Capitolo 1

TEST E VALUTAZIONI SPERIMENTALI

Dopo aver illustrato il funzionamento e l'implementazione del TED, passiamo ora ad analizzare i dati relativi alla trasmissione di pacchetti tramite la nostra applicazione. Lo scopo dei test effettuati è quello di analizzare la QoS del segnale e come può variare in diversi scenari. Questo ci è utile in quanto, in base ai risultati ottenuti, si può decidere se cambiare NIC per l'invio di determinati pacchetti.

Andremo ora a mostrare quali test sono stati effettuati. In particolare mostreremo quali dispositivi sono stati utilizzati per fare le prove, i parametri di valutazione ed i risultati ottenuti.
Lo scopo di questi test è di andare ad osservare la QoS del segnale e come può variare in scenari diversi.

1.1 Dispositivi

Per effettuare delle prove sperimentali, sono stati utilizzati diversi dispositivi. In particolare tre per simulare il client (o nodo mobile) ed altri per creare traffico nella rete, in modo da impegnare il canale per avere condizioni simili ad un tipico scenario di utilizzo. Per ogni dispositivo è interessante mo-

strare per cosa è stato utilizzato. Nel caso di un nodo della rete, mostreremo anche il sistema operativo, il kernel, il processore e scheda di rete.

ZyXEL NBG4615 v2 Access point utilizzato durante i test. È stata impostata la modalità Wi-Fi 802.11b/g/n.

LB-LINK BL-WN151 Adattatore Wireless USB. Velocità fino a 150Mb/s, supporta 802.11b/g/n.

NETGEAR WG111 Adattatore Wireless USB. Velocità fino a 54Mb/s, supporta 802.11b/g.

HP Pavilion dv6 Entertainment PC Questo notebook è stato utilizzato come client. Abbiamo montato un kernel versione 4.0.1, modificato tramite la procedura illustrata precedentemente. Le specifiche tecniche sono:

- Kernel: Linux versione 4.0.1 modificata
- Processore: Intel Core i5 CPU M 430 @ 2.27GHz x 4
- Sistema operativo: Ubuntu 14.04 LTS 64-bit
- Scheda di rete: Broadcom BCM43225 802.11b/g/n

DELL Latitude E6400 Questo notebook è stato utilizzato come client. Abbiamo montato un kernel versione 4.0.1, modificato tramite la procedura illustrata precedentemente. Le specifiche tecniche sono:

- Kernel: Linux versione 4.0.1 modificata
- Processore: Intel Core 2 Duo
- Sistema operativo: Ubuntu 14.04 LTS 32-bit
- Scheda di rete: Intel Corporation WiFi Link 5100 [802.11a/g/n (1x2)]

Raspberry Pi Model 2 Abbiamo utilizzato questo raspberry per creare traffico sulla rete. Le specifiche tecniche sono:

- Kernel: Linux versione 3.18.0-20-rpi2
- Processore: 900MHz quad-core ARM Cortex-A7
- Sistema operativo: Raspbian

UDOO Quad Abbiamo utilizzato questo raspberry per creare traffico sulla rete. Le specifiche tecniche sono:

- Kernel: Kernel Linux 3.0.35
- Processore: Freescale i.MX 6 ARM Cortex-A9 CPU Dual/Quad core 1GHz
- Sistema operativo: UDOObuntu

Raspberry Pi Model B Abbiamo utilizzato questo raspberry per creare traffico sulla rete. Le specifiche tecniche sono:

- Kernel: Kernel Linux 3.18.7+
- Processore: 700 MHz single-core ARM1176JZF-S
- Sistema operativo: Raspbian Wheezy

1.2 Parametri di valutazione

Per poter analizzare i test in modo ottimale e per avere dei dati su cui lavorare abbiamo deciso di controllare alcuni parametri.
I parametri che ci interessano maggiormente sono:

- **Id**: l'Id del pacchetto inviato.
- **ACK**: se è stato ricevuto un ACK o un NACK da parte dell'AP.

- **Data**: è dato dalla data e dall'orario di invio del pacchetto.
- **Tempo**: è il tempo in millisecondi tra l'invio del pacchetto e la ricezione della notifica da parte dell'access point.
- **Retry count**: è il numero di tentativi di invio di un determinato pacchetto
- **Versione IP**: IPv4 o IPv6
- **Configurazione**: è la configurazione dei dispositivi utilizzati durante l'esperimento.
- **Wait**: indica se la recv è bloccante.

Abbiamo scelto questi parametri perché ci permettono di poter giudicare in maniera chiara l'andamento dei pacchetti e la situazione della rete. In particolare sono molto significativi il tempo, l'ACK ed il retry count. Grazie a questi dati si può analizzare in modo dettagliato la situazione di ogni singolo pacchetto. L'applicazione può leggere l'ACK e successivamente decidere di rimandare il pacchetto in base ai millisecondi passati prima di ricevere la notifica. Il numero di retry count risulta rilevante per confrontare differenti situazioni di traffico, oppure per notare cosa succede in caso di trasmissione in movimento. Gli altri parametri che abbiamo deciso di utilizzare hanno un valore più trascurabile per un singolo pacchetto, ma possono diventare eloquenti per analizzare i dati a posteriori. In particolare si potrebbe notare in base all'orario se c'è un evidente rallentamento della trasmissione. Ad esempio si potrebbe notare come in una zona industriale la QoS migliori durante la sera/notte. Un altro dato che può essere utilizzato per esaminare i dati raccolti è la versione IP, si può controllare se c'è un differenza notevole tra IPv6 e IPv4 a parità di condizioni. Le configurazioni, invece, riguardano i dispositivi utilizzati durante un test e lo scenario applicativo. Andremo a mostrare quali configurazioni sono state provate in modo più dettagliato successivamente. Per quanto riguarda la wait abbiamo deciso di fare sia una

recv bloccante che una non bloccante. Abbiamo fatto dei test con entrambe e abbiamo analizzato le differenze, che andremo a descrivere più avanti.

Si potrebbero utilizzare anche altre informazioni (ad esempio la bitrate) per analizzare meglio i risultati, che saranno approfondite negli sviluppi futuri.

1.3 Configurazioni

Per ottenere dei risultati che potessero rispecchiare un reale utilizzo da parte di un nodo mobile abbiamo creato diverse configurazioni di dispositivi. In particolare abbiamo combinato l'utilizzo di uno o più dispositivi contemporaneamente.

1.4 Risultati

Andiamo ora ad analizzare i risultati ottenuti.

Tabella 1.1: Prima tabella

	Da solo	Un host	Due hosts	Tre hosts
prima	ye	ery	rey	456
seconda	ye	ye	ey	ru
terza	ewy	ye	yer	54
quarta	ewy	ye	yer	54
Media	ewy	ye	yer	54