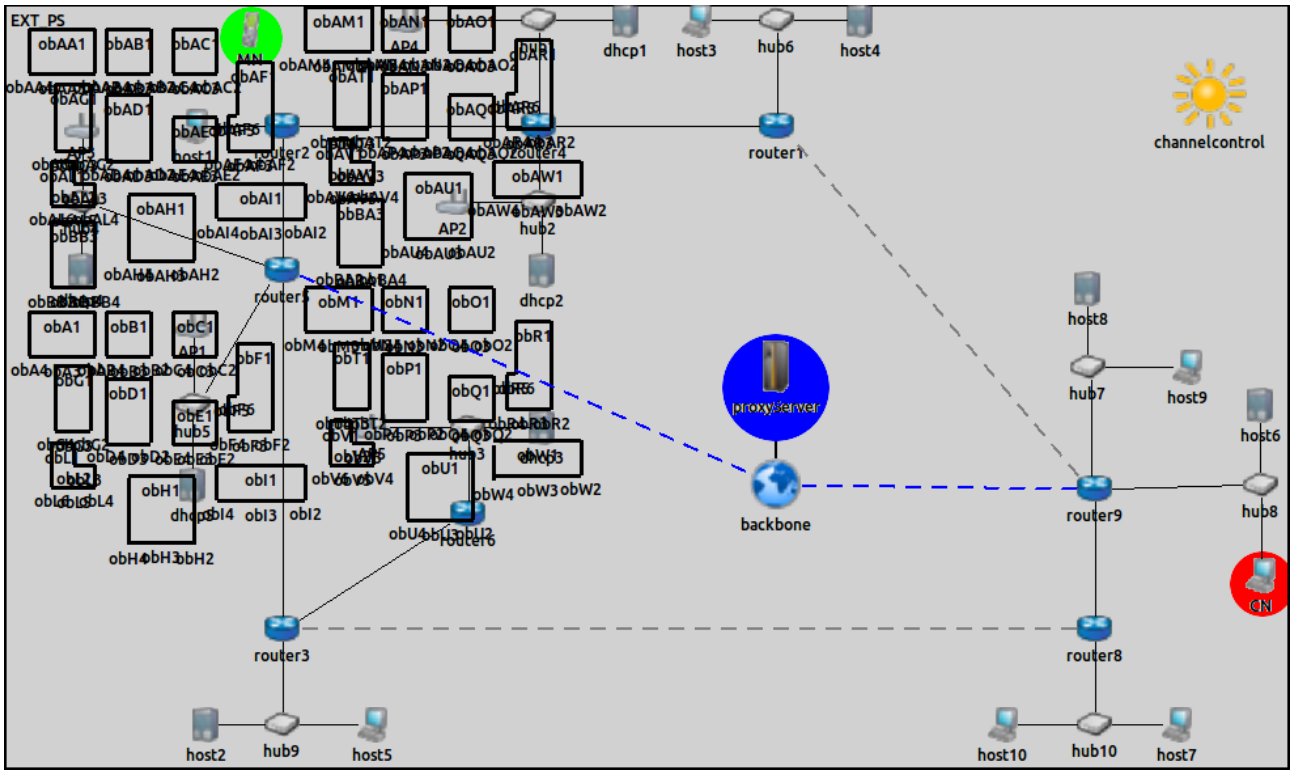

Fig. 28 – Scenario urbano VoWiFiNetworkCity – ABPS[External-Proxyserver]

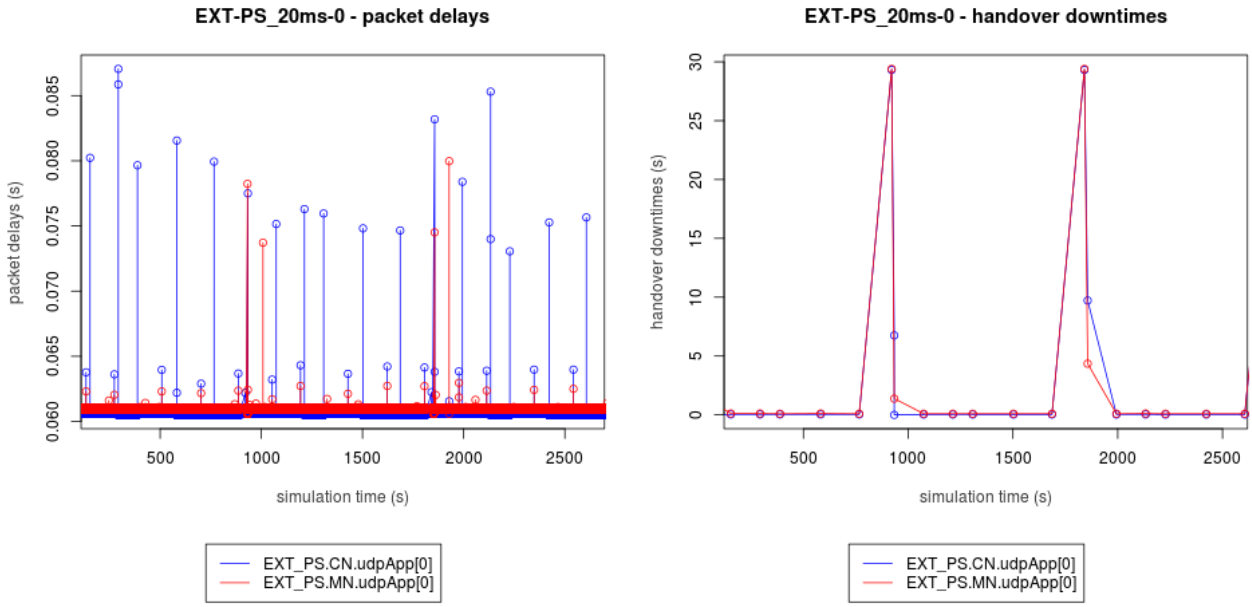

Fig. 29 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[External-Proxyserver] (sendInterval = 20ms)

