Università degli studi di Napoli Parthenope
Corso di laurea in Informatica
Dipartimento di Scienze e Tecnologie

Bitcoin

Progetto d'esame Reti dei Calcolatori

Vittorio FONES 0124001384

A. A. 20018-2019

Contents

1 Descrizione del progetto **1**

2 Descrizione e schemi dell'architettura **2**

3 Descrizione e schemi del protocollo applicazione **3**
3.1 Primo aggancio, scarico blockchain e flooding 3
3.2 Aggancio wallet. 4
3.3 Invio transazione 4

4 Dettagli implementativi **5**
4.1 Dettagli del nodo 5
4.2 Dettagli del wallet 5

5 Manuale utente **6**
5.1 Instruzioni per la compilazione 6
5.2 Instruzioni per l'esecuzione 7

List of Figures

2.1 Architettura di sistema. 2

3.1 Aggancio nodo. 3
3.2 Aggancio wallet. 4
3.3 Invio transazione. 4

Chapter 1

Descrizione del progetto

Con il seguente elaborato si intende illustrare il lavoro svolto per la realizzazione del progetto Bitcoin, il cui scopo è quello di creare una rete peer to peer usata per la gestione di una blockchain e l'utilizzo della stessa. Il progetto è stato sviluppato nel linguaggio C, attenendosi allo standard Posix come illustrato durante il corso e sul libro GAPIL: per l'appunto il software gira anche sotto piattaforma Mac OS, oltre che su distribuzioni Linux quali Debian e Arch Linux.

Di seguito viene allegata l'introduzione della traccia sviluppata:

> Si vuole realizzare un sistema per la gestione di una criptovaluta basato su una rete P2P. Il sistema è basato sulla gestione di una blockchain, ovvero una sequenza di blocchi in cui ogni blocco contiene una transazione. Il sistema si compone di due tipi di nodi: NodiN e NodiW. I NodiN creano la rete P2P e gestiscono la blockchain. Inoltre stampano la blockchain ogni volta che viene aggiunto un blocco: (blocco 1)-> (blocco 2)-> (blocco 3)-> (blocco 4). I NodiW gestiscono i wallet (portafogli virtuali) che consentono di inviare e ricevere pagamenti. Ad ogni nuovo pagamento inviato o ricevuto il nodo stampa la transazione ed il totale del portafogli.

Per evitare qualunque tipo di problema d'ora in avanti i nodi_w verranno chiamati semplicemente wallet.

Chapter 2

Descrizione e schemi dell'architettura

Per creare la rete si è deciso di non adottare entità esterne, che seppure fossero venute in aiuto per l'automatizzazione della stessa rete, avrebbe reso il concetto di cryptovaluta meno reale, andando a rendere semi-centralizzata o peggio la rete peer to peer. Nonostante ciò si è cercato di non andare oltre la traccia, per cui tutti i casi reali quali il mining, la cifratura, creazione di hash e validificazione dei blocchi e altro, non sono stati presi in cosiderazione.
La rete peer to peer è composta dai nodi_n che nel loro insieme formano una rete decentralizzata. Ogni nodo ha un pannello di controllo in cui è possibile interfacciarsi con altri nodi e vedere lo stato delle proprie attività, tra cui la blockchain al momento. A questi verranno poi connessi i nodi_n: una volta collegato ad un nodo_n, un wallet potrà effettuare movimenti di BitCoin, sarà poi il nodo_n che provvederà a creare il blocco e diffonderlo.

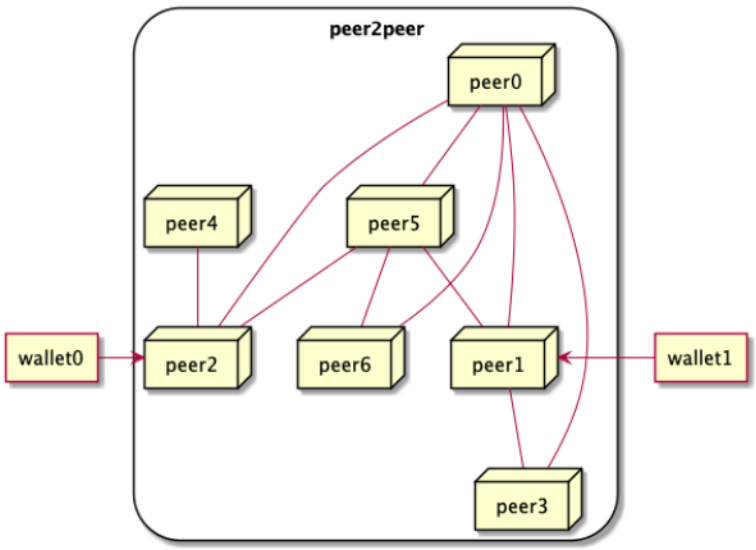


Figure 2.1: Architettura di sistema.

Chapter 3

Descrizione e schemi del protocollo applicazione

Di seguito sono riportati gli schemi dei principali protocolli applicazione.

3.1 Primo aggancio, scarico blockchain e flooding

Per capire meglio lo scenario mostrato in figura, definiamo un Nodo_0 e un Nodo_N. Al Nodo_N viene fatta richiesta di aggancio da parte del Nodo_0, connesso a sua volta con più nodi e un wallet. Inviando un tipo definito da programma, **request_t**, il Nodo_0 richiede al Nodo_N di entrare a far parte della rete. Avvenuta la connessione bisognerà sincronizzare la blockchain, inviandosi i blocchi mancanti, per cui i nodi si scambieranno i size delle rispettive blockchain **locali**, in tal modo i due sapranno chi dovrà ricevere e chi dovrà inviare i blocchi. Ipotizzando che il Nodo_N abbia la blockchain più aggiornata, ad ogni ricezione di un nuovo blocco, il Nodo_0 lo diffonderà ai nodi X connessi e nel caso, al wallet.

