UNIVERSITÀ DEGLI STUDI DI GENOVA

Dipartimento di Informatica e Scienze dell'Informazione

Anno Accademico 2004/2005

TESI DI LAUREA

Applicazione di algoritmi di routing dinamico su reti wireless in ambiente portuale

Tesi svolta presso il Fantuzzi Reggiane Electronic Department (FRED) di Genova

Relatore	Correlatore	Relatore esterno
Prof. G. Dodero	Prof. M. Ancona	Dott. F. Parodi

Candidato: Daniele Venzano

Indice

Elenco delle figure

Introduzione

L'integrazione di servizi informatici all'interno dell'attività portuale è destinata a svolgere un ruolo fondamentale ovunque ci sia la necessità gestire grandi quantità di traffico in maniera rapida ed efficiente. La presenza di una rete basata su TCP/IP che metta in comunicazione tutti i macchinari in movimento in un porto con gli uffici centrali permette di facilitare un gran numero di operazioni, da semplici comunicazioni operative sulle merci da muovere, ad operazioni di diagnostica remota, fino alla trasmissione dati GPS e di telemetria per la guida automatica.

Una rete senza fili ad instradamento dinamico (MANET) si presta molto bene per fornire una copertura flessibile e ad ampio raggio in un ambiente come quello portuale in cui siano presenti molte fonti di interferenza ed in cui alcuni nodi siano in costante movimento. Inoltre la soluzione MANET offre grandi vantaggi anche dal punto di vista economico, dato che non è necessario posare cavi e fibra ottica nei piazzali di lavoro causando rallentamenti nell'attività portuale ed un aggravio dei costi di installazione.

In questa tesi si è sviluppato un sistema hardware e software completo, di facile installazione, che consente di costruire una rete MANET disponendo i nodi in posizioni sia fisse che mobili. Il routing dinamico è realizzato grazie al protocollo OLSR, di cui sono state studiate, anche sul campo, le caratteristiche di funzionamento. Per completare il sistema è stata sviluppata un'interfaccia di gestione remota via web. L'hardware utilizzato è stato testato ed assemblato in modo da resistere agli estremi di temperatura ed umidità tipici di un ambiente portuale. Il sistema, sviluppato presso il Fantuzzi Reggiane Electronic Department di Genova, è stato destinato fin dall'inizio della sua progettazione, ad essere installato preso l'Interporto Campano vicino a Nola (NA) e verrà inserito tra i prodotti commercializzati da Fantuzzi Reggiane col nome di MeshAP.

La distribuzione dei capitoli nel seguito della tesi riflette in maniera abbastanza precisa lo svolgimento temporale del lavoro. Nella prima parte vengono presentati i risultati della fase di ricerca della tecnologie esistenti, sia dal punto di vista delle reti senza fili, che da quello degli algoritmi di routing dinamico e delle loro implementazioni. La seconda parte esamina l'hardware che è stato utilizzato ed i test effettuati per assicurarsi del suo funzionamento in condizioni difficili. Poi viene descritto il software svilup-

pato appositamente per l'integrazione del sistema all'interno del progetto di Nola e che verrà poi riutilizzato per le successive versioni del prodotto. Infine viene esposta la struttura dell'interporto di Nola e come si prevede che il MeshAP verrà in esso integrato.

Parte I

Valutazione dello stato dell'arte

Capitolo 1

Reti mobili Ad-Hoc

In questo capitolo si espongono le tecnologie che stanno alla base del lavoro svolto nel resto della tesi. La trattazione riguarderà soltanto i livelli del modello ISO/OSI[15] dall'uno al tre, inoltre si intendono conosciute nozioni di base sul funzionamento di reti Ethernet (IEEE 802.3[4]) e sulla suite di protocolli TCP/IP[18, 19].

1.1 Livelli del modello ISO/OSI

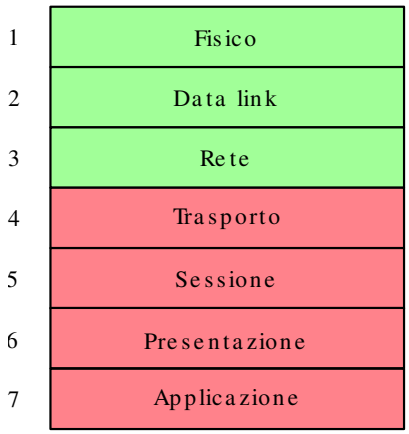

Figura 1.1: Schema del modello ISO/OSI, in verde i livelli che devono essere definiti in una rete senza fili con routing dinamico, in rosso quelli che devono rimanere indipendenti dalla particolare implementazione della rete sottostante.

Una rete senza fili con routing dinamico basata sullo standard 802.11 deve implementare i seguenti strati del modello ISO/OSI:

- Strato fisico e data link. Nel nostro caso si è scelto di utilizzare la suite

802.11 per la sua ampia diffusione ed il basso costo. Altre tecnologie alternative forniscono direttamente capacità di routing a livello data link e forniscono ai livelli superiori una visione trasparente della rete.

- Strato di rete. È necessario tener conto della mobilità dei nodi nel momento in cui deve decidere l'instradamento dei pacchetti. I diversi protocolli di routing dinamico presi in considerazione non modificano direttamente il protocollo IP, ma aggiornano le tabelle di instradamento statico su ogni nodo utilizzando una serie di informazioni che vengono scambiate mediante broadcast tra tutti i partecipanti.

L'ipotesi di lavoro è che al di sopra del livello IP (livello 3 ISO/OSI) non ci sia alcuna conoscenza degli strati inferiori. In realtà il routing dinamico non è perfetto e sono molto probabili brevi periodi di disconnessione per i singoli nodi durante il normale funzionamento della rete.
Il software spesso non è robusto rispetto a questo tipo di errori perché si tratta di eventi molto rari nelle reti fisse e possono essere gestiti manualmente. Nel caso preso in considerazione in questa tesi, al contrario, tutto il software deve essere capace di lavorare in maniera non supervisionata, dato che i singoli nodi verranno disposti in posizioni difficilmente accessibili e che ogni periodo di mancato funzionamento provocherebbe gravi mancanze alla funzionalità dell'interporto.

1.2 Reti senza fili

Le reti senza fili stanno assumendo un'importanza sempre più ampia in tutti i settori di applicazione delle reti di trasmissione dati. La loro grande flessibilità consente una veloce installazione ed un funzionamento immediato, con meno possibilità di guasti e costi minori in fase di installazione e manutenzione. L'eliminazione dei cavi per la trasmissione dati è di estrema importanza, specialmente in quelle applicazioni in cui non sono possibili estesi lavori di muratura o scavo, come edifici protetti da vincoli storici o architettonici[11].
A seconda della tecnologia e del tipo di antenna usata è possibile fornire una copertura generalizzata ad un'area, utile ad esempio per estendere l'accesso a Internet in una zona in cui non sia possibile aggiungere ulteriori cavi, oppure ottenere un collegamento punto a punto su lunghe distanze, ad esempio per collegare direttamente due edifici tramite antenne poste sui tetti. D'altra parte le reti senza fili sono spesso soggette a disturbi che ne limitano la larghezza di banda e il raggio di comunicazione. Le ultime tecnologie commerciali basate su microonde offrono velocità teoriche di circa 100Mbs, ma in realtà offrono quelle velocità solo in condizioni ideali, attestandosi spesso a circa un quarto di ciò che viene promesso a causa di interferenze di vario tipo. Inoltre coprire distanze superiori al chilometro richiede costosi ampli-

ficatori e bande di frequenza apposite a causa dei limiti di legge in vigore sulle bande senza licenza (come quelle usate dal sistema WiFi).

Esistono sistemi senza fili basati sulla luce infrarossa anziché sulle onde radio, sotto forma di trasmettitori e ricevitori LASER che permettono di comunicare su lunghe distanze ed in maniera sicura grazie alla difficoltà di intercettazione. Per contro sistemi di questo tipo sono adatti solo a collegamenti punto a punto e possono essere difficoltosi da installare su lunghe distanze a causa della precisione necessaria per l'allineamento trasmettitore/ricevitore. In [16] c'è uno studio su come sia possibile collegare diversi villaggi indiani a basso costo, in uno dei primi esperimenti è stato possibile coprire una distanza di 20 chilometri con una telecamera ed un puntatore laser, essendo limitati sulla banda passante solo dal numero di quadri al secondo che la telecamera era in grado di registrare. Sono allo studio anche sistemi per la comunicazione terra $\leftrightarrow$ satellite e satellite $\leftrightarrow$ satellite[25].
Lo standard 802.11 prevede una modalità di funzionamento ad infrarossi, come descritto nella sezione 1.3.

1.3 Reti WiFi (IEEE 802.11)

Lo standard IEEE 802.11[3] descrive i livelli fisico e di accesso al mezzo per la comunicazione senza fili su reti locali o metropolitane utilizzando bande di frequenza dello spettro elettromagnetico non regolamentate. La maggior parte dei dispositivi utilizza la banda dei 2.5Ghz, ma gli ultimi dispositivi possono anche utilizzare la banda non regolamentata a 5 Ghz che consente di ottenere trasmissioni meno disturbate da altri dispositivi come forni a microonde, telefoni cordless e telecomandi.

La prima versione dello standard, risalente al 1999, prevede velocità di 1 o 2 Mbit usando due tecniche complementari per il livello fisico: *Direct Sequence Spread Spectrum* (DSSS) e *Frequency Hopping Spread Spectrum* (FHSS).
Revisioni successive dello standard hanno portato ad un aumento della banda fino a 54Mbit (802.11g), sfruttando l'*Orthogonal Frequency Division Multiplexing* (OFDM) che consente una maggiore velocità di trasmissione, a scapito della resistenza ai disturbi e quindi del raggio effettivo di comunicazione.

Una terza modalità, non compatibile con le prime due, è basata sullo spettro di frequenze tra gli 850nm e i 950nm (luce infrarossa). Questo metodo prevede la messa in opera di comunicazioni con una distanza massima di 20 metri sfruttando anche la riflessione della luce infrarossa, in modo da evitare il requisito che i dispositivi siano allineati[1]. Il corto raggio di comu-

[1]Come invece è necessario per il protocollo IRdA, molto usato su palmari e telefoni cellulari

nicazione e l'impossibilità di usare la rete IR all'aperto non hanno permesso una diffusione di questa tecnologia.

A livello MAC sono previste due modalità di funzionamento *Infrastructure* ed *Ad-Hoc*, descritte rispettivamente nelle sezioni 1.3.2 e 1.3.3.

I dispositivi che implementano gli standard 802.11 possono ottenere la certificazione WiFi (Wireless Fidelity) che garantisce l'interoperabilità di dispositivi di produttori differenti. L'acronimo WiFi è ormai entrato nel gergo comune come nome per indicare tutti i dispositivi ad onde radio che usano la famiglia di protocolli 802.11.
Nell'industria sono molto diffusi ed ancora molto richiesti dispositivi compatibili con 802.11b, aventi una velocità massima di 11Mbit ed una buona resistenza ai disturbi. Inoltre le caratteristiche di propagazione sono ben conosciute e la gamma di dispositivi esistenti in commercio copre le esigenze di mantenere comunicazioni in ambienti difficili (temperatura, umidità, vibrazioni).
Al contrario, le caratteristiche dello standard 802.11g non sono ben conosciute e molto spesso il guadagno di velocità non è necessario per le applicazioni industriali, anzi il ridotto raggio di comunicazioni è spesso controproducente. Questo, unito al fatto che la comunicazione a 5Ghz richiede antenne differenti e spesso più costose (perché meno diffuse), provoca una migrazione molto lenta verso l'802.11g.

1.3.1 Sicurezza

Le trasmissioni a mezzo onde radio sono inerentemente insicure, dato che sono molto facili da intercettare con mezzi semplici e poco costosi. Per questo motivo, quando è necessario trasmettere informazioni riservate via radio, si utilizzano tecniche crittografiche per rendere molto più difficile la lettura dei messaggi. Durante la seconda guerra mondiale queste tecniche hanno avuto uno sviluppo formidabile, in parallelo alla crittoanalisi, cioè alle tecniche di decodifica a partire dal solo testo cifrato.
Lo standard 802.11 prevede sia trasmissioni in chiaro che trasmissioni criptate tramite una cifra RC4 con una chiave a 64 o 128 bit che, nello schema più semplice (e più diffuso) deve essere condivisa tra tutti gli utenti. Questo tipo di trasmissioni criptate, chiamato *Wireless Equivalent Privacy* (WEP) offre, in realtà, un livello di sicurezza molto basso, adatto solo a comunicazioni non sensibili. Esistono infatti un certo numero di attacchi diretti all'implementazione WEP o direttamente all'algoritmo RC4 che consentono di ricavare la chiave di codifica in seguito alla ricezione ed esame di qualche milione di pacchetti[2].[13]

In attesa che il nuovo standard IEEE 802.11i venga completato è stato implementato e diffuso uno schema basato su WEP, ma con le miglio-

[2]Per quanto possano sembrare molti, in una situazione di traffico intenso, l'attacco può essere completato in qualche ora

rie necessarie a coprire i problemi di sicurezza. Il *Wi-Fi Protected Access* (WPA)[1] usa uno schema di chiavi dinamiche, possibilmente differenti per ogni utente del sistema, per eliminare la possibilità che la chiave venga recuperata attraverso un confronto statistico del traffico di rete, offrendo in più la possibilità di autenticare gli utenti, cosa non prevista nel WEP.

1.3.2 Modalità *Infrastructure*

In questa modalità ogni area deve essere coperta da un Access Point (AP), al quale tutti i nodi che vogliono usare la rete devono riferirsi per coordinare le trasmissioni. L'AP publicizza la sua presenza inviando ad intervalli regolari dei messaggi (Beacon Frame) che contengono i parametri necessari al collegamento. Il nodo mantiene un elenco di tutti gli AP raggiungibili e sceglie quello che ha il segnale radio più potente, inviandogli un frame di associazione.

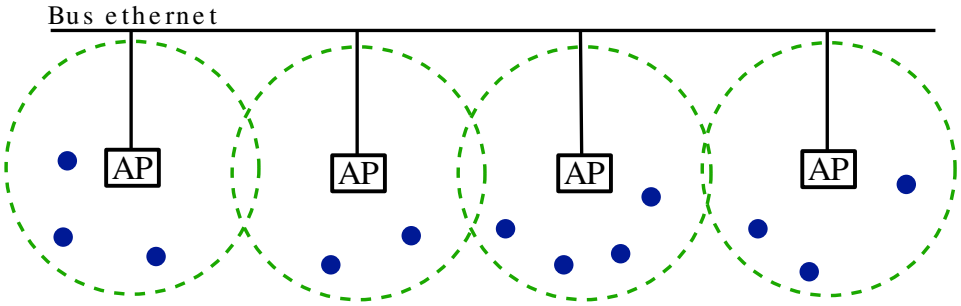

Figura 1.2: Rete Wifi in modalità infrastructure. Ogni Access Point fornisce l'accesso alla rete a tutti i nodi presenti sotto la sua copertura radio.

Lo standard non prevede che un nodo possa passare da un AP ad un altro in maniera dinamica, lasciando i dettagli di una simile procedura ai singoli produttori, che infatti hanno prodotto una serie di protocolli proprietari incompatibili tra di loro. Attualmente il Wireless Distrbution System (WDS) sta cercando di porre rimedio alla situazione, fornendo uno standard per il bridging dei pacchetti tra AP. Nella figura 1.2 è visibile la struttura di una rete infrastructure con quattro AP collegati da una rete ethernet che consente di trasportare i pacchetti da un nodo all'altro anche se questi sono associati a due AP diversi. Questa rete può essere facilmente collegata ad Internet, fornendo l'accesso a tutti i nodi collegati.

1.3.3 Modalità *Ad-Hoc*

La modalità Ad-Hoc è pensata per permettere un metodo di scambio dati molto veloce da impostare, ed a corto raggio, tra un numero limitato di nodi. A questo livello, infatti, è sufficiente accordarsi su un canale radio ed

un nome della rete (SSID) per iniziare a comunicare. Volendo utilizzare il protocollo IP è anche necessario scambiarsi i rispettivi indirizzi. Come si può vedere nella figura 1.3 i nodi possono comunicare a coppie, ma lo standard 802.11 non prevede nessuna forma di routing dei pacchetti, così che non è possibile utilizzare un nodo intermedio come ponte senza un ulteriore strato software.

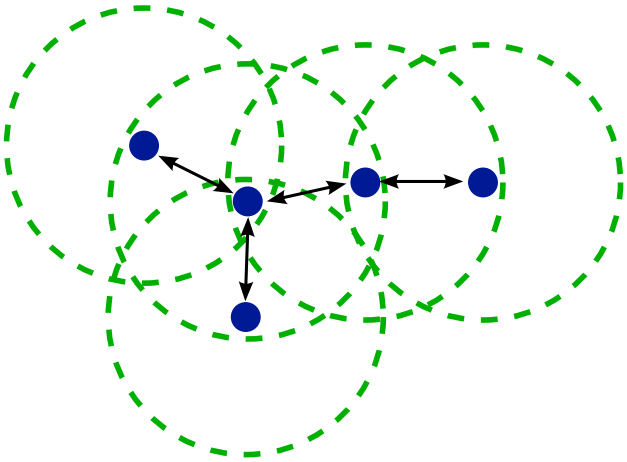

Figura 1.3: Rete WiFi in modalità Ad-Hoc. I dispositivi possono comunicare a coppie, non appena si trovano entro il raggio di comunicazione della radio.