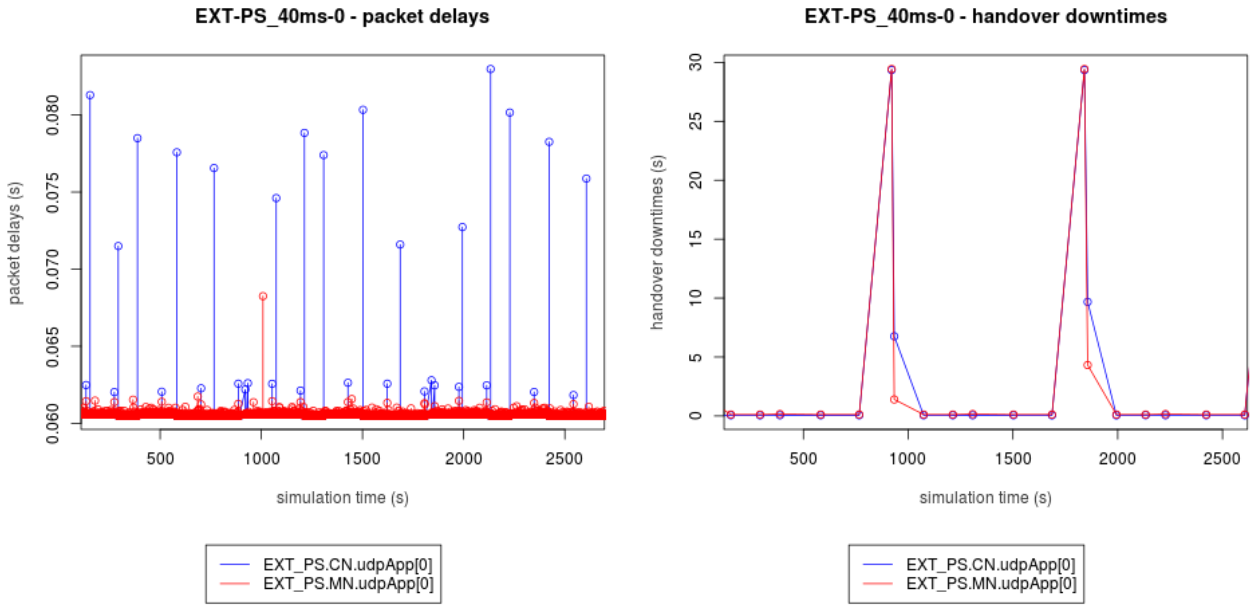

Fig. 30 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[External-Proxyserver] (sendInterval = 40ms)

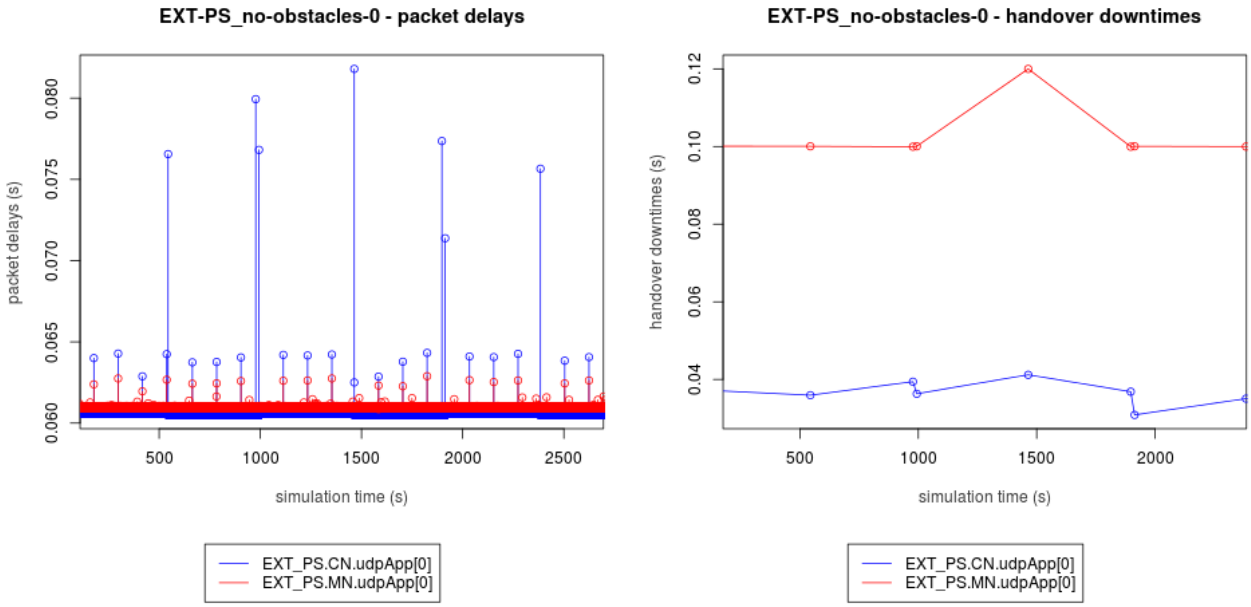

Fig. 31 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[External-Proxyserver] (sendInterval = 20ms; ostacoli non presenti)

Nelle *Fig. 29*, *30* e *31* sono riportati i risultati ottenuti da questa configurazione.
Nelle prime due si presentano nuovamente i medesimi buchi di copertura già incontrati in precedenza, ma una volta esclusi quelli si può notare come i tempi di indisponibilità del MN siano prossimi allo zero; questo è dovuto alla capacità di ABPS di sfruttare due diverse interfacce di rete contemporaneamente, permettendogli di passare immediatamente (o quasi) alla seconda nel caso la prima per la connessione al proprio access point.
Una volta rimossi gli ostacoli, gli intervalli di indisponibilità del MN diventano davvero minimi, in particolar modo per quanto riguarda l'invio dei pacchetti: avendo una seconda interfaccia già configurata ed connessa, il passaggio dalla prima ad essa è praticamente istantaneo.
Per quanto riguarda la latenza, essa presenta gli usuali picchi in corrispondenza dei cambi di rete da parte del MN, seppur di entità più contenuta rispetto a quanto visto per gli altri protocolli.

• **ProxyServer-near-MN**

La seconda configurazione prevede invece la presenza del proxy-server all'interno dell'area urbana, in una delle sottoreti visitate dal MN durante il proprio percorso.