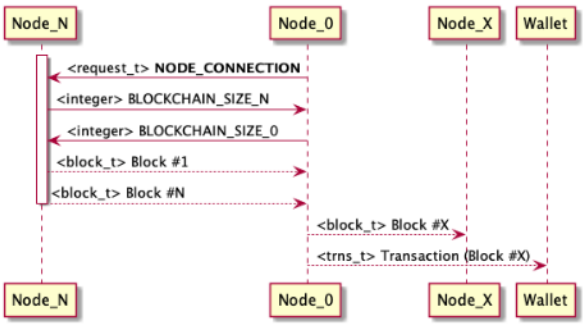


Figure 3.1: Aggancio nodo.

3.2 Aggancio wallet.

La procedura dell'aggancio del wallet è più semplice in quanto il wallet invierà al nodo scelto una richiesta del tipo **request_t** e in caso di disponibilità il nodo invierà un intero a conferma di avvenuta connessione.

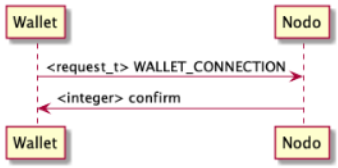


Figure 3.2: Aggancio wallet.

3.3 Invio transazione

Dopo aver opportunamente creato una transazione, il wallet provvederà a mandarla al peer a cui è connesso. A seguito di una conferma, il nodo, aggiungerà alla blockchain il blocco appena creato contentente la transazione e provvederà ad inviare a tutti i peer della rete a cui è connesso il blocco.

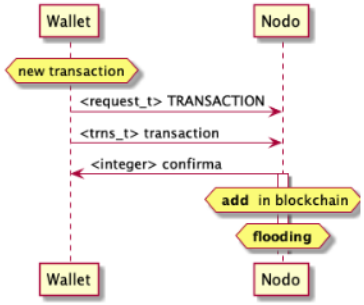


Figure 3.3: Invio transazione.

Chapter 4

Dettagli implementativi

Per la realizzazione di ogni applicativo si è optato per protocolli TCP/IP e Socket di Berkley.

4.1 Dettagli del nodo

I nodi poiché devono spedire i vari blocchi, controllare l'arrivo di nuovi, servire le richieste di connessione e di transazione dei wallet, sono stati implementati secondo uno schema **I/O Multiplexing** che usa i thread posix della libreria pthread. In particolar modo ad ogni nuova connessione verrà creato un thread che userà a sua volta uno schema I/O Multiplexing. In tal modo un nodo che vuole connettersi ad un peer della rete avrà un thread a gestirgli la connessione, mentre ne avrà N in relazione a quante connessioni avrà ricevuto.
Per ovviare ai relativi problemi dovuti all'accesso concorrente alla blockchain si è scelto di sincronizzare i thread tramite **lock rw** disponibili nella libreria PTHREAD.H.

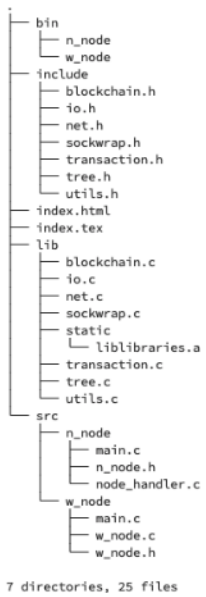
4.2 Dettagli del wallet

I nodi wallet dovendo gestire input da tastiera e l'arrivo dalla rete di richieste dal peer connesso, si è deciso di implementarlo con uno schema I/O Multiplexing come nel caso precedente con la differenza che saranno processi del tutto iterativi, bloccandosi nel caso ci sia attesa dovuta alla connessione.

Chapter 5

Manuale utente

Il progetto è stato strutturato nel seguente modo: nella cartella src ci saranno i file .c e .h dei due processi sotto opportune cartelle. Nella cartella include vi sono gli header file delle librerie, che si trovano invece nella cartella src. Nella cartella lib viene creata anche la cartella static contentente la libreria statica per il corrispettivo sistema operativo.



5.1 Instruzioni per la compilazione

Per la creazione dei file eseguibili si è scelto di affidarsi al software CMake che genera automaticamente il make file. Per cui basterà creare la cartella build e spostarvici all'interno:

```
$ mkdir build && cd build
```

Lanciare cmake con il comando:

```
$ cmake ..
```

A seguito della creazione, lanciare make:

```
$ make
```

A questo punto nella directory bin saranno presenti i file eseguibili.

5.2 Instruzioni per l'esecuzione

Per avviare i programmi basterà spostarsi nella cartella bin, e avviare N processi n_node, e a seguito i processi w_node. Poiché i processi w_node si connetteranno alla rete peer to peer tramite opzione da riga di comando andranno eseguiti prima i processi n_node. Poiché si presuppone che un ipotetico firewall sia abilitato e che i processi possano essere raggiunti tramite il loro indirizzo pubblico, l'unica opzione attivabile è quella per esporre la porta.

```
$ n_node -p 5555
```

I wallet invece possono anche selezionare un indirizzo IP oltre che la porta a cui connettersi. Per motivi pratici è stata aggiunta come opzione facoltativa, per cui:

```
$ w_node -p 5555
```