1.4 Routing dinamico

I protocolli di routing dinamico consentono ai nodi di una rete in modalità Ad-Hoc di funzionare allo stesso tempo come punti d'accesso e come router, permettendo di costruire una rete molto estesa in breve tempo, senza le complicazioni dovute alla stesura di cavi e all'impostazione di parametri di connessione e tabelle di routing. Di seguito, nella sezione 2.3, vengono esaminati diversi protocolli, secondo i criteri stabiliti nel capitolo 2.1, portando alla scelta di OLSR per lo sviluppo della rete dell'interporto di Nola.

Una rete del genere presenta un'utilità straordinaria negli scenari più disparati, dal coordinamento dei mezzi di soccorso in zone disastrate dove non esistano più le infrastrutture normali di comunicazione (cavi interrati, reti cellulari), alla possibilità di offrire servizi Internet in vaste aree a basso costo, potendo coprire eventuali 'angoli bui' con ulteriori dispositivi senza che sia necessaria una riconfigurazione dei nodi esistenti.

Portando all'estremo il concetto di rete dinamica si arriva alle reti di sensori o anche *dust networks*, sono reti formate da migliaia di nodi, che possono essere sparsi su un'ampia zona da mezzi aerei o navali. Ogni nodo

è molto semplice, poco costoso e raccoglie dati che provvede a distribuire ai nodi adiacenti, consentendo di monitorare ampie zone di territorio a basso costo ed in tutta sicurezza per gli operatori [14]. Questo tipo di reti ha, attualmente, applicazioni militari, anche se sta iniziando un certo interesse per il monitoraggio di grandi installazioni come raffinerie ed oleodotti. Esiste anche uno standard IEEE, l'802.15.4 sulle reti di sensori ed un consorzio, sulla falsariga del WiFi, chiamato ZigBee che ha esteso lo standard aggiungendo alcuni livelli superiori per fornire routing e sicurezza, garantendo allo stesso tempo bassissimi consumi energetici [5] [8].

Capitolo 2

Algoritmi di routing dinamico

In preparazione al lavoro successivo è stata fatta una classificazione degli algoritmi esistenti per il routing dinamico, stabilendo una serie di criteri che permettano un confronto oggettivo.
Per alcuni algoritmi esistono diverse implementazioni, in fase di valutazione queste sono state prese in considerazione, valutandone lo stato di avanzamento e l'attività del gruppo di sviluppo.

2.1 Classificazione

Gli algoritmi di routing possono essere classificati secondo diversi parametri e modalità di funzionamento. La caratteristica principale è il metodo utilizzato per creare e mantenere le tavole di routing.
Esistono due approcci, spiegati di seguito:

proattivi : viene mantenuta la topologia di tutta la rete, che è aggiornata ad intervalli fissati di pochi secondi. Tutti i nodi sanno come raggiungere ogni altro nodo ad ogni istante.

reattivi : costruiscono un cammino ogni volta che ce n'è bisogno e lo mantengono in una cache finché è valido (approccio *lazy*, pigro). Hanno un certo ritardo ogni volta che si deve mandare un pacchetto ad una macchina mai raggiunta prima.

Mentre un approccio proattivo consente una comunicazione senza ritardi iniziali, esso richiede un costante consumo di banda e di risorse su ogni nodo per mantenere le informazioni sui cammini, inoltre un algoritmo proattivo è adatto a quelle situazioni in cui potenzialmente tutti i nodi vogliono comunicare tra loro e non ci sono dei cammini preferenziali.

Al contrario un approccio reattivo stabilisce un cammino solo quando ce n'è effettivamente bisogno, limitando l'uso di risorse al minimo indispensabile. Per quanto i cammini costruiti vengano mantenuti in una cache per un certo tempo, è molto probabile che avvengano dei ritardi ogni volta che una nuova connessione deve venir stabilita.

Esiste anche una classificazione basata su di un'eventuale disposizione gerarchica dei nodi. La maggior parte degli algoritmi sono *flat* (piatti), e la ricerca effettuata ha permesso di trovare un solo algoritmo gerarchico, *Zoned Routing Protocol* - ZRP, di cui però non è stato possibile trovare un'implementazione.

flat : tutti i nodi partecipano allo stesso modo al routing.

cluster based : i nodi sono divisi in gruppi, ognuno con un nodo-capo, che conosce la topologia del suo gruppo e attraverso cui passano tutte le comunicazioni da e per i nodi da lui gestiti (es. ZRP - Zoned Routing Protocol).

Imporre una gerarchia ha senso solo se il numero di nodi è abbastanza grande da raggiungere i limiti delle tabelle di routing nel kernel o se c'è una spiccata differenza tra i nodi, con alcuni di essi posti in posizioni strategiche o con particolari configurazioni hardware.

Quando un pacchetto deve essere instradato ed esistono più route attraverso cui potrebbe passare per arrivare a destinazione viene presa in considerazione la *metrica* di ogni cammino e viene scelto quello con il valore minore. Questa metrica è un valore numerico che rappresenta la distanza tra due nodi. Solo in casi molto particolari questa distanza è effettivamente una distanza fisica, di solito è semplicemente il numero di salti necessari per arrivare a destinazione (2.1).

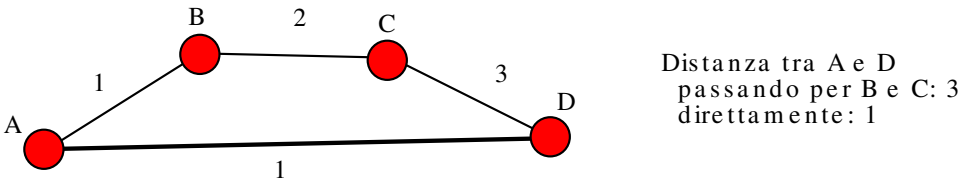

Figura 2.1: Distanza calcolata in base al numero di salti. Il cammino migliore è quello diretto tra A e D, segnato con una linea più spessa.

In casi particolari possono venir utilizzati altri parametri per calcolare la distanza tra i nodi, in particolare su reti senza fili può essere utile tenere in considerazione i seguenti fattori:

- banda disponibile (fig. 2.2)
- energia necessaria per la comunicazione

- rapporto segnale/rumore
- stabilità di associazione
- informazioni sul posizionamento fisico dei nodi (GPS)[27]

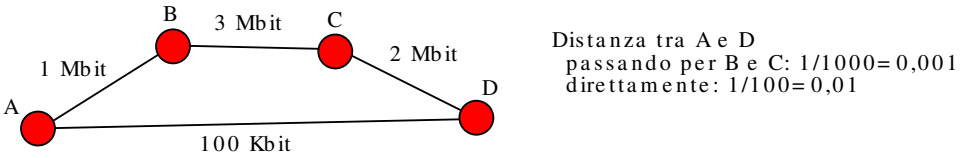

Figura 2.2: Distanza calcolata in base alla banda disponibile tra i nodi. Il cammino migliore è quello che passa attraverso i nodi B e C.

Le metriche speciali, per quanto molto utili, spesso sono difficilmente utilizzabili a causa della difficoltà di procurarsi informazioni sufficientemente accurate con hardware facilmente disponibile in commercio. Inoltre differenti sistemi operativi offrono interfacce molto diverse per l'accesso a dati di basso livello, causando difficoltà nel mantenere implementazioni multipiattaforma.

Un ultimo importante fattore che riguarda strettamente l'implementazione di un algoritmo è se questa è realizzata in *kernel space* oppure in *user space*. Entrambe le soluzioni hanno dei vantaggi e degli svantaggi, elencati di seguito:

kernel :

Pro

- prestazioni più alte dato che i pacchetti non devono essere copiati nello spazio di memoria del programma utente e poi ricopiati indietro nello spazio kernel per la ritrasmissione.
- migliore integrazione con il resto del sistema

Contro

- maggiore difficoltà di debug, bisogna conoscere la struttura interna del kernel, e sono necessarie competenze specifiche, anche di linguaggio assembler per usare un *kernel debugger*.
- necessaria revisione ad ogni nuova versione del kernel, in particolare per Linux, in cui non viene garantita la stabilità delle ABI (Application Binary Interface)/API (Application Programming Interface) tra una versione e l'altra.

userspace :

Pro

- sviluppo e debug semplificato, possono essere usati linguaggi ad alto livello e sono disponibili tutte le librerie necessarie.
- indipendenza dalla versione del kernel, dato che le funzioni di accesso al sottosistema sono in gran parte mascherate dalle librerie C (per Linux) o dalle API ufficiali (per Windows).

Contro

- prestazioni inferiori, ogni pacchetto ricevuto deve essere copiato all'applicazione utente, che dopo aver preso le decisioni di routing, deve ricopiarlo in un buffer lato kernel per il passaggio verso la scheda di rete.

Esistono vie di mezzo, con una parte integrata nel kernel e una parte del lavoro realizzata in *user space*. In queste soluzioni si cerca di ridurre l'impatto sulle prestazioni in cambio di una maggiore complessità del codice da scrivere e mantenere.

2.2 Determinazione dei criteri di valutazione

Per poter decidere quali, tra gli algoritmi esistenti, siano più adatti al campo di interesse, si è stabilita una lista di criteri utile anche a valutare lo stato di maturità della particolare implementazione. Questa lista è descritta sinteticamente di seguito, come riferimento e senza un particolare ordine. A partire dalla sezione 2.2.1 vengono spiegati i criteri scelti e le motivazioni che stanno dietro ad essi.

1. Licenza dell'implementazione che permetta modifiche al codice sorgente
2. Limite massimo sul numero di nodi in rete contemporaneamente di almeno 40
3. Esistenza e maturità dell'implementazione
4. Implementazione su più sistemi operativi
5. Esistenza di test in ambienti reali
6. Sicurezza

2.2.1 Licenza

Si è stabilita la necessità di una licenza *open source* o equivalente che permetta la modifica del codice sorgente dell'implementazione dell'algoritmo. Questo è necessario sia per poter effettuare direttamente eventuali modifiche necessarie all'integrazione nell'attuale sistema di Fantuzzi Reggiane, sia

per garantire il mantenimento del codice in futuro se gli sviluppatori attuali cessassero il loro contributo.

Tutte le implementazioni di cui è disponibile su Internet il codice sorgente sono coperte da licenze GPL o BSD. Questo requisito ha escluso diverse soluzioni proprietarie, spesso coperte da brevetti e descritte in maniera molto vaga sui relativi siti web.

2.2.2 Dimensioni della rete

Dovendo garantire l'espandibilità della rete in momenti successivi alla prima installazione e l'utilizzo del sistema di routing dinamico in reti di dimensione anche molto diversa è necessario che l'algoritmo non preveda limiti sul numero di nodi che possono partecipare contemporaneamente alla rete.

Tra gli algoritmi presi in considerazione solo uno (LUNAR[7]) consiglia l'utilizzo in reti con meno di dieci nodi. Tutti gli altri non pongono limiti o tali limiti sono abbastanza grandi da poter essere ignorati.

2.2.3 Maturità dell'implementazione

Visti i tempi molto lunghi necessari allo sviluppo da zero di un'implementazione ragionevolmente completa e utilizzabile in produzione, si è deciso di prendere in considerazione solo algoritmi per cui ci fosse già del codice disponibile. Inoltre si sono preferite implementazioni con un gruppo di sviluppo attivo.

2.2.4 Sistemi operativi

I nodi che faranno parte della rete costruita durante lo svolgimento di questa tesi utilizzano GNU/Linux come sistema operativo, con kernel 2.4. Quindi è obbligatoria l'esistenza di un'implementazione per Linux. Inoltre sono preferiti algoritmi con implementazioni anche per altri sistemi operativi e piattaforme hardware, in particolare per Microsoft Windows e per palmari (Pocket PC, Linux Familiar). Questa preferenza è collegata al problema dei nodi esterni, vedi la sezione 2.2.7.

Molti algoritmi sono stati esclusi a causa di questo requisito perché esistono solo sotto forma di simulazioni per NS2 o per altri simulatori. I rimanenti hanno tutti un'implementazione per Linux o per un sistema operativo della famiglia BSD. E' invece molto difficile trovare software di questo tipo per Windows o per Pocket PC.

2.2.5 Test e prove

Normalmente chi implementa un algoritmo di instradamento dinamico effettua dei test su un simulatore. Questi test, per quanto permettano di verificare semplicemente se l'algoritmo funziona come inteso, sono ben lontani dal

riprodurre tutti i problemi che possono venir fuori durante il funzionamento in un ambiente reale. Per questo motivo si sono cercati i risultati di prove con più di 3 nodi, con collegamenti tra i nodi su protocollo 802.11 e i nodi posizionati su processori differenti (portatili, palmari e macchine fisse).

Questo requisito è servito a stabilire una preferenza per algoritmi di cui siano documentate delle prove. Purtroppo è molto raro trovare informazioni in merito in quanto, molto spesso, la fase di test si compie interamente su simulatori.

2.2.6 Sicurezza

E' importante che l'algoritmo possa funzionare normalmente anche con la crittografia WEP attivata, inoltre sono da preferire implementazioni che permettano di definire delle regole di controllo d'accesso in modo che nodi non autorizzati non possano entrare a far parte della rete.

2.2.7 Nodi esterni

Possibilità di collegare sistemi WiFi alla rete senza ulteriore software[1]

Questo requisito è molto restrittivo, nessuno dei sistemi presi in considerazione tiene presente un'esigenza del genere. In seguito ad uno scambio di email con l'autore di `olsrd` si è arrivati alla conclusione che è possibile utilizzare un solo nodo della MANET come router verso una rete senza multihop. Non è possibile utilizzare più nodi come gateway a causa di grossi problemi nel redirigere le connessioni esistenti e nella difficoltà, da parte del nodo senza capacità multihop, di stabilire una *default route*.

2.3 Algoritmi

Due algoritmi sono subito risultati molto sviluppati, AODV e OLSR. Nel corso del tempo si sono susseguite diverse implementazioni, sempre più complesse e sempre meno a livello di prototipo. Attualmente questi due algoritmi sono stati formalizzati in due RFC per stabilire un punto fermo a cui tutte le implementazioni possano convergere e rendersi interoperabili. Nel seguito vengono anche descritti altri algoritmi che sono stati presi in considerazione. A seguito di questa valutazione è stato deciso di utilizzare l'algoritmo OLSR, nell'implementazione sviluppata da Andreas Tønnesen presso l'università di Oslo, che viene descritta in maniera estesa nel capitolo 3.

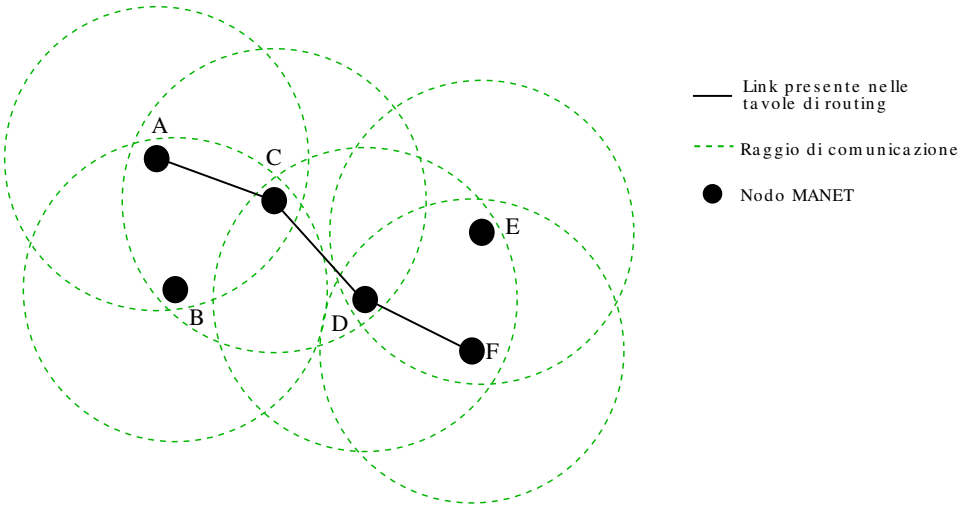

Figura 2.3: Link presenti nelle tavole di routing per un cammino dal nodo A al nodo F

2.3.1 AODV - Ad-hoc On-demand Distance Vector (RFC 3561)

L'algoritmo è di tipo reattivo e costruisce il cammino ogni volta che un pacchetto deve essere inviato ad una destinazione nuova o irragiungibile. Questo provoca un ritardo iniziale nella comunicazione, per il tempo necessario ad ottenere dal resto della rete una route valida. Una volta che un cammino sia stato stabilito viene mantenuto nelle cache di tutti i nodi intermedi con l'informazione di quale sia il nodo successivo da contattare. Quando uno di questi collegamenti punto a punto viene a mancare, l'ultimo nodo raggiungibile dalla sorgente rispedisce indietro un messaggio di servizio che causa la rimozione dalle cache del cammino non più valido e innesca una nuova ricerca del cammino a partire dalla sorgente.

Esistono due implementazioni ad un buon livello di sviluppo, ma dato che per l'algoritmo è fondamentale sapere quando un pacchetto viene scartato a causa di un cammino non più valido, entrambe richiedono che venga caricato un modulo per il kernel[2].
In alternativa è possibile effettuare le operazioni di routing in userspace, riscrivendo gli indirizzi di partenza/destinazione ad ogni salto, provocando, però, almeno due copie in memoria per ogni pacchetto (kernel → user e user

[1]Questi sistemi extra dovrebbero poter sfruttare il resto della rete come dominio di instradamento, ma non farne parte essi stessi.