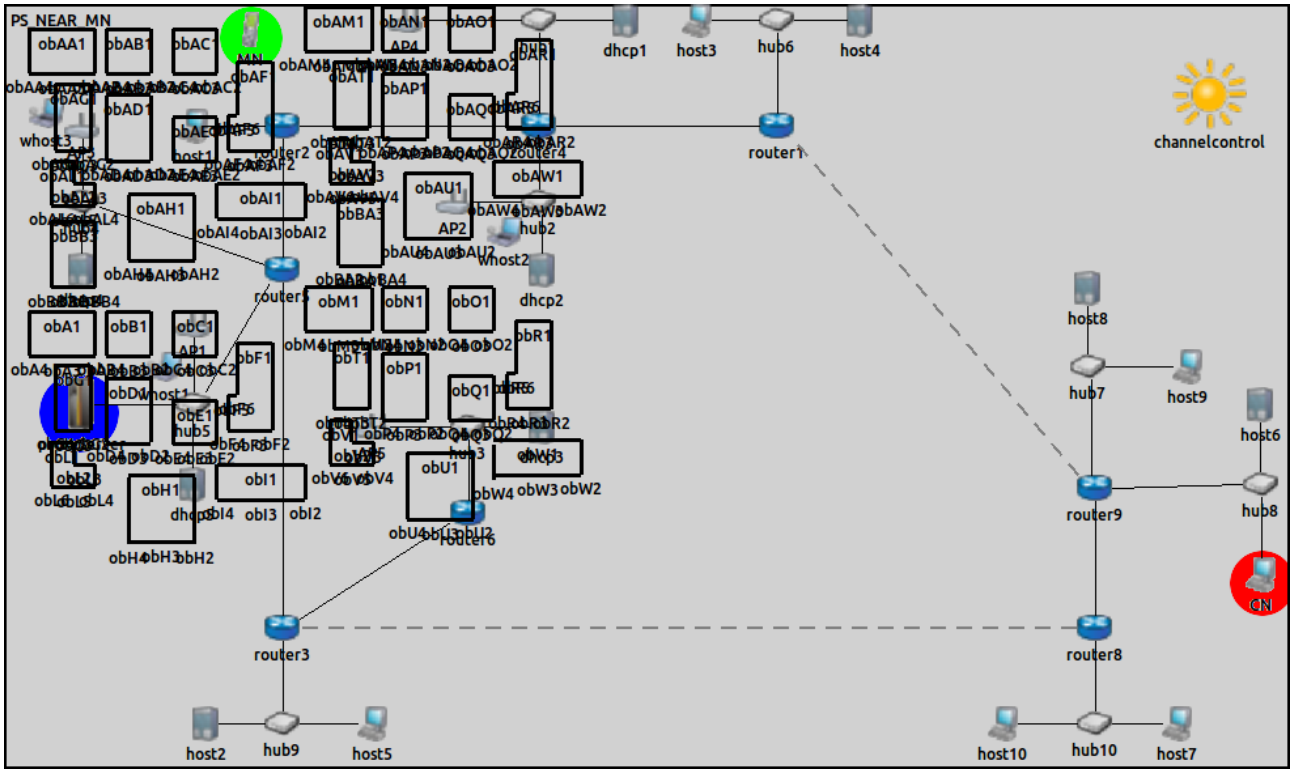

Fig. 32 – Scenario urbano VoWiFiNetworkCity – ABPS[ProxyServer-near-MN]

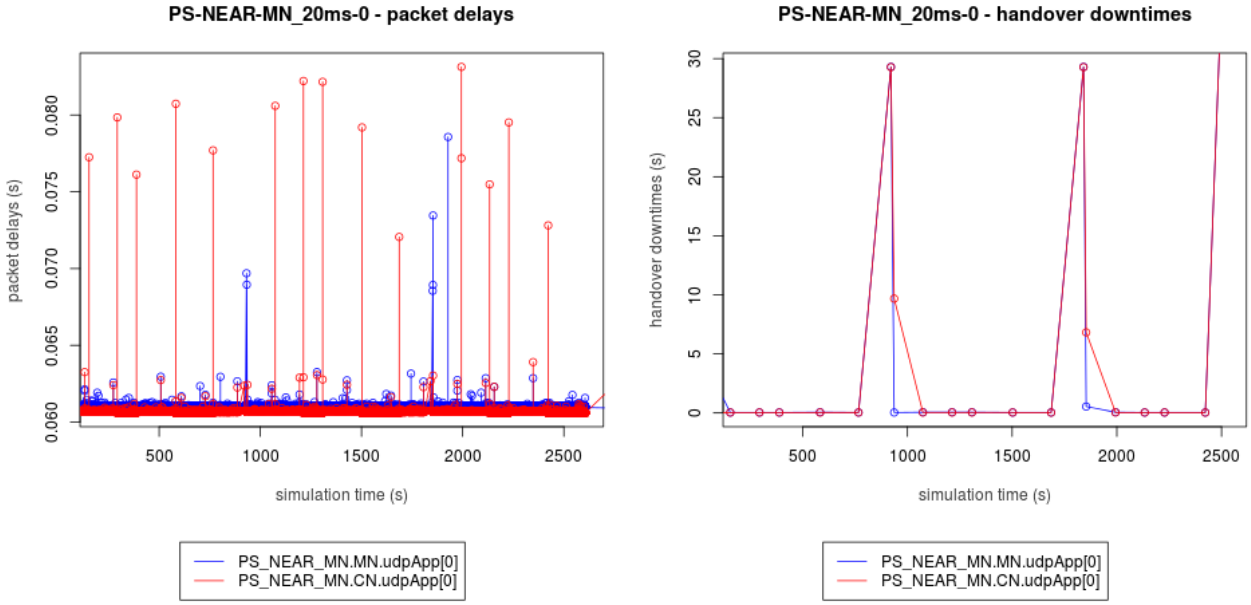

Fig. 33 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[ProxyServer-near-MN] (sendInterval = 20ms)

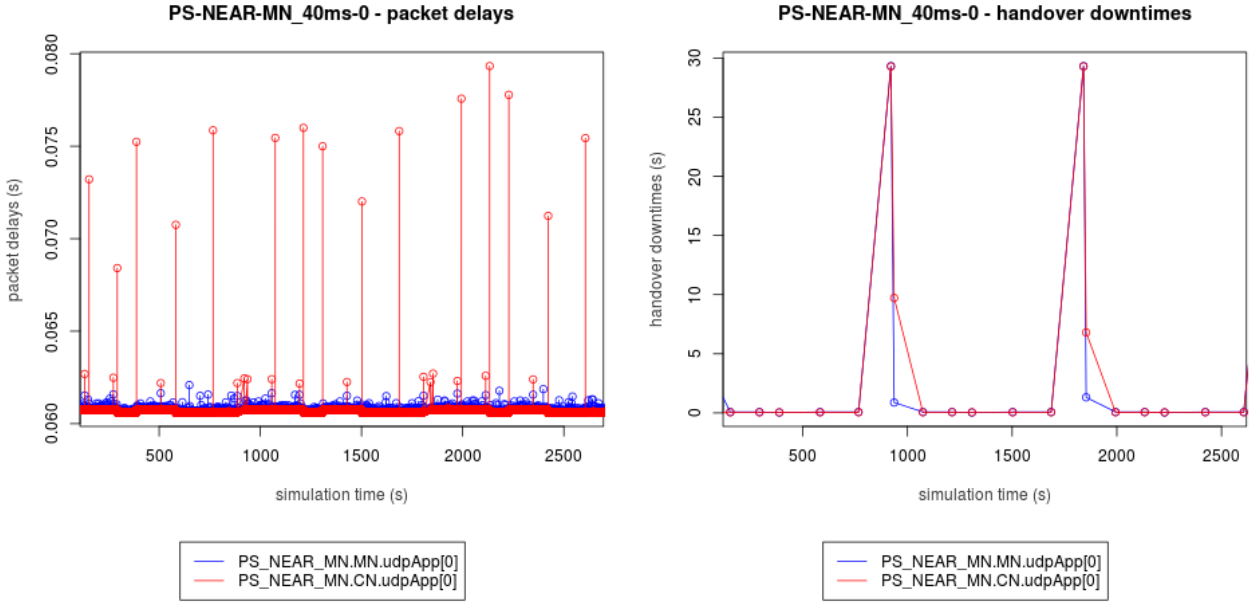

Fig. 34 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[ProxyServer-near-MN] (sendInterval = 40ms)

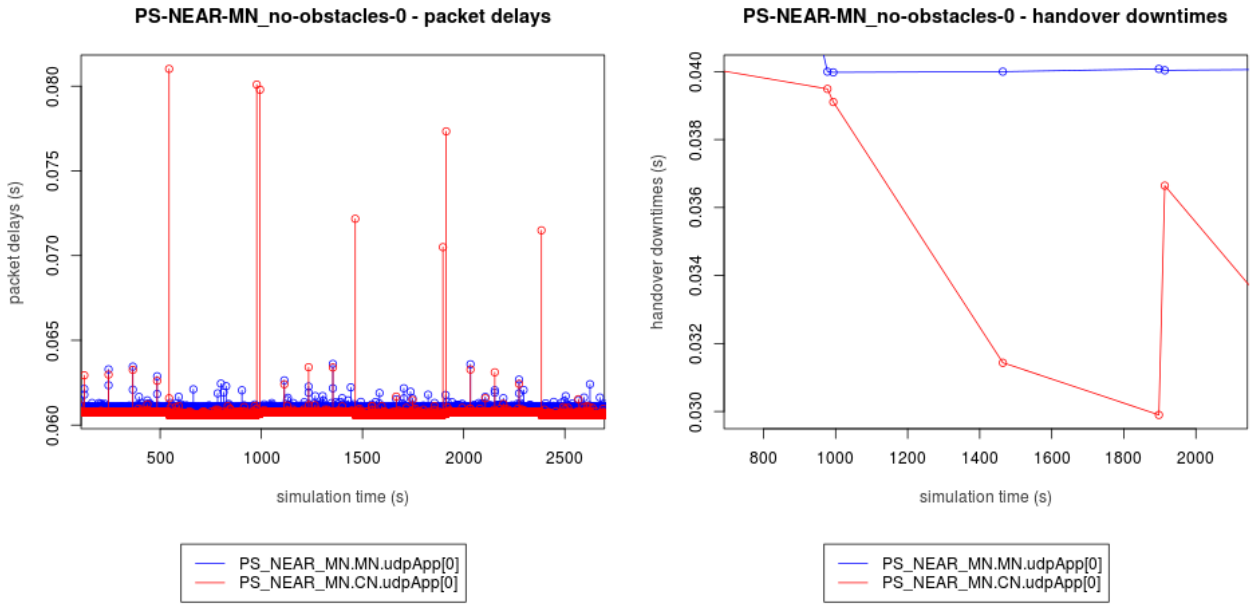

Fig. 35 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[ProxyServer-near-MN] (sendInterval = 20ms; ostacoli non presenti)

I risultati, riportati nelle *Fig. 33*, *34* e *35*, si presentano simili a quelli ottenuti dalla precedente configurazione; come prevedibile, tuttavia, ridurre le distanze tra proxy-server e mobile node ha avuto un effetto positivo sugli intervalli di indisponibilità, che risultano di durata mediamente inferiore, come si riesce ad apprezzare in particolare dalla simulazione senza ostacoli (in cui non compaiono i picchi dovuti ai buchi di copertura del segnale).

• CN-near-ProxyServer

La terza ed ultima configurazione realizzata agisce anch'essa sulla posizione del proxy-server, spostandolo questa volta però nella porzione di rete vicina al correspondent node.