[2]Tutti i sistemi operativi scartano i pacchetti per i quali non esiste un cammino valido e se si vuole intervenire su questo meccanismo è necessario aggiungere del codice all'interno dello stack di rete

→ kernel), con pesanti ripercussioni sulle prestazioni.
Esiste anche un'implementazione per sistemi Windows, ma non è stato possibile verificare l'interoperabilità tra questa e le versioni Unix/Linux.

Implementazioni:

NIST[26] : modulo per kernel 2.4, non c'è traccia di test reali, solo simulazioni. Codice sorgente rilasciato nel pubblico dominio, senza licenze.

UU (Upsala University)[12] : implementazione solo in parte user space, usa netfilter per farsi passare i pacchetti, testato realmente con al più 5 nodi e 4 hop, parecchie simulazioni. L'ultima versione ha un modulo da caricare nel kernel per ridurre l'impatto della copia dei pacchetti tra user space e kernel space. Licenza open source (GPL).

Pro:

- Basso utilizzo di banda per il mantenimento dei cammini, lasciando spazio alla trasmissione dati
- Efficiente utilizzo delle risorse di sistema (implementazione in-kernel)

Contro:

- Ritardo significativo (fino a qualche secondo) ad ogni inizio di nuova comunicazione verso nodi mai visti prima
- Implementazione in-kernel, gestione più complessa e delicata di nuove versioni
- Uso di tempo di CPU per copie in memoria. Su sistemi embedded questo è un fattore molto importante
- Test molto limitati

2.3.2 MIT SrcRR

Basato su DSR, è usato in una rete metropolitana nel progetto Roofnet[9] del Massachussets Institute of Technology (MIT). E' un algoritmo proattivo ed usa una metrica basata sulla qualità del collegamento, che stima il numero di volte che un pacchetto deve essere ritrasmesso prima che sia ricevuto correttamente. La metrica di un cammino di più salti corrisponde alla somma delle metriche di ogni salto, consentendo di penalizzare cammini lunghi e cammini con alte perdite di pacchetti.

Il progetto è molto attivo e in uso correntemente, ma è progettato per nodi fissi o con scarsa mobilità. L'implementazione è solo per GNU/Linux e viene fornita anche come un LiveCD completo di tutto il necessario per facilitare l'installazione di nuovi nodi.

2.3.3 LUNAR - Light Underlay Network Ad-hoc Routing

LUNAR[6, 7] è sviluppato dall'università di Upsala ed è un algoritmo ibrido, sia proattivo che reattivo. E' implementato come modulo per il kernel di Linux e non esistono versioni per altri sistemi operativi. E' pensato per essere estremamente semplice da utilizzare, infatti come parte del suo funzionamento si occupa anche di assegnare gli indirizzi IP ai singoli nodi, un compito di solito lasciato all'utente. E' anche pensato per reti molto piccole, al massimo di una decina di nodi, e di piccolo raggio, con una distanza tra i nodi di non più di 50 metri.

2.3.4 Altri algoritmi

DSR-Monarch[10] : Esiste solo un'implementazione per FreeBSD[21], una precedente versione per Linux è stata abbandonata e non è più disponibile su Internet. Aveva il problema, abbastanza fondamentale, di non poter lavorare con connessioni TCP.

TORA : (Temporally Ordered Routing Algorithm) ha bisogno di clock sincroni, si propone di operare bene in ambienti molto dinamici, pare sviluppato dalla Marina degli USA, ma non è stato possibile trovare un'implementazione. Alcuni riferimenti puntano al *Ad-Hoc Networking Research Group* dell'università del Maryland, ma riportano anche gravi problemi di stabilità.

Capitolo 3

Optimized Link State Routing - OLSR

OLSR è l'algoritmo scelto per l'implementazione del sistema di routing dinamico. In questo capitolo si descrivono i motivi che hanno portato a questa scelta, il funzionamento dell'algoritmo e dell'implementazione utilizzata, chiarendo i motivi che hanno portato a preferirla rispetto alle altre disponibili. Per evitare confusioni, nel seguito si indicherà con OLSR l'algoritmo e con `olsrd` la particolare implementazione usata.

3.1 L'algoritmo

OLSR è formalizzato in una RFC[23] proposta dal progetto Hypercomm, dell'INRIA[1]. Attualmente si trova ancora in una fase sperimentale, ma il nucleo del protocollo è ormai ben definito e la discussione rimanente verte su alcune estensioni che sono ancora in fase di studio. L'esistenza di una formalizzazione ufficiale consente alle diverse implementazioni di convergere verso un insieme comune di funzionalità di base e di realizzare delle estensioni aggiuntive mantenendo l'interoperabilità. Questa fase di convergenza sta lentamente avvenendo, anche se al tempo in cui è stata sviluppata questa tesi non tutte le implementazioni esistenti sono interoperabili tra loro.

Negli ultimi due anni si sono tenuti due *OLSR Interop & Workshop*, il primo negli Stati Uniti, il secondo a Parigi, sponsorizzati da diverse grandi aziende, per facilitare il processo di integrazione e dare maggior coesione alla comunità che ruota intorno ad OLSR.

3.1.1 Funzionamento

OLSR è un algoritmo di tipo proattivo, quindi alla partenza ogni nodo collabora con i sui vicini per costruire una tabella di instradamento contenente

[1] *Institut National de Recherche en Informatique*

i cammini verso tutti i nodi esistenti. Dopo questa inizializzazione i nodi si scambiano periodicamente dei messaggi per mantenere aggiornati i propri dati topologici.

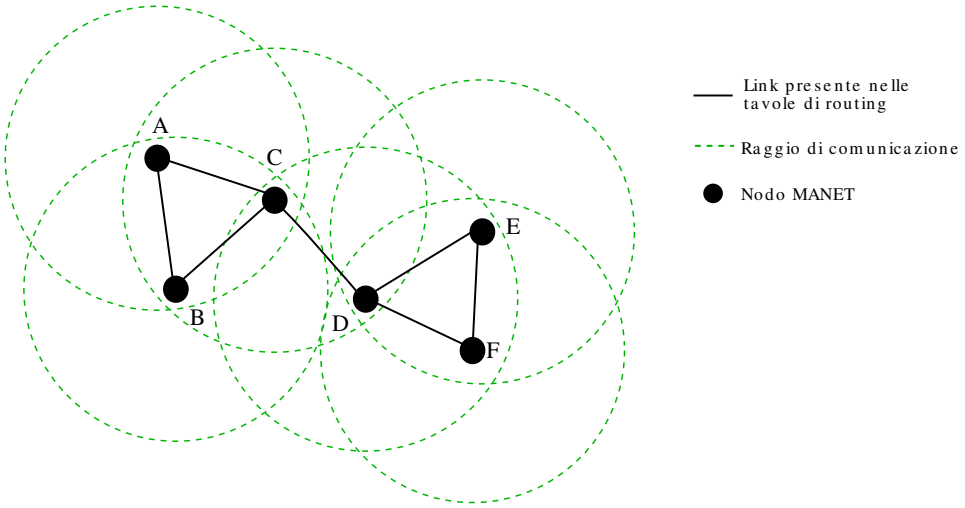

Figura 3.1: OLSR: i nodi conoscono i cammini verso tutti gli altri nodi

Come suggerisce il nome dell'algoritmo, OLSR utilizza il broadcast delle informazioni sullo stato di tutti i collegamenti conosciuti ad ogni nodo per permettere la ricostruzione della topologia di rete. Questo broadcast viene però ottimizzato per limitare al massimo lo spreco di banda utilizzando un sistema chiamato *MultiPoint Relaying* (MPR).

La tecnica MPR nasce dall'osservazione che in una situazione di broadcast non ottimizzato ogni nodo riceve più volte le stesse informazioni causando un notevole spreco di banda e di potenza di calcolo. Questa situazione può essere migliorata scegliendo un particolare sottoinsieme di nodi che possono ritrasmettere le informazioni. Le figure 3.2 e 3.3 mostrano la differenze nel numero di ritrasmissioni.
In OLSR ogni nodo sceglie un sottoinsieme dei suoi vicini simmetrici[2] tale che ogni secondo vicino sia raggiungibile tramite questo sottoinsieme. È stato dimostrato[24] che questa scelta è NP-completa, infatti l'RFC, nella sezione 8.3.1, descrive un semplice algoritmo euristico per calcolare il sottoinsieme MPR.

Il sistema MPR viene usato come metodo di default per la ritrasmissione dei messaggi e fa parte del nucleo di OLSR che deve essere presente in tutte le implementazioni, consentendo a tutti i nodi appartenenti ad una

[2]Tali per cui esiste una comunicazione diretta in entrambi i sensi, cosa non sempre vera in una comunicazione radio.

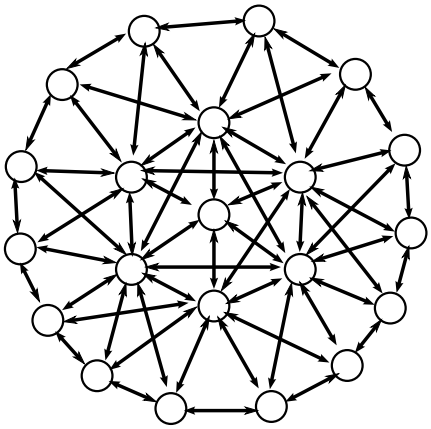

Figura 3.2: Broadcast completo: ogni nodo riceve più volte le informazioni, direttamente ed in seguito a rimbalzi sui nodi vicini.

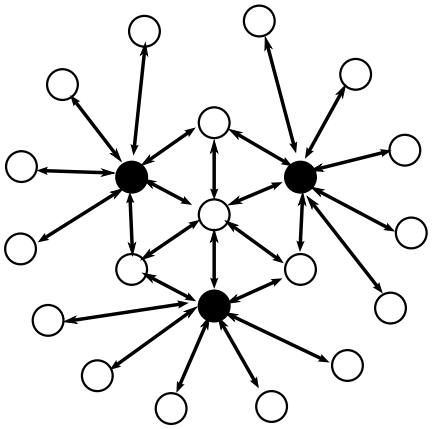

Figura 3.3: Broadcast MPR: il nodo centrale seleziona un certo numero di vicini come MPR e solo questi ultimi effettuano la ritrasmissione. Il numero di comunicazioni è di molto inferiore.

rete di ritrasmettere correttamente i messaggi anche senza comprenderne il contenuto.

OLSR utilizza pacchetti UDP per trasferire le informazioni di controllo, ogni pacchetto può contenere più di un messaggio, proveniente da mittenti differenti, per meglio sfruttare il tempo di trasmissione. Un pacchetto generico è descritto nella tabella 3.1. I messaggi richiesti dalla funzionalità base di OLSR sono:

- HELLO: inviati ad intervalli regolari, compiono le funzioni di rilevamento dei vicini, comunicazione dei nodi MPR e *link sensing*
- TC: servono a comunicare informazioni topologiche dal punto di vista di ogni nodo
- MID: usati dai nodi con più interfacce per dichiararne l'esistenza al resto della rete.

Per evitare che i nodi compiano trasmissioni sincronizzate (e quindi che i pacchetti collidano sulla rete), l'RFC stabilisce che prima di ogni invio un nodo debba aspettare un tempo casuale.

Durante il funzionamento ogni nodo, in base al contenuto dei messaggi che riceve, mantiene un certo numero di tabelle. Tipicamente ad ogni dato memorizzato in queste tabelle è associato un tempo di scadenza, dopo il quale il dato non è più valido e deve essere cancellato. Le tabelle più importanti sono:

Dimensione pacchetto		Numero di sequenza
Tipo del messaggio	Tempo di validità	Dimensione messaggio
Indirizzo del mittente		
Tempo di vita	Numero di salti	Numero di sequenza
Messaggio		
Tipo del messaggio	Tempo di validità	Dimensione messaggio
Indirizzo del mittente		
Tempo di vita	Numero di salti	Numero di sequenza
Messaggio		

Tabella 3.1: Formato di un pacchetto OLSR generico contenente due messaggi. Gli header del protocollo IP non sono mostrati.

- Neighbour set: insieme dei primi vicini, cioè quelli per cui è stato ricevuto un messaggio HELLO
- 2-hop neighbour set: insieme dei nodi pubblicizzati come primi vicini dagli appartenenti al neighbour set. L'intersezione tra questo ed il neighbour set è vuota.
- MPR set: insieme dei nodi che agiscono come MPR per questo nodo
- MPR selector set: insieme dei nodi che hanno scelto questo come MPR
- Topology set: ogni nodo esistente nella rete ha una riga in questa tabella che contiene il suo indirizzo e quello del nodo che lo ha pubblicizzato come vicino

Inoltre ogni nodo mantiene una routing table, in comune con il sistema operativo, in cui vengono riflessi i cambiamenti che avvengono nelle tabelle sopra elencate. Ogni volta che avviene una variazione viene applicato un algoritmo *shortest path* sul grafo indiretto che ha come nodi tutti i nodi della rete conosciuti e come archi i link bidirezionali tra primi vicini, ricostruendo così l'intera topologia, e quindi la tabella di instradamento.

3.2 L'implementazione

Attualmente esistono diverse implementazioni di OLSR, in vari stadi di maturità. INRIA sta sviluppando OOLSR in C++, il Laboratorio di Ricerche Navali degli Stati Uniti sta lavorando a NROLSR, il Communications Research Center in Canada stava (fino al 2003) lavorando a CRCOLSR ed infine il *Laboratoire de Recherche en Informatique* dell'università di Parigi-sud sta attivamente implementando QOLSR, ponendo un particolare accento sul rispetto di criteri di *Quality of Service*. Oltre a queste c'è l'implementazione

più matura, `olsrd`[2], sviluppata in una tesi di dottorato presso l'università di Oslo da Andreas Tønnesen ed ancora in pieno sviluppo.

`olsrd` è stata testata in una rete mista wireless/fissa con circa trenta nodi dall'autore durante la conferenza Wizard of Operating Systems a Berlino nel 2004, è completamente in user space e utilizza dei meccanismi standard per aggiornare le tabelle di routing del kernel. Inoltre ha un sistema di plugin che permette di estenderne le funzionalità senza toccare il codice sorgente dell'applicazione principale. Esistono già diversi plugin implementati che offrono servizi aggiuntivi come la comunicazione dello stato della batteria o l'autenticazione dei nodi nella rete. Un altro punto di forza è che viene distribuito con una licenza *Open Source*, garantendo una vita al progetto anche qualora l'autore originale smettesse di occuparsene.

Attualmente `olsrd` viene utilizzato a Lille, Francia per il progetto Lille Sans Fil[3] che intende offrire una copertura della città (e in previsione dell'intera regione) con una rete senza fili con routing dinamico. Sempre `olsrd` è parte del progetto Freifunk[4] per la diffusione di reti libere in aree di lingua tedesca.

Sono disponibili versioni per Linux, Windows (2000 e XP), Os X e Linux Familiar (per palmari). Questo è importante per permettere l'interoperabilità del sistema sviluppato in questa tesi con sistemi concorrenti di altri produttori[5].

`olsrd` ha alcune caratteristiche che si sono evidenziate durante lo svolgimento di questa tesi: in particolare hanno richiesto uno studio approfondito i parametri utilizzati per decidere quando un collegamento è abbastanza affidabile. Questi risultano dipendere in maniera molto grande dal tipo di rete che si sta sviluppando.

3.2.1 Isteresi

Il metodo più semplice che `olsrd` può usare per determinare la bontà di un link è basato sull'isteresi. Questo metodo non consente di dare un peso al link, semplicemente fornisce un dato binario del tipo utilizzabile/non utilizzabile. Per funzionare ha bisogno di tre parametri, due soglie ed un valore di scala. Quando il valore di un link scende al di sotto della soglia minima, il link viene dato per non attivo, mentre quando sale al di sopra della soglia massima viene dato per attivo. Più le due soglie sono vicine e più il routing è instabile, ma più adattabile a rapidi cambiamenti nella topologia.

Le formule usate per il calcolo del valore di isteresi per ogni link sono due, una applicata 'in salita', quando cioè un messaggio di HELLO viene

[3] http://www.lillesansfil.org

[4] http://www.freifunk.net

[5] Vedi anche il terzo requisito preso in considerazione nella valutazione degli algoritmi, sezione 2.2.7

ricevuto con successo:

$$LinkQuality = (1 - scaling) * LinkQuality + scaling$$

ed una in discesa, quando un messaggio di HELLO viene perso

$$LinkQuality = (1 - scaling) * LinkQuality$$

Si ricorda infatti che i messaggi HELLO vengono generati ad intervalli regolari ed è quindi facile determinare se sono stati persi durante la trasmissione.

3.2.2 Qualità del collegamento

`olsrd` offre anche un altro metodo per determinare la disponibilità di un link basandosi sul numero di pacchetti persi in un certo intervallo di tempo. Questo metodo permette di assegnare un peso ad ogni link a seconda della probabilità che una trasmissione vada a buon fine.

La definizione di OLSR nell'RFC non comprende questa misura di qualità che è ancora in via di formalizzazione.

3.3 I motivi della scelta

La decisione di utilizzare OLSR ed `olsrd` anziché qualcos'altro si è basata soprattutto sul fatto che intorno ad OLSR gravita una comunità molto attiva in grado di offrire supporto per lungo tempo. Al contrario gli altri algoritmi sono sembrati, nel momento di questa scrittura, dormienti, spesso con siti web non aggiornati da diversi anni.