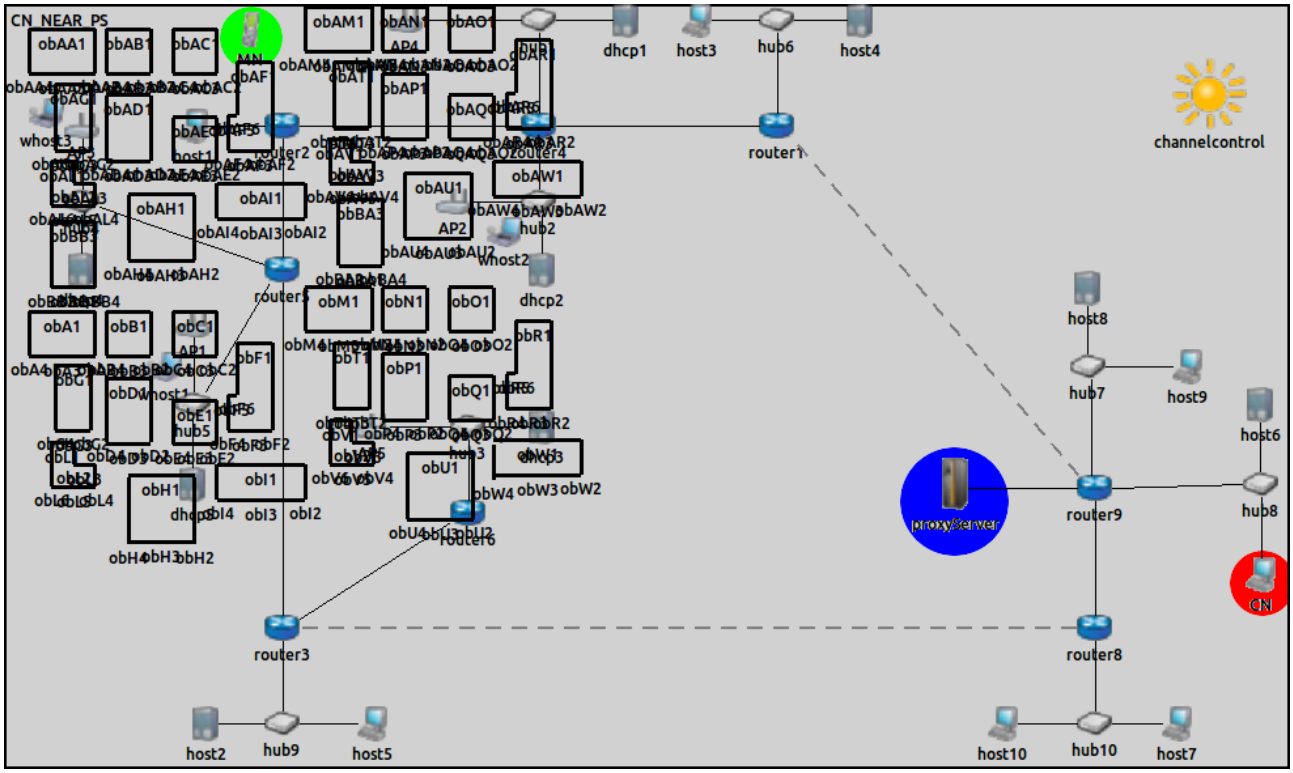

Fig. 36 – Scenario urbano VoWiFiNetworkCity – ABPS[CN-near-ProxyServer]

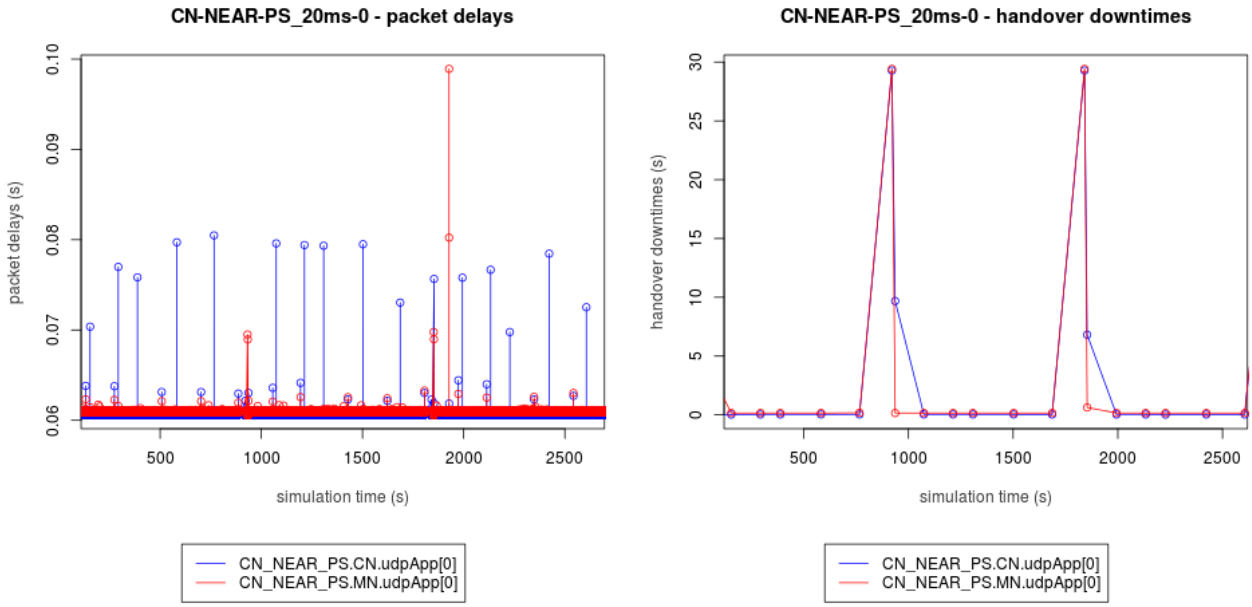

Fig. 37 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[CN-near-ProxyServer] (sendInterval = 20ms)

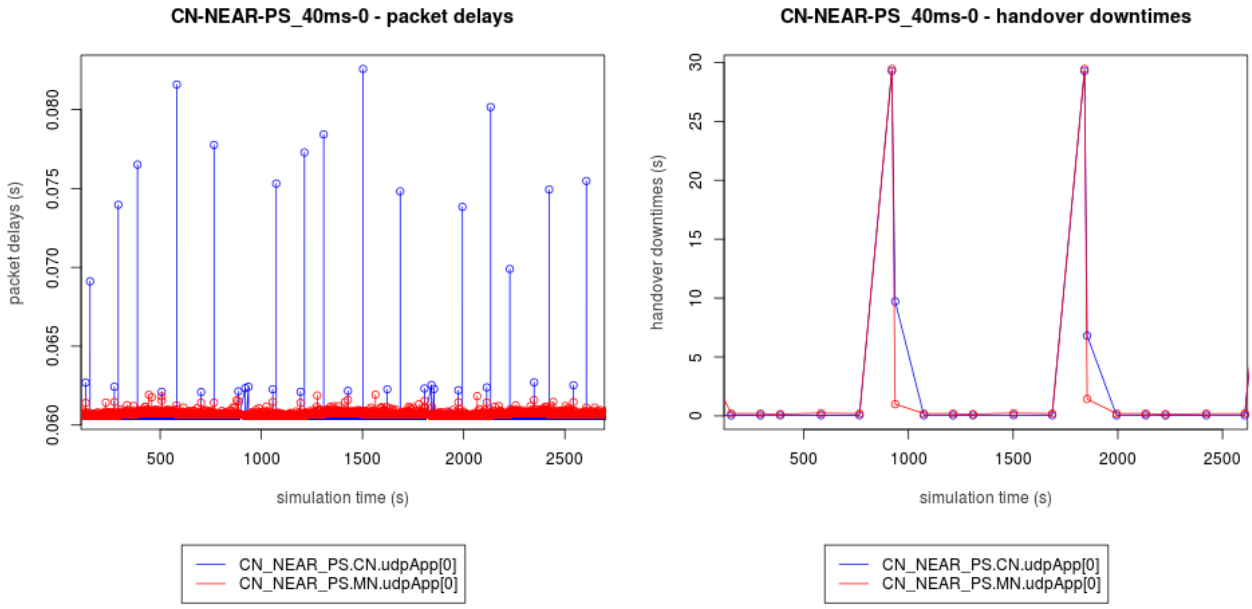

Fig. 38 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[CN-near-ProxyServer] (sendInterval = 40ms)