Questo aspetto è stato ritenuto più importante rispetto a fattori tecnici, come il fatto che nell'ambito portuale le comunicazioni avvengono esclusivamente tra due sottoinsiemi fissi di nodi, con quelli intermedi che fungono solo da ripetitori. In questo ambito un algoritmo reattivo come AODV avrebbe potuto essere più indicato, in quanto non mantiene informazioni di raggiungibilità che non verranno mai usate.

D'altra parte si ipotizza di utilizzare le reti basate su `olsr` anche per trasportare la voce, sostituendo le radioline attualmente in uso nei porti. Un'applicazione del genere farebbe un uso molto più esteso della rete e di conseguenza sarebbe di molto avvantaggiata dalla proattività di OLSR.

Segue un elenco di pro e contro tenuti in considerazione al momento della scelta di usare OLSR ed `olsrd`.
Pro:

- Gruppo di sviluppo molto attivo
- Test realizzati con successo su ampia scala
- Implementazione totalmente in spazio utente

- Espandibilità con plugin
- Nessun ritardo per trovare nuovi cammini

Contro:

- Utilizzo di banda per il mantenimento di cammini che potrebbero non venir mai usati
- Spreco di risorse (memoria e processore) per mantenere informazioni su cammini che potrebbero non venir mai utilizzati
- Tempo di inizializzazione, quantificato da 2 a qualche decina di secondi, a seconda del numero di nodi e della topologia

Parte II

Applicazione all'interporto di Nola (NA)

Capitolo 4

Hardware utilizzato

In questo capitolo si esamina l'hardware utilizzato per la messa a punto della rete a Nola. In modo particolare si approfondiscono i tre componenti principali che vengono utilizzati, due sistemi IBM compatibili, Il BlueBox ed il Display, progettati ed assemblati da Fantuzzi Reggiane ed un router WiFi della Linksys[1] su cui è stato sostituito il sistema operativo preesistente con uno modificato appositamente per lo svolgimento di questa tesi. Infine si tratta brevemente delle antenne WiFi utilizzate e si motivano le scelte fatte a livello hardware.

4.1 BlueBox e Display

Il BlueBox è un computer progettato e costruito per resistere a estremi di temperature e vibrazioni. La scatola di metallo in cui è racchiuso è completamente impermeabile e ne permette il montaggio ed il funzionamento continuo all'aperto.
Su un lato sono disponibili i connettori per porte seriali, ethernet, antenne radio WiFi ed alimentazione. Non sono previsti collegamenti per tastiera e video perché normalmente il collegamento avviene via porta seriale o tramite protocolli come telnet[20, 17] o ssh sulla connessione Ethernet.

Il BlueBox è molto versatile e può venir utilizzato sia come semplice ripetitore di segnali radio, con un montaggio, ad esempio, sulla cima di un palo dell'illuminazione, sia come processore per il calcolo della posizione tramite segnali GPS su di un mezzo in movimento.

Il Display ha la parte hardware, spiegata di seguito, in comune con il BlueBox, ma è dotato di uno schermo LCD sensibile al tocco per l'interazione con l'utente con un risoluzione di 800x600 pixel. Il Display viene montato nella cabina di manovra per fornire all'operatore un gran numero di informazioni, ad esempio possono venir mostrate visuali da telecamere esterne per facilitare la manovra, informazioni di diagnostica sullo stato della macchina,

[1] Una sottomarca di Cisco

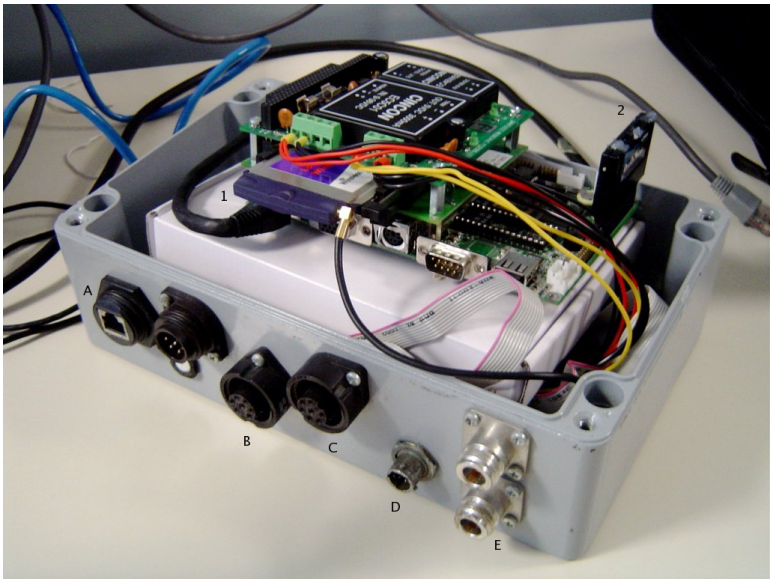

Figura 4.1: Foto del BlueBox utilizzato durante i test. L'hardware interno è stato alzato per permetterne una visione più facile.
Didascalia hardware:
1: Scheda wireless PCMCIA
2: Memoria permanente (Disk on Module)
Didascalia connettori:
A: Ethernet 100Mbit
B, C: Porte seriali
D: Alimentazione 24V
E: Connessioni per le antenne esterne

con segnali di allarme in caso di sovraccarichi o malfunzionamento, oppure un semplice elenco di operazioni da compiere.

4.1.1 Hardware

Il BlueBox è basato su un processore Geode a 230Mhz, con architettura Intel IA-32, che integra anche tutte le funzioni di scheda video e audio. Il processore è montato su una scheda madre Boser che comprende un comune chip Realtek per le funzionalità di rete e quattro porte seriali. Sempre sulla scheda sono anche presenti connettori per floppy, porta parallela e due porte USB 1.1, che però non vengono utilizzate. Il disco fisso è realizzato tramite un DOM (Disk On Module) con un'interfaccia IDE con connettore a 44 pin, uguale a quella usata per i dischi fissi per portatili, i quali, essendo dotati di parti mobili, non sono abbastanza resistenti alle sollecitazioni a cui sono tipicamente sottoposti i BlueBox.

Sulla scheda madre è possibile montare a torre delle schede di espansione su un bus PC/104, molto simile all'ISA. Nella configurazione utilizzata si è montata una scheda di alimentazione a 24 Volt e un'espansione per il bus PCMCIA[2].

Integrazione della scheda Senao

Tradizionalmente sui BlueBox vengono montate delle schede WiFi Cisco, che offrono buone prestazioni, ma mancano completamente del supporto per agire come AP (Access Point). È stato quindi necessario cercare una scheda PCMCIA, compatibile con lo standard WiFi 802.11b, in grado di funzionare in modalità AP e con un buon supporto a livello driver per il kernel Linux. In seguito ad una ricerca sui diversi progetti open source si è giunti alla conclusione che l'unico driver in grado di funzionare in quella modalità è HostAP[22] che supporta, però, solo schede basate su chipset Prism. Ulteriori ricerche tra i diversi produttori di hardware wireless hanno portato a scegliere un scheda prodotta da Senao[3], che risponde a tutti i requisiti tecnici richiesti per l'integrazione nei sistemi della Fantuzzi Reggiane.

4.1.2 Software

Il sistema operativo utilizzato è Linux con kernel versione 2.4 e Busybox versione 1.0 per tutto ciò che riguarda i comandi da shell e la gestione dei moduli del kernel.

Programmi aggiunti

Tutto il software aggiunto alla configurazione base è stato ricompilato con le versioni delle librerie C e del compilatore gcc[4] disponibili sui BlueBox per poter mantenere la compatibilità a livello binario con il sistema esistente. Per fare ciò è stato necessario creare un ambiente di compilazione apposito in quanto le versioni distribuite attualmente di quelle stesse librerie sono differenti. Per quanto possibile, infatti, non è consigliabile compilare software sul BlueBox a causa di problemi di velocità e di eccessivo consumo della Flash EPROM che realizza il disco fisso. Inoltre la dimensione dei pacchetti necessari alla compilazione avrebbe portato al limite la capacità di memorizzazione del disco.

Con il procedimento descritto si è proceduto all'installazione di `olsrd`, il processo che si occupa di realizzare il protocollo di routing OLSR (vedi il capitolo 3) ed il driver HostAP necessario alla scheda WiFi, con i sui programmi di contorno. Inoltre sono stati scritti tutti gli script necessari

[2]Personal Computer Memory Card International Association

[3]Il modello scelto è il 2511CD PLUS EXT2, che non ha antenna integrata e fornisce due uscite per antenne esterne

[4]Il compilatore C della suite sviluppata da GNU

affinché il sistema fosse completamente pronto ad operare all'avvio, senza alcuna supervisione esterna.

4.2 Linksys WRT54G e WRT54GS

Figura 4.2: Vista di fronte del router wireless Linksys WRT54g, sono visibili le spie utilizzate per controllare il funzionamento del router. Il LED marcato come DMZ viene usato per segnalare l'avvio del sistema operativo, viene spento quando il sistema ha finito la sequenza di boot ed è pronto ad iniziare le operazioni di routing.

Si è valutata la possibilità di utilizzare un dispositivo commerciale, a basso costo, al posto di uno o più BlueBox. La differenza di costo è all'incirca di un ordine di grandezza a svantaggio del BlueBox e quindi l'ipotesi è stata valutata molto seriamente. La Linksys commercializza due router con caratteristiche simili, il WRT54g ed il suo fratello maggiore WRT54gs. L'unica differenza consiste nella maggiore dimensione della memoria flash del modello gs, quindi per gli scopi di questa tesi, i due modelli sono funzionalmente identici e possono essere considerati interscambiabili in qualunque momento.

L'affidabilità del dispositivo è stato il problema principale affrontato. Il WRT54g è un router di fascia *consumer* e non è stato progettato né costruito per sopportare i rigori di un ambiente ostile. L'intervallo di temperature a cui il WRT54g può operare secondo le specifiche (da 0 a 40 gradi centigradi) è troppo limitato, ma si può ovviare predisponendo un sistema di condizio-

namento all'interno del contenitore impermeabile comunque necessario per poter posizionare il WRT54g all'esterno.

Un altro problema di affidabilità molto più delicato riguarda il software, infatti il dispositivo deve essere in grado di recuperarsi completamente in seguito a mancanze di corrente elettrica e deve essere del tutto privo di blocchi di sistema per cui si renda necessario un riavvio manuale.

4.2.1 Hardware

Il Linksys WRT54g è un router dotato di quattro porte ethernet in configurazione switch, una porta per la connessione ad una rete WAN (ADSL o linea dedicata) ed una scheda wireless con due antenne in modalità *diversity*. Queste ultime sono collegate da un attacco a vite e quindi facilmente sostituibili con altre di caratteristiche differenti.

Il processore è basato sull'architettura MIPS ed è collegato direttamente alla scheda wireless 802.11g[5], alla memoria volatile e alla memoria *flash* che mantiene il sistema in mancanza di alimentazione. Quest'ultima è suddivisa in tre aree distinte a livello software, di cui solo la parte centrale è facilmente riscrivibile e contiene il sistema operativo vero e proprio:

1. Bootloader: eseguito all'avvio del sistema legge alcuni parametri dall'NVRAM, predisponendo l'hardware e rimanendo in ascolto come server TFTP alcuni secondi sull'interfaccia Ethernet per permettere la riscrittura del sistema operativo. Dopodiché cede il controllo al codice contenuto nel primo blocco della parte centrale della memoria flash.

2. Sistema: occupa la maggior parte dello spazio e contiene tutto il sistema operativo ed i programmi utente. A seconda della configurazione può essere suddiviso in due partizioni oppure una sola.

3. NVRAM: occupa pochi kB e mantiene un gran numero di parametri della forma nome=valore. Ha lo scopo di mantentere informazioni di configurazione tra i riavvii e addirittura tra i cambi di sistema operativo. È Possibile memorizzare qualunque tipo di informazione, binaria o testuale.

4.2.2 Software

Il sistema operativo basato su Linux fornito di fabbrica nel WRT54g è stato sostituito con la distribuzione OpenWRT[6], creata appositamente per i diversi modelli compatibili di Linksys. Questa distribuzione fornisce tutti i pacchetti di base e consente in più di scaricare altro software da Internet

[5] In grado di interoperare anche col protocollo 802.11b, condizione necessaria per permettere la comunicazione con gli altri dispositivi utilizzati.

[6] http://openwrt.org

con una procedura semplificata, usando il sistema di pacchetti ipkg, diffuso anche su altre distribuzione per palmari. Inoltre il progetto OpenWRT fornisce tutto il software necessario alla compilazione per architettura MIPS.

OpenWRT è uno dei due sistemi che sono stati testati sul WRT54g ed è quello che ha superato le prove descritte nel capitolo 6, l'altro, chiamato FreiFunk, dal nome di un progetto di rete wireless regionale in paesi di lingua tedesca e basato su una versione precedente di OpenWRT, ha mostrato di avere problemi di stabilità della parte wireless, anche se è dotato di un'ottima interfaccia di configurazione via web, che infatti è stata riadattata, come descritto nella sezione 5.4.
Esistono anche altri progetti, alcuni a pagamento, di firmware alternativi per il WRT54g, ma non sono stati presi in considerazione perché non offrono il codice sorgente di alcune parti utilizzate, e non sono dotate nativamente di un sistema di pacchetti flessibile come quello di OpenWRT.

4.3 Antenne

La scelta delle antenne è stata molto lunga e laboriosa perché si sono dovuti tenere in conto parecchi parametri diversi oltre alla qualità effettiva dell'apparato, già difficile da misurare di per sè. In particolare si sono tenuti in conto lunghezza dei cavi, tipo di connettore, facilità di approvvigionamento, prezzo e tipo di montaggio.

Le antenne differiscono anche per il tipo di irradiazione, esistono antenne omnidirezionali, settoriali e punto a punto (paraboliche), infatti sui cataloghi, disponibili anche su Internet, vengono mostrati gli schemi di diffusione che mostrano sul piano orizzontale e verticale il guadagno dell'antenna nelle diverse direzioni.

4.3.1 Cisco AIR-ANT2506

Quest'antenna Cisco è dotata di un cavo di circa un metro, un connettore RP-TNC (abbastanza difficili da trovare in commercio) e di una fascetta di metallo per il montaggio lungo un palo. È omnidirezionale sul piano orizzontale e genera dei lobi verso il basso e verso l'alto sul piano verticale, atti a coprire le aree direttamente sopra e sotto il punto di montaggio.

4.3.2 Huber+Suhner SOA 2400/360/4/20/V

Quest'antenna ha la particolarità di dover essere montata a soffitto, infatti viene fornita di viti e tasselli per il fissaggio. È pensata per irradiare verso il basso in un'ampia zona come in una stazione del treno od una sala d'aspetto d'aeroporto. È dotata di un connettore N ed è senza cavo. Visto lo schema di irraggiamento è una valida alternativa all'antenna Cisco per il montaggio

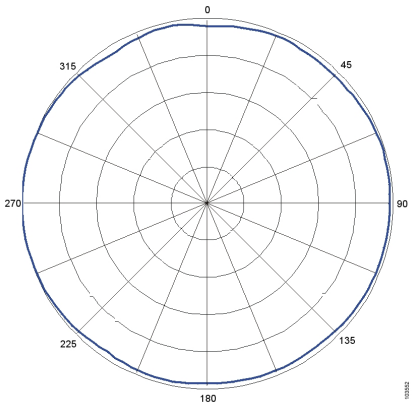

Figura 4.3: Diagramma dell'antenna Cisco sul piano orizzontale.

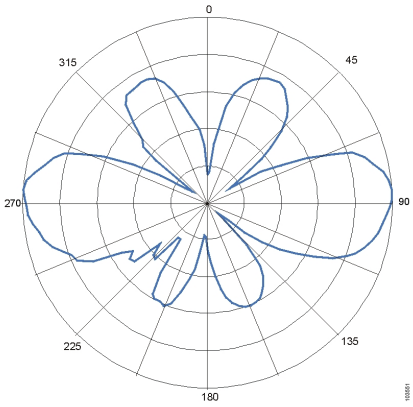

Figura 4.4: Diagramma dell'antenna Cisco sul piano verticale.

in cima alle torri faro, garantendo un irraggiamento sia in orizzontale, che verso il basso.

4.4 Scelte fatte e motivazioni

L'applicazione all'interporto di Nola prevede l'uso di tre o quattro nodi fissi, montati sulla cima di torri faro alte circa 20 metri e di 2 nodi mobili, montati sugli *stacker*[7].

Si prevede quindi di assemblare sette nodi, tutti dotati di una sola antenna. Per semplicità costruttiva e di fornitura si è deciso di adoperare la stessa antenna e lo stesso hardware per tutti i nodi. Vista la piccola differenza di prestazioni tra le due antenne, come evidenziato nel capitolo 6, si è deciso di usare l'antenna Cisco in quanto più adatta al montaggio sugli stacker.

Per l'hardware, dopo i diversi test di affidabilità, si è deciso di usare il router Linksys, che malgrado la necessaria dotazione di riscaldatore, termostato e scatola impermeabile, ha un prezzo decisamente inferiore.

[7]Mezzi mobili su ruote dotati di un braccio estensibile in grado di afferrare un container da un vagone ferroviario od un camion e posarlo su una pila alta fino a 12 metri.

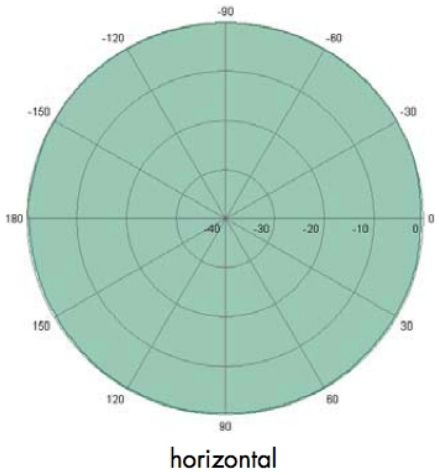

Figura 4.5: Diagramma dell'antenna Huber+Suhner sul piano orizzontale.

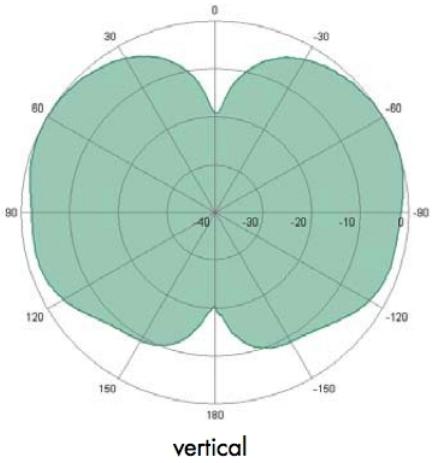

Figura 4.6: Diagramma dell'antenna Huber+Suhßner sul piano verticale.

Capitolo 5

Software

Il lavoro sul software si è svolto su diversi fronti, toccando tutti i livelli, dal sistema operativo alle applicazioni utente. Una prima parte di implementazione ha riguardato la nuova distribuzione Linux che è stata installata sui router Linksys. Questi sono a tutti gli effetti dispositivi embedded che richiedono sia un kernel apposito, sia applicazioni compilate ad hoc. Inoltre la particolare applicazione ha richiesto una personalizzazione di alcuni elementi software perché si adattassero alle esigenze di Fantuzzi Reggiane. Infatti in azienda non era mai stato sviluppato un prodotto embedded basato su un'architettura diversa da Intel e con tali ristrettezze di risorse di memoria.

Per il sistema operativo ci si è basati sulla distribuzione Linux embedded OpenWRT alla quale è stata modificata la configurazione dei pacchetti installati e del kernel Linux che viene compilato ed installato sul router. Inoltre, dato che non tutto il software necessario era disponibile in OpenWRT, sono stati creati alcuni nuovi pacchetti, modificando il sistema di build di OpenWRT in modo che questi diventassero parte integrante della distribuzione a tutti gli effetti, facilitando enormemente ogni operazione successiva di aggiornamento che dovesse rendersi necessaria.

Il software applicativo a livello utente che è stato sviluppato permette al personale che gestisce la rete di osservare in ogni istante lo stato di funzionamento di tutti i nodi MeshAP, sia fissi che mobili grazie al programma di monitoraggio. In caso di problemi il programma di diagnostica è stato scritto per essere eseguito da un portatile direttamente collegato al nodo malfunzionante tramite un cavo *ethernet* installato appositamente.

Nel caso in cui si rendano necessarie riconfigurazioni significative dell'intera rete senza fili, tutti i MeshAP sono dotati di un'interfaccia web di configurazione che, in un'area controllata da password, permette di modificare i parametri di funzionamento della scheda wireless e di `olsrd`, gli indirizzi IP e di effettuare l'aggiornamento del firmware.

5.1 OpenWRT

OpenWRT è la distribuzione Linux che è stata utilizzata come base per il sistema operativo modificato installato sui router WRT54G. OpenWRT fornisce il kernel ed un insieme di programmi di base; tra questi è disponibile il sistema `ipkg` per l'installazione di ulteriori pacchetti.

Per effettuare i test è stato necessario inserire in OpenWRT `netperf`, un software di misura della banda di rete. Questo programma presenta diversi problemi di compilazione quando portato sull'architettura mips ed ha richiesto alcune modifiche al codice sorgente. È stata effettuata anche la ricompilazione dell'ultima versione di `olsrd`, dato che non era ancora disponibile in OpenWRT. Sono stati creati anche altri pacchetti, tra cui uno per il server del programma di monitoraggio e diagnostica ed uno per gli script e programmi utilizzati durante la fase di test (cap. 6).
Per inserire questi software nel sistema di compilazione di OpenWRT si sono dovuti modificare alcuni *makefile*, creare le istruzioni di pacchettizzazione per `ipkg` ed infine scrivere un file di descrizione del contenuto del pacchetto contenente informazioni quali autore, licenza, sito web, ecc.

La scrittura ed il porting di software non è sempre facile ed immediata, bisogna infatti tener conto che l'architettura usata sui WRT54g è mips, che ha un diverso ordinamento dei byte rispetto ad Intel, inoltre OpenWRT usa ucLib come libreria standard C che fornisce la maggior parte, ma non tutte, delle chiamate di sistema disponibili nelle più usate (e più grosse) GNU libc. Queste differenze possono causare dei problemi di ricompilazione nei software sviluppati solo su Intel senza le dovute precauzioni per il porting del software su ambienti differenti.

OpenWRT fornisce inoltre una serie di strumenti che velocizzano la preparazione di una nuova immagine del sistema che può essere scritta al posto di quella fornita di fabbrica dalla Linksys grazie ad alcuni comandi appositi.

5.2 Software di monitoraggio

Il software ha una struttura client/server, con la comunicazione che avviene tramite il protocollo UDP sulla porta 9000. La figura 5.1 aiuta a comprendere l'architettura del sistema: il server è in esecuzione su ogni nodo, sia fisso che mobile (M ed R) e raccoglie tutte le informazioni da inviare al client che è attivo sulla postazione del sistemista (D o L).

Il client presenta un'interfaccia grafica in grado di visualizzare i dati di funzionamento di un nodo alla volta, mostrando informazioni quali:

- Raggiungibilità
- Numero di pacchetti e byte inviati e ricevuti
- Numero di errori in trasmissione e ricezione

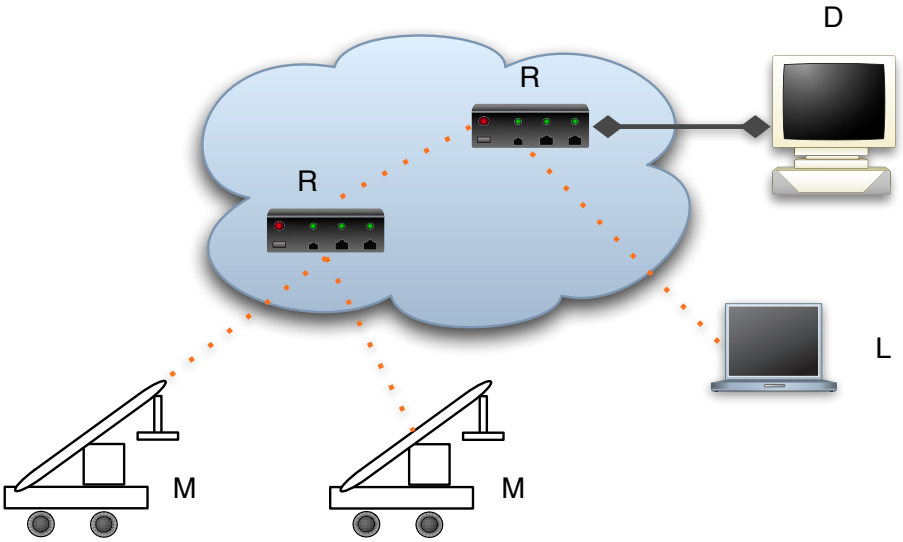

Figura 5.1: Struttura del sistema basato su MeshAP. I nodi mobili M ed L comunicano via WiFi con un certo numero di nodi fissi R che si occupano del routing dinamico. Infine uno di questi nodi è collegato via cavo con la rete fissa del porto e col computer del sistemista (D).

- Livello di segnale e rumore in dBm dell'apparato radio
- Tempo di funzionamento dall'ultimo riavvio
- Memoria libera
- numero di processi in esecuzione
- Disponibilità della funzione di disegno della rete e di interfaccia web

L'elenco dei nodi si trova sulla parte sinistra dell'interfaccia (vedi fig. 5.2) e fornisce una visione immediata dello stato di tutta la rete. Per ogni nodo viene mostrato un segnale:

- verde: nodo raggiungibile
- rosso: nodo non raggiungibile

Selezionando un nodo vengono mostrate sulla destra tutte le informazioni disponibili, aggiornate all'ultimo pacchetto ricevuto. Infatti il processo client fa solo un controllo per scartare pacchetti arrivati fuori ordine e non chiede la ritrasmissione dei pacchetti persi durante l'invio.

In basso a sinistra sono disponibili i pulsanti per modificare l'elenco dei nodi. Il pulsante di aggiunta causa l'apertura di una finestra che richiede

l'inserimento di tutti i dati necessari e ne controlla la validità prima di proseguire con l'esecuzione. Il pulsante di rimozione chiede una conferma per evitare errori dovuti a distrazione. In entrambi i casi le modifiche vengono salvate su disco immediatamente ed in maniera trasparente.

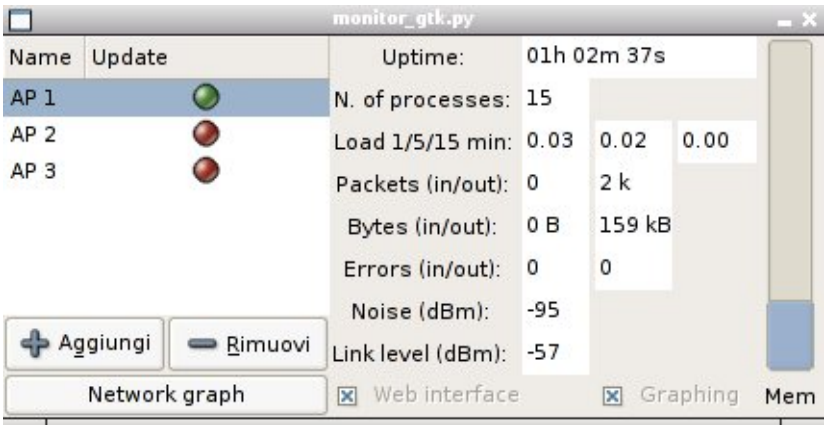

Figura 5.2: Finestra principale del programma di monitoraggio. L'immagine è stata presa su Linux, ma il programma funziona senza modifiche anche su Windows e Mac Os X. Il nodo AP 1 è attivo, mentre gli altri due sono irraggiungibili.

Il programma client fornisce anche la possibilità di disegnare un grafo della rete a routing dinamico se esiste almeno un nodo con il necessario plugin di `olsrd`. Il sistema funziona convertendo i dati in formato DOT ricevuti da `olsrd` in un'immagine PNG che viene mostrata all'utente e che è possibile salvare su disco per riferimento.

Il server è scritto in C e tenendo conto dell'hardware molto limitato su cui viene eseguito mantiene uno stato in memoria molto limitato. All'avvio l'esecuzione si ferma dopo una breve inizializzazione in attesa di un comando di avvio dal client, dopodiché inizia ad inviare ad intervalli regolari le informazioni lette tramite varie chiamate al sistema. La durata di questo intervallo viene definita dal client ed è misurata in secondi.

5.3 Software di diagnostica

Anche questo software ha una struttura client/server, ma per la parte server sfrutta lo stesso programma in esecuzione sui nodi utilizzato per il monitoraggio. Questa scelta è stata presa per risparmiare risorse ed evitare inutili duplicazioni di codice.

Il programma è pensato per essere eseguito da un portatile Windows collegato direttamente al nodo malfunzionante tramite un cavo Ether-

net lasciato in una posizione di facile accesso al momento dell'installazione iniziale.

Figura 5.3: Finestra principale del programma di diagnostica.

Le operazioni che è possibile effettuare sono:

- Riavvio dell'interfaccia di rete
- Riavvio del sistema
- Accesso all'interfaccia web (se disponibile)
- Accesso al terminale tramite shell remota ssh

Le prime due operazioni sono da effettuare solo se il nodo risulta non raggiungibile dal programma di monitoraggio. In fase di test non si è riusciti a ricreare una situazione di blocco totale di sistema, ma è possibile che avvenga in situazioni estreme di temperatura. In particolare l'interfaccia Ethernet si è sempre mostrata completamente affidabile, consentendo l'accesso via cavo anche quando la parte WiFi, più delicata e meno affidabile, fosse irraggiungibile.

L'interfaccia web consente di ottenere informazioni di funzionamento molto più dettagliate di quelle disponibili tramite il software di monitoraggio, inoltre consente di accedere ad una sezione amministrativa in cui è possibile modificare alcuni parametri di funzionamento.

L'accesso al terminale offre un accesso shell al router in cui sono disponibili la maggior parte dei comandi UNIX. L'accesso avviene con un utente senza privilegi di root e quindi non è possibile eseguire comandi privilegiati senza prima autenticarsi tramite il comando `su`.

5.4 Interfaccia web di configurazione

Per semplificare i compiti più frequenti di amministrazione dei singoli nodi è stata realizzata un'interfaccia web che consenta di accedere a due aree distinte: la prima, visibile senza restrizioni, consente di visualizzare un gran nume-

ro di dati sul funzionamento corrente del nodo. Le informazioni visualizzate comprendono:

- Generali
  - indirizzi IP
  - stato delle interfacce
  - codici di identificazione
  - log di avvio (dmesg)
  - log di sistema (syslog)
- Scansione delle reti WiFi presenti in zona
- Tabella di routing
- Stato interno di `olsrd`

Nella seconda area, di amministrazione vera e propria, sono disponibili pagine per:

- Cambio password (utente root ed interfaccia web)
- Configurazione di `olsrd`
- Impostazioni WiFi (paramtri 802.11 e indirizzo IP)
- Restrizione indirizzi MAC
- Indirizzi IP LAN
- Aggiornamento firmware
- Riavvio

L'accesso a questa area è controllato da una password che viene tenuta in sincronia con la password dell'utente root. Questo è possibile perché si prevede che le comunicazioni tra i nodi siano sempre crittografate tramite WEP, altrimenti la password verrebbe inviata in chiaro ad ogni accesso di una delle pagine di amministrazione. Una nota informa l'utente di questo pericolo.

Questo sistema di configurazione web, basato sull'idea dell'interfaccia di FreiFunk, ma in gran parte riscritto da zero, funziona grazie a lighttpd, un server HTTP grande qualche decina di kilobyte ed una serie di script CGI scritti in linguaggio POSIX shell[1].

Mentre la maggior parte delle pagine è basata su una serie di chiamate al sistema per modificare i parametri in NVRAM o nei file di configurazione,

[1]Su OpenWRT è disponibile soltanto ash, una shell minimale che usa un sottoinsieme del linguaggio di bash.

l'interfaccia di aggiornamento del firmware è stata scritta in modo particolare, infatti funzionando a partire da uno script CGI, viene eseguita dal processo del server HTTP, che legge la nuova immagine del sistema tramite una richiesta POST e inizia a scriverla su flash non appena sia terminato il trasferimento e si siano fatti alcuni controlli di integrità. Il problema consiste nel fatto che il processo di scrittura non deve essere interrotto a metà per evitare che il router, come si dice in gergo, diventi un mattone buono solo per fermare le porte. A questa esigenza si somma la necessità di fornire all'utente un feedback sull'operazione in corso.
Terminata l'operazione di scrittura sono necessari tre riavvii del sistema: il primo, eseguito direttamente dallo script CGI, serve ad inizializzare il filesystem JFFS ed il secondo, effettuato da uno script apposito chiamato al boot, per renderlo scrivibile. Un terzo riavvio è necessario al sistema per cancellare lo script responsabile del riavvio, in quanto il filesystem in sola lettura non consente tale opearzione dopo il secondo riavvio. Per fortuna il sistema è veloce e tutta l'operazione di aggiornamento non dura più di due minuti.

La parte di amministrazione è stata scritta tenendo in conto esigenze di espandibilità, infatti ogni pagina costituisce un modulo indipendente costituito da uno scheletro comune a tutti ed una parte specifica. Questo consente di poter aggiungere e togliere moduli molto facilmente, senza causare danni involontari in altre parti dell'interfaccia.
Grazie a questa progettazione e al linguaggio usato per l'implementazione è possibile utilizzare la stessa interfaccia in un gran numero di situazioni ed architetture differenti con un numero molto limitato di modifiche.

5.5 Configurazione di olsrd

Seguono le parti più rilevanti del file di configurazione di `olsrd`, segnalando, ove necessario, le differenze tra la configurazione di un router di frontiera ed uno all'interno della rete ad instradamento dinamico.

5.5.1 HNA (Host Network Advertise)

```
Hna4
{
# Published subnet:
192.168.1.10 255.255.255.255
}
```

HNA è un tipo speciale di messaggio che `olsrd` utilizza per pubblicizzare a tutti gli altri nodi la presenza di un gateway verso un'altra rete. Questa configurazione è fondamentale per i nodi frontiera, che sono collegati via ethernet ad uno o più host che non eseguono OLSR. Unito al proxy ARP,

questo meccanismo consente di creare una rete a routing dinamico completamente trasparente agli utilizzatori, che non devono né installare software aggiuntivo, né cambiare parametri di configurazione sulle loro macchine per poter accedere a tutta la rete.