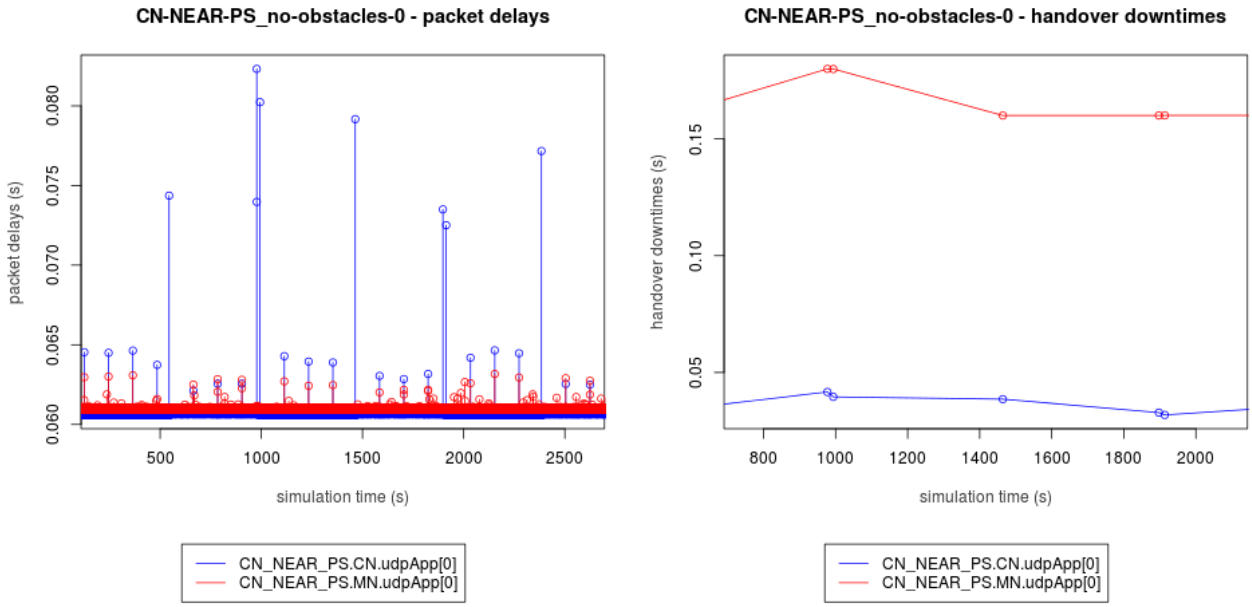

Fig. 39 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – ABPS[CN-near-ProxyServer] (sendInterval = 20ms; ostacoli non presenti)

Dalle misurazioni effettuate emerge, come mostrato nelle *Fig. 37*, *38* e *39*, un andamento analogo ai precedenti per i valori della latenza, mentre per quanto riguarda gli intervalli di indisponibilità si registrano, come previsto, delle attese dovute agli handover di durata leggermente superiore rispetto agli altri due casi, dovute alla maggior lontananza tra MN e server-proxy in termini di hop di rete attraversati.

6.5 Considerazioni conclusive

Partendo dai risultati ottenuti dalle simulazioni effettuate, è infine possibile effettuare un confronto prestazionale tra le architetture di supporto alla mobilità di Mobile IPv6 e Always Best Packet Switching.

Le *Fig. 40*, *41* e *42* riportano le misurazioni relative agli scenari urbani realizzati all'interno dei due simulatori, dove lo scenario che fa uso di MIPv6 viene confrontato alle tre differenti configurazioni di quelli che utilizzano invece ABPS, e che differiscono per la posizione del proxy-server, che si ricordano essere le seguenti:

- *External-Proxyserver*, in cui il proxy-server è connesso alla backbone della rete, ed è dunque esterno sia alle sottoreti visitate dal MN che a quella in cui è situato il CN;
- *ProxyServer-near-MN*, dove il proxy-server è collocato nella sezione della rete attorno alla quale si sposta il nodo mobile;
- *CN-near-ProxyServer*, dove questo si trova invece nella stessa zona del correspondent node, che è separata da quella del mobile node tramite link ad alta latenza (60ms).

Per semplificare l'esposizione dei risultati, si è nuovamente optato (come per il confronto tra MIPv6 e LISP) di omettere i dati relativi ai pacchetti ricevuti dal correspondent node, e di concentrarsi solamente su quelli ricevuti dal mobile node, che a causa dei suoi spostamenti e conseguenti cambi di indirizzo diventa più difficile da raggiungere rispetto al CN, che si ricorda essere un nodo fisso.

Per brevità, sono inclusi soltanto i confronti relativi alla configurazione standard, ovvero quella in cui MN e CN inviano pacchetti ogni 20ms, e dove è prevista la presenza di muri che attenuano i segnali radio; in particolare, i dati a cui si fa riferimento sono quelli già riportati nella *Fig. 17* per MIPv6, ed nelle *Fig. 29*, *33* e *37* per ABPS.

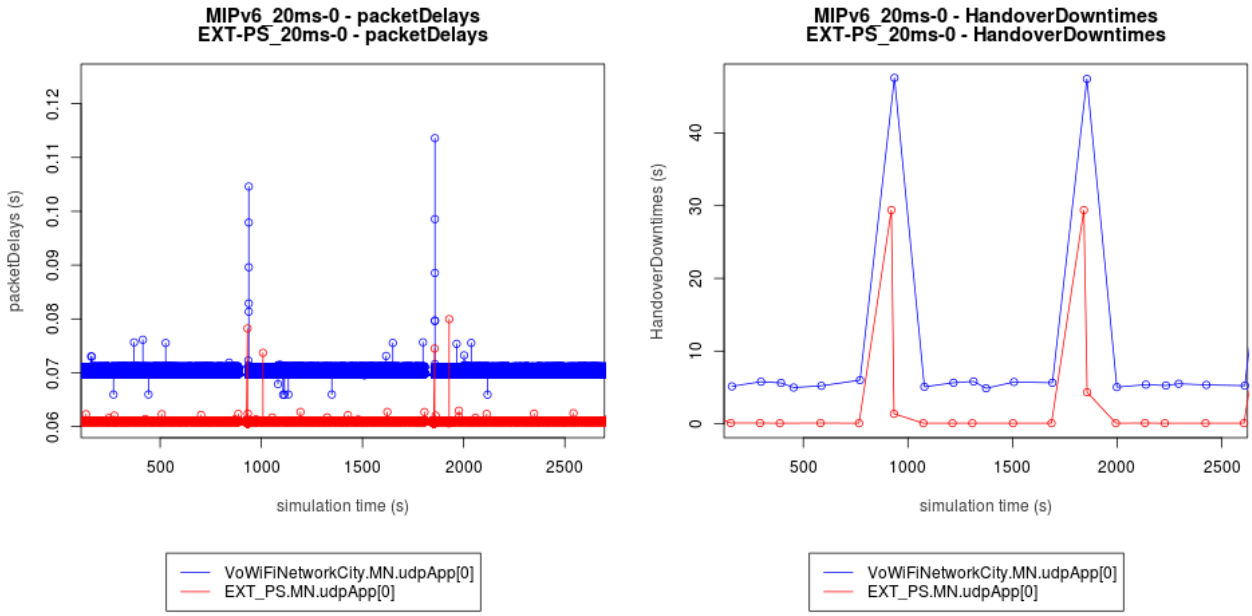

Fig. 40 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra ABPS[External-Proxyserver] e MIPv6 (sendInterval = 20ms)

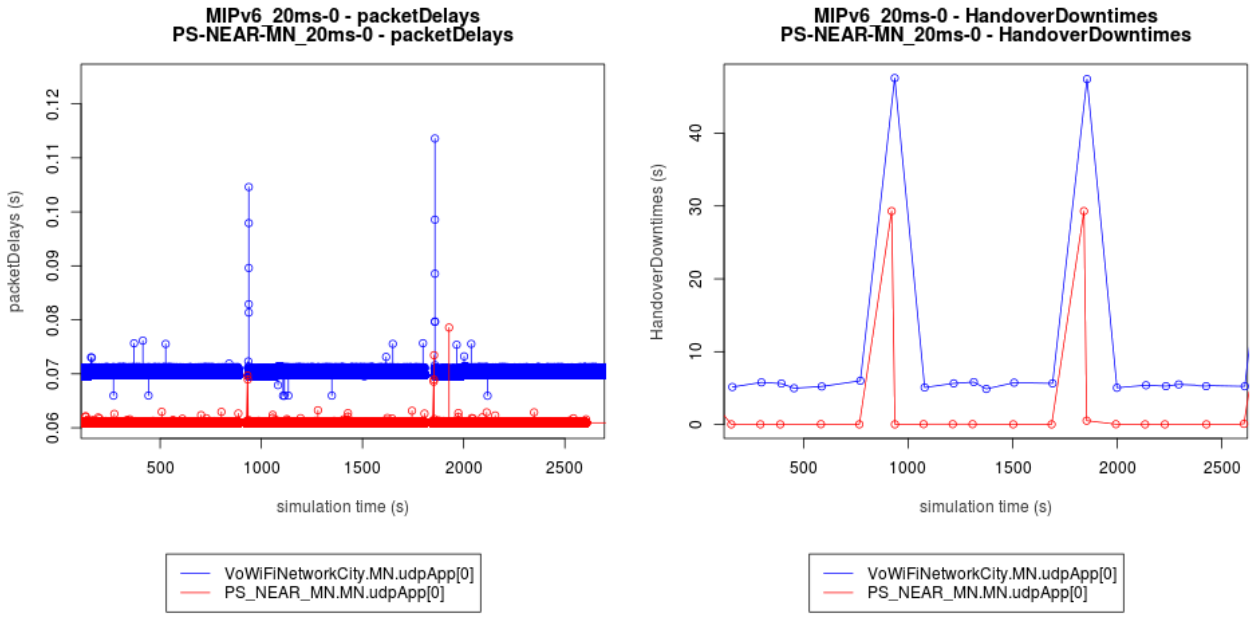

Fig. 41 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra ABPS[ProxyServer-near-MN] e MIPv6 (sendInterval = 20ms)

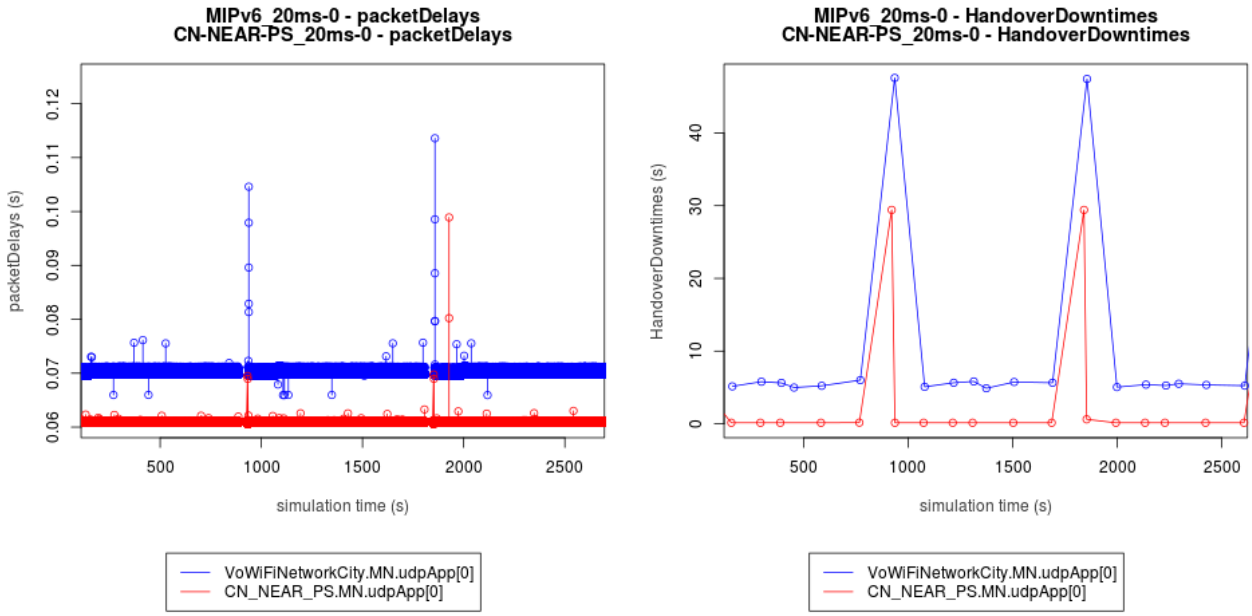

Fig. 42 – Latenza e intervalli di indisponibilità per la rete VoWiFiNetworkCity – confronto tra ABPS[CN-near-ProxyServer] e MIPv6 (sendInterval = 20ms)

Dal confronto emerge chiaramente come le performance di ABPS superino quelle di MIPv6, per tutte e tre le configurazioni testate:

- oltre che per la latenza media, ABPS presenta valori inferiori anche per quanto riguarda l'entità dei picchi massimi di latenza, in corrispondenza delle riconfigurazioni avvenute in seguito ad un handover;
- gli intervalli di indisponibilità di ABPS hanno una durata notevolmente inferiore, sia per quanto riguarda il funzionamento del protocollo in condizioni standard, sia in risposta ai buchi di copertura dovuti all'attenuazione del segnale radio da parte degli ostacoli.