5.5.2 Isteresi

```
UseHysteresis yes

HystScaling 0.30
HystThrHigh 0.70
HystThrLow 0.15
```

L'isteresi e la formula usata da `olsrd` per calcolare la raggiungibilità di un nodo sono descritte nella sezione 3.2.1. I valori dei parametri indicati sopra sono stati scelti empiricamente, in base a prove sul campo (vedi 6.4.3), con l'obiettivo di rendere più stabile la scelta dei cammini, che altrimenti risulta troppo rapida ad adattarsi ai cambiamenti nella topologia e meno stabile nelle configurazioni relativamente statiche.

5.5.3 Interfacce e tempi di validità

```
Interface "vlan0" "eth1"
{
    # Hello interval in seconds
    HelloInterval    2.0

    # HELLO validity time
    HelloValidityTime 10.0
}
```

`olsrd` ha bisogno di sapere su quali interfacce deve inviare e ricevere i suoi pacchetti. Per i nodi interni non è necessario l'invio di messaggi sull'interfaccia vlan0 (rete fissa), ma non ci sono controindicazioni. Al contrario consente ad un operatore dotato di portatile con OLSR in esecuzione di avere accesso a tutta la rete per risolvere eventuali problemi.
I messaggi di Hello servono a costruire l'elenco di primi vicini, primo passo per la comprensione della topologia di rete. Avendo aumentato il tempo in cui tali messaggi vengono considerati validi, si rallenta ulteriormente il tasso di cambiamento dell'instradamento, aumentandone la stabilità.

5.5.4 Plugin

```
LoadPlugin "olsrd_dot_draw.so.0.3"
{
```

```
        PlParam "port" "2004"
        PlParam "accept" "192.168.1.10"
}

LoadPlugin "olsrd_httpinfo.so.0.1"
{
        PlParam "port" "8080"
        PlParam "Net" "192.168.1.0 255.255.255.0"
}
```

La sezione LoadPlugin indica a `olsrd` quali plugin deve caricare e con quali parametri. Questa configurazione viene usata solo su un router di frontiera, in modo tale che esso possa fornire le informazioni necessarie al programma di monitoraggio per disegnare un grafo della rete ed a fornire un'interfaccia web per il monitoraggio dello stato interno di OLSR.

Tutte le sezioni sopra descritte sono state evidenziate in modo particolare nel file di configurazione, in modo da facilitarne la modifica tramite l'interfaccia web.

5.6 Proxy ARP

I comandi evidenziati di seguito attivano la funzione di proxy ARP tra le interfacce senza fili ed ethernet.

```
echo 1 > /proc/sys/net/ipv4/conf/vlan0/proxy_arp
echo 1 > /proc/sys/net/ipv4/conf/eth1/proxy_arp
```

Il proxy ARP risolve in modo molto elegante e poco dispendioso un problema abbastanza grave che si è presentato nel momento in cui si è voluto collegare un sistema senza funzionalità di routing dinamico alla rete col minimo possibile di impostazioni sul routing IP. Il problema, quindi, riguarda soltanto quei casi in cui si utilizzerà il MeshAP come ponte tra un rete tradizionale ed una a routing dinamico.

Questi nodi frontiera sono collegati ad un'altra rete che può essere costituita da una LAN vera propria, o soltanto da un host, come nel caso dei Display sugli stacker, dove si collegherà un nodo MeshAP al Display tramite un semplice cavo Ethernet cross.

Impostare sul MeshAP come router IP non basta, infatti gli host esterni alla rete OLSR necessiterebbero comunque di impostare un cammino specifico, che ciò che si vuole evitare. Infatti le richieste ARP non vengono passate da un'interfaccia all'altra all'interno del nodo MeshAP, dato che questo si occupa soltanto di routing a livello IP.

Il proxy ARP funziona rispondendo col proprio indirizzo MAC a tutte le richieste ARP dirette ad indirizzi IP per cui esiste un cammino nella tabella

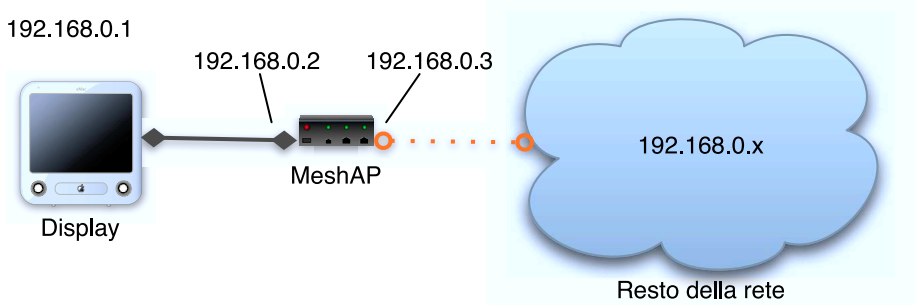

Figura 5.4: Schema di una rete di esempio per il funzionamento di proxy ARP

di routing. Nella tabella 5.1 vengono descritti i passi compiuti dai due host in una situazione normale ed in una configurazione con proxy ARP. La rete usata è visibile in figura 5.4.

Tabella 5.1: Scambio di messaggi con e senza proxy ARP. Nella colonna sinistra sono descritte le operazioni compiute dal Display (il nodo 'stupido') e sulla destra quelle del MeshAP.

Senza proxy ARP	
Pacchetto da inviare a 192.168.0.10	
Invio richiesta ARP per 192.168.0.10	Richiesta scartata perché l'indirizzo non è conosciuto
Nessuna risposta, invio del pacchetto fallito	
Con proxy ARP	
Pacchetto da inviare a 192.168.0.10	
Invio richiesta ARP per 192.168.0.10	Ricezione richiesta, l'indirizzo è raggiungibile secondo la tabella di routing attraverso un'altra interfaccia.
Attesa	Invio risposta ARP con il proprio indirizzo MAC
Invio pacchetto al MeshAP, ma con destinazione IP differente	Routing del pacchetto verso l'interfaccia wireless.

Quindi il proxy ARP funziona ingannando l'host con cui sta comunicando, facendogli mantenere nella cache ARP una coppia (IP, MAC) falsa, ma che consente la trasmissione dei pacchetti.

L'unione del proxy ARP in una direzione e dei messaggi HNA di OLSR nell'altra consente di rendere la rete senza fili a routing dinamico completamente trasparente a tutti gli host collegati. Infatti questi non devono avere

impostazioni particolari di routing avendo a disposizione, a livello IP, una visione globale della rete, come se fosse una tradizionale rete fissa. Tutta la complessità dei nodi mobili viene gestita in maniera trasparente dai MeshAP e dal protocollo OLSR.

5.7 Sviluppi futuri

Sia i programmi di monitoraggio e diagnostica che l'interfaccia di amministrazione via web possono essere facilmente riutilizzati per altri progetti essendo stati scritti tenendo presenti concetti di modularità e portabilità.

Il sistema presente sui MeshAP si presta molto bene ad essere riutilizzato per la copertura WiFi di vaste aree, ma con la limitazione che, date le dimensioni, il peso e le necessità energetiche, possa essere montato solo in posizione fissa o su mezzi a motore. La possibilità di realizzare un MeshAP su hardware differente, in grado di funzionare a batterie e fornire accesso ad un computer portatile o addirittura ad un palmare è molto interessante ed andrebbe esplorata a fondo.
È anche possibile utilizzare direttamente l'hardware del MeshAP per altri compiti, oltre al routing dei pacchetti che lo tiene impegnato solo per un tempo minimo, realizzando in modo compatto un gran numero di applicazioni, dalla visualizzazione dati alla comunicazione vocale tramite *Voice over IP*.

Capitolo 6

Test e prove effettuate

In questo capitolo si descrivono le prove effettuate per verificare la funzionalità di hardware e software prima dell'applicazione a Nola. I test sono stati effettuati con l'obiettivo di valutare le prestazioni dell'implementazione di OLSR e dei diversi dispositivi hardware (schede, antenne) e si sono svolti sia al chiuso con l'ausilio di tecniche di *firewalling* per simulare la visibilità tra i nodi della rete, sia all'aperto, cercando di ricostruire gli stessi scenari presenti a Nola in quanto a distanze, altezze e velocità dei nodi della rete.

6.1 Latenza nel routing di OLSR

Questa serie di test ha avuto lo scopo di verificare quanto pesi il routing sui tempi di latenza della rete. Dato che a Nola non si prevede di avere cammini di lunghezza maggiore di tre/quattro salti è stato possibile effettuare una simulazione con quattro dispositivi: due portatili, il WRT54g ed un BlueBox. Tutti i dispositivi sono stati collegati via Ethernet attraverso lo switch fornito dal WRT54g per poter comunicare via telnet comandi e dati senza inquinare la rete senza fili con pacchetti estranei a quelli del test.

Questo test, più che misurare una latenza introdotta dall'uso di `olsrd`, verifica le prestazioni del codice di routing all'interno del kernel Linux. Infatti una volta che `olsrd` ha stabilito un percorso verso una certa destinazione aggiunge una *route statica* alle tabelle del kernel, che si comporta da quel momento in poi come se fosse stata inserita manualmente. Inoltre il test è statico ed è iniziato dopo che `olsrd` ha scelto una configurazione stabile, visibile di seguito:

```
--- 08:42:41.12 --------------------------------- LINKS

IP address       hyst   LQ     lost   total  NLQ    ETX
192.168.128.1    1.000  0.000  0      0      0.000  0.00
192.168.128.2    1.000  0.000  0      0      0.000  0.00
192.168.128.11   1.000  0.000  0      0      0.000  0.00
```

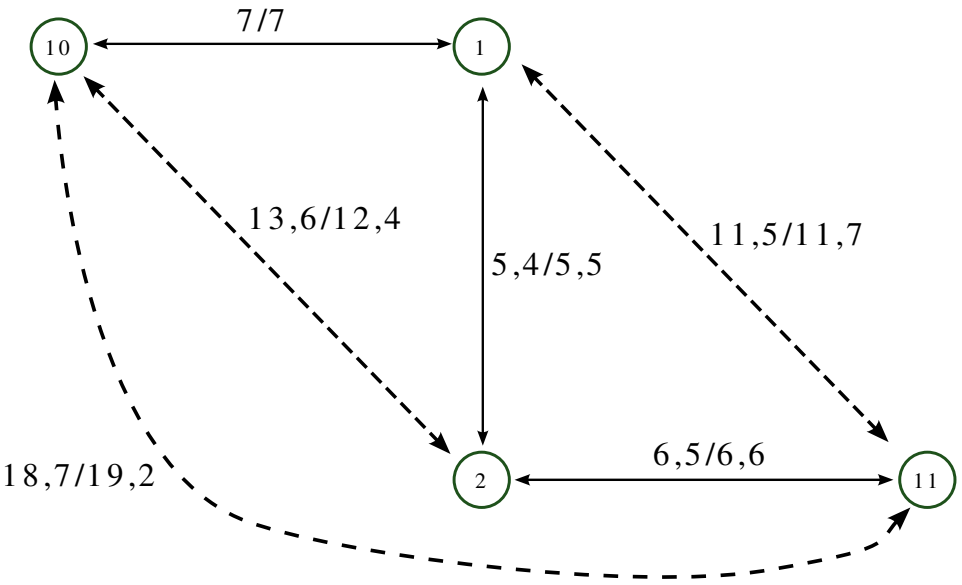

Figura 6.1: Grafo della rete utilizzata per i test di latenza, i dati sono stati ricavati da una media di 40 misurazioni effettuate con pacchetti ICMP echo request/reply.

```
--- 08:42:41.12 ------------------------------ NEIGHBORS

IP address       LQ     NLQ    SYM   MPR   MPRS  will
192.168.128.1    0.000  0.000  YES   NO    NO    3
192.168.128.2    0.000  0.000  YES   NO    YES   3
192.168.128.11   0.000  0.000  YES   NO    NO    3

--- 08:42:41.12 ------------------------------- TOPOLOGY

Source IP addr   Dest IP addr     LQ     ILQ    ETX
192.168.128.1    192.168.128.10   0.000  0.000  0.00
192.168.128.1    192.168.128.2    0.000  0.000  0.00
192.168.128.2    192.168.128.1    0.000  0.000  0.00
192.168.128.2    192.168.128.11   0.000  0.000  0.00
192.168.128.10   192.168.128.1    0.000  0.000  0.00
192.168.128.11   192.168.128.2    0.000  0.000  0.00
```

Le parti denominate LINKS e NEIGHBORS danno informazioni sui vicini del nodo che ha dato queste informazioni in output. Invece la sezione TOPOLOGY mostra la topologia della rete, indicando le coppie di indirizzi IP sorgente/destinazione dei collegamenti. Una rappresentazione grafica della

rete è visibile in figura 6.1, dove sono anche visibili i tempi di latenza per pacchetti di 64 byte. Si può notare come il tempo necessario ad andare dal primo nodo (il 10) all'ultimo (l'11) equivalga alla somma dei tempi dei singoli passaggi, la piccola differenza tra i due tempi è spiegabile con il tempo introdotto da ciascun nodo per le decisioni di routing, che comunque risulta essere al più di un millisecondo.

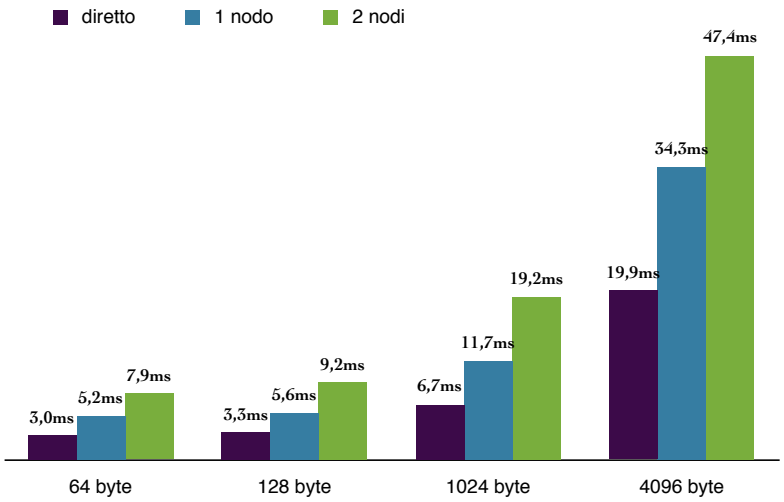

Figura 6.2: Latenza della rete OLSR al variare del numero di nodi intermedi e della dimensione dei pacchetti usati per la misura. I dati sono derivati dalla media di 40 misurazioni effettuate con pacchetti ICMP *echo request/reply*.

Con gli stessi dati è stato possibile ricavare il grafico visibile in figura 6.2, dal quale è evidente l'aumento con una progressione lineare dei tempi di latenza all'aumentare del numero di nodi attraverso cui il pacchetto deve passare. Ogni passaggio introduce, infatti, un ritardo costante dovuto al tempo necessario a:

- leggere il frame dall'interfaccia di rete
- attendere l'arrivo di tutti gli altri frame nel caso il pacchetto sia stato frammentato
- scegliere un instradamento consultando le tavole di routing[1]

[1] In Linux questo passo avviene in O(log(n)) con n numero di cammini presenti nella tavola

- inviare il pacchetto all'interfaccia di rete, eventualmente frammentato

Le misure sono state effettuate anche con pacchetti che superano la dimensione massima accettata dal protocollo 802.11 di 1500 byte. In questo caso il livello IP ha dovuto gestire più frame di dimensione minore, riassemblando e suddividendo nuovamente i pacchetti durante il passaggio in ogni nodo. Come ci si aspettava si è visto che il routing di pacchetti frammentati non comporta ritardi evidenti rispetto al caso non frammentato.

6.2 Affidabilità al riavvio

Questa classe di test si è resa necessaria quando, lavorando intensamente con il WRT54g, si è notato che in un certo numero di casi il firmware FreiFunk non caricava le impostazioni relative all'interfaccia wireless. Essendo questo un comportamento non accettabile per un dispositivo che deve essere in grado di operare in maniera continuativa, si è deciso di utilizzare una versione più recente del firmware, che ancorché sperimentale, garantiva maggiore stabilità.

Per il test si è utilizzato un timer che ha scollegato e ricollegato l'alimentazione ogni 5 minuti per un periodo di 4 giorni. Per verificare il funzionamento delle interfacce di rete si è utilizzato un portatile dotato di scheda WiFi, sul quale uno script scritto appositamente ha provveduto a mantenere un file con le informazioni di raggiungibilità sia per la scheda Ethernet, che per la scheda WiFi.
Il nuovo firmware, OpenWRT, ha avuto una percentuale di successo del 100% in questo test, ma solo dopo che è stato aggiunto un comando nella sequenza di avvio per forzare la scheda WiFi in modalità 802.11b, che altrimenti, in maniera casuale, rimaneva impostata in 802.11g, impedendo la comunicazione con il resto della rete.

6.3 Prove in camera climatica del WRT54g

Un altra prova necessaria per verificare la possibilità di usare il WRT54g sul campo è una verifica del suo funzionamento in condizioni ambientali estreme, sia per verificare il reale intervallo di temperature a cui può funzionare, sia per decidere sulla necessità o meno di racchiudere il router in una scatola dotata di condizionamento ambientale.

Visto che il WRT54g ha già normalmente dei problemi di surriscaldamento quando si trova sotto carico, gli è stata montata una ventola di raffreddamento che soffiasse aria sui componenti principali della scheda interna prima ancora di cominciare il test nella camera climatica.
Quest'ultima ha la forma di un grosso figorifero, ma è in grado di generare al suo interno un ambiente con temperature tra -20° e +90° e umidità relativa

fino al 100%, consentendo di provare hardware in ambienti molto diversi in poche ore ed in modo totalmente controllato.