Ciò è una conferma del vantaggio dell'architettura Always Best Packet Switching ottenuto dall'utilizzo simultaneo di più interfacce di rete, grazie al quale è possibile rispondere in maniera tempestiva alla caduta della connessione con un access point in seguito all'eccessivo allontanamento del nodo mobile da esso (dovuta dunque al path loss), o all'ingresso in una zona in cui la presenza di ostacoli interrompe la comunicazione (causata invece dalla penetration loss).

6.6 Sviluppi futuri

I test effettuati non vogliono e non devono comunque essere considerati come esaustivi né definitivi: vi è ancora molto che può essere fatto per migliorare i simulatori esaminati, per accrescerne il realismo e correggerne i problemi che ancora non sono stati individuati.

- Per prima cosa, si può ovviamente pensare di sperimentare altre configurazioni: sono presenti numerosi parametri sui quali andare ad agire, nonché la possibilità di strutturare la rete a proprio piacimento in termini di nodi ed interconnessioni tra essi, così come si può cambiare il comportamento che gestisce gli spostamenti del MN e il tipo di comunicazioni che questo instaura con il CN.

- Sempre per quanto riguarda eventuali modifiche alla struttura della rete, si potrebbe considerare l'idea di inserire un maggior numero di nodi mobili all'interno della stessa simulazione, così come eventualmente più correspondent node; questo richiederebbe tuttavia alcune modifiche piuttosto sostanziali alle implementazioni attuali dei simulatori, in quanto diverse procedure e funzionalità danno per ora per scontato che siano presenti un singolo MN e/o un singolo CN all'interno dello scenario.

- Come estensione del punto precedente, si può valutare anche l'opzione di instaurare una comunicazione tra due differenti nodi mobili (che dunque agirebbero anche da CN l'uno per l'altro), piuttosto che soltanto su uno mobile e uno fisso come avviene correntemente.

- Utilizzare metriche aggiuntive per misurare le prestazioni delle architetture di supporto alla mobilità: quelle di latenza nella ricezione dei pacchetti e durata degli intervalli di indisponibilità del MN sono state scelte in quanto particolarmente caratterizzanti del comportamento di un protocollo per quanto riguarda l'ambito della mobilità, ma è ovviamente possibile individuare altre misurazioni con cui valutare i simulatori.

- Tra le eventuali modifiche minori effettuabili sui simulatori, si segnala ad esempio quella di separare le funzionalità di DHCP dall'applicazione voice-over-Wi-Fi utilizzata da ABPS.

- Per fornire maggiore flessibilità alle simulazioni effettuate con MIPv6, si potrebbe lavorare sulla fase di configurazione iniziale del protocollo, che richiede necessariamente che il nodo mobile si trovi originariamente nella propria rete domestica, precludendo la realizzazione di scenari in cui questo cominci invece i propri spostamenti da una foreign network.

- Fra le anomalie riscontrate, ma di cui non si è riusciti ad individuare la causa, si segnala nuovamente il comportamento di MIPv6 in assenza di ostacoli (*vedi capitolo 6.3, Fig. 19*), su cui sarebbe ovviamente opportuno indagare ulteriormente.

- Durante i test effettuati, la scansione dei canali radio avveniva entro un intervallo di tempo dettato dai seguenti parametri:
  ```
  **.wlan*.agent.minChannelTime = 0.15s
  **.wlan*.agent.maxChannelTime = 0.3s
  ```
  Questi sono stati ereditati dalle simulazioni precedentemente effettuate sullo scenario urbano preesistente all'interno del simulatore ABPS; potrebbe essere interessante studiare il comportamento delle architetture in caso tale periodo di scansione venga ridotto, ad esempio secondo la configurazione seguente:
  ```
  **.wlan*.agent.minChannelTime = 0.10s
  **.wlan*.agent.maxChannelTime = 0.12s
  ```

- Dato che la superiorità delle prestazioni offerte dall'architettura ABPS è riconducibile al suo utilizzo di più interfacce di rete, si potrebbe considerare di apportare le modifiche necessarie a sfruttare il multihoming anche nelle simulazioni che fanno uso di MIPv6 e LISP, tenendo conto che queste sarebbero piuttosto sostanziali.

- In conclusione, sarebbe ovviamente ideale continuare ad aggiornare i simulatori di LISP e ABPS perché siano compatibili con l'ultima versione disponibile della libreria INET, in maniera tale da poterli importare ed utilizzare nel medesimo ambiente di OMNeT++; il simulatore MIPv6 è tecnicamente già parte delle release ufficiali, ma come discusso questo non è comunque esente da varie problematiche e mancanze, che ancora non sono state affrontate.

• Fonti bibliografiche e sitografia

- [1] C. Perkins, D. Johnson, J. Arkko, *"Mobility Support in IPv6"*, RFC 6275, July 2011 .
- [2] D. Farinacci, V. Fuller, D. Meyer, D. Lewis, *"The Locator/ID Separation Protocol (LISP)"*, RFC 6830, January 2013.
- [3] V. Ghini, S. Ferretti, F. Panzieri, *"The "Always Best Packet Switching" architecture for SIP-based mobile multimedia services"*, Journal of Systems and Software, vol. 84, no. 11, pp. 1827-1851, 2011.
- [4] F. Z. Yousaf, C. Bauer, C. Wietfeld, *"An accurate and extensible mobile IPv6 (xMIPV6) simulation model for OMNeT++"*, Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops, Article No. 88, 2008.
- [5] D. Klein, M. Hoefling, M. Hartmann, M. Menth, "Integration of LISP and LISP-MN into INET", Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques, March 2012.
- [6] L. Regazzi, "Simulazione del protocollo TCP a ritrasmissione asimmetrica anticipata su WiFi" (tesi di laurea), Dipartimento di Informatica - Scienza e Ingegneria, Università degli Studi di Bologna, March 2014.
- [7] A. Conta, S. Deering, *"Generic Packet Tunneling in IPv6"*, *Section 8. IPv6 Tunnel Error Processing and Reporting*, RFC 2473, December 1998 .
- [8] R. Droms, *"Dynamic Host Configuration Protocol"*, RFC 2131, March 1997.
- [9] "OMNeT++", <https://omnetpp.org>
- [10] "INET Framework", <https://inet.omnetpp.org>