Durante tutto il test, durato circa 4 ore, il WRT54g, insieme al suo alimentatore, è rimasto sempre collegato via Ethernet con un cavo passante per l'apposita apertura nella camera termica e via WiFi, con una modalità simile a quella usata per il descritto nella sezione 6.2. Utilizzando ping, impostato per richiedere una risposta una volta al secondo, e telnet, manualmente per verificare il funzionamento più generale del sistema operativo, si è potuto monitorare lo stato del sistema per tutta la durata del test.

Il grafico in figura 6.3 mostra i tempi di ping (latenza) al variare della temperatura e dell'umidità.

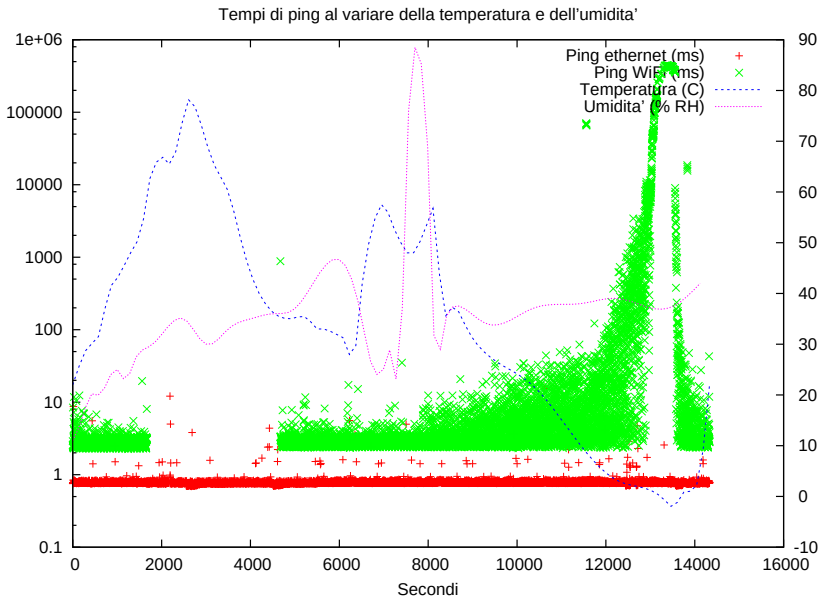

Figura 6.3: Grafico che descrive l'andamento del tempo di latenza misurato con ping durante la prova in camera termica. Sull'asse y a sinistra sono indicati i tempi, mentre sull'asse di destra ci sono temperatura e umidità relativa in percentuale.

6.3.1 Alta temperatura

Da 0 a 6000 secondi si è svolto il test ad alta temperatura, partendo dalla temperatura ambiente (circa 23° centigradi) si è arrivati fino a 80°. Mentre

la connessione Ethernet non ha mostrato cedimenti, neanche sotto forma di variazioni fuori dalla media dei tempi di ping, la parte wireless ha ceduto intorno a 65-70 gradi, senza più riprendersi fino a che non è stato eseguito il comando `iwconfig` per richiedere informazioni sul funzionamento del driver wireless.

Le specifiche del produttore indicano una temperatura massima di funzionamento di 50°, quindi si era già previsto di utilizzare una scatola dotata di ventole di raffreddamento, ma il test ha confermato questa necessità, in quanto la temperatura all'interno di una scatola chiusa sotto il sole, in particolare vicino a Napoli, può superare abbondantemente i 70°, se non viene prevista un'adeguata ventilazione.

6.3.2 Alta umidità

Una volta fatta scendere la temperatura è iniziato il test di umidità, che è durato fino a 8500 secondi. L'umidità relativa è stata fatta salire fino al 90%, quando, vedendo che il WRT54g non presentava problemi, il test è stato interrotto bruscamente, aprendo la porta della camera termica per far uscire il vapore, per evitare la formazione di condensa che avrebbe potuto danneggiare permanentemente il sistema.

6.3.3 Bassa temperatura

L'ultima parte del test è servita a verificare l'affidabilità alle basse temperature, dimostrando che il limite di 0°, dato come limite inferiore dalle specifiche tecniche, è molto preciso. Come si può vedere dal grafico, infatti, non appena la temperatura si è avvicinata al punto critico il router ha iniziato a perdere pacchetti ed a impiegarci un tempo sempre più lungo rispondere ai pacchetti di test, arrivando prima dell'interruzione completa delle risposte a dei tempi di risposta di diversi minuti.

6.3.4 Conclusioni

Questa prova ha dimostrato la necessità di racchiudere il router in una scatola dotata di termostato ed impianto di riscaldamento, dato che in inverno è possibile che la temperatura si avvicini a 0 gradi. La possibilità di aggiungere anche un impianto di ventilazione per il raffreddamento, invece, dovrà essere valutata in un secondo test, da effettuarsi con un prototipo completo da lasciare una giornata intera sotto il sole, verificandone periodicamente il funzionamento.

6.4 Prestazioni di due antenne differenti e uso di VNC

In questo test si è voluto misurare la capacità delle due antenne più adatte ad essere montate sui nodi fissi, posizionati sulla cima delle torri faro. Per simulare al meglio il posizionamento finale le antenne sono state montate sulla cima di un palazzo alto circa 20 metri. Un primo nodo si trova al terzo piano dello stesso palazzo, collegato con un cavo Ethernet al secondo nodo, posto sul terrazzo. Da qui le antenne forniscono copertura al terzo nodo, posto su un automobile che si è mossa lungo il percorso segnato in figura 6.4. Il test si è svolto in due tempi, provando prima con un'antenna e poi con l'altra, per questo i grafici 6.5 e 6.6 non sono perfettamente sincronizzati tra le due antenne, infatti la brevità del percorso ha causato un'aumento dell'importanza delle condizioni di traffico locale.

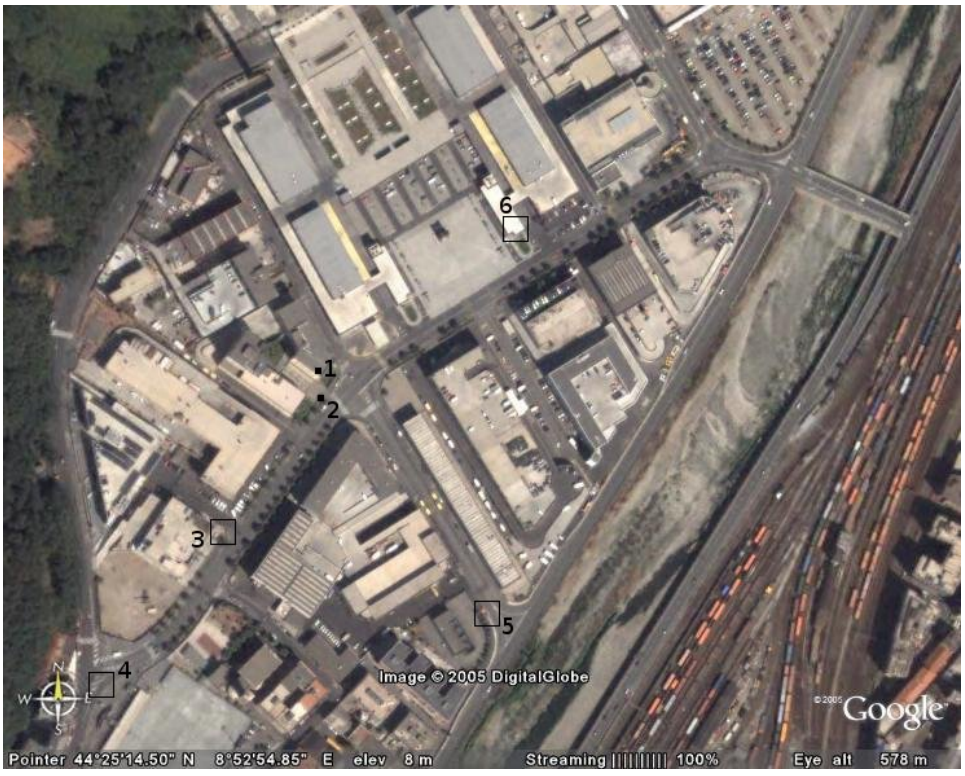

Figura 6.4: Visualizzazione del percorso compiuto per la misurazione della qualità del segnale. I punti numerati corrispondono alle posizioni segnate sui grafici 6.5 e 6.6. Nel punto 1 si trovano le antenne fisse. La distanza tra i punti 1,4 1,5 ed 1,6 è di circa 200m.

Nei punti indicati con i numeri ci si è fermati per registrare la qualità del

segnale e per usare VNC, collegandosi con una sessione di desktop remoto al calcolatore posto in ufficio. Questo per provare la fattibilità dell'uso di tale applicazione su una rete mesh.

6.4.1 Conclusioni - antenne

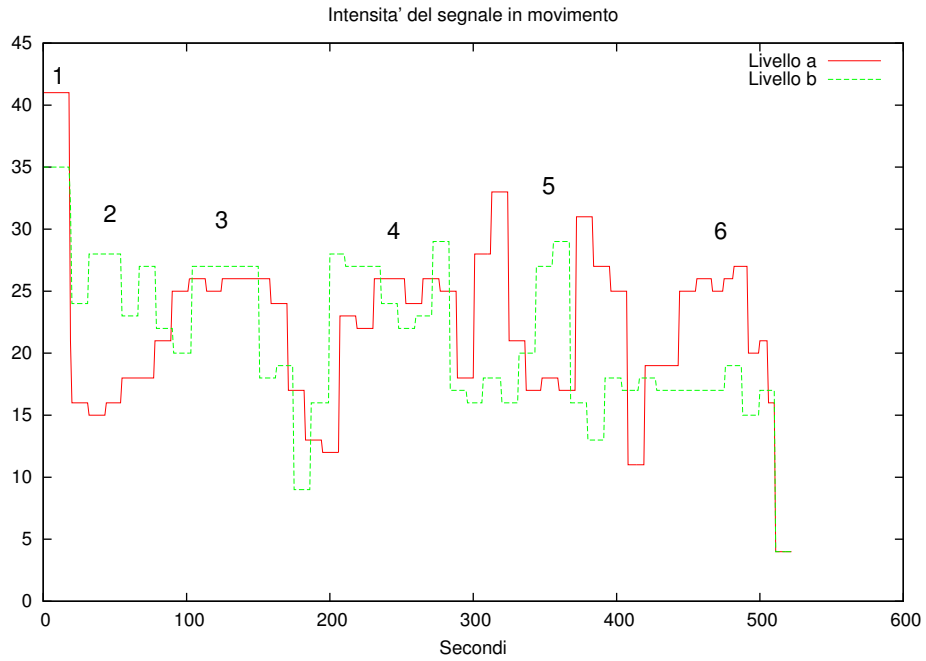

Figura 6.5: Intensità del segnale per le 2 antenne in prova. L'antenna a è la Cisco4.3.1, l'antenna b è la Huber+Sunher4.3.2.

La comparazione delle due antenne ha dimostrato che entrambe hanno caratteristiche tutto sommato equivalenti, consentendo di sceglierne una anche in basi ad altri parametri, come costo e facilità di fornitura. Le differenze riscontrate sono dovute alle diverse forme dei lobi di propagazione, che sono visibili nelle figure 4.3-4.6.
L'antenna Cisco offre un segnale più forte ad un ricevitore posto alla stessa altezza, questo è un vantaggio, dato che i nodi fissi saranno posti tutti alla stessa altezza, mentre i due nodi mobili avranno una copertura di almeno due antenne per la maggior parte del tempo.

Il buon rapporto segnale/rumore presente ad una distanza di 200m, permette di prevedere che il raggio massimo di comunicazione sia molto ampio e superiore alla distanza a cui è previsto il montaggio dei nodi fissi.

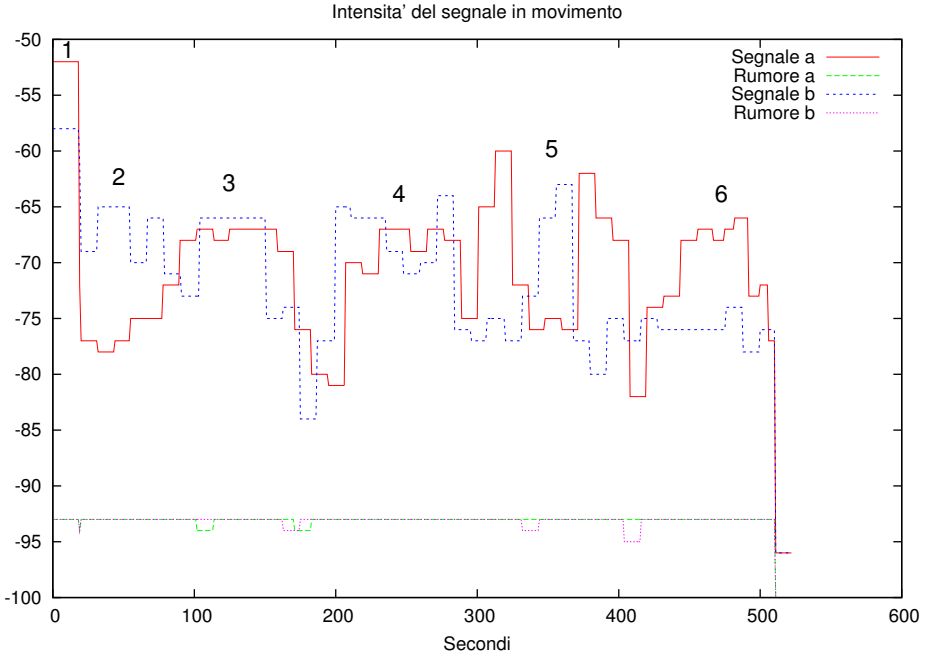

Figura 6.6: Grafico del segnale e del rumore ricevuto dall'antenna mobile, le antenne a e b sono le stesse di figura 6.5.

6.4.2 Conclusioni - VNC

L'uso di VNC è risultato fluido ed utilizzabile ad una risoluzione e profondità di colore superiori a quelli che sono necessari per l'applicazione di Nola. Un problema riscontrato è stato la facilità con cui il server chiude la connessione non appena si verifica una perdita di pacchetti, comportando la chiusura del client. Una semplice riconnessione è stata sempre sufficiente per ripristinare l'ambiente di lavoro.
Questo problema è risolvibile aumentando il tempo di timeout del server o modificando il client per ricollegarsi al primo segno di difficoltà.

Un altro test, molto simile a questo, servirà a verificare il funzionamento di VNC con due nodi intermedi, questo è molto importante in quanto esperimenti effettuati staticamente hanno dimostrato che la banda a disposizione dimezza ad ogni hop che viene attraversato.

6.4.3 Simulazione conclusiva con 3 hop e quattro nodi

Questa simulazione è servita a verificare se la stabilità del routing dinamico di `olsrd` è abbastanza buona da permettere un utilizzo continuo della rete e se la banda disponibile è abbastanza ampia da consentire due sessioni VNC contemporaneamente.

La prova ha consentito di determinare dei nuovi valori per l'isteresi e per l'HELLOValidityTime (vedi sezioni 3.2.1 e 5.5), più adatti ad un utilizzo di una rete a lenta mobilità[2] come quella che si è progettata per Nola.

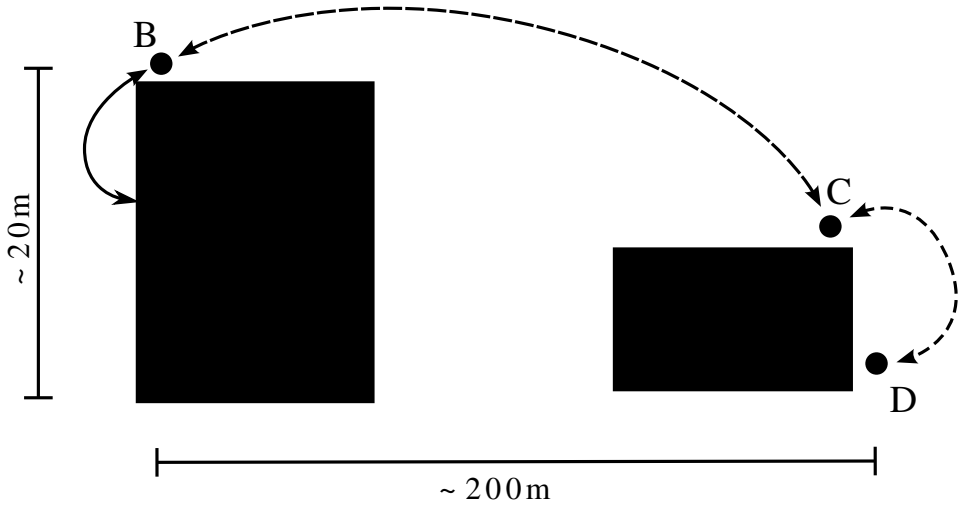

Figura 6.7: Disposizione dei nodi durante la simulazione. Il collegamento tra i nodi A e B è dato da un cavo Ethernet, mentre gli altri nodi usano 802.11. Tutto l'instradamento è generato da olsrd.

Il test ha anche mostrato quanto il segnale radio WiFi sia sensibile a fattori ambientali quali riflessioni su superifici metalliche. In particolare durante il test un TIR che manovrava nelle vicinanze ha causato la riflessione di abbastanza segnale da far credere ad `olsrd` di poter stabilire un nuovo cammino, causando gravi perdite di pacchetti.

La disposizione dei nodi è visibile in figura 6.7: A è un portatile con sistema operativo Windows 2000 ed un server VNC ed FTP. In esecuzione c'è un versione preliminare dell'interfaccia grafica che verrà usata a Nola. B e C sono dei router WRT54g, con software OpenWRT modificato per renderli capaci di stabilire un instradamento dinamico. Infine D è un secondo portatile, con Linux, un sistema di logging scritto apposta per questo test, un client ftp ed un visualizzatore VNC.

Il protocollo FTP è servito a misurare la banda disponibile tra il primo e l'ultimo nodo, trasferendo un file da 2 MB. Allo stesso tempo il traffico FTP è servito a simulare un secondo flusso VNC, per poterne verificare gli effetti sulla connessione VNC reale.

[2]Su 7 nodi, 5 saranno fissi e solo due in moto, al massimo a 30Km/h.

Conclusioni

Questa simulazione è stata di un'importanza cruciale. Ha permesso di capire l'importanza di trovare dei valori corretti per i parametri di `olsrd`, in modo da evitare un instradamento instabile che è causa di perdite di pacchetti e quindi a riduzioni improvvise della banda disponibile, con conseguenze disastrose per la connessione VNC che è, per molti aspetti, assimilabile ad un flusso video. Inoltre, essendo una prova sul campo, ha consentito di verificare il funzionamento di tutti gli apparati in condizioni realistiche.

6.5 Altre informazioni derivate dai test

Nello svolgere le prove descritte sopra si è giunti ad una serie di risultati molto interessanti, anche se non direttamente collegati ai test in svolgimento. Questi vengono elencati di seguito e spesso possono essere anche spunto di ulteriore indagine o lavoro di miglioramento.

6.5.1 Driver schede wireless per Linux

Il supporto delle schede senza fili su kernel Linux è ancora ad uno stadio molto primitivo rispetto a quello fornito da altri sistemi operativi. Questo non è solo causato dalla mancanza delle specifiche hardware necessarie a scrivere dei driver che supportino tutte le funzionalità fornite dal produttore, ma anche dalla mancanza un supporto software ad un livello più alto di quello hardware che sia in grado di astrarre le particolarità di ogni modello di scheda WiFi supportata.

Durante le varie misurazioni ci si è scontrati con il fatto che driver diversi utilizzano differenti unità di misura per riportare la qualità del segnale ed il livello di rumore. In alcuni casi le informazioni fornite non possono essere convertite in Decibel perché usano una scala completamente diversa (es. valori da 1 a 5). Inoltre i driver riportano quel tipo di informazioni in modo diverso, alcuni facendo una media dei valori letti dall'hardware su un certo periodo di tempo, causando problemi quando si vuole misurare la differenza di segnale che si ha tenendo, ad esempio, l'antenna in posizioni differenti. Per il driver hostap si è potuto misurare un ritardo di circa 10 secondi tra il valore visualizzato ed il valore reale, sia per la quantità di segnale che per il rumore. Il driver Cisco (modulo airo_cs), invece riporta i valori in modo continuo senza tentare di stabilizzarli, causando una continua variazione nel valore osservato, con dei picchi molto distanti dal valore medio.

Altro problema affrontato è che driver diversi assegnano un nome diverso all'interfaccia di rete senza fili, costringendo ad un continuo cambio delle impostazioni IP da wlan0 (driver hostap) a eth1 (driver Cisco).

Capitolo 7

Applicazione all'Interporto Campano (Nola)

L'apparato, completo di hardware e software come descritto nei capitoli precedenti, sarà impiegato nelle operazioni quotidiane dell'Interporto Campano presso Nola, vicino a Napoli. La messa in opera del sistema sarebbe dovuta avvenire durante lo svolgimento di questa tesi e farne parte integrante. Per quanto tutto il materiale fosse pronto per essere consegnato diverse settimane prima della data preventivata, alcune modifiche nel progetto dell'interporto hanno causato una serie di ritardi, facendo slittare la data di partenza indipendentemente dalla nostra volontà.

In questo capitolo, quindi, si descrive come verrà realizzato l'impianto di Nola e come funzionerà una volta che il sistema sarà portato a pieno regime. Infine si da anche qualche previsione su futuri impieghi della stessa tecnologia.

7.1 Planimetria e topologia

L'interporto di Nola è una grossa struttura in via di completamento dotata di un collegamento diretto sia alla rete ferroviaria che alla all'autostrada A30. Parte dell'interporto è dedicata allo scambio di container tra convogli ferroviari e trasporto su gomma. Proprio per questa area è stata richiesta la fornitura di due stacker e di una rete senza fili alla Fantuzzi Reggiane. Un'altra azienda si è occupata della realizzazione dell'applicativo software in grado di comunicare con il database (basato su AS/400) e di presentare all'operatore sul mezzo l'elenco delle operazioni da effettuare.

Il piazzale in cui i due mezzi operano è un rettangolo di circa 1,5 chilometri di lunghezza e 500 metri di larghezza. I container vengono impilati uno sopra l'altro a due o al massimo a tre alla volta per problemi di vento. Tutta la zona, infatti, ha la stessa valutazione di Trieste, con la possibilità di venti che superano i 100Km/h. A distanza di 100 metri l'uno dall'altro sono

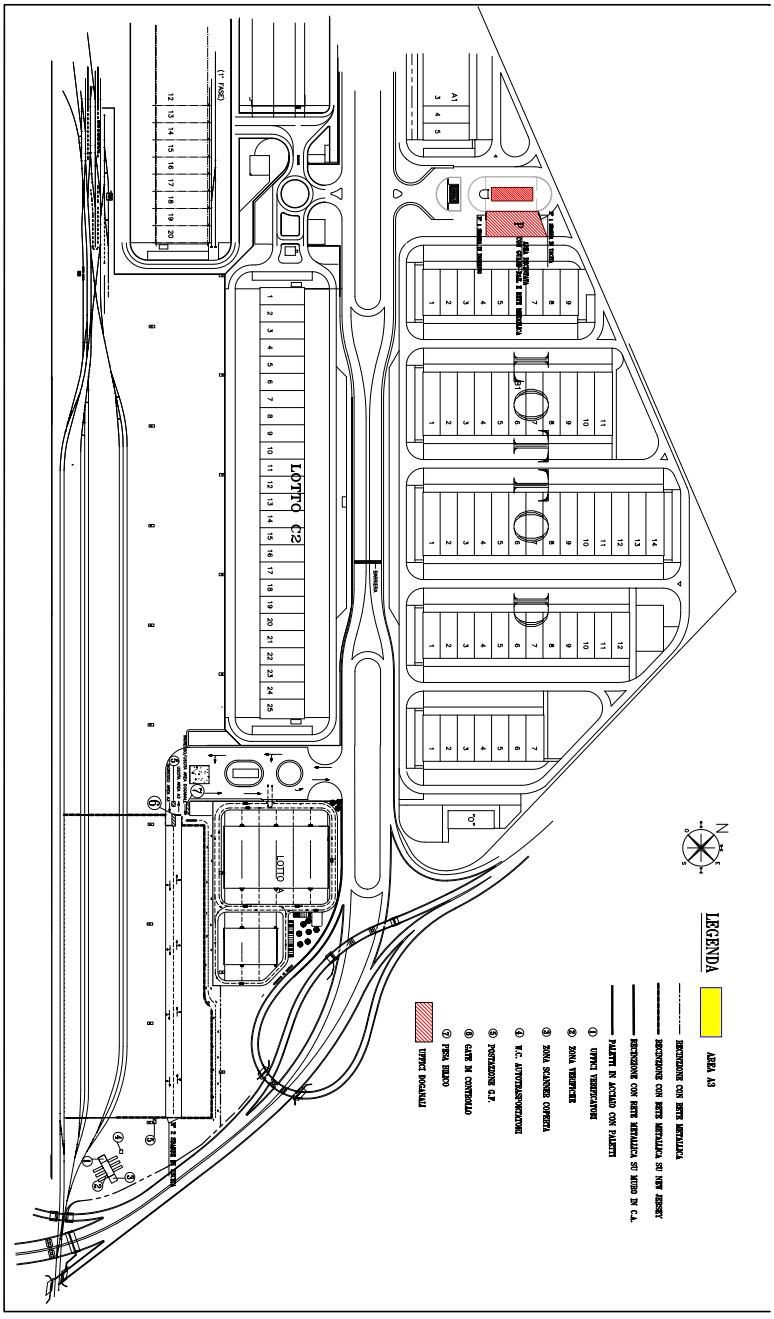

Figura 7.1: Veduta completa dell'interporto di Nola

installate delle torri faro alte 25 metri sulle quali è presente l'alimentazione elettrica a 220 volt.

7.1.1 Nodi fissi

Si prevede di installare tre nodi sulla cima delle torri faro ad una distanza di 400 metri l'uno dall'altro, con un nodo di riserva per coprire eventuali angoli bui. Per l'alimentazione sarà sfruttata quella già presente sul posto per le lampade. Per ognuno di questi nodi saranno installate due antenne in modalità *diversity* per coprire le due aree da una parte e dall'altra nella lunghezza del piazzale. Infine verrà installato un cavo ethernet che permetta il collegamento al nodo da terra, senza che sia necessario raggiungere la cima della torre faro per compiere operazioni di aggiornamento o diagnostica.

Figura 7.2: Veduta aerea dell'interporto con la disposizione dei nodi fissi montati sulle torri faro in corrispondenza dei quadrati rossi.

7.1.2 Nodi mobili

I nodi mobili saranno posizionati sui due stacker che opereranno all'interno del piazzale. Dato che su questi mezzi è presente la sola alimentazione a 24 volt di corrente continua è necessario che questi nodi dispongano di un piccolo trasformatore 24V/12V anziché di quello 220V/12V montato di serie. Inoltre questi nodi saranno montati in posizione coperta, probabilmente al

di sotto del braccio di sollevamento, in modo che siano protetti al massimo dagli sbalzi di temperatura. Su questi nodi verrà montata una sola antenna, in cima ad un palo aggiunto appositamente in coda allo stacker, ad un'altezza di 3-4 metri da terra.

Questi MeshAP forniranno l'accesso alla rete senza fili in maniera totalmente trasparente al Display montato nella cabina dell'operatore attraverso un cavo Ethernet. Sul Display verranno visualizzate diverse schermate che presentano informazioni sullo stato della macchina e su eventuali allarmi o altri eventi. In più sarà presente una schermata che visualizza l'elenco delle operazioni che l'operatore dove compiere. Per semplificare al massimo l'interfaccia di comunicazione tra il Display e l'applicazione che lavora sui database dell'amministrazione dell'interporto questa schermata sarà realizzata esportando attraverso VNC il desktop di un computer fisso mantenuto in ufficio. Questo computer, quindi, manterrà una risoluzione di 640x480 a 16 bit di colore ed un'unica applicazione a pieno schermo.

7.2 Montaggio

Il montaggio finale avverrà in due momenti diversi. I MeshAP mobili saranno installati sugli stacker negli stabilimenti di Fantuzzi Reggiane prima della spedizione finale, in modo da ridurre al massimo il lavoro da svolgere a Nola. I nodi fissi dovranno essere installati per forza di cose sul posto da parte del personale addetto alla manutenzione delle torri faro.

7.3 Impieghi futuri

Il MeshAP è un prodotto molto innovativo nel suo campo e si prevede di utilizzarlo in molte delle future installazioni di reti senza fili che saranno richieste a Fantuzzi Reggiane. L'impiego di reti informatiche basate su WiFi (802.11) sta prendendo sempre più piede nel campo delle comunicazioni senza fili grazie al basso costo ed all'assenza di concessioni governative per l'uso di canali radio riservati.

Il prossimo passo di sviluppo del MeshAP prevede una riduzione delle dimensioni e del peso per poterlo proporre come soluzione ad un insieme di problemi più ampio, per arrivare ad un dispositivo che possa essere alimentato a batteria per un tempo ragionevole e sostituire completamente anche le radio per la comunicazione voce tramite il protocollo Voice over IP.

Figura 7.3: Montaggio del MeshAp sullo stacker: la lettera A indica la posizione della hardware, mentre in B sarà montata l'antenna sulla cima di un palo.

Bibliografia

[1] W. Alliance. Wi-fi protected access: Strong, standards-based, interoperable security for todays wi-fi networks, 2003. URL `http://www.wifialliance.com/OpenSection/pdf/Whitepaper_Wi-Fi_Security4-29-03.pdf`.

[2] Andreas Tønnesen. Implementing and extending the Optimized Link State Routing protocol. Master's thesis, UniK - University Graduate Center, 2004. URL `http://www.olsr.org/docs/report.pdf`.

[3] ANSI. *Information processing systems: local area networks — Part 11. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. American National Standards Institute, 1430 Broadway, New York, NY 10018, USA, ANSI/IEEE Std 802.11-1999 edition. International standard ISO/IEC 8802-11, IEEE edition, 2003. ISBN 0-7381-1658-0.

[4] ANSI. *Information processing systems: local area networks — Part 3. Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*. American National Standards Institute, 1430 Broadway, New York, NY 10018, USA, ANSI/IEEE Std 802.3-1990 edition. International standard ISO/IEC 8802-3, IEEE product number: SH13482 edition, 1992. ISBN 1-55937-049-1.

[5] E. H. Callaway. *Wireless Sensor Networks: Architectures and Protocols*. Auerbach Publications, 2003. ISBN 0-84931-823-8.

[6] Christian Tschudin and Richard Gold. LUNAR - Lightweight Underlay Network Ad-hoc Routing. Technical report, Department of Computer Systems, Upsala University, Apr. 2002. URL `http://www.it.uu.se/research/reports/2003-021/2003-021-nc.pdf`.

[7] Christian Tschudin, Richard Gold, Olof Rensfelt, and Oskar Wibling. In proceedings of next generation teletraffic and wired/wireless advanced networking. In *LUNAR - A Lightweight Underlay Network Ad-hoc Routing Protocol and Implementation*, 2004. URL `http://cn.cs.unibas.ch/pub/doc/2004-new2an-lunar.pdf`.

[8] W. C. Craig. Zigbee: wireless control that simply works, 2004. URL `http://www.zigbee.org`.

[9] Daniel Aguayo, John Bicket, Sanjit Biswas, Douglas S. J. De Couto, and Robert Morris. *MIT Roofnet Implementation*. MIT, 2003. URL `http://pdos.csail.mit.edu/roofnet/doku.php?id=design`.

[10] David B. Johnson, David A. Maltz, and Josh Broch. *Ad Hoc Networking*. Addison-Wesley, 2001.

[11] G. Dodero, V. Gianuzzi, and M. Ancona. Ramses: a mobile computing system for field archaeology, 1999. URL `http://www.disi.unige.it/person/DoderoG/ramses/papers/pubramses.htm`.

[12] Erik Nordström, Björn Wiberg, and Henrik Lundgren. AODV-UU, 2002. URL `http://core.it.uu.se/AdHoc/AodvUUImpl`.

[13] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. *Lecture Notes in Computer Science*, 2259:1–24, 2001. URL `citeseer.ist.psu.edu/fluhrer01weaknesses.html`.

[14] *Design considerations for distributed microsensor systems*, 1999. IEEE. URL `citeseer.ist.psu.edu/chandrakasan99design.html`.

[15] International Organization for Standardization. *ISO/IEC 7498-1:1994: Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*. International Organization for Standardization, Geneva, Switzerland, 1994. URL `http://www.iso.org/cate/d20269.html`.

[16] A. Mehta. Wireless optical communications, a fast and cost-effective means to bring telecommunications to villages, 2001. URL `http://indataportal.com/optical/WOC.htm`.

[17] B. Miller. RFC 1097: Telnet subliminal-message option, Apr. 1989. URL `http://www.rfc.net/rfc1097.txt`. Status: UNKNOWN.

[18] J. Postel. RFC 791: Internet Protocol, Sept. 1981. URL `http://www.rfc.net/rfc791.txt`. Obsoletes RFC0760. See also STD0005. Status: STANDARD.

[19] J. Postel. RFC 793: Transmission control protocol, Sept. 1981. URL `http://www.rfc.net/rfc793.txt`. See also STD0007. Status: STANDARD.

[20] J. Postel and J. K. Reynolds. RFC 854: Telnet Protocol specification, May 1983. URL `http://www.rfc.net/rfc854.html`. Obsoletes RFC0764, NIC18639. See also STD0008. Status: STANDARD.

[21] R. M. Project. Dsr implementation for freebsd, 2000. URL `http://www.monarch.cs.cmu.edu/dsr-impl.html`.

[22] J. M. SSH Communications Security Corp. HostAP driver and related software, 2001. URL `http://hostap.epitest.fi/`.

[23] T. Clausen and P. Jacquet. RFC 3626 - Optimized Link State Routing Protocol (OLSR). Technical report, Network Working Group, Project Hypercom INRIA, 2003. URL `http://www.rfc.net/rfc3626.html`.

[24] L. Viennot. Complexity results on election of multipoint relays in wireless networks. Technical Report RR-3584, INRIA, 1998. URL `http://citeseer.ist.psu.edu/viennot98complexity.html`.

[25] William H. Mott IV and Robert B. Sheldon. *Laser Satellite Communication, The Third Generation*. Quorum Books, Westport, Conn. 2000, 2000. ISBN 1-56720-329-9. URL `http://bex.nsstc.uah.edu/RbS/greenwood.html`.

[26] Wireless Communications Technologies Group NIST. Kernel aodv, 2001. URL `http://w3.antd.nist.gov/wctg/aodv_kernel/`.

[27] Xia Jiang and Tracy Camp. A review of geocasting protocols for a mobile ad hoc network, 2002.